UNIVERSITY OF LIVERPOOL

REPRESENTATION OF CASE LAW FOR ARGUMENTATIVE REASONING

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of Doctor in Philosophy by

Latifa Mohammed Al-Abdulkarim

18/04/2017

Contents

Illustrations

List of Figures

List of Tables

Dedication

To my beloved parents for their endless support.

Abstract

Modelling argumentation based on legal cases has been a central topic of AI and Law since its very beginnings. The current established view is that facts must be determined on the basis of evidence. Next, these facts must be used to ascribe legally significant predicates (factors and issues) to the case, on the basis of which the outcome can be established.

This thesis aims to provide a method to encapsulate the knowledge of bodies of case law from various legal domains using a recent development in AI knowledge representation, Abstract Dialectical Frameworks (ADFs), as the central feature of the design method. Three legal domains in the US Courts are used throughout the thesis: The domain of the Automobile Exception to the Fourth Amendment, which has been freshly analysed in terms of factors in this thesis; the US Trade Secrets domain analysed from well-known legal case-based reasoning systems (CATO and IBP); and the *Wild Animals* domain analysed extensively in AI and Law.

In this work, ADFs play a role akin to that of Entity-Relationship models in the design of database systems to design and implement programs intended to decide cases, described as sets of factors, according to a theory of a particular domain based on a set of precedent cases relating to that domain. The ADFs in this thesis are instantiated from different starting points: factor-based representation of oral dialogues and factor-based analysis of legal opinions.

A *legal dialogue representation model* is defined for the US Supreme Court Oral Hearing dialogues. The role of these hearings is to identify the components that can form the basis of an argument that will resolve the case. Dialogue moves used by participants have been identified as the dialogue proceeds to assert and modify argument components in term of issues, factors and facts, and to produce what are called Argument Component Trees (ACTs) for each participant in the dialogue, showing how these components relate to one another. The resulting trees can be then merged and used as input to decide the accepted components using an ADF. The model is illustrated using two landmark case studies in the Automobile Exception domain: *Carney v. California* and *US v. Chadwick*.

A *legal justification model* is defined to capture knowledge in a legal domain and to provide justification and transparency of legal decisions. First, a legal domain ADF is instantiated from the factor hierarchy of CATO and IBP, then the method is applied to the other two legal domains. In each domain, the cases are expressed in terms of factors organised into an ADF, from which an executable program can be implemented in a straightforward way by taking advantage of the closeness of the acceptance conditions of the ADF to components of an executable program.

The proposed method is evaluated to test the ease of implementation, the efficacy of the resulting program, the ease of refinement, transparency of the reasoning and transferability across legal domains. This evaluation suggests ways of improving the decision by incorporating the case facts, and considering justification and reasoning using portions of precedents.

The final result is ANGELIC (ADF for kNowledGe Encapsulation of Legal Information from Cases), a method for producing programs that decide the cases with a high degree of accuracy in multiple domains.

KEY WORDS: Legal Reasoning, Legal Dialogue, ADF, Case-Based Reasoning, Factors, Facts.

Acknowledgements

This thesis is the result of a long period of study at University of Liverpool, beginning as a Software Engineering MSc student online. I became a member of the Argumentation in AI research group, where I found my research passion in reasoning, analysis and decision making. This led me to explore the challenges of a multi-disciplinary research project, in the critical legal domain. This thesis presents the lessons learned from developing ANGELIC, a legal knowledge encapsulation method in the domain of Argumentation in AI and Law. A number of remarkable individuals were behind ANGELIC, whom I also wish to acknowledge and thank.

I am deeply grateful to my supervisors, the real supportive hands in this research. They helped me face the challenges and learn my strengths.

I still remember the first time I met **Katie Atkinson**, when she introduced me to Dung's Argumentation Frameworks in order to help two agents to have lunch at a restaurant that satisfied both their values. This became a turning point, not only in my research but in my whole life, since I began to apply agent strategies to everyday life. Katie has been a role model; she is an outstanding researcher, an excellent supervisor, and a great professor and head of department. I appreciate all the opportunities Katie has given to me to be a better academic researcher. Above all, her sense of humour was always there during all my difficult moments.

I am very honoured and fortunate to have been **Trevor Bench-Capon's** last PhD student. Trevor believed in me, and unhesitatingly supported my decision to shift the direction of my research to AI and Law after reading a legal case study in ASPIC+ (Figure 2.12 in this thesis). Grasping the required background for this research would not have been possible without Trevor's expert help and guidance. Throughout the rough road to finishing this thesis, Trevor has taught me that "Best is the enemy of Good" to help me overcome the difficulties and submit the thesis on time. Working with Katie and Trevor, I have learned many lessons, both consciously and unconsciously, and our collaboration has been one of joy and enthusiasm.

Behind all my achievements in life are my **angelic parents**, who raised me with love. Truly, I cannot thank you enough. I would not have reached this milestone without your support in all my pursuits. Your prayers surrounded my days, giving me the power to stand up and face all the challenges. Eleven years ago, after the first days in my academic job, I remember my father welcoming me, saying "Welcome back my princess Dr. Latifa." That was the last words I have heard from him, but was powerful enough to help me achieve my dream. In our daily skype calls, I told him about my progress; his eyes and smiles, along with my mother's voice, were my source of love and energy after a long working day. My mother's words taught me that the best is yet to come, and everything will end perfectly one day. It has been emotionally hard to be away from my parents all this time; I hope to be the daughter they are always proud of.

The warmest gratitude goes also to my extended **family**: sisters; brothers; sisters in law; nephews and nieces for all their emotional support, for encouraging me to take this important step in my life by handling all my needs, and visiting me frequently. A special thanks goes to my brother and his magic word “Focus,” which he used often to help me get back into research mode. I know I am blessed in having such a family.

I would not be who I am today without my soulmate **Bedour AlRayes** (Gamary) beside me: she has never stopped supporting me, holding my hand through all the tough times in my life. Thanks for being my family in the UK, my host in London, a shoulder to cry on and a good listener and adviser. Thanks a lot for all the research and academic tips that helped me avoid many problems. What Bedour has done for me means more than words can say.

Deep thanks go to my lifelong friends Lina and Nora for their encouraging words and wishes, calls and surprise gifts. You were always there, believing in me and understanding all my weird thoughts. You are true friends and amazing people.

I am also especially grateful for my Liverpudlian PhDs group; the three sisters Amal, Naseem and Nada (Etban Liverpool) who were with me in grad school. The group has been a source of friendship as well as good advice and collaboration. Knowing you on this journey was a delight; you have made Liverpool a place of memories with all the fun stories, along with the nightmares and research panics.

Prior the end of my PhD journey, I met Hanaa Namankani (my viva supporter) whom I would like to thank for her encouraging and calming words during my preparation for the big day (the viva).

I would also like to thank my office mates in room 2.11 and 2.13 since September 2012. A special thanks goes to Maduka Attamah for his advice and support on writing up, and to Matola for kindness when I was feeling down during the writing-up year. Thanks also to other friends whom I have had the pleasure to work alongside: Agneaue Belanyek for his inspirational discussions, and Subheyiah AlSalhey and Esraa Alshdaifat for their lovely friendships. Special thanks go also to the amazing people I worked with at the Saudi Students Scientific Society in the UK (SSSSUK): the skills I obtained from the society have immensely enriched my personal and professional experience.

A special acknowledgement goes also to my advisors throughout my PhD years: Paul Dunne, Davide Grossi and Floriana Grasso for their annual feedback, which helped me clarify the direction of the research. I would also like to thank the reviewers and audience in the Argumentation and AI and law conferences (ICAIL, Jurix and COMMA) and the AI and Law journal, for their helpful comments and insightful questions. A special thanks to Gerhard Brewka for his talk at Jurix 2013 that inspired the research in this thesis. I also recognize the support and help from the “Diamond system” team, especially Stefan Ellmauthaler, during the evaluation of ANGELIC.

My thank goes also to the lovely administration staff at the department of computer science, especially the PGR director, Clare Dixon, and the PGR team of Helen Mattocks and Hanna Fosh, who kept me organized and were always ready to help.

I also would like to thank my sponsor, King Saud University, and the Saudi Cultural Bureau for their financial support. I gratefully acknowledge the Saudi cultural attaché Dr. Faisal Abalkhail for being extremely supportive, and for all the opportunities he gave me to be the representative of Saudi students in the UK at many events.

Lastly, I would like to thank a lot of people who entered my life during my research journey and left an impact without even knowing it: a morning smile or wishes that changed my day; advice or a question that changed my thoughts; a complement for my research that helped me generate more creative ideas; and new lessons in life that helped me become more confident, stronger, more mature and a better decision maker. Thank you!

Chapter 1

Introduction

"Justice? – You get justice in the next world. In this one you have the law."
William Gaddis, A Frolic of His Own

1.1 Overview on Artificial Intelligence and Law

Understanding the human reasoning process used in conducting arguments, resolving conflicts, and reaching a decision is an interdisciplinary study. Modelling argumentation has had a great impact on the development of theories and applications in Artificial Intelligence (AI), especially in critical domains that involve richness of reasoning such as Law. Law is at the heart of every society, and is vitally important to people's lives in all aspects, through applying justice, addressing human rights, and promoting social values.

AI and Law have crossed paths [31, 110] for over 25 years. Creating and applying Law involves information processing, reasoning, and decision making, which correspond to the techniques of information retrieval and extraction, knowledge representation and reasoning, natural language processing, machine learning and data mining. This makes Law a natural application area and a rich test-bed for research in AI. In addition to these shared interests, AI and Law also have a shared subject [31, p.307]: In law, rules have exceptions, reasons are weighed, and principles provide guidance. Similarly in AI, reasoning is uncertain, knowledge is context-dependent, and behaviour is adaptive.

A particular concern of AI and Law is legal reasoning. Law often involves conflicting interests of opposite parties. The extensive documentation of disputes and the adversarial nature of legal procedure makes AI and Law an excellent domain for providing methods to compare and contrast between alternative opinions and for giving explanations and justifications for decisions. Therefore, argumentation has been recognised as a core topic in AI and Law. Understanding of the structure and acceptability of arguments is essential for a computer system to be able to engage in exchanges of arguments. Legal systems have benefitted considerably from the development of argumentation in AI, as in these domains there is a need for decision making based on incomplete or uncertain information. Incorporating elements from the theories of argumentation into AI applications has the obvious advantage of allowing these systems to make use of uncertain or incomplete knowledge available to them. In such situations argumentation can play the important role of providing tentative conclusions for or against a claim in the absence of further information to the contrary.

From the early 1990s several areas of computer science also became interested in the dialogical aspect of argumentation, including AI and Law. Dialogue systems were originally introduced into AI and Law as a way of modelling legal procedures [70], and more recently for legal reasoning [107] or for exploring particular legal phenomena such as burden of proof [71, 109]. Dialogues in the legal domain combine arguments from different resources, i.e. argument about the case evidence and facts, argument from legal rules, argument from precedent cases, argument from hypothetical tests and others that are required to resolve ambiguity and conflicts. In general, in conflict resolution dialogues the outcome is not fully determined by the participants. A typical example is legal procedure, where a third party, such as a judge or jury, normally determines the outcome of the case.

At first sight, people might think that Law as a body of rules and legal reasoning is developed using techniques of knowledge representation and automated deduction [93], but it is not. Legal reasoning goes beyond the literal meaning of the legal rules and requires consideration of context. Further aspects of law, related to providing a *test* so that a landmark case can be applied to future cases, applying the law to particulars of the case taking into consideration the social effects and changes in social objectives, in addition to the adversarial nature of legal procedures, may all require that a legal rule is overridden or changed.

A legal case starts with a story told by a client. The lawyer then interprets this story, according to the case context, based on a set of facts so that it can be fitted to an applicable law. Once the interpretation is determined an argument is constructed using some case precedents. Thus a legal case passes through various modes of reasoning: determining facts, classifying the facts under legal concepts of conditions, and deriving legal consequences from these facts. Therefore, computational argumentation systems developed to simulate legal reasoning are used to target different challenges [36, 110]. Systems can be used to store conflicting interpretations [98], provide reasoning with conflict rules [107, 108], propose alternative case decisions, use precedent cases to generate arguments [10, 14], or as mediators between disputing parties by structuring and recording their arguments and responses [34, 70, 92].

All this makes AI and Law a thoroughly interdisciplinary field, as it benefits from the synergy between the different kinds of systems investigated; theoretical systems (legal theory) are used to learn about artificial systems while remaining grounded by the perspective on natural systems (argumentation, practice of law) (see Figure 1.1) [31, p.308]. There is a synergy not only between law and AI, but also between AI and AI and Law. Better understanding of AI problems can benefit law, and applying law in AI can benefit AI. Applying argumentation theories and techniques in real, documented examples in Law instead of hypothetical ones has contributed significantly to the improvement of AI applications from which society as a whole can benefit. It also provides insights about the limitations of existing techniques, suggesting refinements and extensions of the models, or developing new approaches.

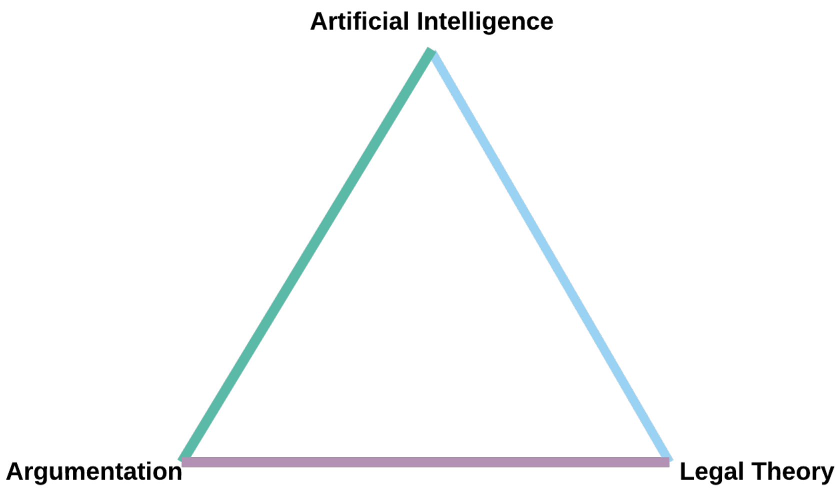


FIGURE 1.1: The Interdisciplinary Nature of AI and Law Systems

1.1.1 Legal Reasoning Model

Modelling reasoning in legal cases has been a central topic of AI and Law from the beginning, and there is now a good degree of consensus, especially with regard to the main elements involved. This consensus can be expressed as a tree of inference, with a legal decision as the root and evidence as the leaves. Between the two there are a number of distinct layers. Immediately below the decision there is the issues level [55], or values [38], which provide the reasons why the decision is made.

The idea here is that laws are made (and applied) so as to promote social values. Whether a value is promoted or not by a particular decision is an issue. Where more than one value is involved and they point to different decisions, the conflict needs to be resolved. Sometimes it is appropriate to give priority to one value over another (as in [38]); sometimes a balance needs to be struck (as in [87]).

At the next level down there are a number of *factors* [10]. Factors are stereotypical fact patterns which, if present in a case, favour one side or the other by promoting a value if that side wins, and so are used to resolve the issues. Factors are required to enable generalisation across the infinitely varied fact situations that can arise, and so permit the comparison of cases. Sometimes (as in [10]) it may be convenient to group several factors together under more abstract factors, so that there may be two or three layers of factors, moving from base level factors through more abstract factors, before reaching the issues. Below the factors there are the fact patterns used to determine their presence. These may offer necessary and sufficient conditions, but more often they offer either a set of sufficient conditions, or in less clear-cut cases, a number

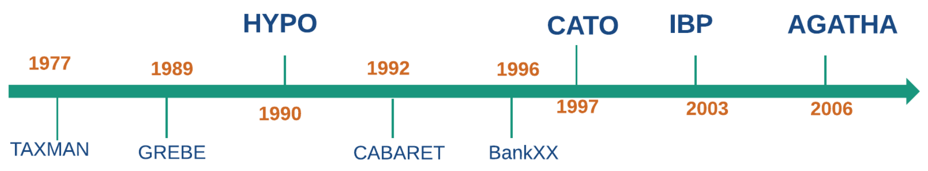


FIGURE 1.2: Legal Case-Based Reasoning Systems Time Line

of facts supplying reasons for and against the presence of the factor, which need to be considered and weighed to make a judgment [20]. At the lowest level there is the evidence. Facts are determined by particular items of evidence, and where evidence conflicts, a judgment will need to be made: often this judgment is made by a jury of lay people rather than lawyers. In the lower courts there will be real items of evidence, particular witness testimonies and the like. But by the time a case reaches the Supreme Court, the facts are usually considered established and beyond challenge. The Supreme Court does, however, need to consider what is admissible and what should count as evidence, and whether this will generally be available, so that the rule governing their decisions can be applied in future cases.

Thus a complete argument for a case will comprise a view on what can be considered as evidence for relevant facts: what facts are required to establish the presence of various factors, and how they relate; how the factors can be used to determine the issues; and, where issues and values conflict, how these conflicts should be resolved.

Several systems have contributed to this understanding in legal reasoning in the domain of AI in law. HYPO [14], the parent, is a case-based reasoning system that supports the selection of similarity between cases based on factors treated as dimension points. The main goal of HYPO is to create arguments for a legal case using precedent cases, without making decisions on the current case. CATO (HYPOs child) [10] was primarily directed at law school students, and was intended to help them form better case-based arguments, in particular to improve their skills in distinguishing cases by grouping these factors into a factor hierarchy. IBP (HYPOs grandchild) [17] partitions the factors into case issues and uses case-based techniques to resolve the conflicts within issues and predict the outcomes of cases.

HYPO, CATO and IBP all operates on the domain of US Trade Secret Law, which concerns the protection of technological and commercial information not generally known in the trade against unauthorised commercial use (this legal domain is explained in Chapter 3). AGATHA system [57, 58] has also used cases from the domain of US Trade Secret Law, however, the main purpose of AGATHA is to construct a case law theory of the domain, starting from factors rather than the domain itself, which explains as many of the existing cases as possible while giving the desired outcome for the current case.

Figure 1.2 shows a time line for some of the landmark legal reasoning systems; all these systems are based on an analysis of the case opinion argument and finding the similarities between cases in a specific case domain. The systems above the line are those directly relevant to this thesis, will be discussed in detail in Chapter 2.

1.1.2 Motivation

Although AI and Law has been around for more than three decades, and there has been much interesting research [31], there has been disappointingly little penetration into legal practice. One important exception is the approach to moving from written regulations to an executable expert system based on [85], which has been developed through a series of ever larger companies: Softlaw, Ruleburst and, currently, Oracle.

In the past few years or so, however, there is an unprecedented degree of interest in AI and its potential for supporting legal practice. This is evidenced by articles in the legal trade press such as *Legal Business*[1] and *Legal Practice Management*[2]; national radio programmes such as *Law in Action*[3] and *Analysis*[4] and Professional Society events, such as panels run by the Law Society of England and Wales[5]. The legal profession has never been so interested in, and receptive to, the possibilities of AI for application to their commercial activities. There are, therefore opportunities which need to be taken.

One of the major lessons that can be drawn from Softlaw and its successors is the need for a methodology. A methodology gives some assurance to clients that their engagement with AI has some prospects of success: they are not so interested in furthering research activities as in increasing profits and the use of an established methodology can allow them to know how their particular problem will be addressed and what will be produced at the end of the process. Equally important, as the Softlaw experience also showed, is the existence of tools to support the methodology. Such tools reinforce the methodology, make it more teachable and reproducible and shorten the development time.

This thesis targets this need by presenting a knowledge engineering methodology to facilitate analysis and provide a way for recording the fruits of this analysis by encapsulating the knowledge of different legal domains, rather than just one domain as in the previous legal case-based reasoning systems, using *Abstract Dialectical Frameworks* (ADF) [53, 54], a recent development in abstract argumentation (see Chapter 2 for argumentation frameworks) and a generalisation of Dungs Abstract Argumentation Framework (AF) [61]. The final result is ANGELIC (ADF for kNowledGe Encapsulation of Legal Information from Cases) methodology. Note that ADFs are primarily used to record knowledge, not to execute it.

As mentioned above, work in AI and Law has established a fairly clear model of legal reasoning in which cases are described in terms of legally significant factors, and these sets of factors provide reasons to decide cases in a particular way, determined by precedent cases. AI and Law has not, however, provided a methodology to extract the relevant factors in the domain, show the factors presented in a case and find the relationship between these factors and other components of the reasoning, such as facts and issues in a more comprehensive way.

[1] *AI and the law tools of tomorrow: A special report*. www.legalbusiness.co.ukindex.phpanalysis4874-ai-and-the-law-tools-of-tomorrow-a-special-report. All websites accessed in January 2017.

[2] *The Future has Landed*. www.legalsupportnetwork.co.uk. The article appeared in the March 2015 edition.

[3] *Artificial Intelligence and the Law*. www.bbc.co.uk/programmes/b07dlxmj.

[4] *When Robots Steal Our Jobs*. www.bbc.co.uk/programmes/b0540h85.

[5] The full event of one such panel can be seen on youtube at www.youtube.com/watch?v=8jPB-4Y3jLg. Other youtube videos include Richard Susskind at www.youtube.com/watch?v=xs0iQSyBoDE and Karen Jacks at www.youtube.com/watch?v=v0B5UNWN-eY.

HYPO, CATO and IBP systems focus on the reasoning involved and provide a program to model this reasoning. In contrast, ANGELIC provides a method for performing the domain analysis required for such systems. Moreover, unlike AGATHA, the theory in AGATHA is constructed from factors rather than the domain itself.

The legal case analyses provided in ANGELIC show

- *how argument components*, not the argument, tend to be proposed and refined in oral argumentation dialogues, in (i) *a dialogue representation model* that produces a visual representation of the argument components exchanged by the dialogue parties in a graphical form, taking as input the dialogue speech acts and showing how the accepted components can be formed, the conditions under which they apply,
- and how relationships between them can be related to the decision in a (ii) *a decision justification model*. The legal reasoning methodology is then applied over various legal domains, using analyses produced by different people for use by different programs for different tasks to show that it is able to documenting knowledge derived by cases in a variety of domains.

The remainder of this chapter outlines an introduction to the research described in this thesis. The next section (Section 1.2) presents the research questions and the associated research issues that the work is directed towards. Section 1.3 describes how the research issues are addressed (the research methodology) while Section 1.4 describes the contributions of the work presented in this thesis. Section 1.5 presents the published work to date resulting from the content of the thesis and Section 1.6 describes how the rest of this thesis is structured. This chapter is concluded with a brief summary in Section 1.7.

1.2 Research Issues and Questions

Given the overview from the introduction above, the aim of the research presented in this thesis is:

> *To provide a methodology to encapsulate analyses of a variety of case law domains, potentially performed by different people, for different purposes, using different techniques, in a common format that can be readily realised in a computational form.*

To address this aim, the research is directed to answer the following research questions:

RQ1 *Does abstract argumentation provide a format capable of capturing a variety of case law domains?*

RQ2 *How can computational dialogue be used to identify relevant components to instantiate the abstract argumentation format used?*

RQ3 *How can we produce an executable program from this argumentation format in a standard manner such that cases are decided by the program to an acceptable degree of accuracy?*

RQ4 *How can this executable representation be used to evaluate and refine the analysis?*

RQ5 *What additional information needs to be provided to support the particular aspects of applications such as explanation?*

The first part of this research aims to address RQ1 and RQ2 by modelling a representation for the factor-based argumentation through which Supreme Court Oral Hearings (in which counsels for the parties present and clarify their arguments), can be analysed in order to identify the components from which arguments are constructed for delivering the opinion (in which the justices supply a reasoned judgment). The second part concerns providing *analysis* and *justifications* for legal decisions through a *representation* using *Abstract Dialectical Frameworks (ADFs)* to explore RQ3, RQ4, and RQ5 in addition to RQ1. This ensures the following research objectives are fulfilled:

Objective 1 Provide an approach to analyse and represent a legal procedure dialogue in a real context to provide an argument structure from the argument components.

Objective 2 Employ a knowledge engineering approach to capture knowledge in a legal domain, and to reason with this representation, to provide justification and transparency of legal decisions.

Objective 3 Develop a justification model that can be applied over different legal case domains using abstract argumentation, and compare the outcome to the actual legal decisions.

Objective 4 Conduct a comparative evaluation to compare the performance of the defined approach between the different domains, and other legal reasoning systems.

Objective 5 Discover various methods to improve the explanation of legal decisions in the program output.

In the conclusions in Chapter 8, these research questions and objectives will be re-visited to discuss how well they have been met using the methodology presented in the next section.

1.3 Research Methodology

Five research phases are followed in the order below to fulfill the aims of the research conducted in this thesis, and to address the research questions identified in the previous section. Figure 1.3 illustrates the research framework, showing the relationships between these phases. Note that all the selected legal domains in the research are related to the US Courts, where all the case briefs, Oral Hearings and Opinions are readily and freely available to the public and have been previously examined in the AI and Law literature.

Phase 1: Analysis and Representation of Dialogue This phase presents a model that provides a structured analysis of the dialogues conducted within the US Supreme Court Oral Hearings to enable the construction of what has been called *argument component trees* (ACTs)

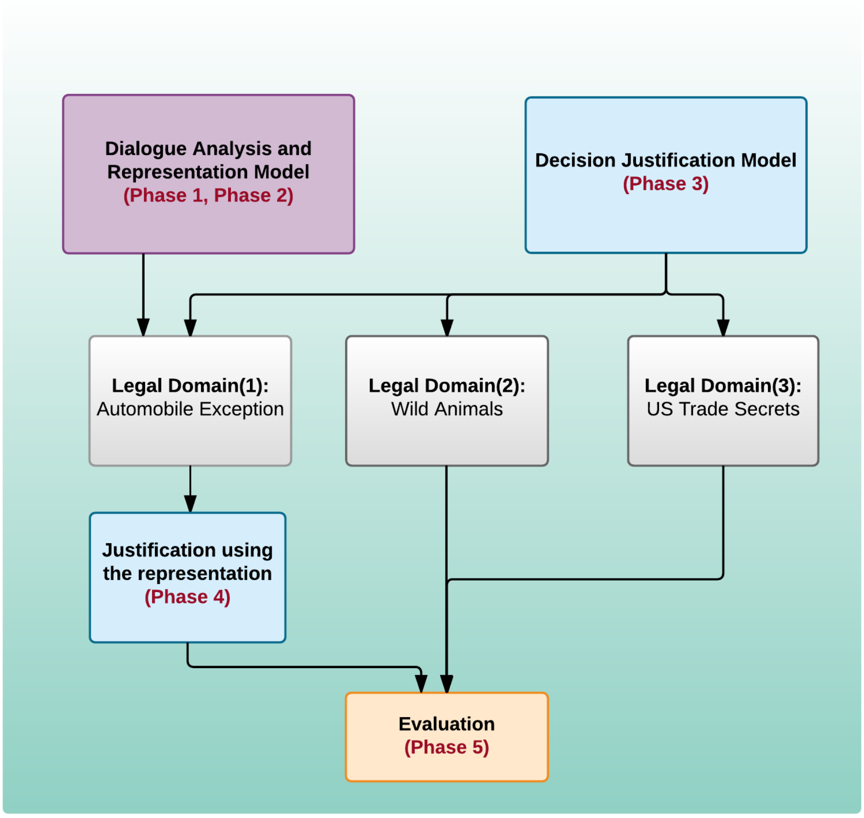


FIGURE 1.3: Research Framework

in which the issues, facts and factors are proposed and refined through the use of a set of defined speech acts. Following on from the analysis, *a dialogue representation model* is defined to identify the moves made during the hearings. The model is validated through application to a selected case domain, *the Automobile Exception to the US Fourth amendment*. The oral hearing dialogues of two landmark cases, *US v. Chadwick* and *California v. Carney*, related to this domain are analysed and their ACTs are constructed to show the relevant components. The following steps show the methodology applied to fulfill the aims of this phase:

1. Locate the Oral Hearing dialogues in the overall Supreme Court process; identify that the dialogue consists of three distinct sub-dialogues; and characterise the three sub-dialogues in terms of their initial state, and individual and collective goals following Walton and Krabbe's dialogue typology in [133].
2. Define a set of speech acts to enable the goals of the dialogues to be achieved in the form of assertions and associated questions, and determine the pre- and post-conditions of the moves to define a protocol for the dialogue.

3. Define a *visual representation* as an inference ACT to show how the components of the arguments are asserted, and related to each other, updated after each move in the dialogue.
4. For each case study, analyse the transcript of the Oral Hearing, mark each utterance with the appropriate speech act and construct the ACTs by executing the dialogue.
5. Update a *legal ontology* that serves as a repository for the argument components in the domain.
6. Establish a *legal case analysis work flow* to clarify the stages of the analysis and how they are related to the legal ontology used, the implemented program and to the legal opinion.

Phase 2: Semi-automated Dialogue Representation Model Having established the model for conducting the analysis task, the next step was to move towards automation, as follows:

1. Define a precise grammar setting out the rules for how the components of the ACTs can be combined to construct the trees.
2. Implement a program to provide a visual representation of the dialogue by producing ACTs of the relevant components for each party in the dialogue and provide input to the decision making process. The program will focus only on validating the analysis of the Oral Hearing dialogues and organising the ACT components.
3. Test the program over scenarios from the two analysed case studies.

Phase 3: Abstract Dialectical Framework to Justify Legal Decisions To find the relationship between the opinion argument and the represented issues, factors and facts, the factor-based reasoning [10] has been expressed in terms of ADFs [54]. Further clarification is provided to show how the constructed ADF can be used to give justifications for case decisions in the domain by implementing a Prolog program that ascends through this ADF, identifying the factors present in the case and resolving the issues to predict the outcome. The program thus takes as input the case representation and produces the accepted components that determine the case decision. Following the analysis of CATO [10], IBP [17] and AGATHA [57], the US Trade Secrets domain is selected to provide a *comparative evaluation* with IBP (Issue-Based Prediction) and AGATHA (Argument Agent for Theory Automation), through the following method:

1. Study the analysis of CATO [10] and IBP [17] and map the factor hierarchy tree into an ADF.
2. Construct the ADF that expresses the theory of knowledge of the analysed domain.
3. Define the ADF's *acceptance conditions* for the different components using the cases' representation and decision.
4. Implement a Prolog program by rewriting the acceptance conditions as groups of Prolog clauses to determine the acceptability of each node in terms of its children and adding some reporting to indicate whether the node is satisfied.

5. Apply the program to the 32 cases used in AGATHA and record the results (the accepted components).
6. Compare the achieved results with the CATO, IBP and AGATHA programs reported in [38] and [57], and analyse the findings.

Phase 4: Justification using the Representation The previous phase described how the ADF and factor-based reasoning can be related. This stage shows how this finding can be used to provide justification for the decision based on the ACTs constructed from the Oral Hearing dialogues. This will involve:

1. Using the ACTs constructed from the case studies to provide an approach for merging the ACTs into one case ADF.
2. Defining the *acceptance conditions* for the different ACT components according to the factor acceptance conditions defined above.
3. Constructing the final case ADF that represents the decision justification from the accepted components.
4. Providing steps for the decision explanation and justification model that generates the ADF from ACTs.

Phase 5: Generalisation and Evaluation Following the methodology presented above, a further evaluation is conducted by applying the justification model over different legal case domains, examining the results and comparing the outcome with the actual case decisions. This involves:

1. Capturing and representing the knowledge of two more domains: The Wild Animals (5 cases) [30], and Automobile Exception to The Fourth Amendment (10 cases) [29]. The analysis of the Wild Animals domain is based on [30]; while the analysis of the Automobile Exception domain, although starting from [115] and [29] was produced specifically for this thesis.
2. Conduct a comparative evaluation between the domains that tests the ease of implementation, the performance and efficacy of the resulting program, the ease of refinement and the transparency of the reasoning.
3. Investigate further improvements to the program output by incorporating the *case facts* to improve the decision justification and reasoning using *portions of precedents*.
4. Amend the program rules to produce another opinion, the *dissenting* (minority) opinion.

1.4 Research Contributions

The work described in this thesis makes a number of contributions, as follows:

1. Examination of the legal procedure in the US Supreme court and characterisation of the three sub-dialogues of the Oral Hearing in terms of their initial state, and individual and collective goals as required by Walton and Krabbe in [133].

2. A dialogue representation model, which provides a structured analysis of US Supreme Court Oral Hearings that enable the construction of *argument component trees* in which the issues, facts and factors are proposed and refined through the use of a set of defined speech acts. This involves a manual mark-up for the argument components of the Oral Hearing transcripts of two legal cases and a workflow for analysing the legal cases and relate the produced argument components to the legal opinion. This mark-up can then be input to the oral hearing dialogues program to provide a visual representation by executing the dialogue.

3. A correspondence between the components of the arguments in the dialogue representation and the legal opinion.

4. The application of ADFs as powerful, general, abstract frameworks for argumentation to drive and record the design of a knowledge base to encapsulate a body of case law.

5. A solid formal basis for factor-based reasoning using ADFs that was lacking from previous systems used for reasoning about cases.

6. ANGELIC, a methodology for developing a decision justification model using Abstract Dialectical Frameworks that is able to encapsulate and apply theories of case law represented as ADFs. The method has been tested by producing programs for three domains: US Trade Secrets from the factor hierarchy of CATO [10] and IBP [55] (ANGELIC Trade Secrets), the Wild Animals domain from [30] (ANGELIC Animals), and a set of cases in the Automobile Exception to The Fourth Amendment domain (ANGELIC Automobile).

7. A comparative evaluation between the domains that tests the ease of implementation, the performance and efficacy of the resulting program, the ease of refinement and the transparency of the reasoning.

8. A comparative evaluation between ANGELIC, the methodology applied in this thesis, and previous legal case-based reasoning systems.

9. A means of improving the justification of the ANGELIC program by incorporating the *case facts*, to provide closeness to the reported opinion.

10. Identification of ways to improve the explanation of the output from the decision justification model using portion of precedents and the generation of multiple opinions.

1.5 Publications

Much of the work included in this thesis has been reported in various articles, refereed conference proceedings, workshops and seminars. Segments of work presented in this thesis have been published in papers along with the author's supervisors, Katie Atkinson and Trevor Bench-Capon, as shown below:

Journal Paper

1. L. Al-Abdulkarim, K. Atkinson and T. Bench-Capon (2016): *A methodology for designing systems to reason with legal cases using abstract dialectical frameworks*. Artificial Intelligence and Law. Vol 24(1), pp. 1-49.

   Paper Abstract [8]: This paper presents a methodology to design and implement programs intended to decide cases, described as sets of factors, according to a theory of a particular domain based on a set of precedent cases relating to that domain. We use Abstract Dialectical Frameworks (ADFs), a recent development in AI knowledge representation, as the central feature of our design method. ADFs will play a role akin to that played by Entity-Relationship models in the design of database systems. First, we explain how the factor hierarchy of the well-known legal reasoning system CATO can be used to instantiate an ADF for the domain of US Trade Secrets. This is intended to demonstrate the suitability of ADFs for expressing the design of legal cased based systems. The method is then applied to two other legal domains often used in the literature of AI and Law. In each domain, the design is provided by the domain analyst expressing the cases in terms of factors organised into an ADF from which an executable program can be implemented in a straightforward way by taking advantage of the closeness of the acceptance conditions of the ADF to components of an executable program. We evaluate the ease of implementation, the performance and efficacy of the resulting program, ease of refinement of the program and the transparency of the reasoning. This evaluation suggests ways in which factor-based systems, which are limited by taking as their starting point the representation of cases as sets of factors and so abstracting away the particular facts, can be extended to address open issues in AI and Law by incorporating the case facts to improve the decision, and by considering justification and reasoning using portion of precedents.
   This paper is an extension of ICAIL 2015 [5] conference paper. Much of the content of this paper provided the basis of the work presented in Chapters 5, 6 and 7.

Conference Proceedings Contributions

1. L. Al-Abdulkarim, K. Atkinson and T. Bench-Capon (2016): *ANGELIC Secrets: Bridging from Factors to Facts in US Trade Secrets* - JURIX 2016: The Twenty-Ninth Annual Conference, Nice, France, pp.113-118. Frontiers in Artificial Intelligence and Applications, Vol. 294. IOS Press, Amsterdam, The Netherlands.

   Paper Abstract [9]: The ANGELIC (ADF for kNowledGe Encapsulation of Legal Information from Cases) project provided a methodology for implementing a system to predict the outcome of legal cases based on a theory of the relevant domain constructed from precedent cases and other sources. The method has been evaluated in several domains, including US Trade Secrets Law. Previous systems in this domain were based on factors, which are either present or absent in a case, and favour one of the parties with the same force for every factor. Evaluations have, however, suggested that the ability to represent different degrees of presence and absence, and different strengths, could improve performance. Here we extend the methodology to allow for different degrees of presence and support, by using dimensions as a bridge between facts and factors. This new program is evaluated using a standard set of test cases.

   The work presented in this paper acted as a foundation for part of future work (discussed briefly in Chapter 8) related to this thesis.

2. L. Al-Abdulkarim, K. Atkinson and T. Bench-Capon (2015): *Evaluating an approach to reasoning with cases using abstract dialectical frameworks.* In: Proceedings of the Fifteenth International Conference on Artificial Intelligence and Law (ICAIL 2015), San Diego, USA, pp. 156 - 160. ACM Press.

   Paper Abstract [5]: Abstract Dialectical Frameworks (ADFs) are a recent development in computational argumentation which are, it has been suggested, a fruitful way of implementing factor-based reasoning with legal cases. In this paper we evaluate this proposal, by reconstructing CATO using ADFs. We evaluate the ease of implementation, the efficacy of the resulting program, ease of refinement of the program, transparency of the reasoning, relation to formal argumentation techniques, and transferability across domains.

   The content of this paper summarises the material presented in Chapters 5 and 6 and part of the evaluation in Chapter 7.

3. L. Al-Abdulkarim, K. Atkinson and T. Bench-Capon (2015): *Factors, issues and values: revisiting reasoning with cases.* In: Proceedings of the Fifteenth International Conference on Artificial Intelligence and Law (ICAIL 2015), San Diego, USA, pp. 3-12. ACM Press.

Paper Abstract [6]: In this paper we revisit reasoning with legal cases, with a view to articulating the relationships between issues, factors, facts and values, and to identifying areas for future work. We start from the different ways in which attempts have been made to go beyond a fortori reasoning from the precedent base, so that conclusions not fully justified by the precedents can be drawn. We then use a particular example domain taken from the literature to illustrate our preferred approach and to relate factors and values. From this we observe that much current work depends critically on the ascription of factors to cases in a Boolean manner, while in practice there are compelling reasons to see the presence of factors as a matter of degree. On the basis of our observations we make suggestions for the directions of future work on this topic.

The work presented in this paper acted as a foundation for part of the future work related to this thesis, discussed briefly in Chapter 7.

4. L. Al-Abdulkarim, K. Atkinson and T. Bench-Capon (2015): *Using abstract dialectical frameworks to argue about legal cases* In: Proceedings of the First European Conference on Argumentation (ECA 2015), Lisbon, Portugal, pp.163-180.

   Paper Abstract [7]: Recent work has shown how to map factor hierarchies for legal reasoning into Abstract Dialectical Frameworks (ADFs), by defining acceptance conditions for each node. In this paper we model as ADFs bodies of case law from various legal domains, rewrite them as logic programs, compare the results with previous legal reasoning systems and propose improvements by increasing the scope of reasoning downwards to facts.

   The content of this paper is an extended version of ICAIL conference paper no.1. The aim of this paper is to present the application of abstract argumentation in the legal domain, to a different audience, concerned with argumentation in general rather than law in particular.

5. L. Al-Abdulkarim, K. Atkinson and T. Bench-Capon (2014): *Abstract dialectical frameworks for legal reasoning.* In R. Hoekstra (editor): Legal Knowledge and Information Systems - JURIX 2014: The Twenty Seventh Annual Conference, Krakow, Poland, pp.61-70. Frontiers in Artificial Intelligence and Applications, Vol. 271. IOS Press, Amsterdam, The Netherlands.

   Paper Abstract [3]: In recent years a powerful generalisation of Dung's abstract argumentation frameworks, Abstract Dialectical Frameworks (ADF), has been developed. ADFs generalise the abstract argumentation frameworks introduced by Dung by replacing Dung's single acceptance condition (that all attackers be defeated) with acceptance conditions local to each particular node. Such local acceptance conditions allow structured argumentation to be straightforwardly incorporated. Related to ADFs are prioritised ADFs, which allow for reasons pro and con a node. In this paper we show how these structures provide an excellent framework for representing a leading approach to reasoning with legal cases.

The work presented in this paper acted as the foundation for the work presented in Chapters 5 and 6.

6. L. Al-Abdulkarim, K. Atkinson and T. Bench-Capon (2014): *Support for factor-based argumentation.* In S. Parsons, N. Oren, C. Reed and F. Cerutti (editors): Computational Models of Argument, Proceedings of COMMA 2014, The Scottish Highlands, UK, pp. 447-448. Frontiers in Artificial Intelligence and Applications, Vol. 266. IOS Press, Amsterdam, The Netherlands.

   Paper Abstract [4]: In this paper we describe the analysis of arguments for the purpose of building computational models which use factor based reasoning. As such our emphasis is on the identification of the components of arguments and their relationships, rather than the structure of particular arguments. Factor-based reasoning is characterised by its use of concepts which are resistant to definition in terms of necessary and sufficient conditions. Such concepts instead require classification on the basis of "family resemblance", whereby a number of features must be considered. None of these factors, individually or collectively, can be seen as necessary or sufficient and so a judgment must be made by balancing the reasons in favour and the reasons against. We also describe a program which will provide support for some of the stages central to the analysis process.

   This is a short paper accompanying the demonstration of the program and case studies presented in Chapter 4.

7. L. Al-Abdulkarim, K. Atkinson and T. Bench-Capon (2013): *From Oral Hearing to Opinion in the US Supreme Court.* In K.D. Ashley (editor): Legal Knowledge and Information Systems. JURIX 2013: The Twenty Sixth Annual Conference, Bologna, Italy, pp.1-10. Frontiers in Artificial Intelligence and Applications, Vol. 259. IOS Press, Amsterdam, The Netherlands.

   Paper Abstract [2]: In this paper we provide a structured analysis of US Supreme Court Oral Hearings to enable identification of the relevant issues, factors and facts that can be used to construct a test to resolve a case. Our analysis involves the production of what we term "argument component trees"(ACTs) in which the issues, facts and factors, and the relationships between these, are made explicit. We show how such ACTs can be constructed by identifying the speech acts that are used by the counsel and Justices within their dialogue. We illustrate the application of our analysis by applying it to the Oral Hearing that took place for the case of Carney v. California, and we relate the majority and minority opinions delivered in that case to our ACTs. The aim of the work is to provide a formal framework that addresses a particular aspect of case-based reasoning: enabling the identification and representation of the components that are used to form a test to resolve a case and guide future behaviour.

This paper extends the CMNA workshop paper mentioned below. Part of the material presented in this paper set the basis of Chapter 3 and Chapter 4.

Workshop Contributions

1. L. Al-Abdulkarim, K. Atkinson and T. Bench-Capon (2013): *Dialogues in US Supreme Court Oral Hearings.* In: Proceedings of the Thirteenth Workshop on Computational Models of Natural Argument (CMNA 2013).

   Paper Abstract [1]: Dialogue protocols in Artificial Intelligence and Law have become increasingly stylised, intended to examine the logic of particular legal phenomena such as burden of proof, rather than the procedures within which these phenomena occur. While such work has provided some valuable insights, the original motivation still matters, and so in this paper we will return to the original idea of using dialogue moves to model particular procedures by examining some very particular dialogues - those found in Oral Hearings of the US Supreme Court. We will characterise these dialogues, and illustrate the paper with examples taken from a close analysis of a case often modelled in AI and Law, California v Carney (1985). This paper presents the preliminary investigation required to identify tools to provide computational support for the analysis of Oral Hearings.

   The work presented in this paper acted as the foundation for the dialogue representation model. Part of the paper's material is presented in Chapters 3 and 4.

Doctoral Consortium Reports

1. L. Al-Abdulkarim (2013): *Dialogue Interactions in Oral Hearings.* In: CEUR Proceedings, The Doctoral Consortium of the Twenty Sixth International Conference on Legal Knowledge and Information Systems (JURIX 2013), Bologna, Italy.

   The work presented in this report is related to phase 1 and phase 2 (Chapter 3 and Chapter 4) of the research methodology. This paper and accompanying presentation won the prize for the best Doctoral Consortium contribution.

1.6 Thesis Structure

This section outlines the structure of this thesis, indicating the related appendices for each chapter.

Chapter 2 presents an overview of the research literature that is relevant to the issues addressed in this thesis. The chapter is divided into three separate parts: Argumentation in AI and Law, dialogues, and frameworks for argumentation. Each section explains the topics related to this research, with an emphasis on how it has been used in this thesis.

Chapter 3 provides some necessary legal background for this thesis. Firstly, it explains the *procedures* of the US Courts, with a focus on the US Supreme Court to show the role played by the *Oral Hearings* and *Legal Opinion* stages, the main procedures on which this research is based. Secondly, the chapter gives an overview of the *legal cases domains* that have been used as case studies in the thesis.

Chapter 4 provides a structured analysis of US Supreme Court Oral Hearings. The chapter proposes a legal dialogue representation model to enable the identification and representation of the argument components (issues, factors, and facts) that are used to form the tests proposed to resolve a case and so establish a knowledge base for a legal domain that can guide the design of the legal decision justification model. The analysis involves the production of *argument component trees* by identifying speech acts to mark the counsels' and Justices' utterances within their dialogues. A legal decision workflow is presented to show how the Oral Hearing analysis leads to the case decision, taking the role of an ontology that serves as a repository for the argument components of a legal domain. The specification and implementation of the representation model are briefly discussed, and the design details are provided in **Appendix A**. The model is illustrated by analysing the Oral Hearings of two case studies *US v. Chadwick* and *Carney v. California*. The full analyses of these cases are provided in **Appendix B** and **Appendix C** respectively.

Chapter 5 presents ANGELIC, a knowledge engineering methodology for encapsulating legal information from cases, and shows how it is used for expressing the design of legal case-based reasoning systems. The factor hierarchy of CATO [10] is used to instantiate an ADF for the domain of US Trade Secrets considering the similarity in the structure between the factor hierarchy in (CATO, IBP) and ADF. The factor definitions of the US Trade Secrets domain are given in **Appendix D**. Moreover, the chapter investigates the relationship between the opinion arguments and the represented issues, factors, and facts in the ACTs produced in Chapter 4, showing how these ACTs can be merged and expressed in terms of ADFs.

Chapter 6 explores the application of ANGELIC to a range of legal domains to construct systems that predict the outcome of legal cases described as sets of factors, according to a theory of a particular domain, based on a set of precedent cases relating to that domain. The design is provided by the domain analyst, with each domain expressing the cases in terms of factors organised into an ADF from which an executable program can be implemented. A number of evaluations are conducted to test the ease of implementation, the performance and efficacy of the resulting program and the ease of refinement of the program. The chapter is extended for the Automobile Exception domain (the factors definitions are explained in **Appendix E**) to explore the relationship between the Oral Hearing and the Opinion. Furthermore, the domain program applies reasoning using the fact layer, and includes new rules to consider dissenting opinions.

The Prolog program and the results for each legal domain are included in the appendices: US Trade Secrets domain in **Appendix F**, Wild Animals in **Appendix G**, and Automobile Exception domain in **Appendix H**.

Chapter 7 After applying the methodology to several legal domains, this chapter provides a number of comparative evaluations: an evaluation to compare the performance of ANGELIC on the three legal domains; and a comparative evaluation to compare ANGELIC with other legal reasoning systems. Further suggestions are discussed to compare the transparency in reasoning at different reasoning levels, and to set the foundations for future work.

Chapter 8 This chapter concludes the thesis by reviewing the contributions and main findings in terms of the identified research questions and issues. The chapter also revisits the research objectives and presents some ideas for future work.

1.7 Summary

The main aim of this research is to provide a method to encapsulate analyses of a variety of case law domains, performed by different people, for different purposes, using different techniques, in a common format that can be readily realised in a computational form. This introductory chapter has presented a general overview of AI and Law, the main research domain in this thesis. In particular, this chapter explained the research questions together with the associated research objectives to be addressed, the adopted research methodology, and the contributions and publications that have been produced in the course of the research. Finally, the chapter provides the thesis road map, starting from Chapter 2, which provides the literature review aimed to establish the necessary background to the work presented in this thesis.

Chapter 2

Literature Review

"Student: Dr. Einstein, aren't these the same questions as last year's final exam?
Dr. Einstein: Yes; But this year the answers are different." Albert Einstein

---

This chapter presents an overview of the existing research literature that is relevant to the issues addressed in this thesis. As discussed in the introductory chapter, the main concern of this thesis is to provide a method to encapsulate analyses of a variety of case law domains in a common format. This involves extracting the relevant argument components from dialogues and legal decisions, then representing this knowledge in a suitable format and realising it in executable form, so that it can be tested, evaluated and refined. In order to fulfill this research aim in the appropriate broader context, this chapter is divided into three separate sections, covering relevant literature:

1. Argumentation in AI and Law: The chapter starts with an overview of the AI and Law domain in Section 2.1, describing the leading legal case-based reasoning systems, and discussing the current consensus in legal reasoning models, which has moved from case evidence to legal consequences.

2. Dialogues: Dialogue typologies and the structure of the dialogue models will be discussed in Section 2.2 with a focus on examples of legal dialogue systems, which relates to the material that will be presented in Chapter 3 and Chapter 4.

3. Frameworks for Argumentation: Section 2.3 presents an overview of argumentation frameworks, starting with abstract argumentation frameworks, Dung's frameworks [61] and refinements of these, giving more emphasis to Abstract Dialectical Frameworks [53, 54], and ending with structured argumentation: Argumentation Schemes [132], ASPIC+ [105] and Carneades [71].

The chapter concludes with a summary of the key points addressed, and their relationship to the research contribution of this thesis. Figure 2.1 provides a diagram showing the scope of the literature that addresses the issues of this thesis.

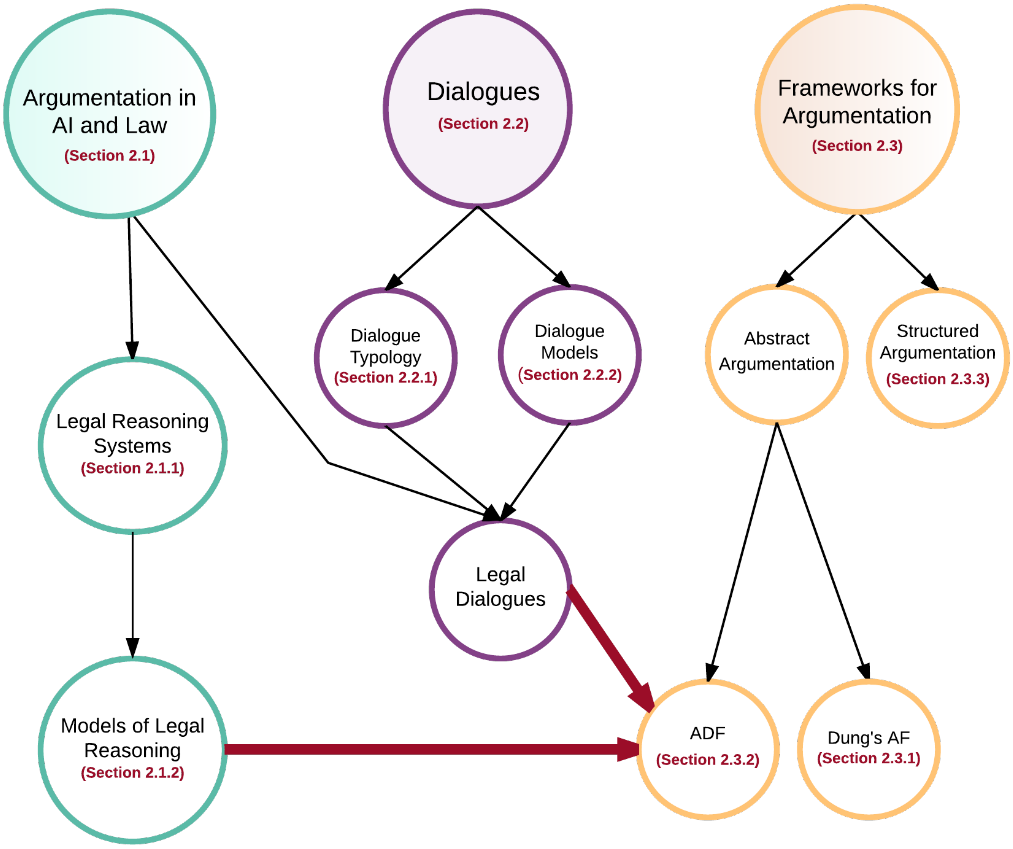


FIGURE 2.1: Main Topics in The Literature Review

2.1 Argumentation in AI and Law

The overview of AI and Law provided in the introductory chapter shows that a central theme in modelling legal reasoning is *argument*. Arguments can be reconstructed from legal documents (natural language texts) or generated from sources such as legislation rules, evidence, precedent cases and other sources.

Basically four main approaches are considered in modelling legal argument: logical models of legal argument, as recently presented by Prakken and Sartor in [110]; legal reasoning with argumentation schemes [73] (see sub-section 2.3.3 for argumentation schemes); arguments generated from legal procedures (e.g. [64, 70, 71, 109]); and arguments produced from legal case-based approaches (e.g. [10, 14, 55]).

Originally legal practices were interpreted as a set of rules that could produce and explain legislation and expert knowledge; however, these rules are defeasible and subject to interpretation and exceptions. Therefore, gaps and conflict between rules need to be resolved, and exceptions to these rules have to be considered. In case-based reasoning, the case is decided based on knowledge of law and precedent cases. Prakken and Sartor in [108] represent a precedent as two conflicting rules; pro-plaintiff factors and pro-defendant factors, plus a priority between them: if the plaintiff won then the first rule has priority over the second, if the defendant won then the second rule has priority over the first. This provides the bridge between case- and rules-based representation, allowing for the logical representation of case-based reasoning, enabling a theory-based reasoning approach [38] where precedent cases are used to explain the new case and conflicts are resolved by using rules from expert knowledge.

As stated in Chapter 1, every case starts with evidence and ends with a legal consequence. Modelling legal reasoning can be seen as a series of steps; on the basis of this evidence, a set of facts is established. After that, intermediate predicates [89], which relate strongly to the notion of factors as found in CATO [10] and IBP [55], are used to bridge from factors to normative consequences.

This section addresses the issues required to provide the knowledge representation used in this thesis. The first sub-section (Sub-section 2.1.1) describes the landmark case-based reasoning systems in the legal domain, the sub-section 2.1.2 shows where these systems are located in the models of legal reasoning. In addition to the legal case-based reasoning, this section provides a brief overview of another two branches of research in AI and Law: legal text processing and legal ontologies.

2.1.1 Legal Reasoning Systems

This sub-section explores a number of landmark case-based reasoning systems in the legal domain, which are foundational for discussion of reasoning with legal cases in AI and Law as illustrated in Figure 2.2. HYPO [14], one of the foundational systems, is a case-based reasoning system that supports the similarity between cases based on factors treated as dimension points. CATO [10] is used to explain how to distinguish cases by grouping these factors into a factor hierarchy. IBP [17] partitions the factors into case issues and uses case-based techniques to resolve the conflict within issues.

These systems are explained in more detail in the following sub-sections: an overview of each system is provided showing the purpose of the system, the legal case domain(s) used, and the case representation and reasoning approach with examples where appropriate. Some additional legal reasoning systems will be discussed briefly in the last sub-section.

HYPO

HYPO (1990) ([14, 116]) is a seminal system that introduced a new approach in legal reasoning using case-based reasoning. Background about the system showing the main features and functionality is presented here.

Purpose: The main goal of HYPO is to create arguments for a legal case using precedent cases, without making decisions on the current case.

Domain: HYPO operates on the domain of US Trade Secret Law, which concerns the protection of technological and commercial information not generally known in the trade against unauthorised commercial use. This domain is described in more detail in the next chapter.

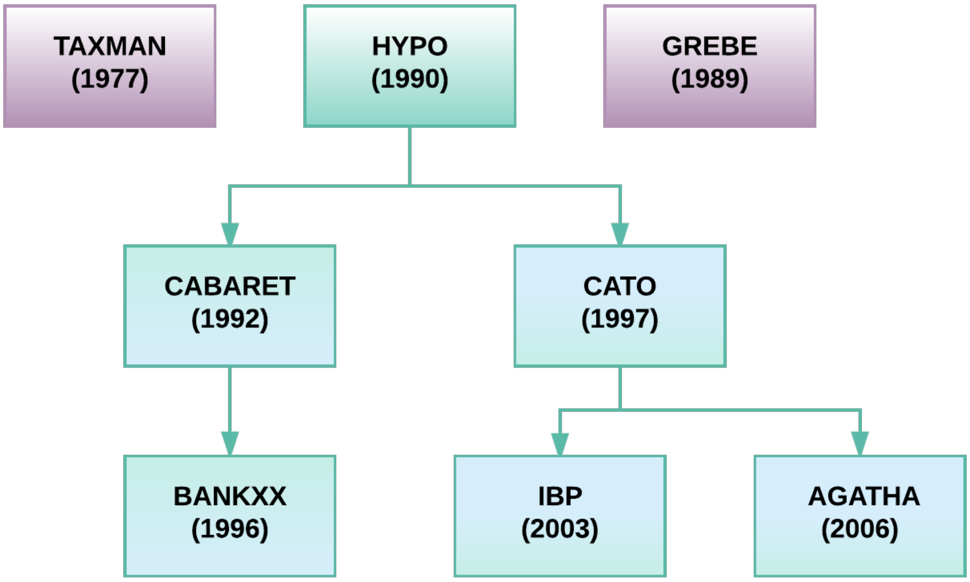


FIGURE 2.2: Landmark Case-Based Reasoning Systems.

Case Representation: Cases in HYPO are represented by *Dimensions*. These dimensions are invoked on the basis of the facts of the cases, and range from an extreme pro-plaintiff position to an extreme pro-defendant position. HYPO uses a Case Knowledge Base (CKB) which is a structured database consisting of a small number of both actual and hypothetical legal cases (30 cases) and a set of thirteen dimensions used as an index to retrieve these cases.

Reasoning process: HYPO was the first system to reason about precedent cases by using adversarial case-based reasoning, that is, by drawing an analogy to similar past cases in order to provide a justified conclusion that the new case should be decided in a similar manner to specific past cases. To fulfill this requirement HYPO's reasoning process consists of the following steps:

1. Analysing the fact situation for the case under consideration dimensionally using the dimensions prerequisites. HYPO then assigns both the applicable dimensions and near-miss dimensions.
2. Drawing factual analogies to past cases, by finding the similarities between the dimensions of the case under consideration and the precedent cases in the CKB. HYPO retrieves all the cases that have one or more dimensions in common with the new case. These cases may have been decided for the plaintiff or defendant.
3. Positioning the new case with respect to the retrieved precedent cases using claim lattices. HYPO creates two claim lattices: a regular claim lattice for applicable dimensions and an extended claim lattice for near-miss dimensions. The branches in these lattices represent different ways of arguing about the new case. The new

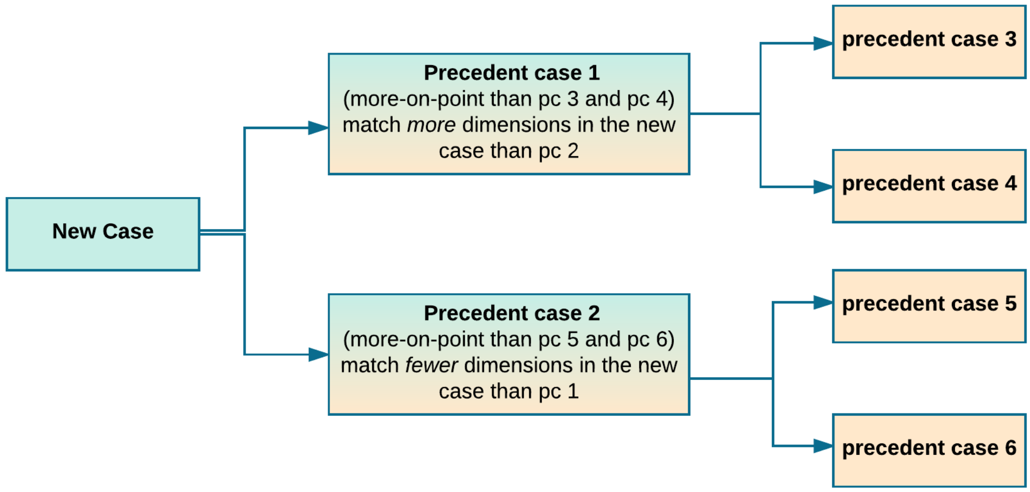


FIGURE 2.3: Example of Claim Lattice in HYPO

case is matched to the precedent cases with more on-point dimensions. Figure 2.3 shows an example of these lattices.

4. Select the best precedent cases. HYPO compares the most on-point precedent cases and selects the best cases that have at least one dimension that favours the party who won the case. The *more* on-point the cases are, the *less* opportunity there is for the opponent to distinguish the case.

5. Generate 3-ply argument for the current fact situation citing precedents. The first side applies analogies, from the best case that supports his side, to the new case. The opponent then responds by distinguishing the cited precedent or providing counter-examples. The first side responds again to create a 3-ply argument. The argumentation moves in HYPO are:
   - Analogising a problem to a past case with a favourable outcome.
   - Distinguishing a case with an unfavourable outcome.
   - Citing a more on-point counter-example to a case cited by an opponent.
   - Citing an as-on-point counter-example.

6. Modify the current fact situation hypothetically to strengthen or weaken a specific dimension, and generate a 3-ply argument for the selected hypothetical. This involves: moving a case along a related dimension; making a case into a near-miss of that dimension; making the case extreme along a dimension; making a near-miss dimension apply; or strengthening or weakening a case along an applicable dimension.

7. Explain by illustrating arguments and comparing arguments for the new case and selected hypothetical cases.

Example: In HYPO there are a number of dimensions associated with the Trade Secrets domain. For example, HYPO has a dimension *Security-Measures* that has *none* as the extreme pro-defendant position and then steps through a series of more rigorous measures until the extreme pro-plaintiff position is reached. Another dimension is *Disclosures-to-Outsiders*, which ranges from the extreme pro-plaintiff position of *none* through increasing numbers of disclosures to the extreme pro-defendant point, where the information is in the public domain.

In addition to the development of the first case-based reasoning system in the legal domain, a significant contribution by HYPO concerns the form of the three-ply argument structure; as will be seen in the dialogues discussion in section 2, this is similar to the moves by the parties in the legal dialogues examined in this research in Chapter 4.

CATO

This sub-section provides an overview of CATO [10, 11], which was developed from Rissland and Ashley's HYPO. More detailed discussion will be provided in Chapter 5.

Purpose: CATO was not intended to predict or recommend decisions. It was primarily directed at law school students, and was intended to help them form better case-based arguments, in particular to improve their skills in distinguishing cases [10]. As in the second ply of HYPO, a precedent cited for the plaintiff can be distinguished from the current case by drawing attention to pro-plaintiff factors in the precedent missing from the current case or pro-defendant factors in the current case absent from the precedent. Similar considerations are used in the third ply to distinguish precedents cited for the defendant.

Domain: CATO also functions in the domain of US Trade Secret Law, using 148 cases indexed by 26 factors. Each factor favours either the plaintiff or the defendant. Each case in the database contains a list of factors and a squib (a short explanation of the case).

Case Representation: A core idea in CATO was to describe cases in terms of factors, which are legally significant abstractions of patterns of facts found in the cases, and to build these *base-level factors* into a hierarchy of increasing abstraction, moving upwards through intermediate concerns (abstract factors) to issues. Each factor is a specific feature that is either present or not present, in contrast to a dimension, which refers to a range of points. CATO's factor hierarchy consists of 26 base factors, 11 intermediate factors, 5 high level factors (issues) and 50 links. The definition of these factors is given in [10] and is presented in this thesis in Appendix D.

Reasoning process: The *base level factors* in the factor hierarchy are arranged as leaves, contributing to the presence or absence of their parents (*abstract factors*). A child is seen as a reason for the presence or absence of its parent, and the abstract factors are also taken as present or absent. Their role in CATO is that children of the same parent may be substituted for or used to cancel one another when emphasising or downplaying distinctions between cases with different base level factors.

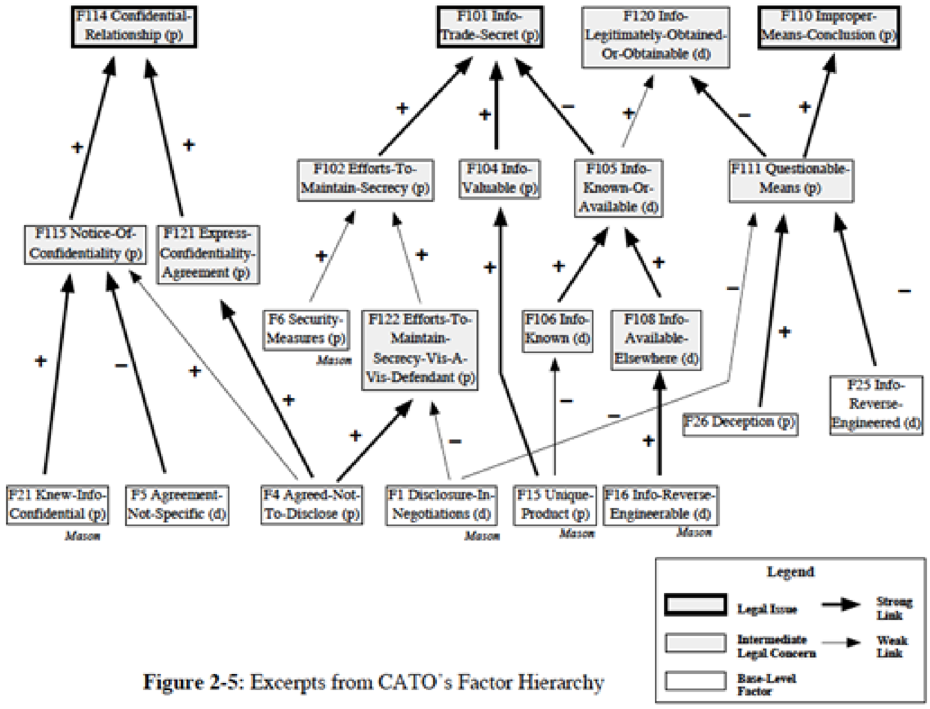


FIGURE 2.4: CATO Abstract Factor Hierarchy From [10]

- Emphasise strengths related to the issue: CATO reasons why such issues matter in terms of abstract factors and cites cases that led to favourable outcomes.
- Downplay weaknesses: points to factors that are closely related in the factor hierarchy and therefore compensate for the weakness.

At the root of the hierarchy is an issue. There may be several layers of abstract factors before the issue is reached. An extract from the factor hierarchy, showing details of the support and attack relationships between the factors, is shown in Figure 2.4. The full factor hierarchy is illustrated in Figure 5.3 in Chapter 5.

The CATO program matches precedent cases with a current case to produce arguments in three plies; this is very similar to HYPO:

- First a precedent with factors in common with the case under consideration is cited, suggesting a finding for one side.
- Then the other side cites precedents with factors in common with the current case but a decision for the other side as counter-examples, and distinguishes the cited precedent by pointing to factors not shared by the precedent and current case.
- Finally the original side rebuts by downplaying distinctions, citing cases to prove that weaknesses are not fatal and distinguishing counter examples.

CATO and HYPO: The factors in CATO can be related to the dimensions of HYPO. For example in CATO there are factors *No-Security-Measures* and *Security-Measures*. These factors divide the *SecurityMeasures* dimension of HYPO into the extreme pro-defendant point of *No-Security-Measures* and all points beyond this (*Security-Measure*s), which suggests that any security measures at all are a point in favour of the plaintiff. Corresponding to *Disclosures-to-Outsiders* there are factors *Secrets-Disclosed-Outsiders* and *Disclosure-In-Public-Forum*. Here the pro-plaintiff factor is not explicit: the advantage to the plaintiff of no disclosures is represented by the absence of the other two factors, which represent factors favourable to the defendant (one stronger than the other, although it is unclear in [10] if this strength manifests itself in resolving issues or is used only in emphasising distinctions).

HYPO and CATO identify but do not resolve conflicts between arguments. The next system, *IBP*, provides a means of adjudicating between conflicting arguments.

IBP: Issue Based Prediction

Brüninghaus and Ashley have adapted CATO for prediction in IBP [55], [17] and [18]. IBP used a logical domain model that relates the various elements of the CATO factor hierarchy.

Purpose: IBP is firmly based on CATO, though the aim is not simply to discover and present arguments, but to predict the outcomes of cases. To enable this, the issues of CATO's hierarchy are tied together using a logical model derived from the *Uniform Trade Secret Act*, which has been adopted by the majority of States in the US, and the *Restatement of Torts*. In the Restatement of Torts:

> One who discloses or uses another's trade secret, without a privilege to do so, is liable to the other if:
> (a) he discovered the secret by improper means,
> (b) his disclosure or use constitutes a breach of confidence reposed in him by the other in disclosing the secret to him,
> (c) he learned the secret from a third person with notice of the facts that it was a secret and that the third person discovered it by improper means or that the third person's disclosure of it was otherwise a breach of his duty to the other,
> or (d) he learned the secret with notice of the facts that it was a secret and that its disclosure was made to him by mistake"

Domain: IBP uses 186 cases from US Trade Secret Law, including the 148 cases analysed by CATO. The additional cases are used as a test set to evaluate predictions.

Case Representation: IBP employs CATO factors in the case representations, and groups these in an abstract factor hierarchy rooted in issues. It also separates factors into 3 groups:

- Knockout (KO) Factors: when present in a case, the case is won by the side it favours.

- Weak Factors: IBP does not allow the issue to be discussed if it is represented only by a weak factor.
- Normal Factors: all the rest.

Reasoning process: IBP identifies the issues raised in a case and determines which party is favoured for each issue. It is an algorithm that combines reasoning with an abstract domain and case-based reasoning techniques to predict the outcome and provide an explanation in an argument-like outline of its reasoning. IBP translates the *Uniform Trade Secret ACT* and the *Restatement of Torts* into a high level logical structure of the domain; see Figure 5.4 in Chapter 5. The domain model captures logical relationships between 5 major issues, each of which is associated with 5 to 7 abstract-level factors from the factor hierarchy.

To predict the outcome of a case, IBP does the following:

- If all the issues favour one party, the issue is decided for that party.
- Otherwise, if there are conflicting factors, IBP uses three case-based reasoning techniques (Theory-Testing, Explain-Away and Broaden-Query) to resolve the conflict.

As part of the evaluation in [55], nine other systems were also considered to provide a comparison. Most of these were different forms of machine learning systems, but programs representing CATO and HYPO were also included. IBP performed successfully among these systems.

The factor hierarchy and analysis in CATO, along with the IBP logical model, have been used in the research in this thesis to instantiate an Abstract Dialectical Framework for the US Trade Secret Law domain as will be shown in Chapter 5 and Chapter 6, and the results obtained are compared with these systems in Chapter 7.

AGATHA

Legal reasoning can be either rule-based, where cases are used as a knowledge source, or case-based, as in the previous systems, where cases are explicitly represented and precedent cases are explicitly deployed to form an argument in the context of a particular legal case. A middle way is where a body of knowledge can be seen as a form of theory, as in [38]. In this approach, cases are pairs of sets of factors and an outcome, and factors are triples of a factor, the value promoted by finding for a particular party when that factor is present, and the party is favoured by that factor. The motivation behind this approach was to ground the rule priorities in preferences for the social values promoted [28]. The idea is that following the various rules promotes the values associated with the factors in their antecedents, and that a preference for rules can be justified in terms of a preference for the values promoted by following them. The precedents thus enable the value preferences to be identified and these preferences can then be applied to new cases, with a different set of factors relating to the same values. Theories in [38] are five-tuples comprising:

- the set of precedent cases on which the theory is based,

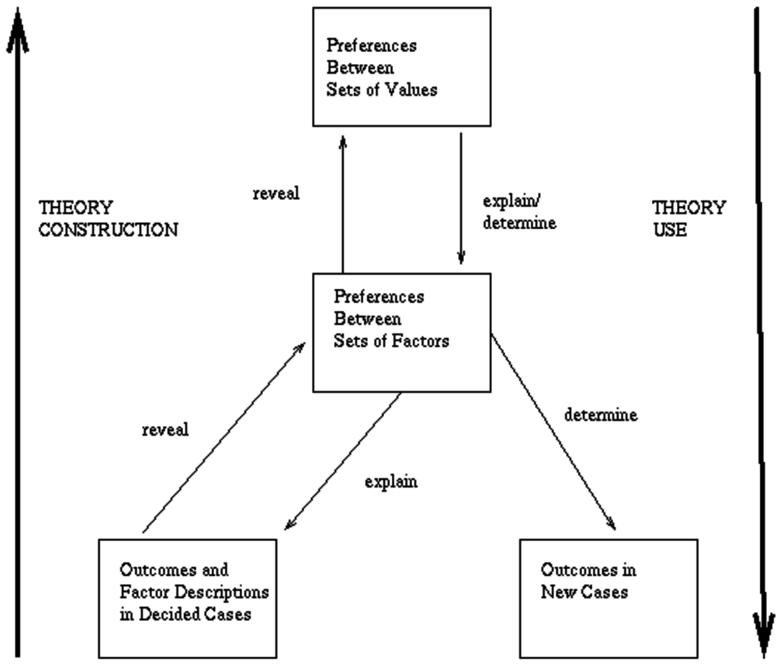


FIGURE 2.5: Construction and Use of Theories From [28]

- the set of factors present in those cases,
- a set of rules linking sets of factors and outcomes,
- a set of preferences over these rules,
- and a set of value preferences.

A set of constructors is also provided so that theories can be constructed on the basis of the available cases. Figure 2.5 illustrates the construction of these theories, the idea being that the disputants would each construct a theory giving the decision to their side, and the best theory (judged using criteria such as explanatory power and simplicity) would win. This approach was embodied and empirically evaluated in the AGATHA (Argument Agent for Theory Automation) system of [57, 58] as seen below:

Purpose: The main objective of the reasoning applied in AGATHA is to construct a case law theory of the domain which explains as many of the existing cases as possible while giving the desired outcome for the current case.

Domain: AGATHA has been tested over the 32 analysed cases available to the public from the US Trade Secrets law. These cases consist of 26 factors and 5 values. It also uses three simple cases from the Wild Animals legal domain, much used in the AI and Law literature [40].

Case Representation: Cases are represented as sets of factors and an outcome, which is either find for the plaintiff or the defendant. Factors are represented by a factor name, an outcome favoured by the presence of that factor, and a value which is the reason why the factor favours that outcome.

Reasoning process: Prior to AGATHA, a Case Theory Editor (CATE) [58] was implemented to facilitate the manual construction of theories and translate them into executable code. From CATE, AGATHA was implemented to explore the automation of theory construction. AGATHA, like HYPO and CATO, simulates case-based dialogue using a set of argument moves, each associated with theory constructors taken from [38]. Plaintiff and defendant play these moves in turn, and each move has the effect of modifying the developing theory. The five moves used in AGATHA are the following:

1. Analogise Case. This is the first move in the dialogue and is used to cite a precedent case which has the outcome desired by the party making the move. The factors in both cases (the cited case and the problem case) are sorted into factors which support that outcome and those factors which support the opposite outcome. A rule preference is made with the supporting factors preferred over the contrary factors.
2. Distinguish with Case, by citing a new case to distinguish the case that has been already cited.
3. Distinguish with Arbitrary Preference. This is similar to the previous move, except that it distinguishes the cited cases with factor preferences for the party making the move.
4. Distinguish Problem. This move distinguishes the current case instead of the previously cited case. If, for example, AGATHA is making a plaintiff move, it creates two rules using the plaintiff and defendant factors. AGATHA takes all the plaintiff (defendant) factors from the problem case and conjoins them as the antecedent into a single rule with plaintiff (defendant) as consequent. The values associated with the factors in these two rules create a value set and a value preference corresponding to the plaintiff factors being preferred over the value set from the defendant factors. Finally a rule preference is created using this value preference.
5. Counter with Case. This move counters the previously cited move with another case that is as-on-point as or more-on-point than that for the other side. The original rule and value preferences supported by the previously cited case are replaced by new preferences which are supported by the new case.

AGATHA uses a variety of heuristic search algorithms (for example A* search) to allow a reasonable number of precedent cases to be available in order to produce effective theories that ensure good precedent cases are chosen. Theories are evaluated according to their simplicity and explanatory power and their value in terms of the theory position in the game tree as shown in the example in Figure 2.6.

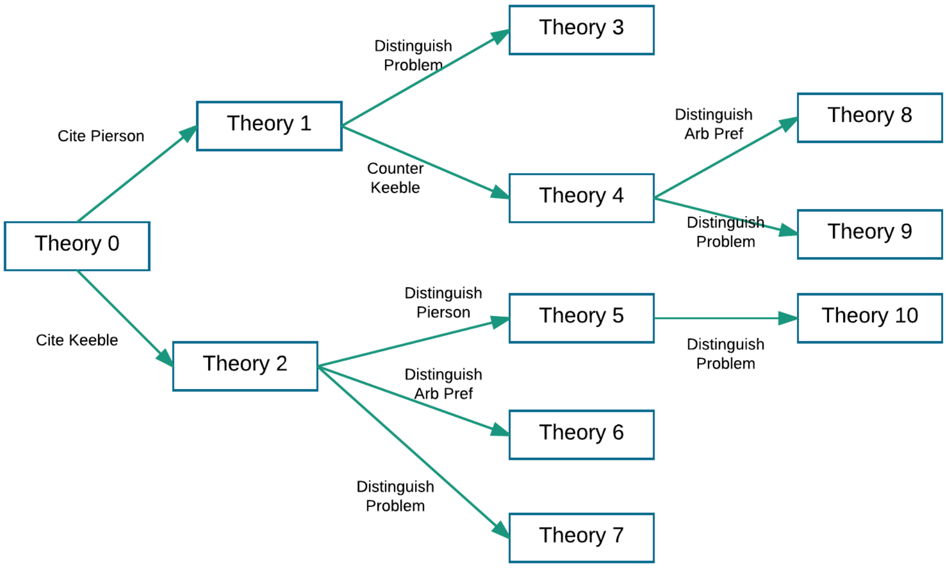


FIGURE 2.6: Example of Produced Theories (reproduced from [57])

Legal Systems	Purpose	Case Representation	CBR Reasoning
HYPO	Create arguments	Dimensions (range of points)	Using dimensions
CATO	Generate issue-based arguments	Factors (one boolean point)	Using factors
IBP	Predict case decision and resolve conflicts	Factors and Issues	Logical model
AGATHA	Generate case law theory (theory refinements)	Factors and Values	Theory comparison

TABLE 2.1: A Summary of Landmark Legal CBR Systems

AGATHA is different from IBP in that it can be used even if there is no accepted structural model, whereas IBP relies on using the structure provided by the Restatements of Torts. Direct comparison between AGATHA and the previous systems is hampered by the fact that evaluation in AGATHA is directed towards evaluating the different heuristics and search algorithms used in that system, and so no version can be considered "definitive", and, of course, many fewer cases were used in the experiments. However, typically 27-30 of the 32 ($\approx 84-93\%$) cases were correctly decided by the theories produced by AGATHA [57].

HYPO, CATO, IBP and AGATHA are summarised in Table 2.1 to distinguish between the systems' goals and the reasoning approach used over the representation of the cases. In terms of encapsulating the analyses of the case law domain, HYPO used dimension points to analyse the cases in order to find the similarity between the current case and precedent cases, CATO introduces a factor hierarchy for the legal domain in which the cases are represented in terms of domain factors. Domain issues are determined in IBP as a logical model that links the top abstract factors of CATO to the leaf issues of the logical model. Figure 2.7 illustrates the legal reasoning stage corresponds to each legal CBR system. As indicated by the figure, HYPO and CATO are not aimed to decide a case but to create arguments about precedent cases. None of the programs represented the domain independently of the program code they embody rather than encapsulated their domain theory.

These systems are considered the foundation of the work presented in this thesis. However, the methodology presented here provides an implementation independent record of the domain knowledge, forming a tree that runs from base level factors at the bottom (the leaf nodes) to a verdict at the top (the root node) passing through issues using an Abstract dialectical framework which can also provide a formal basis for factor-based reasoning that was lacking in previous legal CBR systems.

Unlike previous systems, all of which used a single domain, to ensure the applicability of the presented methodology, several legal domains from the US Courts, often used in the literature of AI and Law, have been used throughout the thesis.

Other Notable Systems

In addition to the four CBR systems discussed previously, there are other landmark legal CBR systems that took a different direction in their development. The discussion in this section ends with a brief overview of these systems.

TAXMAN: The TAXMAN project of McCarty and Sridharan [98] had as its goal providing a computational means of generating the majority and minority opinions in a landmark Supreme Court Case in US tax law case, *Eisner vs Macomber*, 252 U.S. 189 (1920) to gain insights into legal reasoning approaches. The main contribution from TAXMAN is the recognition that legal argument involves theory construction from a knowledge base. McCarty summarises his position in [97]:

> "The task for a lawyer or a judge in a "hard case" is to construct a theory of the disputed rules that produces the desired legal result, and then to persuade the relevant audience that this theory is preferable to any theories offered by an opponent" (p. 285).

GREBE: This program was introduced by Branting in [47]. GREBE represents cases as a semantic network, giving a very fine grained representation of cases and their facts. The reasoning process in GREBE requires finding a pattern of relationships in the new case that corresponds to facts in the past cases. Later in [48] Branting found that matching in case-based reasoning can be improved by comparing new cases to portions of precedents. A new case may match the facts of *some* of the precedent case more strongly than the *entire* set of each precedent case. The use of portions of precedents is discussed further in evaluating the ADF approach in Chapter 7.

CABARET: Skalak and Rissland describe CABARET [121] as a development of HYPO. It models the Home Office Deduction in US Tax Law, as derived from statute. CABARET generates skeletal arguments using a control strategy, incorporating a top-down process (like CATO) that specifies the ideal case independent of the case present in the case base, and a bottom-up process (like HYPO) that generates the arguments based on the cases in the case base.

BankXX: BankXX [117] was developed by Rissland, Skalak and Friedman and evaluated in [118]. It combines both heuristics and argumentation. BankXX creates arguments using the knowledge base, which is a semantic network of argument components of legal cases and legal theories represented by nodes and links between them, and uses a heuristic best first search to build the arguments. This process involves analysing the new case and creating a claim lattice, as in HYPO [14]. BankXX chooses one of the on-point cases randomly as a starting point and places it onto an open list. This open list contains all the nodes that have been harvested during the search.

The summaries of the legal reasoning systems discussed so far are concerned with representing cases as factors, combined sometimes with issues or values. This is related to a stage in legal reasoning known as *intermediate concepts*; various models of reasoning are applied to determine the legal consequence from these intermediate concepts. These can be seen as layers in legal reasoning as proposed in the next sub-section.

2.1.2 Models for Legal Reasoning

Modelling reasoning in the legal domain has been the central question for many researchers in the domain of AI and Law since [98] and [68]. Over the years a picture of reasoning has evolved, which can be seen as a series of reasoning steps. The current consensus in AI and Law (expressed in several contributions to the overview of the field provided in [31]) is that reasoning about legal cases passes through four stages, as in Figure 2.7:

1. from evidence to agreed facts,
2. from agreed facts to base factors; intermediate concepts of legal significance which abstract from the case facts to facilitate comparison of cases,
3. from base factors to higher level factors and issues, and
4. from issues to legal consequences.

Evidence

Every case begins with evidence. Evidence may be in conflict, contain gaps or lack plausibility. Moreover, it may be unclear which inferences should properly be drawn from the evidence, and how it should be interpreted in terms of the governing legislation. The evidence itself may take many forms, including the testimony of eyewitnesses, expert testimony, physical evidence and perhaps even video evidence (as in *Popov v Hayashi* [30]). The evidence itself is certain: in *Pierson v. Post*, that Post testified that he was within 100 yards of the fox is a matter of record and a transcript of the testimony is available in case of doubt. Whether Post is believed or not

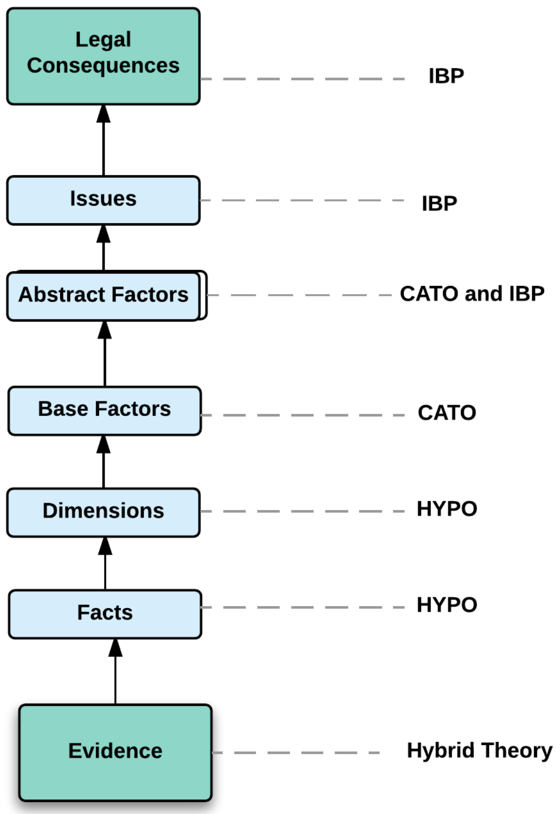


FIGURE 2.7: Legal Reasoning Stages

is a different matter, especially if other witnesses testify that he was a good furlong behind the fox. The methods used to draw inferences from the evidence to beliefs as to what was the case are as many and various as the evidence presented. Witnesses may be more or less personally credible and may have been better or worse placed to view the events. Expert witnesses will disagree. Statistics must be interpreted. In principle any form of standard reasoning might be called into play here. It is, of course, not specifically legal reasoning and very often the evidence is not assessed by judges but by juries composed of lay people who are supposed to be as good as anyone in deciding what is true on the basis of testimony.

Evidence forms the leaves of the tree of reasoning. On the basis of this evidence, a set of facts is established. The process may appear to be akin to forward chaining: a lot of testimony is presented and the facts of the case are deduced from this body of information. In fact, however, the process is far more structured and more akin to backward chaining. Evidence is not random, but elicited by the counsels for the plaintiff and the defendant in order to resolve the issues in favour of their clients. The move from evidence to an accepted body of facts is discussed next.

Fact

Next, facts must be determined on the basis of evidence. Facts are descriptions of states or events, the truth of which is currently unknown and has to be proven. Facts and evidence should not be confused [42]: if evidential data exists it does not guarantee the truth of the fact evidenced (e.g. the witness might lie).

Therefore, the first stage - the move from evidence to facts -requires a rather different style of reasoning (as is evidenced in that facts are often determined by lay people (juries) and are typically not capable of challenge at the appeal stage). This is the topic of [44], and this stage has been explored using argumentation that focuses on arguments based on evidence, stories or narratives that use hypothetical stories or scenarios to explain the evidence, and hybrids of the two [42]. The idea behind the hybrid argumentative-narrative theory is to decide which fact to accept from both arguments and stories. Stories explain "what happened" by organising a set of case facts into one or more hypotheses, while arguments support or attack the facts in these hypothetical stories according to the defeasible inferences based on evidence. The accepted facts are determined from the acceptability of the stories in light of the evidential arguments in the case. The hybrid argumentative-narrative theory has been extended in [46] to show the role of legal stories in finding the interaction between factual reasoning (a proof for what happened in a case) and legal reasoning (making a decision based on the proof). Moreover, a development of the hybrid theory is the integrated theory [43] which employs stories (causal arguments) and evidential arguments in one reasoning account, and [32] which consider the role of dimension at this stage of reasoning.

In addition to the hybrid approach, reasoning with evidence has been considered by several projects in AI and Law, including the use of probability and Bayesian reasoning (e.g. [86]) and combinations of these by using a support graph as an intermediate structure between Bayesian networks and argumentation models to reconstruct different argumentative results about the case (e.g. [123]).

From another perspective, each dimension in HYPO represents the stereotypical facts of legal cases, whilst the facts of a particular case represented on particular points on these dimensions. Thus dimensions provide a way of determining what are the *relevant* features of the case. Other work concerned with this stage includes Gordon's Pleadings Game [70], which identifies the facts that are agreed by the parties and those that will require resolution in the trial itself.

The second stage, the move from accepted facts to factors, was recently explored in [24], reusing the notion of dimensions. In that paper facts are written as dimension points, and the dimension points are used to drive the factor-based reasoning, seen as a set of argumentation schemes.

Intermediate Concepts

Now these facts must be used to ascribe legally significant predicates to the case. These predicates, which serve as intermediaries between the world of fact and the world of law (as discussed in e.g. [50]), have been termed intermediate concepts [89], [17], but are more often called *factors*, following the highly influential systems HYPO [14] and CATO [10], and the high level factors and issues in IBP [18, 55].

Factors

The notion of factors was established in Ashley and Aleven's CATO. Factors are stereotypical patterns of fact, present or absent in a case, and favour one or other of the parties in the case. CATO did not consider the role of facts and dimensions, instead representing cases as bundles of factors from the outset. The factors are then used to establish the legal conclusions (see e.g. [89], [83]). The use of factors effectively addresses the third stage of the legal reasoning process enumerated above, by using the base level factors to infer more abstract high level factors, which are then used to establish the legal conclusions (fourth stage) (see e.g. [83, 89]).

Factors provide a level of abstraction that allows the particular facts of a case to be viewed in terms of precedent cases. The level of abstraction also enables the case to be used as a precedent for future cases. Factors are therefore very important for precedential reasoning [83, 108]. Since there are factors which favour the plaintiff and factors which favour the defendant, typically there will be factors favouring both parties in the case. A further development can be found in [114] which identifies a different kind of precedent - the framework precedent - which requires a different treatment, since it establishes a number of issues, which will be discussed further below.

To this end CATO organises the factors (base level factors) into an abstract factor hierarchy. This hierarchy has abstract factors as its non-leaf nodes: the children of these nodes are reasons to consider them present or absent, depending on whether they favour the same party or the other side. There are potentially several layers of abstract factors before reaching the leaf nodes (the original base-level factors). This hierarchy is used to play down distinctions by substituting factors with the same parent favouring the same side, and canceling factors with the same parent favouring different sides. At the root of the hierarchy is an issue. There may be several layers of abstract factors, before the issue is reached.

Since CATO, factors have received much more attention than dimensions: the fact that they are either present or absent, and always favour the same side, greatly facilitates their use in investigating the logic of factor-based reasoning. The investigation of the logic of factor-based reasoning was significantly advanced by [108], which demonstrated how factor-based reasoning with precedents could be represented as rules. Essentially each precedent gives rise to three rules. Suppose that the precedent can be described as $P \cap D$ where P is the set of pro-plaintiff factors and D is the set of pro-defendant factors. The three rules are :

R1: $P \rightarrow$ plaintiff.

R2: $D \rightarrow$ defendant.

R3: a priority rule $R1 \succ R2$ or $R2 \succ R1$ depending on the outcome of the precedent.

This approach was further developed in [83], in which R1 is allowed to use a subset of P, so that it is possible to go beyond simple a *fortiori* reasoning.

The work described so far in the intermediate concepts takes cases represented as bundles of factors and considers how they can be used to determine legal consequences. Further developments consider the legal issues stage before deciding the case, as presented next.

Issue

An issue is a point of contention. In CATO there are five root nodes, corresponding to the leaf issues of the logical model of IBP. Since CATO is not predicting a verdict but teaching students how to distinguish cases, and since distinctions are made within issues, there is no need to tie these issues together with the logical model used in IBP. Issues and abstract factors could be eliminated by unfolding into the base level factors, but they are both useful for the purposes of exposition and presentation of the argument.

In IBP the precedents are organised into a set of issues, which are then related logically through the logical model. Thus the leaves of the logical model are resolved using CATO-style case-based reasoning, but the outcome is then deduced on the basis of these values. The relationship between the logical model at the high level and cases used to resolve the leaves of the logical model is similar to the use of statutes and cases in CABARET, well described in Loui's description of CABARET in section 3.4 of [31].

The logical model may come from several sources: from statute, as in [120] and CABARET [121], from a commentary or other summary of the common law as in [55], or where some precedent case explicitly sets out such a model, i.e. as a framework precedent [114].

It might at first sight seem that the logical model takes us little further forward: the model in IBP begins by telling us only that the court can find that a trade secret was misappropriated if and only if there was a trade secret and it was misappropriated. However, going down a level, there is more detail of how to establish the existence of a trade secret (it must be of value, and efforts must have been taken to maintain secrecy), and how to establish misappropriation (either by showing that the information was obtained improperly, or that a confidential relationship was breached by the use of the information). Thus the logical model tells us precisely what arguments can be presented, what *issues* are relevant, how a case should be presented in terms of these issues, and the consequences of resolving these issues in particular ways. Issues are thus addressed in the last stage identified at the beginning of this section.

The verdict is logically entailed by the resolution of the issues forming the logical model. As such, from a logical viewpoint, the question can be unfolded into the leaves of the logical model of issues. Thus, in the IBP model of US Trade Secrets Law, the question of whether a Trade Secret was misappropriated unfolds into

$$(\text{IV} \wedge \text{MS}) \wedge ((\text{IU} \wedge \text{CR}) \vee \text{QM})$$

where IV is Information-Valuable, MS is Maintain-Secrecy, IU is Information-Used, CR is Confidential-Relationship and QM is Questionable-Means.

Although logically unnecessary, the intermediate issues above the layer of leaf issues are useful for exposition and work in the same ways as the brackets in the logical expression, to provide a decomposition into subtasks which show how to tackle the problem, and how to present the solution.

Rigoni's example of a framework precedent in [114], *Lemon v. Kurtzman*[1] states

[1] 403 U.S. 602 (1971).

> Three such tests may be gleaned from our cases. First, the statute must have a secular legislative purpose; second, its principal or primary effect must be one that neither advances nor inhibits religion; and finally, the statute must not foster "an excessive government entanglement with religion." *Walz v Tax Commission* at 397 U. S. 674.

This explicitly sets out three issues, all of which must be resolved in favour of the plaintiff if the case is to be found for the plaintiff. Once the issues have been resolved, the verdict can be justified in purely deductive terms: it is in the resolution of the issues that the distinctively legal argumentation is encountered.

Although the preceding stages may suffice to decide a particular case, some work moves a step further and reasons about what the decisions tell us about the purposes of the law [40], or the social values promoted by the law [38]. This results in a theory of the relevant case law, intended to generalise the previous decisions, and in some especially difficult cases to justify the decision. These are the *landmark* cases which represent a shift in the law, as discussed with regard to AGATHA in the previous section, where values were attached to every factor to provide reasons for factor preferences. Whether or not the factors sufficiently favour the plaintiff can be decided by looking at precedent cases, or in the absence of suitable precedents, values (cf [106]: any value preferences may themselves be justified by precedents, as in [28]) or by commentaries.

Verdict, Decision, Opinion

At the top level there is the *verdict* of the court. The verdict can be regarded as a performative utterance in the sense of [26]. It is not true or false: a pronouncement by the appropriate person that he or she finds for the plaintiff is simply how a case is decided for the plaintiff: the pronouncement of the verdict makes it so. As such, these statements have assertability (sometimes called *felicity*) conditions rather than acceptance conditions. Of course, coming from an AI system, the verdict cannot be performative, and rather corresponds to a recommendation to the court or a prediction of what the court will do. This latter, which follows the IBP system [55], is perhaps the best way to categorise the aims of such systems. As explained in the previous section, not all AI and Law systems have sought to make recommendations or predictions: HYPO [14], the system from which so much work in AI and Law on arguing with cases stems, was designed to find, but not to evaluate, arguments and its child CATO [10] was specifically designed to instruct law students in a particular kind of argument, namely distinguishing cases, and emphasising or downplaying these distinctions.

The verdict is binary: the verdict must be "yea or nay": the judge cannot refuse to answer the question, nor express uncertainty or doubt as to the verdict. Leave to appeal may be given, but the verdict remains in force until it is overturned or quashed on appeal. The court is required to justify the verdict. Justification is in terms of issues. The relationship between issues and verdict, is, as agreed by all of [121], [55] and [114], deductive: the verdict is expected to follow logically once the issues have been resolved.

Perhaps the purest treatment of the deductive relationship between verdict and issues is found in the so-called *logical models of legislation systems*, most sharply presented in [120]. In those systems there is only the logical model: everything below what is explicit in legislation is left to the user for resolution. An example of a logical model is shown in Figure 5.4, taken from [55], based on the *Restatement of Torts*.

The Court Opinion is more than a case verdict. In addition to the legal decision, the court opinion provides an explanation for the justification that yields this decision showing the majority, minority (dissenting) and concurring opinions. In some opinions the justification involves providing a test to resolve a new factual justification arising from the case (landmark case) and provides a means of deciding for future cases; this is an essential part of what the court is trying to achieve.

Thus a complete argument for a case will comprise a view on what can be considered as evidence for relevant facts: what facts are required to establish the presence of various factors, and how they relate; how the factors can be used to determine the issues; and, where issues and values conflict, how these conflicts should be resolved.

In the lower courts there will be real items of evidence, particular witness testimonies and the like. But by the time a case reaches the Supreme Court, the facts are usually considered established and beyond challenge. The Supreme Court does, however, need to consider what should count as evidence, and whether this will generally be available, so that the rule can be applied in future cases. This research work investigates the Oral Hearings stage (Chapter 4) and legal decisions (Chapter 5, Chapter 6, Chapter 7) in the context of the Supreme Court process, to provide a basis for the representation of the relevant argument components (facts, factors and conflict issues).

This concludes the discussion of legal reasoning models that have been constructed from the previous work in the domain of AI and law. This discussion has highlighted the particular aspects of the reasoning layers that are applicable to the ideas presented in this thesis. More detailed accounts of argumentation in legal systems can be found in the surveys conducted by Bench-capon et.al. [31] and Prakken and Sartor [110]. The next section investigates the role of dialogues, in particular argumentation-based dialogue models in the legal domain. Prior to this, it is useful to briefly present another important research area in the AI and Law that concerns processing of legal texts and defining legal ontologies.

2.1.3 Legal Text Processing

The analysis of the text in legal case decisions and factor extractions in previous legal reasoning systems, such as HYPO and CATO, have all been conducted manually; see for example the factor extractions in CATO by students from a Law school (Figure 2.8). In general this is an intensive, time consuming and error-prone task. For this reason, providing automated support tools to annotate the unstructured linguistic information has become an important research topic to address the knowledge acquisition bottleneck with textual analysis, which is the problem of getting the textual data into mark-up form for conceptual retrieval.

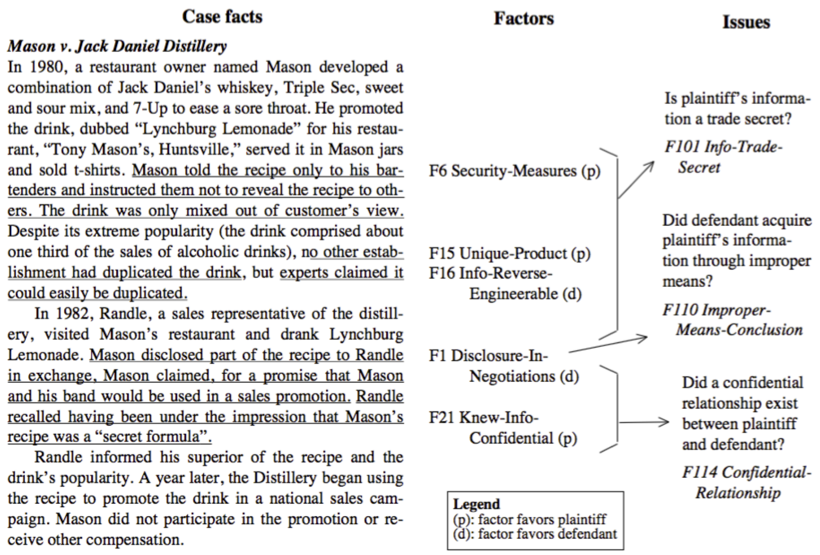


FIGURE 2.8: Factor Extraction in CATO [10]

Several methods and mediated systems have been developed to address these difficulties, starting with automated marking-up tools and progressing to the use of more powerful text analytic tools as in Flexlaw Legal Text Management System [122] and SALOMON [125]. In the argumentation domain, the *Araucaria* system [113] supports manual argument analysis by organising text taken from a complex argument into an inference tree, using the argument markup language (AML). AML is defined in XML (eXtensible Markup Language). To describe argument structure, it uses the original text tags to indicate the evolving structure and to establish a relationship between the text and the inference tree. Araucaria has been tested on an excerpt from an extended argument taken from a US Supreme Court case. The tool follows the assumption that any text excerpt can be analysed in different ways depending on a variety of analytical options, which lead in turn to alternative proposals.

SMILE (SMart Index Learner) is a computer program developed by Ashley and Brüninghaus in [18] to bridge case-based reasoning and extract information from texts. SMILE + IBP represents case text for the purpose of automated text classification. SMILE applies natural language processing (NLP) techniques to a squib, which is a manually constructed summary of the case decisions that represents the factors of the case along with factor indices. From a set of squibs, a learning set is constructed from a list of statements of each factor, and machine learning techniques are applied to acquire a classifier (a pattern) for each factor. The classifiers are applied to the (test set) of squibs, and each text is classified as to the factors contained within it. A nearest-neighbour machine learning algorithm is applied to a learning set of squibs, where the classifying pattern is compared to sentences in the test set to find sentences most similar to the classifying pattern. The success of the classification is measured against a gold standard of squibs, which have been manually classified. Squibs are further preprocessed using a range of NLP techniques. The three representations used for each sentence are:

- Bag of words (BOW) - the degree to which one squib is similar to another squib in terms of the lexical items in each.
- Role replaced (RP) - the name of an individual is replaced by their role in the case, e.g. IBM for plaintiff.
- Propositional patterns (Prop) - part of speech patterns.

The results in Ashley and Brüninghaus [18] report that the accuracy and completeness of SMILE's classification tasks was insufficient (F_measure around 0.70) in terms of realising the goal of identifying factors automatically from a legal text. Note also that SMILE does not address the knowledge bottleneck at the point of identifying and annotating the factors from unstructured text.

In contrast, Wyner and Peters [140, 141] presented another approach for text annotation to support the identification and extraction of legal case factors that is rule-based, following a bottom-up, knowledge-heavy strategy and using the General Architecture for Text Engineering system (GATE)[2], which is an open-source framework for language engineering applications. The approach uses original, unstructured text to annotate factors, rather than structured text in squibs which classify cases with respect factors. It involves selecting salient lexical items and identifying and extracting high-level components of rules from regulations using NLP tools. Some of the results are promising, while others are less so.

Their approach applied a case study in [142] to produce a gold standard corpus of annotated texts in collaboration with law school students, using an online tool to annotate a corpus of legal cases for a variety of annotation types, e.g. citation indices, legal facts, rationale, judgment, cause of action, and others.

In addition to the use of Semantic Web technologies such as XML, a number of interesting contributions have used the power of ontologies extensively in the domain of AI and Law to analyse legal texts and provide support for legal reasoning, as well as making the law available to the public. Legal ontologies are discussed further in the next sub-section.

2.1.4 Legal Ontology

An ontology as defined by [143] as "an explicit, formal, and general specification of a conceptualisation of the properties of and relations between objects in a given domain". Ontologies were introduced by Gruber in [77] and have long been a feature of AI and law (see e.g. [126, 129] and [128]).

The role of an ontology cannot be neglected in the current development of legal case-based reasoning systems. Ashley considers its importance in [16] and distills the ontological requirements for modelling case-based arguments. General and specific roles of ontologies have been determined in [16] based on the purposes of the CBR system.

1. Exchange and re-use of knowledge and information among knowledge bases and other resources on the Internet.

[2]URL: https://gate.ac.uk/

2. Make assumptions about concepts explicit so that the program can reason with them and manage relations and distinctions among concept types.

3. Generate natural language explanations to help in drawing inferences and provide comparisons to relevant cases.

Moreover, three more specific roles for the ontology have also been defined: (i) to provide support for case comparison by drawing inferences to find relevant cases and generate arguments about how to decide a problem; (ii) to distinguish deep and shallow analogies to allow more abstract matching between the cases; and (iii) to induce defensible hypotheses from a database about how to decide a problem using hypothetical reasoning.

A legal CBR ontology is used for representing the case, i.e. in addition to the case name, parties, and decision, the case facts can be used as concepts to determine the factors and enable case comparisons by finding the similarities and differences between cases. Using such representation, ontology can provide explanations for the decisions of cases and the inferences drawn from the comparisons. Finally, ontologies can be used to represent case-based argument by capturing typical, schematic, domain-specific inferences. For example, Wyner and Hoekstra [139] proposed a legal ontology that is relevant to annotating texts in legal cases to extract the main ontological elements. These elements are related to abstract concepts in the domain, which can be added to the structure of an argumentation scheme, factor hierarchy, legal theories and rules.

One of the most important legal ontologies, which addresses the role of exchanging knowledge bases, is the Legal Knowledge Interchange Format (LKIF) which is an OWL ontology of legal concepts, allowing legal knowledge bases to be represented in OWL. The main roles of LKIF as given in [82] are to translate legal knowledge bases written in different representation formats and to provide a knowledge representation formalism. In [82] a legal core ontology is defined to play a role in the translation of existing legal knowledge bases to other representation formats, in particular into LKIF as the basis for articulate knowledge serving.

To this end, there are a number of other alternative approaches and developments that target automatic text annotation in general and factor annotation in particular for unstructured linguistic information. However, this is not the main focus of the research presented in this thesis. For this reason, a manual analysis was applied in this research to mark-up the dialogues in the US Supreme Court Oral Hearings and use the elements to feed a legal ontology which serves as a repository for the argument components of a legal domain as shown in Chapter 4. Dialogues and dialogues models are explained in the next section.

2.2 Dialogues

A dialogue is an exchange of speech acts among a number of participants, in some sequence, aimed at achieving a collective goal. Austin [26] defines a speech act as a performative utterance with a variety of forces: locution, illocution and perlocution. Searle [119] focuses on the illocutionary force of speech acts and clarifies them as: representatives (e.g. reciting a creed), directives (e.g. requests, commands and advice), commissives (e.g. promises and oaths), expressives (e.g. congratulations, excuses and thanks) and declarations (e.g. pronouncing someone guilty). Not all the speech acts are explicit. Searle introduced the notion of “indirect” speech acts, when the participants can employ implicit inferences based on sharing background information. The terminology of speech acts was used to define the semantics of utterances in KQML [65] and FIPA [66].

This section explores the different types of dialogues and shows why it is important to distinguish between these types to define the required components of a dialogue model that fulfills the goals of the dialogue. The following sub-sections examine these topics and provide examples of various legal dialogue systems.

2.2.1 Dialogue Typology

A number of distinct dialogue types used in human communication have been identified by Walton and Krabbe [133] as follows: Persuasion, Negotiation, Inquiry, Information-Seeking, Deliberation, and Eristic Dialogues[3]. Table 2.2 explains each dialogue based on the following characteristics:

- The dialogue initial situation, which identifies the initial conditions that give rise to the dialogue.
- The overall collective goal, shared by all participants, which defines the characteristics of a successful dialogue outcome.
- The individual goals of the participants, which help to determine the reasons for particular utterances by the participants, which should lead towards the main goal, while at the same time respecting their own best interests.

The dialogue descriptions are summarised according to [133] as follows:

- A Persuasion dialogue occurs when one participant endorses some proposition or statement to persuade another participant to accept his statement. If the participants are guided only by the force of argument, then whichever participant has the more convincing argument, taking into account the burden of proof, should be able to persuade the other to endorse the statement at issue, within finite time.
- A Negotiation dialogue happens when two or more parties attempt to jointly divide some resource, where the competing claims to this resource cannot all be satisfied simultaneously. Negotiations require some level of cooperation between the involved parties, but, at the same time, each participant is assumed to be seeking to achieve the best possible deal for him or herself. If a negotiation dialogue terminates with an agreement, then the resource has been divided in a manner acceptable to all participants.
- An Inquiry dialogue involves two or more participants, each being ignorant of the answer to some question, and each believing the others to be ignorant also, jointly seek to

[3]This list is not exhaustive e.g. consider the examination of dialogue [62].

Type	Initial Situation	Main Goal	Participants Aims
Persuasion	Conflicting view points	Resolution of conflicts by verbal means	Persuade the other(s)
Negotiation	Conflict of interest and need for cooperation	Making a deal	Get the best out of it for oneself
Inquiry	General ignorance	Growth of knowledge and agreement	Find a proof or destroy one
Info-seeking	Personal ignorance	Spreading knowledge and revealing positions	Gain, pass on, show or hide personal knowledge
Deliberation	Need for action	Reach a decision	Influence the outcome
Eristic	Conflict and antagonism	Reaching and accommodation in a relationship	Strike the other party and win in the eyes of onlookers

TABLE 2.2: The Six Types of Dialogues (reproduced from Walton and Krabbe [133])

determine the answer. Inquiry dialogue does not start from a position of conflict, since the participants have not taken a particular position on the issue at question; but they are trying to find out some knowledge.

- An Information-Seeking dialogue occurs when one participant does not know the answer to some question, and believes (perhaps erroneously) that another party does so. The first party seeks to obtain the answer from the second by means of the dialogue.
- A Deliberation dialogue involves two or more parties attempt to agree on an action, or a course of action, in some situation. The action may be performed by one or more the parties in the dialogue or by others not present. When the participants are deliberating, they share a responsibility to decide the action(s) to be undertaken in the circumstances, or, at least, they share a willingness to discuss whether they have such a shared responsibility. As with negotiation dialogues, if a deliberation dialogue terminates with an agreement, then the participants have decided on an acceptable course of action.
- An Eristic dialogue occurs when the participants quarrel verbally aiming to vent grievances, the dialogue may act as a substitute for physical fighting.

Most human dialogues are in fact combinations of these ideal types embedded in a larger interaction. Often, however, a dialogue will contain nested dialogues, of the same or different types, and dialogues may shift, licitly or illicitly, between types as the dialogue proceeds. For example, a legal dispute may contain both persuasion, when each party is trying to provide an argument to persuade the Justices, and deliberation between Justices in a court to decide a case. The most usual type of dialogue to be embedded within persuasion and deliberation is information-seeking. In both persuasion and deliberation, facts need to be established, and if these are unknown to the participants, the information may need to be sought elsewhere. More recently, Atkinson et.al. [25] studied the differences between deliberation and persuasion as shown in the following summary:

1. Deliberation starts from a desire to cooperate, whereas persuasion starts with a conflict.
2. In a deliberation, none of the participants has a commitment, but all seek to find an acceptable course of action. There is, therefore, no burden of proof in a deliberation dialogue, whereas in persuasion, a proponent is committed to a particular statement and required to satisfy the criteria to which the opponents are committed.
3. In deliberation, the roles of all participants are initially the same. In persuasion, the roles are asymmetric: there is a proponent and opponents.
4. In deliberation, the aim is to satisfy some public decision rule constructed and committed to in the course of the dialogue. In persuasion, the participants have a private decision rule, which they are entitled to see satisfied before being persuaded.
5. In deliberation, all parties supply and request information. In persuasion, one party is expected to supply information which the other one request it.

In this sense, dialogue typologies have proved to be of importance in argumentation theory and are applied in computational argumentation in AI for a number of reasons, including:

- By identifying a dialogue as falling under one of the particular types, the participants are aware of the goal they are trying to achieve by engaging in the dialogue interaction.
- The pragmatic meaning of speech acts, e.g., "assert" or "inform", is determined based on the type of the dialogue.
- The dialogue shifts are considered to avoid fallacies (if the shift is illicit) and misunderstandings (if the shift goes unnoticed).

It is important to note that not all the dialogues are primarily argument-based. The main aim of some dialogues is, for example, giving and receiving explanations but not proving something. In these kind of dialogues, such as Information-seeking, there is no burden of proof; when a questioner asks for an explanation, there is an obligation on the part of the other party to provide one. More about dialogues and burden of proof can be found in [131].

The work presented in the next chapter focuses on analysing the Oral Hearing Dialogues according to the features of the dialogue types stated by Walton and Krabbe: this is important to define the representation model required for these dialogues. The next sub-section explores what is needed to define a dialogue model.

2.2.2 Legal Dialogue Models

In each dialogue, participants are involved in a rule-governed interaction known as the *dialogue protocol*. At each point in the dialogue, the participants exchange utterances (or speech acts) known as *dialogue moves* according to a specific dialogue protocol until the dialogue *terminates* when reaching certain conditions. Each move is described by a name with a set of components that fulfill the goal of the move. The protocol rules specify the turn-taking between the dialogue participants, and the effect of the dialogue move on *participants' commitment* in a fair and effective way. Together the moves and protocol form the structure of what is called a *Dialogue model*. Gordon and Walton in [74] defined a dialogue formally as an ordered 3-tuple O,A,C where:

- O is the opening stage,
- A is the argumentation stage,
- and C is the closing stage.

At the opening stage the dialogue type is defined according to the initial situation, and the dialogue participants are determined. Each participant has an individual goal, and the dialogue itself has a collective goal. The dialogue moves through the opening stage toward the closing stage where the dialogue is terminated. During the argumentation stage, the participants exchange a number of dialogue moves according to the applicable protocol.

Each dialogue move is associated with pre- and post-conditions that are important to avoid misunderstandings and breakdown of the dialogue. Pre-conditions are constraints that must be met in order for the speech act to be made; post-conditions are brought about by the speech act being made. These conditions in turn construct a structure for the overall protocol of the dialogue. Prakken classifies dialogue protocols in [104] as: unique- or multiple-move protocol; unique- or multiple-reply protocol; or immediate- or non-immediate-response protocol, depending on the dialogue context and dialogue type that is in progress. For example, arguments in natural language dialogues may leave elements implicit. Participants may postpone their replies, or return to earlier choices, and move to alternatives.

The formal study of dialogue systems for argumentation was initiated by Hamblin [79]. Hamblin built a very simple system for argumentation in dialogue called the WhyBecause System with Questions. At each move in a dialogue, a participant is allowed to say various things (called locutions by Hamblin, but nowadays they are called speech acts) used in a dialogue.

An example of a persuasion dialogue is the well-known game "DC" first described in Mackenzie [95]. DC is a symmetric two-person dialogue game, where each player is committed to various propositions which they either assert or accept to be true or they abandon these commitments. These propositions are stored in, and removed from, a commitment store for each player. Changes to these commitment stores result from the application of post-conditions of certain moves. The goal is to derive a contradiction in the opponent's commitments, forcing retraction.

Many dialogue systems do not have much in common. For this reason, several proposals for formal dialogue games have been presented for most of the atomic dialogue types in the dialogue typology. Prakken, for example, has proposed a number of formal dialogue game protocols to model persuasion dialogues (see [104] for an extensive survey). In [104] Prakken identifies the speech acts typically used in such dialogues as:

- Claim P (assert, statement ...). The speaker asserts that P is the case.
- Why P (challenge, deny, question ...). The speaker challenges that P is the case and asks for reasons why it would be the case.
- Concede P (accept, admit ...). The speaker admits that P is the case.
- Retract P (withdraw, no commitment...). The speaker declares that they are not committed (anymore) to P. Retractions are real retractions if the speaker is committed to the retracted proposition, otherwise it is a mere declaration of non-commitment (e.g. in reply to a question).
- P since S (argue, argument ...). The speaker provides reasons why P is the case. Some protocols do not have this move but require instead that reasons be provided by a claim P or claim S move in reply to a why move (where S is a set of propositions). Also, in some systems the reasons provided for P can have structure (e.g. a proof tree or a deduction).
- Question P (...). The speaker asks another participant's opinion on whether P is the case.

Very similar speech acts are used in deliberation dialogues. Atkinson et.al. [25] have recently defined the pre-and post-conditions of these speech acts for both the deliberation and persuasion dialogues in order to distinguish between the two dialogues, finding the differences largely in the pragmatics of the utterance.

Prakken considers only persuasion dialogues; further research efforts introduce dialogue templates [45] that are equally suited for other types of dialogues. Dialogue templates are schemas that encode the generic structure of utterances and replies in dialogue, where dialogue protocols can be mixed and matched to form different types of protocols. Dialogue templates also lay down the basics for automatically executing dialogue protocols, as they provide the possible moves at each point in a dialogue, and determine whether the player of a move is winning.

Dialogue systems were originally introduced into AI and Law as a way of modelling legal procedures [70], but more recently they have been used to capture the logic of aspects of legal reasoning, such as reasoning with cases (e.g. [107]) or particular legal phenomena such as burden of proof (e.g. [71, 109]), and in consequence have become somewhat stylised and removed from any particular legal dialogue. In general, in conflict resolution dialogues the outcome is not fully determined by the participants' points of view and commitments. A typical example is in legal procedures, where a third party determines the outcome of the case. The following sub-sections give an overview of some examples of dialogues in the legal domain.

Pleadings Game (1994)

Gordon formalised the procedural context of legal argument in *The Pleadings Game* [70]. The Pleadings Game sets the formal foundations for a type of mediation system that can be used by lawyers to support discussions about alternative theories by making sure that the rules of procedure are obeyed and by keeping track of the arguments exchanged and theories constructed.

The specification of the Pleadings Game is given below, following Prakken's [104] models of persuasion dialogues.

Context Legal procedure as persuasion dialogue. This was intended as a normative model of civil pleading in Anglo-American legal systems. The purpose of civil pleading is to identify the conflicting issues that need to be resolved in order to decide the case.

Participants Plaintiff and defendant

Initial State Each game starts with an initial background theory shared by the parties. During the game it is continuously updated with each claim or premise of a party that is conceded by the other party.

Moves The game contains speech acts for conceding and challenging a claim, for stating and conceding arguments, and for challenging challenges of a claim. Three kinds of moves can be made during a dialogue: open moves, which have not yet been replied to; conceded moves, which are the arguments and claims that have been conceded; and denied moves, which are the claims and challenges that have been challenged and the arguments that have been attacked with counterarguments. Move legality is further defined by specific rules for the various speech acts, which are mostly standard.

Commitment The game does not have an explicit notion of commitments, but special protocol conditions enforce the participants' dialogical consistency.

Protocol The protocol is multi-move but unique-reply. At each turn a player must respond in some allowed way to every open move of the other player that is still "relevant" and may reply to any other open move.

Termination and Winning Rules The result of a terminated game is twofold: a list of issues identified during the game (i.e., the claims on which the players disagree), and a winner, if there is one. Winning is defined relative to the background theory constructed during a game. If issues remain, there is no winner and the case must be decided by the court. If no issues remain, then the plaintiff wins if his main claim is defeasibly implied by the final background theory, while the defendant wins otherwise.

More recently, Gordon et.al. have developed Carneades [71], which has also been designed to be embedded in a procedural context. Unlike the Pleadings Game, Carneades does not itself define a dialogue protocol. No roles, speech acts, termination criteria, or procedural rules are defined. Instead Carneades is intended to be a reusable component providing services generally needed when specifying such argumentation protocols. Carneades is explored further in subsection 2.3.3.

DiaLaw (1995)

DiaLaw [90, 92] is a dialogical framework for modelling legal reasoning; it incorporates the notion of propositional commitment. The dialogue model formalises a legal procedure using both rule-based and case-based reasoning. Further discussion, in the same format as the Pleadings Game above, about the persuasion dialogue game in DiaLaw (following [104]) is shown below:

Context DiaLaw is applied to legal disputes, and has since been developed for on-line dispute resolution [91].

Participants Two participants with symmetric dialectical roles are involved in the dialogue game.

initial State The dialogue begins with a claim from one of the players.

Moves The game contains detailed protocol rules for each specific type of move. They can use locutions for *claiming* a proposition and for *questioning*, *accepting* and *withdrawing* a claimed proposition. A claim can also be attacked by claiming its negation. Arguments are constructed implicitly, by making a new claim in reply to a challenge. More arguments are related to the procedural correctness of dialogue moves. A supporting claim is not required to logically imply the supported claim. However, the game does not provide means to attack arguments on their invalidity.

Commitment DiaLaw contains the usual commitment rules for claims, concessions and retractions. The commitments are not logically closed, but several rules render the making, conceding or retraction of claims obligatory depending on what logically follows from one's commitments.

Protocol DiaLaw is for pure persuasion. As for turn taking, each dialogue begins with a claim by one player, and then the turn switches after each move, except in a few cases where surrenders are moved.

Termination and Winning Rules A dialogue terminates if no disagreement remains, i.e., if no commitment from one player is not a commitment by the other. The first player wins if at termination he is still committed to his initial claim; otherwise the second player wins.

Toulmin Dialogue Game (TDG) (1998)

The specification and implementation of this dialogue game is based on the argument schema of Toulmin [124] (See Argumentation scheme in sub-section 3.3.1). This schema has been found effective for the presentation of legal argument [94, 96], as an argument is seen as a set of premises which entail a particular conclusion rather than a formal logic representation as in previous dialogue systems.

The Toulmin scheme presented in Figure 2.9 shows different roles for premises, which give arguments a richer structure, and corresponds to real arguments. Thus the structure proposed by Toulmin incorporates three elements important in legal arguments: their defeasibility, via the rebuttal; their need to appeal to extra logical justifications, through the backing; and the degree of proof required, through the qualifier. The dialogue game based on the toulmin scheme was used for the British Nationality Act in [39].

TDG [34] is a persuasion dialogue game with a set of moves rich enough to model reasonably realistic dialogues. A State Transition Diagram is used to show what moves, and sequences of moves, are possible in TDG. Each move is illustrated by a description, pre- and post-conditions and completion conditions. The richness achieved by distinguishing the premises is essential to capture elements of organisation and context integral to real arguments.

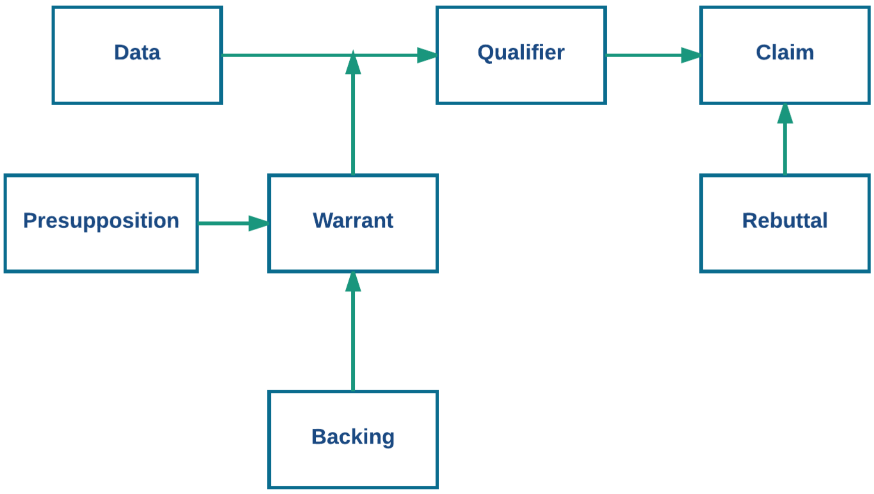


FIGURE 2.9: Toulmin Argumentation Scheme (reproduced from [34])

Context The scheme is applied to give more flexible legal dialogues. The idea is to generate better dialogues by defining the communication language in terms of the argumentation scheme defined by Toulmin.

Participants The game is symmetric for proponent and opponent, in that all the moves can be made by either of these participants, at any given time. One of three roles is undertaken by the participants: proponent of the claim, opponent of the claim, or referee, which enforces implicit commitments.

initial State In TDG there is a claim stack that stores claims on the top of the stack, either explicitly as claims, or implicitly when data or warrant s are supplied. Initially, this claim stack is empty, and dialogue begins when a claim is made by one of the players.

Moves Speech acts are based on an adapted version of Toulmin's argument scheme (Figure 2.9). A *Claim (C) move* is supported by data (*Supply Data(C)*), which can be sought if not provided using (*Why(C)*). The data support is warranted by an inference licence using (*Supply Warrant(C)* or *So(C)* to request a warrant) which possibly has *presuppositions*, and which is *backed* by grounds for its acceptance. Finally, a claim can be attacked with a *rebuttal*, which itself is a claim and thus the starting point of a counterargument. Arguments can be chained by also regarding data as claims, for which further data can be provided. The proponent can also withdraw his commitment to C using (*Withdraw(C)*).

Commitment Each player is committed to the truth of certain propositions as the result of moves. The commitment stores, maintained for each player other than the referee, record these commitments.

Protocol The protocol is multi-move and multi-reply. The proponent starts with a claim and then the parties can reply to each other's moves according to a communication language. To handle this, propositions are added to a "claim stack" when stated and removed from it when retracted or conceded; each move replies to the top claim in the stack, but with (*Switch Focus(C)*) control move, a player can move an earlier claim to the top of the stack and so reply to it. After a surrendering move, or when an argument structure has been completed, the turn switches to the referee. The referee assigns the turn to the opponent of the top of the claim stack using (*Current Claim(C)*) or invites the opponent to make a rebuttal move (*Rebuttal (C)*) .

Termination and Winning Rules The referee terminates the game when all claims have been either conceded or retracted (claim stack is empty). The game has no explicit definition of a winner.

PADUA (2009)

PADUA [135] is a *Protocol for Argumentation Dialogue Using Association Rules*, designed to support two agents debating a classification in a persuasion dialogue by offering arguments based on association rules mined from individual datasets. The participants have debate on the basis of their experience of particular cases stored in a database. Arguing from experience is akin to case-based argument where arguments about a case are typically backed by precedents, therefore the best examples of legal PADUA are in legal domains where simple facts are used, as transcribed from claim form to database, to find factors or similar intermediate predicates. The purpose of PADUA dialogues is the resolution of conflicting opinions about the classification of an instance. This purpose is met when the dialogue is terminated, and is identified with the classification proposed by the winner of the dialogue game. PADUA specification is given below:

Context PADUA is applied in a legal domain that addresses the classification of routine claims for a hypothetical welfare benefit system in the UK.

Participants PADUA has two agents, each with distinct datasets of records relating to a classification problem. These agents produce reasons for and against classifications by mining association rules from their datasets using standard data mining techniques "association rule" which means that the antecedent is a set of reasons for believing the consequent.

initial State PADUA models argument from experience: that is, the agents will have considerable experience of classifying examples in the particular domain and will draw on this experience to offer reasons for classifying the new example. A topic language is defined in PADUA that consists of set of items where each item is a set of value related to the database attributes, a set of database records, and association rules in the form ($premises \rightarrow conclusion, confidence$) according to a specific threshold of confidence (Conf) that represents the lowest acceptable confidence, where rules with confidence lower than this threshold are considered invalid arguments.

Moves The dialogue moves in PADUA are related to the three main argument moves used in the legal CBP systems which involve: citing a case, distinguishing a case and providing counter example. One represents citing a generalization of experience, by proposing a new rule with a confidence higher than the threshold (Conf) (*Propose rule*), three pose the different types of distinction each with different force: either adding some new premise(s) to the proposed rule, such that the confidence of the rule is lower than the confidence threshold (*Distinguish*), or arguing that certain consequences (conclusions) of the proposed rule do not match the studied case (*Unwanted consequences*), or that the confidence in the classification would be increased if some additional features (premises) were present (*Increase confidence*). The (*Counter rule*) move is different in that it poses that there are reasons to think the case untypical by proposing a new rule. Finally, (*Withdraw unwanted consequences*) excludes the unwanted consequences of the rule it previously proposed, while maintaining a certain level of confidence.

Commitment PADUA defines four effect rules that change the commitment store of the player after a dialogue move.

Protocol A protocol that specifies the legal moves at each stage of a dialogue is defined [135]. PADUA protocol applies a simple turn taking policy, in which each player is allowed to play exactly one move in its turn, and the turn shifts to the other agent in the dialogue.

Termination and Winning Rules The dialogue purpose is met when the dialogue is terminated, and is identified with the classification proposed by the winner of the dialogue game i.e. the dialogue ends when a player fails to play a legal move in its turn. In this case, this particular player loses the game while the other player wins it. The winner of a PADUA is the participant whose goal matches the output of the dialogue.

The dialogue moves presented above are defined within a dialogue structure in an argumentation-based dialogue system. These moves are different from the argument moves presented in HYPO, CATO and AGATHA in sub-section 2.1; the goal of the moves in these systems was to provide an approach to a case-based reasoning, but they are not embedded within a dialogue.

In summary, the review of the several works presented in this section shows that dialogue models are helpful in reasoning about exchanges of information by participants in different contexts, including law. The work on this topic has produced some valuable insights. This thesis will return to the original motivation of representing a specific legal procedure, running in a real context, by considering some particular natural dialogues that form a clearly defined stage of the US Supreme Court process, namely the Oral Hearings stage.

The three-ply approach of HYPO is similar to that used in the Oral Hearings of the US Supreme Court. First, the plaintiff cites some cases supporting his side. In the second ply, the defendant can distinguish the cases cited by the plaintiff and cite cases supporting the defendant:

these are counter-examples to those cited in the first ply. In the third ply, the plaintiff is allowed a rebuttal: the counter-examples can be distinguished and further cases advanced to show that weaknesses in the original argument are not fatal, and to emphasise any strengths in the case.

Chapter 4 presents the structure of the dialogue representation model, defining the speech acts and the rules that form the protocol of these Oral Hearing dialogues. The model presented is not a dialogue game; also, there is no argument in these dialogues, no winners or losers, and no burden of proof. However, the key idea of the dialogue is to extract the relevant argument components, not to make a decision. To investigate how these argument components can be represented, the next section discusses various frameworks for abstract and structured argumentation.

2.3 Frameworks For Argumentation

Argumentation Theory is the study of the process of making a decision through logical reasoning, which itself involves different types of reasoning [33]. Contributions in argumentation theory provide a number of mechanisms which are useful in their application to AI. One particularly important contribution is Dung's notion of an "argumentation framework" for the evaluation of the acceptability status of arguments.

This section first gives an overview of Dung's abstract argumentation frameworks, and the various refinements of Dung's frameworks. More detail is given regarding the most recent generalisation of Dung's framework, *Abstract Dialectical Frameworks* which are the core of the work presented in this thesis. Afterwards, the discussion moves from the abstract nature of argument to examine structured argument.

2.3.1 Dung Abstract Argumentation Frameworks and Dung's Refinements

Dung developed a formal abstract theory of argument. A Dung argumentation framework (AF) [61] consists of a set of arguments related by a binary conflict-based attack relationship (Figure 2.10 (a)).

Definition 1: An Argumentation Framework AF is a pair $AF = < A, Def >$[4] where

- A is a set of arguments.
- Def is a binary relation of A. $Def \subset A \times A$ is the attack relationship for AF. A comprises a set of ordered pairs of distinct arguments in Def.

The argumentation process moves through a number of steps: first, identify the arguments and the attack relationship between them; then, evaluate the arguments and their interactions according to one of the several available semantics; and finally, select the most acceptable arguments (or sets of arguments) and draw a conclusion or choose a decision from the accepted

[4]Dung called the relationship "defeat". Subsequent work has led to "attack" being more representative.

arguments. The sets of acceptable arguments are determined under different extensional semantics. For R, S, subsets of A, Dung defines several types of extensions as explained below (reproduced from [21]).

- $s \in S$ is attacked by R if there is some $r \in R$ such that $< r, s > \in A$.
- $x \in X$ is *acceptable* with respect to S if for every $y \in X$ that attacks x, there is some $z \in S$ that attacks y (i.e. z, and hence S, defends x against y).
- S is *conflict free* if no argument in S is attacked by any other argument in S.
- A conflict free set is *admissible* if every argument in S is acceptable with respect to S.
- S is a *preferred extension* if it is a maximal (with respect to set inclusion) admissible subset of X.
- S is a *stable extension* if S is conflict free and every argument y, $\neg(y \in S)$, is attacked by S.
- S is a *complete extension* if S is a subset of A, S is admissible, and each argument which is defended by S is in S.
- S is a *grounded extension* if it is the least (with respect to set inclusion) complete extension.
- An argument x is *credulously accepted* if there is some preferred extension containing it.
- An argument x is *sceptically accepted* if it is a member of every preferred extension.

Each AF has a grounded extension (which may be the empty set) that is contained in all other extensions, and at least one preferred extension.

The framework abstracts from the underlying logic in which the arguments and attack relation are generated. The abstract nature of Dung's theory of argumentation accounts for its widespread application as a general framework for various species of non-monotonic reasoning, and, more generally, reasoning in the presence of conflict, whether such conflict arises from uncertain or incomplete information or as a result of differing opinions or preferences.

Dung's theory has been developed in a number of directions. One is to provide some means for preferring one argument to another as in [12], so that an argument a successfully attacks b if and only if a is not preferred to b according to some given preference relationship defined within the framework. Another form of preference is in *Value Based Argumentation Frameworks* (VAFs) [28] where each argument is associated with a social value that is promoted according to a value preference ordering which may vary from one audience to another. Depending on this process, the argument will be accepted or defeated by other attacks. That is, a is defeated if and only if a value $v1$ promoted by an argument a is ranked higher than the value $v2$ promoted by b as in Figure 2.10 (b). This social value determines the strength of an argument according to a certain preference ordering by an audience.

Modgil's *Extended Argumentation Frameworks* (EAFs) [100] refine Dung's framework in a different way. Modgil allows attacks on attacks in addition to attacks on arguments, assuming that the defeat relationship in AF is a basic attack relationship. EAF arguments express preferences between other arguments using attacks on attacks. Intuitively, if argument c claims that argument b is preferred to argument a, and a attacks b, then c undermines the success of a's attack on b by pref-attacking a's attack on b (Figure 2.10 (c)). This idea has been further generalised by Baroni et al. [27] so that not only can arguments attack attacks, but these attacks on attacks can themselves be attacked in *Argumentation Frameworks with Recursive Attacks* (AFRA). Furthermore, Dung AFs have been used to reason about other arguments rather than about the domain, by using preference ordering and attacks on attacks to determine a unique set of justified arguments; this results in *Meta-argumentation Frameworks* [99] (Figure 2.10 (d)). The arguments defined in all these frameworks can be evaluated under the full range of Dung's extensional semantics.

Further work has applied formal argumentation models to legal argumentation. For example, the abstract argumentation framework of Dung [61] has been applied to law in [103] and many subsequent papers, and an extension to accommodate values [28] was applied to legal argument in [38] and elsewhere. The computational and theoretical strands of factor-based reasoning have, however, by no means remained separate, and there have been a number of attempts to express the factor-based reasoning embodied in systems such as CATO in a rule-based format suitable for representation in a more formal framework (e.g. [108]).

Finally, a number of works augment Dung's framework to include a *support relationship*, in addition to an attack relationship, in arguments such as *Bipolar Abstract Argumentation Frameworks* [56]. BAFs present two independent kinds of information: arguments supporting other arguments, and arguments attacking arguments. In Dung's framework, if an argument a attacks argument b and argument c attacks argument a, then c supports b. This is different from the support relationship in a BAF, which is assumed to be totally independent of the attack relation and thus is not defined using the attack relation. The links in BAFs have been generalised to include any interpretation in *Abstract Dialectical Frameworks*, with acceptance conditions local to each particular node as shown in the next sub-section. Figure 2.10 (E,F) illustrates BAFs and ADFs respectively.

2.3.2 Abstract Dialectical Frameworks

Abstract Dialectical Frameworks (ADFs) were introduced in [54] and revisited in [53]. ADFs provide a generalisation of Dung's abstract argumentation frameworks (AFs) [61]. ADFs, like AFs, consist of a set of nodes and directed links between them, but whereas the links in an AF have a uniform interpretation, namely defeat, the links in an ADF can be given a variety of interpretations including the second relationship (support) in bipolar argumentation frameworks [56]. In addition, ADFs generalise the single acceptance condition introduced by Dung's AF (that all attackers be defeated) with acceptance conditions local to each particular node. The nodes in ADF are statements, not necessarily arguments, that are related by a variety of types of links.

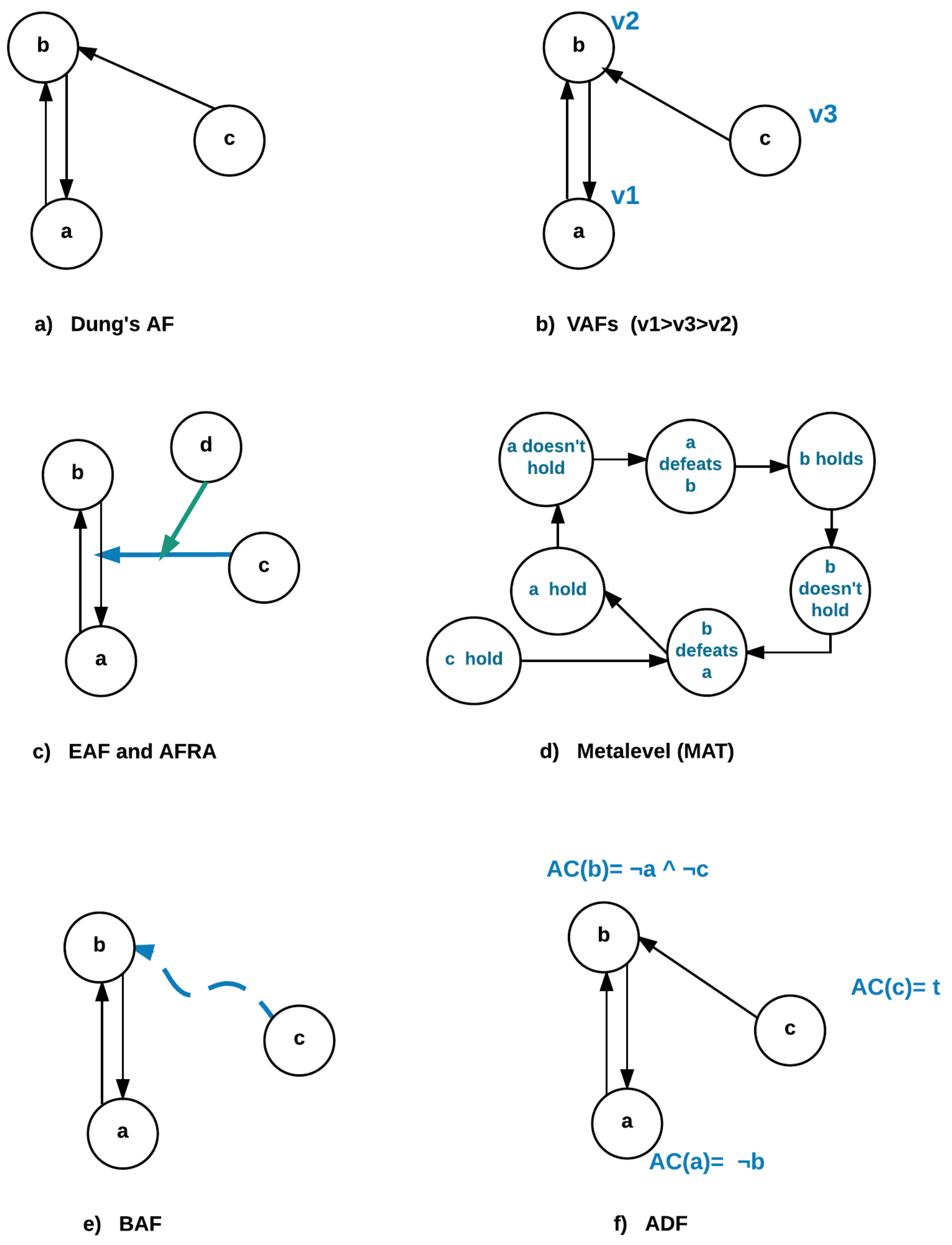


FIGURE 2.10: Abstract Argumentation Frameworks

Although the acceptance conditions are often expressed as propositional functions, this need not be the case: all that is required is the specification of conditions for the acceptance or rejection of a node in terms of the acceptance or rejection of its children. Table 2.3 shows the differences between AFs, bipolar AFs and ADFs.

ADFs are defined in [53] as follows:

Definition 2: An ADF is a tuple $ADF = < S, L, C >$ where

- S is the set of statements (positions, nodes).
- L is a subset of $S \times S$, a set of links.
- $C = \{C_{s \in S}\}$ is a set of total functions $C_s : 2^{par(s)} \to \{t, f\}$, one for each statement s. C_s is called the acceptance condition of s.

In a Prioritised ADF, L is partitioned into $L+$ and $L-$, supporting and attacking links, respectively.

Definition 3: A prioritised ADF (PADF) is a tuple $A = (S, L+, L-, >)$ where

- S is the set of statements (positions, nodes).
- $L+$ and $L-$ are subsets of $S \times S$ the supporting and attacking links.
- $>$ is a strict partial order (irreflexive, transitive, antisymmetric) on S representing preferences among the nodes.

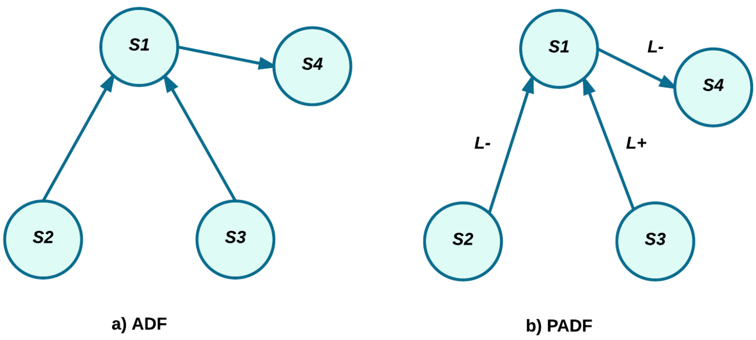


FIGURE 2.11: ADF and PADF

Prioritised ADFs (PADFs), illustrated in Figure 2.11, were designed specifically to reflect Preference-Based Frameworks [12]. This partitioning is used to separate links into supporting and attacking links, but continues to see the acceptance conditions as specifying preferences locally rather than adopting a global ordering of nodes as will be used in Chapters 5 and 6. Brewka and Woltran have also introduced *weighted ADFs* where a weight is added to the links using a weight function; i.e. by assigning qualitative or numerical weights to the links in the ADF; the acceptance conditions will then be defined based on the weights of links rather than on the involved positions.

Different semantics, including Dung's semantics, have been defined for ADFs to evaluate the arguments and determine the acceptable arguments in ADFs [51]. The notion of *a model* in ADF means that all acceptance conditions are satisfied. The Diamond system [63] is a tool for computing various interpretations with respect to the semantics for a given ADF. It takes an ADF as an input and produces different models for different ADF semantics.

The advanced features of ADFs motivate the core work in this thesis. ADFs have taken the design role to drive and record the knowledge base for a system to apply a body of case

	AF	BAF	ADF	PADF
Nodes	Abstract argument	Abstract argument	Any statement	Any statement
Links	Attacks	Supports and attacks	Any type of links	Supports and attacks (L+,L-)
Accepted nodes	A node that defeats all attacks	A node that defeats all attacks	The node's acceptance condition is true	The node's acceptance condition is true according to a particular order

TABLE 2.3: AF, BiPolar and ADF

law taking advantage of the knowledge features embedded in the ADF, as is fully discussed in Chapter 5. Legal decisions are typically not truth functional: they require both a context (e.g. a set of cases) and a procedure (e.g. *stare decisis*). Both of these can be represented in the "acceptance conditions" of ADFs, as in Chapter 5, applied in different legal domains in Chapter 6 and finally evaluated in Chapter 7.

2.3.3 Structured Argumentation

Dung's arguments are entirely abstract, mainly exploring how acceptability of arguments follows from a defeat relationship according to a number of different semantics. However, the structure of the arguments in Dung's frameworks is unknown. How can an argument be constructed from a set of premises by performing consecutive reasoning steps? What are the different ways in which an argument can be attacked? For this internal structure many different formalisms have been proposed. This sub-section discusses such structured approaches, specifically examining the argumentation schemes, ASPIC+ and Carneades.

Argumentation Schemes

Argumentation schemes [132] are stereotypical patterns of human reasoning used in conversational argumentation, and in other contexts as well, including legal and scientific argumentation. They are forms of inference from premises to a conclusion in which one party is trying to get another to come to accept a conclusion that is at issue. They represent deductive and inductive reasoning in some instances but typically they represent defeasible inference rules. These schemes are defined by studying many examples of arguments and finding patterns and structures common to these arguments. They are a powerful tool as a bridge between human and computational models of argument; they are useful in understanding the structure of the arguments and making significant improvements in reasoning systems in AI.

Schemes are useful for identifying arguments by finding missing premises, analysing arguments and finally evaluating them. In addition to the premises and conclusion, a key feature of any argumentation scheme [132] is a set of characteristic Critical Questions (CQs). A set of appropriate CQs is associated with each scheme. These questions represent attacks, challenges or criticisms that, if not answered adequately, make the argument fitting the scheme fail to hold since argumentation schemes establish their conclusions only presumptively; the CQs are needed to test these presumptions. Some critical questions ask whether a premise of the argument is stated to identify a hidden premise, or whether a premise is true. Such questions create a burden on the other side to back the premise with further grounds.

Argumentation schemes are a tool for constructing arguments to put forward in a dialogue. A typical method is to instantiate a scheme from an underlying knowledge base (e.g. Carneades [71]) or modelling logic as in the Action-based Alternating Transition System (ATAS) used in [22]. The relationship of a set of critical questions to a scheme brings in the notion of a dialogue sequence between a questioner and an answerer. An argument is a move in a dialogue, and the scheme that it instantiates determines the allowed and required responses to the move by the other side. For an argument to be justified it is not sufficient for it to fit a recognised argument scheme: it should also survive a dialogical testing process with critical questions.

Legal arguments can be raised from different sources: from evidence, precedent cases, social values, rules, policy goals and jurisprudence doctrine. Two functions of argumentation schemes can be distinguished: argument patterns useful for reconstructing arguments from natural language texts; and methods for generating arguments from argument sources such as legislation or precedent cases. Legal reasoning typically requires a variety of argumentation schemes to be used together. In addition to Toulmin's argumentation scheme [124] (discussed previously in the Toulmin Dialogue Game in section 2.2.2), a leading example of a presumptive argumentation scheme is the one representing argument from expert opinion as formulated in [132]: A is a proposition; E is an Expert; D is a domain of knowledge.

Argument From Expert Opinion

Major Premise: Source E is an expert in the subject domain S containing proposition A.

Minor Premise: E asserts that proposition A in domain S is true.

Conclusion: A may plausibly be taken as true.

The standard six critical questions, given in [132], matching the expert opinion argumentation scheme are:

CQ1: How credible is E as an expert source?

CQ2: Is E an expert in the field that A is in?

CQ3: Does $E's$ testimony imply A?

CQ4: Is E reliable?

CQ5: Is A consistent with the testimony of other experts?

CQ6: Is A supported by evidence?

Another form of argumentation scheme that is used frequently in legal reasoning is witness testimony, which is considered as a basic source of evidence about a fact in a case in trial courts. Witnesses are not always reliable; that is what needs to be clarified through the following scheme taken from [134, p. 310]:

Argument from Witness Testimony

Position to Know Premise: Witness W is in a position to know whether A is true or not.

Truth Telling Premise: Witness W is telling the truth (as W knows it).

Statement Premise: Witness W states that A is true (false).

Conclusion: A may plausibly be taken to be true (false).

Five critical questions are used with this witness testimony argumentation scheme:

CQ1: Is what the witness said internally consistent?

CQ2: Is what the witness said consistent with the known facts of the case (based on evidence apart from what the witness testified to)?

CQ3: Is what the witness said consistent with what other witness have (independently) testified to?

CQ4: Is there some kind of bias that can be attributed to the account given by the witness?

CQ5: How plausible is the statement A asserted by the witness?

Practical reasoning is reasoning about what to do. An argumentation scheme for practical reasoning is introduced in Walton's [132] and elaborated by Atkinson et.al. in [23] by redefining the notion of goal intro three distinct elements: states, goals, and values. This results in the construction of sixteen critical questions presented in [23].

Practical Reasoning (PRAS)

Circumstance premise In the circumstances R

Consequent premise to achieve new circumstances S

Goal premise which will realise some goal G

Value premise which will promote some value V.

Conclusion we should perform action A

Legal reasoning can also be seen as practical reasoning as in [20] by instantiating the argumentation scheme so that the action (A) is to decide for, say, the plaintiff, given the facts (R) and a decision to resolve the case (G) which will promote value (V). This justification has been applied to reconstruct the reasoning of the majority and dissenting opinions of several legal cases, for example *Pierson v. Post*.

Modelling the legal dialogues discussed in Chapter 4 in this thesis uses a set of critical questions that are akin to the CQs attached to an argumentation scheme. However, the conclusions of these dialogues are not arguments but argument components in the form of issues, factors, facts and relationships between them. Argumentation schemes are intended to supply reasons why these components should be considered and tested for their presence in a case. In this sense, the role of these critical questions is to provide moves in the dialogues by providing challenging questions about the argument components, or seeking additional components to be proposed.

ASPIC+

Another refinement of Dung's abstract approach is the ASPIC+ framework for structured argumentation [105]. The framework enables some content to be given to the arguments assuming an unspecified logical language and knowledge base. This may include facts, strict rules, and defeasible rules. The arguments are defined as inference trees, formed by applying inference rules (which may be either strict or defeasible) to a knowledge base.

The root node of the tree is the argument claim, and the leaf nodes are the argument premises. Premises are facts in the knowledge base; they are either assumptions (ordinary premises), which are initially unjustified and are defeated if challenged and cannot be argued for, or axioms, which cannot be questioned [101].

In ASPIC+ an argumentation theory is $AT = (AS, K)$ where

- The argumentation system AS consists of an argumentation language L and a set of strict rules Rs and defeasible rules Rd and a partial function n that corresponds defeasible rules to the language such that $Rd \to L$.
- Knowledge base K consists of two disjoint subsets Kn (the axioms) and Kp (the ordinary premises).

Arguments in ASPIC+ relative to AT are defined as chain applications of the inference rules from AS into inference trees, starting with elements from the knowledge base K. For a given argument,

- the function *Prem* returns all the formulas of K (called premises) used to build the argument,
- *Conc* returns its conclusion, *Sub* returns all its sub-arguments,
- $DefRules$ returns all the defeasible rules of the argument and
- $TopRule$ returns the last inference rule used in the argument.

The notion of an argument as an inference tree leads to three ways of attacking an argument (Figure 2.12):

- Rebuttal (a conclusion attack): A rebuts an argument B (on B') iff $Conc(A) = \neg\varphi$ for some $B' \subset Sub(B)$ of the form $B1''$,,,,$Bn'' \Rightarrow \varphi$.
- Undermining (a premise attack): A undermines an argument B (on B') iff $Conc(A) = \neg\varphi$ for some $B' = \varphi, \varphi \notin Kn$
- Undercutting (an inference attack): A undercuts an argument B (on B') iff $Conc(A) = -n(r)$ for some $B' \subset Sub(B)$ so that B''s top rule r is defeasible.

Rebutting and undermining attacks succeed only if the attacked argument is not stronger than the attacking argument, while undercutting attacks express exceptions to defeasible inference rules and thus succeed as defeats independently of preferences over arguments.

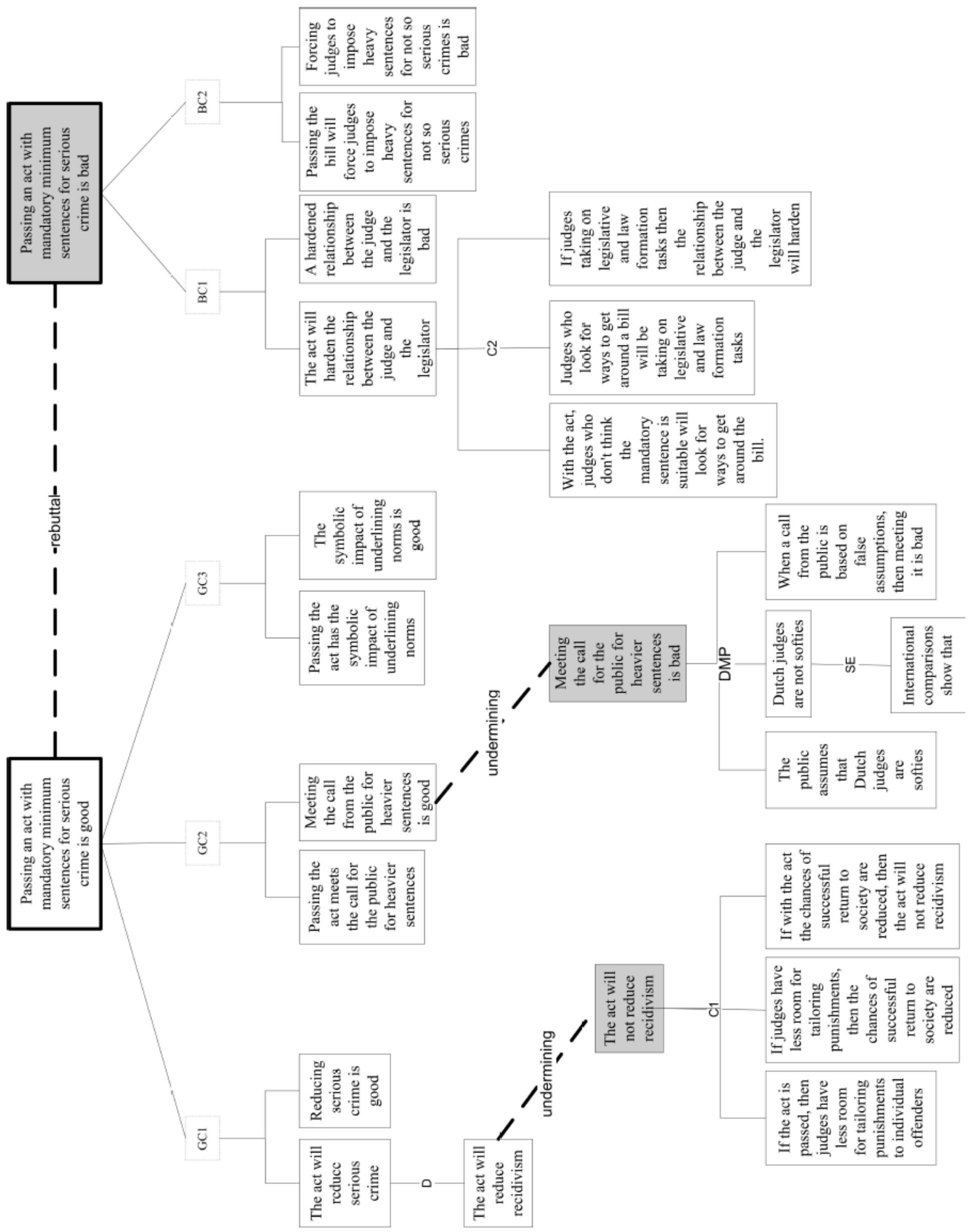


FIGURE 2.12: Formalising a Legal Opinion in ASPIC+ From [106]

The framework has a number of applications in the legal domain. Wyner et.al. [145] provide argumentation schemes for CATO-style case-based reasoning, which have been reanalysed and formalised in ASPIC+ in [111] where defeasible legal case-based reasoning is represented, formalised in argumentation schemes and evaluated in ASPIC+. In [106] ASPIC+ has been used as a tool for reconstructing natural legal argument about legislative proposals in order to present some of the main ASPIC+ features, the distinction between deductive and defeasible inference rules and the explicit preference ordering on arguments.

The analysis of legal cases produced by the ADF representation in Chapters 5 and 6 in this thesis bears a strong resemblance to the kind of structured argumentation found in ASPIC+.

This topic is explored further in Chapter 7.

Carneades

The Carneades system [71] is software developed for constructing, evaluating and visualising arguments from formal models of legal concepts, rules, and cases. Carneades is useful for modelling procedural context as in legal dialogues. It shows how proof standards make it possible to shift the burden of proof during a dialogue from one side to the other as it progresses from stage to stage when a critical question matching an argumentation scheme is asked by one party.

Critical questions are then modelled as additional premises of different types to capture the varying effects on the burden of proof of different kinds of questions. Burden of proof is distributed by distinguishing between three types of premises: those that must always be supported with further grounds (ordinary premises); those that can be assumed until they have been questioned (assumptions); and those that do not hold in the absence of evidence to the contrary (exceptions).

The originality of Carneades lies in its method for allocating the burden of proof to the proponent or respondent, for each premise separately, using premise types, proof standards and the dialectical status of statements (stated, questioned, accepted or rejected). The notion of burden of proof in Carneades[71] refers to: *burden of production*, which means the amount of evidence required to produce the legal facts for the plaintiff's main claim, or producing the exceptions for the defendant, and *burden of persuasion*, which defines the degree to which the fact-finder must be persuaded in order for the ultimate claim to be proved, i.e. if the party having that burden has failed to satisfy it, the issue is to be decided against that party. The initial allocation of the *burden of production* is regulated by the premise types of the argumentation scheme applied. *The burden of persuasion* is allocated by assigning the appropriate proof standard. As the dialogue progresses, subject to the argumentation protocol, the burdens may be reallocated by changing the assignment of premise types and proof standards via speech acts designed for this purpose.

Multiple forms of reasoning can be integrated in Carneades architecture, which allows a number of computational models of argumentation to be used together to search and construct argument as in [73]. Unlike Dung's framework, Carneades is a formal model of argument structure. Arguments are designed to model instantiations of argumentation schemes linking a set of premises to a conclusion, where the premises and the conclusion of arguments are statements about the world, which may be accepted or rejected. However, Carneades does not depend on a particular logical language for expressing statements, inference rules or argumentation schemes.

Carneades constructs an argument graph, which is a kind of proof tree that provides a basis for explanations and justifications. Argument graphs have two kinds of nodes; statement nodes and argument nodes. The evaluation of arguments in Carneades depends on whether statements have been questioned or decided, the allocation of the burden of proof, and the proof standard applicable to questioned statements.

Carneades has been reconstructed as an ADF in [52]. This translation yields acyclic Carneades structures; however, this property has been generalised using the resulting ADF to produce cyclic argument structures. Further benefits from this Carneades-ADF translation allows the generalisation of argument evaluation structures on ADF nodes, and helps put Carneades on a solid formal foundation by clarifying the relationship between Carneades and Dung AFs (both are instances of ADFs). Another recent translation considers finding the relationship between Carneades, AF, and ASPIC+ [127] producing a cycle-free argumentation framework, thus always inducing a unique Dung extension which is the same in all Dung's semantics. Carneades has also been used to reconstruct legal cases decisions (Pierson v. Post [72] and Popov v. Hayashi in [75]).

2.4 Summary

This chapter has presented an overview of a number of areas this research has drawn upon in proposing the work in the forthcoming chapters. The main background setting for this research is the area of Argumentation in AI and Law. Three main topics have been discussed, starting with the core topic of AI and Law and covering legal reasoning models and legal CBR systems, legal text processing and legal ontologies. Then, dialogue typologies and the components of dialogue models were presented, showing examples of legal dialogue models. Finally, an overview of argumentation frameworks was provided, showing abstract and structured argumentation. A summary of the key points from each topic is briefly presented below:

Legal CBR Systems The chapter began by exploring a number of landmark legal case-based reasoning systems in the domain of AI and Law, mainly HYPO and HYPO's descendants (CATO, IBP and AGATHA). All four systems have used cases from the US Trade Secrets domain, varying in their knowledge base size, to identify arguments. However, the systems are distinct in their case representation and reasoning: HYPO uses dimensions, CATO uses factors organised in a factor hierarchy, IBP groups CATO factors into an abstract factor hierarchy rooted in issues, and AGATHA uses factors and values to empirically evaluate the theory-based reasoning of [38]. Three key moves have emerged from these systems: (i) citing a case, (ii) distinguishing a case, and (iii) providing a counter-example. In contrast to the other systems, IBP resolves the conflict between the arguments and predicts the outcomes of cases. AGATHA produces a theory of the domain, while HYPO and CATO present the arguments without resolving the conflict between them.

The research undertaken in this thesis has reused the factor hierarchy of CATO, and mapped it to the logical model of IBP to instantiate an Abstract Dialectical Framework for 32 cases of US Trade Secrets domain used in AGATHA. Chapter 5 defines the acceptance conditions for all the base factors and abstract factors of CATO and IBP and translates it into an executable form to evaluate the approach. Possible refinements are suggested in Chapter 6. The findings are compared with other domains and different legal CBR systems in Chapter 7.

Models for Legal Reasoning Next, this chapter investigated various approaches in the domain of AI and Law applied at different layers in modelling legal reasoning: (i) from evidence to agreed facts; (ii) from agreed facts to intermediate concepts (factors, and issues); and

(iii) from intermediate concepts to legal consequences, which is the stage concerning legal CBR systems discussed previously. The legal case representation applied in this thesis considers the second and third layers, showing an inference tree (ACT) with facts, factors and issues in Chapter 4, and considering the case facts layer in reasoning with ADF in Chapter 6.

Legal Ontologies and Legal Text Processing The knowledge representation encapsulated in this research was all done manually, as in the case-based reasoning discussed previously. This chapter considers research efforts in the AI and Law domain that pertain to processing legal texts by providing examples of the use of NLP techniques for ontology construction and factor annotation, specifically of unstructured linguistic information, to address the knowledge acquisition bottleneck.

Research in this thesis considers the role of legal ontology in the development of legal representation models to provide explanations for the components extracted from natural language, but not NLP techniques. In Chapter 4, a manual analysis is applied to mark-up the dialogues in the US Supreme Court Oral Hearings. The elements are used to feed a legal ontology which serves as a repository for the argument components of a legal domain.

Legal Dialogue Models Another aspect of concern in this research is modelling legal dialogues. Dialogue systems were originally introduced into AI and Law as a way of modelling legal procedures. The outcome in most of these dialogues is not fully determined by the participants but by a referee (a judge or jury). This chapter has discussed a number of legal dialogue games including Pleadings Games, DiaLaw, TDG and PADUA. All these systems examine persuasion dialogues in different legal contexts. The protocol and moves of these dialogue games work together to fulfill the goals of the dialogue.

One of the key aims of this thesis is to provide a computational model for a legal procedure, running in a real context, by considering some particular natural dialogues that form a clearly defined stage of the US Supreme Court process, namely the Oral Hearings stage. In comparison to other contexts, dialogues in the legal domain combine arguments from different sources, i.e. argument about the case evidence and facts, argument from legal rules, argument from precedent cases, argument from hypothetical tests and others which are required to resolve the ambiguity of the conflict issues. However, the structure of exchanging arguments in legal dialogues is not clear: the argument types are interleaved and there is no particular order for the parties to pose arguments, which makes the analysis of oral hearings more complicated. Chapter 3 first examines the goal of the Oral Hearing dialogues in order to define the structure of the dialogue representation model, including the speech acts and the rules that form the protocol of these Oral Hearing dialogues in Chapter 4. However, the main goal of the dialogue is to extract the relevant argument components, i.e. there is no outcome or winner.

The dialogue moves defined in these dialogues are different from the argument moves presented in HYPO, CATO and AGATHA since these systems do not embed their moves within a dialogue. However, the three ply approach of HYPO is similar to that used in the Oral Hearings of the US Supreme Court. First, the plaintiff cites some cases supporting his side. The defendant cites cases supporting his side in the second ply (counter-examples to those cited in the first ply). In the third ply, the plaintiff is allowed a rebuttal.

Frameworks For Argumentation Argumentation Theory is the study of the process of making a decision through logical reasoning, which itself involves different types of reasoning [33]. Argumentation theory provides a number of mechanisms which are useful in their application to AI. The final topic discussed in this chapter is the various frameworks for abstract and structured argumentation, starting from Dung's abstract argumentation frameworks, and explaining the distinction between the various refinements of Dung's frameworks, including ADFs. The advanced features of ADFs motivated the core work in this thesis: the nodes in ADFs are statements rather than specifically abstract arguments, and the links can be a variety of interpretations that define acceptance conditions local to each particular node.

In particular, ADFs can take a design role to drive and record the knowledge base in order to apply a body of case law, taking advantage of the knowledge features embedded in the ADF, as is fully discussed in Chapter 5. Legal decisions are typically not truth functional: they require both a context (e.g. a set of cases) and a procedure (e.g. stare decisis). Both of these can be represented in the "acceptance conditions" of an ADF. The method is applied in different legal domains in Chapter 6 and evaluated in Chapter 7.

Several formalisms for structured arguments have also been presented, showing how an argument can be constructed from a set of premises by performing consecutive reasoning steps through argumentation schemes. The various types of argument attacks in structured argumentation in ASPIC+ have a strong resemblance to legal cases represented in ADFs (discussed in Chapter 7). This chapter finally described Carneades and how it has been reconstructed as ADF.

Each of these areas plays an important part in the research presented in the following chapters for the computational representation of legal domains. The next chapter will provide the necessary background about the legal procedures and the legal domains relevant to this thesis.

Chapter 3

Legal Case Decision Process and Legal Domains

"To succeed, planning alone is insufficient. One must improvise as well." Isaac Asimov, Foundation

---

The purpose of this chapter is to provide some necessary background about the procedures of the US Courts, with a focus on the US Supreme Court since this is the main legal procedure body examined in this research. Furthermore, an overview of the legal case domains that have been used as case studies in the thesis is provided. All the selected legal domains are related to the US Courts where all the case briefs, Oral Hearings and Opinions are available to the public and many have been previously analysed in the AI and Law literature. This chapter gives an overview on courts procedures in Section 3.1. Then, briefly describes the Supreme Court processes in Section 3.2 and the role played by the Oral Hearings and Legal Opinion stages in Section 3.3 and Section 3.4 respectively. The chapter also provides an overview of the particular legal case domains that have been used as examples in this thesis in Section 3.5 and concludes with a Summary in Section 3.6.

3.1 Overview on Court Procedures

Following the legal reasoning model described in the previous chapter (Section 2.1.2), every legal case starts from evidence and ends with a legal decision. This decision-making process may require the involvement of several courts, starting from the Lower Courts and moving up to the Appellate and Supreme Courts. Every Court requires the investigation of specific stages in the legal reasoning model: Figure 3.1 illustrates the stages of reasoning in combination with the relevant Courts. This diagram illustrates the idea that the lower courts determine the facts and apply the law as established, whereas the higher courts take the facts as given and interpret the law. The US legal system is derived from the English *Common Law* system that uses the principle of *stare decisis* ("let the decision stand"), where cases are decided according to consistent principled rules as established in precedent cases, so that similar facts yield similar results. Decisions in Common Law jurisdictions (rather than civil law as in the continental European usage), are heavily influenced by precedent cases. Case law, in common law, is the set of decisions that can be cited as precedents. If the current case is distinct from precedents, judges have the authority to make law by providing tests and creating precedents for future cases. The focus in this research is on civil cases (in the Anglo-American usage, where it refers to non-criminal cases), such as those applicable to Trade Secrets and Property Law. In criminal law cases the emphasis is much more strongly on what the evidence establishes, rather than distinctively legal reasoning. New cases, whether they are Civil or Criminal, are presented first to Trial Courts, where evidence and testimonies are discussed and examined. This *evidence* is used as a basis to determine *facts* by following the procedure of law in order to resolve the dispute between the conflict parties[1]. Typically, there will be standards of proof that must be met, and often the responsibility of deciding facts will be given to a jury. Once the facts have been established, determining the outcome should typically be a straightforward application of existing law. In many legal processes, including UK and US Appeals, the facts are decided by the court of first instance, and cannot be revisited at the appeal and subsequent stages. Thus, for example, in US Supreme Court cases, which are the subject of much study in AI and Law, the facts are never in dispute. In some scenarios, decisions made by Lower Courts are reviewed in Appellate Courts, and Supreme Courts. The role of these courts is to affirm, reverse or quash the decision made by Lower Courts. The facts decided by the Lower Court are used to determine the presence or absence of *base factors*. The Courts' decision shows what are the factors that require consideration, how the conflicts between these factors are resolved, and what are the *issues* served by this decision. All the examples in this research are in the higher courts. The next section explains in more detail the processes of the US Supreme Court, the highest court in the United States, with further investigations of the Oral Hearing dialogues and the opinions produced by this Court.

3.2 Supreme Court Process

Typically the Supreme Court reviews cases that have been decided in lower courts, affirming, quashing or reversing the lower court decision. The Supreme Court receives a number of *certiorari* requests from parties (*petitioners*) who are not satisfied with lower court decisions, asking for a review of their cases. Normally, when a case for consideration of *certiorari* is accepted, the petitioner and respondent write briefs setting out their positions and recommendations to prepare the Justices for the oral argumentation. Briefs may also be supplied by other interested parties, such as the Solicitor General. These are the so-called *amicus curiae* (friend of the court) briefs. When the Justices have considered all the briefs, the Oral Hearings take place. The total time for the oral argumentation is just one hour; thirty minutes for each party. Normally the petitioner will begin, reserving some of his thirty minutes for rebuttal. The respondent will follow for thirty minutes, and the petitioner will finish, taking the remaining time for a rebuttal. Following the Oral Hearing, the Justices meet in a Justice conference to discuss and vote on the

[1]US Courts URL: http://www.uscourts.gov/about-federal-courts/court-role-and-structure.

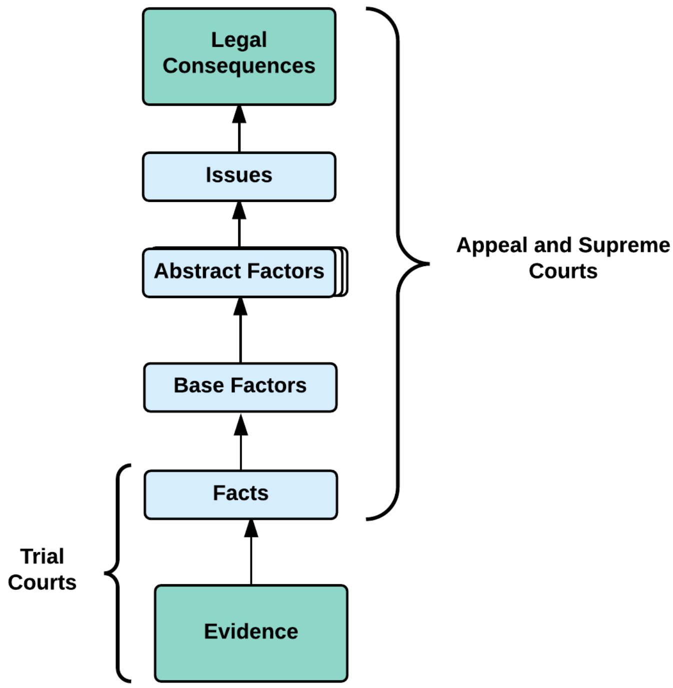


FIGURE 3.1: Legal Reasoning Stages in Different Courts

case. Following this the opinions are prepared: one Justice will be chosen to write the opinion of the Court, and the other Justices may, if they wish, write their own concurring or dissenting opinions. Figure 3.2 illustrates the procedure of the Supreme Court from the time of receiving the *certiorari* until the court opinion. The top level illustrates the main processes in the procedure, while more detailed sub-processes are presented in lower levels. The Supreme Court is expected to give a decision in the case under review, but it needs to look to the past and the future as well. The decision is often expressed as a rule which will be applicable to future cases, and which will, as far as possible, be consistent with previous decisions of the Court: see e.g. [83].

3.2.1 Oral Hearing Dialogues

This section describes the initial situation, the individual goals and the collective goals for Oral Hearings of the US Supreme Court as required to characterise dialogues in Walton and Krabbe [133], which will help to drive the analysis of the Oral Hearing dialogues in terms of the computational perspective. Within the Supreme Court procedure, there are three nested dialogues in the main oral argumentation dialogue (Figure 3.3). The *overall aim* is to establish the various elements, and the connections between them, expressed as clearly and unambiguously as possible, which can be used by the Justices to construct the arguments they will use in their opinions. Tests for the elements may also be proposed and refined. Each of the three dialogues will involve a counsel and up to nine Justices. No distinction between the Justices is presented in the analysis.

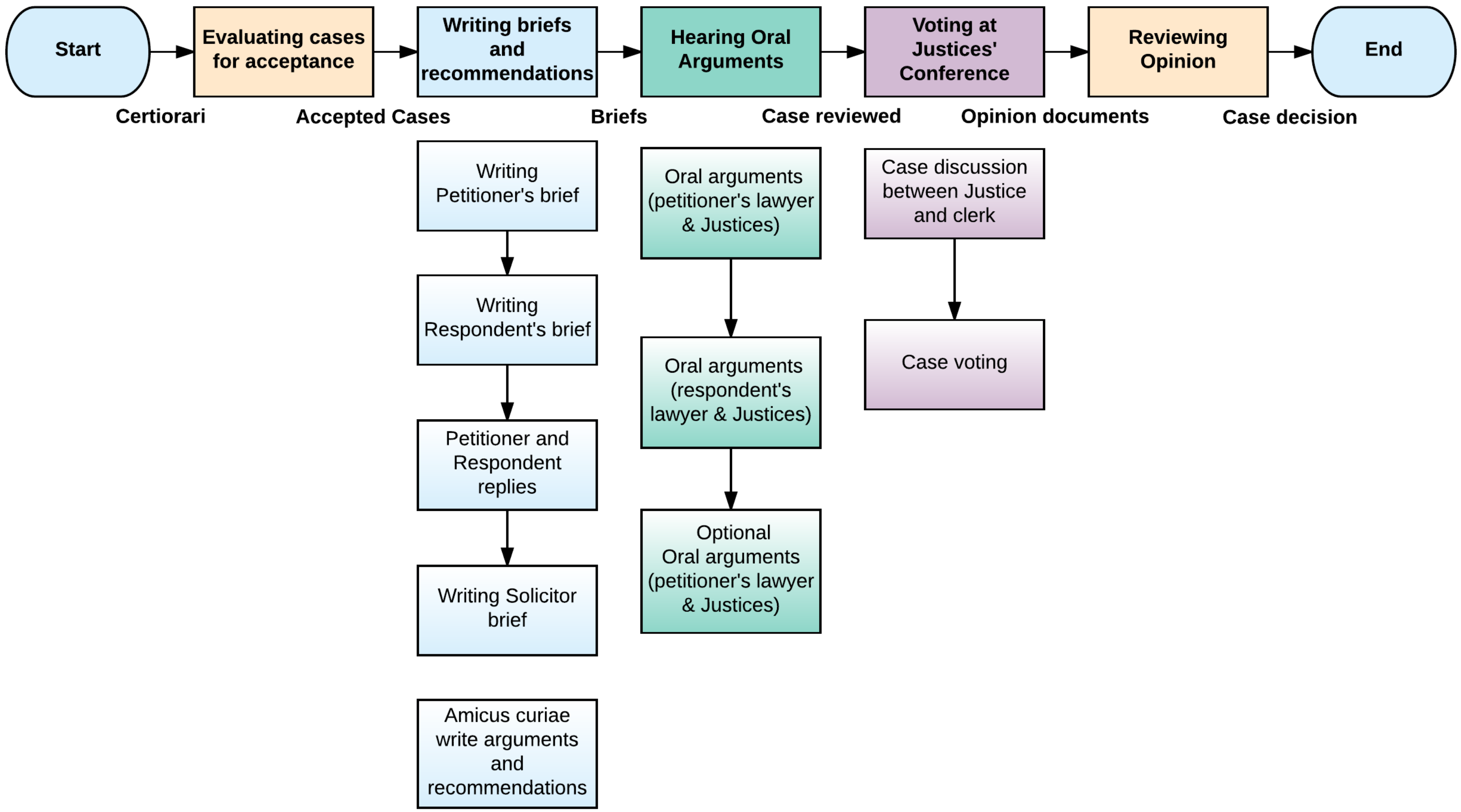


FIGURE 3.2: U.S. Supreme Court Procedure

Essentially they will all ask questions to clarify and challenge the argument components proposed by counsel, although the particular questions they pose may well be motivated by their own views of the case, and their developing ideas of the argument they will use to decide the case. The arguments produced in the opinion will essentially introduce and use a *test* [19], which will be binding on future cases satisfying the test, and which will allow a decision to be made by using the facts of the current case to establish the presence of a set of factors which will resolve the issues in favour of one of the parties.

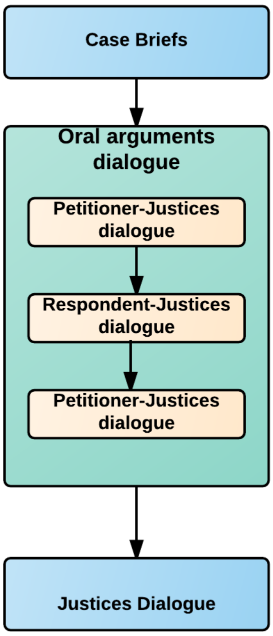


FIGURE 3.3: Oral Hearing Nested Dialogues

In the *initial state* of the **petitioner presentation**, briefs from the petitioner, respondent and any "friends of the court" are available. These will set out (and justify) a set of tests which would provide candidate arguments: counsels will in turn present the elements of a test which, if accepted, will ground an argument for their clients. The briefs will also state the accepted facts of the case, and draw attention to relevant precedent cases. The *collective goal* is to obtain a clear statement of a set of elements that can be used to form an argument which will resolve the case. Individually *the counsel* will wish to present an acceptable test which will lead to a decision for the petitioner and to answer any critical questions satisfactorily: modifying his tests if necessary. *The Justices* will wish to clarify any points that have not been made clear in the original brief, and to pose challenges arising from the other briefs. The *collective goal* of the second dialogue, **the respondent presentation**, is to obtain a clear statement of the test advocated for the respondent. The respondent dialogue differs in its *initial state* because the petitioner has already presented. Thus as well as presenting his own test, *counsel for the respondent* may find some difficulties with the test proposed by his adversary. *The Justices* remain interested in clarification and eliciting answers to questions arising from the other briefs. While the *collective goal* of the

	Initial State	**Individual Goal**	**Collective Goal**
Petitioner OH	Briefs	Petitioner: Proposing tests and answering Justice questions that would decide the case for the petitioner. Justice: Clarify ambiguous points and pose questions form other briefs.	Clear statements of set of elements which can be used as an argument that resolves the case
Respondent OH	Briefs + Petitioner's tests and elements	Respondent: Proposing tests, rebutting petitioner's tests, answering Justice questions that would decide the case for the respondent. Justice: Clarify ambiguous points and pose questions form other briefs.	Clear statements of set of elements which can be used as an argument that resolves the case
Petitioner Rebuttal	Respondent's tests and elements	Petitioner: Rebutting respondent's tests, propose elements and tests that would decide the case for the petitioner. Justice: Clarify points if required.	Clear statements of set of elements which can be used as an argument that resolves the case

TABLE 3.1: Characteristics of Oral Hearing Dialogues

rebuttal dialogue is again a clear statement of the tests and the elements composing them, and the choices that must be made when deciding between the tests, the *initial state* now also contains the respondent's test and its elements, and the individual goal of the petitioner's *counsel* is to pose questions against this test. *Justices* usually say very little during this stage, but they may wish to seek clarification of some points. The summary of this characterisation of the three dialogues is shown in Table 3.1. The goal of the three dialogues as a whole is to provide a clear statement of possible tests and the elements used in them, which the Justices will employ to decide the case and construct the arguments in their opinions.

3.2.2 Legal Arguments in the Court Opinion

After the Oral Hearings, there will be a set of available argument components, in the form of facts, factors and issues, and possible linkages between them. The task now is to analyse these alternatives to produce an answer for the case, and a test applicable to future cases. This is the role of the Justices' conference stage. Different Justices may make different choices, which may lead Justices to write individual opinions either dissenting from the majority, or concurring by expressing a different justification of the majority or minority decision. These different Justices' opinions are explained below:

Majority Opinion Is the opinion that prevails, when the majority (five or more out of the nine Justices) provide an opinion on an issue and that opinion is the one that prevails.

Dissenting Opinion The other Justices, who disagreed with the majority opinion, sometimes provide one or more *dissenting opinions* to express their disagreements.

Concurrence Sometimes, one or more Justices who agree with the ultimate decision made by the majority or minority, but disagree on the decision justification, wish to provide comments in a concurring opinion.

The Court Opinion is more than a case verdict. The opinion provides an explanation for the justification that leads to this verdict, showing the majority, minority (dissenting) and concurring opinions. In some opinions the justification involves providing a test to resolve a new case (landmark case) and provides a means of deciding for future cases: this is an essential part of what the court is trying to achieve. A typical case Opinion document in the US Supreme Court is structured in the following parts:

Syllabus Every opinion starts with a syllabus which is a summary of the case, outlining the case facts and the decisions of the previous courts. The last portion of the syllabus states the Justices supporting the majority (Main) opinion, and which Justices have issued concurring or dissenting opinions.

Main Opinion This is the official opinion of the case, written by one of the Justices. The main opinion involves the case decision and an explanation for the decision, stating the issues that the Justices relied upon in their justification. The majority opinion shows that all the Justices (unanimous opinion), or at least more than half of the Justices agree to one decision and one Justice is chosen to write the majority opinion.

Dissenting and Concurring Opinions Often, the opinion document involves multiple opinions; Justices rarely agree to one main opinion. They might agree with the decision but have different justifications, presented as concurring opinions. Likewise, Justices might disagree with the decision and thus have dissenting opinions.

Disposition At the end of the main opinion, the document clarifies the court action, which might be affirming, reversing the lower court decision, or returning the case to the lower court (remanding).

3.3 Legal Case Domains

Now that a general review of the US Courts, and specifically the US Supreme Court process, has been presented, it is time to provide a brief introduction to the legal case domains from these Courts, which are used as examples in this research. Civil case domains are selected from the Common Law. Concentrating on these areas follows the majority of work in AI and Law, recognising that the case law of a legal domain goes through a life cycle of three stages as stated by Levi in [88]. The selected cases are related to the second stage where there is a period of stability and the theory is settled.

> But it is not simply deductive. In the long run a circular motion can be seen. The first stage is the creation of the legal concept which is built up as cases are compared. The period is one in which the court fumbles for a phrase. Several phrases may be tried out; the misuse or misunderstanding of words itself may have an effect. The concept sounds like another, and the jump to the second is made. The second stage is the period when the concept is more or less fixed, although reasoning by example continues to classify items inside and out of the concept. The third stage is the breakdown of the concept, as reasoning by example has moved so far ahead as to make it clear that the suggestive influence of the word is no longer desired.

Each of the following sub-sections gives an overview of what the domain is, the specification of the cases used in that domain, and the role this domain plays in the thesis.

3.3.1 Automobile Exception to The Fourth Amendment Domain

The U.S. Fourth Amendment protects the "right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures[2]." A search is considered reasonable if a warrant has been obtained. However, when there is a high probability of losing the evidence so that there is an urgent reason to search, obtaining a warrant may become impossible. One such situation is a moving automobile. This domain thus considers the interaction of two competing considerations: the enforceability of the law, which makes the exigency issue important, and citizens' rights, which include the right to privacy [29].

This exception was first established by the United States Supreme Court in 1925, in the Carroll v. US.[3] decision which states:

> Various acts of Congress are cited to show that, practically since the beginning of the Government, the Fourth Amendment has been construed as recognizing a necessary difference between a search for contraband in a store, dwelling-house, or other structure for the search of which a warrant may readily be obtained, and a search of a ship, wagon, automobile, or other vehicle which may be quickly moved out of the locality or jurisdiction in which the warrant must be sought.

The Automobile Exception is developed further as more cases are decided with new conditions needing to be taken into consideration, such as the type of vehicle or movable container and the status of the vehicle, which influences whether there was an urgent need to search it. For example, was the vehicle traveling on the highway (as in *Carroll*) or it was parked in a parking lot but capable of moving? (as in *California v. Carney*[4]). Was it parked in a private place that is used for accommodation (*Coolidge v. New Hampshire*[5]) and so not subject to inspection without warrant or was it in a public location? Whatever the conditions, there must be a probable cause to search, but is it legal to search the whole vehicle if the probable cause applies only to

[2]Fourth Amendment - U.S. Constitution URL: http://constitution.findlaw.com/amendment4.html.
[3]Carroll v United States, 267 U.S. 132 (1925).
[4] California v. Carney, 471 US 386 (1985).
[5]Coolidge v. New Hampshire, 403 U.S. 443 (1971).

a container inside the vehicle? What if an authorized warrant is easy to obtain?. The aim of the court opinions in this domain is to determine whether there is enough exigency with respect to a possibly lowered expectation of privacy given the particular case facts.

Two case studies from the Automobile Exception domain, US v. Chadwick[6] and California v. Carney, have been used to address the research question of providing a computational dialogue to identify relevant components and instantiate the abstract argumentation format used as shown in Chapter 4 and Chapter 5. These two case studies are briefly explained below:

US v. Chadwick (1977)

The test in this case was to consider whether a *footlocker search* was justified under either the "automobile exception" or "search incident to a lawful arrest", another possible exception to the fourth amendment. The agent officers arrested respondents (one of whom was Chadwick[7]), who were transporting a double-locked footlocker on the train and then loaded it in a car trunk outside the station. The agents had probable cause to believe that the footlocker contained narcotics due to information received. The respondents, together with the automobile and footlocker, were admittedly under the agents' control. Later, at the police station, the agents opened the footlocker in the automobile trunk without warrant, and found large amounts of marijuana.

California v. Carney (1985)

California v Carney[8] concerns the applicability of the automobile exception to motor homes. It arose when drug agent officers arrested Carney, who was distributing marijuana from inside a motor home parked in a public parking lot in the downtown of San Diego for an unknown period of time. After entering the motor home, without first obtaining a warrant, the police officer observed marijuana. This motor home was an integral vehicle with wheels, engine, back portion and was registered as a house car, which requires a special license in California. On the other hand, it did have some interior home attributes such as refrigerator, cupboard, table and curtains covering all the windows. The question was whether the automobile exception applied in this case, or whether such vehicles have too much in common with dwellings and so come with expectations of privacy too great to satisfy this exception.

Most of the examples given in the thesis are from the case of *Carney v. California*, which has been the subject of several papers in AI and Law, especially those which consider the Oral Hearing stage [29, 76, 115]. The opinions of *ten* US Supreme Court cases from the same domain are newly analysed and applied in Chapter 6 to form a knowledge base to instantiate an Abstract Dialectical Framework for legal reasoning.

3.3.2 Wild Animals Domain

The Wild Animals cases were introduced into AI and Law in [40] and extended to the baseball case of *Popov v Hayashi* in [144] [30]. Briefly, the Wild Animals cases concern plaintiffs

[6]United States v. Chadwick, 433 U.S. 1 (1977).
[7]US v. Chadwick Opinion: https://supreme.justia.com/cases/federal/us/433/1/case.html.
[8]California v. Carney Opinion: https://supreme.justia.com/cases/federal/us/471/386/case.html.

Wild Animal Case	Plaintiff Factors	Defendant Factors	Decision
Pierson v. Post	HotPursuit, Impolite, PSport,Vermin	NotCaught	D
Keeble v. Hickergill	OwnsLand,Malice, Nuisance, Pliving	NotCaught, DSport	P
Young v. Hitchens	HotPursuit, Impolite, Pliving	NotCaught, DLiving	D
Ghen v. Rich	Convention, NoBlame, Pliving	NotCaught, DLiving	P
Popov v. Hayashi	HotPursuit, Assulate, NoBlame, PGain	NotCaught, DGain	not P not D

TABLE 3.2: Factor-Based Cases in Wild Animals Domain, where P=Plaintiff, D=Defendant.

chasing Wild Animals when their pursuit was interrupted by the defendant. The Court decision is to determine whether the plaintiff has the right to make a claim against the defendant. Five Wild Animal cases including *Popov v. Hayashi*[9] have been used in this research to provide a simple illustration and clarification for the construction of ADFs for legal reasoning from a factor hierarchy (Chapter 6). In Pierson v. Post[10], Post was chasing a fox for sport, Keeble[11] was hunting ducks, Young[12] was pursuing fish and Ghen[13] a whale, all in pursuit of their livelihoods. *Popov v Hayashi* concerned disputed ownership of a baseball (valuable because it had been hit by Barry Bonds to break a home run record). Popov had almost completed his catch when he was assaulted by a mob of fellow spectators and Hayashi (who had not taken part in the assault) ended up with the baseball when it came free. The Wild Animals cases were cited when considering whether Popov's efforts had given him possession of the ball. The domain factors, and case representations are presented in Table 3.2 and described further in Chapter 6.

3.3.3 U.S. Trade Secrets Domain

HYPO [14], CATO [10] and IBP [55] are all legal case-based reasoning systems designed in the US Trade Secrets domain. Trade Secret law is concerned with the protection of technological and commercial information not generally known in the trade against unauthorized commercial use by others. This involves considering the efforts to maintain secrecy through, for example, a confidentiality agreement and various protection measures to prevent information being obtained through legitimate means, such as reverse engineering or other questionable means. The trade secrets might be disputed when the Plaintiff disclosed the information to defendant in the context of negotiations about a joint venture or sale of a business or disclosed its information in a public forum without obtaining any security measures.

[9]Popov v. Hayashi: WL 31833731 Ca. Sup. Ct. 2002.
[10]Pierson v. Post: 3 CM. R 175,2 Am. Dec. 264 (Supreme Court of N.Y, 1805).
[11]Keeble v. Hickerimzill: 11 East 574, 103 Eng. Rep. 1127 (Queen'S Bencch 1707).
[12]Young v. Hitchens: 1 Dav. & Mer. 592, 6Q.B. 606 (1844).
[13]Ghen v. Rich, 8 F. 159 (1881).

Case	Plaintiff Factors	Defendant Factors	Outcome
Arco		F10,F16,F20	D
Boeying	F4,F6,F12,F14,F21	F1,F10	P
Bryce	F4,F6,F18,F21	F1	P
CollegeWatercolour	F15,F26	F1	P
Den-Tal-Ez	F4,F6,F21,F26	F1	P
Ecologix	F21	F1,F19,F23	D
Emery	F18,F21	F10	P
Ferranti	F2	F17,F19,F20,F27	D
Robinson	F18,F26	F1,F10,F19	D
Sandlin		F1,F10,F16,F19,F27	D
Sheets	F18	F19,F27	D
SpaceAero	F8,F15,F18	F1,F19	P
Televation	F6,F12,F15,F18,F21	F10,F16	P
Yokana	F7	F10,F16,F27	D
CMI	F4,F6	F10,F16,F17,F20,F27	D
DigitalDevelopment	F6,F8,F15,F18,F21	F1	P
FMC	F4,F6,F7,F12	F10,F11	P
Forrest	F6,F15,F21	F1	P
Goldberg	F21	F1,F10,F27	P
KG	F6,F14,F15,F18,F21	F16,F25	P
Laser	F6,F12,F21	F1,F10	P
Lewis	F8,F21	F1	P
MBL	F4,F6,F13	F5,F10,F20	D
Mason	F6,F15,F21	F1,F16	P
MineralDeposits	F18	F1,F16,F25	P
NationalInstrument	F18,F21	F1	P
NationalRejectors	F7,F15,F18	F10,F16,F19,F27	D
Reinforced	F4,F6,F8,F15,F21	F1	P
Scientology	F4,F6,F12	F10,F11,F20	D
Technicon	F6,F12,F14,F21	F10,F16,F25	P
Trandes	F4,F6,F12	F1,F10	P
Valco-Cincinnati	F6,F12,F15,F21	F1,F10	P

TABLE 3.3: Factor-Based Cases in US Trade Secret Domain

For example, in *Televation*[14] copies of the plaintiff's drawings had been used by the defendant which called into question whether the secret counted as reverse-engineerable:

> The mere fact, however, that a competitor could, through reverse engineering, duplicate plaintiff's product does not preclude a finding that plaintiff's techniques or schematics were trade secrets, particularly where, as here, the evidence demonstrated that the reverse engineering process would be time-consuming.

In Boeing Company v. Sierracin Corporation[15], Sierracin argues that if Boeing had secrets, they were lost by passage into the public domain:

[14] Televation Telecommunication Systems, Inc. v. Saindon, 522 N.E.2d 1359 (Ill.App. 2 Dist. 1988).
[15] The Boeing Company v. Sierracin Corporation, 108 Wash.2d 38, 738 P.2d 665 (1987).

> Engineering drawings are prima facie trade secrets. If Boeing automatically lost trade secrets upon submission to the FAA, it could never have them, because FAA submission is required before manufacture of a single airplane. Submission of confidential data to the FAA does not mean that the information is available to the public upon demand. On the contrary, such information is exempt from public disclosure because it could substantially harm the competitive position of the entity which was required to submit the information.

IBP used 186 cases in its very thorough empirical evaluation to predict the outcome of cases; 148 cases were analysed for CATO; and 38 were analysed specifically for IBP. Unfortunately, the analyses of these cases are not all publicly available; therefore this thesis uses the 32 cases harvested from public sources by Alison Chorley and used to evaluate her AGATHA system [58], [57]. All these cases belong to higher courts; either Appellate Courts or Supreme Courts.

This research extends the use of the US Trade Secrets domain further in constructing legal reasoning using an Abstract Dialectical Framework from the factor hierarchy of CATO and the logical model of IBP as discussed in Chapter 5 and Chapter 6. The factor background of the CATO system is described in [10], and the factor definitions are explained in Appendix D in this thesis. CATO uses 26 factors, each identified as pro-plaintiff or pro-defendant. Table 3.3 gives the factor representation of the 32 cases from the AGATHA system [57, 58].

3.4 Summary

This chapter has laid the essential legal foundation for the work presented in this thesis. First, the legal procedure of the US Supreme Court was explained, showing the legal reasoning types involved at this level of the Justice system. In particular, the chapter has characterised the Oral Hearing dialogues taking place at the Supreme Court, as a preparation for constructing the dialogue representation model in the next chapter. The chapter also explained the structure of the legal Opinion document, showing the different opinion types. Furthermore, a brief overview was given for the three legal domains used as test cases in this thesis. All the legal cases presented in these domains operate under the Common Law system where the decisions are heavily influenced by precedent cases.

Chapter 4

Legal Dialogue Representation

"When people talk to each other, they never say what they mean. They say something else and you're expected to just know what they mean." Alan Turing, The Imitation Game

---

This chapter provides a structured analysis of US Supreme Court Oral Hearings to enable identification of the relevant issues, factors and facts that can be used to construct a *test* to resolve a case. The analysis involves the production of what are called 'argument component trees' (ACTs) in which the issues, facts and factors, and the relationships between these, are made explicit. The aim of the work proposed in this chapter is to provide a model to enable the identification and representation of the components that are used to form a test and establish a knowledge base for a legal domain that can guide the design of the legal decision model, i.e. to address the second research question in this thesis. The model has been implemented to develop an oral hearing dialogue program to produce a visual representation of the ACTs for each player in the dialogue in graphical form, taking as input the components from the dialogue speech acts.

The first section of this chapter shows how such ACTs can be constructed by identifying the speech acts that are used by the counsel and Justices within their dialogues (Section 4.1 and Section 4.2) and illustrating the model of the analysis by applying it in Section 4.6 to the Oral Hearings that took place in two case studies, *US v. Chadwick* and *Carney v. California*. The work also involves building an ontology that serves as a repository for the argument components of a legal domain in Section 4.3 and shows how the dialogue representation and ontology work together in a legal decision workflow in Section 4.4. Details of the specification and implementation of the representation model are discussed in Section 4.5. Finally, a summary of the chapter is provided in Section 4.7.

4.1 Oral Hearings Speech acts

The Oral Hearings stage, as explained in the previous chapter, is conducted between the Justices and the counsels of the conflict participants in three recorded oral dialogues. The Oral Hearings are also available in a transcript form, the example below shows an excerpt from the Oral

Hearing transcript of *California v. Carney*[1].

> Unidentified Justice: How long was this vehicle on the parking lot?
> Louis R. Hanoian: The record does not show how long it was there, Your Honor. There was... Mr. Agent Williams witnessed the vehicle in the parking lot for a period of an hour and a quarter, and it had been there prior at various times.
> Unidentified Justice: Does the record tell us the character of the parking lot?
> Louis R. Hanoian: The record doesn't specifically say that.

To enable the proposal and critique of the Oral Hearing components required to construct the arguments of the opinion and resolve the case, a number of dialogue moves are articulated, which are inspired by argumentation schemes and critical questions [132]. The moves challenge the ACT components, or seek additional components to be proposed. Thus, although there are no conclusions, and hence no arguments as such, in the Oral Hearings, the moves have many similarities to those arising from argumentation schemes. Moves can be made either explicitly by one party, or implicitly, triggered by another move which is tacitly presupposed by the move. This is because, as in any natural language interaction, participants make presumptions about what their hearers already understand and infer and so the Oral Hearings will contain ellipses.

Note that the evidence level is not considered here: the facts to be used have already been determined by the lower court, although, as in Q10 below, the answers may provide further considerations are required to show how the facts will be related to factors, and how they can be determined, since some facts may be difficult or costly to obtain, and so may not be appropriate in a given situation. For example, a police officer will be unable to determine how long a vehicle has been parked.

Now consider the questions that can be posed against the assertions. The structure as a whole is meant to provide a *test*. The questions relate to the *test too broad* and *test too narrow* arguments of [15], but the articulated moves below offer a finer granularity since they identify various different aspects with respect to which the test may be deficient. In the following questions, *relevant* means *relevant to deciding the case*.

Q1: Are all the issues relevant?

Q2: Are there other issues that are relevant?

Q3: Are the issues linked correctly?

Q4: Are all the factors really relevant to this issue?

Q5: Is there an additional factor relevant to this issue?

Q6: Is the relationship between factors correct?

Q7: Are all the facts relevant to determining the presence of this factor?

Q8: Is there an additional fact relevant to the presence of this factor?

[1]California v. Carney, Oyez, https://www.oyez.org/cases/1984/83-859.

Q9: Is the relationship between facts correct?

Q10: Can these facts be readily observed by the appropriate person?

These questions permit a test to be challenged as too broad or too narrow at all three levels, and in two ways. As well as challenging the breadth and narrowness in terms of the elements used (e.g. Q1 and Q2), the test can also be challenged in terms of the way the elements are combined, as in Q3. It should also be noted that it is quite common to combine questions: for example Q1 and Q2 can be combined, effectively suggesting the substitution of one element for another. These could be expressed as additional questions, but here combinations of questions are used. Note also that Q10 relates to whether the tests can be applied by the person responsible for applying them in the operational situation: a test that cannot be applied in the actual situation is not acceptable, because providing a test that is applicable in future cases is an essential part of what the court is trying to achieve.

In [14] the response to such questions is said to be one of:

Save the test: Effectively deny that the question is pertinent to the test; for example, if Q8 is posed suggesting that an additional fact would change the position with respect to some factor, it can be maintained that the same position continues to hold.

Modify the test: Exclude an item (e.g. Q1), add an item (e.g. Q2) or change the linkage (e.g. Q3);

Abandon the test: This means withdrawing the current proposal and proffering a new one.

A definition of the dialogue moves is provided below. There are four moves that control the beginning and end of each dialogue and the Oral Hearing of the legal case, while the rest of the moves mainly enforce the result of posing the questions mentioned above by adding new components, combining components or emphasising certain components, similar to the three ply HYPO [14].

For each move a description of the pre-condition(s) and post-condition(s) is clarified according to the protocol of the Oral Hearing dialogues shown in the transition diagram in Figure 4.1. Moves might be specified *explicitly* from the party's utterance, or in some moves, a component might be within the party's knowledge and thus asserted *implicitly* as illustrated by the dotted lines in Figure 4.1.

Note that *"P"* is used to represent the party who plays the move, *"N"* is the party's counsel name in the dialogue, *"C"* is the running case and *"D"* is the running dialogue. Note also that, in the Oral Hearing transcripts, there is no prescribed turn-taking rule (in the sense of Prakken [104]); counsel makes multiple moves and continues until interrupted by a Justice.

Open Case Is an *implicit* move to open the Oral Hearings of a new case.

Format *OpenCase (P, C)*

Pre-condition No other case is open.

Post-condition The case is open.

Example To open the Oral Hearings of *"California v Carney"*:
OpenCase (ChiefJustice, CvC)

Open Dialogue Every dialogue in the case Oral Hearing is *explicitly* opened by the Chief Justice.

Format *OpenDialogue (P, D, N, C)*

Pre-condition The Oral Hearings of the case is open and no other dialogue is open.

Post-condition The dialogue *D* is open.

Example To open the petitioner's dialogue:
OpenDialogue(ChiefJustice, PD, Louis Hanoian, CvC)

End Dialogue The dialogue ends *implicitly* when the dialogue end time is reached or when the petitioner *explicitly* asks to end the first dialogue to save time for rebuttal.

Format *EndDialogue (P, D, N, C)*

Pre-condition The dialogue *D* is open.

Post-condition The dialogue *D* is closed.

Example To close the the petitioner's dialogue:
EndDialogue(ChiefJustice, PD, Louis Hanoian, CvC)

Close Case is the final move that is stated *explicitly* to close the case after listening to all the dialogues in the Oral Hearings.

Format *CloseCase (P, C)*

Pre-condition Case *C* is open, and no other dialogue is open.

Post-condition Case *C* is closed.

Example To end the Oral Hearings:
CloseCase(ChiefJustice, CvC)

Assert Component Asserts an argument component (issue, factor or fact) *explicitly or implicitly*.

Format 1 *AssertIssue (C, D, P, Issue)*

Format 2 *AssertFactor (C, D, P,* $Factor, Factor'sparent)$

Format 3 *AssertFact (C, D, P,* $Fact, Fact'sparent)$

Pre-condition The dialogue *D* is open and the *parent* of the component is asserted. Issues are the root of the ACT; the parent of a factor is either an issue (or combination of issues) or an abstract factor (or combination of abstract factors); and the parent of a fact is a base factor (or combination of base factors).

Post-condition The component is asserted and the child relationship between the component and its parent is asserted, and the ACT is modified accordingly.

Issues Example *AssertIssue (CvC, PD, Petitioner, Exigency)*

Factors Example *AssertFactor (CvC, PD, Petitioner, automobile, Exigency)*

Facts Example *AssertFactor (CvC, PD, Petitioner, car, automobile)*

Combine Components This move is asserted when a relationship between two components, or more, has been stated *implicitly or explicitly*. The relationship could be a conjunction, a disjunction or (+) which means that both components are considered but neither are necessary or sufficient as explained in the following section (Section 4.2).

Format 1 *AssertCombIssues (C, D, P, RelationType,* $Issue1, Issue2...$ *)*

Format 2 *AssertCombFactors (C, D, P, RelationType,* $Factor1, Factor2...$*, parent)*

Format 3 *AssertCombFacts (C, D, P, RelationType,* $Fact1, Fact2...$*, parent)*

Pre-condition All the components have been initially asserted.

Post-condition The new relationship between the components is asserted, and the ACT is updated.

Issues Example *AssertCombIssues (CvC, PD, Petitioner, or,* $\{$*Exigency, Privacy*$\}$*)*

Factors Example *AssertCombFactors (CvC, PD, Justice, and,* $\{$*homeProperties, mobile*$\}$*, Exigency+Privacy)*

Facts Example *AssertCombFacts (CvC, PD, Petitioner, or,* $\{$*car, mobileHome*$\}$*, automobile)*

Emphasise Component This move is used to stress the importance of a specific base factor or fact. Components are emphasised when they have been repeated several times or proceeded with another word that indicates their importance.

Format 1 *Emphasise Factor (C, D, P, Base Factor)*

Format 2 *Emphasise Fact (C, D, P, Fact)*

Pre-condition The component has been asserted.

Post-condition The new component is emphasised in the player's ACT.

Base factor Example *Emphasise Factor (CvC, PD, Petitioner, automobile)*

Fact Example *Emphasise Fact (CvC, PD, Petitioner, wheels)*

Note that the dialogues of the actual hearings are often not well structured: the questions are not posed in any particular order, and may be interleaved with the presentation of the proposal, so that the proposal is modified as it is presented. Moreover, different moves or components can be chosen for an utterance, following the assumption that text in the Oral Hearing transcript can be analysed in different ways, even by legal experts, depending on a variety of analytical options, which lead in turn to alternative moves. Nonetheless, the aim of each counsel is to present and defend the components required for a test which will be favourable to their client, and the Justices aim to get a clear statement of the various components which they can use to build the arguments in their opinions.

4.2 Argument Component Trees

The goal of the Oral Hearing dialogues involves identifying the components that can be used to construct tests that will provide arguments to resolve the case, and the relationships between these components. To reach this goal, the representation model of these dialogues organises the argument components identified above as an Argument Component Tree (ACT), so that for each dialogue in the Oral Hearing one ACT is formed for the counsel (petitioner or respondent) and one for the Justice.

- Each ACT is constructed starting from the issues. Issues (e.g. the issues in *California v. Carney* are Exigency and Privacy) may be *conjunctive* so that all issues must be considered, or they may be *disjunctive* so that the issues are independent, and one positive will suffice. These are shown in the ACT using "and ($\wedge$)" and "or ($\vee$)" respectively. Sometimes, however, the relationship is not truth functional: like factors, all must be considered, but none is necessary or sufficient, e.g. when Case law is elaborating conditions to meet circumstances not foreseen in the original drafting. The issues are then defined by conditions which must be considered, although never individually necessary or sufficient (see [20] for a fuller discussion of these relationships). The non-truth functional relation is shown in the ACT using "+".

- Throughout the dialogue, the participants' ACTs are updated by the assertion of new factors (e.g. Mobility to the issue Exigency) to resolve issues, or facts that indicate the presence of factors (e.g. Wheels indicates Mobility factor) or the linkage between them in order to construct a test. These links may also be truth functional (conjunction or disjunction), or reasons that must be considered, essentially the usual relations between factors of [10].

- The base factors and facts which emerge as the most important in the dialogue are indicated with"*R*" at the end of the component name in the ACT.

By the end of the dialogue, each ACT shows a *complete* representation of a perspective on the components exchanged in the course of the dialogue and possible linkages between them (see Figure 4.2). The next stage is to merge the alternatives from the ACTs to produce an answer

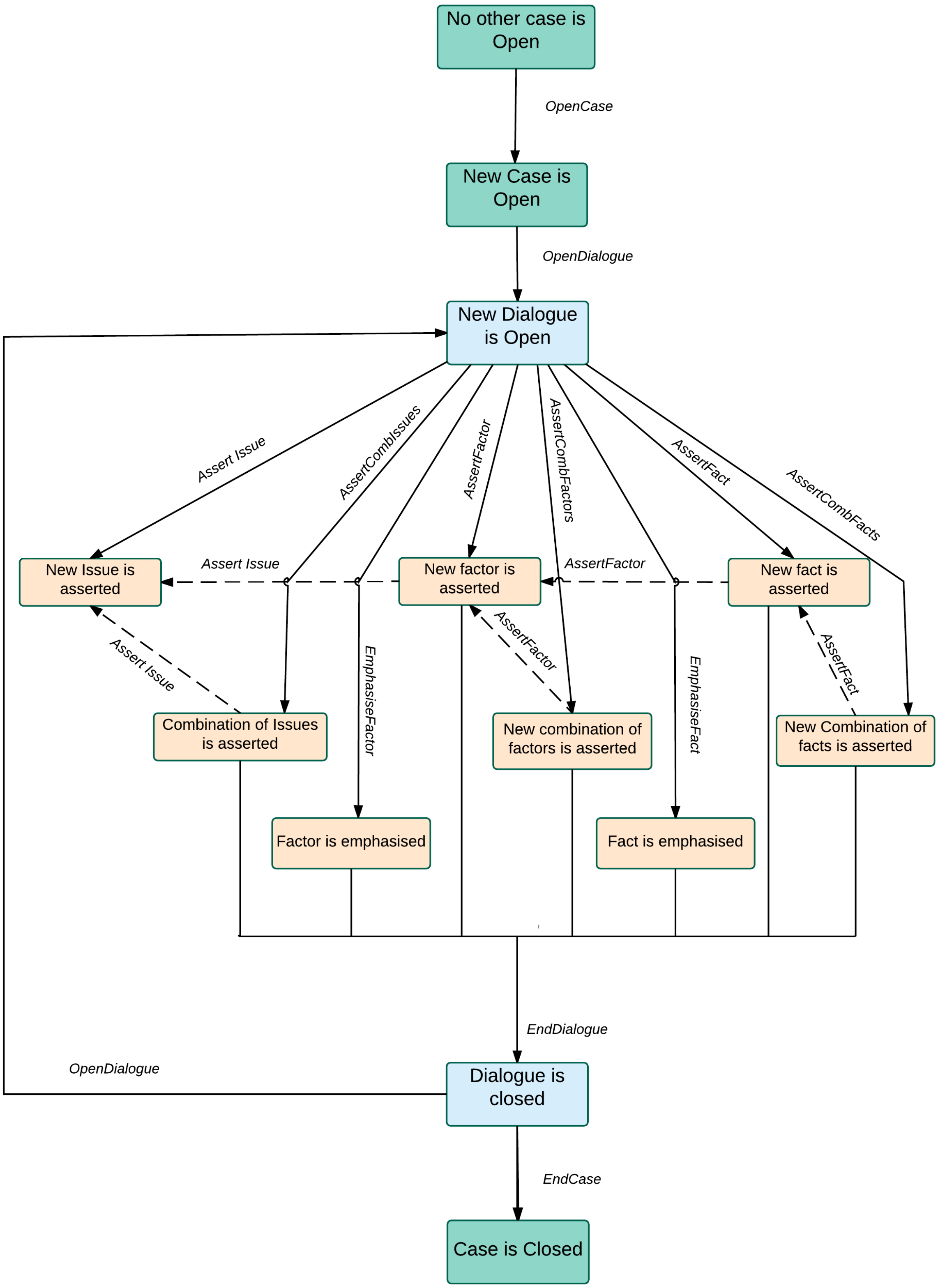


FIGURE 4.1: Oral Hearing Dialogues Protocol

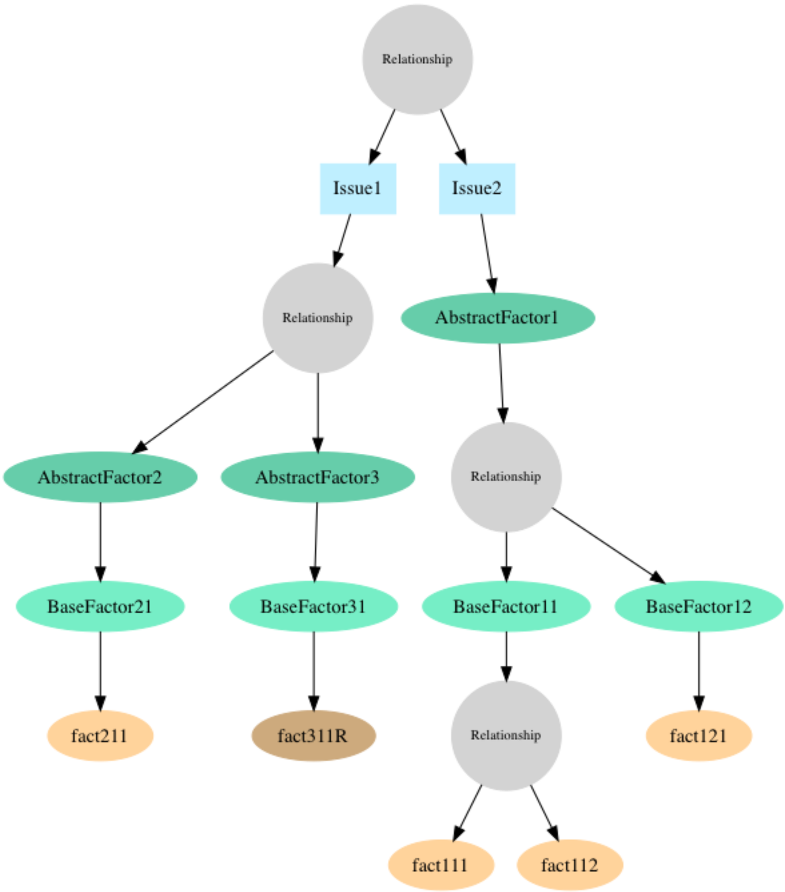


FIGURE 4.2: Argument Component Tree (ACT)

for the case, justifiable by a test applicable to future cases as shown in Chapter 5. These ACTs work as a knowledge base for constructing the Abstract Dialectical Framework for the legal domain. The case studies (Section 4.6) show the ACT construction in a number of scenarios from different cases in the Automobile Exception domain.

4.3 An Ontology For Oral Hearing Dialogues

The above model requires a source of domain knowledge, which has been supplied in the form of an ontology. The ontology is constructed from a manual analysis of the statutes governing the case, the factors identified in precedent decisions, the various facts in previous cases, and from the previous trials of the current case, contained in the briefs. Similar analyses, although not presented explicitly as ontologies, have been performed by Rissland [115], Grabmair and Ashley [76] and Bench-Capon [29], as well as by lawyers writing briefs for related cases. As with these predecessors, all the natural language processing is performed by the analyst and not the program. The references cited above have all included an analysis of *Carney*.

The ontology is incrementally constructed using Protégé[2], an open source visual ontology editor. The structure of the ontology consists of *classes* that represent sets of individuals (object

[2]Protégé URL: http://protege.stanford.edu/

instances); *data properties*, which are the attributes of the classes; *object properties*, which represent the relationship between the objects (individuals) in the domain; and *individuals*, which are the actual objects for the class. The work of [139] presents a legal case ontology that shows the main elements needed to describe case law domains and the number of argumentation schemes that can be used to determine the decision. Although there are a number of common classes between [139] and the ontology presented here, the main focus in this ontology is to provide a basis of legal knowledge of the argument components in the Oral Hearing.

The next section presents the specification of the legal domain ontology, showing the classes required in the domain of Oral Hearings, their class properties and examples of individuals as shown in Figure 4.3 and Figure 4.4.

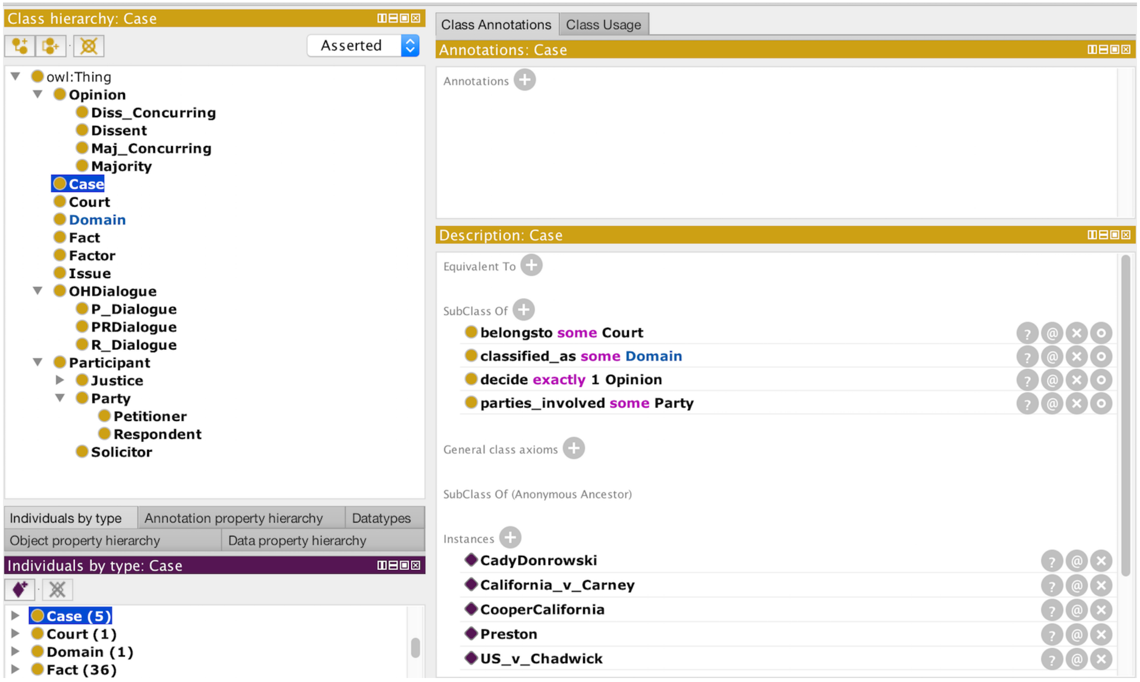


FIGURE 4.3: Screenshot From the Legal Domain Ontology

4.3.1 Entities: Classes and Properties

Class: Participant is a super-class that represents the participants in every case. This class consists of sub-classes for each participant role; i.e. Justice (which includes a Chair sub-class), Party (which consists of Petitioner and Respondent sub-classes) and Solicitor.

Data Properties Every participant has a *"name"*. Moreover, the petitioner and respondent sub-classes in particular have the *"Counsel name"* that represents each party.

Object Properties Every participant is *involved* in one or more dialogues.

Individuals *e.g. State of California:Petitioner; Carney:Respondent*

Class: Court which represents the Court where the Oral Hearing is conducted.

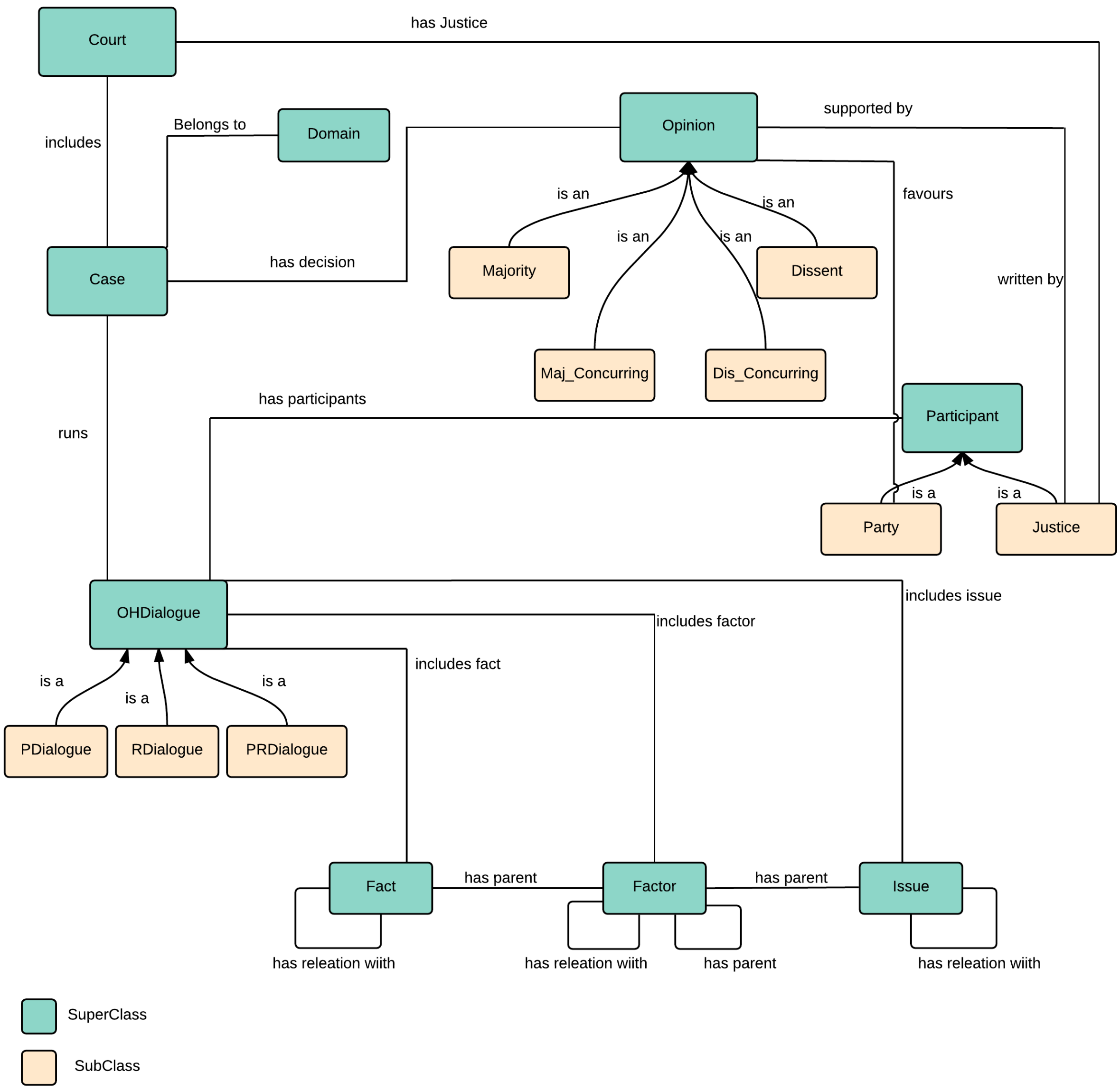


FIGURE 4.4: Legal Ontology Diagram

Data Properties *"Court name"*, *"Court place"*

Object Properties Every court *"has_Justice"* (Different instances of the Supreme Court are often referred to by the name of the Chief Justice: e.g. the *Warren Court*.)

Individual *e.g. US Supreme Court:Court*

Class: Case Domain Legal cases are classified according to specific categories. In the ontology the cases are classified according to common domains.

Data Properties *"domain name"*

Object Properties A case domain *consists* of a number of cases related to that domain.

Individual *e.g. Searches and Seizures:Case Domain*

Classes: Issue, Factor, Fact These are the classes that represent the argument components in the Oral Hearing.

Data Properties *"Issue name"*, *"Factor name"*, *"Fact name"* represent the name of each class respectively. The main source of a factor (*"Factor Source"*) in the factor class, i.e. whether the factor is from the constitutional law, new in the case, hypothetical or it has been retrieved from precedent cases. The *"Fact Type"* that also shows if the fact is truly related to the case, hypothetically mentioned or retrieved from precedent cases.

Object Properties The object properties *"has_parent_issue"*, *"has_abstract _factor"* are used to represent that a factor has either been connected to an issue or has an abstract factor, and *"has_parent_factor"* indicates that every *fact* has a parent *factor*. Moreover, there are object properties that reflect the relationship between the components.

Individuals *e.g. Privacy:Issue; Mobility:Factor; Car:Fact*

Class: OH Dialogue The Oral Hearing dialogue is the super-class for the three nested dialogues: petitioner dialogue; respondent dialogue and petitioner rebuttal dialogue subclasses.

Data Properties *"OH Date"* to represent the arguing date.

Object Properties The property *"has_Justice"* shows the Justices that attend the Oral Hearing dialogue, while *"has_petitioner"* and *"has_respondent"* show the party of the petitioner (and the petitioner rebuttal) dialogues, and the respondent dialogue respectively. Moreover, *"include_issue"*, *"include_factor"*,*"include_fact"*, are the object properties that represent the relationship between each sub-dialogue and the component classes (Issue, Factor and Fact).

Individuals *e.g. CvC_OH:OH Dialogue; CvC_PD:PDialogue*

Class: Opinion The opinion super-class represents the case decision and consists of a number of sub-classes: Majority; Majority concurring; Dissent and Dissent concurring subclasses.

Data Properties *"Decision_date"* which reflects the opinion date and *"Decision_no"* which represents the number of the opinion in the case report.

Object Properties The Object properties *"favour_maj"*, *"favour_dis"* show the parties favoured by the majority and dissenting. The majority and dissent opinion are *"supportedBy_maj"*, *"supportedBy_dis"* one or more Justices and *"writtenBy_maj"*, *"writtenBy_dis"* a specific Justice. However, there are some cases where some Justices have a concurring opinion *"has_concurring_maj"*, *"has_concurring_dis"* shows the relationship between the majority and each majority concurring and the dissent and each dissent concurring respectively.

Individuals *e.g. CvC_Maj:Majority*

Class: Case is the main class that represents the legal case in the ontology.

Data Properties *"case_name"*, *"case_num"*, *"case_year"* which refer to the case name, number in the report and the case year respectively.

Object Properties Each case *"belongs_to"* a Court and is *"classified_as"* one or more domains. Each case has an object property *"hear"* with OH dialogue class and another property *"decide"* with the Opinion class. Each case *"has_petitioner"* and *"has_respondent"* for the petitioner and respondent individuals.

Individual *e.g. California v Carney:Case*

Now that the ontology entities have been described, the next step is to show how to retrieve the individuals from the ontology to create dialogue moves and construct the dialogue ACTs. The workflow in the next section provides a description for this process.

4.4 Legal Case Analysis Workflow

Given the information available to the analyst, the following method is identified for the analysis of cases relating to the domain of law, for which a factor-based analysis is well suited as shown in Figure 4.5.

To begin, a suitable set of cases are identified to represent a domain. These can be obtained by finding a recent case from the desired area and then tracing back through the citations to earlier cases. Several such sets of cases can be found in AI and Law: the US Trade Secrets cases of [10], the Home Deduction cases of [117], the Wild Animals cases of [40] and [37] and the Fourth Amendment cases of [29]. From the available cases, the landmark decisions that shaped the development of the law are selected: these can usually be identified from commentaries on the law concerned, or simply from the number of times they are cited in subsequent cases. A number of researchers have built networks of precedents e.g. Foweler et.al. [67] and Bommarito et.al. in [84].

1. The first step is to process the cases in chronological order. Initially there are no components, but as the knowledge of the domain is built, each new case will, as well as making use of components from previous cases, introduce new components and relate them to existing components. These new cases will most often have new sets of facts, which lead to the ascription of existing factors: cases will typically have quite different concrete facts and so will rarely match exactly on facts. Quite often, however, a case will also introduce one or two new factors, required to resolve the questions raised for the first time by that particular case, and in virtue of which the case is considered a landmark case. For example

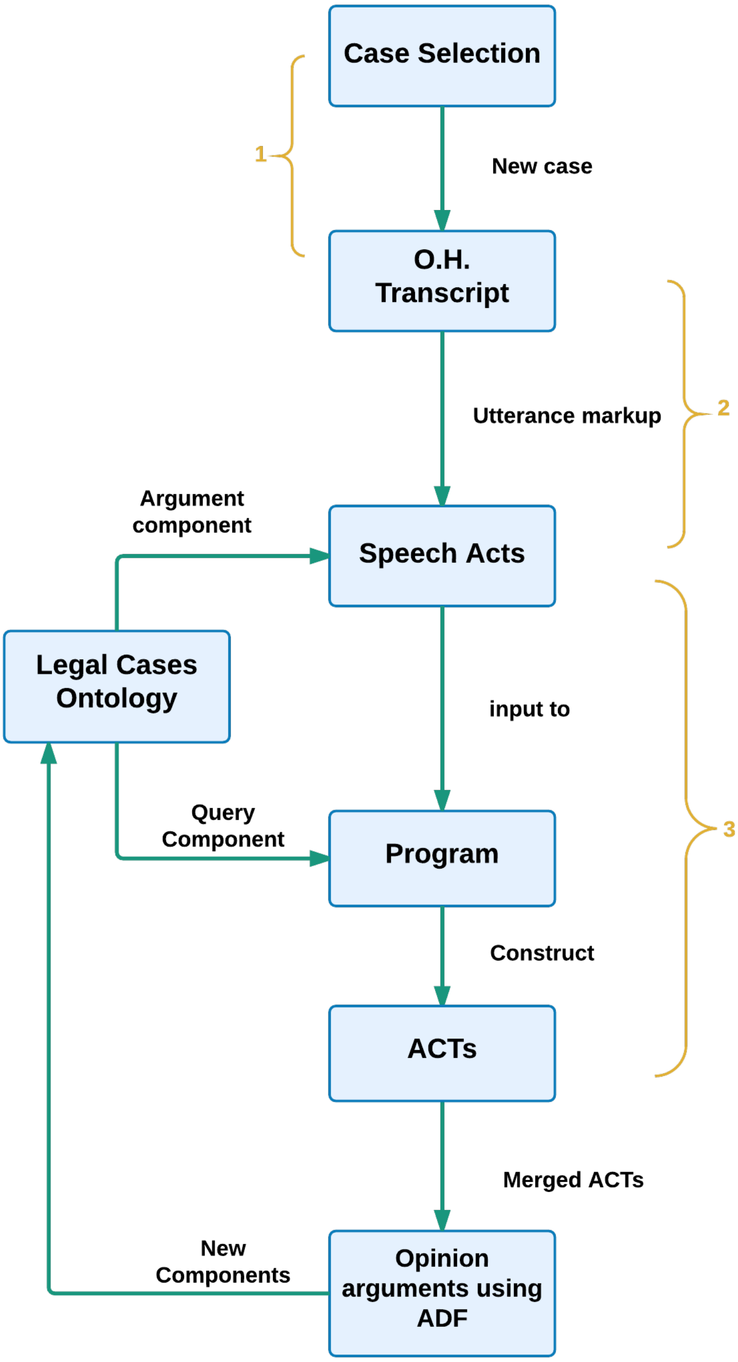


FIGURE 4.5: Legal Case Analysis Method

California v Carney concerned a vehicle that could also be used as a dwelling: this was novel and needed the factors available to be extended in order to represent this question. It is also possible, although rare in a mature domain of law, for issues to be introduced. In regard to the automobile exception to the Fourth Amendment, *expectations of privacy* were not considered to be an issue in the first case (Carrol, decided in 1925) but had been recognised as a key issue and a central part of the discussion by 1985, when *Carney* was decided.

2. Next, for each leading case, the transcript of the Oral Hearing is analysed and each utterance is marked with the appropriate speech act, identifying the content of the speech act either from the existing components recorded in the ontology or introducing a new component from the current case. This provides a method for analysing the transcript:

   - Proceeding utterance by utterance, ensures that all significant elements in the transcript are systematically addressed
   - By classifying the utterances in terms of a limited set of speech acts, making the process more objective.
   - Existing components in the ontology guide the decision of when to introduce a new component.
   - New relationships between existing components and relationships between new and existing components are identified.
   - Alternative perspectives can be shown and used to guide the analysis of the opinion.
   - The need for further components is identified. For example, if a new factor is introduced but it is not applicable in terms of existing facts, additional facts become relevant and need to be included as argument components.

3. Once the transcript has been marked up, using the speech acts, the sequence of speech acts and their contents can be input to a program, to effectively reproduce the dialogues using the restricted vocabulary. The oral hearing dialogue program will process the sequence of speech acts to build the various component trees corresponding to petitioner's, respondent's and Justices' perspectives. The role of the oral hearing dialogue program is:

   - To verify the coherence of the mark-up. If speech acts have been used that fail to satisfy their preconditions, this will be identified by the program so that the mark-up of the transcript can be revised (for example, when a factor for a newly asserted fact has not been previously annotated in the transcript, or is implicitly asserted).
   - To handle implicit assertions. Speakers in dialogues, and the participants in the Oral Hearings are no exception, often presuppose that the hearers will have, and make use of certain background knowledge. If the presuppositions are not obvious or are ambiguous, the utterances may be challenged in the hearing and the party forced to make the presuppositions explicit. Very often, however, they really are obvious to the hearer, and so no challenge appears in the dialogue. In such cases the dialogue

needs to be completed by implicit moves, since the party is tacitly committing to these unstated presuppositions.

- To enable the visualisation of the components and their relationships and the different perspectives of petitioner, respondent and Justices. In this way the choices that need to be made to reconstruct the opinions in the next stage are much more readily identified.

Now the various perspectives on the argument components are established, the nest step is to turn to the opinions. Note, however, the labels of the nodes are not from free text of the decisions, but components taken from the ACTs of the Oral Hearing. Also the relationship between children and their parent may not be the propositional functions usual in such diagrams: rather the diagrams will be ADFs [54], where the acceptance conditions of a node in terms of its children need not be propositional (i.e. may be in terms of factor-based reasoning).

4.5 Specification and Implementation

4.5.1 BNF Grammar

To enable precise specification and implementation of the Oral Hearing dialogues representation model, a Backus-Naur Form grammar is provided to define the syntax of the objects in the model, as shown below. The grammar shows that case law consists of one or more cases where each case is specified by a name, the Oral Hearing (OH) and the case decision. The Oral Hearing is defined by a number of nested dialogues where the outcome of each dialogue is a set of ACTs, one ACT for each participant (Petitioner, Respondent or Justice). The ACT is constructed from components (issues, factors and facts) and the relations between these components. The factor belongs to an issue or another abstract factor, and consists of a number of facts. The factors and facts may or may not be emphasised.

$\langle CaseLaw \rangle$ ::= $\langle Case \rangle \langle Case \rangle$*

$\langle Case \rangle$::= $\langle name \rangle \langle OH \rangle \langle Decision \rangle$

$\langle OH \rangle$::= $\langle Dialogue \rangle$3

$\langle Dialogue \rangle$::=$\langle PACT \rangle$*

$\langle PACT \rangle$::=$\langle Player \rangle \langle ACT \rangle$

$\langle Player \rangle$::=Petitioner | Respondent | Justice

$\langle ACT \rangle$::= $\langle Issues \rangle$ $\langle Factors \rangle \langle Facts \rangle$

$\langle Issues \rangle$:=$\langle Issue \rangle \langle Issue \rangle$*

$\langle Issue \rangle$::=$\langle IssueName \rangle | \langle IssueRelation \rangle$

$\langle IssueRelation \rangle$::= $\langle Issue \rangle \langle rel \rangle \langle IssueRelation \rangle | \langle Issue \rangle$

$\langle rel \rangle$::= and | or | +

$\langle Factors \rangle$::= $\langle Factor \rangle \langle Factor \rangle$*

$\langle Factor \rangle$::=$\langle FactorParent \rangle \langle FactorName \rangle \langle Emph \rangle | \langle FactorRelation \rangle$

$\langle FactorParent \rangle$::=$\langle Issue \rangle | \langle AbstractFactor \rangle$

$\langle AbstractFactor \rangle$::=$\langle Factor \rangle$

$\langle Emph \rangle$::=Y|N

$\langle FactorRelation \rangle$::=$\langle Factor \rangle \langle rel \rangle \langle FactorRelation \rangle | \langle Factor \rangle$

$\langle Fact \rangle$::=$\langle Factor \rangle \langle FactName \rangle \langle Emph \rangle | \langle FactRelation \rangle$

$\langle FactRelation \rangle$::=$\langle Fact \rangle \langle rel \rangle \langle FactRelation \rangle | \langle Fact \rangle$

4.5.2 Implementation

The role of the oral hearing dialogue program has been explained in the workflow presented in Section 4.4: the aim is to implement the representation model using the analysis of the Oral Hearing dialogues. The input to the program is the components identified from the ontology. Then, the output is a visual representation of the ACTs for each player in the dialogue in graphical form. The oral hearing dialogue program was developed in Java, using the support of the Jena library from Jena API, an open source API for ontology management. The design specification of the program is given in Appendix A.

Based on the analysis of Oral Hearing transcripts, the Oral Hearing in the program starts from an *open case* move and terminates with a *close case* move. In between there are two or three nested dialogues according to the Oral Hearing of the case, where each dialogue starts from an *open dialogue* move and ends through an *end dialogue* move. For the main two dialogues (petitioner and respondent dialogues), the program instantiates two new ACTs for the counsel and the Justice directly after the open dialogue move. Following the order of the moves in the transcript analysis, the program retrieves the individual component required for the move. The retrieved component is used to update the ACT instance of the player and display the changes in the trees after each move.

4.6 Case Study: Automobile Exception to The Fourth Amendment

The Fourth Amendment protects the "right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures." A search is considered reasonable if a warrant has been obtained. There are several possible exceptions; one is for automobiles, since their mobility can make it impossible to obtain a warrant, but the exception does not apply to all vehicles in all circumstances. Using the Oral Hearing transcripts of two landmark cases in this domain, *US v. Chadwick* and *California v. Carney*, a manual analysis was performed for the three nested Oral Hearing dialogues using the mark-up scheme shown in Table 4.1.

Tag	Attributes	Appropriate Move(s)	ACT Status
<CASE>	Speaker Name Date Conflict Parties	OpenCase	None
<Dialogue>	Speaker DialogueID Parties	OpenDialogue	If (D1or D2) Initiate a new ACT for each party If (D3) update D1 ACTs
<Issue>	Speaker Name Synonym	AssertIssue	Update speaker's ACT by a new issue
<Factor>	Speaker Name Parent Source Synonym	AssertFactor	Update speaker's ACT by a new factor and a new link to the parent of the factor
<Fact>	Speaker Name Parent Type Synonym	AssertFact	Update speaker's ACT by a new fact and a new link to the parent of the fact
<CombIssue> **<CombFactor>** **<CombFact>**	Speaker Relation type Components	AssertCombIssues AssertCombFactors AssertCombFacts	Update speaker's ACT by a new relation
<Emphasise>	Speaker Component	EmphasiseFactor EmphasiseFact	Emphasise component in the speaker's ACT
<EndDialogue>	Speaker DialogueID Parties	EndDialogue	Complete ACTs
<CloseCase>	Speaker Name Date Conflict Parties	CloseCase	None

TABLE 4.1: Mark-up Scheme for the Oral Hearings Transcript

This mark-up describes the main annotated tags, and their attributes in the transcripts, which are equivalent to the ontology classes and attributes discussed previously. Each tag is related to a particular dialogue move, where the tag attributes construct the move components.

In this section two short scenarios are explained for each case study, demonstrating the dialogue moves according to the dialogue protocol conditions and the tree construction after each move for each party's ACT in the dialogue. Every mark-up tag is related to one explicit move, whether it is for a component itself or the parent of the component. Each component in the dialogue move has either the name of the mark-up text in the transcript or an alternative name that has been used previously in the dialogue, or the legal ontology. Although multiple dialogue moves are captured for an utterance, the dialogue moves may not presented in the same order as they appear in the utterance: the component's parent move is added, explicitly or implicitly, prior to its children.

Note that for each scenario there are:

Case: US v. Chadwick *"USvCh"*, or California v. Carney *"CvC"*.

Parties: Petitioner (*"P"*), Respondent (*"R"*), Justice (*"J"*).

Dialogue: Petitioner Dialogue (*"PD"*), Respondent Dialogue (*"RD"*), Petitioner Rebuttal Dialogue (*"PRD"*).

Fact Type: True in current case (*"T"*), hypothetical (*"H"*), precedent case(s) (*"case name"* or *"PC"* if case name is not mentioned).

Factor Source: New from the current case (*"N"*), precedent case(s) (*"case name"* or *"PC"* if case name is not mentioned), constitutional Law (*"L"*), hypothetical (*"H"*).

Parent Type: Issue (*"I"*), Factor (*"F"*), Issue Relation (*"IR"*), Factor Relation (*"FR"*)

Move Status: *Explicit* moves are shown according to the mark-up tags, while *implicit* moves are illustrated in *italic*.

Appendix E presents the definitions of the Automobile Exception components, while the full analysis and the complete Oral Hearing ACTs for *US v. Chadwick* and *California v. Carney* can be found in Appendix B and Appendix C respectively.

4.6.1 US v. Chadwick Scenario A

Context: This transcript shows how the footlocker has been compared to an automobile in order to use the automobile exception from precedent cases.
Dialogue: Petitioner Dialogue.
Participants: Petitioner and unidentified Justice.
ACTs: Show the construction of the petitioner and Justices' ACTs according to the speech acts (Figure 4.6).

> **Petitioner**: we think that the <*Issue*> reason a search warrant <*CombIssues*> is not required is the same reason a search warrant was not required in the automobile cases. There is, we submit, no rational distinction between the <*Fact*> footlocker involved in this case <*CombFactor*> and the <*Fact*> glove compartment of an <*Factor*> automobile<*CombFact*> or <*Fact*>the trunk of an <*Factor*> automobile, which can be opened <*Issue*> without a warrant. (See Figure 4.6 (a))

Assert Issue (USvCh, PD, P, Exigency)
Assert Issue (USvCh, PD, P, Privacy)
Assert CombIssues (USvCh PD, P, or, {Exigency, Privacy})

Assert Factor (USvCh, PD, P, automobile, Exigency, I, L)
Assert Factor (USvCh, PD, P, container, Exigency, I, PC)
Assert CombFactors (USvCh, PD, P, and, {container,automobile}, Exigency, I, PC)
Assert Fact (USvCh, PD, P, glove compartment, container, F, PC)
Assert Fact (USvCh, PD, P, trunk, container, F, PC)
Assert Fact (USvCh, PD, P, footlocker, container, F, T)
Assert CombFacts (USvCh, PD, P, or, {glove compartment, trunk,footlocker}, container, F, H)

- The issue tag for *reason a search warrant is not required* is renamed to *"Exigency* to reflect the meaning in the automobile exception.
- the CombIssues tag for *is not required* is used to show that the *"Exigency"* issue is treated independently from the privacy requirement in warrant search.
- the fact tag on *footlocker* is combined using CombFact tag with the *glove compartment* and the *trunk* facts of an *automobile* factor. Since that the footlocker is not part of an automobile a CombFactor tag is added to compare between large containers and automobile containers.
- the issue tag, for searching without warrant, is repeated here again to show the parent of the automobile factor.

Unidentified Justice: Well that supports the original seizure of the <*Fact*> footlocker. There was an <*Factor*> automobile and that the seizure of the footlocker took place <*Issue*> without a warrant <*Factor*>because it was an automobile <*CombFactor*> and because<*Factor*> it was incident to a lawful arrest.
So that supports the seizure, but as I understand it the issue is not the seizure of the footlocker, but the <*Factor*> opening and the search of it?. (See Figure 4.6 (b))

Assert Issue (USvCh, PD, J, WarrantlessSearch)
Assert Issue (USvCh, PD, J, WarrantSearch)
Assert Factor (USvCh, PD, J, automobileExigency, WarantlessSearch, I, L)
Assert Factor (USvCh, PD, J, automobile, automobileExigency, F, L)
Assert Factor (USvCh, PD, J, IncidentLawfulArrest, WarantlessSearch, I, L)
Assert ComFactors (USvCh, PD, J,and,{automobileExigency,IncidentLawfulArrest}, WarantlessSearch, I, L)
Assert CombIssues (USvCh, PD, J, +, {WarrantSearch,WarrantlessSearch})
Assert Factor (USvCh, PD, J, container, {WarrantlessSearch+WarrantSearch}, IR, N)
Assert Fact (USvCh, PD, J, footlocker, container, F, T)
Assert CombFactors (USvCh, PD, J,and,{container,automobile}, automobileExigency, F, PC)
Assert Factor (USvCh, PD, J, search location, {WarrantlessSearch+WarrantSearch}, IR, N)

Assert CombFactors (USvCh, PD, J,and,{container, automobile, search location}, { Warrant Search + WarrantlessSearch }, IR, N)

Petitioner: We think that <*Fact*> could be opened back <*CombFactors*> at the <*Fact*> agents' office <*Issue*>for the same reason that the <*Fact*>trunk <*CombFactors*>of an <*Factor*> automobile can be opened back in the <*Fact*>police impoundment lot.
The <*Fact*> glove compartment as in *Cooper v. California* opened back at the <*Fact*> police station. We do see a distinction between what happened here and the search of, for example, <*Fact*> a person's bedroom. (See Figure 4.6 (c))

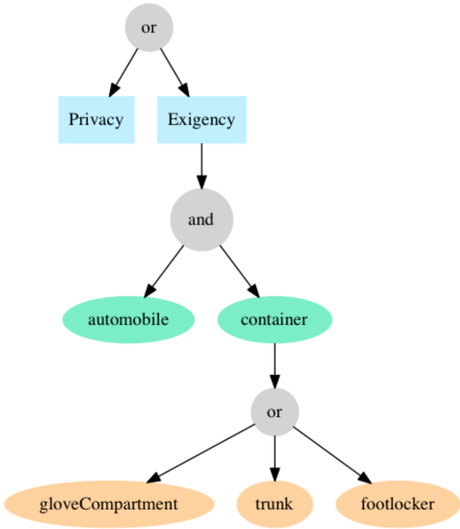
Assert Factor (USvCh, PD, P, Search location, Exigency, I, PC)
Assert Fact (USvCh, PD, P, Police office, Search location, F, PC)
Assert CombFactors (USvCh, PD, P, and, {search location, automobile}, Exigency, I, L)
Assert CombFactors (USvCh, PD, P, and, {search location, automobile, container}, Exigency, I, Cooper v. California)
Assert Factor (USvCh, PD, P, PrivateRoom, Privacy, I, L)
Assert Fact (USvCh, PD, P, bedroom, PrivateRoom, F, H)

4.6.2 US v. Chadwick Scenario B

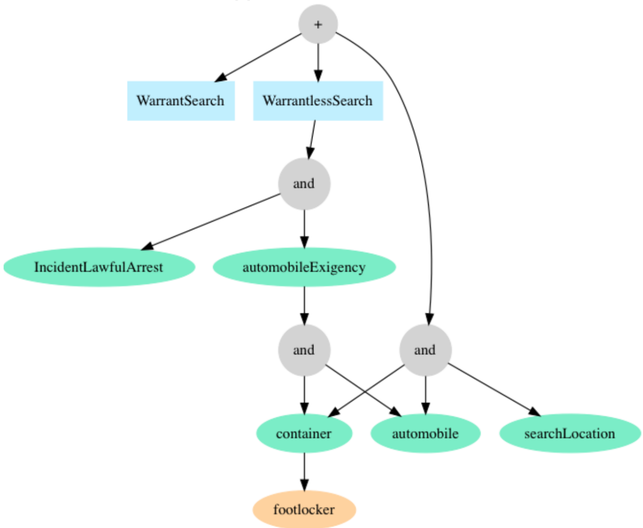
Context: The respondent in this extract is trying to clarify why the footlocker is not like an automobile using precedent cases. This scenario assumes that the conflict issues have been asserted in previous contexts.
Dialogue: Respondent Dialogue
Participants: Respondents and unidentified Justice
ACTs: Show the construction of the respondent and Justices' ACTs according to the speech acts as in Figure 4.7.

Unidentified Justice: What justified the search of the <*Factor*> automobile in *Cooper*? That was not the <*Factor*> automobile theory? It was based on the idea that the <*Fact*> car <*CombFactor*> had been used to <*Fact*> transport contraband. (See Figure 4.7 (a))

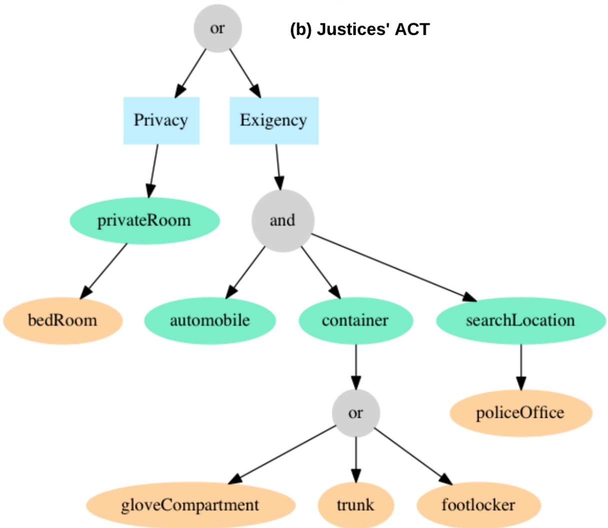
Assert Factor (USvCh, RD, J, automobileTheory, Exigency, I, L)
Assert Factor (USvCh, RD, J, automobile, automobileTheory, F, L)
Assert Fact (USvCh, RD, J, car, automobile, F, Cooper)
Assert Factor (USvCh, RD, J, probableCause, Exigency, I, L)
Assert Fact (USvCh, RD, J, transportContraband, probableCause, F, Cooper)
Assert CombFactor (USvCh, RD, J, and, {probableCause and autombile}, Exigency, I, Cooper)



(a) Petitioner's ACT



(b) Justices' ACT



(c) Petitioner's ACT

FIGURE 4.6: ACTs Construction US v. Chadwick, Scenario A

Respondent: It is only in terms of there is a <*Factor*> forfeiture statute governing the <*Fact*> containers of contraband. However, that statute only goes for <*Issue*> seizure, it does not go to search and the justification for the search in *Cooper*. (See Figure 4.7(b))

Assert Factor (USvCh, RD, R, forfeitureStatute, Exigency, I, Cooper)
Assert Factor (USvCh, RD, R, probableCause, Exigency, I, Cooper)
Assert Fact (USvCh, RD, R, containers of contraband, probableCause, F, Cooper)
Assert CombFactors (USvCh, RD, R, and, {probableCause, automobile, forfeitureStatute}, Exigency, I, Cooper)

Unidentified Justice: I know and then I am just asking why would *Cooper* not pick up, if the United States has the right of <*CombFactor*> probable cause to <*Fact*> seize the footlocker and hold it pending <*Factor*> forfeiture proceedings? (See Figure 4.7 (c))

Assert Factor (USvCh, RD, J, forfeitureStatute, Exigency, I, Cooper)
Assert CombFactor (USvCh, RD, J, and, {probableCause, automobile, forfeitureStatute}, Exigency, I, Cooper)
Assert Factor (USvCh, RD, J, container, {Exigency+Privacy}, IR, N)
Assert Fact (USvCh, RD, J, footlocker, container, F, N)
Assert CombFactor (USvCh, RD, J, and, {probableCause, container, forfeitureStatute}, {Exigency+Privacy}, IR, Cooper)

Respondent: Because, *Cooper* is an <*Fact*> automobile and automobiles have <*Issue*> less of an expectation of privacy than <*Fact*> luggage <*CombFacts*> and footlocker. (See Figure 4.7 (d))

Assert Factor (USvCh, RD, R, luggage, container, F, H)
Assert CombFact (USvCh, RD, R, or, {luggage, footlocker}, container, F, H)
Assert CombFactor (USvCh, RD, R, and,{probableCause, container, forfeitureStatute}, Privacy, I, Cooper)

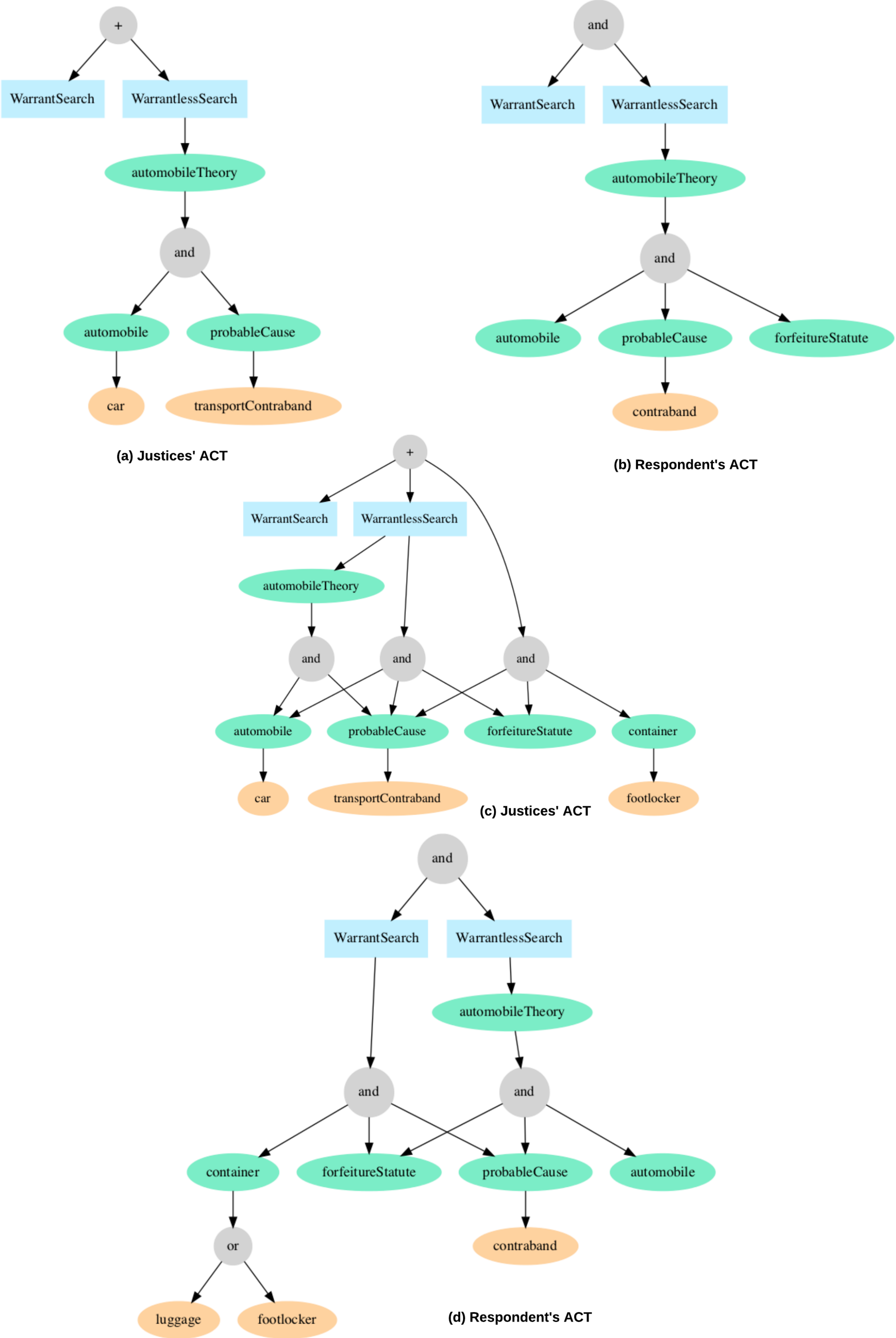


FIGURE 4.7: ACTs Construction US v. Chadwick, Scenario B

4.6.3 California v. Carney Scenario A

Context: Here the Justice is asking questions to clarify the privacy properties of the motor home used in California v. Carney and emphasising the requirements of a bright line facts. The conflict issues are previously asserted in this scenario.
Dialogue: Petitioner Dialogue.
Participants: Petitioner and unidentified Justice.
ACTs: Figure 4.8 shows the ACT construction for both the petitioner and the Justices after each move.

Unidentified Justice: And this <*Fact*> mobile home was <*Fact*> not a tractor-drawn (sic), was it? (See Figure 4.8 (a))

Assert Factor (CvC, PD, J, mobileConfiguration, Exigency+Privacy, IR, N)
Assert Fact (CvC, PD, J, tractorDrawn, mobileConfiguration, F, H)

Petitioner: <*Fact*> No it was not. It was an <*Factor*> integral vehicle with <*Fact*> an engine <*CombFact*>, <*Fact*> wheels <*CombFact*> and <*Fact*> backportion. (See Figure 4.8 (b))

Assert Factor (CvC, PD, P, mobileConfiguration, mobile, F, N)
Assert Fact (CvC, PD, P, notTractorDrawn, mobileConfiguration, F, H)
Assert Factor (CvC, PD, P, integral vehicle, mobileConfiguration, F, T)
Assert Fact (CvC, PD, P, wheels, integral vehicle, F, T)
Assert Fact (CvC, PD, P, engine, integral vehicle, F, T)
Assert Fact (CvC, PD, P, backPortion, integral vehicle, F, T)
Assert CombFacts (CvC, PD, P, and, {wheels, engine, backPortion}, F, T)
Assert CombFactors (CvC, PD, P, or, {notTractorDrawn, integral vehicle}, mobileConfiguration, F, H)

Unidentified Justice: <*CombFactor*> And not tied up to <*Fact*> water, electric <*Factor*> facilities, or anything like that? (See Figure 4.8 (c))

Assert Factor (CvC, PD, J, homeFacilities, Privacy, I, H)
Assert CombFactor (CvC, PD, J, and, {homeFacilities, mobileConfiguration}, {Exigency+Privacy}, IR, H)
Assert Fact (CvC, PD, J, WaterElect, homeFacilities, F, H)

Petitioner: <*Fact*> That's correct, in this particular instance [presents additional facts of the case]. So it is essential that the officers be provided with <*CombIssue*> bright line guidance, guidance by which they know the limits of their power, and so that the people know the limits of their protection. (See Figure 4.8 (d))

Assert Factor (CvC, PD, P, homeFacilities, Privacy, I, N)
Assert Fact (CvC, PD, P, no WaterElect, no homeFacilities, F, H)
Assert CombIssues (CvC, PD, P, or, {Exigency, Privacy})

Unidentified Justice: Would you buy the <*EmphasiseFact*> guideline of wheels? That if the vehicle has wheels on it, it's not a home. (See Figure 4.8 (c))

Assert Factor (CvC, PD, P, integral vehicle, mobileConfiguration, F, T)
Assert Fact (CvC, PD, J, wheels, IntegralVehicle, F, T)
Emphasise Fact (CvC, PD, J, wheels)

Petitioner: If the <*Fact*> vehicle has wheels on it, I think that that makes it mobile and it would be subject to the exception. <*EmphasiseFact*> That would provide a bright line. (See Figure 4.8 (d))

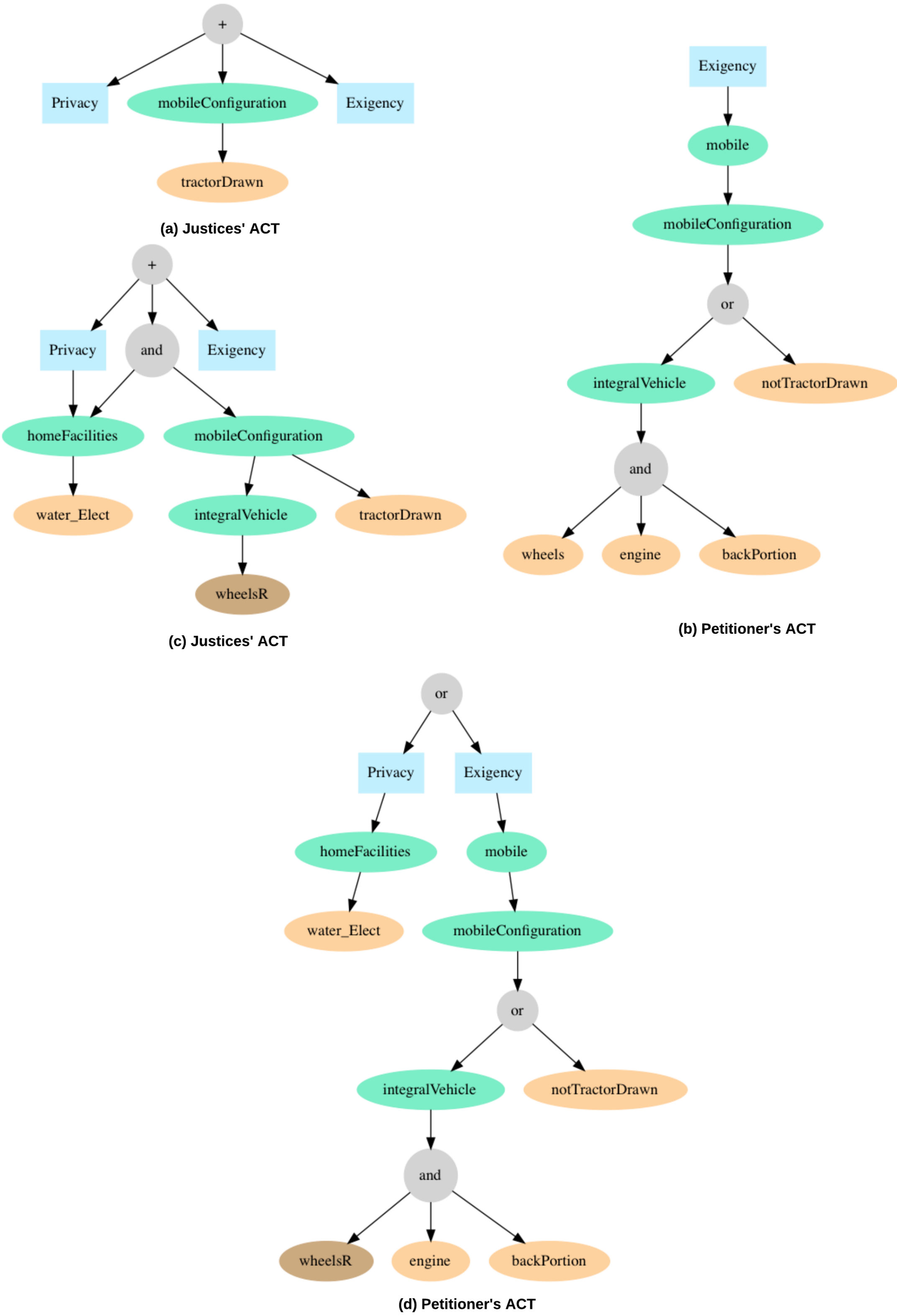
Emphasise Fact (CvC, PD, P, wheels)

4.6.4 California v. Carney Scenario B

Context: This is where the motor home is compared to hypothetical facts (camper tent) and (movable suitcase) from a precedent case *US v. Chadwick*.
Dialogue: Petitioner Dialogue.
Participants: Petitioner and unidentified Justice.
ACTs: Petitioner and Justices' ACTs as in Figure 4.9.

Unidentified Justice: What about a <*Fact*> camper's tent, if the camper takes his things out of the <*Fact*>motor home and pitches a tent next to it? (See Figure 4.9 (a))

Assert Factor (CvC, PD, J, mobility, {Exigency+privacy}, I, L)
Assert Fact (CvC, PD, J, camperTent, mobility, F, H)
Assert Fact (CvC, PD, J, motor home, mobility, F, T)
Assert CombFact (CvC, PD, P, and, {camperTent, motor home}, mobility, H)



(a) Justices' ACT

(b) Petitioner's ACT

(c) Justices' ACT

(d) Petitioner's ACT

FIGURE 4.8: ACTs Construction California v. Carney, Scenario A

Petitioner: The <*Fact*> motor home would be subject to search <*CombFact*>, but <*Fact*> not the tent, not under this particular exception. (See Figure 4.9 (b))

Assert Fact (CvC, PD, P, mobileHome, mobile, F, T)
Assert Fact (CvC, PD, P, notTent, mobile, F, H)
Assert CombFact (CvC, PD, P, and, {notTent, mobileHome}, mobile, H)

Unidentified Justice: <*Factor*> But it is movable. And I should think your reasoning would apply. I'm not saying you are right or wrong. (See Figure 4.9 (c))

Assert Factor (CvC, PD, J, movable, mobility, F, H)
Assert Fact (CvC, PD, J, camperTent, movable, F, H)

Petitioner: <*Fact*> It is <*Factor*> movable. Well, I think the reasoning does apply.
<*CombFact*> But again, this Court has been very careful in drawing the lines to vehicles. For example, the <*Fact*> suitcase in the *Chadwick case*. That's capable of movement, but the Court was reluctant to apply the analysis of allowing <*Issue*>a warrantless search of the suitcase.
However, when the <*CombFact*> suitcase is placed into the <*Fact*> trunk of a car, <*CombFactor*>and the <*Factor*> probable cause arises afterwards, the suitcase is subject to search. (See Figure 4.9 (d))

Assert Factor (CvC, PD, P, movable, Exigency, I, H)
Assert Fact (CvC, PD, P, tent, movable, F, H)
Assert Factor (CvC, PD, P, movable, Privacy, I, Chadwick)
Assert Fact (CvC, PD, P, suitcase, movable, F, Chadwick)
Assert CombFact (CvC, PD, P, or, {tent, suitcase}, movable, F, H)
Assert Fact (CvC, PD, P, cartrunk, mobileConfiguration, F, PC)
Assert Factor (CvC, PD, P, probableCause, Exigency, I, H)
Assert CombFactor (CvC, PD, P, and, {mobile and movable,probableCause}, Exigency, I, PC)

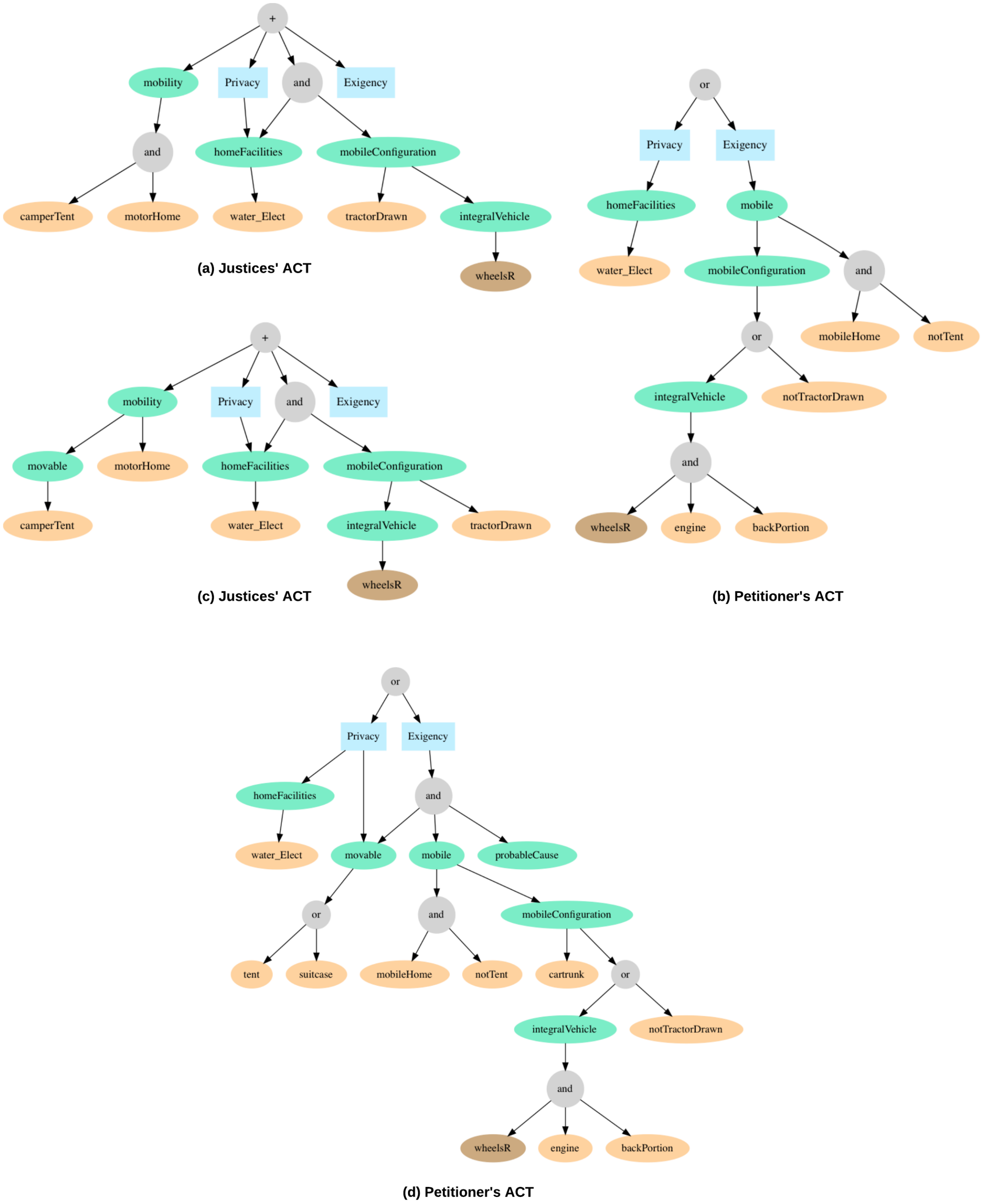


FIGURE 4.9: ACTs Construction California v. Carney, Scenario B

4.7 Summary

This chapter has presented a dialogue representation model for the US Supreme Court Oral Hearings whereby the issues, factors and facts of concern to a case can be identified. Argument component trees have been constructed to provide visualisation for the components and their relationships for the various legal parties. Moving from the Oral Hearing transcripts to ACTs, through the use of a set of defined speech acts identified in a legal ontology, has defined in a legal analysis workflow. This workflow shows how the sequence of speech acts is input to a program to effectively produce the ACTs corresponding to petitioner's, respondent's and Justices' perspectives. The oral hearing dialogue program was then applied to scenarios from case studies showing the dialogue moves and the constructed ACTs. The next chapter explores the role of these ACTs in encapsulating the knowledge of the domain and providing the decision for the case.

Chapter 5

System Design for Reasoning with Legal Cases Using Abstract Dialectical Frameworks

"First comes thought; then organization of that thought, into ideas and plans; then transformation of those plans into reality. The beginning, as you will observe, is in your imagination." Napoleon Hill

---

This chapter presents ANGELIC, a knowledge engineering method for encapsulating legal information from cases using Abstract Dialectical Frameworks (ADFs) [54], and demonstrates the suitability of ADFs for expressing the design of legal case-based reasoning systems. ADFs will play a role in the system design akin to that played by Entity-Relationship models in the design of database systems. The factor hierarchy of CATO is used to instantiate an ADF for the domain of US Trade Secrets, taking into consideration the similarities between the factor hierarchies in CATO and IBP and ADF.

To find the relationship between the opinion arguments and the represented issues, factors, and facts in ACTs, this chapterl investigates how these ACTs can be merged and expressed in terms of ADFs, where the nodes represent factors, the edges reflect the support between the factors and the acceptance conditions are defined to determine the accepted nodes (argument components).

The outline for this chapter is as follows. A general overview of ADFs and their role in designing knowledge-based legal systems is discussed in Section 5.1. ANGELIC is presented in Section 5.2 explaining the approach followed in using ADFs for legal reasoning. Section 5.3 provides an overview of the factors and the factor hierarchies in CATO and IBP in order to explain how ADFs encapsulate the factor hierarchy of CATO, showing the natural mapping of the statements and links from the factor hierarchy to the ADF in Section 4. Next, a methodology for mapping from Oral Hearings ACTs to ADF is provided in Section 5.5 and the chapter concludes with a summary in Section 5.6.

5.1 Abstract Dialectical Frameworks Role in AI and Law

As discussed in Chapter 2, Abstract Dialectical Frameworks were introduced in [54] and revisited in [53]. ADFs provide a generalisation of Dung's abstract argumentation frameworks (AFs) [61]. ADFs, like AFs, consist of a set of nodes and directed links between them, but whereas the links in an AF have a uniform interpretation, namely defeat, the links in an ADF can be given a variety of interpretations. Moreover, in ADFs the nodes are statements in general, rather than specifically abstract arguments. ADFs also generalise bipolar argumentation frameworks [56] which introduce a second relationship (support) in addition to the attack relationship of AFs. However, the ADF goes further and allows any type of link to be used. Although the acceptance conditions are often expressed as propositional functions, this need not be the case: all that is required is the specification of conditions for the acceptance or rejection of a node in terms of the acceptance or rejection of its children.

Partitioning will be used to divide the links into supporting and attacking links, but will continue to see the acceptance conditions as specifying preferences locally rather than adopting a global ordering of nodes[1].

ADFs can both drive and record the design of the knowledge base for a system to apply a body of case law. In particular the otherwise monolithic rule base is modularised by being distributed as the acceptance conditions of nodes. Good modularisation - tightly coherent and loosely coupled - is an essential feature of good program design (see e.g. [112]). The modularisation of the knowledge base achieved by using an ADF is indeed tightly coherent, since each set of rules is concerned only with the acceptance of a single statement, and contains all the rules needed to decide the acceptance of that statement. Loose coupling is achieved by limiting the components required to determine the acceptability of a node to the children of that node.

Knowledge engineering advantages of ADFs include, in addition to the effective modularisation of the system:

- Effective partitioning of the problem space, which limits the number of precedents required to determine the outcome of cases.
- Ready visualisation of the possible paths through a Prolog program: the connection between sets of rules is readily visible from the structure, whereas tracing which rules are invoked when a given rule is executed from a monolitic rule base of any reasonable size can often be difficult and error-prone.
- Assurance of completeness of the rule base, since it can readily be seen that each non-leaf node has its own acceptance condition.
- Straightforward inclusion of additional nodes. Addition of nodes can be performed without fear that there will be unanticipated ramifications in the knowledge base.
- Awareness of what nodes will be affected by the removal (or modification) of a node.

[1]Whereas PADFs were designed specifically to reflect Preference-Based Frameworks [12], the approach taken here reflects Value-Based Frameworks [28], which are more commonly used in AI and Law. The relationship between Preference- and Value-Based Frameworks is formally characterised in [99]

- Support for verification: each node can be considered, either by reference to precedents or otherwise, in isolation, on its own merits.
- Neutrality and integration between frame-based and rule representations. The node structure provides the former and the acceptance conditions the latter.
- Division of labour across a team of implementors: realisation of the part of the program which decides one node can be done independently of the realisation of other nodes; how the fragments will link together is explicitly specified. The children determine the inputs required by each module (node): that is, how the modules interface with each other.

All of these things are highly desirable when designing and developing a system of any real size and substance to reason with legal cases. The lack of such support has been a significant barrier to the uptake of such systems in practice (see, e.g., [48] for the difficulties in developing a sizable knowledge base to represent case law without effective design support).

5.2 The ANGELIC Methodology

From the ADF definition, it is clear that there is similarity in structure between the ADF, the factor hierarchy in CATO and Argument Component Trees (ACTs). Moreover, the additional flexibility that ADFs provide over AFs allows a more natural representation than has been possible in AFs, especially since there are both pro and con reasons.

This section presents ANGELIC, an ADF knoweldge encapsulation approach for legal information from cases. Instantiations of ADFs both for factor hierarchies and ACTs will be created in order to encapsulate and apply the theories that the hierarchies represent in a formal representation of knowledge in the legal domain. The methodology described here goes through three iterated phases:

Domain analysis and representation: The ANGELIC methodology is applied to three domains. Recognising that the case law of a legal domain goes through a life cycle of three stages as stated by Levi [88], this methodology is intended to be applied to the domains during the second stage where there is a period of stability and the theory is settled.

The first starts with the use of the existing knowledge of US Trade Secrets in CATO and IBP as shown in Section 5.4. Second, in Chapter 6, ADFs in the domain of the Wild Animals cases will be constructed (*Popov v Hayashi* and related cases as modelled in [30]), for which several different representations exist. Finally, a set of cases are used from the Automobile Exception to The Fourth Amendment, which has been discussed in the literature in [115], [76] and [29], but has only been partially represented to illustrate particular points in these papers. Therefore, there will be examples of several variant analyses and a new analysis specifically produced for this thesis. Figure 5.1 shows the two approaches used in analysing and representing these domains by instantiating ADFs from different sources.

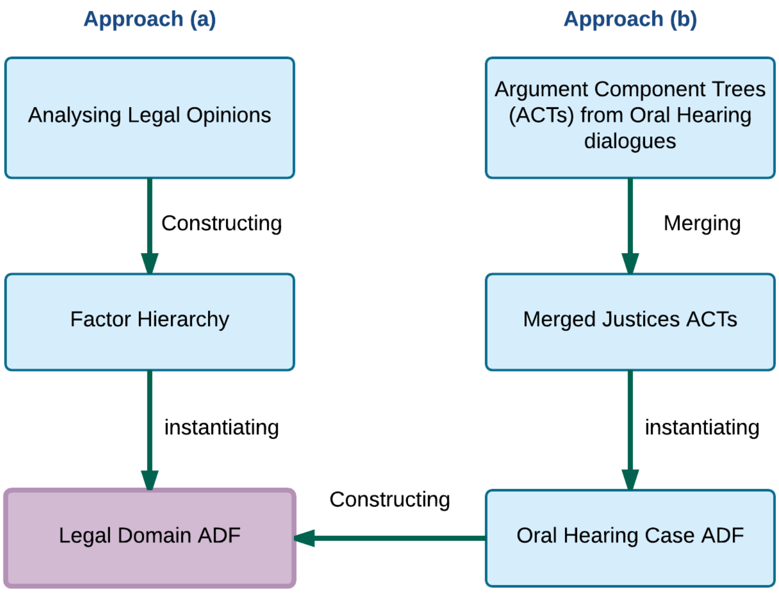


FIGURE 5.1: Domain analysis and representation Phase: Different Starting Points for ADF instantiation.

Implementation: Next, a move from analysis to an executable program is done in a direct and immediate way using Prolog, by translating the acceptance conditions into Prolog procedures, adding some control and reporting information, and then executing the resulting program. Comparison of the program outcomes with the actual outcomes of the cases allows the theory to be refined, exploiting the software engineering benefits afforded by the formal representation.

Testing and Refinements: For each domain, an evaluation is conducted to test the ease of implementation, the performance and efficacy of the resulting program, the ease of refinement of the program and the transparency of the reasoning. This yields further insights as to how the case facts can be incorporated and how the justification and reasoning can be improved using portions of precedents [48]. The implementation and testing phases are discussed in the next chapter.

5.3 Factor-Based Reasoning Using ADFs

A number of landmark case-based legal reasoning systems were discussed in Chapter 2, including CATO and IBP. This section proposes how the factor hierarchy of CATO and the logical model of IBP can be used to instantiate an ADF for the domain of US Trade Secrets.

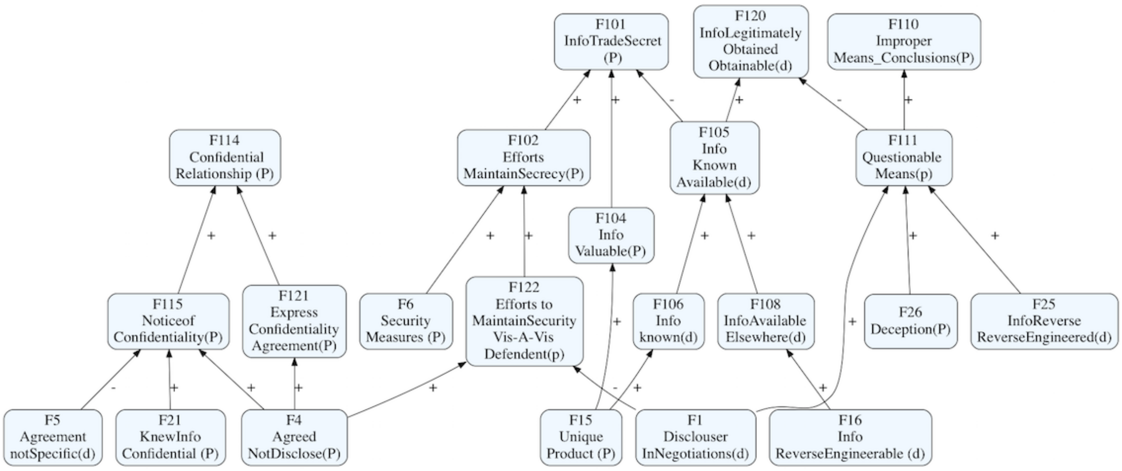


FIGURE 5.2: CATO Abstract Factor Hierarchy (reproduced from [10])

5.3.1 CATO

CATO [10], which was based on Rissland and Ashley's HYPO, most fully described in [14], takes as its domain US Trade Secrets Law. CATO was primarily directed at law school students, and was intended to help them form better case-based arguments, in particular to improve their skills in distinguishing cases, and emphasising and downplaying distinctions. A core idea was to describe cases in terms of factors, legally significant abstractions of patterns of facts found in the cases, and to build these *base-level factors* into a hierarchy of increasing abstraction, moving upwards through intermediate concerns (abstract factors) to issues. An extract from the factor hierarchy, showing details of the support and attack relationships between the factors, is shown in Figure 5.2 and the complete hierarchy is shown Figure 5.3. The Figures have been reproduced directly from [10]. Each factor favours either the plaintiff or the defendant[2]. The CATO program matches precedent cases with a current case to produce arguments in three plies:

- First a precedent with factors in common with the case under consideration is cited, suggesting a finding for one side.
- Then the other side cites precedents with factors in common with the current case but a decision for the other side as counter-examples, and distinguishes the cited precedent by pointing to factors not shared by the precedent and current case.
- Finally, the original side rebuts by downplaying distinctions, citing cases to prove that weaknesses are not fatal and distinguishing counter-examples.

[2]Note that the domain analysis in Chapter 5 and the forthcoming chapters refer to the parties as plaintiff and defendant (rather than, for example, petitioner and respondent) following the longstanding practice in AI and law.

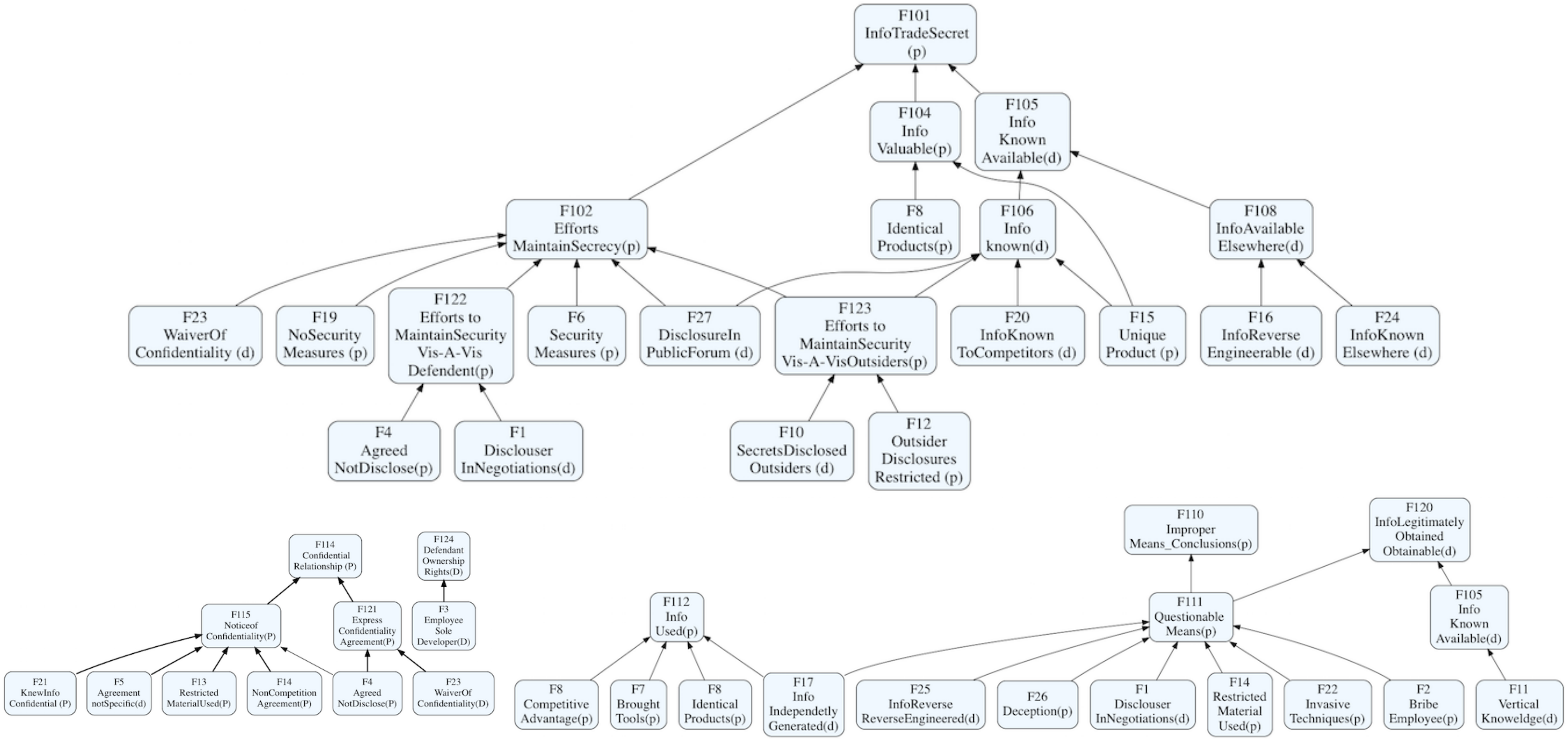


FIGURE 5.3: CATO Abstract Factor Hierarchy (reproduced from [10])

CATO uses twenty-six base level factors (there is no F9), as shown in Table 5.1.

ID	Factor
F1	DisclosureInNegotiations (d)
F2	BribeEmployee (p)
F3	EmployeeSoleDeveloper (d)
F4	AgreedNotToDisclose (p)
F5	AgreementNotSpecific (d)
F6	SecurityMeasures (p)
F7	BroughtTools (p)
F8	CompetitiveAdvantage (p)
F10	SecretsDisclosedOutsiders (d)
F11	VerticalKnowledge (d)
F12	OutsiderDisclosuresRestricted (p)
F13	NoncompetitionAgreement (p)
F14	RestrictedMaterialsUsed (p)
F15	UniqueProduct (p)
F16	InfoReverseEngineerable (d)
F17	InfoIndependentlyGenerated (d)
F18	IdenticalProducts (p)
F19	NoSecurityMeasures (d)
F20	InfoKnownToCompetitors (d)
F21	KnewInfoConfidential (p)
F22	InvasiveTechniques (p)
F23	WaiverOfConfidentiality (d)
F24	InfoObtainableElsewhere (d)
F25	InfoReverseEngineered (d)
F26	Deception (p)
F27	DisclosureInPublicForum (d)

TABLE 5.1: Base Level Factors in CATO

There is, however, no single root for the factor hierarchy as presented in [10]: rather there is a collection of hierarchies, each relating to a specific issue. To tie them together, the Issue Based Prediction (IBP) system of Bruninghaus and Ashley [55], which is also based on the US Tade Secrets domain, has been used as shown in the next sub-section.

5.3.2 IBP

In IBP, which is firmly based on CATO, the aim is not simply to discover and present arguments, but to predict the outcomes of cases. To enable this, the issues of CATO's hierarchy are tied together using a logical model derived from the *Uniform Trade Secret Act*, which has been adopted by the majority of States in the US, and the *Restatement of Torts*. The model is shown in Figure 5.4.

IBP used 186 cases in its very thorough empirical evaluation [55]; 148 cases were analysed for CATO and 38 were analysed specifically for IBP. Unfortunately, the analyses of these cases are not all publicly available and so 32 cases have been used in this thesis. These cases were

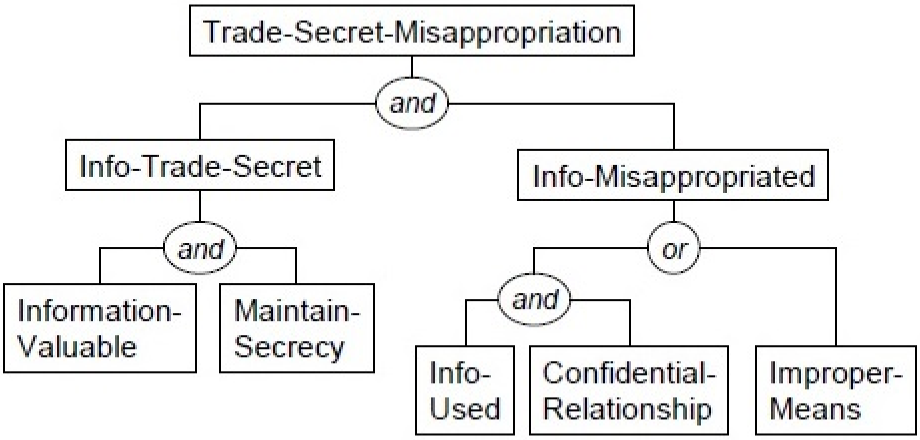


FIGURE 5.4: IBP Logical Model From [55]

harvested from public sources by Alison Chorley and used to evaluate her AGATHA system [57, 58]. As part of the evaluation in [55], nine other systems were also considered to provide a comparison. Most of these were different forms of machine learning systems, but programs representing CATO and HYPO were also included. IBP was the best performer: results are reported in [55] for IBP, Naive Bayes (the best performer of the ML systems), CATO, HYPO and a version of IBP which uses only the model, with no CBR component. This comparison will be revisited in Chapter 7.

5.4 Abstract Dialectical Frameworks For Factor Hierarchy

Now, consider the CATO and IBP factor hierarchies, as shown in Figures 5.2, 5.3 and 5.4. An ADF for capturing the legal knowledge can now be instantiated as represented by these factor hierarchies as follows:

Statements The statements S are the set of all the issues, intermediate concerns and base-level factors.

Links is a subset of $S \times S$, a set of links where $L+$ form the supporting links labelled "$+$" and $L-$ form the attacking links labelled "$-$"

Acceptance Conditions For each abstract factor (non-leaf node), a number of acceptance conditions are defined in terms of their supporting and attacking children. Each acceptance condition is constructed from the factors in the legal domain potentially based on the decisions in precedent cases. Often, however, the nature of the relationship is clear; that precedents would not always be required to resolve the comparisons was recognised in [10].

> "for certain conflicts, it is self-evident how they should be resolved. ... It is not necessary to look to past cases to support that point." (p47).

The acceptance conditions are expressed as a set of tests. The order of the tests expresses preferences. If none are satisfied, the node (abstract factor) is assigned to a default value.

Although [54], [51] and [53] have shown ADFs to be formally sound and their formal properties have been investigated, the ADFs explained here have not explicitly used these properties. They are rather straightforward, since for these domains the ADFs are cycle-free. As such, the grounded, preferred and stable semantics coincide. Moreover, since the input cases will determine the labelling of base-level factors (the leaf nodes), the nodes of the ADFs can be completely and unambiguously labelled within the Prolog program to find the grounded extension. This is important since it means that programs based on a design in the form of a cycle-free ADF are themselves tractably computable.

5.4.1 US Trade Secrets Domain

Using the complete factor hierarchy of [10] given in Figure 5.3, an ADF has been produced having as its leaf nodes the base-level factors of CATO. This is described in tabular form in Table 5.2.

ID	*S*	*L*+	*L*−
F102	EffortstoMaintainSecrecy	F6, F122, F123	F19, F23, F27
F104	InfoValuable	F8, F15	F105
F105	InfoKnownOrAvailable	F106, F108	
F106	InfoKnown	F20, F27	F15, F123
F108	InfoAvailableElsewhere	F16, F24	
F110	ImproperMeans	F111	F120
F111	QuestionableMeans	F2, F14, F22, F26	F1, F17, F25
F112	InfoUsed	F7, F8, F18	F17
F114	ConfidentialRelationship	F115, F121	
F115	NoticeOfConfidentiality	F4, F13, F14, F21	F5, F23
F120	LegitimatelyObtainable	F105	F111
F121	ConfidentialityAgreement	F4	F23
F122	MaintainSecrecyDefendant	F4	F1
F123	MaintainSecrecyOutsiders	F12	F10
F124	DefendantOwnershipRights	F3	

TABLE 5.2: CATO as ADF

The roots of CATO's hierarchies correspond to the leaves of the IBP logical model: these factors can be therefore formed into a single ADF by using this structure. The relevant additions to the ADF needed to integrate the IBP model are shown in Table 5.3 (note that F124 is not discussed in [55]).

In Table 5.2 and Table 5.3 there are eighteen nodes to provide with acceptance conditions. One (F124) has only a single supporting child: thus the acceptance condition will be *Parent* $\longleftrightarrow$ *Child*. This is written (as are the other acceptance conditions) as a set of tests for acceptance and rejection, to be applied in the order given, which allows expression of the priority between

ID	*S*	*L+*	*L−*
F200	TradeSecretMisappropriation	F201,F203	F124
F201	Info-Miasappropriated	F110,F112,F114	
F203	Info-Trade-Secret	F102,F104	

TABLE 5.3: IBP Logical Model as an ADF

them. This form of expression is easier to read in many cases, because it corresponds directly to the defeasible rules with priorities used in formalisms such as ASPIC+ [105], and because it is directly usable as Prolog code. The last test will always be a default. Thus *Parent* $\longleftrightarrow$ *Child* is written as

```
Accept Parent if Child.
Reject Parent.
```

Where NOT is required negation as failure is used. The tests are individually sufficient and collectively necessary, so that the closed world assumption can be applied and ensuring equivalence with the logical expression (corresponding with [59]). Six nodes (F201, F203, F105, F108, F114 and F124) have only supporting links: these can be straightforwardly represented using AND and OR. AND is used for two nodes (F201 and F203), as in the IBP model, and OR for the other four. The most complicated is InfoMisappropriated (F201):

```
Accept InfoMisappropriated if F114 AND F112.
Accept InfoMisappropriated if F110.
Reject InfoMisappropriated.
```

Five nodes have one supporting and one attacking link. These are best seen as forming an exception structure: accept (reject) the parent if and only if supporting (attacking) child unless attacking (supporting) child. Note that the exception may be the supporting *or* the attacking child: in the former case the default will be *reject*, and in the latter the default will be *accept*. Thus:

```
Accept Parent if Supporter AND (NOT Attacker).
Reject Parent.
OR
Reject Parent if Attacker AND (NOT Supporter).
Accept Parent.
```

For F110, F120 and F121 the attacking child is the exception, while for F122 and F123 the supporting children are the exceptions. This leaves seven nodes. For F200 the attacking link is regarded as an exception to the case where the conjunction of the supporting links holds:

```
Accept Trade Secret Misappropriation if
Info Trade Secret AND
Info Misappropriated AND
(NOT Defendant Ownership Rights).
Reject Trade Secret Misappropriation.
```

For F104 and F112 the supporting links offer disjoint ways of accepting the parent, and the attacking child is a way of establishing that the factor is not present. The default is defined to be *accept* because in many of the cases there are no factors for either side present relating to this point, and yet the abstract factor is required to be present. This factor was often simply accepted on the facts and uncontested, and so there was no discussion of the point, therefore no factors were mentioned in the decision. Thus for F112:

```
Accept F112 if F18.
Accept F112 if F8.
Accept F112 if F7
Reject F112 if F17.
Accept F112.
```

The remaining four are more complicated because they involve more factors. This might be where some reasoning with precedents is required, but here (because a very limited number of precedents are available) the approach is to make an initial attempt to supply tests, and remain prepared to adjust these in the light of particular precedents where testing show this to be necessary. For F106 (*InfoKnown*):

```
Accept F106 if F20.
Accept F106 if F27 AND (NOT F15).
Accept F106 if F27 AND (NOT F123).
Reject F106.
```

The rationale is that if the information is known to competitors, it is known, but even if it is disclosed in a public forum, the uniqueness of the product can suggest that the disclosure had no impact (i.e. it was not sufficiently widely known), and so the secret remained effectively unknown (and so F24 (InformationObtainableElsewhere) is more appropriate). The third clause suggests that the public disclosure might be restricted (e.g. if the secret was disclosed in a court of law during a trade secrets hearing), so that the information may be known, but embargoed. For F115, each of the four supporting links (F4, F13, F14 and F21) is regarded as a distinct way of establishing notice of confidentiality. F23 (WaiverOfConfidentiality) is an exception to each of them whereas F5 (AgreementNotSpecific) is an exception only to F4 and F13, since the other two do not relate to agreements.

```
Reject F115 if F23,
Accept F115 if F21.
Accept F115 if F14.
Reject F115 if F5.
Accept F115 if F4.
Accept F115 if F13.
Reject F115.
```

Similarly for F111 (QuestionableMeans), the supporting links can be seen as four different ways in which questionable means can be established. The attacking links here seem like counter-claims rather than exceptions, and suggest that this node requires exploring precedents to identify preferences. However, as a first attempt they are regarded as three distinct ways of rejecting the claim. Thus seven clauses are defined, one for each factor, which are initially ordered as F25, F17, F22, F26, F14, F2, F1, to reflect the strong and weak links shown in CATO.

```
Reject F111 if F25.
Reject F111 if F17.
Accept F111 if F22.
Accept F111 if F26.
Accept F111 if F14.
Accept F111 if F2.
Reject F111 if F1.
Reject F111.
```

This leaves F102, EffortstoMaintainSecrecy, which has three supporting and three attacking links. F19 is applicable only if no security measures at all are taken, which suggests that it has priority. Similarly a waiver of confidentiality (F23) or disclosure in a public forum (F27) could be seen as negating any efforts to maintain secrecy, although F123 provides a possible exception to the latter. The remaining two supporting links are regarded as independent. Thus six clauses are defined, offering reasons to reject or accept, ordered F19, F23 (F27 and NOT F123), F6, F122, F123 with *reject* as the default.

```
Reject F102 if F19.
Reject F102 if F23.
Reject F102 if F27 AND (NOT F123).
Accept F102 if F6.
Accept F102 if F122.
Accept F102 if F123.
Reject F102.
```

5.5 Abstract Dialectical Frameworks from Oral Hearing ACTs

The previous section described how the ADF and factor-based reasoning can be related. This finding influences the investigation of providing a justification for the legal decision based on the ACTs constructed from the Oral Hearing dialogues. Recalling the outcome from the Oral Hearing dialogues analysis and representation, there are four ACTs produced for each legal case: Petitioner's and Justices' ACTs from the petitioner and rebuttal dialogues and respondent's and Justices' ACTs from the respondent's dialogue. These will set out the available facts, factors and issues, and possible linkages between them.

Based on the individual and collective goals of these dialogues, the petitioner and respondent ACTs reflect each party's goals to assert all the argument components that would persuade the Justices to provide arguments supporting their side. The Justices' ACTs, on the other hand, illustrate the assertions of nine Justices and aim to clarify all the conflicting components in the case.

The task now is to merge these alternatives to produce an answer for the current case as defined in the process below. In the US Supreme Court procedure, this is the role of the Justices' conference stage. Given the Justices' ACTs, the goal of mapping the Oral Hearing ACTs is to instantiate a Case ADF from the merged Justices' ACTs and integrate the Case ADF to the domain ADF. The domain ADF is first generated in chronological order starting from the Case ADF of a landmark case in that domain (in the Automobile Exception domain, the first ADF is created after Carroll and updated from every new Case ADF). For every new Case ADF, the domain is modified by the components and acceptance conditions introduced by the case: this will help in tracking the definition and modifications of the acceptance conditions. Furthermore, the source of every component and acceptance condition in the Case ADF is determined. Sources could be new law, Oral Hearings, a precedent case or hypothetical, so that the generated domain ADF provides comprehensive details about the components that construct the opinion arguments and their provenance.

1. Merge the Justices' petitioner ACT (PJACT) and Justices' respondent ACT (RJACT) into one ACT by mapping the components and relationships as shown below in the Carney case study.

2. Instantiate a Case ADF from the merged Justices' ACTs.

3. Ensure that the produced Case ADF shows all the possible acceptance conditions.

5.5.1 Merging Justices Oral Hearing ACTs

Before instantiating a Case ADF, a merge is conducted between the Justices' ACTs, as shown in the steps below, to encapsulate all the components and relationships that have been discussed by the Justices during the Oral Hearing dialogues. Using California v. Carney's Justices' ACTs from the petitioner and respondent dialogues (Appendix C: Figures C.2 and C.4), Table 5.4 and Figure 5.5 illustrate the merging approach, showing all the components from both Justices' ACTs (PJACT and RJACT). Each component is followed by the component's parent in parentheses. The merging process is performed by applying the following steps:

1. Mapping the similar components from the two petitioner and respondent Justices' ACTs directly into the new merged ACT.

2. Adding directly into the merged ACT any component found in only one of the Justices' ACTs.

3. Defining new components when required; for example, defining a new base factor when facts or a combination of facts are directly mapped to an abstract factor e.g. vehicle status.

4. Combining all the relationships from both Justices' ACTs into the merged ACT as shown in Figure 5.5.

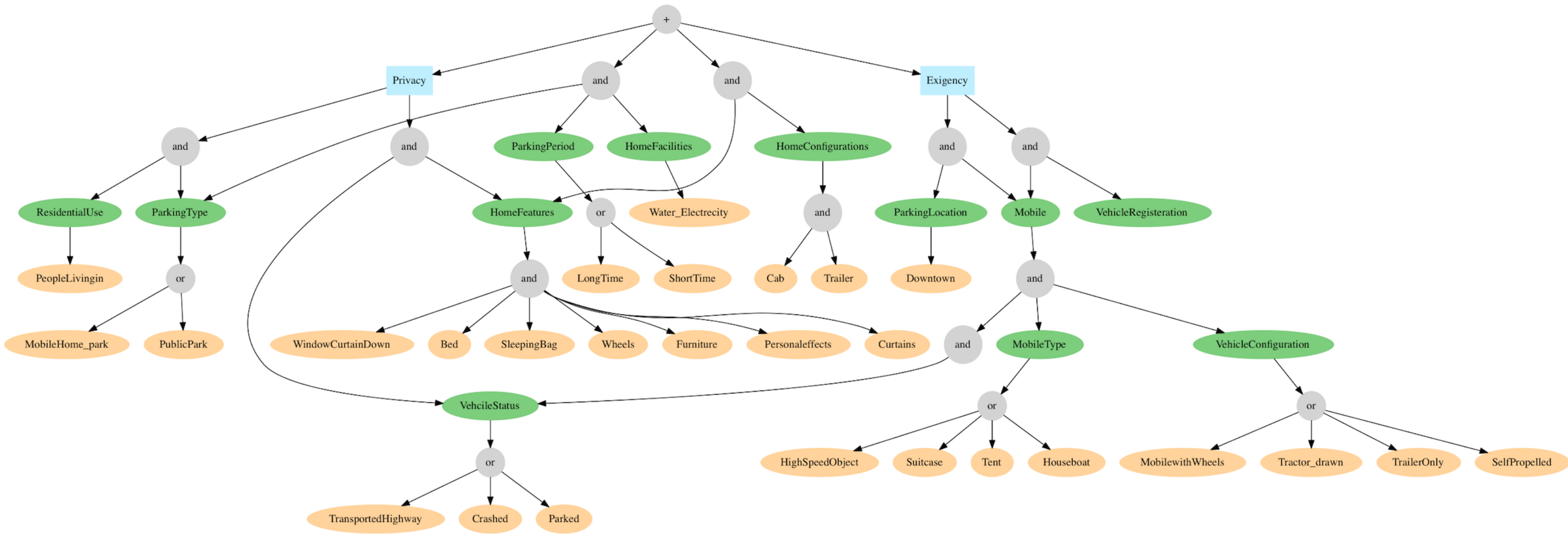


FIGURE 5.5: Merged Justices' ACT

Component	PJACTs	RJACTs	Merged Justices' ACT
Issue	Exigency	Exigency	Exigency
Issue	Privacy	Privacy	Privacy
Issue Relation	Exigency + Privacy	Exigency + Privacy	Exigency + Privacy
Abstract Factor	Mobile (Exigency)	Mobile (Exigency)	Mobile (Exigency)
Base Factor	VehicleRegistration (Exigency)		VehicleRegistration (Exigency)
Base Factor	MobileConfiguration (Mobile)	VehicleConfiguration (Mobile)	VehicleConfiguration (Mobile)
Base Factor	ParkingPeriod (Exigency + Privacy)		ParkingPeriod (Exigency + Privacy)
Base Factor		ParkingLocation (Exigency)	ParkingLocation (Exigency)
Base Factor	ParkingType (Exigency + Privacy)		ParkingType (Exigency + Privacy)
Base Factor	HomeFacilities (Exigency + Privacy)		HomeFacilities (Exigency + Privacy)
Base Factor		HomeConfiguration(Exigency + Privacy)	HomeConfiguration (Exigency + Privacy)
Base Factor		HomeFeatures(Exigency +Privacy)	HomeFeatures(Exigency +Privacy)
Base Factor	ResidentialUse (Privacy)		ResidentialUse (Privacy)
Base Factor		FactsRelation:TransportedVehicle OR CrashedCar OR ParkedVehicle(Mobile)	VehicleStatus(Mobile)
Base Factor	FactsRelation:HouseBoat OR OR Suitcase OR Tent (Mobile)	HighspeedObject (Mobile)	MobileType(Mobile)
Facts		TransportedVehicle,CrashedCar, ParkedVehicle(VehicleStatus)	TransportedVehicle,CrashedCar, ParkedVehicle(VehicleStatus)
Facts	HouseBoat, Suitcase, Tent (MobileType)	HighspeedObject (MobileType)	HouseBoat, Suitcase, Tent, HighSpeedObject (MobileType)
Facts	TrailernoTractor,TractorDrawn, SelfPropelled, Wheels (MobileConfiguration)	Wheels(VehicleConfiguration)	TrailernoTractor, TractorDrawn, SelfPropelled, Wheels (VehicleConfiguration)
Facts	Public, MobileHomePark (ParkingType)		Public, MobileHomePark (ParkingType)
Facts	LongTime, ShortTime (ParkingPeriod)		LongTime, ShortTime (ParkingPeriod)
Facts		Downtown (ParkingLocation)	Downtown (ParkingLocation)
Facts	WaterElectricity (HomeFacilities)		WaterElectricity (HomeFacilities)
Facts		Cab,SelfContainedUnit(Home Configuration)	Cab,SelfContainedUnit(HomeConfiguration)
Facts		HasCurtains, WindshieldCurtainsDown, Bed, SleepingBag, Furniture, PersonalEffects (HomeFeatures)	HasCurtains, WindshieldCurtainsDown, Bed, SleepingBag, Furniture, PersonalEffects (HomeFeatures)

TABLE 5.4: Merged Justices' Components

5.5.2 From Oral Hearing ACTs to Case ADF

The Case ADF, as shown in Table 5.5, is instantiated first from the merged Justices' ACTs in the Oral Hearing, then modified in order to select the acceptance conditions during the Justices' conference stage and finally represent the case opinion as shown in the next chapter. The Case ADF can be defined as follows:

Statements The statements S are the set of all the components: issues, factors and facts. These statements are subsets of the domain ADF.

Links is a subset of $S \times S$, a set of supporting links $L+$ and attacking links $L-$, which are determined based on the relationships between parent and children.

Acceptance Conditions For each issue, abstract factor and base factor, a number of acceptance conditions are defined from the component relationships in the merged ACT. All the relationships and components related to the undetermined relationships between the conflict issues (+) are captured in acceptance conditions related to each conflict issue. These acceptance conditions are used to determine all the possible opinions in deciding the case.

Merged Justices ACT	S	L+	L-
Exigency	Exigency	Mobile,VehicleRegistration PublicParkingLocation, PublicParkingType	HomeFeatures, HomeFacilities, ParkingLong Period
Privacy	Privacy	HomeConfiguration, ResidentialUse HomeFeatures, HomeFacilities, ParkingLongPeriod	PublicParking Location, Public ParkingType
Mobile	Mobile	VehicleConfiguration, MobileType, VehicleStatus	
HomeConfiguration	HomeConfiguration	Cab,SelfContainedUnit	
ResidentialUse	ResidentialUse	PeopleLivingIn	
VehicleRegistration	VehicleRegistration	SpecialLicence	
VehicleConfiguration	VehicleConfiguration	TractorDrawn, SelfPropelled,HighSpeedObject	TrailernoTractor
ParkingPeriod	ParkingLongPeriod	LongTime	ShortTime
ParkingLocation	PublicParkingLocation	Downtown	
ParkingType	PublicParkingType	PublicParking	MobileHomePark
HomeFacilities	HomeFacilities	Water, WaterElectricity	
HomeFeatures	HomeFeatures	Bed, SleepingBag, Furniture, PersonalEffects	
VehicleStatus	VehicleStatus	TransportedVehicle, ParkedVehicle	CrashedCar
MobileType	MobileType	HouseBoat	Tent,Suitcase

TABLE 5.5: California v. Carney Oral Hearing Case ADF

The acceptance conditions from all the possible relationships are shown below:

```
Exigency IF (Mobile AND (VehicleRegistration
        OR PublicLocation).
```

```
Exigency IF (Mobile AND ShortParkingTime AND
         PublicParking AND (NO) HomeFacilities).
Exigency IF (Mobile AND HomeConfiguration AND
         (NO) StorageCompartment).

Privacy IF  (MobileHome AND ResidentialUse
AND (NOT) PublicParking).
Privacy IF  (MobileHome  AND  Parked
AND HomeFeatures).
Privacy IF  (Mobile AND LongParkingTime AND
     MobileHomeParking AND HomeFacilities).
Privacy IF  (Mobile AND HomeConfiguration AND
     StorageCompartment).

Mobile IF  MobileType AND VehicleConfiguration
        OR VehicleReadyStatus

MobileType IF Automobile OR Houseboat OR Tent
OR Suitcase OR HighSpeedObject
VehicleConfiguration IF TractorDrawn OR TrailerOnly OR
SelfPropelled OR AutomobilewithWheels

VehicleStatus IF Transported_Highway OR Crashed OR Parked
VehicleRegistration IF Licence
ParkingPeriod IF LongTime OR ShortTime

ParkingType IF PublicParking OR MobileHomeParking
HomeFacilities IF Water AND Electricity
HomeConfiguration IF Cab AND Trailer
HomeFeatures IF Curtains OR CurtainsDown OR PersonalEffects
OR Bed OR SleepingBag

ResidentialUse IF PeopleLivingin
PublicParkingLocation IF DownTown
```

This Case ADF will be compared in Chapter 6 with the ADF instantiated from the case opinion in order to determine the role of the Oral Hearing analysis in deciding the case. The comparison will help in:

- Determining the argument components that have been derived from the Oral Hearing analysis and used to define the opinion argument.

- Specifying the precedent cases that have been cited in both the Oral Hearing and the Opinion.
- Understanding who proposed the components used for constructing the argument opinion using the petitioner and respondent ACTs.
- Tracking how all the acceptance conditions are initiated and modified.
- Providing further details about the sources of all the argument components, whether they were proposed as a fact in the case, a hypothetical component or a precedent case component.

5.6 Summary

ANGELIC, a knowledge engineering methodology for encapsulating knowledge from legal cases using ADFs has been proposed in this chapter. The first stage in the approach (domain analysis and representation) has been presented. First, a domain ADF was instantiated for the US Trade Secrets domain using CATO and IBP factor hierarchy, which in turn provided a solid formal basis for factor-based reasoning. Second, the ACTs constructed from the legal cases in Chapter 4 were merged to generate a Case ADF. Next, Chapter 6 will consider the remaining stages, the executable form of applying ADF to the various domains, including the results, and possible refinements.

Chapter 6

Applying ADFs to Predict the Outcome of Legal Cases

"His machine was never perfected, though it generated a whole field of research into what became known as "Turing Machines". Today we call them "computers."
Alan Turing, The Imitation Game

Following the design methodology introduced in the previous chapter, this chapter shows how ANGELIC has been applied in different legal domains to predict the outcome of legal cases described as sets of factors, according to a theory of a particular domain, based on a set of precedent cases relating to that domain. In Chapter 5, an ADF for the domain of US Trade Secrets was instantiated from the factor hierarchy of the well-known legal reasoning system CATO. The method is now applied to two other legal domains often used in the literature of AI and Law. In each domain, Wild Animals and Automobile Exception, the design is provided by the domain analyst expressing the cases in terms of factors organised into an ADF, from which an executable program can be implemented in a straightforward way by taking advantage of the closeness of the acceptance conditions of the ADF to components of an executable program. A number of evaluations are conducted to test the ease of implementation, performance and efficacy of the resulting program and the ease of refinement of the program. This chapter is structured as follows: Section 6.1 shows how an ADF is implemented and applied in different legal domains, presenting the results from the program and analysing the output from each domain. Next, a modification to the program in the Automobile Exception domain is applied, first by reasoning using the fact layer as proposed in Section 6.2, and then including new rules to consider the dissenting opinion in Section 6.3. After that, the relationship between the Oral Hearings and the Opinion is given in Section 6.4. Finally a summary of the chapter is provided in Section 6.5.

6.1 ADF Applications to Legal Domains

This section illustrates how the approach has been applied to several different domains that have often been used in AI and Law. First, US Trade Secrets is considered, using the 32 cases from [58]. Next, the work is extended to further domains: starting with a small-scale domain concerning Wild Animals (5 cases) before proceeding to the Automobile Exception to The Fourth Amendment (10 cases). Note that the aim is only to encapsulate (rather than learn) the theory of the applicable law. For this purpose, a limited number of case suffices (cf. HYPO, which used fewer than 25 cases [14]).

The program is implemented using Prolog, taking advantage of its closeness to the expression of the acceptance conditions, which makes the implementation quick, easy and transparent. The Prolog program was created by ascending the ADF, rewriting the acceptance conditions as groups of Prolog clauses to determine the acceptability of each node in terms of its children. This required restating the tests using the appropriate syntax, adding some reporting to indicate whether the node is satisfied (defaults are indicated by the use of "accepted that"), and some control to call the procedure to determine the next node, and to maintain a list of accepted factors.

6.1.1 US Trade Secrets Domain

Using the ADF instantiated from the factor hierarchy of [10] (Table 5.2 and Table 5.3) in the previous chapter, this section provides some examples from the implemented ADF as shown below.

ANGELIC Secrets Program

The program is executed by posing a query, which enables the first factor to be decided in the case and its base-level factors as arguments. Each factor is called in turn, with the factors present passed up as *FactorsSoFar*. The Prolog for F112 (for which the acceptance conditions are given below, and which will call the procedure to get F111) is:

```
Accept F112 if F18.
Accept F112 if F8.
Accept F112 if F7
Reject F112 if F17.
Accept F112.
```

```
getf112(Case,FactorsSoFar):-
member(f18,FactorsSoFar),
write([the,information,was,used]),
nl, getf111(Case,[f112|FactorsSoFar]).

getf112(Case,FactorsSoFar):-
member(f8,FactorsSoFar),
```

```
write([the,information,was,used]),
nl, getf111(Case,[f112|FactorsSoFar]).

getf112(Case,FactorsSoFar):-
member(f7,FactorsSoFar),
write([the,information,was,used]),
nl, getf111(Case,[f112|FactorsSoFar]).

getf112(Case,FactorsSoFar):-
member(f17,FactorsSoFar),
write([the,information,was,not,used]),
nl, getf111(Case,FactorsSoFar).

getf112(Case,FactorsSoFar):-
write([accepted,that,the,information,was,used]),
nl, getf111(Case,[f112|FactorsSoFar]).
```

Each of the four tests in the acceptance condition is applied in a separate clause, using the set of factors currently identified as present in the case, before proceeding to the next factor (F111), with F112 added to the applicable factors if it is accepted. To allow completion of the database [59], a final clause is added to catch any case not covered by any of the preceding clauses. Although the default is normally *reject*, as discussed above, these defaults may favour either side. In some cases the default should be *accept*, such as F112 and F104, because in most cases there are no factors in the case descriptions relating to these abstract factors, and yet they are a *sine qua non* for any claim. These aspects were uncontested and so the factors were not explicitly discussed in the trial, and so do not appear in the CATO analysis. Where it was clear that the factor needed to be explicitly established (e.g. F106 (InformationKnown) and F111(QuestionableMeans)) the default was *reject*. The code for F111 is:

```
getf111(C,Factors):-
member(f25,Factors),
write([questionable,means,were,not,used]),
nl,getf123(C,Factors).

getf111(C,Factors):-
member(f17,Factors),
write([questionable,means,were,not,used]),
nl, getf123(C,Factors).

getf111(C,Factors):-
member(f22,Factors),
write([questionable,means,were,used]),
```

```
nl, getf123(C,[f111|Factors]).

getf111(C,Factors):-
member(f26,Factors),
write([questionable,means,were,used]),
nl, getf123(C,[f111|Factors]).

getf111(C,Factors):-
member(f14,Factors),
write([questionable,means,were,used]),
nl, getf123(C,[f111|Factors]).

getf111(C,Factors):-
member(f2,Factors),
write([questionable,means,were,used]),
nl, getf123(C,[f111|Factors]).

getf111(C,Factors):-
member(f1,Factors),
write([questionable,means,were,not,used]),
nl,getf123(C,Factors).

getf111(C,Factors):-
write([accepted,that,questionable,means,were,not,used]),
nl, getf123(C,Factors).
```

The above demonstrates that it is a straightforward and reasonably objective process to transform a factor-based analysis such as is found in [10] to an executable program via an ADF. Although judgment is sometimes required to form the acceptance conditions, such judgments are not difficult to make. Moreover if there are difficult choices, the effect of the alternatives can be compared using a set of test cases. Overall, the relatively small number of factors relevant to particular nodes greatly simplifies the task.

Results

The program was run on the domain cases as shown in Appendix F. The cases are represented as a list of base-level factors. For example, the Boeing case[1] is represented as

```
case(boeing,[f4,f6,f12,f14,f21,f1,f10]).
```

giving output:

[1]The Boeing Company v. Sierracin Corporation, 108 Wash.2d 38, 738 P.2d 665 (1987).

```
1 ?- go(boeing).
accepted that defendant is not owner of secret
efforts made vis a vis outsiders
efforts made vis a vis defendant
there was a confidentiality agreement
defendant was on notice of confidentiality
there was a confidential relationship
accepted that the information was used
questionable means were used
accepted that the information
    was not available elsewhere
accepted that information is not known
accepted that the information
    was neither known nor available
accepted that the information was valuable
not accepted that the information
    was legitimately obtained
improper means were used
efforts were taken to maintain secrecy
information was a trade secret
a trade secret was misappropriated
find for plaintiff
boeing[f200, f201, f203, f102,f110,f104,f111,
f112,f114,f115,f121,f122,f123,
f4,f6,f12,f14,f21,f1,f10]
decision is correct in accordance with the actual decision
```

The initial program correctly classified 25 out of the 32 cases (78.1%) of the cases. While all ten of the cases won by the defendant were correctly classified, seven of the 22 cases won by the plaintiff were not. The figure for correct answers is remarkably close to the 77.8% reported for the version of CATO used in [55], which, of course, used exactly the same analysis of the domain and cases that has been adopted here. Thus the first conclusion suggests that executing the analysis in [10] as an ADF produces very similar results to those obtained using the original CATO program (albeit a smaller set of cases is used here). Further investigation was conducted to find how the initial program might be improved. The wrongly predicted cases were:

```
case(spaceAero,[f8,f15,f18,f1,f19]).
case(televation,[f6,f12,f15,f18,f21,f10,f16]).
case(goldberg,[f1,f10,f21,f27]).
case(kg,[f6,f14,f15,f18,f21,f16,f25]).
case(mason,[f6,f15,f21,f1,f16]).
case(mineralDeposits,[f18,f1,f16,f25]).
case(technicon,[f6,f12,f14,f21,f10,f16,f25]).
```

Examination of the cases showed that five of the seven had F16 (ReverseEngineerable) present and that these cases were the only cases found for the plaintiff with F16 present. The problem in these five cases is that the program finds for the defendant because the information is available elsewhere (F108). This is established by the presence of ReverseEngineerable and is unchallengeable. Examination of the ADF shows that F16 is immediately decisive: if that factor is present, there is no way the plaintiff can demonstrate that the information is a trade secret. *Goldberg*[2] also fails through F105 (InformationKnownOrAvailable), since disclosure in a public forum (F27) is sufficient to deny the information trade secret status. It would appear that the performance could be significantly improved by refining this branch to allow the plaintiff some way to defend against F16 in particular.

Refinement

At this point it should be noted that CATO is likely to be more robust in the face of imperfect analysis than an approach based on a logical model. Because CATO generates arguments based on considering all the available factors taken together, it is less likely to have an outcome determined by a single factor than a logical model. For example, the presence of F16 or F24 can be seen to immediately determine a decision for the defendant in a logical model, whereas in CATO other factors might outweigh them. Moreover, CATO was designed to assist law students, not to predict outcomes. Similar problems should be expected to arise in IBP, which also uses a logical model, albeit one that is applied at a later stage of the process. In [55] it is stated

> We found that some Factors, called KO-Factors (or Knockout Factors), almost always dominate the outcome of a case. For instance, as an empirical matter, the plaintiff will not win a case with Factor F20, Info-Known.

Such factors are given special treatment in IBP, and so it does not seem unreasonable to use the initial results to suggest possible refinements to the original analysis. First consider *Goldberg v Medtronic*; in that decision it is explicitly stated that

> The district court found that Medtronic could not avoid its obligation of confidence due to the availability of lawful means of obtaining the concept when those means were not employed. We affirm.

Thus the factor which was decisive[3] for the program, F27, was in fact explicitly held to be insufficient in the actual decision. It is not within the remit of the knowledge-base designer to say whether this decision was correct or not, but it does explain why the program misclassifies the case. Assuming the decision to be correct, F27 should either be redefined to include the defendant's actual *use* of this public domain knowledge, so that it is not present in *Goldberg*, allow F21 (KnewInformationConfidential) as an exception to F27 in determining the acceptance of F106, on the grounds that if the defendant believed the information to be confidential, he could

[2] Goldberg v. Medtronic, 686 F.2d 1219 (7th Cir. 1982).
[3] Running a version of *Goldberg* without F27 finds for the plaintiff.

not have been aware that the information was publicly available. Since there are no other cases with F21 and F27 both present, these two solutions cannot be chosen based on the precedents available.

Now turning to the problem created by F16, ReverseEngineerable, in [10] the note on the applicability of this factor states:

> The factor applies if: Plaintiff's information could be ascertained by reverse engineering, that is, by inspecting or analyzing plaintiff's product (regardless of whether defendant actually obtained the information in this way).

Thus it is clear that the defence of Goldberg cannot be used; that the defendant did not in fact reverse engineer the information. Nonetheless, the ease with which the product could be reverse engineered does (among other things) need to be considered. In Mason[4] it is stated:

> In this regard, we note that courts have protected information as a trade secret despite evidence that such information could be easily duplicated by others competent in the given field.

citing *KFC v Marion Kay*[5] and *Sperry Rand v Rothlein*[6]. The KFC decision cited in *Mason*[7] states

> Marion-Kay maintains that the recipes and formulas for the making of KFC seasoning are not unique and that Marion-Kay is capable, both financially and technically, of producing KFC seasoning.

This suggests that the uniqueness of the product (F15) might be a factor capable of attacking the acceptability of F108 as well as F106 (as identified in CATO). Adding F15 as an exception to F16 would give the correct decision in *Televation*[8], *KG*[9] and *Mason*. In *Technicon*[10] the phrase "readily ascertainable" is used:

> Curtis claimed that Technicon's trade secrets were "readily ascertainable" and that the company had not made reasonable attempts to ensure its trade secrets. The Court reasoned that Bridgmon's "wiretap" process had required over two-thousand hours, and still had not yielded a fully functional product. The Court held that this amount of time indicated that a trade secret was not readily ascertainable.

In fact in two of the cases (*KG* and *Technicon*) restricted material was used by the defendants, strongly implying that the information was not, in fact, readily ascertainable. In *Mineral Deposits*[11], it was found that

[4] Mason v. Jack Daniel Distillery, 518 So. 2d 130 (Ala. Civ. App. 1987)
[5] KFC Corp. v. MarionKay Co., Inc., 620 F.Supp 1160 (D.C.Ind 1985)
[6] Sperry Rand Corp. v. Rothlein, 241 F.Supp. 549 (D.Conn.1964)
[7] Mason v. Jack Daniel Distillery, 518 So.2d 130 (Ala.Civ.App.1987)
[8] Televation Telecommunication Systems, Inc. v. Saindon, 522 N.E.2d 1359 (Ill.App. 2 Dist. 1988)
[9] K & G Oil Tool & Service Co. v. G & G Fishing Tool Serv., 314 S.W.2d 782
[10] Technicon Data Systems Corp. v. Curtis 1000, Inc., 224 U.S.P.Q. 286
[11] Mineral Deposits Ltd. v. Zigan, 773 P.2d 606 (Colo.App.1988)

> After Zigan received the spiral, he removed the label which indicated that patent applications were pending and gave the spiral to defendant Zbikowski. Zbikowski then cut the spiral into pieces, made molds of the components, and proceeded to manufacture copies of the spiral. If a trade secret is divulged under an express or implied restriction of nondisclosure or nonuse, a party who engages in unauthorized use of the information will be liable in damages to the owner of the trade secret.

This strongly suggests that F14 was also in fact present in this case. Moreover in *Televation*, whether the secret counted as reverse-engineerable was contested:

> The mere fact, however, that a competitor could, through reverse engineering, duplicate plaintiff's product does not preclude a finding that plaintiff's techniques or schematics were trade secrets, particularly where, as here, the evidence demonstrated that the reverse engineering process would be time-consuming.

There is a strong suggestion that the court believed that copies of the plaintiff's drawings had, in fact, been used by the defendant, which would mean that F14 would apply. Finally *Sperry Rand*, another decision cited in *Mason*, states

> The defendants claim that there is no trade secret if it is disclosed by prior art or if it is readily discernible by others skilled in the field. It is no defense in an action of this kind that the process in question could have been developed independently, without resort to information gleaned from the confidential relationship. As stated in the landmark case of Tabor v. Hoffman, 118 N.Y. 30, 35, 23 N.E. 12, 13 (1889): "Even if resort to the patterns of the plaintiff was more of a convenience than a necessity, still if there was a secret, it belonged to him, and the defendant had no right to obtain it by unfair means, or to use it after it was thus obtained."

suggesting that the use of any kind of questionable means (rather than just F14) could be used to block a defence relying on reverse engineerability. The decisions thus give a number of suggestions for exceptions to F16 as a support for F108; especially uniqueness of the product and use of restricted materials. Incorporating those exceptions would raise the success of the program to 29 out of 32 (90.6%), and removing F27 from *Goldberg* (or allowing F21 as an exception) and adding F14 to *Mineral Deposits*, both of which seem eminently justifiable on the facts of the cases concerned as stated in the decision texts, would give correct decisions in these cases also (96.8%). This leaves only *Space Aero*[12] as an unexplained failure. The output for this case is:

```
?- go(spaceAero).
accepted that defendant is not owner of secret
accepted that efforts made vis a vis outsiders
no efforts made vis a vis defendant
```

[12] Space Aero Products Co. v. R.E. Darling Co., 238 Md. 93, 208 A.2d 74 (1965)

```
accepted that there was no confidentiality agreement
accepted that defendant was not on notice of confidentiality
accepted that there was no confidential relationship
the information was used questionable means were not used
accepted that the information was not available elsewhere
accepted that information is not known
accepted that the information was neither known nor available
the information was valuable
not accepted that the information was legitimately obtained
accepted that improper means were not used
no efforts were taken to maintain secrecy
information was not accepted as a trade secret
no trade secret was misappropriated
find for defendant
spaceAero[f104,f112,f123,
f8,f15,f18,f1,f19]
decision is wrong
```

This case fails on two branches: the information is not a trade secret because no security measures were taken, and because it appears that no confidential relationship existed. A key feature of this case is that the defendants were former employees of the plaintiff, and had been provided with the disputed information when employed by the plaintiff because they needed it to carry out their duties. The decision itself states

> The testimony, taken as a whole, convinces that Darling took precautions to guard the secrecy of its process which, under the circumstances, were reasonably sufficient.

This suggests that F19 (No Security Measures) was not, in fact, accepted as present, and removing this factor from the case is enough to establish that there was a trade secret. Turning to the issue of confidentiality, it is mentioned that

> While none of the former employees had signed a contract with Darling in which they formally agreed not to use the information acquired by them, and while they were free to leave their employment at will, Judge Pugh found that they owed the duty of fidelity to their employer while they were employed. We agree. ... The court below found as a fact that some of the former employees had in their possession, after leaving Darling's employment, certain sketches of oxygen breathing hoses which they had taken while they were employed by Darling, without Darling's knowledge. ... the former employees knew that they were acting wrongfully in violation of their confidential relationship and their duty of loyalty. We agree with the court below that the former employees violated the duty of fidelity and trust which they owed to Darling in respect of the trade secret and that their conduct was such as to entitle Darling to the protection of a court of equity.

Again, whether this decision was correctly made or not cannot be determined. However, it does seem that at least F21 (KnewInformationConfidential) should be included. If this is added, a confidential relationship can be established and the case found for the plaintiff.

6.1.2 Wild Animals Domain

The approach so far has been considered with respect to one single domain. If the approach is to be of general significance, however, it needs to be applicable to other domains. This subsection describes a further exercise designed to show that the approach is more generally applicable. The method is applied to another domain, which has often been used as an illustration of factor-based reasoning and as such analyses are available: the Wild Animals cases and *Popov v Hayashi*. The Wild Animals cases were introduced into AI and Law in [40] and extended to the baseball case of *Popov* in [144]. The factor-based analysis of [30] has been used as a starting point. Briefly the Wild Animals cases concern plaintiffs chasing Wild Animals when their pursuit was interrupted by the defendant. Post was chasing a fox for sport. Keeble was hunting ducks, Young was hunting fish and Ghen a whale, all in pursuit of their livelihoods. *Popov v Hayashi* concerned disputed ownership of a baseball (valuable because it had been hit by Barry Bonds to break a home run record). Popov had almost completed his catch when he was assaulted by a mob of fellow spectators and Hayashi (who had not taken part in the assault) ended up with the baseball when it came free. The Wild Animals cases were cited when considering whether Popov's had given him possession of the ball.

Thirteen base-level factors are identified in [30]. The first task is to form them (together with appropriate abstract factors) into a factor hierarchy, and to use this as the node and link structure to form an ADF. This factor hierarchy is shown in Figure 6.1: some adaptations to [30] have been made: for example a factor *Res* (Residence Status) is included to indicate the attachment of the animals to the land, since it appears to make a difference whether they are on that land permanently, seasonally, habitually, or occasionally. The nodes and links are given in Table 6.1. Acceptance conditions are supplied for the nine non-leaf nodes.

```
Possession for Plaintiff if (NOT) NoBlame
    AND ((Ownership
    OR (RightToPursue AND IllegalAct))

Ownership if (OwnsLand AND Resident)
    OR Convention OR Capture

Capture if (NOT) NotCaught OR (Vermin AND HotPursuit)

RightToPursue if OwnsLand OR
    ((HotPursuit AND PMotive
    AND (NOT) (better) DMotive)
```

```
PMotive if PLiving OR
    ((PSport OR PGain)
    AND (NOT) DLiving)

DMotive if NOT Malice AND
    (DLiving OR DSport OR DGain)

IllegalAct if Trespass OR Assault

Trespass if LegalOwner AND AntiSocial

AntiSocial if (Nuisance OR Impolite)
    AND (NOT) DLiving
```

S	L+	L-
Possession for Plaintiff	Ownership,RightToPursue,IllegalAct	NoBlame
Capture	HotPursuit, Vermin	NotCaught
Ownership	Convention, Capture, OwnsLand, Res	
PMotive	Pliving, PSport, PGain	DLiving
DMotive	DLiving, DSport, DGain	Malice
OwnsLand	LegalOwner	
RightToPursue	OwnsLand, Pmotive, HotPursuit	DMotive
AntiSocial	Nuisance, Impolite	DMotive
Trespass	LegalOwner, AntiSocial	
IllegalAct	Assault, Trespass	

TABLE 6.1: Wild Animals as ADF

The only real controversy here is over the determination of *Right to Pursue* when both the plaintiff and the defendant have good motives. Essentially, the choice is that if the land is not owned by one of them, the right to pursue is given to the party with the better motive. The remainder seem fairly uncontroversial.

ANGELIC Animals Program

The acceptance conditions can easily be expressed as Prolog procedures and then embedded in code as was done for CATO (Appendix G shows the Wild Animals domain program and results). The program can then be executed. For example, running the program for *Young v Hitchens* produces the following output (note that the program abbreviates factor names):

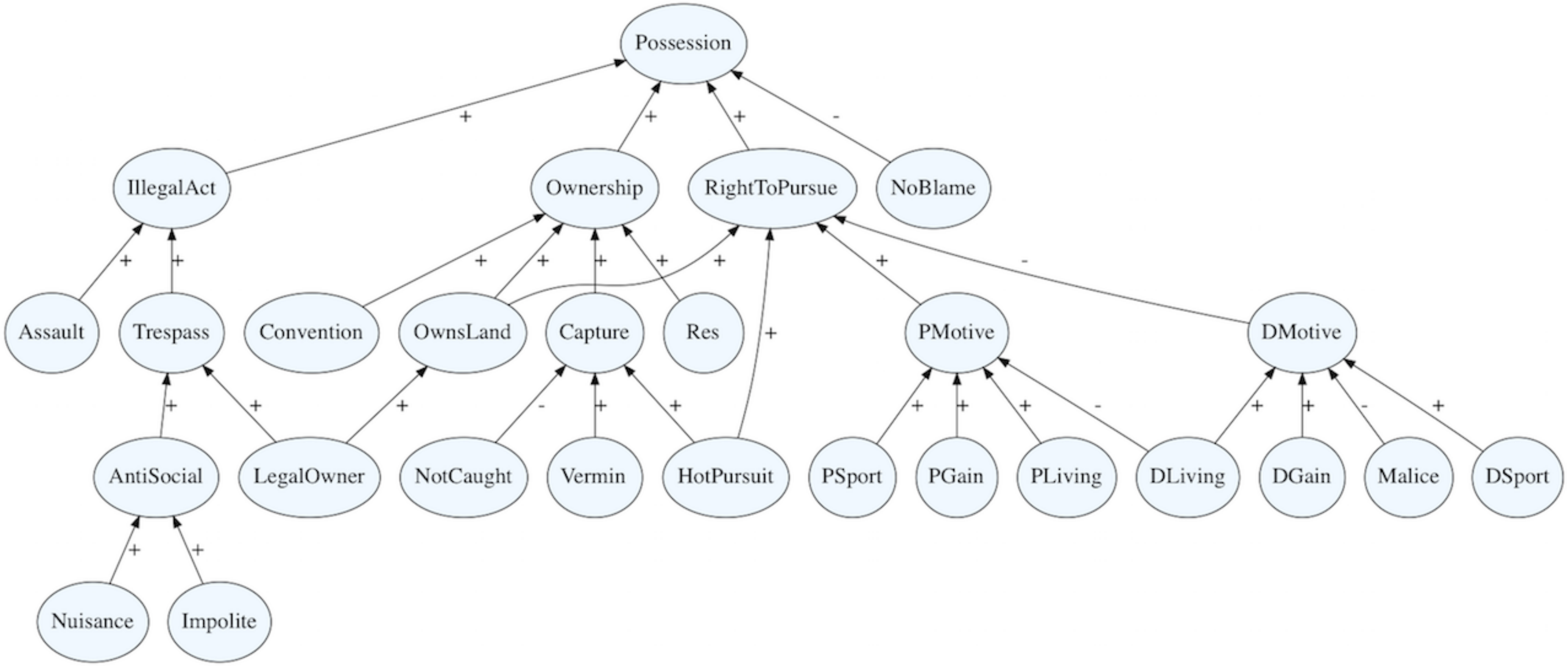


FIGURE 6.1: ADF for Wild Animals Domain

```
 ?- go(young).
the plaintiff had not captured the quarry
the plaintiff did not own the quarry
plaintiff has good motive
defendant has good motive
plaintiff did not own the land
plaintiff had a right to pursue the quarry
defendant committed no antisocial acts
defendant committed no trespass
no illegal act was committed
do not find for the plaintiff
find for the defendant
young[rtToPursue,dMotive,pMotive,
nc,hp,imp,pliv,dliv]
decision is correct in accordance with the actual decision
```

Results

The program produces correct results for all five cases discussed in [30]. This indicates that the ADF representation can be used to encapsulate the knowledge of the domain as represented in [30], suggesting that the method can be applied straightforwardly to a second domain. In general the method can be applied to any domain for which factor-based reasoning in the CATO (or HYPO or IBP) style is appropriate. This has encouraged application of the method to a larger-scale problem in the domain of the US Automobile Exception, for which there is no accepted analysis into factors available, and which therefore requires starting from the case decision texts, as shown in the next section.

6.1.3 US Automobile Exception to the Fourth Amendment Domain

Here the approach has been applied to ten freshly analysed cases in the domain of the Fourth Amendment, specifically in relating to the Automobile Exception. The Fourth Amendment protects the "right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures." A search is considered reasonable if a warrant has been obtained. However, when there is a high probability of losing the evidence so that there is an urgent reason to search, obtaining a warrant may become impossible. One such situation is a moving automobile. This domain thus considers the interaction of two competing considerations: the enforceability of the law, which makes the exigency issue important, and citizens' rights, which include the right to privacy [29].

This exception was first established by the United States Supreme Court in 1925, in the *Carroll v. US.*[13] decision which states:

[13]Carroll v United States, 267 U.S. 132 (1925)

> Various acts of Congress are cited to show that, practically since the beginning of the Government, the Fourth Amendment has been construed as recognizing a necessary difference between a search for contraband in a store, dwelling-house, or other structure for the search of which a warrant may readily be obtained, and a search of a ship, wagon, automobile, or other vehicle which may be quickly moved out of the locality or jurisdiction in which the warrant must be sought.

The Automobile Exception was developed further as more cases were decided (Table 6.4 shows ten landmark cases in the domain and their representation in terms of the factors identified in Table 6.2) with further conditions taken from cases subsequent to Carroll needing to be taken into consideration. For example, the type of the vehicle or movable container, the status of the vehicle, which influences whether there was an urgent need to search it (e.g. was the vehicle traveling on the highway (as in *Carroll*) or it was parked in a parking lot but capable of moving? *California v. Carney*[14]). Was it parked in a private place that is used for accommodation (*Coolidge v. New Hampshire*[15]) and so not subject to inspection without warrant or was it in a public location? Whatever the conditions, there must be a probable cause to search, but is it legal to search the whole vehicle if the probable cause applies only to a container inside the vehicle? What if an authorized warrant was easy to obtain? Such conditions and more are stated in the ADF table (Table 6.3). The aim of the court opinion in this domain is to determine whether there is enough exigency with respect to a possibly lowered expectation of privacy given the particular case facts. This is illustrated by the ADF factor hierarchy in Figures 6.2 and 6.3, based on the base-level factors given in Table 6.2. The definitions of the base and abstract factors are explained in Appendix E.

From the case decisions and representations shown in Table 6.4, acceptance conditions can now be generated for the fifteen non-leaf nodes, using the base-level factors in Table 6.2 the ADF is given in table form in Table 6.3. For each of the acceptance tests, a *default* value is defined when none of the conditions is satisfied. The default value is determined with respect to the nature of the factor: for example, the content is not considered private if none of the (AF131-PrivateContentsCarriage) acceptance conditions are satisfied. For some factors, it is explicitly stated that the factor is not clearly attributable on the basis of the facts as in UrgentReasonToSearch (AF122), and thus is considered unsatisfied.

```
AutomobileException IF Exigency
    AND (LOW)ExpectationOfPrivacy.

Privacy IF EnoughExpectationOfPrivacyInUse
    AND ( (NO InspectionRegulation)
    OR (NO VisibilityofItem) )
    OR (PrivateLocation AND (NOT AuthorizedWarrant) ).
```

[14] California v. Carney, 471 US 386 (1985).
[15] Coolidge v. New Hampshire, 403 U.S. 443 (1971).

```
Exigency IF (Mobile AND ExigencyWhenApproached
    AND ProbableCauseToSearch)
    OR  (NOT EaseToGetWarrant).

EnoughExpectationOfPrivacyInUse IF
    (Residence AND Accommodation)
    OR PrivateContentsCarriage.

Residence IF ConnectedMainLivingServices.

PrivateContentsCarriage IF GoodsCarried
    AND ProtectionLevel AND ContainerType.

Accommodation IF AccommodationSpaces OR RoomsFunctions.

SubjectToInspectionRegulation IF License
    AND (NOT RestrictedArea).

VisibilityofItem IF OnPublicView OR CanBeSeen
    OR (NOT CannotBeSeen).

ExigencyWhenApproached IF (UrgentStatus AND PublicLocation)
    OR (CapableToMove AND
    (PublicLocation OR PublicParking
    OR PermittedParkingTime)).

Mobile IF Automobile OR Vessel
    OR TowableVehicle OR LargeContainer
    OR MovableContainer.

EaseOfObtainingWarrant IF (NOT RiskofLosingEvidence)
    OR (Magistrate availability
    AND AuthorityOfMagistrate).

ProbableCausetoSearchVehicle IF OriginPurpose
    AND LegalUrgentReasonToSearch
    AND LegalSearchScope.

OriginPurpose  IF Information OR Observation
    OR Procedure.
```

```
UrgentReasonToSearch  IF PublicSafety OR Crime.

LegalSearchScope IF (WholeVehicle OR VehicleContainer)
    AND LegalSearchPlace.
```

ID	Base Factor
bf011	Automobile
bf012	Vessel
bf013	Towable
bf014	LargeContainer
bf015	MovableContainer
bf021	AuthorityOfAvailableMagistrate
bf022	RiskOfLosingEvidence
bf023	AvailabilityOfMagistrate
bf031	Licence
bf032	RestrictedArea
bf041	OnPublicView
bf042	CanBeSeen
bf043	CannotBeSeen
bf051	UrgentStatus
bf052	CapableToMove
bf053	Public Parking
bf054	PublicLocation
bf055	PermittedDuration
bf211	Information
bf212	Observation
bf213	Procedure
bf221	PublicSafety
bf222	Crime
bf231	WholeVehicle
bf232	OnlyVehicleContainer
bf233	SearchPlace
bf311	GoodsCarried
bf312	ProtectionType
bf321	ConnectedServices
bf331	AccommodationSpaces
bf332	RoomsFunction

TABLE 6.2: Base-Level Factors in The Automobile Exception as an ADF

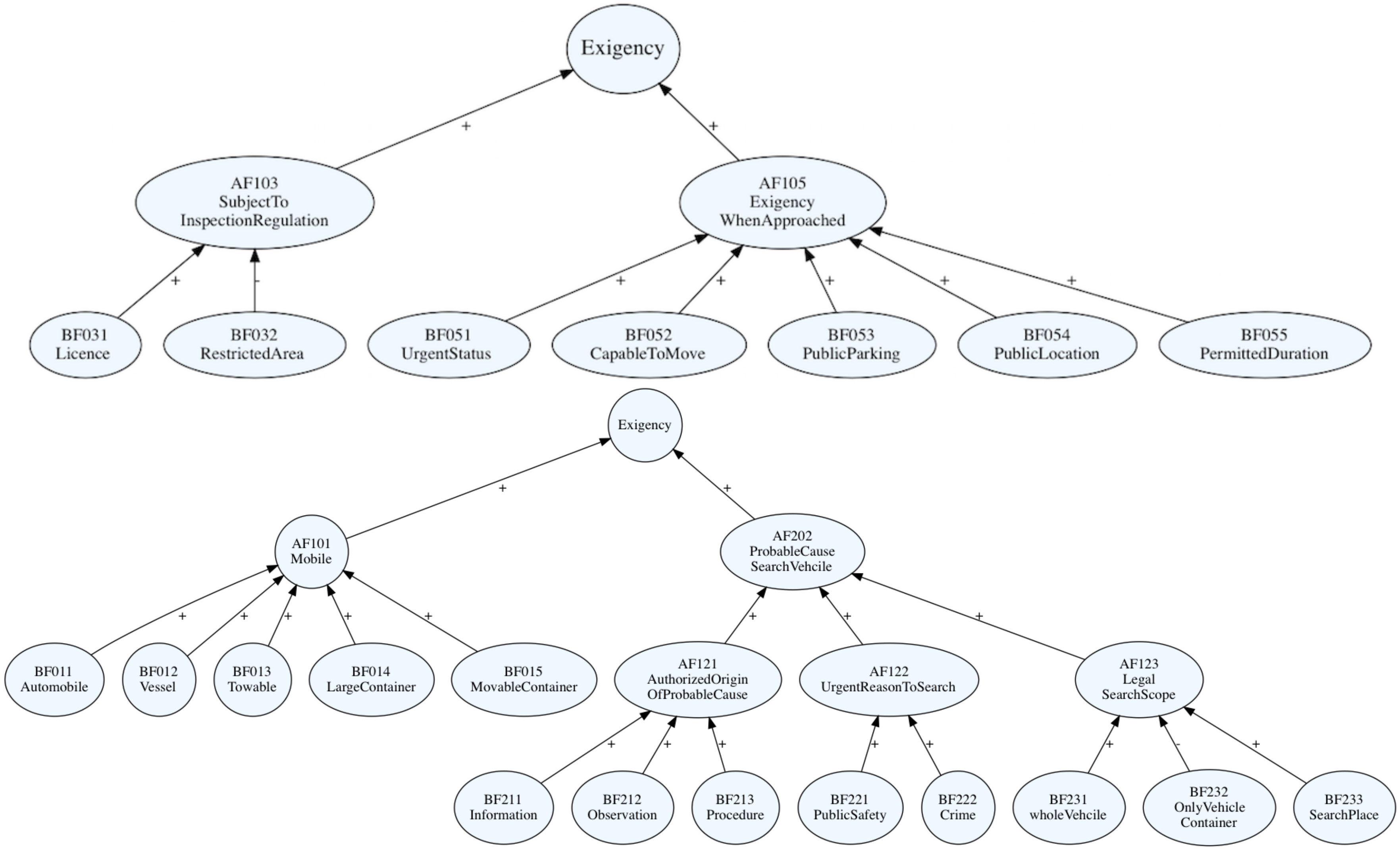


FIGURE 6.2: ADF for Automobile Exception, part 1

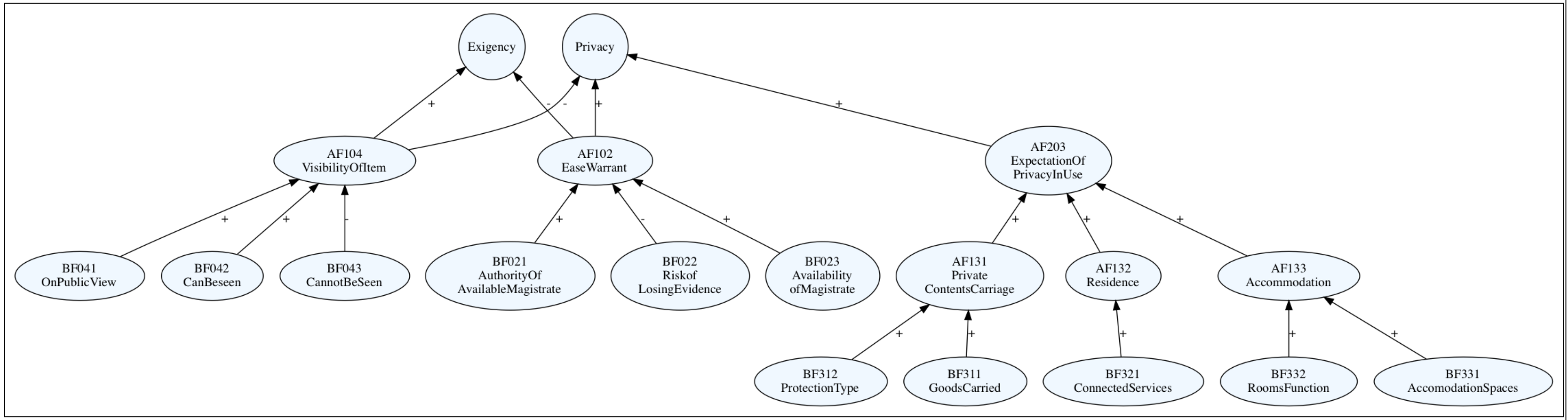


FIGURE 6.3: ADF for Automobile Exception, part 2

ID	S	L+	L-
I1	Exigency	AF202,AF101, AF103,AF104, AF105	AF102
I2	Privacy	AF203	
AF202	ProbableCauseToSearchVehicle	AF121 AF122 AF123	
AF203	ExpectationOfPrivacyInUse	AF131 AF132, AF133	
AF101	Mobile	bf011,bf012,bf013, bf014, bf015	
AF102	EaseWarrant	bf022	bf021, bf023
AF103	SubjectToInspectionRegulation	bf031	bf032
AF104	VisibilityOfItem	bf041,bf042	bf043
AF105	ExigencyWhenApproached	bf051,bf052, bf053,bf054,bf055	
AF121	AuthorizedOrigionOfProbableCause	bf211,bf212,bf213	
AF122	UrgentReasonToSearch	bf221,bf222	
AF123	LegalSearchScope	bf231	bf232,bf233
AF131	PrivateContentsCarriage		bf311,bf312
AF132	Residence	bf321	
AF133	Accommodation	bf331,bf332	

TABLE 6.3: Automobile Exception as an ADF

ANGELIC Automobile Program

The acceptance conditions are implemented using Prolog procedures as in the previous domains, and ordered to ensure that a report is provided giving the status of every non-leaf factor in the domain. The following procedure shows the code for the two acceptance conditions and the default clause for AF202-ProbableCauseSearchVehicle.

```
getProbableCauseSearchVehicle(C,Factors):-
member(af123,Factors),
member(af122,Factors),
member(af121,Factors), !,
write([there, is, a ,probable, cause, to , search,
vehicle ]),nl,getSubjectToInspection(C,[af202|Factors]).

getProbableCauseSearchVehicle(C,Factors):-
member(af122,Factors), member(af121,Factors), !,
write([there, is, a ,probable, cause, to , search,
vehicle , but, the , search, scope, was , illegal ]),
nl,getSubjectToInspection(C,Factors).
```

Case	Base-Level Factors	Favour
Carroll v. United States	bf011,bf051,bf211,bf231,bf233	P
Chambers v. Maroneys	bf011,bf015,bf051,bf054,bf213,bf222, bf231	P
Coolidge v. New Hampshire	bf011,bf021,bf024,bf051,bf213,bf222, bf231	D
Cady v. Dombrowski	bf011,bf042,bf043,bf051,bf053,bf053,bf213, bf222,bf231,bf233	P
South Dakota v. Opperman	bf011,bf015,bf043,bf051,bf053,bf213,bf221, bf231,bf233,bf233	P
United States v. Chadwick	bf011,bf014 ,bf043,bf051,bf053,bf211,bf221, bf232,bf233,bf312	D
Arkansas v. Sanders	bf011,bf014,bf015,bf032,bf043,bf051,bf211, bf221,bf232	D
United States v. Ross	bf011,bf015,bf032,bf051,bf053,bf211,bf221, bf231,bf232,bf233,bf233,bf311	P
California v. Carney	bf011,bf015,bf022,bf031,bf032,bf051,bf052, bf053,bf054,bf211,bf212,bf221,bf231,bf233,bf233, bf331,bf332	P
California v. Acevedo	bf011,bf015,bf032,bf043,bf051,bf054,bf211, bf221,bf232,bf233	P

TABLE 6.4: Automobile Exception Cases

```
getProbableCauseSearchVehicle(C,Factors):-
!,write([default,there, is, no, probable, cause, to ,
search, vehicle ]),nl,getSubjectToInspection(C,Factors).
```

The output below is for *California v Carney* (abbreviated in the code to“cvc”). Carney is concerned with whether the exception for automobiles to the protection against unreasonable search provided by the Fourth Amendment applies to mobile homes, in particular motor homes in which the living area is an integral part of the vehicle. The decision held that the exception does apply:

> When a vehicle is being used on the highways or is capable of such use and is found stationary in a place not regularly used for residential purposes, the two justifications for the vehicle exception come into play. First, the vehicle is readily mobile, and, second, there is a reduced expectation of privacy stemming from the pervasive regulation of vehicles capable of travelling on highways. Here, while respondent’s vehicle possessed some attributes of a home it clearly falls within the vehicle exception.

Thus the case decision indicates that although the mobile home is an automobile with an accommodation space, it was not parked in residential parking and not connected to any services, and so was currently being used as *a vehicle* not a home. There was a probable cause to search the mobile home to protect the public after police agents observed suspicious activity, and it was not possible to obtain a warrant since the vehicle was capable of moving.

```
case(cvc,[bf011,bf015,bf022,bf031,bf032,bf051,
          bf052,bf053,bf054,bf211,bf212,bf221,
          bf231,bf233,bf233 ,bf331,bf332]).
?- go(cvc).
it,is,mobile
there,was,exigency,when,approached
there,was,an,authorized,origin,of,probable,cause
the,main,reason,to,search,was,urgent
the,search,scope,is,legal
there,is,a,probable,cause,to,search,vehicle
subject,to,regular,inspection,but,
    the,search,was,directed,at,restricted,area
accepted,that,it,is,not,visible,to,public
accepted,that,contents,are,not,considered,private
accepted,that,it,is,not,connected,to,one,
    or,more,main,living,services
the,place,could,be,used,for,accommodation
accepted,that,low,expectation,of,privacy,in,use
it,is,not,easy,to,obtain,warrant
justified,under,automobile,exception
reduced,expectation,of,privacy
warantless,search,did,not,violate,the,fourth,amendment
find for the plaintiff
decision is correct in accordance with the actual decision
```

Another earlier case, US. v Chadwick [16], provides further details for the Automobile Exception in terms of the part of the vehicle that was searched. Chadwick found for the defendant so that the exception did not apply. Here, the agents searched a double-locked footlocker placed in the car trunk of a parked automobile without obtaining a warrant. The decision states that even if there was an urgent need to search the vehicle, the probable cause arises only from the footlocker, which should have been seized but not searched until a warrant had been obtained. The case output indicates that the warrantless search here violates the Fourth Amendment rule.

```
case(usvc,[bf011,bf014 ,bf043,bf051,bf053,bf211,bf221,
           bf232,bf233,bf312).
 ?-go(usvc).
it,is,mobile
there,was,exigency,when,approached]
```

[16]United States v Chadwick, 433 U. S. 1 (1977)

```
there,was,an,authorized,origin,of,probable,cause
the,main,reason,to,search,was,urgent
the,search,scope,is,illegal
there,is,a,probable,cause,to,search,vehicle,
    but,the,search,scope,was,illegal
subject,to,regular,inspection
accepted,that,it,is,not,visible,to,public
accepted,that,contents,are,not,considered,private
accepted,that,it,is,not,connected,to,
    one,or,more,main,living,services
accepted,that,the,place,is,not,used,for,accommodation
accepted,that,low,expectation,of,privacy,in,use
it,is,not,easy,to,obtain,warrant
reduced,expectation,of,exigency
intrusion,on,privacy,is,not,justified,
    under,automobile,exception
warantless,search,violates,the,fourth,amendment
find for the defendant
decision is correct in accordance with the actual decision
```

Results and Refinements

The program output shows that 9 of the 10 cases in this domain had been decided correctly. Only the most recent case in the set (*California v Acevedo*[17]) was decided wrongly. The Justices saw *Acevedo* as clarifying some doubtful findings in earlier decisions (in particular *Chadwick* and *Sanders*). In *Acevedo*, searching the vehicle at the police station, when the probable cause arose only from a container in the trunk, without obtaining a warrant, was held to be legal under the Automobile Exception to the Fourth Amendment rule. This is despite the apparent precedents of *Chadwick* and *Sanders*.

> Separate doctrines have permitted the warrantless search of an automobile to include a search of closed containers found inside the car when there is probable cause to search the vehicle, United States v. Ross,456 U. S. 798, but prohibited the warrantless search of a closed container located in a moving vehicle when there is probable cause to search only the container, Arkansas v. Sanders,442 U. S. 753. Pp. 500 U. S. 569-572.

This illustrates an example where citing over-ruled case decisions produces a wrong result. The previously accepted rule was explicitly rejected by the court in the *Acevedo* decision:

> The Chadwick-Sanders rule also is the antithesis of a clear and unequivocal guideline and, thus, has confused courts and police officers and impeded effective law enforcement

[17]California v. Acevedo, 500 U.S. 565 (1991)

The program cannot, of course, detect that an existing understanding of the precedents is about to be changed, and so the output of *Acevedo* follows the *Chadwick* decision and indicates, wrongly, that the search is not justified under the Automobile Exception.

```
case(cva,[bf011,bf015,bf032,bf043,bf051,bf054,bf211,
          bf221,bf232,bf233]).

 ?- go(cva).
it,is,mobile
there,was,exigency,when,approached
there,was,an,authorized,origin,of,probable,cause
the,main,reason,to,search,was,urgent
the,search,scope,is,illegal
there,is,a,probable,cause,to,search,vehicle,but,the,search,
    scope,was,illegal
subject,to,regular,inspection,but,the,search,was,directed,
    at,restricted,area
accepted,that,it,is,not,visible,to,public
accepted,that,contents,are,not,considered,private
accepted,that,it,is,not,connected,to,
    one,or,more,main,living,services
accepted,that,the,place,is,not,used,for,accommodation
accepted,that,low,expectation,of,privacy,in,use
it,is,not,easy,to,obtain,warrant
reduced,expectation,of,exigency
intrusion, on,privacy,is,not,justified,
    under,automobile,exception
warantless,search,violates,the,fourth,amendment
find for the defendant
decision is wrong
```

The program here produces a wrong decision, giving priority to the Chadwick-Sanders rule, whereas the actual decision is that the case should fall under the Automobile Exception rule if there is a probable cause to search a container inside a vehicle and the circumstances mean that no warrant is required.

> Police, in a search extending only to a container within an automobile, may search the container without a warrant where they have probable cause to believe that it holds contraband or evidence. *Carroll v. United States*.

A number of possible refinements can be applied to resolve such a problem. The program can be adjusted by adding new facts to represent the degree of acceptance of the base-level factor, or by using a portion of precedent cases as will be discussed in Chapter 7. Essentially this decision is intended to initiate a new period of settled laws, in which *Chadwick* and *Sanders* do not have force, whereas the program represents the pre-*Avecedo* situation. Of course a factor-based program such as the one presented here cannot get both *Chadwick* and *Avecedo* correct, since they follow different understandings of the law. To decide both cases correctly would require the inclusion of a temporal context. This has been quite widely discussed with respect to statutes, e.g.[102], but there is very little discussion with respect to cases, although the issue was raised in [41] and the effect of cases appearing in different sequences explored in [81].

6.2 Reasoning with Facts

The interpretation of cases cannot be disputed without descending to the level of facts as advocated by [24] in order to increase transparency in the ascription of base factors to cases. Each legal case should be represented as a set of facts that determine the base factors applicable to the case and thus allow the explanation of the attribution of factors, which in turn provides the basis for deciding the case using a set of abstract factors to resolve the conflict in the issues.

For example, consider *Carney*, from the domain of Automobile Exception. Here *Carney* has been redefined using the facts mentioned in the decision of the case rather than the base-level factors, and the program has been extended to provide acceptance conditions for the base factors using a number of possible facts. These facts can be seen as filling slots determined from the main Automobile Exception rule such as the type of vehicle (e.g. car, ship, wagon), personal effects (e.g. paper bag, suitcase), private place (e.g. store, dwelling-home). Additional facts are added to provide similarity and/or differences from precedent cases, or to show the effect of inserting new facts, such as the mobile home in *Carney*'s on the decision of the case.

The output below reports Carney's decision after incorporating the case facts:

```
case(cvc,[ft011mh,ft015pb,ft022nc,ft031mh,ft032ps,
        ft051p,ft052dr,ft053pl,ft054d,ft211pi,
        ft212po,ft221is,ft231all,ft233ps,ft233al,
        ft311is,ft312c,ft331c,ft331as,ft332bd,ft332k]).

 ?- go(cvc).
it,is,a,mobileHome,vehicle
accepted,that,no,largeContainers
paper,bag,is,a,movableContainer
it,is,mobile
there,was,no,urgent,status,automobile,was,parked
accepted,that,the,vehicle,is,capable,to,move
the,vehicle,was,in,public,location
the,vehicle,was,parked,in,public,parking
accepted,that,the,vehicle,was,parked,for,unknown,period
there,was,exigency,when,approached
received,information,from,public,informant
```

```
there,was,an,authorized,origin,of,probable,cause
main,reason,to,search,was,to,protect,the,public
the,main,reason,to,search,was,urgent
all,vehicle,parts,have,been,searched
the,vehicle,was,searched,twice,at,the,original,
    automobile,location,
    and,at,police,station
the,search,scope,is,legal
there,is,a,probable,cause,to,search,vehicle
has,a,special,motorhome,licence
police,station,is,a,restricted,area
subject,to,regular,inspection,but,the,search,was,directed,at,
    restricted,area
accepted,that,item,is,not,on,public,view,or,details,are,
not,provided
accepted,that,item,can,not,be,seen,by,public,
    or,details,are,not,provided
accepted,that,it,is,not,clear,that,items,can,not,be,seen
accepted,that,it,is,not,visible,to,public
illegal,goods
just,closed,but,not,protected
accepted,that,contents,are,not,considered,private
accepted,that,none,of,living,main,services,are,
    specified,or,connected
accepted,that,not,connected,to,
    one,or,more,main,living,services
consists,of,a,cab,and,suitable,accommodation,space
the,place,could,be,used,for,accommodation
accepted,that,low,expectation,of,privacy,in,use
accepted,that,there,is,risk,to,lose,evidence
accepted,that,magistrate,is,not,available
accepted,that,authorized,magistrate,is,not,available
it,is,not,easy,to,obtain,warrant
justified,under,automobile,exception
reduced,expectation,of,privacy
warantless,search,did,not,violate,the,fourth,amendment
find for the plaintiff
decision is correct in accordance with the actual decision
```

Appendix H illustrates the Automobile Exception domain program and results using reasoning with facts.

6.3 Including The Dissenting Opinion

The domain ADF so far is implemented to produce the case decision giving the justification of the case opinion. In addition to the majority opinion, the domain ADF can be modified to provide justifications for other opinions, such as dissenting or Justices' concurring opinions. In this case, new acceptance conditions that support other opinions can be defined straightforwardly using the same ADF statements and links. Using the Automobile Exception ADF in Table 6.3, further analysis of the dissenting opinions in the ten cases has been completed to define the acceptance conditions required to capture the dissenting opinions. However, not all the dissenting opinions are related to the Automobile Exception: some Justices argued that the case should be justified under different exceptions. In Chadwick, for example, Blackmun's dissent argues that the search was permissible as a search incident to arrest:

> The overbroad nature of the Government's principal argument, however, has served to distract the Court from the more important task of defining the proper scope of a search incident to an arrest. The Court fails to accept the opportunity this case presents to apply the rationale of recent decisions and develop a clear doctrine concerning the proper consequences of custodial arrest. Accordingly, I dissent from the judgment.

The program is thus modified to capture the dissenting opinion for five cases which did relate to the Automobile Exception: *Cady v. Dombrowski*, *South Dakota v. Opperman*, *United States v. Ross*, *California v. Carney* and *California v. Acevedo*. New acceptance conditions for the conflict issues, Privacy and Exigency, have been added to the domain ADF, in addition to new report messages for some components to provide justification for the dissent opinion in the program without modifying the existing components or acceptance conditions. The modified rules are shown below. The exigency in warrantless search is considered only if the vehicle is not suitable for accommodation. Otherwise, the priority should be given to the expectation of privacy.

```
AutomobileException IF Exigency
    AND (LOW)ExpectationOfPrivacy.

Privacy IF  Mobile AND EnoughExpectationOfPrivacyInUse
    OR ( (NO InspectionRegulation) OR (NO VisibilityofItem) )
    OR ( (NOT PublicLocation) AND (NO AuthorizedWarrant) )
    OR EaseToGetWarrant.

Exigency IF  (Mobile AND ExigencyWhenApproached
    AND ProbableCauseToSearch) AND (NOT AccommodationUse)
    OR (NOT EaseToGetWarrant).
```

Using the same case representation in Table 6.4, after producing the majority opinion, the program is directed to check the satisfied factors according to the new acceptance conditions to provide the dissenting opinion. To continue with Carney as an example, the dissent opinion states:

> In this case, police officers searched a Dodge/Midas Mini Motor Home. The California Supreme Court correctly characterized this vehicle as a "hybrid" which combines "the mobility attribute of an automobile . . . with most of the privacy characteristics of a house.... By choosing to follow the latter route, the Court errs in three respects: it has entered new territory prematurely, it has accorded priority to an exception, rather than to the general rule, and it has abandoned the limits on the exception imposed by prior cases.

Following Carney's majority justification shown in the previous section, the dissent opinion produced from the ANGELIC Automobile program is given as:

```
dissenting,argument
it,is,a,mobileHome,vehicle
accepted,no,largeContainers
paper,bag,is,a,movableContainer
it,is,a,hybrid,with,mobility,features
accepted,the,vehicle,is,capable,to,move
there,was,exigency,when,approached
there,is,risk,to,lose,evidence
it,is,not,easy,to,obtain,warrant
reduce,expectation,of,exigency
hybrid,with,mobility,and,home,features,shows
,enough,expectation,of,privacy
warantless,search,violates,the,fourth,amendment
true.
```

Another interesting case is *California v. Acevedo*. As explained above, the ANGELIC Automobile program in Acevedo produces the wrong decision by giving priority to the Chadwick-Sanders rule, which is what the Justices in the dissent opinion argued:

> Relying on arguments that conservative judges have repeatedly rejected in past cases, the Court today – despite its disclaimer to the contrary, ibid. – enlarges the scope of the Automobile Exception to this "cardinal principle," which undergirded our Fourth Amendment jurisprudence prior to the retirement of the author of the landmark opinion in United States v. Chadwick, 433 U. S. 1(1977).

```
dissenting,argument
it,is,a,vehicle
accepted,no,largeContainers
```

```
paper,bag,is,a,movableContainer
it,is,a,mobile
accepted,the,vehicle,is,capable,to,move
there,was,exigency,when,approached
there,is,risk,to,lose,evidence
it,is,not,easy,to,obtain,warrant
accepted,reduce,expectation,of,exigency
enlarge,scope,of,fourth,amendment,exception
any,container,inside,automobile,shows,enough,privacy,interest
warantless,search,violates,the,fourth,amendment
true .
```

The ANGELIC Automobile program showing the dissent rules and the output of the five cases are given in Appendix H. Furthermore, concurring opinions related to some Justices' justification of either the majority or the dissent opinions can also be captured and considered. Acceptance conditions for each concurring opinion can be defined and the program can be controlled to consider different opinions.

6.4 Correspondence Between Oral Hearings and Opinion Case ADFs

Chapter 5 demonstrated how a Case ADF can be instantiated from the merged Justices ACTs in the Oral Hearings. Now, to investigate the relationship between the Opinion and the Oral Hearings, a comparison is conducted between the Opinion Case ADF and the Case ADF instantiated from the Oral Hearing of California v. Carney to determine how the statements, links and acceptance conditions are related and modified. Table 6.5 shows the accepted components in the California v. Carney ADF based on the example in Section 6.2.

Giving this, Table 6.6 provides a number of findings that can be detected by examining the relationship between the statements and relations in both the Oral Hearing and the Opinion Case ADFs (CADFs) in Carney:

- The same Automobile Exception domain conflict issues, *Exigency* and *Privacy*, and *Mobile* abstract factors continue to appear in the opinion Case ADF.
- *PermittedDuration*, *ConnectedServices* and *AccommodationSpace* are three new hypothetical base factors in the domain ADF that had been introduced by the Justices earlier in the Oral Hearing of the Plaintiff Dialogue as *ParkingTime*, *HomeFacilities*, and *HomeConfiguration* respectively. These base factors are used to update the domain ADF but only *AccommodationSpace* is considered in the opinion Case ADF.
- A number of base factors in the Oral Hearings are related directly to existing factors from precedent cases in the domain ADF such as: *PublicLocation, PublicParking and ResidentialUse*. However, the parents of these base factors are abstract factors in the domain ADF.

S	L+	L-
Exigency	Mobile, ExigencyWhenApproached, SubjectToInspectionRegulation, ProbableCauseToSearchVehicle	
Privacy		SubjectTo Inspection Regulation
Mobile	Automobile, MovableContainer	
ExigencyWhenApproached	CapableToMove, PublicParking, PublicLocation.	
SubjectToInspectionRegulation	Licence	
ProbableCauseToSearchVehicle	AuthorizedOriginOfProbableCause, UrgentReasonToSearch, LegalSearchScope	
AuthorizedOriginOfProbableCause	Information	
UrgentReasonToSearch	PublicSafety	
LegalSearchScope	WholeVehicle,SearchPlace	
Accommodation	AccommodationSpaces	
EaseWarrant		RiskOfLosing Evidence

TABLE 6.5: California v. Carney Opinion Case ADF

- *MobileType* and *VehicleConfiguration* base factors are related to multiple base factors *Automobile, Vessel, Towable and MovableContainer* in the domain ADF.
- The parents of some base factors in the Oral Hearings have been changed in the domain ADF, e.g. The parent of *VehicleStatus* is now *ExigencyWhenApproached* instead of *Mobile* in the Oral Hearing Case ADF and *SubjectToInspectionRegulation* is the parent of *VehicleRegistration*. These changes modify the acceptance of the components as discussed below.
- The Oral Hearing Case ADF has enriched the domain ADF with a number of new facts for both existing and new base factors, e.g. *MobileHome (Automobile)*, *MobileHomePark (PublicParking)* and *DowntownLocation (PublicLocation)*.
- A number of factors previously existed in the domain ADF from precedent cases, such as the base factors of *ProbableCauseToSearchVehicle*; however, these were not mentioned in the Oral Hearing Case ADF.

The acceptance conditions have also been modified as a result of the component mapping, as shown here:

Level	O.H. CADF	Opinion CADF	Parent	Accepted
I	Exigency	Exigency	-	Y
I	Privacy	Privacy	-	Y
AF	Mobile	Mobile	Exigency	Y
BF	MobileType, VehicleConfiguration	Automobile, Towable, Vessel, Movabler Container LargeContainer	Mobile	Y
BF	VehicleStatus	UrgentStatus	ExigencyWhen Approached	N
BF	VehicleRegistration	Licence	SubjectTo InspectionRegulation	Y
BF	PublicLocation	PublicLocation	ExigencyWhen Approached	Y
BF	PublicParking	PublicParking	ExigencyWhen Approached	Y
BF	ParkingTime	PermittedDuration (NEW)	ExigencyWhen Approached	N
BF	HomeFacilities	ConnectedServices (NEW)	Residence	N
BF	HomeConfiguration	AccomodationSpaces (NEW)	Accomodation	Y
BF	HomeFeatures	RoomsFunction	Accomodation	N
BF	ResidentialUse	Residence	ExpectationOf PrivacyInUse	N

TABLE 6.6: Mapping Oral Hearing and Opinion Case ADFs

- The case is represented by all the facts from the Oral Hearing Case ADF, which in turn have modified the definition of the existing base factors such as *MobileHome* and *MobileHomeParking*.
- New acceptance conditions have been defined for the new base factors *PermittedDuration*, *Connected Services* and *AccommodationSpace*, which in turn have modified the acceptance conditions of the existing abstract factors *ExigencyWhenApproached*, *Residence*, *Accommodation* and *ExpectationOfPrivacyInUse*.
- The case is accepted as an Automobile Exception, giving priority to high exigency from the acceptance condition

```
Exigency IF Mobile AND ExigencyWhenApproached
    AND ProbableCauseToSearch
    AND SubjectToInspectionRegulation
```

This is equivalent to the acceptance condition of *Exigency* in the Oral Hearing Case ADF, where *ParkingType and Location* determine the acceptance of *ExigencyWhenApproached*. However, *ParkingDuration* is not taken into consideration in increasing the exigency. Also *Accommodation* only was not enough to determine *ExpectationOfPrivacyInUse* because the vehicle was not parked in a private parking place.

- On the other hand, the Oral Hearing Case ADF considered the dissenting opinion in the acceptance condition of the *Privacy* issue, i.e. regardless of the parking type, having features of accommodation space should raise the privacy expectation.

```
Privacy IF  Mobile AND EnoughExpectationOfPrivacyInUse

EnoughExpectationOfPrivacyInUse  IF  Accommodation
    OR Residence
    OR PrivateContentsCarriage
```

These findings indicate that the Oral Hearing stage plays a role in deciding the opinion of a case; the different components from the Oral Hearings feed the domain ADF, which in turn provides more justification for the majority and dissent opinions in the cases. Moreover, the sources of these components can be easily followed by tracking the Oral Hearing ACTs to determine the relationship between the opinion arguments.

6.5 Summary

This chapter has shown, for several legal domains, how ADFs can be used to encapsulate a theory of knowledge of the domains case law, in order to form the basis of the design of a program to decide cases in those domains. The theories are constructed by the domain analyst by expressing the cases in terms of factors and are implemented taking advantage of the closeness of the representation to an executable form by translating the acceptance conditions of the ADF into Prolog procedures. The results obtained for each domain have clearly shown the good performance of the ADF approach over three different domains, and demonstrated how the representation can be easily refined, giving a transparent output, to identify whether a factor was omitted, wrongly attributed, or acceptance conditions were wrongly identified. The discussion suggested ways in which factor-based systems, which are limited by taking as their starting point the representation of cases as sets of factors and so abstracting away the particular facts, can be extended to address open issues in AI and Law by incorporating the case facts to improve the decision, by considering justification and reasoning using precedents and allowing explanation of why factors were held to be present. The evaluation of the approach will be explored further in the next chapter.

Chapter 7

Evaluation

"Who judges the judge who judges wrong?" Gail Carson Levine, Fairest

---

This chapter provides a number of evaluations of the ADF approach as a design tool to predict the outcome of legal cases in several domains. Section 7.1 provides an evaluation to compare the performance of ANGELIC in the three legal domains discussed in previous chapters. Another comparative evaluation is conducted in section 7.2 to see how ANGELIC performs compared to other legal reasoning systems. In Section 7.3, the ADF approach is evaluated to compare the transparency in reasoning at different reasoning levels. Finally, the chapter concludes with a summary in Section 7.4 that evaluates the role of the ADF as a design tool for developing a knowledge base to represent case law.

7.1 Evaluation Over Legal Domains

This section evaluates the use of ADFs to design and implement a program to decide legal cases based on knowledge derived from a number of precedents in a particular case law domain. As shown by the examples in the previous chapter, firstly a number of decided cases are analysed. After that, the ADF is constructed from the factors and issues, showing the support and attack relationships between parents and their children and having the base-level factors as the leaves of the ADF. The acceptance conditions are defined for each node, and translated to Prolog procedures. The ANGELIC program is then run against a set of test cases and the decisions from the program output are compared to the actual outcomes. Table 7.1 summarises the specification of the three legal domains, showing the number of cases, the knowledge base size in the ADF, results obtained and the possible refinements.

As a result of evaluating the approach in these three domains, a number of findings are stated below:

- Applying the approach in three domains shows the effectiveness of the method for encapsulating the theory developed in the analysis.
- The success of the approach depends to a large extent on the quality of the analysis. It is important that the domain modelled be in a stable state (the second stage of Levi's

Domain	No.of Cases	No. of Factors	Results
US Trade Secrets	32	(26 BFs, 18 AFs)	Direct CATO: 78.1% (25 cases) Refinements 1: 90.6%(29 cases) Refinements 2: 96.8%(31 cases)
Wild Animals	5	(16 BFs, 10 AFs)	100%
Automobile Exception	10	(31 BFs, 15 AFs)	90% (9 cases)

TABLE 7.1: ADF for Three Legal Domains

life cycle as explained in [88]). However, the ADF design facilitates refinement of the ANGELIC program in the light of its performance on test cases, to remedy any defects in the initial analysis.

- The method can be applied to various domains where factor-based representation is considered an appropriate approach to capture the knowledge of the domain. Normally, the results are affected by the purpose of the analysis; US Trade Secrets, for example, required more refinements as that had been translated from the analysis of CATO which indeed was not to predict outcomes, while the cases in the Automobile Exception domain were analysed mainly to construct a domain ADF to provide justification for the case decisions.

- Results are also affected by the size of the available knowledge. The small number of cases and ADF representations found in the Wild Animals domain partly explains the correct results obtained, although here the intention was only to encapsulate the theory governing those cases.

- When using a set of factors identified for the case, programs based on the ADF design provide very transparent output that identifies precisely where the outcomes suggested by the implementation diverge from the actual case decision. After running the program, and noting any divergences from the actual outcomes, reading the original decision texts may suggest one or more of four solutions to refine the behaviour of the program. These are, in ascending order of divergence from the original analysis:

  – Removing a factor wrongly attributed to the case
  – Adding a factor wrongly omitted from the case
  – Modifying an acceptance condition: e.g. changing the priorities
  – Modifying the ADF: e.g. adding a supporting or attacking node for the problem node.

In some cases, such as those encountered in the US Trade Secrets domain, the problem seems to lie with the attribution of the factors. Should *Goldberg* really have F27? Should *Mineral Deposits* have F14? Should *Space Aero* include F14 or F21 and exclude F19?

Such matters were contested in the actual case, and ascribing the presence or absence of particular factors requires interpretation of the case by the analyst.

As well as disputed factors, a decision like *Goldberg* suggests modifying the description of factors intended to guide the analyst. In that decision it was suggested that to count for the defendant, the information not only had to be publicly available but also the public source needed to be known and *used by the defendant*, which would narrow the applicability of F27 from that described in [10].

- In general, adding or removing a factor to or from a particular case provides a local solution, which will often solve a problem with a particular case. The results, however, indicated a general problem that was applicable to several cases: in the Trade Secrets domain, F16, reverse engineerable, had a dominant effect, which led to an incorrect decision in several cases. It seemed clear that the presence of F16 should not by itself be sufficient for a finding for the defendant. Again, the decisions themselves suggested several possible ways of arguing against F16: in particular the use of restricted materials and the uniqueness of the product. Either or both of these exceptions could be incorporated into the ADF without adversely affecting the other cases.

- Moreover, it should be remembered that the decisions themselves may be erroneous. Assuming that there are least some poor decisions, which should not serve as precedents, a certain number of divergences from the results should be tolerated. Moreover, a landmark case like *Avecedo* may significantly change the interpretation of some previous decisions, revising the applicable case law and necessitating revision of the program.

To summarise:

- Analysing a legal domain by simply translating the analysis of [10] into an ADF and executing the resulting program gave results almost identical to those found for CATO in the IBP experiments reported in [55] and shown in Table 7.2. Note that this is achieved without the need for balancing pro and con factors, which is central to existing case-based reasoning systems.

- The reasons for the “incorrect” decisions can be readily identified from the output and the ADF, as seen in the discussion of the wrongly decided cases in US Trade Secrets and Automobile Exception domains.

- Examination of the texts of the decisions readily explained why the results diverged, and suggested ways in which the analysis could be improved, either at the case level by changing the factors attributed, or at the domain level by including additional supporting or attacking links.

All this indicates that the use of ADFs leads to a system which provides good performance, and has a number of positive features from a software engineering (and domain analysis) standpoint, which enable instantiations of ADFs to be refined where needed and their performance improved. For a practical system there are often several ways to fix a problem, and thus a reasonably large set of test cases is needed in order to choose between the different solutions and to guard against over-fitting.

7.2 Comparative Evaluation Over Legal Reasoning Systems

The evaluations so far have been conducted to ensure the applicability of the approach in different legal domains, but how does the performance of the ADF in legal domains compare to other legal reasoning systems?

To investigate this, a quantitative evaluation was used to compare the empirical results obtained in different legal reasoning systems. Although the goal in developing each legal system was different, they all use case-based reasoning in legal domains. In [55] a number of systems were evaluated, including several machine learning systems: one using Naive Bayes; programs representing HYPO, which is considered the first system that used case-based reasoning in legal domains; CATO, which was designed to identify arguments rather than to predict outcomes, though prediction is the goal of the logical model of IBP without the case-based reasoning component; and IBP itself. Results taken from [55] are shown in Table 7.2[1]. Some other systems are included in the table, including AGATHA [57]. A number of different heuristics were investigated in AGATHA [57], and so two lines are included, both a high figure (for brute force search with the largest number of seed cases, labelled AGATHA 8) and a low figure (for the version which used A* for heuristic searches).

Another two lines are added for ADF representation in the US Trade Secrets domain, one representing an ADF encoding of the CATO hierarchies (ANGELIC CATO) and another an ADF refined (ANGELIC Trade Secrets) in the light of re-examination of matters wrongly decided by the original version. The result shows that the ADF constructed directly from CATO performed as well as CATO (78.1%), which, as would be expected, takes into consideration the similarity in the factor hierarchy structure and the case representation in factors. However, applying a few refinements to seven wrong cases considerably improved the performance, making it the best of all the legal reasoning systems. Although these findings are promising, the approach still needs to be applied further on a large number of test cases.

In order to show the effect of abstention, an additional column is included showing the percentage of correct answers for which an opinion was given (akin to precision in an Information Retrieval system). Finally, two hypothetical programs are included: CATO-coin and HYPO-coin, in which the abstentions are decided arbitrarily and it is assumed that half these decisions will be right and half wrong.

What Table 7.2 shows is that simply guessing the outcome can improve the percentage of correct answers of most systems, although it greatly reduces of the precision of, for example, HYPO, which abstains on a very large number of cases, but is correct in an excellent percentage of those for which it does yield an answer. Anything that can be done to reduce abstentions, even

[1]The numbers appear to contain some discrepancies, but these are present in the source, [55]

	correct	error	abstain	accuracy	no-abst
ANGELIC Trade Secrets	31	1	0	96.8	96.8
AGATHA 8	30	2	0	93.7	93.7
IBP	170	15	1	91.4	91.9
AGATHA A*	29	4	0	90.6	90.6
CATO-coin	163	30	0	89.0	89.0
Naive Bayes	161	25	0	86.5	86.5
HYPO-coin	152	34	0	81.7	81.7
ANGELIC CATO	25	7	0	78.1	78.1
CATO	152	19	22	77.8	88.8
HYPO	127	9	50	68.3	93.4
IBP-model	99	15	38	72.6	86.8

TABLE 7.2: ANGELIC and Legal CBR Systems Results From [55] and [57]

if the success rate is only 50%[2], is thus going to be beneficial in terms of successfully predicted outcomes.

7.2.1 Relation to Structured Argumentation

ASPIC+ is a structured framework for specifying systems [105]. It defines the notion of an abstract argumentation system as a structure consisting of a logical language, and two subsets of strict and defeasible inference rules. In addition to a knowledge base that consists of two disjoint subsets, the axioms and the ordinary premises. Three types of attacks are defined in ASPIC+: rebuttal (a conclusion attack); undermining (a premise attack); and undercutting (an inference attack).

Argumentation systems in ASPIC+ are applied to knowledge bases to generate arguments. Arguments in ASPIC+ are defined as inference trees formed by applying strict or defeasible inference rules to premises formulated in some logical language. Two main ideas can be seen about ASPIC+ [101]:

- conflicts between arguments are sometimes resolved with explicit preferences, having that arguments are built with two kinds of inference rules: strict, which logically entail their conclusion, and defeasible rules, which only create a presumption in favour of their conclusion.
- conflicts between arguments may arise from the inconsistency of a knowledge base or the defeasibility of the reasoning steps in an argument.

ASPIC+ has been used as a tool for reconstructing natural legal argument about legislative proposals in [106] where defeasible inference rules are modelled using argumentation schemes, and argument ordering provides a suitable way to evaluate debates. Examining the overall output

[2]Note, however, that the unrefined ADF seems to get the cases on which CATO abstained uniformly wrong.

of the ANGELIC program, it bears a strong resemblance to the kind of structured argumentation found in ASPIC+ [105].

From a formal perspective, acceptance conditions are supplied for each node in ANGELIC. These take the form of tests: sufficient conditions for accepting or rejecting the parent node, in terms of the status of its children. These tests are arranged in a priority order and a default is given to cover cases where none of the tests are satisfied. The union of the acceptance conditions in ANGELIC can be seen as comparing to the underlying ASPIC+ knowledge base. Determining the acceptance or rejection of the various nodes produces sub-arguments, and these can be linked to produce the argument for finding for the plaintiff (or defendant) which follows the argument and (or-) sub-argument structure of ASPIC+ arguments.

From an implementation perspective, there are also differences: because the ordering of clauses expresses priority between arguments, only the winning arguments are generated. Thus the output does not include all arguments, but only the winning line of argument. Where, however, potential attackers are children rather than siblings, but are not acceptable, this is reported. Thus, although there are correspondences, especially through the argument - sub-argument structure, the control regime employed by the program means that there are also important differences.

These relate mainly to conflicts: the output shows only the winning side of the case, and does not provide a good record of the rejected arguments available to the losing side. Although Section 6.3 in the previous chapter extended the ANGELIC automobile program to generate the dissent argument, that was by redefining the nodes with additional acceptance conditions related to the dissent opinion, rather than examining different opinions in the same node.

Moreover, there are also resemblances between the argument graph constructed from Carneades and ANGELIC. Both graphs provide explanations and justifications. However, the argument graphs in Carneades have two kinds of nodes; statement nodes and argument nodes. Arguments are designed to model instantiations of argumentation schemes linking a set of premises to a conclusion, multiple forms of reasoning can be integrated in Carneades architecture, which allows a number of computational models of argumentation to be used together to search and construct argument. Whereas in ANGELIC all the nodes are statements of argument components, the statement acceptance or rejection based strongly on the acceptance or rejection of its children through a set of defined acceptance conditions. On the other side, the evaluation of arguments in Carneades depends on whether statements have been questioned or decided, the allocation of the burden of proof, and the proof standard applicable to questioned statements. Carneades has been reconstructed as an ADF in [52]. which allows the generalisation of argument evaluation structures on ADF nodes, and helps put Carneades on a solid formal foundation.

Although there are similarities between ANGELIC and structured argumentation, the strength of ANGELIC relies on the ability of applying ADF as a knowledge engineering tool to provide a simple methodology for instantiating ADF to construct a domain theory by partitioning the problem into set of acceptance conditions, which limits the number of precedents required to determine the outcome of cases. In addition to the ease of implementation by translating the acceptance conditions to Prolog clauses. Although examples of implementations are provided in the thesis, the main role of ANGELIC is to provide the knowledge in which systems can be realised. ANGELIC program performed very well comparing to previous systems, giving that the diverges in the program output can be easily be refined taking in advantage the transparency of the output produced. There are also straightforward paths from ANGELIC to both ASPIC+ and Carneades.

7.3 Evaluation Over Reasoning Levels

As the ANGELIC program proceeds, it reports on the acceptability or otherwise of the various abstract factors and the resolution of issues. As shown in the outcome from the previous chapter, this provides a good diagnostic for divergent decisions, but how does it measure up to the actual opinions found in cases?

In this section a qualitative evaluation is conducted to compare the quality of explanation between the program outcome derived from ADF with base factors as leaf nodes and the outcome after considering the integration of facts as leaf nodes in addition to the citation of precedents, which are such an important feature of real decisions. Moreover, further evaluation of reasoning types using portions of precedents and values are discussed to improve the explanation produced using the ADF approach in legal reasoning.

7.3.1 Quality of Explanations Using Factors and Issues

As stated earlier in Chapter 3, the court opinion provides the case verdict and an explanation that justifies the case verdict. Sometimes, in some landmark cases, this justification provides a test that gives a means of deciding in future cases: a typical case opinion document in the US Supreme Court begins with a *syllabus* that summarises the case, followed by the *majority opinion* that has been written by one of the Justices, which supports the justification for the majority opinion. The main opinion involves the case decision and explanation for the decision, giving the issues that the Justices relied upon in their justification. However, often, the opinion document involves multiple opinions. Justices might agree with the decision but have different justifications (*concurring opinions*); likewise, Justices might disagree with the decision and thus have dissenting opinions. At the end of the main opinion, the document clarifies the court action, which might be affirming or reversing the lower court decision, or returning the case to the lower court (remanding).

Now consider a re-ordering of the elements found in the ANGELIC program outcome from the previous chapter for a case, say *Boeing* from the US Trade Secrets domain (Section 6.1.1), by omitting some elements such as some default statements reported from some factors, and adding a little linking text to the explanation. Recall too that the ANGELIC program is written and used thus far to “decide” the cases: in a version to supply explanations, the text reports could be customised to indicate the particular clause being used for a node by giving the base level factors used. Below is what such an explanation might look like: the current program output is shown in boldface, possible clause-specific customisations in italics and linking text in ordinary font.

> We **find for plaintiff**. The **information was a trade secret**: **efforts were taken to maintain secrecy**, *since disclosures to outsiders were restricted and the defendant entered into a non-disclosure agreement and other security measures were applied.* **The information was unique**. It is **accepted that the information was valuable** and it is **accepted that the information was neither known nor available**. **A trade secret was misappropriated**: **there was a confidential relationship** *since the defendant entered into a non-disclosure agreement* and it is **accepted that the information was used**. Moreover **improper means were used** *since the defendant used restricted materials*.

This explanation matches the following questions found in Boeing claims in the actual *Boeing v. Sierracin* opinion:

> **QUESTION 1:** Did Sierracin breach a confidential relationship with Boeing?
>
> **QUESTION 2:** Did Sierracin breach a contract with Boeing?
>
> **QUESTION 3:** Did Sierracin misappropriate Boeing trade secrets?

This seems to have the makings of a reasonable summary compared to the *Boeing* decision. There are three problems: it does not indicate what the defendant contended, since the clauses of the ANGELIC program which were not reached do not feature in the report, no precedents are cited, and the facts on which the finding is based are not present. Nonetheless, the output found is a distinct improvement on previous work such as [57]. Also, the output from the current program could be readily used to drive a program of the sort envisaged by Branting [49], where an issue-oriented approach to judicial document assembly has been used to provide an explicit representation of legal rules, to construct a justification from these rules and create a judicial document. This will be even further improved if a fact layer, to allow the explanation of the attribution of factors, is added.

7.3.2 Quality of Explanations Using Facts

Of course, without facts, the decision will not be followed very closely: all the decisions begin with a summary of facts. Thus, the next technical step is to supply the fact layer. Further analysis of the original decisions and oral transcripts is required to extract facts, so as to permit argument about the ascription of factors, and to enable grounding of the explanations in the particular facts of a case. One possible way is to revive the idea of dimensions to provide a way of tackling this issue, as indicated by [24], [13] and [32]. Descent to the fact layer will result in a considerable improvement in the transparency of the output by enabling consideration of important features from the actual case decisions. In comparison to the output of the case presented above, there is a clear improvement in the quality of the explanation which provides the justification for the acceptance of the base-level factors. The order of the output states the status of the base-level factors first (children) and then indicates whether the abstract factor (parent of base-level factors) is satisfied or not. Following the approach discussed previously, the explanation of the Automobile Exception case, *Carney*, is produced from the program report:

> The **warrantless search did not violate the Fourth Amendment** rule. The vehicle was **capable to move** but found **parked** in a **public location**.The two **justifications for the vehicle exception** come into play. First, the vehicle is **readily mobile** and there is a **reduced expectation of privacy**.
> It is **subject to inspection under vehicle regulation**. Also, there was a **probable cause to search the vehicle**.

Now by considering the facts from the example in Section 6.2, the following output provides more transparency in the reasoning as shown here, where the facts have been highlighted:

> The **warrantless search did not violate the Fourth Amendment** rule. The respondent's **mobile home** was **capable to move** but found **parked** in a **public location**. The two **justifications for the vehicle exception** come into play. First, the vehicle is **readily mobile** and there is a **reduced expectation of privacy**.
> While the **mobile home** here **consists of cab and accommodation space**, it is **subject to inspection under vehicle regulation**. Also, there was a **probable cause to search the vehicle** based on the fact that the *respondent* was using the **mobile home** to sell **illegal contraband**

This decision now is very close to the actual decision, which states:

> When a vehicle is being used on the highways or is capable of such use and is found stationary in a place not regularly used for residential purposes, the two justifications for the vehicle exception come into play. First, the vehicle is readily mobile, and, second, there is a reduced expectation of privacy stemming from the pervasive regulation of vehicles capable of travelling on highways. Here, while the respondent's vehicle possessed some attributes of a home it clearly falls within the vehicle exception.

It clearly justifies what base-level factors have been accepted (or rejected). Moreover, the case facts can provide further advantage in specifying to what level the base factor is satisfied; this has been included as part of the future work discussed in Chapter 8.

7.3.3 Reasoning with Portion of Precedents

Using the ADF approach, there are no confrontations between large sets of pro and con factors covering the whole case. Instead, factors are opposed to one another in the context of accepting or rejecting particular nodes, and so represent a specific point in the debate. Thus two cases may be identical with respect to a subset of factors, which may be used to establish a particular abstract factor, even though the eventual outcome may differ.

Two points are significant here: first that some apparent distinctions are insignificant, since they relate to different issues; this was partly what the factor hierarchy was introduced in CATO to address. More importantly, however, a precedent might be citable to establish the existence of a certain plaintiff factor, even though the case as a whole was found for the defendant, because of some other issue. For this reason it is sometimes desirable to be able to reason with portions of precedents, as urged by Branting in [48]:

> This paper argues that the task of matching in case-based reasoning can often be improved by comparing new cases to portions of precedents. An example is presented that illustrates how combining portions of multiple precedents can permit new cases to be resolved that would be indeterminate if new cases could only be compared to entire precedents.

This is borne out by a reading of the decisions in a range of cases: rarely do they begin with a precedent and then discuss similarities and differences; rather, they use precedents at particular points of the decisions to identify questions and issues to be addressed, and to justify answers and consequences. This is effectively what is done in the ADF approach: competing factors are considered in the context of accepting or rejecting a particular node. Part of the output of the program is, for each case, the set of nodes satisfied. This information could be used to find the precedents needed to make particular points.

For example, in US Trade Secrets, all cases where the defendant had agreed not to disclose (F4) and yet efforts to maintain secrecy (F102) were not established could be retrieved. This query would return *CMI* where the information was known to competitor (F20), and so can cite *CMI* as a precedent when arguing (in the context of a case containing F4 and F20) that the efforts taken to maintain secrecy were insufficient to establish the information as a trade secret. In addition, the decision in *California v. Acevedo*, from the Automobile Exception domain, cites some precedents that support the defendant, but does not follow these citations due to the perceived lack of clarity of the decision in the precedents. Thus, it is important to clarify these citations in the output of the program in order to provide more transparency and closeness to the actual decision.

Matching at this level of granularity will help in specifying the precedents most relevant to the specific point needing to be argued. Moreover, such precedents can then be incorporated in the explanations to justify the acceptance or rejection of particular nodes, which is close to the way they are used in practice, and corresponds to the downplaying and emphasis of distinctions, which are such an important feature of CATO. To do this it would be helpful to be able to argue about preferences (perhaps using some form of meta-level argumentation as in [35]).

Another reason to consider portions of precedents is provided by [83]. In that paper *a fortiori* reasoning was explained in terms of identifying a rule using only a subset of factors available for the winning side being preferable to the rule using all the factors available to the losing side. That paper gave, however, no indication of how this subset should be chosen. The output from the ADF, in contrast, does show which factors were instrumental and active in winning the case.

7.3.4 Quality of Explanations Using Precedents Citation

In addition to the facts determining the presence of factors, further improvement to the transparency of the reasoning by justifying the preferences between portions of precedents is required, which better corresponds to legal practice as manifest in real decisions and may express social values and judicial preferences. Ideally each test in the acceptance condition should be related to a portion of a precedent case; this will require annotating the acceptance conditions with citations to precedent cases, which will in turn improve the program output. This is shown in the output of some factors as below:

[it,is,a,mobileHome,vehicle,**cite,carney,v,california**]

[main,reason,to,search,was,to,protect,the,public,**cite,carrol,v,us**]

[all,vehicle,parts,have,been,searched,**cite,carrol,v,us,and,us,v,ross**]

[only,vehicle,containers,have,been,searched,**cite,us,v,chadwick,and,arkansas,v,sanders**]

[justified,under,automobile,exception,**cite,carroll,v,us**]

Citation of some precedents can be found in the program output of the Automobile Exception domain in Appendix H. Now returning to the example of Carney's opinion, adding the citation of the main precedents improves the transparency and the reality of the case decision:

> The **warantless search did not violate the Fourth Amendment** rule. The respondent's **mobile home** was **capable to move** but found **parked** in a **public location**.The two **justifications for the vehicle exception** come into play. First, the vehicle is **readily mobile** Carroll v. US and there is a **reduced expectation of privacy** South Dakota v. Opperman.
> While the **mobile home** here **consists of cab and accommodation space** US v. Chadwick, it is **subject to inspection under vehicle regulation**. Also, there was a **probable cause to search the vehicle** Carroll v. US based on the fact that the *respondent* was using the **mobile home** to sell **illegal contraband**

Moreover, citing precedents could also provide a potential database to support conceptual retrieval of cases, which has been an important issue in AI and Law since its very beginnings (see, e.g. [78]).

7.3.5 Reasoning with Values

Note that the ADF structure, like CATO and IBP, does not include any reference to values. Since [40] it has become usual to resolve conflicts not definitively decided in the precedent base by appealing to the purposes of the law, commonly represented as values in the sense of [28], and as in [38] and other related work. In [38] values appear at the top of the hierarchy, occupying the place taken up by issues in the ADF representation.

The idea in [38] is that factors are associated with values, and that the precedents, by showing preferences between sets of factors, will reveal preferences over values. Since there are more factors than values, and several factors relate to the same value, these value preferences can be used to determine preferences between sets of factors which do not themselves appear in the precedents, allowing the system to draw conclusions that go beyond *a fortiori* reasoning. The use of values has been criticised and does not appear in any of HYPO, CATO or IBP. In AGATHA

[58], which was intended as an empirical evaluation of the approach of [38], the issues of IBP were used in the role of values. The idea in the ADF is that values should not form part of the structure: their role is to justify components of the structure and choices made. They should thus form part of the documentation of the design rather than part of the design itself. A value might therefore, for example, appear as comments on the acceptance conditions explaining why the tests are ordered as they are. As observed in [146], values do not play a single role: they can, for example, justify either the inclusion of a rule, or of an antecedent in the rule. In ADF terms, this means that they can justify the inclusion of a test within an acceptance condition, or a node representing a factor to be used in such tests. Moreover, playing the role assigned to them in [40] and [38], they can explain why the various tests in acceptance conditions are ordered in a particular way. Thus the knowledge engineer will need to identify a set of applicable values, and an ordering of them, to guide the design choices, but they will not form an explicit part of the ADF, just as their appearance in actual decision texts is very limited.

Since the value preferences are now effectively local to particular nodes, it is possible to express different preferences in different contexts, which may add a desirable flexibility. If it is considered undesirable, the knowledge engineer must ensure, as part of the verification activity, that the preferences are in fact consistent.

A means of extracting values from an ADF, from associations between values and factors, was given in [6]. In the current methodology, however, values inform the design but do not form part of it.

7.4 Summary

Evaluating the performance of ADFs in the different legal domains has addressed a number of advantages of applying a knowledge engineering approach that uses ADFs to design and develop a knowledge-base to represent case law, summarised as follows:

- The ADF instantiated from all the domains provides a visual representation (frame-based) and defines one or more acceptance conditions for every non-leaf factor; this ensures the completeness of the rule base and shows how the ADF mediates between frame- and rule-based representations of the domain knowledge.

- The encapsulated knowledge in all the domain ADFs was able to determine the outcome of the case from only portions of precedents (effective partitioning of the problem space), while at the same time supporting the verification by considering all the nodes in the ANGELIC program.

- Once the method had been extended to include the facts of particular cases at the lowest level of the ADF, new nodes were added straightforwardly, along with their acceptance conditions, with no problems or ramifications in the knowledge base. New nodes can be added independently from the rest of the nodes, defining the conditions from the children of the node.

- The evaluation of the ADF refinement process has shown that some factor rules can be removed or modified without affecting the rest of knowledge base.

- Further, comparing the performance of ADFs with previous legal reasoning systems shows that the refined ADF (ANGELIC Trade Secrets) performed the best, taking into consideration the differences between the purposes of these systems. The ready visualisation of the possible paths through the ANGELIC program allows detection of where the decision diverges from the actual decision and enables the application of possible refinements to improve the outcome.

- The modularisation achieved by using the ADF approach for design not only helps to drive the analysis, but also provides a number of software engineering benefits, such as ease of following the flow of control through the knowledge bases, limitation of the effects of modifications, and assistance with quality assurance, verification and validation of the ANGELIC program built upon the design.

The method has also provided advantages by demonstrating the improvement of explanations by ordering the output and considering facts and precedent citations. It might in future prove useful to provide links to values to improve explanations, but this will affect only presentation, not the decision making, since the effect of values and preferences is already present in the representation, in the form of the components included, the tests in the acceptance conditions and the order of these tests.

These findings are all encouraging and offer further pointers as to how to address important issues in future work, as presented in the next chapter.

Chapter 8

Conclusion and Future Work

> *"A story has no beginning or end: arbitrarily one chooses that moment of experience from which to look back or from which to look ahead."* Graham Greene, The End of the Affair

---

This chapter provides a summary for the overall research story, presenting the contributions of this research in the field of argumentation in AI and Law and discussing some potential areas for future work. The summary is presented in Section 8.1, the contributions from the research are given in Section 8.2, while potential research future extensions are discussed in Section 8.3.

8.1 Thesis Story

The aim of this thesis, as stated in the introductory chapter, was to attempt to fulfill the following:

> *To provide a method to encapsulate analyses of a variety of case law domains, potentially performed by different people, for different purposes, using different techniques, in a common format that can be readily realised in a computational form.*

Throughout the previous chapters, a number of issues have been addressed that all contribute toward fulfilling the aim above and answering the research questions and objectives set out in **Chapter 1**. This section summarises the story of the research, showing how the following research objectives (reproduced from Chapter 1) have been addressed.

Objective 1 Provide an approach to analyse and represent a legal procedure dialogue in a real context to provide an argument structure from the argument components.

Objective 2 Employ a knowledge engineering approach to capture knowledge in a legal domain, and to reason with this representation, to provide justification and transparency of legal decisions.

Objective 3 Develop a justification model that can be applied over different legal case domains using abstract argumentation, and compare the outcome to the actual legal decisions.

Objective 4 Conduct a comparative evaluation to compare the performance of the defined approach between the different domains, and other legal reasoning systems.

Objective 5 Discover various methods to improve the explanation of legal decisions in the program output.

As indicated by these research aims, this is interdisciplinary research integrating artificial intelligence, argumentation, and legal reasoning. After exploring the previous work related to the topic of this research in **Chapter 2**, this thesis presented a background to the US Supreme Court and its particular legal decision process, and examined selected legal domains in **Chapter 3**. The method presented in this thesis was applied to three legal domains from the US Courts, often used in the literature of AI and Law, which have been used as examples throughout this research. All the selected cases from these domains are taken from a period where the theory was relatively stable, although the Automobile Exception domain also includes the *Carroll* case, which ended the first stage and *Acevedo* case which extended the third stage.

1. The Automobile Exception domain (10 cases), which has been discussed previously in the literature [115], [76] and [29], and has been analysed in terms of argument components by the author in this research;

2. The Wild Animals domain (5 cases), where the factor-based analysis of [30] has been adapted;

3. And the US Trade Secrets domain using the factor hierarchy of CATO [10], IBP logical model [17, 18], and tested on 32 cases from AGATHA [57, 58].

Thus one domain is newly analysed (Automobile Exception), one adapts an existing representation (US Trade Secret), and the third rewrites a detailed existing analysis (Wild Animals) taking account of other analyses of this domain.

Starting with the Automobile Exception domain, **Chapter 4** showed how the legal dialogues in Oral Hearings can be analysed and modelled in a legal dialogue representation model, giving the specification of the Oral Hearing and the legal opinion in **Chapter 3**. The transcripts of the Oral Hearing of two landmark case studies in the domain, *US v. Chadwick* and *Carney v. California*, were analysed so that each utterance was marked with the appropriate speech act. A limited set of speech acts make the process more objective: four moves control the beginning and end of each dialogue and the Oral Hearing of the legal case, while the rest of the moves are assertions that result in adding new components, combining components or emphasising certain components. These are identified either from existing components recorded in a legal ontology from precedent cases or introduced as a new component from the current case. The sequence of speech acts and their contents can be input to a program, to effectively reproduce the dialogues using the restricted vocabulary. The program processes the sequence of speech acts to build the various component trees (ACTs) corresponding to petitioner's, respondent's and Justices' perspectives. This legal analysis workflow addresses **Objective 1** in this thesis.

To find the relationship between the opinion arguments and the represented issues, factors, and facts in ACTs, **Chapter 5** investigated how these ACTs can be merged and expressed in terms of ADFs [54] where the nodes represent the components from the ACTs of the Oral Hearing, the links between the children and their parent reflect the relationship type between the components, and the acceptance conditions are used to determine the accepted nodes (argument components). A Case ADF was instantiated from the merged petitioner and respondent Justices' ACTs, showing all the possible acceptance conditions. This was revisited in **Chapter 6** to compare the Case ADF instantiated from the Oral Hearing with the one produced from the opinion and to determine the accepted components that support the opinion argument.

Moreover, the legal decisions of ten cases from the Automobile Exception domain were analysed in **Chapter 6** to encapsulate a theory of the domain and instantiate a domain ADF of issues, factors and facts. The acceptance conditions were then implemented using Prolog procedures. Each test in the acceptance conditions is applied in a separate clause, using the set of factors currently identified as present in the case, before proceeding to the next factor. Accepted factors are added to a list of applicable factors, and ordered to ensure that a report is provided giving the status of every factor in the domain giving the case base-level factors or facts. The inclusion of facts in the case representation increased transparency in the ascription of base factors to cases. The program showed that 9 out of the 10 cases were decided correctly, the exception being for the most recent case where certain decisions in precedent cases were overruled or reinterpreted. In addition, the domain ADF was modified to provide justifications for dissenting opinions by defining additional acceptance conditions' priorities differently to support other opinions using the same ADF statements and links. The exploration of further opinions will be considered in future work, as discussed in Section 8.3.4.

The ADF definition shows a clear correspondence between the structure of the ADF and the factor hierarchy in CATO. This motivates the demonstration of the suitability of ADFs in expressing the design of legal case-based systems, and instantiating an ADF for the domain of US Trade Secrets (CATO, IBP). **Chapter 5** defined the acceptance conditions for the domain using the factor hierarchy of CATO [10], the logical model of IBP [17] and [18], in terms of the base factors, and the subset of case representations used in AGATHA [57, 58]. These acceptance conditions were then executed in Prolog, in the same way as in the Automobile Exception domain, by posing a query which gets the first factor to be decided with the case and its base-level factors as arguments. The initial results showed that 25 out of the 32 cases (78.1%) were decided correctly. This is very close to CATO's results (77.8%) reported in [55], which is to be expected given the same analysis of the cases in the domain. Further investigations were conducted to consider possible refinements by adding missing factors, removing wrongly attributed factors or changing the default. The results following these refinements improved performance to 96.8% correct.

The approach performed well on the third domain, Wild Animals, producing correct results for all five cases expressed in terms of the factors in [30]. Following the same approach, the ADF was instantiated from the domain factors and the acceptance conditions, from which an executable program was implemented. This shows how the methodology has been applied and refined in different legal domains to produce the outcomes of legal cases described as sets of factors, according to a theory of a particular domain, based on a set of precedent cases relating to that domain **(Objective 3)**. In all three domains, ADFs were applied to play a role akin to that played by Entity-Relationship models in the design of database systems, providing the ability to record the design of the knowledge derived from a number of precedents in a particular case law domain **(Objective 2)**. In general the method can be applied to any domain for which factor-based reasoning is appropriate. This knowledge engineering approach ensures the quality of the modularisation of the knowledge base achieved by using an ADF, in that the rules are concerned only with the acceptance of a single statement (tightly coherent), and the acceptability of a node is determined only by the children of that node (loosely coupled).

To address **Objective 4** a number of evaluations were conducted in **Chapter 7** to compare the performance of the ADF approach in the three legal domains. This concluded that the success of the approach depends highly on:

- The quality of the analysis, i.e. examination of the texts of the decisions explained why the results diverged, and suggested ways in which the analysis could be improved, either at the case level by changing the factors attributed, or at the domain level by including additional supporting or attacking links; and
- The domain model being in a stable state (the second stage of Levi's life cycle explained in [88] as summarised in **Chapter 3**). This can be seen in the wrong decision results from the reinterpretation and overruling of precedents in the law in the Automobile Exception domain.

Comparison with other legal case-based reasoning systems in [55] showed that applying a few refinements in the seven wrong cases, readily identified using the methodology, considerably improved the performance to make the ANGELIC Trade Secrets the best of all the legal reasoning systems. Although these findings are promising, **Chapter 7** introduced additional aspects that need to be considered to improve the performance and the output from the program **(Objective 5)**. These include comparison of the transparency of the outputs and explanation of the reasoning at different reasoning levels (factors and facts), in addition to explanation using portions of precedents, which could specify the precedents most relevant to the specific point to justify the acceptance or rejection of particular nodes, and also reasoning with values. All these aspects set the foundations for future work as discussed in Section 8.3.

8.2 Main Findings and Contributions

By providing a summary of the research, this section revisits the research questions mentioned in Chapter 1 to determine how they have been addressed in the thesis and how they are related to the thesis contributions, as follows:

8.2.1 RQ1-Does abstract argumentation provide a format capable of capturing a variety of case law domains?

This thesis proposed ANGELIC, a new knowledge engineering methodology for legal reasoning using a powerful generalisation of Dung's abstract argumentation frameworks [61], *Abstract Dialectical Frameworks* [53, 54]. ADFs generalise abstract argumentation frameworks by replacing Dung's single acceptance condition (that all attackers be defeated) with acceptance conditions local to each particular node. The nodes in ADFs are statements, not necessarily arguments, and can be related by a variety of types of links, not just attackers. Legal decisions are typically not truth functional, but require both a context (e.g. a set of cases) and a procedure (e.g. *stare decisis*). Both of these can be represented in the acceptance conditions of ADFs. ANGELIC is then applied over various case law legal domains: Automobile Exceptions to the Fourth Amendment, Wild Animals and US Trade Secrets using analyses produced by different people for different purposes as presented in the previous section to encapsulate and apply theories of case law represented as ADFs. The statements of an ADF are the set of all the issues, intermediate concerns and base-level factors. The links forms the supporting and attacking links, and the acceptance conditions are defined for each abstract factor (non-leaf node) in terms of their supporting and attacking children. Each acceptance condition is a propositional function constructed from the factors in the legal domain potentially based on the decisions in precedent cases. This work has been published in [3], [5], and [8], and is relevant to the following research contributions:

- *The application of ADFs as powerful, general, abstract frameworks for argumentation to drive and record the design of a knowledge base to encapsulate a body of case law.*
- *A solid formal basis for factor-based reasoning using ADFs that was lacking from existing systems used for reasoning about cases.*

8.2.2 RQ2-How can computational dialogue be used to identify relevant components to instantiate the abstract argumentation format used?

A semi-automated dialogue model (published in [2, 4]) was implemented in Chapter 4 to provide a visual representation for the analysis of the Oral Hearing dialogues. The role of these hearings is to identify the components used to construct an appropriate test that can form the basis of an argument which will resolve the case. The specification of the computational dialogue covers the dialogue moves used by participants to assert and modify argument components in term of issues, factors and facts; an Argument Components Tree (ACT) structure showing how these components relate to one another; and an ontology to capture the relevant concepts of the domain that are used in the dialogue and ACTs. The dialogue model has been applied to some widely discussed Supreme Court cases. The program constructs the ACTs of the participants as the dialogue proceeds and the final resulting trees can be then used as input to the next stage of the judicial process, which will organise these components by merging the Justices' ACTs and a Case ADF that is used to decide the case as in Chapter 5 and Chapter 6. The contributions related to this research question are stated below:

- *Examination of the legal procedure in the US Supreme court and characterisation of the three sub-dialogues of the Oral Hearing in terms of their initial state, and individual and collective goals as required by Walton and Krabbe in [133].*
- *Provision of a dialogue representation model, which provides a structured analysis of US Supreme Court Oral Hearings that enables the construction of* argument component trees *in which the issues, facts and factors are proposed and refined through the use of a set of defined speech acts. This involved a manual mark-up for the argument components of the Oral Hearing transcripts of two legal cases and a workflow for analysing the legal cases and relate the produced argument components to the legal opinion. This mark-up can then be input to a program to provide a visual representation by executing the dialogue.*
- *Demonstration of the correspondence between the components of the arguments in the dialogue representation and the legal opinion.*

8.2.3 RQ3-How can we produce an executable program from this argumentation format in a standard manner such that cases are decided by the program to an acceptable degree of accuracy?

This was achieved by defining the ANGELIC methodology described in Chapter 5, and applied in Chapter 6 to three legal domains. After the analysis and representation of three legal domains discussed in the summary (Section 8.1) was completed, the approach was applied to the existing knowledge of US Trade Secrets as represented in CATO and IBP, exploiting the similarity between the ADF definition and the factor hierarchy. Next, ADFs in the domain of Wild Animals cases were constructed (*Popov v Hayashi* and related cases were used as modelled in [30]) which contain different representations. Finally, a set of cases were used from the Automobile Exception to The Fourth Amendment, which had been discussed in the literature in [29, 76, 115], but had been represented only partially to illustrate particular points in those papers, so an analysis had to be specifically developed for this thesis. After that, a move from the analysis to an executable program was conducted in a direct and immediate way using Prolog, translating the acceptance conditions into Prolog procedures, adding some control and reporting information, and then executing the resulting program. Comparison of the program results with the actual outcomes of the cases allowed the theory to be refined, exploiting the software engineering benefits afforded by the ADF representation. Moreover, ANGELIC was shown to be performing very well compared to other legal reasoning systems.

The contributions relating to RQ3 are:

- *ANGELIC, a methodology for developing a decision justification model using Abstract Dialectical Frameworks that is able to encapsulate and apply theories of case law represented as ADFs. The method has been tested by producing programs for three domains: US Trade Secrets from the factor hierarchy of CATO [10] and IBP [55] (ANGELIC Trade Secrets), the Wild Animals domain from [30] (ANGELIC Animals), and a set of cases in the Automobile Exception to The Fourth Amendment domain (ANGELIC Automobile),* as proposed in [3, 5, 7, 8].

- *A comparative evaluation between the methodology applied in this thesis and previous legal case-based reasoning systems.*

8.2.4 RQ4-How can this executable representation be used to evaluate and refine the analysis?

This question was addressed by identifying ways in which the ADF can be modified with respect to misclessified test cases as presented in Chapter 6 and Chapter 7 and published in [5, 7, 8]. The ADF design facilitates refinement of the program in light of its performance on test cases, to solve defects in the initial analysis. When using a set of factors identified for the case, programs based on the ADF design provide very transparent output that identifies precisely where the outcomes suggested by the implementation diverge from the actual case decision. After running the program, and noting any divergences from the actual outcomes, reading the original decision texts can suggest up to four solutions to refine the behaviour of the program. These are as stated in Chapter 7:

- Removing a factor wrongly attributed to the case
- Adding a factor wrongly omitted from the case
- Modifying an acceptance condition: e.g. changing the priorities
- Modifying the ADF: e.g. adding a supporting or attacking node for the problem node.

The contribution for this question is *a comparative evaluation between the domains that tests the ease of implementation, the performance and efficacy of the resulting program, the ease of refinement and the transparency of the reasoning*.

8.2.5 RQ5-What additional information needs to be provided to support the particular aspects of applications such as explanation?

These different aspects were explored in Chapter 7, mainly to examine the quality of explanation in the reports produced from the program output, and the actual opinions found in cases at different reasoning levels; by re-ordering the elements and adding a little linking text as explained below:

1. The reasoning of the first level used the base factors as leaf nodes: this yielded output that provided a general explanation, with a lack of specific reference to the case and no summary of case facts.
2. The next step was to consider the integration of facts, one level down, which yielded a clear improvement in the quality of the explanation by providing justification for the acceptance of the base-level factors. The order of the output states the status of the base-level factors first (children) and then indicates whether the abstract factor (parent of base-level factors) is satisfied or not.

3. To produce an output that is very closely related to the real decision requires the consideration of precedent citations: that is, when an acceptance condition is related to a portion of a precedent case. This will require annotating the acceptance conditions with citations to the precedent cases they are taken from.

4. In addition to the majority opinion, rules for producing the dissenting opinion were also considered in Chapter 6 to improve the output of the program.

Normally, legal decisions use precedents at particular points to identify questions and issues to be addressed, and to justify answers and suggest consequences. In addition, evaluation of reasoning types is conducted using portions of precedents rather than whole cases. This is effectively what is done in the ADF approach: competing factors are considered in the context of accepting or rejecting a particular node. Furthermore, although values are rarely stated explicitly in the decision texts, they might be considered as comments within the acceptance conditions to explain why the tests are ordered as they are. The priority reflects an ordering of the values they promote, based on a set of applicable values, and an ordering of them. In ADF terms, this means that values can justify the inclusion of a test within an acceptance condition, or a node representing a factor to be used in such tests. Since the value preferences are effectively local to particular nodes, it is possible to express different preferences in different contexts, which may add a desirable flexibility. A means of extracting values from an ADF, from associations between values and factors, was given in [6]. In the current methodology, however, values inform the design but do not form part of it. This discussion has also been reported in [8], covering the following related contributions:

- *A means of improving the justification of the program by incorporating the* case facts*, to provide closeness to the reported opinion.*

- *Identification of ways to improve the explanation of the output from the decision justification model using portions of precedents and the generation of multiple opinions.*

The implications of this research work are that the proposed method opens up new possibilities for developing more realistic and better, more objectively engineered models to simulate legal decision procedures. The work contributes to the long tradition of reasoning with cases in AI and Law by reporting results on the role of using a formal, well-defined theory of argumentation, Abstract Dialectical Frameworks, to encapsulate the analyses of a variety of case law domains, from different resources, in a common format.

8.3 Future Directions

Future work can extend the functionality of the provided method and enhance the application of the proposed models. This thesis lays the foundations for some potential directions for future work, as discussed below.

8.3.1 Evaluation of Abstract Dialectical Frameworks

This research proposed that ADFs play a role in encapsulating the knowledge of a legal domain and defining acceptance conditions for the ADF nodes to provide reasoning and produce justification for the legal decision. Although [54], [51] and [53] have shown ADFs to be formally sound and their formal properties have been investigated, the ADFs explained here have not explicitly used these properties. The ADFs used here are rather uncomplicated, since for these domains the ADFs are cycle-free, so that the grounded, preferred and stable semantics coincide. Nevertheless, the implemented programs based on a design in the form of a cycle-free ADF are important to show that ADFs themselves are tractably computable.

However, it is important to consider the output in terms of the ADF semantics in order to take advantage of the theoretical developments of ADFs and apply them directly in the legal domain. Therefore, tools for evaluating the specifications of the ADF, such as the *Diamond* system [63], can be used as a starting point. Diamond directly implements ADFs and produces extensions corresponding to a variety of semantics for a given ADF. The first step will be to compare the grounded extension produced from Diamond with the result from the Prolog program. When considering multiple opinions, it will be interesting to produce the preferred semantics to capture variant opinions in a case.

An initial investigation has been done on Diamond using the Wild Animals domain in collaboration with the founders of the Diamond system from Leipzig University. The ADF for this domain has been translated into the defined ADF code in Diamond, and the various semantics models have been produced. However, since Diamond is not yet fully mature, another attempt will be made with the latest version of Diamond once it is fully operational.

8.3.2 Considering Values and Dimensions

One potential development from the research presented in this thesis is to discuss the possibility of going beyond what is already present in the precedents, so that not all legislations need to be *a fortiori*. Three main approaches were presented in [6] to reduce the number of comparisons of factors and issues, including partitioning so that the problem is decomposed into smaller problems, each with fewer factors, although the total number of factors remains the same. A representation using 2-regular ADFs was adopted, where pairs of factors are compared. For a concrete illustration, the approach has been tested in the CATO system [10] where every parent has exactly two children. Following this example, it is possible to assign factors to cases in a Boolean manner, without varying degrees of presence. Further, the role of values in determining the acceptance of the node is considered. Values are attached to the leaf nodes (base-level factors), where the parents' values are determined from their children's values, as explained in [6].

This returns the discussion to reasoning with values in Chapter 7. It might be useful in future to provide links to values to improve explanations of the program output, but this will affect only presentation, not the decision making, since the effect of values and preferences for them is already cached out in the representation, in the form of the components included, the tests of the acceptance conditions and the priority of these tests.

The work in this thesis and [6] assigns factors to cases in a Boolean manner, present or absent, while in practice there are compelling reasons to see the presence of factors as a matter of degree. The ascription of factors is not always a Boolean, black and white, affair, but rather may sometimes be a matter of degree. Another potential development of this research is to consider the strength of reasons in legal arguments. This has been introduced in [6] and an initial investigation is described in [9], reviving the use of dimensions, from which factors were originally developed, as a bridge between facts and factors to achieve a more precise representation of the legal significance of factors, in terms of the extent to which they contribute to a finding for the plaintiff (or defendant). Dimension points were also used as representations of the facts of a case to provide a way of allowing for different degrees of presence of factors [24]. Factors are seen as ranges on dimensions, and rules are used to establish how far along the dimension the case is, on the basis of its facts. The decision will then require consideration of whether the plaintiff is sufficiently favoured to decide in his favour. Further illustration using Wild Animals cases is provided to explain this approach. Future work may involve investigation of the role of dimensions in bridging between facts and factors, a long-standing issue in AI and law.

8.3.3 Statement Types in Legal Argument

Also worthy of exploration is revisiting the reasoning stages. As proposed in Chapter 2, the reasoning process begins with evidence and ends with the decision of the court, and passes through several stages in between. Following the idea of considering the dimension presented in [9], in the future, further attempts will be considered to provide the whole picture of reasoning with legal cases from evidence to verdict, focusing in particular on the transitions between stages, and the different types of statement used at the various stages. This involves consideration of the following:

- The different types of statements used in legal reasoning. Conflating these different types can lead to problems in deciding how their acceptability should be established.
- The transition between reasoning about facts to reasoning about law. The importance of this transition was noted in [50], but the vast majority of work has been on one side or the other, and how this barrier is crossed remains an open question;
- How and where uncertainty is dealt with. Although many legal concepts are characterised by *open texture* [130] and so have a *penumbra of uncertainty* [80], the verdict itself can permit no qualification. How and where the doubts are resolved remains an open question. Most approaches (e.g. [114]) represent legal concepts only as Booleans. For example factors are considered present or absent and values are promoted or demoted, whereas differing degrees of presence and absence (e.g. [6, 37, 58]) need to be recognised.

This will help in investigating why sometimes Bayesian reasoning can be used fruitfully when reasoning from evidence to legal facts, but not when reasoning from factors to issues. Also when should the reasoning be seen as Boolean and when should varying degrees of uncertainty and degree be introduced and eliminated? How could dimensions play a pivotal role in bridging the gap between the realm of the world and the realm of law? Given the different types of statements, ADFs can be applied to provide a very natural tool for differentiating between statement types, by characterising statements not as a homogeneous type, but as types appropriate to the stage of reasoning, and which will form distinct levels in the ADF.

8.3.4 Beyond Majority Opinion

The Prolog program presented in this research directly implements the acceptance conditions for each node in the ADF, and, for a given set of facts, the program always produces a single outcome relevant to the grounded semantics of ADFs. This is to be expected given the absence of cycles in the domain ADF, which captures the tree-based structure of CATO's factor hierarchy. The program has been extended to consider dissenting opinions by defining new acceptance conditions to produce the Justices' justification of the disagreement. However, the long-term goal is to go beyond that, by producing a variant program to generate all the possible arguments for the majority, dissenting or concurring arguments required to resolve conflicts explicitly, so as to facilitate comparison with ASPIC+ and give a complete picture of the potential arguments and their attacks. Such a program would effectively represents the Justice Conference stage of the US Supreme Court.

8.3.5 Multi-agent Deliberation Dialogue

The presented dialogue model provides insight into deliberation dialogues by a close study of one extensive source of well-documented examples: US Supreme Court Oral Hearings. Looking at these rich, varied, real world examples of deliberation improves understanding of what deliberation really involves. Deliberation dialogues are as yet less thoroughly understood than the closely related class of persuasion dialogues [25]. The process of the US Supreme Court breaks the deliberation into several stages. First, in Oral Hearings the components that might be used to form the rule are proposed. After that, the Justices form, apply and justify a particular rule, or test, from these components. Mutually acceptable rules emerge to form the legal decision. This process of coming to agreement on a new, jointly acceptable rule stands in need of exploration and clarification.

One possible topic for future work is to consider this model as an example for further studies in deliberation dialogues. For example, this dialogue model could represent a multi-agent deliberation dialogue, where there are a number of agents participating in the dialogue. Each agent has a different role, and while some agents participate in all the dialogues, others have a role in only some of the dialogues, as in e.g. [137]. Examples include a chair agent who takes control over the procedure of the dialogues; one or more Justice agent(s) participating in all the nested dialogues; and the counsel agent (plaintiff agent or defendant agent) participating in some of the dialogues. The ACT in the dialogue system plays the role of the commitment store commonly used in agent dialogue systems. Each agent will have its own ACT, so that for each nested dialogue one ACT is formed for each participant for the counsel agent (petitioner agent or respondent agent) and one for the Justice agent. This can be generalised and exploited for a range of less formal deliberation dialogues.

8.3.6 Legal Text Processing

Another potential area for future work could be the development of a fully automated dialogue model using approaches in legal text processing for legal documents such as legal transcripts. One possible approach is by using GATE, a General Architecture for Text Engineering [1], to automatically extract the argument components from the legal text. Preliminary investigation on this topic have been conducted in two projects by final year undergraduate students ([60], [69]) at the University of Liverpool, using the marked-up legal transcripts for the two case studies presented in Appendix B and Appendix C. The next step is to consider different NLP approaches in collaboration with legal experts and NLP researchers in the legal domain to help in capturing the dialogue moves directly from the free text, e.g. [140, 141].

Moreover, post-processing for the decision justification in the program output can be reconsidered to resemble the texts of actual decisions as presented in the explanations in Chapter 7.

8.3.7 Other Jurisdictions, Other Domains and New (Undecided) Cases

Would the methodology presented in this thesis be applicable in different jurisdictions? Could it be extended to more legal domains? And could it be tested on more recent or undecided cases? This is one research direction that is worth exploring to evaluate the functionality of the provided method and consider the requirements for possible refinements and improvements in the future.

The examples presented in this thesis have all been taken from the AI and Law literature, including what is probably the most substantial analysis to date (CATO/IBP). Even so, the examples are still relatively small and limited to one jurisdiction. Ideally this research will be generalised, by first testing the ADF of the legal domains discussed in this thesis (e.g. US Trade Secret or Automobile Exception) over recent cases or undecided cases and applying the required refinements. Considering this improvement will in turn help in extending the application of this model to encapsulate the knowledge of freshly analysed legal domains. Conducting these further tests on the model would motivate studying the requirements for generalising the application to cover a range of legal jurisdictions.

The true test of a methodology and its supporting tools comes from its use in practice, and so it is hoped that the proposal will be adopted by people wishing to build substantial systems to perform factor-based reasoning in a legal domain. Such a tool cannot be the subject of a research project carried out by only academic computer scientists: it requires the motivation and personnel that only a practical project can provide. Practical applications appear to be moving closer[2], which increases the hope and expectation that the methodology will be used and proven in practice.

[1] https://gate.ac.uk/

[2] As evidenced by a spate of articles such as the *Role of AI in Law*, published in *The Times* newspaper in the UK and available at http://raconteur.net/business/time-for-technology-to-take-over, and the many discussion threads on LinkedIn for the group for the International Association for AI and Law.

8.3.8 Generalisation to Non Legal Context

Another potential area for future work would be to explore whether techniques for modelling legal reasoning can be transferred to other open texture domains. The research findings raise the belief that factor-based reasoning has implications much wider than law.

This style of reasoning is appropriate to the sorts of concept well recognised in philosophy as having a "family resemblance" [138] or "open texture" [130]. Many concepts are not susceptible to definition in terms of necessary and sufficient conditions, so another way of expressing the conditions for their acceptance is required. An understanding of the concept requires an understanding of the context and the arguments on which this consensus is based, and a deep understanding of the concept and its liability to change requires an understanding of the considerations (factors) on which these arguments are based. As such, the following steps have been identified to generalise the model and evaluate its effectiveness.

- Apply the representation model over dialogues found in broadcast media.
- Select a suitable dataset from a non-legal context using, for example, the UCI Machine Learning Repository[3]. This could then be compared with the experiments reported in [136] and [137].
- Describe the new context, and interpret the classification used in the selected dataset into a factor hierarchy tree.
- Map the *factor hierarchy tree* into the defined representation model.
- Apply the selected dataset in the developed system and record the results, which are the *accepted factors*.
- Compare the results obtained with the classification results in the dataset, and analyse the findings.

8.4 Summary

At this point, the story of this thesis is complete, showing how the proposed research questions have been addressed and outlining the contributions of this research to the domain of argumentation in AI and Law. However, this is not the end: further extensions and new versions can be explored to shape the future of the work and demonstrate the real advantages of the method advocated in this thesis.

[3]UCI Machine Learning Repository URL: https://archive.ics.uci.edu/ml/datasets.html

Appendix A

Design Documentation for the Java Implementation of the Legal Dialogue Representation

This appendix provides the design documentation for the Java program that was implemented to produce a visual representation for the legal dialogues in the form of ACTs (as described in Chapter 4), using Graphviz Java API. Section A.1 illustrates the class diagram for the model, showing the members and methods for each class separately.

A.1 The Design of the Legal Dialogue Representation

The class diagram for the dialogue representation model is shown in Figure A.1. As stated in Chapter 4, the program was implemented in Java to form the analysis of the moves in the Oral Hearing dialogues, and produce an ACT for each player in each dialogue. *Main* is the control class, Figure A.2, where all the dialogue moves are created. The main specifications/roles of the moves methods are explained below:

- *OpenCase*: instantiate a new case object, which in turn instantiates a new Oral Hearing (OH) and three nested dialogues objects implicitly. Figures A.3 and A.4 illustrate the members and methods for the case and Oral Hearing classes.
- *OpenDialogue*: Open a certain dialogue and specify the dialogue players. The method also instantiates a player ACT (PACT) objects for each player in the dialogue implicitly. The specifications for ACT and PACT classes are presented in Figures A.5 and A.6.
- *AddIssue, CombineIssue, AddIssueRel*: are three methods related to the *AssertIssue* and *CombineIssues* moves respectively. *AddIssue* instantiates a new issue object in the case, if the issue has not been added before, and adds the issue to the player's ACT (PlayerIssue class). To combine issues, a new object, *IssueRel*, is instantiated for the related issues. The issue is added implicitly if it was not in the player's ACT or not in the case. Figure A.7 shows the Issue, PlayerIssue and IssueRel classes.

- *AddFactor, CombineFactor, AddFactorRel, EmphFactor*: A new factor object is instantiated in the "AddFactor" method, which is related to the *AssertFactor* move. Every factor has a parent. Once a factor object is added to a case, a new "Player_Factor" object is instantiated and the factor's parent object is assigned to the new factor. The parent can be an Issue (Player_Issue object), a combination of issues (IssueRel object), an abstract factor (Player_Factor Object) or a combination of abstract factors (Factor_Rel object). If the parent does not exist, the program implicitly instantiates a new object based on the object type. Figure A.8 illustrates the factor classes. Moreover, Factors can also be combined by instantiating a new object from the "FactorRel'' class or emphasised by using the "EmphFactor'' method, applying *CombineFactors* and *EmphasiseFactor* moves.

- *AddFact, CombineFact, AddFactRel, EmphFact* Figure A.9 shows all the related fact classes. Similar to the factor methods, "AddFactor" instantiates a new fact object from the "Player_Fact" class, if an object for the same fact already exists in the case. Otherwise, the method instantiates a new "Fact" object. Facts can also be combined (*CombineFact move*) using the Fact_Rel class, or emphasised (*EmphasiseFact move*). The fact's parent can be either a base factor (Player_Factor object) or a combination of base factors (Factor_Rel object).

- *DisplayGraphACT* This method is called after each move to produce an updated ACT for the current move. The ACTs are presented using a *GraphViz Java API*[1]: an object from the "GraphViz" class (Figure A.10) is instantiated for each player in a dialogue by simply writing the GraphViz dot code, and saving the code in a source file under a specific type image format.

[1]GraphViz URL: http://www.graphviz.org. GraphViz Java API: https://github.com/jabbalaci/graphviz-java-api.

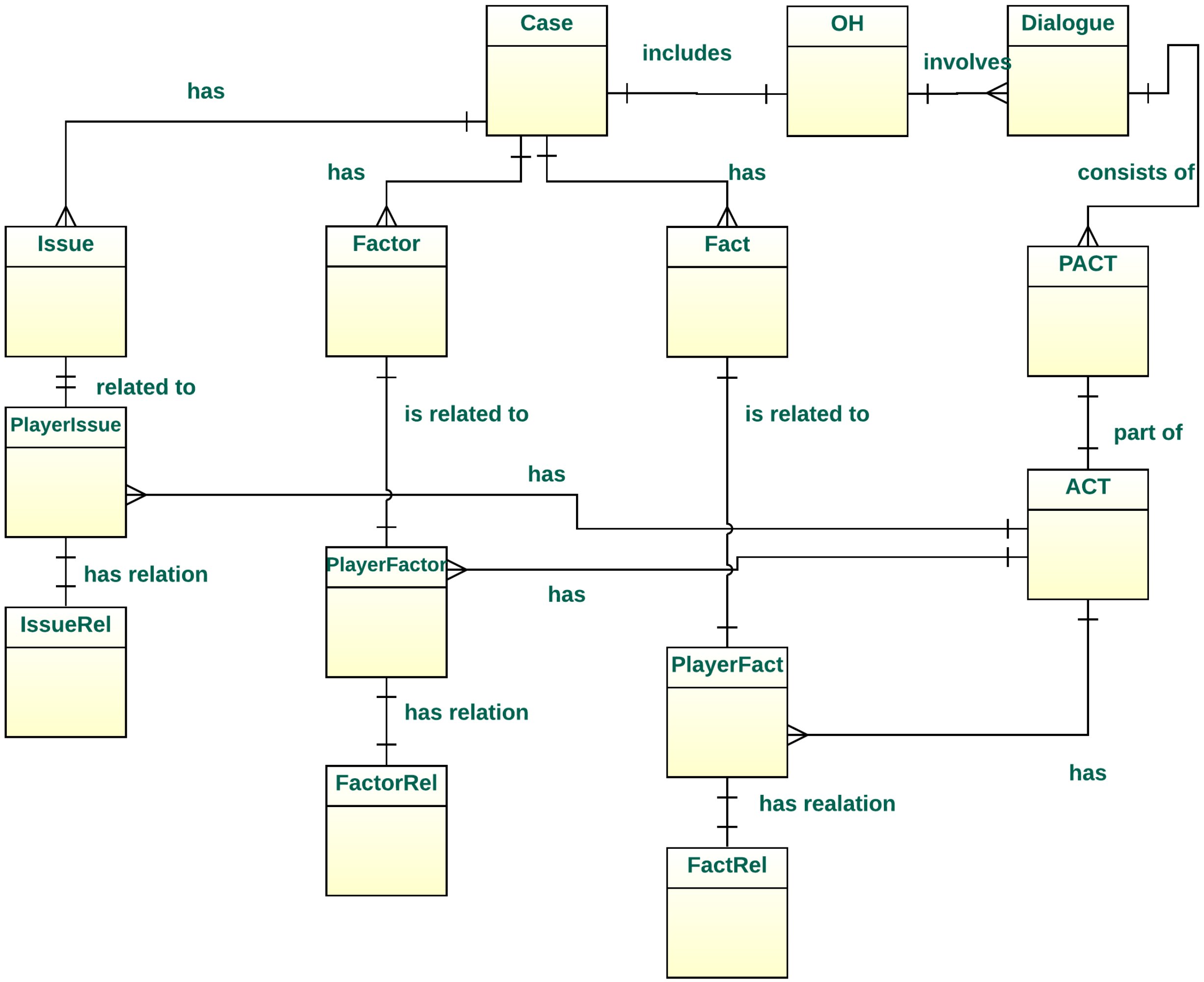


FIGURE A.1: Class Diagram

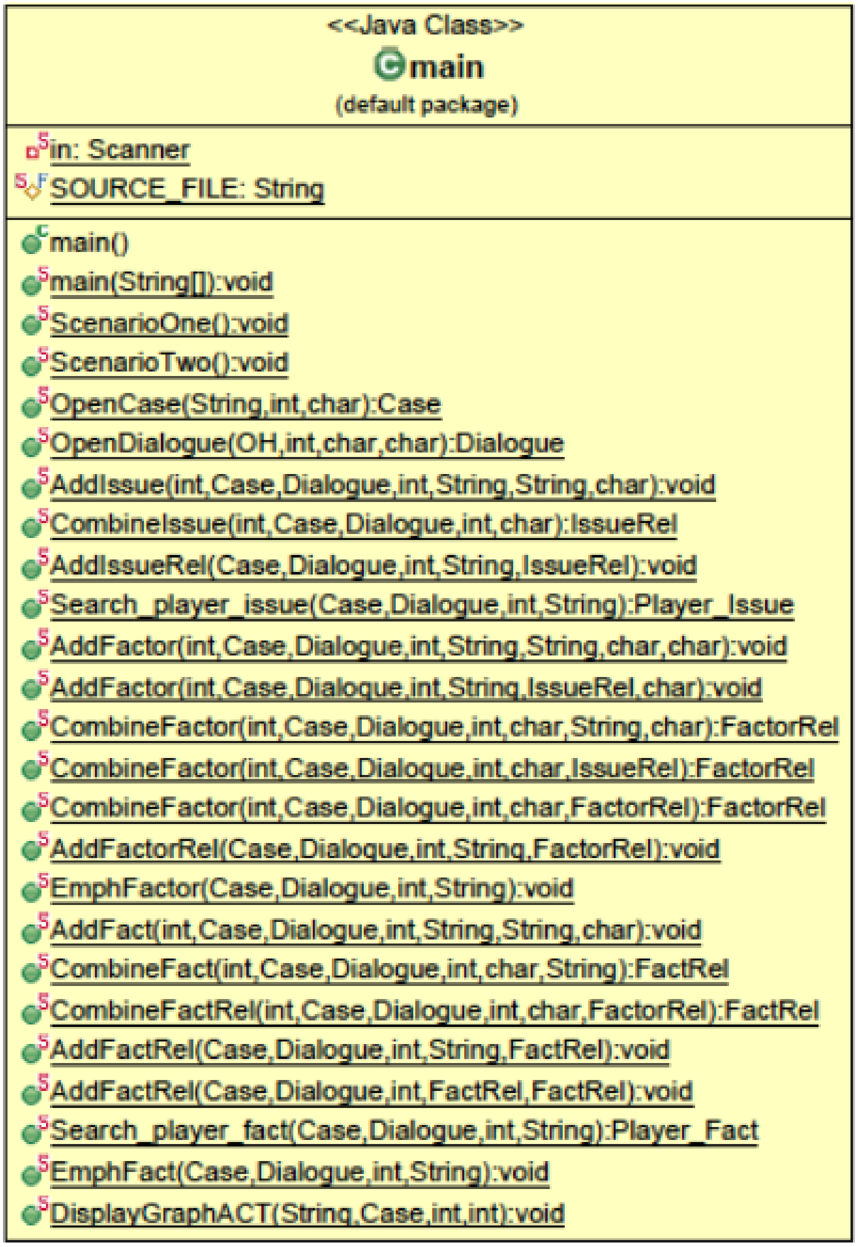


FIGURE A.2: Main Class

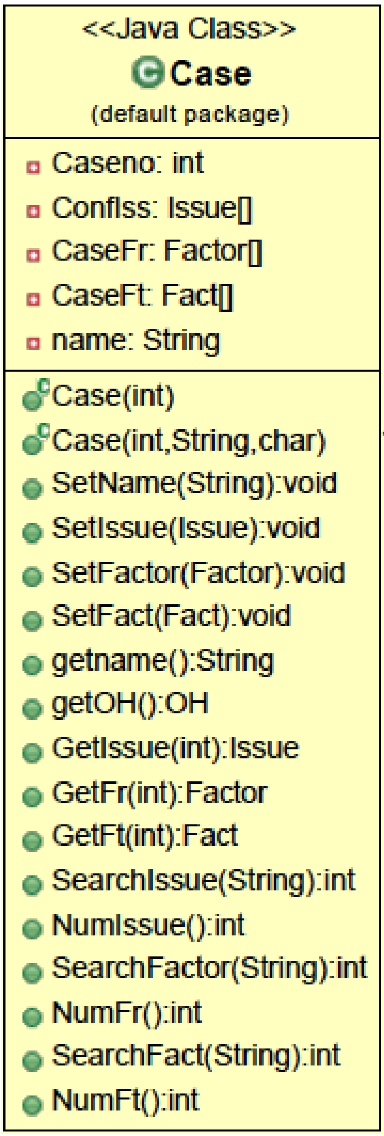
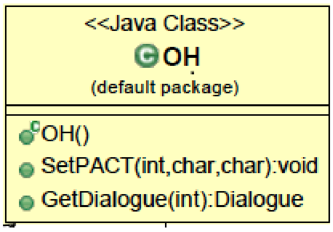
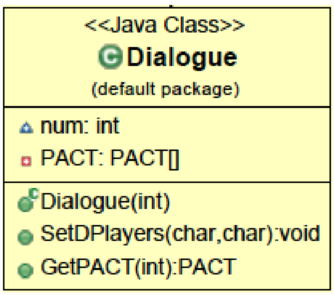


FIGURE A.3: Case Class



a. Oral Hearing Class



b. Dialogue Class

FIGURE A.4: Oral Hearing and Dialogue Classes

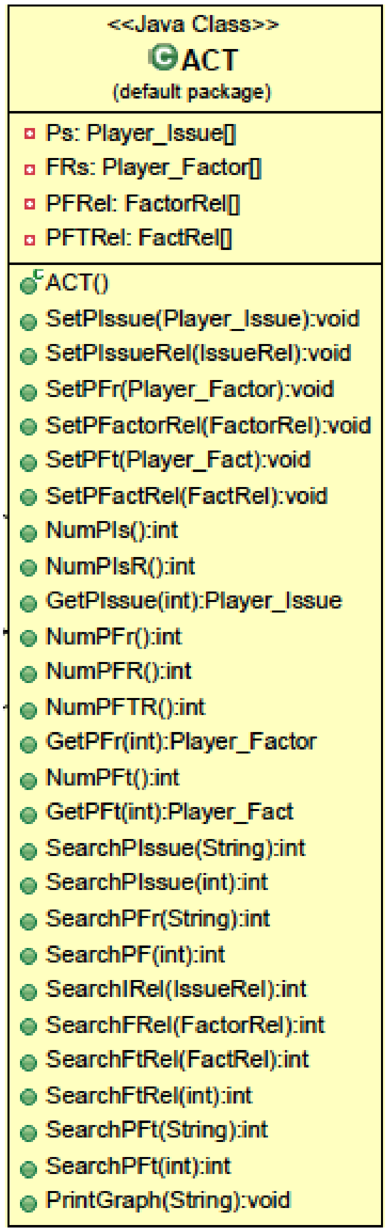


FIGURE A.5: ACT Class

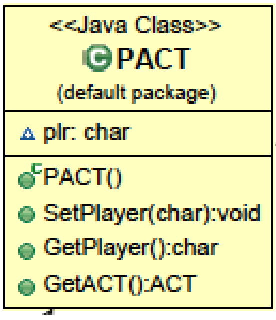


FIGURE A.6: Player ACT Class

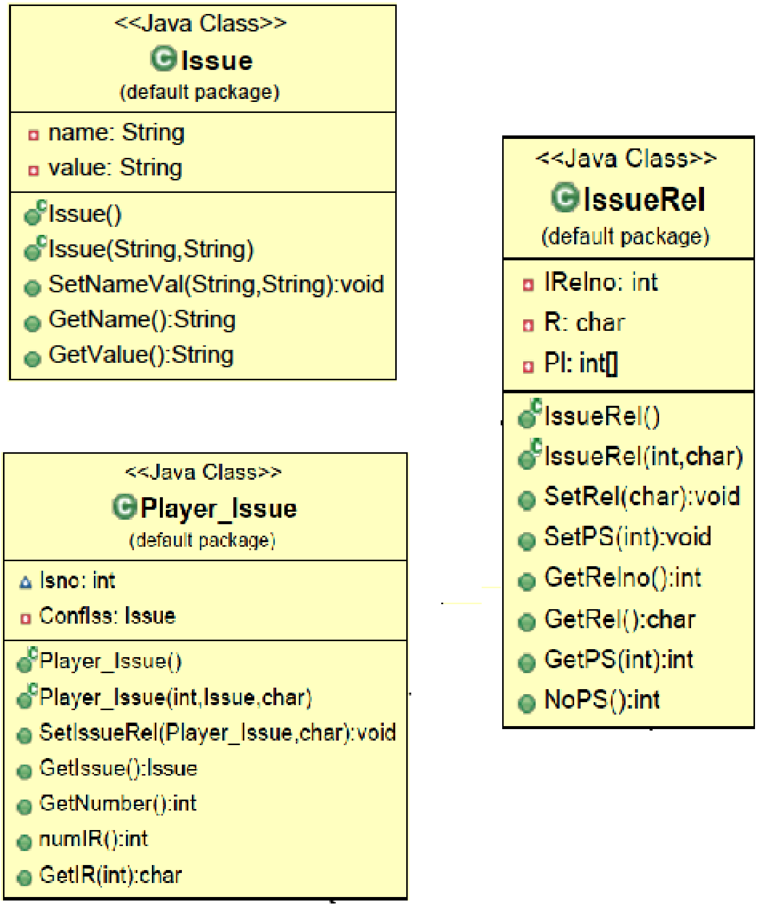


FIGURE A.7: Issue, PlayerIssue and IssueRelation Classes

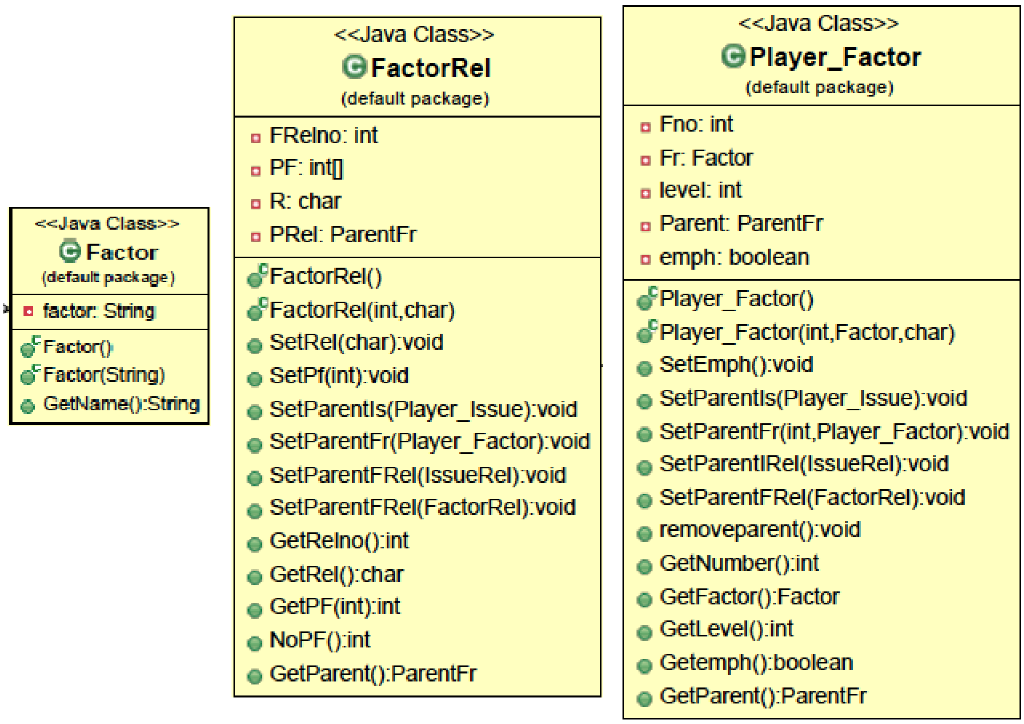


FIGURE A.8: Factor, PlayerFactor, FactorRelation Classes

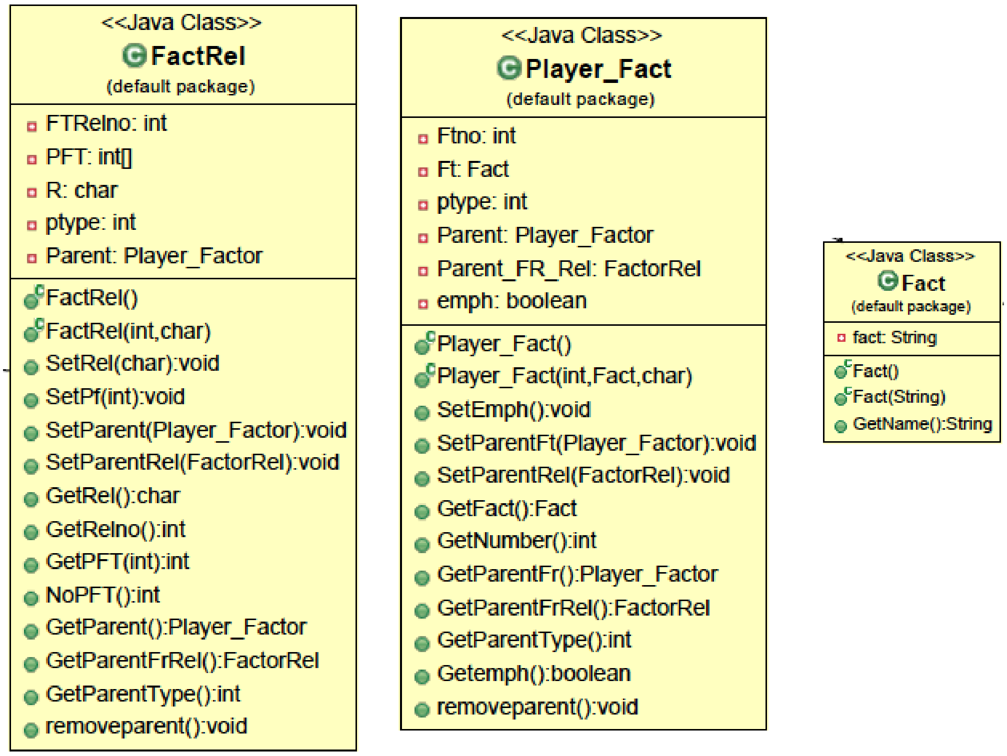


FIGURE A.9: Fact, PlayerFact, FactRelation Classes

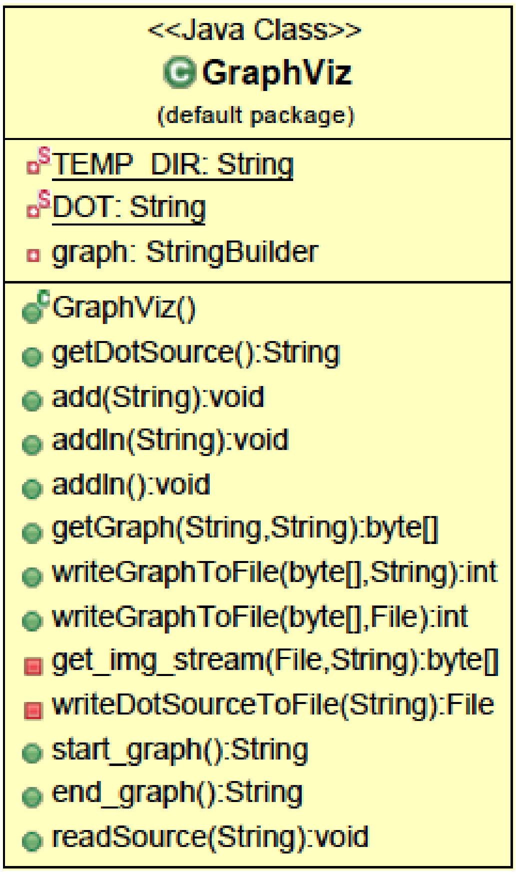


FIGURE A.10: Graphviz Class

Appendix B

Case Study: US v. Chadwick Dialogue Moves and ACTs

This appendix provides the dialogue moves for the marked-up Oral Hearing transcript of the case study *US v. Chadwick*. Three dialogues are analysed in this transcript: Petitioner dialogue (Section B.1); Respondent dialogue (Section B.2) and Petitioner rebuttal (Section B.3). Each dialogue is divided into a number of contexts, and an ACT for each player is produced after the petitioner and respondent dialogues, and updated after the petitioner rebuttal dialogue.

B.1 Petitioner Dialogue

Context 1: Introducing the Case

```
OpenCase(ChiefJustice, USvCh)
OpenDialogue(ChiefJustice,PD,Raymond Randolph,USvCh)
AssertIssue(USvCh,PD,P, WarrantRequired)
AssertIssue(USvCh,PD,P, WarrantLessSearch)
AssertCombIssues(USvChPD,P, +, {WarrantLessSearch, WarrantRequired})
AssertFactor(USvCh,PD,P, container, WarrantLessSearch, I,PC )
AssertFact(USvCh,PD,P, footlocker, Container, F, T)
AssertFactor(USvCh,PD,P, lockStatus, WarrantLessSearch, I,T )
AssertFact(USvCh,PD,P, locked, lockStatus, F, T)
AssertFactor(USvCh,PD,P, lawfulSeized, WarrantLessSearch, I, L)
AssertCombFactors(USvCh,PD,P, and, {container, lockStatus},
{WarrantLessSearch+WarrantRequired}, IR, T, N )
AssertCombFactors(USvCh,PD,P, and, {lawfulSeized ,
{container, lockStatus}}, {WarrantLessSearch+WarrantRequired}, IR, N )
AssertFactor(USvCh,PD,P, probableCause, WarrantLessSearch, I, PC)
AssertFactor(USvCh,PD,P, contraband, probableCause, F, PC)
AssertFact(USvCh,PD,P, heavyWeight, contraband, F, T)
AssertFact(USvCh,PD,P, leaking, contraband, F, T)
AssertCombFacts(USvCh,PD,P, and, {heavyWeight, leaking, smell},
contraband, F, T )
```

Context 2: Shipping Status

```
AssertIssue(USvCh,PD,J, WarrantRequired)
AssertIssue(USvCh,PD,J, WarrantLessSearch)
AssertCombIssues(USvCh PD, J, +, {WarrantLessSearch, WarrantRequired})
AssertFactor(USvCh,PD,P, container, {WarrantLessSearch+ WarrantRequired}, IR, PC )
```

```
AssertFactor(USvCh,PD,J, shippingStatus, {WarrantLessSearch+ WarrantRequired},
 IR, N)
AssertCombFactors(USvCh,PD,J, and, {container, shippingStatus},
 {WarrantLessSearch+WarrantRequired}, IR, T, N )
AssertFactor(USvCh,PD,P, shippingStatus, WarrantLessSearch, I, N)
AssertCombFactors(USvCh,PD,P, and, {container, shippingStatus},
 WarrantLessSearch, I, T, N )
AssertFact(USvCh,PD,P, onTrain, shippingStatus, F, N)
AssertFact(USvCh,PD,P, doublelocked, lockStatus, F, T)
AssertFact(USvCh,PD,J, baggageAccompanying, shippingStatus, F, T)
AssertFact(USvCh,PD,J,checkedBaggage, container, F, T)
AssertFactor(USvCh,PD,P, smallContainer, WarrantRequired, I, N)
AssertFact(USvCh,PD,P, suitcase, smallContainer, F, T)
AssertFact(USvCh,PD,P, notCheckedbaggage, smallContainer, F, T)
AssertFact(USvCh,PD,P, onTrain(baggageCompartment),
 shippingStatus, F, N)
AssertFact(USvCh,PD,P, baggageAccompanying, shippingStatus, F, T)
Emphasise Fact(USvCh,PD,P, doublelocked, lockStatus)
AssertCombFactors(USvCh,PD,P, and, {shippingStatus, container,
probableCause, lockStatus}, WarrantLessSearch, I, T, N )
```

Context 3: Probable Cause Evidences and Searching Place

```
AssertFact(USvCh,PD,P, dogSmell, contraband, F, T)
AssertFactor(USvCh,PD,P, automobile, WarrantLessSearch, I, L)
AssertFact(USvCh,PD,P, car, automobile, F, T)
AssertFactor(USvCh,PD,P, placeinAutomobile, WarrantLessSearch, I, PC)
AssertFact(USvCh,PD,P, carTrunk, placeinAutomobile, F, T)
AssertCombFactors(USvCh,PD,P, and, {automobile, placeinAutomobile},
 WarrantLessSearch, I, T, N )
AssertFactor(USvCh,PD,P, searchLocation, WarrantLessSearch, I, L)
AssertFact(USvCh,PD,P, agentOffice, searchLocation, F, T)
AssertCombFactors(USvCh,PD,P, and, {lawfulSeized, searchLocation,
ProbableCause}, WarrantLessSearch, I, T, N )
AssertCombIssues(USvCh PD, P, or, {WarrantLessSearch, WarrantRequired})
```

Context 4: Seizure and Search

```
AssertFactor(USvCh,PD,J, searchLocation, {WarrantLessSearch+WarrantRequired},
 IR, N)
AssertFactor(USvCh,PD,J, searchTime, {WarrantLessSearch+WarrantRequired},
 IR, N)
AssertCombFactors(USvCh,PD,J, and, {searchTime, searchLocation},
 {WarrantLessSearch+WarrantRequired}, IR, T, N )
AssertFactor(USvCh,PD,P, searchTime, WarrantLessSearch, I, N)
AssertCombFactors(USvCh,PD,P, and, {searchTime,
searchLocation}, WarrantLessSearch, I, T, N )
AssertFact(USvCh,PD,P, shortTime, searchTime, F, T)
AssertFactor(USvCh,PD,P, IncidentArrest , WarrantLessSearch, I, N)
AssertFactor(USvCh,PD,P, mobile , WarrantLessSearch, I, N)
AssertFactor(USvCh,PD,P, automobile , mobile, I, N)
AssertFactor(USvCh,PD,P, container , mobile, I, N)
AssertFactor(USvCh,PD,P, privatePlace, WarrantRequired, I, L)
AssertFact(USvCh,PD,P, home, privatePlace, F, H)
AssertFactor(USvCh,PD,J, probableCause, WarrantLessSearch, I, H)
AssertFactor(USvCh,PD,J, lawfulSeized, {WarrantLessSearch+WarrantRequired},
 IR, H)
AssertFactor(USvCh,PD,J, container, {WarrantLessSearch+WarrantRequired},
 IR, N)
AssertCombFactors(USvCh,PD,J, and, {lawfulSeized, probableCause,
```

```
container}, WarrantLessSearch, I, N )
AssertCombFactors(USvCh,PD,P, and, {lawfulSeized, probableCause,
container}, WarrantLessSearch, I, N )
AssertCombFactors(USvCh,PD,J, and, {lawfulSeized, searchLocation},
 {WarrantLessSearch+WarrantRequired}, IR,N )
```

Context 5: Open Locker and Comparison to Automobile Search

```
AssertFact(USvCh,PD,P, footlocker, container, F, T)
AssertFact(USvCh,PD,P, glove compartment, placeinAutomobile, F, PC)
AssertFact(USvCh,PD,P, trunk, placeinAutomobile, PC, PC)
AssertCombFacts(USvCh,PD,P, or, {glove compartment, trunk},
placeinAutomobile, F, PC)
AssertCombFactors(USvCh,PD,P, or, {container, automobile},
WarrantLessSearch, I, N )
AssertFactor(USvCh,PD,J, ILA, WarrantLessSearch, I, L)
AssertCombFactors(USvCh,PD,J, and, {footlocker, ILA, automobile},
 WarrantLessSearch, I, T, N)
AssertIssue(USvCh,PD,J, WarrantRequired)
AssertCombIssues(USvCh,PD,J, +, {WarrantLessSearch, WarrantRequired})
AssertFactor(USvCh,PD,J, searchLocation, {WarrantLessSearch +
WarrantRequired}, IR, N)
AssertFactor(USvCh,PD,P, searchLocation, WarrantLessSearch, I, PC)
AssertFact(USvCh,PD,P, policeOffice, searchLocation, F, PC)
AssertCombFactors(USvCh,PD,P, and, {search location, automobile},
WarrantLessSearch, I, H)
AssertCombFactors(USvCh,PD,P, and, {search location, container},
WarrantLessSearch, I,  Cooper v. California)
AssertFactor(USvCh,PD,P, lock status, WarrantLessSearch, I, PC)
AssertFact(USvCh,PD,P, locked, lock status, F, PC)
AssertCombFactors(USvCh,PD,P, and, {lock status, automobile},
WarrantLessSearch, I, H)
AssertCombFacts(USvCh,PD,P, and, {locked, trunk}, {lock status and automobile},
FR, Cady v. Dombrowski )
AssertCombFacts(USvCh,PD,P, and, {locked, footlocker}, {lock status and
automobile}, FR, T)
AssertCombFacts(USvCh,PD,P, or, {locked and trunk, locked and footlocker},
,FR, T,N )
AssertFactor(USvCh,PD,P, bedroom, privacy, I, L)
```

Context 6: Search Incident to Arrest

```
AssertFactor(USvCh,PD,J, lockStatus, {WarrantLessSearch+
WarrantRequired}, IR, N)
AssertFact(USvCh,PD,J, locked,  lockStatus, F, N)
AssertFactor(USvCh,PD,J, seizedIncidentArrest, {WarrantLessSearch+
WarrantRequired}, IR, N)
AssertFactor(USvCh,PD,J, smallPersonalContainer, WarrantRequired, I, N)
AssertCombFactors(USvCh,PD,J, and, {smallPersonalContainer, arrestted},
{WarrantLessSearch+WarrantRequired}, IR, H)
AssertFact(USvCh,PD,J, smallbox,  smallPersonalContainer, F, H)
AssertFact(USvCh,PD,J, sealedEnvelope,  smallPersonalContainer, F, H)
AssertFact(USvCh,PD,P, closedCigarratePack,  smallContainer, F, Robinson)
AssertCombFactors(USvCh,PD,P, and, {smallPersonalContainer, arrestted},
WarrantLessSearch, I, Robinson)
AssertCombFactors(USvCh,PD,J, and, {lockStatus,container, arrestted},
{WarrantLessSearch+WarrantRequired}, IR, H)
AssertFact(USvCh,PD,P, zipperClosed,  lockStatus, F, Daper)
AssertCombFactors(USvCh,PD,P, and, {lockStatus, container, arrestted},
WarrantLessSearch, I, Daper)
```

```
AssertCombFactors(USvCh,PD,J, and, {smallPersonalContainer, arrestted},
{WarrantLessSearch+WarrantRequired}, IR, H)
AssertFact(USvCh,PD,P, papers,  Container, F, Opperman)
AssertCombFactors(USvCh,PD,P, and, {smallContainer, arrestted},
WarrantLessSearch, I, Opperman )
AssertFactor(USvCh,PD,P, easyWarrant, WarrantLessSearch, I, Cooper v. California,
Texas v. White , Chambers v.Maroney, Edwards)
AssertCombFactors(USvCh,PD,P, and, {easyWarrant, arrestted}, WarrantLessSearch,
I, Cooper v. California,  Texas v. White , Chambers v.Maroney, Edwards)
AssertFact(USvCh,PD,P, jail,  searchLocation, F, Edward)
AssertFact(USvCh,PD,P, stationHouse,  searchLocation, F, Edward)
AssertCombFactors(USvCh,PD,P, or, {easyWarrant, probableCause},
 WarrantLessSearch, I, Cooper v. California, Opperman,  Edwards)
```

Context 7: No Precedent Cases(no exception)

```
AssertFactor(USvCh,PD,J, mobile, {WarrantLessSearch+WarrantRequired}, IR, N)
AssertFactor(USvCh,PD,J, automobile, mobile, F, N)
AssertFactor(USvCh,PD,J, container, mobile, F, N)
AssertCombFactors(USvCh,PD,J, or, {container, automobile}, mobile, F, H)
AssertCombFactors(USvCh,PD,J, and, {probableCause, automobile},
WarrantLessSearch, I, L)
AssertCombFactors(USvCh,PD,P, or, {Container, automobile}, WarrantLessSearch,
 I, Opperman )
AssertFact(USvCh,PD,J, wheels, automobile, F, L)
AssertFact(USvCh,PD,J, engine, automobile, F, L)
AssertCombFacts(USvCh,PD,J, and, {wheels, engine}, automobile F,
PC and L )
AssertCombFactors(USvCh,PD,P, or, {Container, automobile}, mobile, I,
Coolidge v Newhampshire)
AssertCombFactors(USvCh,PD,J, and, {Container, automobile}, WarrantLessSearch, I,
 Coolidge v Newhampshire)
AssertCombFactors(USvCh,PD,J, or, {Container, automobile},{WarrantLessSearch+
WarrantRequired}, IR, N)
AssertFactor(USvCh,PD,P, automobileStatus, WarrantLessSearch, I, N)
AssertFact(USvCh,PD,P, stopped, automobileStatus, F, H)
AssertCombFactors(USvCh,PD,P, and, {automobileStatus, mobile},
WarrantLessSearch, I, H )
AssertCombFactors(USvCh,PD,J, and, {automobileStatus, mobile,
searchLocation}, {WarrantLessSearch+WarrantRequired}, IR, N)
AssertCombFactors(USvCh,PD,P, and, {searchLocation, automobile},
 WarrantLessSearch, I, Maroney and White and Dombrowski)
```

Context 8: WarrantRequired Expectation

```
 AssertCombFactors(USvCh,PD,J, or, {smallPersonalContainer, Container},
  {WarrantLessSearch+WarrantRequired}, IR, N)
AssertCombFactors(USvCh,PD,P, and, {probableCause, Container},
 WarrantLessSearch, I, N)
AssertCombFactors(USvCh,PD,J, or, {{probableCause and arrested},{probableCause and
Container}}, {WarrantLessSearch+WarrantRequired}, IR, N)
CloseDialogue(ChiefJustice,PD,Raymond Randolph,USvCh)
```

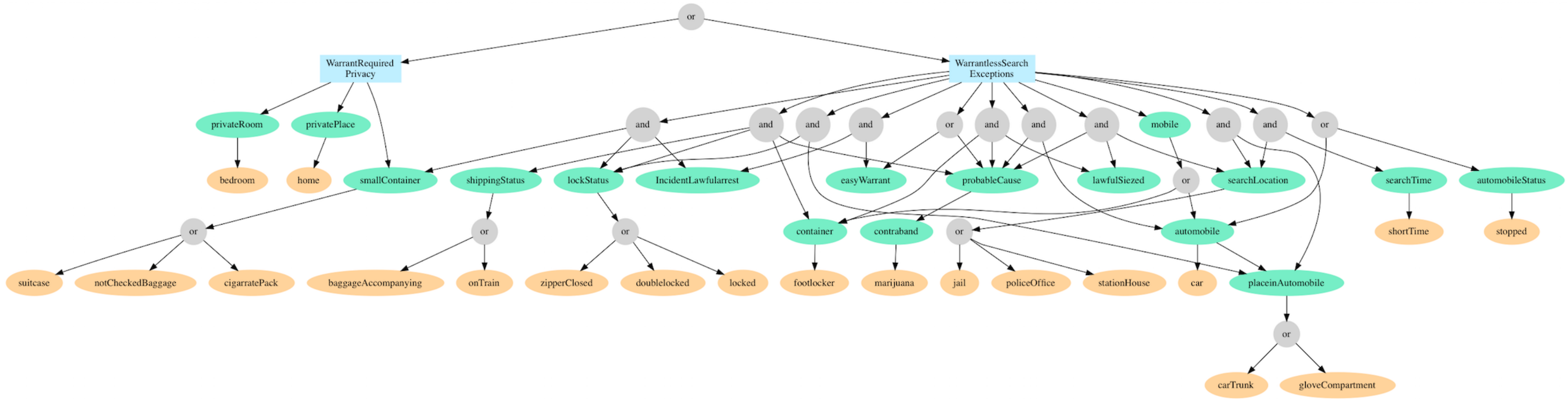


FIGURE B.1: US v. Chadwick Petitioner Dialogue - Petitioner ACT

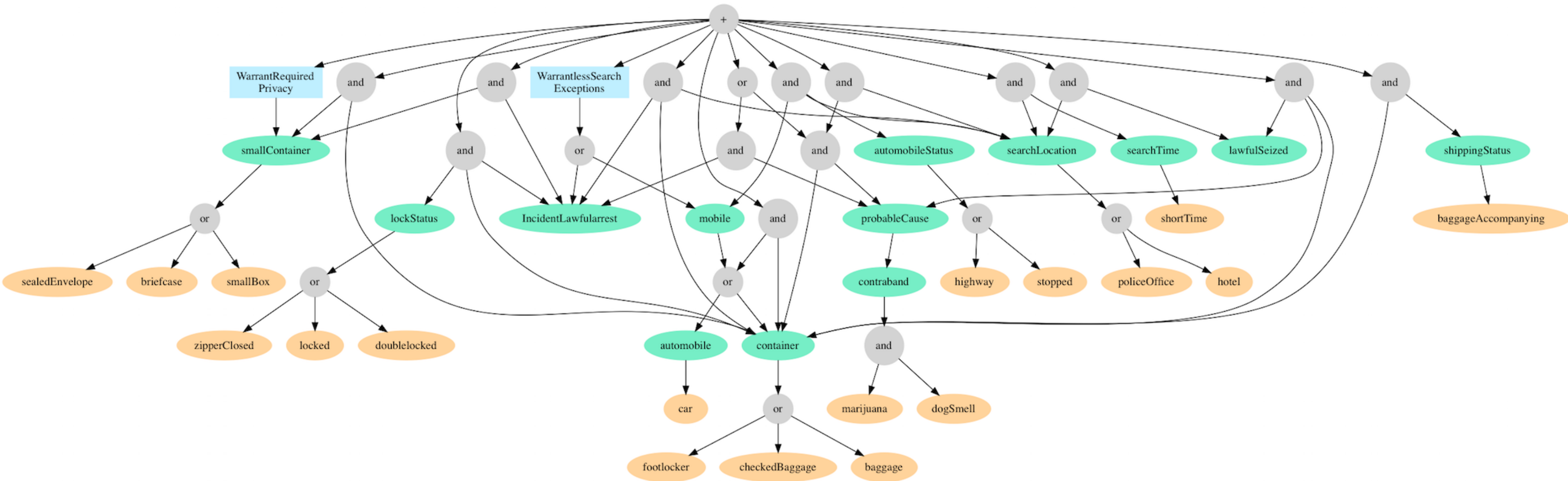


FIGURE B.2: US v. Chadwick Petitioner Dialogue - Justices' ACT

B.2 Respondent Dialogue

Context 1: Rebutting Petitioner Conflict Issues

```
OpenDialogue(ChiefJustice,RD, Martin G. Weinberg,USvCh)
AssertIssue(USvCh, PD, R,  WarrantSearch(WarrantRequired Expectation) )
AssertIssue(USvCh, PD, R,  WarrantlessSearch)
AssertFactor(USvCh,RD,R, Search and Seize, WarrantLessSearch, I, L)
AssertFactor(USvCh,RD,R, automobile Exception, WarrantLessSearch, I, L)
AssertCombFactors(USvCh,RD,R, and, {Search and Seize, automobile
Exception}, WarrantLessSearch, I, L)
AssertFactor(USvCh,RD,R,  privateProperty, WarrantSearch
(WarrantRequired Expectation), I, L)
AssertFactor(USvCh,RD,R,  container, privateProperty , F, N)
AssertFact(USvCh,RD,R, footlocker, container, F, T)
AssertFactor(USvCh,RD,R,  automobile, automobile Exception, F, L)
AssertCombFactors(USvCh,RD,R, and, {container, Search and Seize},
WarrantSearch, I, L)
AssertFact(USvCh,RD,R, briefcase, container, F, H)
AssertFact(USvCh,RD,R, trunk, container, F, H)
Assert CombFact(USvCh,RD,R, or, {briefcase, trunk,footlocker}, container,
 F, H)
AssertFactor(USvCh,RD,R,  inspectionRegulation, automobile Exception,
 F, L)
AssertCombFactors(USvCh,RD,R, and, {Search and Seize, automobile
 Exception}, WarrantLessSearch, I, Opperman and Cady)
AssertCombFactors(USvCh,RD,R, and, {inspectionRegulation, automobile},
automobile Exception,F, Opperman and Cady)
AssertFactor(USvCh,RD,R,  not container, automobile , F, N)
AssertFactor(USvCh,RD,R,  automobile, automobile Exception, F, L)
AssertFactor(USvCh,RD,R,  container, privateProperty , F, N)
AssertFact(USvCh,RD,R, car, automobile, F, PC)
```

Context 2: Expectation of WarrantRequired from Previous Cases

```
AssertIssue(USvCh,PD,J,  WarrantRequired(WarrantRequired Expectation) )
AssertIssue(USvCh,PD,J,  WarrantlessSearch )
AssertFactor(USvCh,RD,J, automobile Exception, WarrantLessSearch, I, L)
AssertFactor(USvCh,RD,J,  automobile, automobile Exception, F, L)
AssertCombIssues(USvCh PD, J, +, {WarrantlessSearch, WarrantRequired})
AssertFactor(USvCh,RD,J,  probableCause ,  WarrantlessSearch+
WarrantRequired, IR, L)
AssertCombFactors(USvCh,RD,J, and, {probableCause, automobile},
 WarrantLessSearch, I, Opperman and Cady)
AssertFact(USvCh,RD,R, luggage, container, F, H)
Assert CombFact(USvCh,RD,R, or, {briefcase, trunk,footlocker,luggage},
container, F, H)
AssertCombFactors(USvCh,RD,J, and, {probableCause, automobile},
 WarrantRequired, I, Opperman and Cady)
AssertCombIssues(USvCh PD, J, +, {WarrantlessSearch, WarrantRequired})
AssertFactor(USvCh,RD,R,  home, privateProperty , F, L(fourth amendment) )
AssertCombFactors(USvCh,RD,R, or, {home, container},
privateProperty,F, Lee and Jackson)
AssertFact(USvCh,RD,R, sealed package(mail,letters), container, F, Lee and
 Jackson)
Assert CombFact(USvCh,RD,R, or, {briefcase, trunk,footlocker,luggage,
sealed package(mail,letters)}, container , F, H)
```

Context 3: Arrested Man with Close Container

```
AssertFactor(USvCh,RD,J,  search incident arrest, WarrantLessSearch, I, L)
AssertFactor(USvCh,RD,J,  container, WarrantLessSearch+warrantRequired,
 IR, L)
AssertFactor(USvCh,RD,J,  lockStatus, {WarrantLessSearch+
WarrantRequired}, IR, N)
AssertFact(USvCh,RD,J, closed, lockStatus, F, N)
AssertFact(USvCh,RD,J, arrestedPerson, search incident arrest, F, L)
AssertFact(USvCh,RD,J, briefcase, container, F, H)
AssertCombFactors(USvCh,RD,J, and, {container, lockStatus,
search incident arrest}, {WarrantRequired+ WarrantLessSearch}, IR, N)
AssertFactor(USvCh,RD,R,  AccessibilityEvidence, search incident arrest, F, H)
AssertFactor(USvCh,RD,R,  immediateControl, AccessibilityEvidence, F, Chimel)
AssertFact(USvCh,RD,R, arrestedPerson, search incident arrest, F, L)
AssertFactor(USvCh,RD,J,  lockStatus, privateProperty, F, N)
AssertFact(USvCh,RD,R, closed, lockStatus, F, N)
AssertCombIssues(USvCh PD, J, +, {WarrantlessSearch, WarrantRequired})
AssertCombFactors(USvCh,RD,R, and, {container, lockStatus,search
incident arrest}, {WarrantLessSearch and WarrentRequired}, IR, N)
AssertFact(USvCh,RD,J, sealedEnvelope, container, F, H)
Assert CombFact(USvCh,RD,J, or, {briefcase, sealedEnvelope}, container, F, H)
AssertFact(USvCh,RD,J, car, automobile, F, PC)
AssertFact(USvCh,RD,J, seizable Material, ProbableCause, F, H)
AssertCombFactors(USvCh,RD,J, and, {automobile, ProbableCause,search
incident arrest},{WarrantLessSearch + WarrentRequired}, IR, N)
AssertCombFactors(USvCh,RD,J, and, {{automobile, ProbableCause,search
 incident arrest}, {container and lockStatus}}, {WarrantLessSearch +
 WarrentRequired}, IR, N)
AssertFactor(USvCh,RD,R,  ProbableCause, {WarrantLessSearch and
WarrentRequired},IR, N)
AssertCombFactors(USvCh,RD,R, and, {automobile, probableCause,
search incident arrest}, automobileException , F, N)
AssertFact(USvCh,RD,R, doublelocked, lockStatus, F, N)
AssertCombFactors(USvCh,RD,R, and, {container, probableCause,
search incident arrest,lockStatus},Warrantrequired , I, N)
AssertFact(USvCh,RD,J, zipperbag, container, F, Daper)
AssertCombFactors(USvCh,RD,J, and, {search incident arrest,container,
lockStatus}, WarrantLessSearch, I, Daper)
AssertFact(USvCh,RD,R, zipperbag, container, F, Daper)
AssertCombFactors(USvCh,RD,R, and, {container, lockStatus}
search incident arrest, F, Daper)
EmphasizeFact(USvCh,RD,R,doublelocked, lockStatus)
AssertFactor(USvCh,RD,R,  containerWeight, privateproperty, F, N)
AssertCombFactors(USvCh,RD,R, and, {container, containerWeight}
privateproperty, F, N)
AssertFact(USvCh,RD,R, heavy, containerWeight, F, N)
AssertCombFactors(USvCh,RD,R, and, {container,lockStatus, containerweight},
privateProperty , F, N)
```

Context 4: Searching Location and Time

```
AssertFactor(USvCh,RD,R,  searchPlace, Warrantrequired, I, N)
AssertFact(USvCh,RD,R, agentOffice, searchPlace, F, T)
AssertCombFactors(USvCh,RD,R, and, {privateproperty,searchPlace},
Warrantrequired , I, N)
AssertFactor(USvCh,PD,J, contraband, probableCause, F, T)
AssertFact(USvCh,RD,J, dogSmell, contraband, F, T)
AssertFactor(USvCh,PD,J, searchPlace, {WarrantLessSearch+
WarrentRequired},IR, T)
AssertCombFactors(USvCh,RD,J, and, {container,probableCause,searchPlace},
WarrantLessSearch+ WarrentRequired, IR, N)
```

```
AssertCombFactors(USvCh,RD,R, and, {container, searchPlace, lockStatus,
 probableCause}, Warrantrequired , I, N)
AssertCombFactors(USvCh,RD,R, and, {automobile,searchPlace,probableCause},
automobileException , F, N)
AssertFact(USvCh,RD,J, agentOffice, searchPlace, F, T)
AssertFact(USvCh,RD,J, depot, searchPlace, F, T)
Assert CombFact(USvCh,RD,J, or, {agentOffice, depot}, searchPlace, F, H)
AssertFact(USvCh,RD,R, notaccessible, AccessibilityEvidence, F, T)
AssertCombFactors(USvCh,RD,R, and, {no accessibilityEvidence,lockStatus,
searchPlace,container},WarrantRequired , I, N)
AssertCombFactors(USvCh,RD,J, and, {accessibilityEvidence, lockStatus,
searchPlace,container},WarantlessSearch +WarrantRequired, I, N)
```

Context 5: Search Time (warrant Time)

```
AssertFact(USvCh,RD,J, ownersLand, searchPlace, F, collidge)
Assert CombFact(USvCh,RD,J, or, {agentOffice, depot, ownersLand},
searchPlace, F, collidge)
AssertCombFactors(USvCh,RD,J, and, {searchPlace,automobile},
WarrantRequired , I, collidge)
AssertCombFactors(USvCh,RD,J, and, {container,automobile},
WarrantLessSearch , I, H)
AssertFactor(USvCh,RD,J,  privateProperty, WarrantRequired,I, L)
AssertFact(USvCh,RD,J, home, privateProperty, F, collidge)
AssertCombFactors(USvCh,RD,J, and, {container,home},
WarrantRequired , I, H)
AssertCombFactors(USvCh,RD,R, and, {container,probableCause},
WarrantRequired , I, L)
AssertCombFactors(USvCh,RD,R, and, {automobile,probableCause},
WarrantLess , I, L)
AssertCombFactors(USvCh,RD,J, and, {container,probableCause,
search and seize}, WarrantlessSearch , I, H)
AssertCombFactors(USvCh,RD,R, and, {container,probableCause,
search and seize}, WarrantRequired , I, H)
AssertFactor(USvCh,RD,J, seizeDuration, {WarrantRequired+
WarantlessSearch},IR, H)
AssertFact(USvCh,RD,J,indefinitely, seizeDuration, F, N)
AssertCombFactors(USvCh,RD,J, and, {container,probableCause, seizeDuration},
{WarrantRequired+WarrantlessSearch}, IR, N)
AssertFactor(USvCh,RD,R,  seizeDuration, WarrantRequired,I, H)
AssertFact(USvCh,RD,R,reasonableTime, seizeDuration, F, Lee)
AssertFact(USvCh,RD,J,suffecientPeriod, seizeDuration, F, N)
Assert CombFact(USvCh,RD,J, or, {suffecientPeriod, not indefinitely},
SeizeDuration, F, H)
AssertFactor(USvCh,RD,J,  riskLoseEvidence, {WarrantRequired+
WarantlessSearch},IR, H)
AssertCombFactors(USvCh,RD,J, and, {riskLoseEvidence,seizeDuration},
WarrantRequired+warantlessSearch , IR, H)
AssertCombFactors(USvCh,RD,R, and, {container,seizeDuration},
WarrantRequired, I, N)
AssertCombFactors(USvCh,RD,R, or, {container,home},
PrivateProperty, F, N)
AssertFactor(USvCh,RD,J,  mobile, WarantlessSearch,I, H)
AssertCombFactors(USvCh,RD,R, and, {container,mobile,searchPlace},
WarantlessSearch, I, N)
AssertCombFactors(USvCh,RD,J, and, {home,searchPlace},
WarantlessRequired, I, N)
AssertCombFactors(USvCh,RD,J, or,{{container and mobile and searchPlace}
{home and searchPlace}}, WarantlessRequired, I, N)
AssertCombFactors(USvCh,RD,J, and, {container,mobile,searchPlace},
```

```
WarantlessSearch, I, N)
AssertFact(USvCh,RD,J,heroin, contraband, F, H)
AssertFact(USvCh,RD,J,bomb, contraband, F, H)
AssertFact(USvCh,RD,J,smallTactical, contraband, F, H)
Assert CombFact(USvCh,RD,J, or, {heroin, bomb,smallTactical}, contraband,
 F, H)
AssertCombFactors(USvCh,RD,J, and, {container,probableCause},
WarantlessSearch+WarrantRequired, IR, N)
AssertFact(USvCh,RD,R,heroin, contraband, F, H)
AssertFact(USvCh,RD,R,bomb, contraband, F, H)
AssertFact(USvCh,RD,R,smallTactical, contraband, F, H)
AssertCombFacts(USvCh,RD,R, or, {heroin, bomb,smallTactical}, contraband,
 F, H)
AssertFactor(USvCh,RD,R,personalContent,privateproperty ,F, H)
AssertFact(USvCh,RD,R,books and Diaries, personalContent, F, H)
AssertCombFactors(USvCh,RD,R, and, {container,personalContent},
WarrantRequired, I, Jackson and Lee and Kartz)
AssertFactor(USvCh,RD,R,not footlocker,automobile ,F, N)
AssertFactor(USvCh,RD,R,EasyWarrant,warrantRequired ,F, N)
AssertFactor(USvCh,RD,R,courtDistance,EasyWarrant ,F, N)
AssertFactor(USvCh,RD,J,EasyWarrant,warrantRequired+warantlessSearch,
IR, N)
AssertFactor(USvCh,RD,J,unavaiableMagistrate,EasyWarrant ,F, N)
AssertCombFactors(USvCh,RD,R, and, {avaiableMagistrate,courtDistance},
EasyWarrant, F, N)
```

Context 6: Not Automobile

```
AssertFactor(USvCh,RD,R, caretakingFunction, WarrantLessSearch, I, PC)
AssertCombFactors(USvCh,RD,R, and, {caretakingFunction, automobile},
WarrantLessSearch, I, Cooper)
AssertFactor(USvCh,RD,J, probableCause, WarrantLessSearch, I, L)
AssertFact(USvCh,RD,J, transportContraband, probableCause, F, PC, cooper)
AssertFact(USvCh,RD,J, car, automobile, F, PC, Cooper)
Assert CombFactor(USvCh,RD,J, and, probableCause and autombile, WarrantLessSearch,
 I, Cooper)
Assert CombFact(USvCh,RD,J, and, {transportContraband, car}, {probableCause
and autombile}, FR, PC, Cooper)
AssertFactor(USvCh,RD,R, container, WarrantRequired, I, PC)
AssertFact(USvCh,RD,R, footlocker, container,F, T)
AssertFactor(USvCh,RD,R, forfeitureStatute, WarrantLessSearch, I, Cooper)
Assert CombFactor(USvCh,RD,R, and, {probableCause, automobile, forfeitureStatute},
 WarrantLessSearch, I, Cooper)
Assert CombFactor(USvCh,RD,J, and, {probableCause, footlocker, forfeitureStatute},
 {WarrantLessSearch+warrantRequired}, IR, Cooper)
AssertFact(USvCh,RD,R, luggage, container, F, H)
Assert CombFact(USvCh,RD,R, or, {luggage, footlocker}, container, F, H, H)
Assert CombFactor(USvCh,RD,R, and, {probableCause, container, forfeitureStatute},
 WarrantRequired, I, Cooper)
```

Context 7: Search by Police v Search by Court

```
AssertFact(USvCh,RD,J, courtSearch, searchPlace, F, H)
Assert CombFact(USvCh,RD,J, or, {agentOffice, depot, ownersLand,courtSearch},
searchPlace, F, H)
AssertCombFactors(USvCh,RD,J, and, {searchPlace,container,probableCause},
WarantlessSearch , I, N)
AssertFact(USvCh,RD,R, courtSearch, searchPlace, F, H)
Assert CombFact(USvCh,RD,R, or, {agentOffice, depot, not courtSearch},
searchPlace, F, H)
```

```
AssertCombFactors(USvCh,RD,R, and, {legalsearchPlace,container,
probableCause},WarantlessSearch , I, N)
AssertCombFactors(USvCh,RD,J, and, {searchPlace,container,probableCause},
forfeitureStatute , I, N)
AssertCombFactors(USvCh,RD,R, and, {searchPlace,container,probableCause},
Warantrequired , I, N)
CloseDialogue(ChiefJustice,RD, Martin G. Weinberg ,USvCh)
```

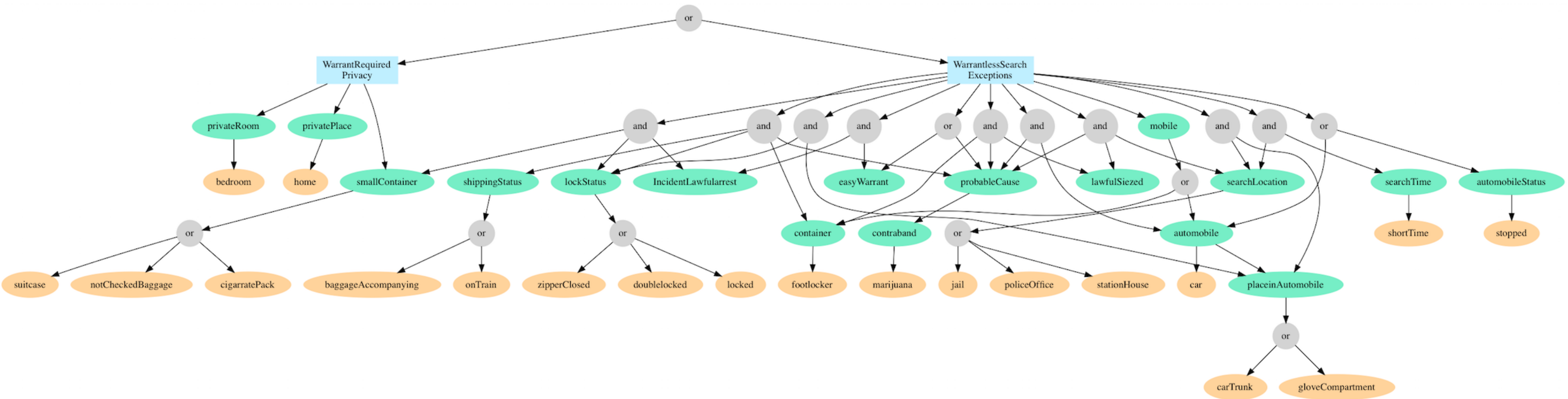


FIGURE B.3: US v. Chadwick Respondent Dialogue - Respondent ACT

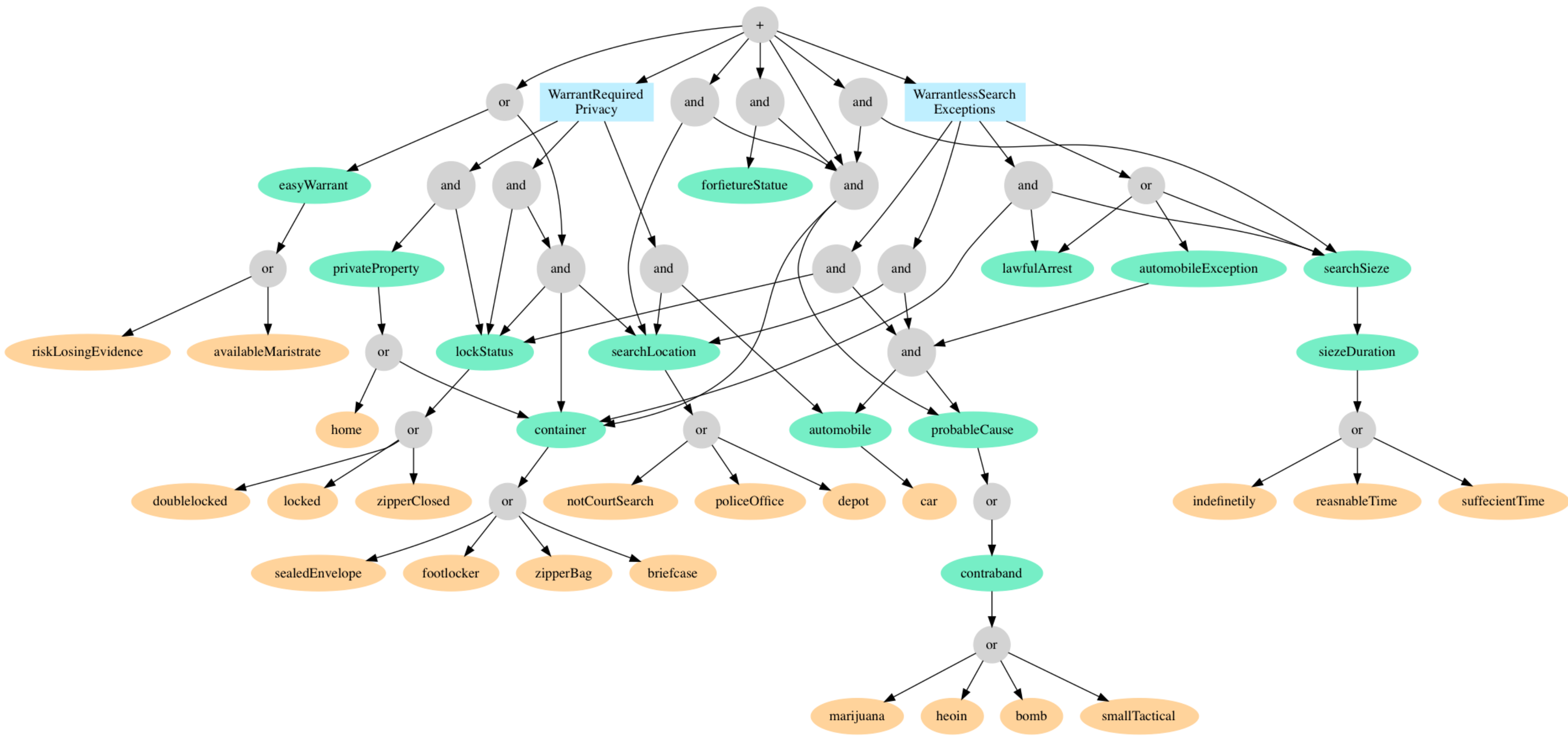


FIGURE B.4: US v. Chadwick Respondent Dialogue - Justices' ACT

B.3 Petitioner Rebuttal Dialogue

```
OpenDialogue(ChiefJustice,PRD,Raymond Randolph,USvCh)
AssertCombFactors(USvCh,PRD,P, and, {container, arrested},
searchIncidenttoArrest, F, Hanz and Sariano and Schelis and Zaicek)
AssertCombFactors(USvCh,PRD,J, and, {container, arrested,searchLocation},
 {WarrantLessSearch+WarrantRequired}, IR,N)
AssertFact(USvCh,PRD,J, breifcase, container, F, H)
AssertFact(USvCh,PRD,J, Hotel, searchLocation, F, H)
AssertCombFactors(USvCh,PRD,J, and, {container, probableCause,
searchLocation}, {WarrantLessSearch+WarrantRequired}, IR,N)
AssertFact(USvCh,PD,J, ontheWay, automobileStatus, F, L)
AssertCombFactors(USvCh,PRD,J, and, {ProbableCause, automobile,
automobileStatus}, WarrantLessSearch, I,L)
AssertFact(USvCh,PRD,J, baggage, container, F, USvJohnston)
AssertFact(USvCh,PRD,J, baggageCar, automobile, F, USvJohnston)
AssertFact(USvCh,PRD,J, dogsmelledContraband, contaband, F, USvJohnston)
AssertCombFactors(USvCh,PRD,P, and, {container, ProbableCause,lawfulSeize},
WarrantLessSearch, I, USvJohnston)
CloseDialogue(ChiefJustice,PRD,Raymond Randolph, USvCh)
CloseCase(ChiefJustice, USvCh)
```

Appendix C

Case Study: California v. Carney Dialogue Moves and ACTs

This appendix provides the dialogue moves for the marked-up Oral Hearing transcript of the case study *California v. Carney*. Three dialogues are analysed in this transcript: Petitioner dialogue (Section C.1); Respondent dialogue (Section C.2) and Petitioner rebuttal (Section C.3). Each dialogue is divided into a number of contexts, and an ACT for each player is produced after the petitioner and respondent dialogues, and updated after the petitioner rebuttal dialogue.

C.1 Petitioner Dialogue

Context 1: Introducing the Case

```
OpenCase (ChiefJustic, CvC)
OpenDialogue(ChiefJustic,PD, Louis Hanoian,CvC)
AssertIssue (CvC,PD,P, Exigency)
AssertIssue (CvC,PD,P, Privacy)
AssertCombIssues (CvC PD, P, or, {Exigency, Privacy})
AssertFactor (CvC,PD,P, mobility, Exigency, I, Carroll and Marony and Ross)
AssertFactor (CvC,PD,P, Parking, Exigency, I, N)
AssertFact (CvC,PD,P, downtown, parking, F, T)
AssertFactor (CvC,PD,P, origin, Exigency, I, N)
AssertFact (CvC,PD,P, agenrOfficerObserve, origin, F, T)
AssertFact (CvC,PD,P, agenrOfficerInform, origin, F, T)
AssertCombFacts (CvC,PD,P, and, {agenrOfficerInform, agenrOfficerObserve },
 origin, F, T )
AssertFactor (CvC,PD,P, contraband, Exigency, I, N)
AssertFact (CvC,PD,P, marijuana, contraband, F, T)
```

Context 2: Parking

```
AssertIssue (CvC,PD,J,Exigency)
AssertIssue (CvC,PD,J,Privacy)
AssertFactor (CvC,PD,J,ParkingTime, Exigency+Privacy, IR, N)
AssertFactor (CvC,PD,P, ParkingTime, Exigency, I, N)
AssertFact (CvC,PD,P, unknown, ParkingTime, F, T)
AssertFactor (CvC,PD,J,ParkingType, Exigency+Privacy, IR, N)
AssertFactor (CvC,PD,P, ParkingType, Exigency, I, N)
AssertFact (CvC,PD,P, public, ParkingType, F, T)
AssertFact (CvC,PD,P, privatelyowned, ParkingType, F, T)
```

```
AssertCombFacts (CvC,PD,P, and ,{Public, privatelyowned, downtown },
ParkingType, F, T )
AssertFact (CvC,PD,J,not trailerPark, ParkingType, F, T)
AssertFact (CvC,PD,P, not trailerPark, ParkingType, F, T)
AssertCombFacts (CvC,PD,P, and ,{Public, privatelyowned, downtown,
not trailerPark}, ParkingType, F, T )
```

Context 3: Vehicle Configuration and Circumstances

```
AssertFactor (CvC,PD,J,mobileConfiguration, Exigency+Privacy,IR, N)
AssertFact (CvC,PD,J,tractorDrawn, mobileConfiguration, F, H)
AssertFactor (CvC,PD,P, mobileConfiguration, mobility, F, N)
AssertFact (CvC,PD,P, notTractorDrawn, mobileConfiguration, F, H)
AssertFactor (CvC,PD,P, integral vehicle, mobileConfiguration, F, T )
AssertFact (CvC,PD,P, wheels, integral vehicle, F,T)
AssertFact (CvC,PD,P, engine, integral vehicle,  F,T)
AssertFact (CvC,PD,P, backPortion, integral vehicle, F,T)
Assert CombFacts (CvC,PD,P, and, {wheels, engine,backPortion}, integral
vehicle, F,T)
AssertFactor (CvC,PD,J,homeFacilities, Privacy, I, H)
AssertCombFactors  (CvC,PD,J,and, {homeFacilities,mobileConfiguration},
Exigency+Privacy,IR, H)
AssertFact (CvC,PD,J,waterElect, homeFacilities, F, H )
AssertFactor(CvC, PD, P, homeFacilities, Privacy, I,H)
AssertFact (CvC,PD,P, noWaterElect, homeFacilities, F, H)
AssertCombIssues (CvC PD, P, or, {Exigency, Privacy})
AssertFactor (CvC,PD,J,mobileConfiguration, Exigency, I, H)
AssertFact (CvC,PD,J,wheels, mobileConfiguration, F, L)
Emphasise Fact (CvC,PD,J,wheels, mobileConfiguration )
Emphasise Factor (CvC,PD,P, mobility, Exigency)
Emphasise Fact(CvC, PD, P, wheels, mobileConfiguration)
AssertFact (CvC,PD,P, self-Propelled, mobileConfiguration, F,H)
AssertFact (CvC,PD,P, self-locomotion, mobileConfiguration, F,H)
Assert Comb Fact (CvC,PD,P, or, {self-locomotion, self-Propelled}, F, H)
```

Context 4: License Type

```
AssertFactor (CvC,PD,J,VehicleRegisteration, Exigency, IR, N)
AssertFactor (CvC,PD,P, VehicleRegisteration, Exigency, IR, N)
AssertFact (CvC,PD,P, SpecialLicence, VehicleRegisteration, F, T)
AssertCombFactors  (CvC,PD,P, and, {VehicleRegisteration,mobile},
Exigency,I, N)
AssertFact (CvC,PD,J,TruckLicence, VehicleRegisteration, F, H)
AssertCombFacts (CvC,PD,J,or ,{TruckLicence, VehicleLicence},
VehicleRegisteration, F, T )
AssertCombFacts (CvC,PD,J,or ,{SpecialLicence, VehicleLicence},
VehicleRegisteration, F, T )
```

Context 5: Compare to Solicitor opinion

```
AssertFact (CvC,PD,J,mobilehome, Privacy, I, T)
AssertCombFacts (CvC,PD,P, or ,{vehicle, mobileHome}, automobile, F, T )
AssertFact (CvC,PD,J,mobilehomePark, ParkingType, F, H)
AssertCombFacts (CvC,PD,J,or ,{trailerPark, mobilehomePark}, ParkingType, F,T)
AssertFactor (CvC,PD,J,homeFacilities, Privacy, I, H)
AssertFact (CvC,PD,J, WaterElect, homeFacilities,F, H)
AssertCombFactors  (CvC,PD,J,and, {homeFacilities,mobile,
MobilehomeparkingType}, Exigency+Privacy,IR, H)
AssertFact (CvC,PD,J,Public, ParkingType, F, H)
AssertCombFacts (CvC,PD,J,or ,{trailerPark, mobilehomePark,PublicPark},
ParkingType, F, T )
```

```
AssertFactor (CvC,PD,P, capabletoMove, Exigency, I, L)
AssertCombFactors  (CvC,PD,P, and, {capabletoMove,automobile} ,Exigency,I, L)
Assert CombFacts   (CvC,PD,P, and, {wheels, engine},automobile,F,T)
```

Context 6: Compare to Self Propelled (trailer-tractor)

```
AssertFactor (CvC,PD,J,ResidentialUse, Privacy, I, H)
AssertFactor (CvC,PD,J,PeopleLivingin, ResidentialUse,F, H)
AssertFactor (CvC,PD,J,ResidentialUse, Privacy, I, H)
AssertFact (CvC,PD,J,trailer(wheels), mobileConfiguration ,F, H)
AssertFact (CvC,PD,J,trailer(wheels), PeopleLivein ,F, H)
AssertFact (CvC,PD,J,home, PeopleLivein ,F, H)
AssertCombFactors  (CvC,PD,J,or, {trailer,home} ,PeopleLivein,F, H)
AssertFact (CvC,PD,P, trailer(wheels), mobileConfiguration ,F, H)
AssertCombFactors  (CvC,PD,P, and, {mobileConfiguration,parkingType},
Privacy,I, H)
AssertCombFactors  (CvC,PD,J,and, {automobile,parkingType,homeFacilities},
Privacy+Exigency,IR, H)
AssertCombIssues (CvC PD, P, or, {Exigency, Privacy})
AssertCombIssues (CvC PD, J, +, {Exigency, Privacy})
AssertCombFactors  (CvC,PD,J,and, {automobile,publicparkingType},
Exigency,I, H)
AssertCombFactors  (CvC,PD,J,and, {automobile,not publicparkingType},
Privacy,I, H)
AssertCombFactors  (CvC,PD,P, and, {automobile,publicparkingType},
Exigency,I, H)
AssertCombFactors  (CvC,PD,P, and, {automobile,not publicparkingType},
Privacy,I, H)
AssertCombFactors (CvC,PD,J,and, {automobile,publicparkingType},
Exigency,I, H)
AssertFact (CvC,PD,J,shortTime, ParkingTime ,F, H)
AssertCombFactors (CvC,PD,J,and, {ParkingTime,automobile,homeFacilities,
ProbableCause}, Exigency,I, H)
AssertFact (CvC,PD,P, longTime, ParkingTime ,F, H)
AssertCombFactors  (CvC,PD,P, and, {ParkingTime,automobile,ParkingType,
ProbableCause},Exigency,I, H)
AssertFact (CvC,PD,P, not individualExpectation, probableCause ,F, H)
AssertCombFactors  (CvC,PD,P, and, {mobileConfiguration,ParkingType},
Exigency,I, H)
EmphasiseFact (CvC,PD,J,wheels)
Assert  CombFacts   (CvC,PD,P, or, {TrailernoTractor,suitcase},MobileObject,F, H)
AssertCombFactors (CvC,PD,J,and, {trailer,trailerParkingType},
Exigency+Privacy,IR, H)
AssertCombFactors  (CvC,PD,P, and, {trailer,trailerParkingType}, Exigency,I, H)
AssertFact (CvC,PD,J,SelfPropelled,mobile ,F, H)
AssertCombFactors (CvC,PD,J,and, {SelfPropelled,homeFacilities},
Exigency+Privacy, IR, H)
AssertCombFactors  (CvC,PD,P, and, {SelfPropelled, homeFacilities, capabletoMove},
Exigency,I, H)
```

Context 7: Compare to Tent

```
AssertFactor (CvC,PD,J,mobility, {Exigency+privacy}, I, L)
AssertFact (CvC,PD,J,camperTent, mobility, F, H )
AssertFact (CvC,PD,J,motor home, mobility, F, H )
Assert CombFacts  (CvC,PD,P, and, {camperTent, motor home},mobile, H )
AssertFact (CvC,PD,P, mobileHome, mobile, F, T )
AssertFact (CvC,PD,P, notTent, mobile, F, H )
Assert CombFacts  (CvC,PD,P, and, {notTent, mobileHome},mobile, H )
AssertFactor (CvC,PD,J,movable, mobile, IR, H )
```

```
AssertFact (CvC,PD,J, camperTent, movable, F, H )
AssertFactor ( CvC, PD, P, movable, Exigency, I, H)
AssertFact ( CvC, PD, P, tent, movable, F, H)\\
AssertFactor ( CvC, PD, P, movable, Privacy, I, Chadwick)
AssertFact ( CvC, PD, P, suitcase, movable, F, Chadwick)
Assert CombFacts  ( CvC, PD, P, or, {tent, suitcase}, movable,F, H)
AssertFact ( CvC, PD, P, cartrunk, mobileConfiguration, F, PC)
AssertFactor ( CvC, PD, P, probableCause, Exigency, I, H)
AssertCombFactors ( CvC, PD, P, and, {mobile and movable,probableCause},
Exigency, I, PC)
```

Context 8: Bright Line for the Test

```
AssertFact (CvC,PD,P, publicSafety, probableCause, F, L)
AssertFactor (CvC,PD,P, mobility, Exigency, I,  carroll)
AssertCombFactors (CvC,PD,P, and, {mobile,capabletomove,residentialUse},
Exigency,I, N)
AssertCombFactors (CvC,PD,P, and, {mobile,probableCause,no mobileConfiguration,
no peopleExpectation}, Exigency,I, Carroll,Chambers,Ross)
```

Context 9: Compare to House-Boat

```
AssertFactor (CvC,PD,J,  mobile, Exigency+Privacy, IR,N)
AssertFact (CvC,PD,J, houseboat, mobile, F, H)
AssertFact (CvC,PD,P, houseboat, mobile, F, L)
Assert CombFacts (CvC,PD,J,or, {houseboat,automobile}, mobile,F, N)
AssertFactor (CvC,PD,P,  vessel,mobile, F, carroll)
AssertFact (CvC,PD,P, houseboat, vessel, F, L)
AssertFact (CvC,PD,P, ship, vessel, F, L)
AssertFact (CvC,PD,J, besidehouse, parkingLocation, F, H)
AssertCombFactors (CvC,PD,J, and, {houseboat,parked,besidehouse,homeFacilities},
Exigency+Privacy,IR, H)
AssertFact (CvC,PD,P, besidehouse, parkingLocation, F, H)
AssertCombFactors (CvC,PD,P, and, {houseboat,parked,besidehouse,homeFacilities},
Exigency,I, H)
AssertFact (CvC,PD,P, driveway, parkingLocation, F, H)
AssertCombFactors (CvC,PD,P, and, {automobile,parked,driveway,homeFacilities},
Exigency,I, H)
AssertFact (CvC,PD,P, glassHouse, residentialUse, Privacy, H)
AssertCombFactors (CvC,PD,P, and, {residentialUse,mobile,ProbableCause},
Exigency,I, H)
AssertFactor (CvC,PD,P,  warrantrequirement, Privacy, I,N)
AssertFact (CvC,PD,P, unauthoirzedWarrant, warrantrequirement, Privacy, Coolidge)
AssertFact (CvC,PD,P, priorKnoweldge, warrantrequirement, Privacy, Coolidge)
CloseDialogue(ChiefJustic,PD, Louis Hanoian,CvC)
```

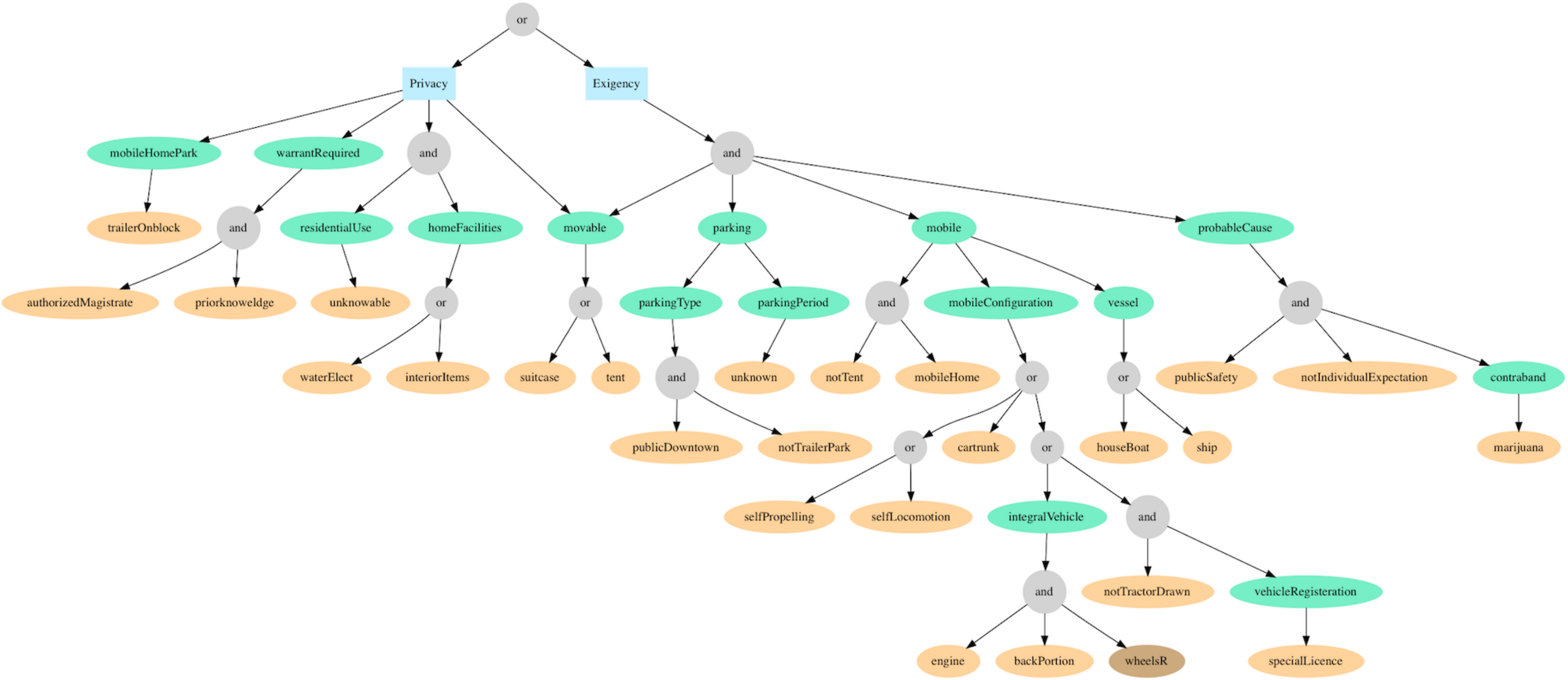


FIGURE C.1: California v. Carney Petitioner Dialogue - Petitioner ACT

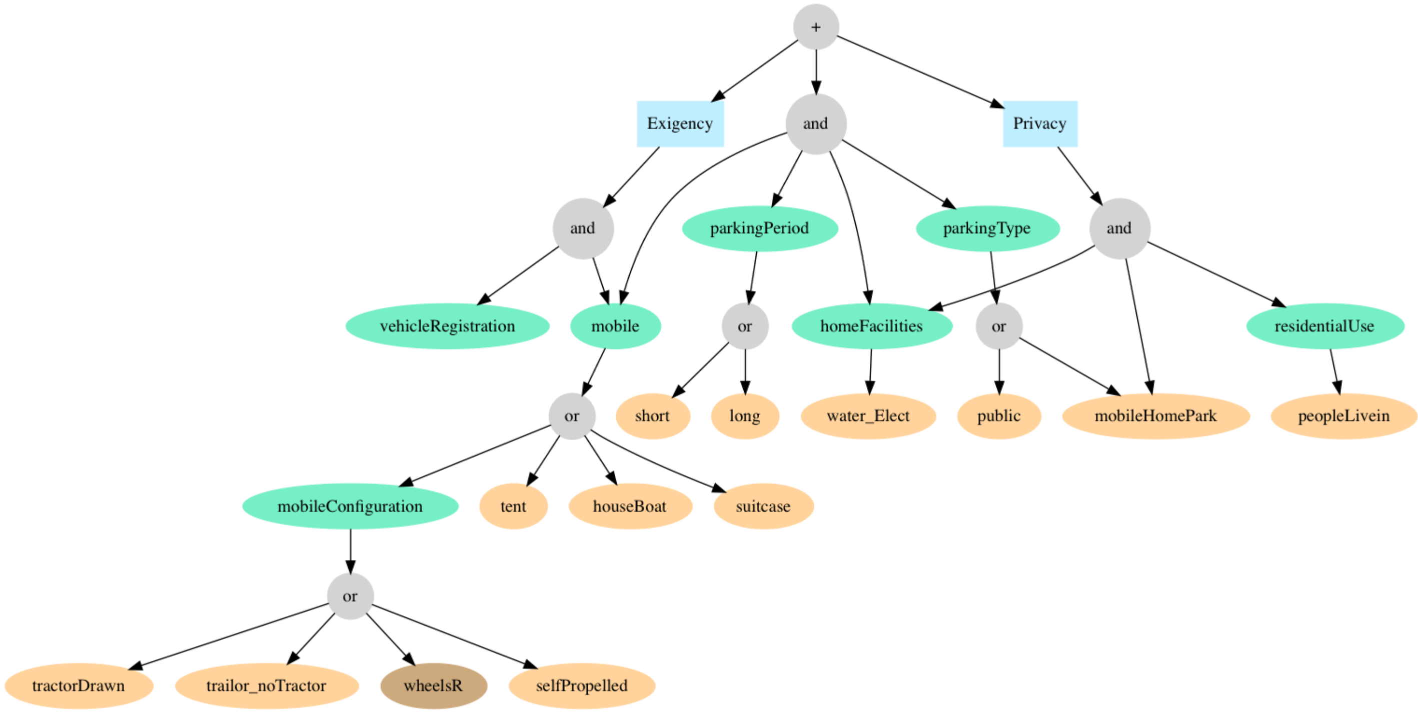


FIGURE C.2: California v. Carney Petitioner Dialogue - Justices' ACT

C.2 Respondent Dialogue

Context 1: Privacy facts-rebut petitioner

```
OpenDialogue(ChiefJustic,RD, Thomas Homann,CvC)
AssertIssue (CvC, RD, R, Privacy)
AssertIssue (CvC, RD, R Exigency)
AssertFactor (CvC, RD, R, residentialArea, Privacy, I, T)
AssertFact (CvC, RD, R, livingCompartment, residentialArea, F, T)
AssertFactor (CvC, RD, R, parked, Privacy, I, T)
AssertFactor (CvC, RD, R, parkingTime, Privacy, I, T)
AssertFact (CvC, RD, R, mobileHome, Exigency, I, T)
AssertFact (CvC, RD, R, extended, parkingTime, F, T)
AssertCombFactors (CvC, RD, R, and, {parked, parkingTime,mobileHome },
Privacy, I, T )
AssertFactor (CvC, RD, R, parked, Privacy, I, T)
AssertIssue (CvC, RD, R, Privacy)
AssertIssue (CvC, RD, R Exigency)
AssertCombIssues (CvC RD, J, +, {Exigency, Privacy})
AssertFactor (CvC, RD, J, parkingLocation,Exigency, I, T)
AssertFact (CvC, RD, J, mobileHome,Exigency, I, T)
AssertFact (CvC, RD, J, Public, parkingLocation, F, T)
AssertFactor (CvC, RD, J, parkingTime, Exigency, I, T)
AssertFact (CvC, RD, J, shortTime, parkingTime, F, T)
AssertCombFactors (CvC, RD, J, and, {parkingLocation, parkingTime },
Privacy+Exigency, I, T )
AssertFactor (CvC, RD, R, visibility, Privacy, I, T)
AssertFact (CvC, RD, R, curtainsDown, visibility, F, T)
AssertFactor (CvC, RD, R, homeproperities, Privacy, I, T)
AssertFact (CvC, RD, R, furniture, homeproperities, F, T)
AssertFact (CvC, RD, R, table, homeproperities, F, T)
AssertFact (CvC, RD, R, refrigerator, homeproperities, F, T)
AssertFact (CvC, RD, R, kitchen, homeproperities, F, T)
Assert CombFacts  (CvC, RD, R, or, {furniture, refrigerator,kitchen },
homeproperities, F, T )
AssertCombFactors (CvC, RD, R, and, {homeproperities,parked, noVisbility,
mobileHome}, Privacy, I, T )
AssertFactor (CvC, RD, R, easyWarrant, Privacy, I, T)
AssertFact (CvC, RD, R, availableMagistrate, easyWarrant, F, T)
AssertFact (CvC, RD, R, closetoCourt, easyWarrant, F, T)
Assert CombFacts  (CvC, RD, R, or, {availableMagistrate, closetoCourt },
EasyWarrant, F, T )
AssertFactor (CvC, RD, R, probableCause, Exigency, I, T)
AssertFactor (CvC, RD, R, mobile, Exigency, I, T)
AssertFact (CvC, RD, R, vehicle/automobile, mobile, I, T)
AssertFactor (CvC, RD, R, mobilestatus, Exigency, I, T)
AssertFact (CvC, RD, R, highway, mobileStatus, F, {Carroll,Ross})
AssertCombFactors (CvC, RD, R, and, {mobile,probableCause,mobileStatus },
Exigency, I, {Carroll,Ross} )
AssertCombFactors (CvC, RD, R, and, {mobileHome,probableCause,parked },
Privacy, I,T )
AssertCombIssues (CvC RD, R, and, {Exigency, Privacy})
```

Context 2: Mobile vs inoperable mobile

```
AssertFactor (CvC, RD, J, CapableToMove, Exigency, I, PC)
AssertFactor (CvC, RD, R, CapableToMove, Exigency, I, PC)
AssertFactor (CvC, RD, R, inOperable, Privacy, I, H)
AssertFact (CvC, RD, R, no driver, inOperable, F, T)
AssertFact (CvC, RD, R, curtainsDown, inOperable, F, T)
```

```
Assert CombFacts  (CvC, RD, R, and, {no driver, closeCurtains }, inOperable, F, T)
AssertFact (CvC, RD, J, CloseCurtains, CapableToMove, F, T)
AssertCombFactors (CvC, RD, R, and, {parked,inOperable}, Privacy, I,T )
AssertFact (CvC, RD, R, curtainsDown (FrontWindshield), inOperable, F, T)
AssertFactor (CvC, RD, R, probableCause, Exigency, I, T)
AssertFact (CvC, RD, R, contraband(marijuana), ProbableCause, F, T)
AssertFactor (CvC, RD, R, accommodationSpace, Privacy, I, N)
AssertFact (CvC, RD, R, livingCompartment, AccommodationSpace, F, T)
AssertFact (CvC, RD, R, automobile, Mobile, F, L)
AssertFactor (CvC, RD, R, residentialPlace, Privacy, I, L)
AssertFact (CvC, RD, R, tempHome, residentialPlace, F, L)
AssertFact (CvC, RD, R, personalHome, residentialPlace, F, L)
Assert CombFacts  (CvC, RD, R, or, {tempHome, personalHome }, residentialPlace, F, T)
AssertFact (CvC, RD, R, tent, residentialPlace, F, H)
AssertFact (CvC, RD, R, motelRoom, residentialPlace, F, H)
Assert CombFacts  (CvC, RD, R, or, {tempHome, personalHome,tent,motelRoom},
 residentialPlace, F, T )
AssertFactor (CvC, RD, J, Mobility, Exigency, I, L)
AssertFact (CvC, RD, J, vehicle, Mobility, F, {PC,L})
AssertFact (CvC, RD, R, crashed, inOperable, F, Cady v Dombrowski)
AssertCombFactors (CvC, RD, R, and, {mobile,inOperable}, Privacy, I,
Cady v Dombrowski )
AssertCombFactors (CvC, RD, R, and, {mobile,ProbableCause, Highway},
 Exigency, I,L)
AssertFactor (CvC, RD, R, Container, Privacy, I, Chadwick)
AssertFact (CvC, RD, R, trunk, Container, F, Chadwick)
```

Context 3: Automobile or Chadwick Trunk

```
AssertFact (CvC, RD, J, mobileHome, mobile, F, T)
Assert CombFacts  (CvC, RD, J, or, {automobile, mobileHome},mobile, F, L )
Assert CombFacts  (CvC, RD, J, or, {trunk, mobileHome},container, F, Chadwick)
AssertCombFactors (CvC, RD, J, or, {container, mobile},{Exigency+Privacy},
IR, Chadwick )
AssertFactor (CvC, RD, R, dresserDrawers, Container, F, T)
AssertFact (CvC, RD, R, cabinet, Container, F, T)
AssertFact (CvC, RD, R, suitcase, Container, F, T)
AssertFact (CvC, RD, R, storageCompartment, Container, F, T)
Assert CombFacts  (CvC, RD, J, or, {trunk, dresserDrawers,cabinet,storage
Compartment, suitcase},Container, I, H )
AssertFact (CvC, RD, R, container, mobile, F, Chadwick)
AssertCombFactors (CvC, RD, R, or, {container, automobile},Privacy, I,Chadwick )
AssertCombFactors (CvC, RD, J, and, {mobile, not capableMove},{Privacy+Exigency},
IR,Chadwick )
```

Context 4: Compare MobileHome to a Big Stretch Cadillac

```
AssertFact (CvC, RD, J, BigStretchCadilac, automobile, F, H)
AssertCombFactors (CvC, RD, J, and, {mobile, not capableMove,parked},{Privacy+
Exigency},IR,H )
AssertFact (CvC, RD, J, home, Privacy, I, L)
AssertFact (CvC, RD, R, BigStretchCadilac, automobile, F, H)
AssertFactor (CvC, RD, J, homeProperties, Privacy, F, T)
AssertCombFactors (CvC, RD, R, and, {mobile, homeProperties,parked,not capableMove},
Privacy, I,H )
AssertFact (CvC, RD, R, bed, homeProperties, F, H)
AssertCombFactors (CvC, RD, R, and, {residentialUse, mobile, homeProperties,parked,
not capableMove},Privacy, I,H )
```

Context 5: Motor Home attributes

```
AssertFact (CvC, RD, R, trailer, accommodationSpace, F, H)
AssertFact (CvC, RD, R, self contained unit, accommodationSpace, F, H)
Assert CombFacts  (CvC, RD, R, or, {trailer, self contained unit},accommodationSpace,
F, H )
AssertFact (CvC, RD, R, cab, accommodationSpace, F, H)
Assert CombFacts  (CvC, RD, R, and, {self contained unit,cab},accommodationSpace,
F, H )
AssertCombFactors (CvC, RD, R, and, { homeProperties,accommodationSpace},
Privacy, I,H )
AssertFactor (CvC, RD, J, visibility, Privacy, I, PC)
AssertFactor (CvC, RD, J, cannotbeSeen, Visibility, F, H)
AssertFactor (CvC, RD, R, visibility, Privacy, I, PC)
AssertCombFactors (CvC, RD, R, and, { residentialUse,mobile},Privacy, I,H )
AssertFact (CvC, RD, J, sleepingBag, homeProperties, F, H)
AssertCombFactors (CvC, RD, J, and, { residentialUse,mobile,homeProperties},
{Privacy+Exigency}, IR,H )
AssertFact (CvC, RD, R, not sleepingBag,  homeProperties, F, H)
AssertFact (CvC, RD, R, personalItems, Containers, F, H)
AssertFact (CvC, RD, R, kitchen, homeProperties, F, H)
AssertFact (CvC, RD, R, bed, homeProperties, F, H)
AssertFact (CvC, RD, R, closeCurtains, not visible, F, H)
Assert CombFacts  (CvC, RD, R, or, {kitchen bed},homeProperties, F, H )
AssertCombFactors (CvC, RD, R, and, { container, mobile, not visible,
homeProperties}, Privacy, I,H )
```

Context 6: Compare to Covered Wagon

```
AssertFact (CvC, RD, J, wagon, mobile, F, H)
AssertFact (CvC, RD, R, wagon, mobile, F, H)
AssertFact (CvC, RD, R, wagon, residentialUse, F, H)
AssertFact (CvC, RD, R, wagon, container, F, H)
AssertCombFactors (CvC, RD, R, and, {residentialUse, mobile, container},
Privacy, I H )
```

Context 7: Privacy of vehicle v. vehicle function

```
AssertFactor (CvC, RD, R, Visibility, Privacy, I, PC)
AssertFactor (CvC, RD, R, Mobile(transportation), Exigency, I, L)
AssertFactor (CvC, RD, R, Mobile(storageCompartment)), Privacy, I, H)
```

Context 8: Moving van v. parked van

```
AssertFact (CvC, RD, J, van, mobile, F, H)
AssertFact (CvC, RD, R, van, mobile, F, H)
Assert CombFacts  (CvC, RD, R, or, {van, mobileHome},mobile, F, H )
AssertFactor (CvC, RD, J, mobileStatus, Exigency, I H)
AssertFact (CvC, RD,J, highway, mobileStatus, F, PC)
AssertCombFactors (CvC, RD, J, and, {mobile,mobileStatus},{Exigency+Privacy},
IR, H )
AssertFactor (CvC, RD, R, mobileStatus, Exigency, I H)
AssertFact (CvC, RD, R, highway, mobileStatus, F, PC)
AssertCombFactors (CvC, RD, R, and, {residentialUse, mobile,mobileStatus},
Exigency, I, H )
AssertFactor (CvC, RD, R, riskLosingEvidence, not EasyWarrant, F, H)
AssertCombFactors (CvC, RD, R, and, {not easyWarrant, mobile,mobileStatus,
probableCause},Exigency, I, H )
AssertFact (CvC, RD, R, parked, not urgentMobileStatus, F, PC)
Assert CombFacts  (CvC, RD, R, or, {parked, highway},mobileStatus, F, H )
AssertCombFactors (CvC, RD, R, and, {easyWarrant, mobile,no urgentMobileStatus,
ProbableCause},Privacy, I, H )
AssertCombFactors (CvC, RD, J, and, {mobile,no urgentmobileStatus, notcapableMove},
```

```
{Privacy+Exigency}, IR, H )
```

Context 9: Car and Mobile Home Parked in the Same Parking Lot

```
Assert CombFacts  (CvC, RD, J, or, {mobileHome, car},automobile, F, H )
AssertCombFactors (CvC, RD, J, and, {publicParking,automobile,probableCause},
 {Exigency+Privacy},IR, H )
AssertCombFactors (CvC, RD, R, and, {mobile, container, publicParking,
 ProbableCause},Privacy, F, H )
AssertCombFactors (CvC, RD, J, and, {publicParking,automobile,probableCause,
 container, capabletoMove }, {Exigency+Privacy}, IR, H )
AssertCombFactors (CvC, RD, R, and, {mobile, container, urgentMobileStatus,
ProbableCause },Exigency, F, H )
```

Context 10: Tractor Pulls Trailer

```
AssertFact (CvC, RD, J, TractorpullTailer, Mobile, F, H)
Assert CombFacts  (CvC, RD, J, and, {mobile, mobileStatus}, Exigency+Privacy,
IR, H )
AssertFact (CvC, RD, R, TractorpullTailer, Mobile, F, H)
Assert CombFacts  (CvC, RD, J, or,{TractorpullTailer, motorhome}, container, F, H)
AssertCombFactors (CvC, RD, J, and,{mobile, mobileStatus}, Exigency, I, H)
AssertFact (CvC, RD, R, tractorpullTailer, residentialUse, F, H)
AssertFact (CvC, RD, R, tractorpullTailer, container, F, H)
AssertFact (CvC, RD, R, trailer, not visible, F, H)
AssertCombFactors (CvC, RD, R, and,{mobile, residentialUse,container,not visible},
 Privacy, I, H)
AssertFact (CvC, RD, R, houseCar, residentialUse, F, H)
CloseDialogue(ChiefJustic,RD, Thomas Homann,CvC)
```

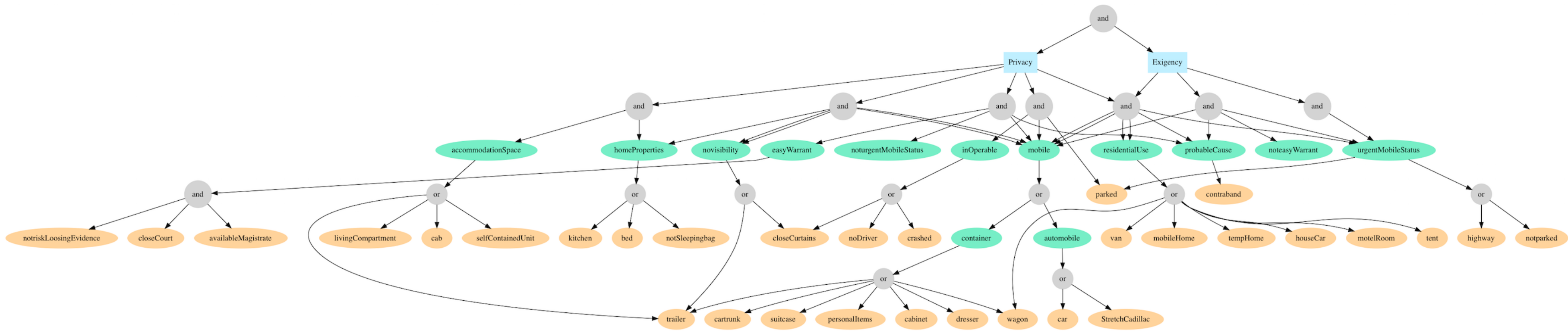


FIGURE C.3: California v. Carney Respondent Dialogue - Respondent ACT

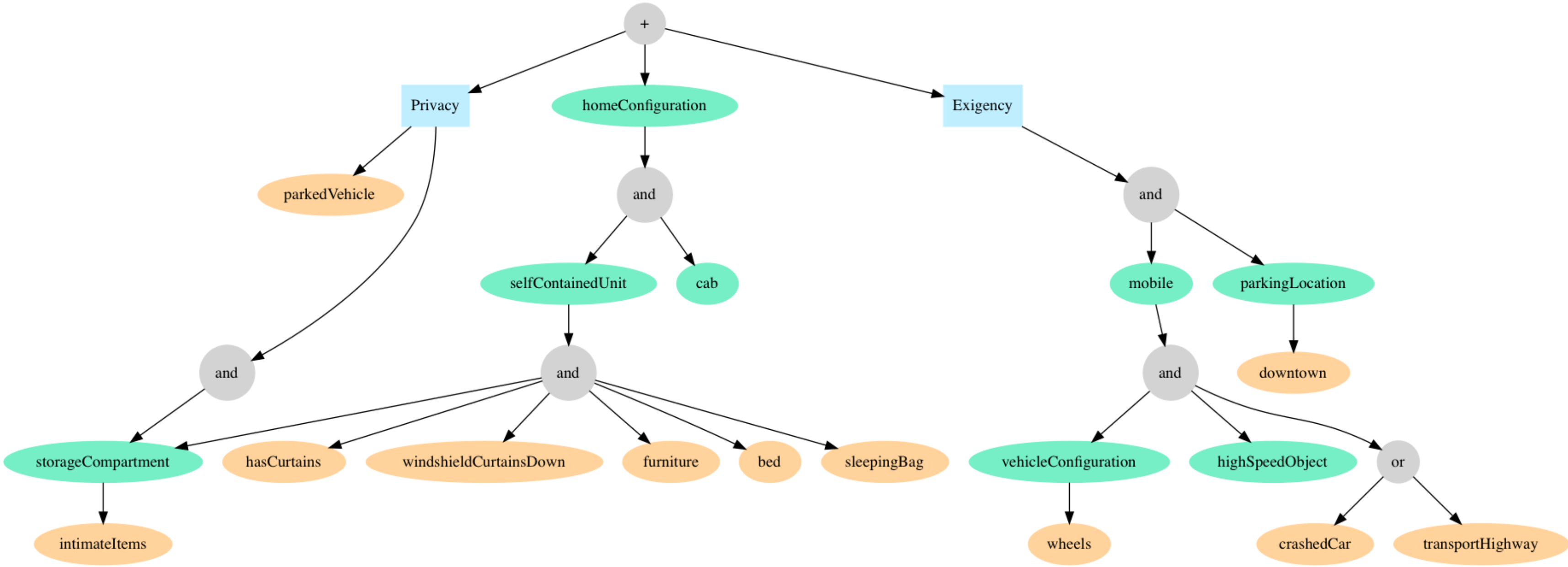


FIGURE C.4: California v. Carney Respondent Dialogue - Justices' ACT

C.3 Petitioner Rebuttal Dialogue

```
OpenDialogue(ChiefJustic,PDR, Louis Hanoian,CvC)
AssertFact (CvC,PD,P, LivingCompartment, ResidentialUse, F, T)
AssertFactor (CvC,PD,P, Homeproperties, Privacy, I, N)
AssertFact (CvC,PD,P, PersonalEffects, Homeproperties, F, T)
AssertFact (CvC,PD,P, Food, Homeproperties, F, T)
Assert CombFacts  (CvC,PD,P, and, {PersonalEffects,Food}, Homeproperties,
F, H)
AssertCombFactors (CvC,PD,P, and, {ResidentialUse,Homeproperties},
 Privacy,I, H)
AssertCombFactors (CvC,PD,P, and, {noResidentialUse,no Homeproperties,
mobile}, Exigency,I, H)
AssertFact (CvC,PD,P, stretchcadillac, automobile, F, H)
AssertFact (CvC,PD,P, pickuptrucks, automobile, F, H)
AssertCombFactors (CvC,PD,P, and, {ResidentialUse,mobile}, Exigency,
I, H)
CloseDialogue(ChiefJustic,PDR, Louis Hanoian,CvC)
OpenCase (ChiefJustic, CvC)
```

Appendix D

U.S. Trade Secrets Factors Definitions

This appendix presents the definitions for CATO's Factor Hierarchy for the domain of US Trade Secrets law from [10, 239-247]. The domain contains 5 Legal Issues, 11 Intermediate Legal Concerns and 26 base-level factors listed below, together with their associated plaintiff and defendant conclusions as indicated by (p) or (d).

D.1 Legal Issues

F101 InfoTradeSecret (p)

P:Plaintiff's information is a trade secret.

D:Plaintiff's information is not a trade secret.

F110 ImproperMeansConclusion (p)

P: Defendant acquired plaintiff's information through improper means.

D: Defendant did not acquire plaintiff's information through improper means.

F112 InfoUsed (p)

P: Defendant may have used plaintiff's information and usurped a competitive advantage.

D: Defendant's information was the result of independent development efforts and investment.

F114 ConfidentialRelationship (p)

P: A confidential relationship existed between plaintiff and defendant.

D:There was no confidential relationship between the parties.

F124 DefendantOwnershipRights (d)

P: Defendant does not have ownership rights in the information.

D: Defendant may have ownership rights in the information.

D.2 Intermediate Legal Concerns

F102 EffortsToMaintainSecrecy (p)

P: Plaintiff took efforts to maintain the secrecy of its information.

D: Plaintiff showed a lack of interest in maintaining the secrecy of its information.

F104 InfoValuable (p)

P:Plaintiff's information was valuable for plaintiff's business.

D:It was not the case that plaintiff's information was valuable for plaintiff's business.

F105 InfoKnownOrAvailable (d)

P: The information apparently was not known or available outside plaintiff's business.

D: Plaintiff's information was known in the industry or available from sources outside plaintiff's business.

F106 InfoKnown (d)

P: The information apparently was not known outside plaintiff's business.

D: Plaintiff's information was known outside plaintiff's business.

F108 InfoAvailableElsewhere (d)

P: It was not the case that plaintiff's information was available from sources outside plaintiff's business.

D: Plaintiff's information was available from sources outside plaintiff's business.

F111 QuestionableMeans (p)

P: Defendant may have acquired plaintiff's information through questionable means.

D: Defendant apparently obtained its information by fair means.

F115 NoticeOfConfidentiality (p)

P: Defendant was on notice that using or disclosing the information would be a breach of confidentiality.

D: It was not the case that defendant was on notice that using or disclosing the information would be a breach of confidentiality.

F120 InfoLegitimatelyObtainedOrObtainable (d)

P: Defendant may have acquired plaintiff's information through improper means.

D: Defendant obtained or could have obtained its information by legitimate means.

F121 ExpressConfidentialityAgreement (p)

P: There was an express agreement to keep the information confidential.

D: There was an explicit disclaimer of confidentiality.

F122 EffortsToMaintainSecrecyVisAVisDefendant (p)

P: Plaintiff took efforts to maintain the secrecy of its information in its dealings with defendant.

D: Plaintiff showed a lack of interest in maintaining the confidentiality of its information in its dealings with defendant.

F123 EffortsToMaintainSecrecyVisAVisOutsiders (p)

P: Plaintiff took efforts to maintain the confidentiality of its information in its dealings with outsiders.

D: Plaintiff showed a lack of interest in maintaining the confidentiality of its information in its dealings with outsiders.

D.3 Base-Level Factors

F1 DisclosureInNegotiations (d)

Description: Plaintiff disclosed its product information in negotiations with defendant. This factor shows that defendant apparently obtained its information by fair means. Also, it shows that plaintiff showed a lack of interest in maintaining the secrecy of its information.

The factor applies if: Plaintiff disclosed the information to defendant in the context of (negotiations about) a joint venture, licensing agreement, sale of a business, etc.

The factor does not apply if: Defendant acquired knowledge of plaintiff's information in the course of employment by plaintiff.

F2 BribeEmployee (p)

Description: Defendant paid plaintiff's former employee to switch employment, apparently in an attempt to induce the employee to bring plaintiff's information.

This factor shows that defendant may have acquired plaintiff's information through questionable means.

The factor applies if: Defendant offered plaintiff's employee or former employee a substantial bonus or salary increase in order to work for defendant.

F3 EmployeeSoleDeveloper (d)

Description: Employee defendant was the sole developer of plaintiff's product.

This factor shows that defendant may have ownership rights in the information.

The factor does not apply if: Defendant contributed to the development or improvement of plaintiff's product, but was not the sole developer.

F4 AgreedNotToDisclose (p)

Description: Defendant entered into a nondisclosure agreement with plaintiff.

This factor shows that defendant was on notice that using or disclosing the information would be a breach of confidentiality. Also, it shows that there was an express agreement to keep the information confidential. Also, that plaintiff took efforts to maintain the secrecy of its information.

The factor does not apply if: Plaintiff obtained nondisclosure agreements from other employees but not from the defendant.

F5 AgreementNotSpecific (d)

Description: The nondisclosure agreement did not specify which information was to be treated as confidential.

This factor shows that it was not the case that defendant was on notice that using or disclosing the information would be a breach of confidentiality.

The factor does not apply if: There is no information about the contents of the nondisclosure agreement.

F6 SecurityMeasures (p)

Description: Plaintiff adopted security measures.

This factor shows that plaintiff took efforts to maintain the secrecy of its information.

The factor applies if: Plaintiff took active measures to limit access to and distribution of its information, for example through employee nondisclosure agreements, notifying employees that information is confidential and not to be divulged to outsiders, keeping important documents under lock and key, document distribution systems, stamping documents confidential, computer passwords, plant security, requiring outsiders to whom information is disclosed to sign nondisclosure agreements, keeping sensitive information hidden when plant tours are conducted, etc.

F7 BroughtTools (p)

Description: Plaintiff's former employee brought product development information to defendant. This factor shows that defendant may have used plaintiff's information and usurped a competitive advantage. Also, it shows that defendant may have acquired plaintiff's information through questionable means.

The factor applies if: Plaintiff's (former) employee, took product development information such as copies of blueprints, documents, customer lists, computer printouts, disks, tapes, actual specimen of plaintiff's product, parts, tools, etc.

The factor does not apply if: Defendant had somehow come into possession of plaintiff's documents, blueprints, etc., but there was no evidence that an employee of plaintiff's was involved.

F8 CompetitiveAdvantage (p)

Description: Defendant's access to plaintiff's product information saved it time or expense.

This factor shows that defendant may have used plaintiff's information and usurped a competitive advantage. Also, it shows that plaintiff's information was valuable for plaintiff's business.

The factor applies if: It was documented that defendant developed its product at lower cost or in less time than it took plaintiff.

The factor does not apply if: All we know is that the information afforded plaintiff a competitive advantage (e.g., by enabling it to manufacture a product that was superior to the products made by competitors). Or if all we know is that plaintiff spent considerable time and money in developing the information.

F10 SecretsDisclosedOutsiders (d)

Description: Plaintiff disclosed its product information to outsiders.

This factor shows that plaintiff's information was known in the industry or available from sources outside plaintiff's business. Also, it shows that plaintiff showed a lack of interest in maintaining the secrecy of its information.

The factor applies if: Plaintiff disclosed its product information for example to licensees, customers, suppliers, subcontractors, etc.

The factor does not apply if: Plaintiff published the information in a public forum. (In that situation, F27 applies.) All we know is that plaintiff marketed a product from which the information could be ascertained by reverse engineering.

F11 VerticalKnowledge (d)

Description: Plaintiff's information is about customers and suppliers (which means that it may be available independently from customers or even in directories).

This factor shows that defendant obtained or could have obtained its information by legitimate means.

The factor applies if: Plaintiff's information consists of customer information such as customer lists or information about customer business methods.

F12 OutsiderDisclosuresRestricted (p)

Description: Plaintiff's disclosures to outsiders were subject to confidentiality restrictions.

This factor shows that the information apparently was not known or available outside plaintiff's business. Also, it shows that plaintiff took efforts to maintain the secrecy of its information.

The factor applies if: Plaintiff required that outsiders who received the information keep it confidential or do not use it for any purpose other than for which it was given.

The factor does not apply if: All we know is that plaintiff restricted the number of disclosees or the extent of the information that was disclosed.

F13 NoncompetitionAgreement (p)

Description: Plaintiff and defendant entered into a noncompetition agreement.

This factor shows that defendant was on notice that using or disclosing the information would be a breach of confidentiality.

The factor applies if: Defendant entered into an agreement, promising not to compete with plaintiff or work for a competitor after termination of his or her employment by plaintiff.

F14 RestrictedMaterialsUsed (p)

Description: Defendant used materials that were subject to confidentiality restrictions.

This factor shows that defendant was on notice that using or disclosing the information would be a breach of confidentiality. Also, it shows that defendant may have acquired plaintiff's information through questionable means.

The factor applies if: Defendant used documents or materials that plaintiff had marked as confidential or that were subject to a confidentiality agreement between plaintiff and defendant.

F15 UniqueProduct (p)

Description: Plaintiff was the only manufacturer making the product.

This factor shows that the information apparently was not known or available outside plaintiff's business. Also, it shows that plaintiff's information was valuable for plaintiff's business.

The factor applies if: Plaintiff's product or process was unique on the market or industry, or had marketable features not found in competitors' products.

F16 InfoReverseEngineerable (d)

Description: Plaintiff's product information could be learned by reverseengineering.

This factor shows that plaintiff's information was known in the industry or available from sources outside plaintiff's business.

The factor applies if: Plaintiff's information could be ascertained by reverse engineering, that is, by inspecting or analyzing plaintiff's product (regardless of whether defendant actually obtained the information in this way).

F17 InfoIndependentlyGenerated (d)

Description: Defendant developed its product by independent research.

This factor shows that defendant's information was the result of independent development efforts and investment. Also, it shows that defendant apparently obtained its information by fair means.

The factor applies if: Defendant developed its product or information independently, without recourse to plaintiff's information.

F18 IdenticalProducts (p)

Description: Defendant's product was identical to plaintiff's.

This factor shows that defendant may have used plaintiff's information and usurped a competitive advantage.

F19 NoSecurityMeasures (d)

Description: Plaintiff did not adopt any security measures.

This factor shows that plaintiff showed a lack of interest in maintaining the secrecy of its information.

The factor does not apply if: Plaintiff took at least some security measures, even if other security measures were not taken, or if there is no information about security measures. Or if all we know is that plaintiff disclosed its information to defendant or to outsiders. (In those situations, F1 and F10 apply, respectively.)

F20 InfoKnownToCompetitors (d)

Description: Plaintiff's information was known to competitors.

This factor shows that plaintiff's information was known in the industry or available from sources outside plaintiff's business.

The factor applies if: The information plaintiff claims as its trade secret is general knowledge in the industry or trade.

The factor does not apply if: Competitors' knowledge of plaintiff's information results solely from disclosures made by plaintiff. (In this situation, F10 applies.) Or if the information could be compiled from publicly available sources, but there was no evidence that competitors had actually done so. (In this situation, F24 applies.)

F21 KnewInfoConfidential (p)

Description: Defendant knew that plaintiff's information was confidential.
This factor shows that defendant was on notice that using or disclosing the information would be a breach of confidentiality.
The factor applies if: Defendant knew that plaintiff intended its information to be treated as confidential (regardless of how defendant had come to know this).
The factor does not apply if: Defendant entered into a nondisclosure agreement with plaintiff, but there is no evidence that defendant knew specifically which information was to be treated as confidential. (In that situation, F4 applies.)

F22 InvasiveTechniques (p)

Description: Defendant used invasive techniques to gain access to plaintiff's information.
This factor shows that defendant may have acquired plaintiff's information through questionable means.
The factor applies if: Defendant used invasive methods in a deliberate attempt to obtain plaintiff's information. These may be illegal methods, such as theft, surreptitious methods, such as rifling through trash or eavesdropping, methods devised specifically to circumvent security measures, methods against which it would be very difficult to guard, such as aerial photography, etc.
The factor does not apply if: Defendant tried to bribe plaintiff's employees to disclose confidential information. (In this situation, F2 applies.) Defendant obtained copies of documents, blueprints, tools, etc. via a (former) employee of plaintiff's. (In this situation, F7 applies.)

F23 WaiverOfConfidentiality (d)

Description: Plaintiff entered into an agreement waiving confidentiality.
This factor shows that there was an explicit disclaimer of confidentiality. Also, it shows that it was not the case that defendant was on notice that using or disclosing the information would be a breach of confidentiality. Also, that plaintiff showed a lack of interest in maintaining the secrecy of its information.
The factor applies if: Plaintiff acknowledged that defendant did not receive any information in confidence.

F24 InfoObtainableElsewhere (d)

Description: The information could be obtained from publicly available sources.
This factor shows that plaintiff's information was known in the industry or available from sources outside plaintiff's business.
The factor does not apply if: Plaintiff's information was general knowledge in the

industry. (In that situation, F20 applies.) Or if plaintiff's information could be discovered by reverse engineering plaintiff's product. (In that situation, F16 applies.)

F25 InfoReverseEngineered (d)

Description: Defendant discovered plaintiff's information through reverse engineering. This factor shows that defendant apparently obtained its information by fair means.

The factor applies if: Defendant reverse engineered plaintiff's product (i.e., examined or analyzed the product to find out its constituent parts or the process by which it was made).

F26 Deception (p)

Description: Defendant obtained plaintiff's information through deception.

This factor shows that defendant may have acquired plaintiff's information through questionable means.

The factor applies if: Defendant deceived plaintiff so as to gain access to its information, or was otherwise dishonest in its dealings with plaintiff.

F27 DisclosureInPublicForum (d)

Description: Plaintiff disclosed its information in a public forum.

This factor shows that plaintiff showed a lack of interest in maintaining the secrecy of its information. Also, it shows that plaintiff's information was known in the industry or available from sources outside plaintiff's business.

The factor applies if: Plaintiff made presentations about its information during meetings that were open to the general public, for example, scientific seminars, trade shows, etc. Also if plaintiff published its information in magazine articles, trade publications, publicity material, patents, etc.

The factor does not apply if: Plaintiff disclosed its information to specific outsiders. (In that situation, F10 applies.)

Appendix E

Automobile Exception Factors Definitions

This appendix presents the identification and definitions of 2 issues (Section E.1), 13 abstract factors (Section E.2) and 31 base-level factors (Section E.3) for the Automobile Exception to the Fourth Amendment domain listed below, together with their associated plaintiff and defendant conclusions as indicated by (p) or (d).

E.1 Legal Issues

I1:Exigency (p)
There is sufficient urgency to allow the plaintiff to search the automobile without obtaining a warrant under the automobile exception to the fourth amendment.

I2:Privacy (d)
There is enough expectation of privacy to obtain a warrant before searching an automobile.

E.2 Abstract Factors

AF2-0-2:ProbableCauseSearchVehicle (p)
The factor is applied when a notification has been received from an authorized origin source about a crime or any other urgent situation that requires immediate search without first obtaining a warrant.

AF2-0-3:ExpectationOfPrivacyInUseRequired(PrivacyRights) (d)
The factor is applied when the searching place or object have a high expectation of privacy that prohibits searching without first obtaining a warrant. That involves any residential area, a place that has some accommodation properties and is not under any exception to the fourth amendment rule, or personal effects in a private container.

AF1-0-1:Mobile (p)

This factor involves any object that is able to move quickly because it has an engine or wheels (such as automobiles and vessels, including towable wagons or vehicles), or that is able to be moved by a person (for example,large containers or small, easily moved containers).

AF1-0-2:EaseWarrant (d)

This factor tests the possibility of obtaining a warrant under the current circumstances. This involves the distance to the closest court and the availability of the magistrate as well as the authority of the available magistrate.

AF1-0-3:SubjectToInspectionRegulation (p)

The mobile object is under inspection regulation and thus it can be searched without first obtaining a warrant giving a certain procedure.

AF1-0-4:VisibilityOfItem (p)

The defendants object is placed in a visible area that makes it possible for anyone to see what is inside the automobile.

AF1-0-5:ExigencyWhenApproached (p)

This factor determines the level of urgency when reaching the automobile by checking if the automobile is moving quickly on the highway, or is parked in a private parking lot or in the middle of the town but is capable of moving at any time.

AF1-2-1:AuthorizedOriginOfProbableCause (p)

The notification of the probable cause to search was received from an authorized person through a proper method: either receiving information, observing the defendant, or through the normal regulations.

AF1-2-2:UrgentReasonToSearch (p)

This factor specifies the reason for the search, e.g. when there is a crime, a robbery, or transportation and distribution of illegal substances that requires immediate action to protect the public.

AF1-2-3:LegalSearchScope (p)

This factor determines the allowed searching areas and places that agent officers can search without obtaining a warrant.

AF1-3-1:PrivateContentsCarriage (d)

The defendant is carrying private materials (e.g. papers or personal effects) in the container placed in the automobile.

AF1-3-2:Residence (d)

The searched area has the main requirements for a residential dwelling, e.g. connected to water, gas or electricity.

AF1-3-3:Accommodation (d)

The searched area shows some home properties and is not capable of moving quickly, which indicates that the defendant is using it for accommodation purposes.

E.3 Base-Level Factors

BF-01-1:Automobile (p)

The defendant's automobile refers to any type of vehicle that has an engine and wheels, making it capable of moving quickly.

The factor applies if: car, truck, tractor, lorry, limo, van, motorhome etc.

BF-01-2:Vessel (p)

The defendant's mobile vehicle is a vessel, which can be any sail-powered or motor-driven boat or ship.

The factor applies if: motorboat, sailboat, rowboat, ship, jet skis.

BF-01-3:Towable (p)

The defendant's vehicle is other than a motor vehicle, designed for habitation, industrial, professional, or commercial purposes, for carrying property on its own structure, and for being drawn by a motor vehicle.

The factor applies if: box trailers, horse trailers, wagon, cart etc.

BF-01-4:LargeContainer (p)

This defendant's mobile object is a large, sometimes heavy, container that is not easy to hold.

The factor applies if: large goods container, footlocker, large baggage etc.

BF-01-5:MovableContainer (p)

This refers to small and easy-to-hold containers that can be found inside the defendant's automobile.

The factor applies if: wallet, pouch, paper bag, briefcase, suitcase etc.

BF-02-1:AuthorityOfAvailableMagistrate) (d)

The warrant is issued by a neutral, detached (unauthorized) magistrate.

The factor applies if: neutral detached magistrate.

The factor does not apply if: authorized magistrate.

BF-02-2:RiskofLosingEvidence (p)

This involves any situations where obtaining a warrant might cause loss of evidence.

The factor applies if: farCourt.

The factor does not apply if: nearbyCourt.

BF-02-3:AvailabilityofMagistrate (d)

This factor refers to the searching time and court type where the plaintiffs authorized magistrate is available.

The factor applies if: workingTime.

The factor does not apply if: overnight.

BF-03-1:Licence (p)

Every automobile driver should have a licence; the licence's factor refers to the type of licence issued for this vehicle or driver.

The factor applies if: vehicle licence , motorhome special licence, truck licence.

The factor does not apply if: if no license is required for a mobile object.

BF-03-2:RestrictedArea (d)

Automobile exception warrantless search is not allowed in some restricted areas that require high privacy or certain procedures.

The factor applies if: home, policeStation, airport.

The factor does not apply if: highways, borderCrossing.

BF-04-1:OnPublicView (p)

The defendant has not protected the object from being visible. The object is visible to the public.

The factor applies if: the object for example is on the vehicles front seat and the window curtains are not down.

The factor does not apply if: the object is in any other place that cannot be seen by the public.

BF-04-2:CanBeseen (p)

The object placed inside the vehicle is not meant to be on public view but it can be seen by the public.

The factor applies if: the object is on the floor, or in a transparent bag.

The factor does not apply if: the object is in any other place that cannot be seen by the public.

BF-04-3:CanotBeSeen (d)

The defendant has protected the object from being visible to the public.

The factor applies if: glove box, car boot or car trunk, living compartment, opaque container.
The factor does not apply if: the object is placed in a visible place.

BF-05-1:UrgentStatus (p)

The factor describes any situation when the evidence might be lost.
The factor applies if: the automobile is moving, or it is parked in a public parking space but is capable of moving quickly at any time.
The factor does not apply if: the automobile is crashed and not capable of moving

BF-05-2:CapableToMove (p)

The automobile is stopped or parked but has all the supporting properties that make it capable of moving quickly.
The factor applies if: for example the automobile driver is inside the vehicle i.e. the vehicle is occupied, the vehicle curtains are open, or there is a motive force.
The factor does not apply if: the vehicle is crashed, no driver inside the vehicle.

BF-05-3:PublicParking (p)

The vehicle is parked in a public parking lot, which means that no privacy measures are expected.
The factor applies if: an automobile is on the highway, or parked in a public parking lot.
The factor does not apply if: an automobile is parked inside a dwelling area or on the owners land, or in a private parking lot.

BF-05-4:PublicLocation (p)

In addition to the parking type, the location specifies where exactly the automobile is parked or moving.
The factor applies if: an automobile is parked in a downtown or public parking location.
The factor does not apply if: an automobile is parked in a dwelling area, whether that be urban residential, suburban or rural.

BF-05-5:PermittedDuration (p)

The factor expresses the allowed parking time for the automobile to determine whether it is used as a private dwelling or as a vehicle.
The factor applies if: an automobile is parked for short time.
The factor does not apply if: an automobile is parked overnight, for a long stay or an unknown time.

BF-21-1:Information (p)

The factor specifies the origin of notification of any suspicious automobile driver.

The factor applies if: information is received from the public or another agent officer.

The factor does not apply if: there is no notified information.

BF-21-2:Observation (p)

Suspicious behavior is observed before being notified.

The factor applies if: the behaviour is witnessed by observers from the public or an agent officer.

The factor does not apply if: a description of the behaviour was obtained from information or by following a specific procedure.

BF-21-3:Procedure (p)

A warrantless search is applied by following a certain automobile procedure.

The factor applies if: an automobile is parked in a multiple parking lot, normal inspection regulation, incident arrest.

The factor does not apply if: the origin of purpose was from information received or observation.

BF-22-1:PublicSafety (p)

The factor refers to the cause that required urgent search action without obtaining a warrant.

The factor applies if: there is a weapon inside the automobile, or the driver is transporting or selling illegal substances.

The factor does not apply if: normal procedural checking.

BF-22-1:Crime (p)

The factor describes that the cause of the immediate search is a certain crime situation.

The factor applies if: for example, knowing that there is smuggling, dealing, a murder has occurred inside the car, or the driver is involved in a robbery.

The factor does not apply if: another reason that is not a crime.

BF-23-1:wholeVehicle (p)

A warrantless search is applied to all parts of the automobile.

The factor applies if: all automobile parts.

The factor does not apply if: there is only a specific container within the automobile.

BF-23-2:OnlyVehicleContainer (p)

A warrantless search is applied only to part(s) of the vehicle but not the entire vehicle.

The factor applies if: car trunk, glove compartment only are searched.

The factor does not apply if: the whole automobile is searched.

BF-23-3:SearchPlace (p)

The factor refers to the legal place or location area to search the automobile.

The factor applies if: police station, a garage, automobile public location.

The factor does not apply if: any other private areas.

BF-31-1:GoodsCarried (d)

The factor specifies the type of private contents inside a container.

The factor applies if: personal effects, papers, commercial items, money.

The factor does not apply if: weapons, illegal substances.

BF-31-2:ProtectionType (d)

Efforts to protect the mobile container from begin opened by someone else.

The factor applies if: locked, double locked.

The factor does not apply if: no effort is required because the container is open, or closed normally.

BF-32-1:ConnectedServices (d)

The defendant's automobile is connected to certain living services.

The factor applies if: gas, electricity, water.

The factor does not apply if: the automobile is not connected to any services.

BF-33-1:AccommodationSpaces (d)

The defendant's automobile has a specific space that is suitable for accommodation.

The factor applies if: the automobile has a suitable accommodation space.

The factor does not apply if: the automobile has a cab area only.

BF-33-2:RoomsFunction (d)

The factor specifies whether there are any room features inside the automobile.

The factor applies if: bed, bathroom, kitchen, living room

The factor does not apply if: there are no room features.

Appendix F

ANGELIC Secrets Prolog Program and Results

This appendix demonstrates the Prolog program for the US Trade Secrets domain, showing the output from 32 cases. Some further brief details about the program are provided in Section F.1, while the results are shown in Section F.2.

F.1 Program

The Prolog program starts with the knowledge base of the case representation. Each case consists of two parameters: the case name; and a list of the *base level factors* representing the case. The procedure *go(C):-case(C,Factors)* holds the case *C* and the factors of *Factors* and starts navigating between the abstract factors. The existence of an abstract factor is determined by the existence of one or more of its children (base factors) in the list *Factors* according to the defined acceptance conditions in Chapter 5. If the abstract factor is satisfied, a report describing the factor is written, and the abstract factor is added to the list *Factors*. Every factor ends with a default procedure that reports the *default* status of the abstract factor in the running case. After examining all the abstract factors and issues, the final case decision is determined by the root (Trade Secrets Misappropriation) in *getf100(C,Factors)* and the results are validated in the *finish* procedure by comparing the obtained result with the case actual result in the *result* fact determined at the beginning of the program.

```
%Cases knowledge base
case(arco,[f10, f16,f20]).
case(boeing,[f4,f6,f12,f14,f21,f1,f10]).
case(bryce,[f4,f6,f18,f21,f1]).
case(collegeWatercolour,[f15,f26,f1]).
case(denTalEz,[f4,f6,f21,f26,f1]).
case(ecolgix,[f21,f1,f19,f23]).
case(emery,[f18,f21,f10]).
case(ferranti,[f2,f17,f19,f20,f27]).
case(robinson,[f18,f26,f1,f10,f19]).
case(sandlin,[f1,f10,f16,f19,f27]).
case(sheets,[f18,f19,f27]).
case(spaceAero,[f8,f15,f18,f1,f19]).
```

```
case(televation, [f6,f12,f15,f18,f21,f10,f16]).
case(yokana,[f7,f10,f16,f27]).
case(cm1,[f4,f6,f20,f17,f16,f10,f27]).
case(digitalDevelopment,[f6,f8,f15,f18,f21,f1]).
case(fmc,[f4,f6,f7,f12,f10,f11]).
case(forrest,[f6,f15,f21,f1]).
case(goldberg,[f21,f1,f10,f27]).
case(kg,[f6,f14,f15,f18,f21,f16,f25]).
case(laser,[f6,f12,f21,f1,f10]).
case(lewis,[f8,f21,f1]).
case(mbl,[f4,f6,f13,f5,f10,f20]).
case(mason,[f6,f15,f21,f1,f16]).
case(mineralDeposits,[f18,f14,f1,f16,f25]).
case(nationalInstruments, [f18,f21,f1]).
case(nationalRejectors,[f7,f15,f18,f10,f16,f19,f27]).
case(reinforced,[f4,f6,f8,f15,f21,f1]).
case(scientology,[f4,f6,f12,f10,f11,f20]).
case(technicon,[f6,f12,f14,f21,f10,f16,f25]).
case(trandes,[f4,f6,f12,f1,f10]).
case(valcoCincinnati,[f6,f12,f15,f21,f1,f10]).
```

% "result" procedure is called by "finish" procedure at the end of the program to compare between the program output and the actual case decision

```
result(arco,d).
result(boeing,p).
result(bryce,p).
result(collegeWatercolour,p).
result(denTalEz,p).
result(ecolgix,d).
result(emery,p).
result(ferranti,d).
result(robinson,d).
result(sandlin,d).
result(sheets,d).
result(spaceAero,p).
result(televation,p).
result(yokana,d).
result(cm1,d).
result(digitalDevelopment,p).
result(fmc,p).
result(forrest,p).
result(goldberg,p).
result(kg,p).
result(laser,p).
result(lewis,p).
result(mbl,d).
result(mason,p).
result(mineralDeposits,p).
result(nationalInstruments,p).
result(nationalRejectors,d).
result(reinforced,p).
result(scientology,d).
result(technicon,p).
result(trandes,p).
result(valcoCincinnati,p).
```

% Retrieve the case factors and navigate between factors

```
go(C):-case(C,Factors),
```

```
                        getf124(C,Factors).

getf124(C,Factors):-member(f3,Factors),
                        write([defendant,is,owner,of,secret]),nl,
                        getf123(C,[f124|Factors]).
getf124(C,Factors):- write([defendant,is,not,owner,of,secret]),nl,
                        getf123(C,Factors).


getf123(C,Factors):-member(f12,Factors),
                        write([efforts,made,vis,a,vis,outsiders]),nl,
                        getf122(C,[f123|Factors]).
getf123(C,Factors):-write([no,efforts,made,vis,a,vis,outsiders]),nl,
                        getf122(C,Factors).

getf122(C,Factors):-member(f4,Factors),
                        write([efforts,made,vis,a,vis,defendant]),nl,
                        getf121(C,[f122|Factors]).
getf122(C,Factors):-write([no,efforts,made,vis,a,vis,defendant]),nl,
                        getf121(C,Factors).

getf121(C,Factors):-member(f4,Factors), not(member(f23,Factors)),
                        write([there,was,a,confidentiality,agreement]),nl,
                        getf115(C,[f121|Factors]).
getf121(C,Factors):-write([there,was,no,confidentiality,agreement]),nl,
                        getf115(C,Factors).


getf120(C,Factors):-member(f105,Factors),
                        write([the,information,was,legitimately,obtained]),nl,
                        getf110(C,[f120|Factors]).
getf120(C,Factors):-write([the,information,was,not,legitimately,obtained]),nl,
                        getf110(C,Factors).


getf115(C,Factors):-member(f21,Factors),
                        write([defendant,was,on,notice,of,confidentiality]),nl,
                        getf114(C,[f115|Factors]).
getf115(C,Factors):-member(f21,Factors),
                        write([defendant,was,on,notice,of,confidentiality]),nl,
                        getf114(C,[f115|Factors]).
getf115(C,Factors):-member(f4,Factors),
                        write([defendant,was,on,notice,of,confidentiality]),nl,
                        getf114(C,[f115|Factors]).
getf115(C,Factors):-member(f14,Factors),
                        write([defendant,was,on,notice,of,confidentiality]),nl,
                        getf114(C,[f115|Factors]).
getf115(C,Factors):-member(f13,Factors),
                        write([defendant,was,on,notice,of,confidentiality]),nl,
                        getf114(C,[f115|Factors]).
getf115(C,Factors):-write([defendant,was,not,on,notice,of,confidentiality]),nl,
                        getf114(C,Factors).

getf114(C,Factors):-member(f115,Factors),
                        write([there,was,a,confidential,relationship]),nl,
                        getf112(C,[f114|Factors]).
getf114(C,Factors):-member(f121,Factors),
                        write([there,was,a,confidential,relationship]),nl,
                        getf112(C,[f114|Factors]).
```

```
getf114(C,Factors):-write([there,was,no,confidential,relationship]),nl,
                        getf112(C,Factors).


getf112(C,Factors):-member(f7,Factors),
                        write([the,information,was,used]),nl,
                        getf111(C,[f112|Factors]).
getf112(C,Factors):-member(f18,Factors),
                        write([the,information,was,used]),nl,
                        getf111(C,[f112|Factors]).
getf112(C,Factors):-member(f8,Factors),
                        write([the,information,was,used]),nl,
                        getf111(C,[f112|Factors]).
getf112(C,Factors):-member(f17,Factors),
                        write([the,information,was,not,used]),nl,
                        getf111(C,Factors).
getf112(C,Factors):-write([the,information,was,used]),nl,
                        getf111(C,[f112|Factors]).


getf111(C,Factors):-member(f25,Factors),
                        write([questionable,means,were,not,used]),nl,
                        getf120(C,Factors).
getf111(C,Factors):-member(f17,Factors),
                        write([questionable,means,were,not,used]),nl,
                        getf120(C,Factors).
getf111(C,Factors):-member(f22,Factors),
                        write([questionable,means,were,used]),nl,
                        getf120(C,[f111|Factors]).
getf111(C,Factors):-member(f2,Factors),
                        write([questionable,means,were,used]),nl,
                        getf120(C,[f111|Factors]).
getf111(C,Factors):-member(f26,Factors),
                        write([questionable,means,were,used]),nl,
                        getf120(C,[f111|Factors]).
getf111(C,Factors):-member(f14,Factors),
                        write([questionable,means,were,used]),nl,
                        getf120(C,[f111|Factors]).
getf111(C,Factors):-not(member(f1,Factors)),
                        write([questionable,means,were,not,used]),nl,
                        getf120(C,Factors).
getf111(C,Factors):-write([questionable,means,were,not,used]),nl,
                        getf120(C,Factors).


getf110(C,Factors):-member(f120,Factors),
                        write([improper,means,were,not,used]),nl,
                        getf108(C,Factors).
getf110(C,Factors):-member(f111,Factors),
                        write([improper,means,were,used]),nl,
                        getf108(C,[f110|Factors]).
getf110(C,Factors):-write([improper,means,were,not,used]),nl,
                        getf108(C,Factors).


% exclude when restricted materials used
getf108(C,Factors):-member(f16,Factors),member(f14,Factors),
                        write([the,information,was,not,available,elsewhere]),nl,
                        getf106(C,Factors).
```

```
% exclude when identical products
getf108(C,Factors):-member(f16,Factors),member(f18,Factors),
                        write([the,information,was,not,available,elsewhere]),nl,
                        getf106(C,Factors).
getf108(C,Factors):-member(f16,Factors),
                        write([the,information,was,available,elsewhere]),nl,
                        getf106(C,[f108|Factors]).
getf108(C,Factors):-member(f24,Factors),
                        write([the,information,was,available,elsewhere]),nl,
                        getf106(C,[f108|Factors]).
getf108(C,Factors):-write([the,information,was,not,available,elsewhere]),nl,
                        getf106(C,Factors).


getf106(C,Factors):-member(f15,Factors),
                        write([information,is,not,known]),nl,
                        getf105(C,Factors).
getf106(C,Factors):-member(f20,Factors),
                        write([information,is,known]),nl,
                        getf105(C,[f106|Factors]).
getf106(C,Factors):-write([information,is,not,known]),nl,
                        getf105(C,Factors).


getf105(C,Factors):-member(f108,Factors),
                        write([the,information,was,known,or,available]),nl,
                        getf104(C,[f105|Factors]).
getf105(C,Factors):-member(f106,Factors),
                        write([the,information,was,known,or,available]),nl,
                        getf104(C,[f105|Factors]).
getf105(C,Factors):-
                        write([the,information,was,neither,known,nor,available]),nl,
                        getf104(C,Factors).

getf104(C,Factors):-member(f15,Factors),
                        write([the,information,was,valuable]),nl,
                        getf102(C,[f104|Factors]).
getf104(C,Factors):-member(f8,Factors),
                        write([the,information,was,valuable]),nl,
                        getf102(C,[f104|Factors]).
getf104(C,Factors):-member(f105,Factors),
                        write([the,information,was,not,valuable]),nl,
                        getf102(C,Factors).
getf104(C,Factors):-write([the,information,was,valuable]),nl,
                        getf102(C,[f104|Factors]).

getf102(C,Factors):-member(f6,Factors),
                        write([efforts,were,taken,to,maintain,secrecy]),nl,
                        getf101(C,[f102|Factors]).
getf102(C,Factors):-member(f122,Factors),
                        write([efforts,were,taken,to,maintain,secrecy]),nl,
                        getf101(C,[f102|Factors]).
getf102(C,Factors):-member(f123,Factors),
                        write([efforts,were,taken,to,maintain,secrecy]),nl,
                        getf101(C,[f102|Factors]).
getf102(C,Factors):-member(f19,Factors),
                        write([no,efforts,were,taken,to,maintain,secrecy]),nl,
                        getf101(C,Factors).
getf102(C,Factors):-
```

```
                        write([efforts,were,taken,to,maintain,secrecy]),nl,
                        getf101(C,[f102|Factors]).

getf101(C,Factors):-member(f102,Factors),member(f104,Factors),
                        not(member(f105,Factors)),
                        write([information,was,a,trade,secret]),nl,
                        getf100(C,[f101|Factors]).
getf101(C,Factors):-
                        write([information,was,not,a,trade,secret]),nl,
                        getf100(C,Factors).

getf100(C,Factors):-member(f124,Factors),
                        write([no,trade,secret,was,misappropriated]),nl,
                        write([find,for,defendant]),nl,
                        finish(C,Factors,d).
getf100(C,Factors):-member(f101,Factors),member(f110,Factors),
                        write([a,trade,secret,was,misappropriated]),nl,
                        write([find,for,plaintiff]),nl,
                        finish(C,[f100|Factors],p).
getf100(C,Factors):-member(f101,Factors),member(f114,Factors),
                        member(f112,Factors),
                        write([a,trade,secret,was,misappropriated]),nl,
                        write([find,for,plaintiff]),nl,
                        finish(C,[f100|Factors],p).
getf100(C,Factors):-
                        write([no,trade,secret,was,misappropriated]),nl,
                        write([find,for,defendant]),nl,
                        finish(C,Factors,d).
```

% Comparing between the decision from the program output and the case actual decision.

```
finish(C,Factors,R):-write(C),write(Factors),nl,
                        result(C,R),write(correct),nl.
finish(C,Factors,R):-write(C),write(Factors),nl,
                        write(wrong),nl.
```

F.2 Results

To obtain the case decision, the user calls the case using *go(case name)*. The output from each case provides the decision justification as a result from the factor reports, and the case decision: *find for plaintiff*,or, *find for defendant*. The list below illustrates the output from the 32 cases. As stated in Chapter 6, the output from the incorrect cases, SpaceAero and Mason, clarifies the difference between the actual decision and the program decision.

Arco

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,not,on,notice,of,confidentiality]
[there,was,no,confidential,relationship]
[the,information,was,used]
```

```
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,available,elsewhere]
[information,is,known]
[the,information,was,known,or,available]
[the,information,was,not,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
arco[f102,f105,f106,f108,f112,f10,f16,f20]
correct
```

Boeing

```
[defendant,is,not,owner,of,secret]
[efforts,made,vis,a,vis,outsiders]
[efforts,made,vis,a,vis,defendant]
[there,was,a,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
boeing[f100,f101,f102,f104,f110,f111,f112,f114,f115,
f121,f122,f123,f4,f6,f12,f14,f21,f1,f10]
correct
```

Bryce

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[efforts,made,vis,a,vis,defendant]
[there,was,a,confidentiality,agreement]
```

```
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
bryce[f100,f101,f102,f104,f112,f114,f115,f121,f122,
f4,f6,f18,f21,f1]
correct
```

CollegeWatercolour

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,not,on,notice,of,confidentiality]
[there,was,no,confidential,relationship]
[the,information,was,used]
[questionable,means,were,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
collegeWatercolour[f100,f101,f102,f104,f110,f111,f112,
f15,f26,f1]
correct
```

DenTalEz

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[efforts,made,vis,a,vis,defendant]
[there,was,a,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
denTalEz[f100,f101,f102,f104,f110,f111,f112,f114,f115,
f121,f122,f4,f6,f21,f26,f1]
correct
```

Ecolgix

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[no,efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
ecolgix[f104,f112,f114,f115,f21,f1,f19,f23]
```

```
correct
```

Emery

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
emery[f100,f101,f102,f104,f112,f114,f115,f18,f21,f10]
correct
```

Ferranti

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,not,on,notice,of,confidentiality]
[there,was,no,confidential,relationship]
[the,information,was,not,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,known]
[the,information,was,known,or,available]
[the,information,was,not,valuable]
[no,efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
```

```
[find,for,defendant]
ferranti[f105,f106,f2,f17,f19,f20,f27]
correct
```

Robinson

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,not,on,notice,of,confidentiality]
[there,was,no,confidential,relationship]
[the,information,was,used]
[questionable,means,were,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[no,efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
robinson[f104,f110,f111,f112,f18,f26,f1,f10,f19]
correct
```

Sandlin

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,not,on,notice,of,confidentiality]
[there,was,no,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,available,elsewhere]
[information,is,not,known]
[the,information,was,known,or,available]
[the,information,was,not,valuable]
[no,efforts,were,taken,to,maintain,secrecy]
```

```
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
sandlin[f105,f108,f112,f1,f10,f16,f19,f27]
correct
```

Sheets

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,not,on,notice,of,confidentiality]
[there,was,no,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[no,efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
sheets[f104,f112,f18,f19,f27]
correct
```

SpaceAero

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,not,on,notice,of,confidentiality]
[there,was,no,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
```

```
[the,information,was,valuable]
[no,efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
spaceAero[f104,f112,f8,f15,f18,f1,f19]
spaceAero[f104,f112,f8,f15,f18,f1,f19]
wrong
```

Televation

```
[defendant,is,not,owner,of,secret]
[efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
televation[f100,f101,f102,f104,f112,f114,f115,f123,f6,
f12,f15,f18,f21,f10,f16]
correct
```

Yokana

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,not,on,notice,of,confidentiality]
[there,was,no,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
```

```
[improper,means,were,not,used]
[the,information,was,available,elsewhere]
[information,is,not,known]
[the,information,was,known,or,available]
[the,information,was,not,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
yokana[f102,f105,f108,f112,f7,f10,f16,f27]
correct
```

CM1

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[efforts,made,vis,a,vis,defendant]
[there,was,a,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,not,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,available,elsewhere]
[information,is,known]
[the,information,was,known,or,available]
[the,information,was,not,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
cm1[f102,f105,f106,f108,f114,f115,f121,f122,f4,f6,f20,
f17,f16,f10,f27]
correct
```

DigitalDevelopment

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
```

```
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
digitalDevelopment[f100,f101,f102,f104,f112,f114,f115,
f6,f8,f15,f18,f21,f1]
correct
```

FMC

```
[defendant,is,not,owner,of,secret]
[efforts,made,vis,a,vis,outsiders]
[efforts,made,vis,a,vis,defendant]
[there,was,a,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
fmc[f100,f101,f102,f104,f112,f114,f115,f121,f122,f123,
f4,f6,f7,f12,f10,f11]
correct
```

Forrest

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
```

```
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
forrest[f100,f101,f102,f104,f112,f114,f115,f6,f15,f21,f1]
correct
```

Goldberg

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
goldberg[f100,f101,f102,f104,f112,f114,f115,f21,f1,f10,f27]
correct
```

Kg

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
kg[f100,f101,f102,f104,f112,f114,f115,f6,f14,f15,f18,
f21,f16,f25]
correct
```

Laser

```
[defendant,is,not,owner,of,secret]
[efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
laser[f100,f101,f102,f104,f112,f114,f115,f123,f6,f12,
```

```
f21,f1,f10]
correct
```

Lewis

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
lewis[f100,f101,f102,f104,f112,f114,f115,f8,f21,f1]
correct
```

MBL

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[efforts,made,vis,a,vis,defendant]
[there,was,a,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,known]
[the,information,was,known,or,available]
[the,information,was,not,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
```

```
[no,trade,secret,was,misappropriated]
[find,for,defendant]
mbl[f102,f105,f106,f112,f114,f115,f121,f122,f4,f6,f13,f5,f10,f20]
correct
```

Mason

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,available,elsewhere]
[information,is,not,known]
[the,information,was,known,or,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
mason[f102,f104,f105,f108,f112,f114,f115,f6,f15,f21,f1,f16]
mason[f102,f104,f105,f108,f112,f114,f115,f6,f15,f21,f1,f16]
wrong
```

MineralDeposits

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
```

```
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
mineralDeposits[f100,f101,f102,f104,f112,f114,f115,
f18,f14,f1,f16,f25]
correct
```

NationalInstruments

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
nationalInstruments[f100,f101,f102,f104,f112,f114,
f115,f18,f21,f1]
correct
```

NationalRejectors

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,not,on,notice,of,confidentiality]
[there,was,no,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
```

```
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[no,efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
nationalRejectors[f104,f112,f7,f15,f18,f10,f16,f19,f27]
correct
```

Reinforced

```
[defendant,is,not,owner,of,secret]
[no,efforts,made,vis,a,vis,outsiders]
[efforts,made,vis,a,vis,defendant]
[there,was,a,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
reinforced[f100,f101,f102,f104,f112,f114,f115,f121,
f122,f4,f6,f8,f15,f21,f1]
correct
```

Scientology

```
[defendant,is,not,owner,of,secret]
[efforts,made,vis,a,vis,outsiders]
[efforts,made,vis,a,vis,defendant]
[there,was,a,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
```

```
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,known]
[the,information,was,known,or,available]
[the,information,was,not,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,not,a,trade,secret]
[no,trade,secret,was,misappropriated]
[find,for,defendant]
scientology[f102,f105,f106,f112,f114,f115,f121,f122,
f123,f4,f6,f12,f10,f11,f20]
correct
```

Technicon

```
[defendant,is,not,owner,of,secret]
[efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
technicon[f100,f101,f102,f104,f112,f114,f115,f123,
f6,f12,f14,f21,f10,f16,f25]
correct
```

Trandes

```
[defendant,is,not,owner,of,secret]
[efforts,made,vis,a,vis,outsiders]
```

```
[efforts,made,vis,a,vis,defendant]
[there,was,a,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
trandes[f100,f101,f102,f104,f112,f114,f115,f121,f122,
f123,f4,f6,f12,f1,f10]
correct
```

ValcoCincinnati

```
[defendant,is,not,owner,of,secret]
[efforts,made,vis,a,vis,outsiders]
[no,efforts,made,vis,a,vis,defendant]
[there,was,no,confidentiality,agreement]
[defendant,was,on,notice,of,confidentiality]
[there,was,a,confidential,relationship]
[the,information,was,used]
[questionable,means,were,not,used]
[the,information,was,not,legitimately,obtained]
[improper,means,were,not,used]
[the,information,was,not,available,elsewhere]
[information,is,not,known]
[the,information,was,neither,known,nor,available]
[the,information,was,valuable]
[efforts,were,taken,to,maintain,secrecy]
[information,was,a,trade,secret]
[a,trade,secret,was,misappropriated]
[find,for,plaintiff]
valcoCincinnati[f100,f101,f102,f104,f112,f114,f115,
f123,f6,f12,f15,f21,f1,f10]
correct
```

Appendix G

ANGELIC Animals Prolog Program and Results

This Appendix demonstrates the Prolog program for the second legal domain, Wild Animals, showing the output from five cases. As described in the US Trade Secret domain (Appendix F), the program and results are illustrated in Section G.1 and Section G.2, respectively.

G.1 Program

The Prolog program starts with the knowledge base of the case representation. Each case consists of two parameters: the case name; and a list of the *base level factors* representing the case. The procedure *go(C):-case(C,Factors)* holds the case *C* and the factors of *Factors* and starts navigating between the abstract factors. The existence of an abstract factor is determined upon the existence of one or more of its children (base factors) in the list *Factors* according to the defined acceptance conditions in Chapter 6. If the abstract factor is satisfied, a report describing the factor is written, and the abstract factor is added to the list *Factors*. Every factor ends with a default procedure that reports the *default* status of the abstract factor in the running case. After examining all the abstract factors and issues, the final case decision is determined at *getdecision(C,Factors)*.

%Cases knowledge base

```
case(keeble,[nc,ol,mal,n,ds,pliv]).
case(pierson,[nc,hp,imp,ps,vermin]).
case(young,[nc,hp,imp,pliv,dliv]).
case(ghen,[nc,con,nob,pliv,dliv]).
case(popov,[nc,hp,ass,nob,pg,dg]).
```

% Retrieve the case factors and navigate between factors

```
go(Case):-case(Case,Factors),
                          getCapture(Case,Factors).

getCapture(Case,Factors):-not(member(nc,Factors)),
                          write([the,plaintiff,had,captured,the,quarry]),nl,
                          getOwnership(Case,[capture|Factors]).
getCapture(Case,Factors):-member(vermin,Factors),
```

```
                        member(hp,Factors),
                        write([the,plaintiff,had,captured,the,quarry]),nl,
                        getOwnership(Case,[capture|Factors]).
getCapture(Case,Factors):-
                        write([the,plaintiff,had,not,captured,the,quarry]),nl,
                        getOwnership(Case,Factors).

getOwnership(Case,Factors):-member(con,Factors),
                        write([the,plaintiff,owned,the,quarry]),nl,
                        getPMotive(Case,[owned|Factors]).
getOwnership(Case,Factors):-member(capture,Factors),
                        write([the,plaintiff,owned,the,quarry]),nl,
                        getPMotive(Case,[owned|Factors]).
getOwnership(Case,Factors):-
                        write([the,plaintiff,did,not,own,the,quarry]),nl,
                        getPMotive(Case,Factors).


getPMotive(Case,Factors):-member(pliv,Factors),
                        write([plaitiff,has,good,motive]),nl,
                        getDMotive(Case,[pMotive|Factors]).
getPMotive(Case,Factors):-member(ps,Factors),
                        not(member(dliv,Factors)),
                        write([plaitiff,has,good,motive]),nl,
                        getDMotive(Case,[pMotive|Factors]).
getPMotive(Case,Factors):-member(pg,Factors),
                        not(member(dliv,Factors)),
                        write([plaitiff,has,good,motive]),nl,
                        getDMotive(Case,[pMotive|Factors]).
getPMotive(Case,Factors):-write([plaitiff,has,no,good,motive]),nl,
                        getDMotive(Case,Factors).

getDMotive(Case,Factors):-member(mal,Factors),
                        write([defendant,has,no,good,motive]),nl,
                        getOwnLand(Case,Factors).
getDMotive(Case,Factors):-member(dliv,Factors),
                        write([defendant,has,good,motive]),nl,
                        getOwnLand(Case,[dMotive|Factors]).
getDMotive(Case,Factors):-member(ds,Factors),not(member(pliv,Factors)),
                        write([defendant,has,good,motive]),nl,
                        getOwnLand(Case,[dMotive|Factors]).
getDMotive(Case,Factors):-member(dg,Factors),
                        not(member(pliv,Factors)),
                        write([defendant,has,good,motive]),nl,
                        getOwnLand(Case,[dMotive|Factors]).
getDMotive(Case,Factors):-
                        write([defendant,has,no,good,motive]),nl,
                        getOwnLand(Case,Factors).

getOwnLand(Case,Factors):-member(ol,Factors),
                        write([plainiff,owned,the,land]),nl,
                        getrightToPursue(Case,[ownLand|Factors]).
getOwnLand(Case,Factors):-
                        write([plainiff,did,not,own,the,land]),nl,
                        getrightToPursue(Case,Factors).

getrightToPursue(Case,Factors):-member(ownLand,Factors),
                        write([plainiff,had,a,right,to,pursue,the,quarry]),nl,
                        getAntiSocial(Case,[rtToPursue|Factors]).
```

```
getrightToPursue(Case,Factors):-member(pMotive,Factors),member(hp,Factors),
                            write([plainiff,had,a,right,to,pursue,the,quarry]),nl,
                            getAntiSocial(Case,[rtToPursue|Factors]).
getrightToPursue(Case,Factors):-member(pMotive,Factors),
                            not(member(dMotive,Factors)),
                            write([plaintiff,had,a,right,to,pursue,the,quarry]),nl,
                            getAntiSocial(Case,[rtToPursue|Factors]).
getrightToPursue(Case,Factors):-
                            write([plaintiff,had,no,right,to,pursue,the,quarry]),nl,
                            getAntiSocial(Case,Factors).


getAntiSocial(Case,Factors):-member(n,Factors),
                            write([defendant,committed,an,antisocial,act]),nl,
                            getTrespass(Case,[antiSocial|Factors]).
getAntiSocial(Case,Factors):-member(imp,Factors),
                            not(member(dMotive,Factors)),
                            write([defendant,committed,an,antisocial,act]),nl,
                            getTrespass(Case,[antiSocial|Factors]).
getAntiSocial(Case,Factors):-
                            write([defendant,committed,no,antisocial,acts]),nl,
                            getTrespass(Case,Factors).

getTrespass(Case,Factors):-member(ol,Factors),
                            member(antiSocial,Factors),
                            write([defendant,committed,trespass]),nl,
                            getIllegalAct(Case,[trepass|Factors]).
getTrespass(Case,Factors):-
                            write([defendant,committed,no,trespass]),nl,
                            getIllegalAct(Case,Factors).


getIllegalAct(Case,Factors):-member(ass,Factors),
                            write([an,illegal,act,was,committed]),nl,
                            getdecision(Case,[illegal|Factors]).
getIllegalAct(Case,Factors):-member(trepass,Factors),
                            write([an,illegal,act,was,committed]),nl,
                            getdecision(Case,[illegal|Factors]).
getIllegalAct(Case,Factors):-write([no,illegal,act,was,committed]),nl,
                            getdecision(Case,Factors).
```

% Generate the case decision

```

getdecision(Case,Factors):-member(owned,Factors),
                            write([find,for,the,plaintiff]),nl,
                            write([find,against,the,defendant]),nl,
                            finish(Case,Factors).
getdecision(Case,Factors):-member(rtToPursue,Factors),
                            member(illegal,Factors),
                            not(member(nob,Factors)),
                            write([find,for,the,plaintiff]),nl,
                            write([find,against,the,defendant]),nl,
                            finish(Case,Factors).
getdecision(Case,Factors):-member(rtToPursue,Factors),
                            member(illegal,Factors),
                            write([do,not,find,for,the,plaintiff]),nl,
                            write([the,defendant,did,not,act,illegally]),nl,
                            write([do,not,find,against,the,defendant]),nl,
                            finish(Case,Factors).
```

```
getdecision(Case,Factors):-write([do,not,find,for,the,plaintiff]),nl,
                        write([find,for,the,defendant]),nl,

finish(Case,Factors).
finish(Case,Factors):-write(Case),write(Factors).
```

G.2 Results

Similar to the US Trade Secret domain, the case decision is obtained by calling *go(case name)*. The output from each case provides justification for the decision as a result of the factor reports, and produces a case decision. Some case decisions in Wild animals cases are not found for the plaintiff and not found for the defendant as in Popov (see the decision procedure above). As stated in Chapter 6, all the five cases were decided correctly. The output below illustrates the decisions for all cases.

Keeble

```
[the,plaintiff,had,not,captured,the,quarry]
[the,plaintiff,did,not,own,the,quarry]
[plaintiff,has,good,motive]
[defendant,has,no,good,motive]
[plaintiff,owned,the,land]
[plaintiff,had,a,right,to,pursue,the,quarry]
[defendant,committed,an,antisocial,act]
[defendant,committed,trespass]
[an,illegal,act,was,committed]
[find,for,the,plaintiff]
[find,against,the,defendant]
keeble[illegal,trepass,antiSocial,rtToPursue,ownLand,
pMotive,nc,ol,mal,n,ds,pliv]
true
```

Pierson

```
[the,plaintiff,had,not,captured,the,quarry]
[the,plaintiff,did,not,own,the,quarry]
[plaintiff,has,good,motive]
[defendant,has,no,good,motive]
[plaintiff,did,not,own,the,land]
[plaintiff,had,a,right,to,pursue,the,quarry]
[defendant,committed,an,antisocial,act]
[defendant,committed,no,trespass]
[no,illegal,act,was,committed]
[do,not,find,for,the,plaintiff]
```

```
[find,for,the,defendant]
pierson[antiSocial,rtToPursue,pMotive,nc,hp,imp,ps,
vermin]
true.
```

Young

```
[the,plaintiff,had,not,captured,the,quarry]
[the,plaintiff,did,not,own,the,quarry]
[plaintiff,has,good,motive]
[defendant,has,good,motive]
[plaintiff,did,not,own,the,land]
[plaintiff,had,a,right,to,pursue,the,quarry]
[defendant,committed,no,antisocial,acts]
[defendant,committed,no,trespass]
[no,illegal,act,was,committed]
[do,not,find,for,the,plaintiff]
[find,for,the,defendant]
young[rtToPursue,dMotive,pMotive,nc,hp,imp,
pliv,dliv]
true
```

Ghen

```
[the,plaintiff,had,not,captured,the,quarry]
[the,plaintiff,owned,the,quarry]
[plaintiff,has,good,motive]
[defendant,has,good,motive]
[plaintiff,did,not,own,the,land]
[plaintiff,had,no,right,to,pursue,the,quarry]
[defendant,committed,no,antisocial,acts]
[defendant,committed,no,trespass]
[no,illegal,act,was,committed]
[find,for,the,plaintiff]
[find,against,the,defendant]
ghen[dMotive,pMotive,owned,nc,con,nob,
pliv,dliv]
true
```

Popov

```
[the,plaintiff,had,not,captured,the,quarry]
[the,plaintiff,did,not,own,the,quarry]
```

```
[plaintiff,has,good,motive]
[defendant,has,good,motive]
[plaintiff,did,not,own,the,land]
[plaintiff,had,a,right,to,pursue,the,quarry]
[defendant,committed,no,antisocial,acts]
[defendant,committed,no,trespass]
[an,illegal,act,was,committed]
[do,not,find,for,the,plaintiff]
[the,defendant,did,not,act,illegally]
[do,not,find,against,the,defendant]
popov[illegal,rtToPursue,dMotive,pMotive,nc,hp,
ass,nob,pg,dg]
true
```

Appendix H

ANGELIC Automobile Prolog Program and Results

This appendix presents the Prolog program and results for the Automobile Exception domain. The analysis and the domain ADFs for 10 cases were provided in Chapter 5 and Chapter 6. The program presented in this appendix will consider the inclusion of facts and the citation of precedent cases in the ADF as shown in Section H.1 and how the program has been updated to include the dissenting opinion rules in Section H.2. Finally the program output for the 10 cases is demonstrated in Section H.3.

H.1 Program

The representation of the cases in the Automobile Exception domain considers the role of facts. Instead of base factors, the Prolog program below considers the inclusion of facts in the domain knowledge base. Each case consists of two parameters: the case name; and a list of the *Facts* representing the case. A fact is represented as (ft) followed by the *base factor number* and the*fact initials* (e.g. ft011c refers to a car (fact)for the automobile (base factor-BF011)). The procedure *go(C):-case(C,Facts)* holds the case *C* and the facts of *Facts* and starts navigating between the base factors first and then abstract factors. The existence of a factor is determined by the existence of one or more of its children (facts) in the list *Facts*, according to the defined acceptance conditions in Chapter 6. If the factor is satisfied, a report describing the factor is written, and the factor is added to the list *Factors*. If the factor is determined from a precedent, the report will cite the precedent case. Every factor ends with a default procedure that reports the *default* status of the factor in the running case. After examining all the base and abstract factors and issues, the final case decision is determined in *decide(C,Issues,Factors)*.

% Cases knowledge base

```
case(cva,[ft011c,ft015pb,ft032ps,ft043b,ft051m,ft054hw,ft211pi,ft211po,ft221is,
ft232ct,ft233al,ft311is,ft312c]).
case(cvc,[ft011mh,ft015pb,ft022nc,ft031mh,ft032ps,ft051p,ft052dr,
ft053pl,ft054d,ft211pi,ft212po,ft221is,ft231all,ft233ps,ft233al,ft311is
,ft312c,ft331c,ft331as,ft332bd,ft332k]).
case(usvr,[ft011c,ft015pb,ft032ps,ft051p,ft053pl,ft211pi,ft221is,ft231all,ft232ct,
```

```
ft233al,ft233ps,ft311is,ft311m,ft312c]).
case(avs,[ft011c,ft014gc,ft015sc,ft032ap,ft043b,ft051m,ft211ao,ft221is,ft232ct,
ft311is,ft312c]).
case(usvc,[ft011c,ft014fl,ft032pl,ft043b,ft051p,ft053pl,ft211pi,ft221is,ft232ct,
ft233ps,ft311is,ft312dl]).
case(sdvo,[ft011c,ft015pb,ft043gb,ft051p,ft053pl,ft213po,ft221is,ft231all,
ft233al,ft311is]).
case(cvd,[ft011c,ft042t,ft043b,ft051c,ft053hw,ft053dw,ft213ri,ft222m,
ft231all,ft233g]).
case(cvnh,[ft011c,ft021na,ft051p,ft053dw,ft054dw,ft213ri,ft222m,ft231all]).
case(cvm,[ft011c,ft015w,ft051m,ft054hw,ft213ri,ft222r,ft231all]).
case(cvus,[ft011c,ft051m,ft211pi,ft221is,ft231all,ft233al,ft311is]).
```

% Retrieve the case factors and navigate between factors

```
go(C):-case(C,Facts),
        getAutomobile(C,Facts,[]).

% BFc-01-1 Automobile
%car(c)
getAutomobile(C,Facts,Factors):-member(ft011c,Facts), !,
        write([it,is,a,vehicle,cite,carroll,v,us]),nl,
        getLargeContainer(C,Facts,[bf011|Factors]).
%mobilehome (mh)
getAutomobile(C,Facts,Factors):-member(ft011mh,Facts), !,
        write([it,is,a,mobileHome,vehicle,cite,carney,v,california]),nl,
        getLargeContainer(C,Facts,[bf011|Factors]).
%Default-false
getAutomobile(C,Facts,Factors):-!,
        write([accepted,that,it,is,not,an,automobilecite,cite,carroll,v,us]),nl,
        getVessel(C,Facts,Factors).

% BF-01-2 Vessel
%Motorboat(mb)
getVessel(C,Facts,Factors):-member(ft012mb,Facts), !,
        write([motorboat,is,a,vessel,cite,carroll,v,us]),nl,
        getLargeContainer(C,Facts,[bf012|Factors]).
%Sailboat(sb)
getVessel(C,Facts,Factors):-member(ft012sb,Facts), !,
        write([sailboat,is,a,vessel,cite,carroll,v,us]),nl,
        getLargeContainer(C,Facts,[bf012|Factors]).
%Rowboat(rb)
getVessel(C,Facts,Factors):-member(ft012rb,Facts), !,
        write([rowboat,is,a,vessel,cite,carroll,v,us]),nl,
         getLargeContainer(C,Facts,[bf012|Factors]).
%Default-false
getVessel(C,Facts,Factors):-!,
         write([accepted,that,it,is,not,a,vessel,cite,carroll,v,us]),nl,
         getTowable(C,Facts,Factors).

% BF-01-4 Towable
%Trailer(t)
getTowable(C,Facts,Factors):-member(ft013t,Facts), !,
        write([trailer,is,towable,cite,carroll,v,us]),nl,
        getLargeContainer(C,Facts,[bf013|Factors]).
%Wagon(w)
getTowable(C,Facts,Factors):-member(ft013w,Facts), !,
        write([wagon,is,towable,cite,carroll,v,us]),nl,
        getLargeContainer(C,Facts,[bf013|Factors]).
%Cart(c)
```

```
getTowable(C,Facts,Factors):-member(ft013c,Facts), !,
        write([cart,is,towable,cite,carroll,v,us]),nl,
        getLargeContainer(C,Facts,[bf013|Factors]).
%Default-false
getTowable(C,Facts,Factors):-!,
        write([accepted,that,it,is,not,towable,cite,carroll,v,us]),nl,
        getLargeContainer(C,Facts,Factors).

% BF-01-4 LargeContainer
%Footlocker
getLargeContainer(C,Facts,Factors):-member(ft014fl,Facts), !,
        write([footlocker,is a,large,container,
        cite,us,v,chadwick]),nl,
        getMovableContainer(C,Facts,[bf014|Factors]).
%GoodsContainer
getLargeContainer(C,Facts,Factors):-member(ft014gc,Facts), !,
        write([large,goodsContainer]),nl,
        getMovableContainer(C,Facts,[bf014|Factors]).
%Default-false
getLargeContainer(C,Facts,Factors):-!,
        write([accepted,that,no,largeContainers]),nl,
         getMovableContainer(C,Facts,Factors).

% BF-01-5 MovableContainer
%Pouch
getMovableContainer(C,Facts,Factors):-member(ft015p,Facts),!,
        write([pouch,is,a,movableContainer]),nl,
        getMobile(C,Facts,[bf015|Factors]).
%Paprebag
getMovableContainer(C,Facts,Factors):-member(ft015pb,Facts),!,
        write([paper,bag,is,a,movableContainer]),nl,
        getMobile(C,Facts,[bf015|Factors]).
%Breifcase
getMovableContainer(C,Facts,Factors):-member(ft015bs,Facts),!,
        write([breifCase,is,a,movableContainer]),nl,
        getMobile(C,Facts,[bf015|Factors]).
%Suitcase
getMovableContainer(C,Facts,Factors):-member(ft015sc,Facts),!,
        write([suitcase,is,a,movableContainer]),nl,
        getMobile(C,Facts,[bf015|Factors]).
%Default-false
getMovableContainer(C,Facts,Factors):-!,
        write([accepted,that,it,is,not,a,movableContainer]),nl,
         getMobile(C,Facts,Factors).

\%AF1-01- Mobile
\%Automobile
getMobile(C,Facts,Factors):- member(bf011,Factors),!,
        write([it,is,a,mobile]),nl,
        getUrgentStatus(C,Facts,[af101|Factors]).
\%Vessel
getMobile(C,Facts,Factors):- member(bf012,Factors),!,
        write([it,is,a,mobile]),nl,
        getUrgentStatus(C,Facts,[af101|Factors]).
\%#Towable
getMobile(C,Facts,Factors):- member(bf013,Factors),!,
        write([it,is,a,mobile]),nl,
        getUrgentStatus(C,Facts,[af101|Factors]).
\%LargeContainer
```

```
getMobile(C,Facts,Factors):-member(bf014,Factors),!,
        write([it,is,a,mobile]),nl,
        getUrgentStatus(C,Facts,[af101|Factors]).
\%MovableContainer
getMobile(C,Facts,Factors):-member(bf015,Factors),!,
        write([it,is,a,mobile]),nl,
        getUrgentStatus(C,Facts,[af101|Factors]).
\%Default-not mobile
getMobile(C,Facts,Factors):-!,
        write([accepted,that,it,is,not,a,mobile]),nl,
         getLicence(C,Facts,Factors).

% BF-05-1 UrgentStatus
%Moving
getUrgentStatus(C,Facts,Factors):-member(ft051m,Facts),!,
        write([there,was,an,urgent,status,when,vehicle,is,
        moving,cite,carroll,v,us]),nl,
        getPublicLocation(C,Facts,[bf051|Factors]).
%Stationary
getUrgentStatus(C,Facts,Factors):-member(ft051s,Facts),!,
        write([there,was,no,urgent,status,automobile,found,stationary]),nl,
        getCapableToMove(C,Facts,Factors).
%Parked
getUrgentStatus(C,Facts,Factors):-member(ft051p,Facts),!,
        write([there,was,no,urgent,status,automobile,was,parked]),nl,
        getCapableToMove(C,Facts,Factors).
%Crashed
getUrgentStatus(C,Facts,Factors):-member(ft051c,Facts),!,
        write([there,was,no,urgent,status,automobile,was,crashed]),nl,
        getCapableToMove(C,Facts,Factors).
%Default-not urgent
getUrgentStatus(C,Facts,Factors):-!,
        write([accepted,that,there,is,no,urgent,status]),nl,
        getCapableToMove(C,Facts,Factors).

% BF-05-2 CapableToMove
%DriverIn
getCapableToMove(C,Facts,Factors):-member(ft052di,Facts),!,
        write([the,vehicle,is,capable,to,move]),nl,
         getPublicLocation(C,Facts,[bf052|Factors]).
%Occupied
getCapableToMove(C,Facts,Factors):-member(ft052oc,Facts),!,
        write([the,vehicle,is,capable,to,move]),nl,
        getPublicLocation(C,Facts,[bf052|Factors]).
%CurtainsOpen
getCapableToMove(C,Facts,Factors):-member(ft052co,Facts),!,
        write([the,vehicle,is,capable,to,move]),nl,
        getPublicLocation(C,Facts,[bf052|Factors]).
%MotiveForce
getCapableToMove(C,Facts,Factors):-member(ft052mf,Facts),!,
        write([the,vehicle,is,capable,to,move]),nl,
        getPublicLocation(C,Facts,[bf052|Factors]).
%Default: True
getCapableToMove(C,Facts,Factors):-!,
        write([accepted,that,the,vehicle,is,capable,to,move]),nl,
         getPublicLocation(C,Facts,[bf052|Factors]).

% BF-05-3 PublicParking
%Highway
```

```
getPublicParking(C,Facts,Factors):-member(ft053hw,Facts),!,
        write([the,vehicle,was,parked,in,public,parking]),nl,
        getPermittedDuration(C,Facts,[bf053|Factors]).
%Dwelling
getPublicParking(C,Facts,Factors):-member(ft053dw,Facts),!,
         write([the,vehicle,was,parked,in,private,parking]),nl,
         getPermittedDuration(C,Facts,Factors).
%Parkinglot
getPublicParking(C,Facts,Factors):-member(ft053pl,Facts),!,
        write([the,vehicle,was,parked,in,public,parking]),nl,
        getPermittedDuration(C,Facts,[bf053|Factors]).
%ownland
getPublicParking(C,Facts,Factors):-member(ft053ol,Facts),!,
         write([the,vehicle,was,parked,in,private,parking]),nl,
        getPermittedDuration(C,Facts,Factors).
%work
getPublicParking(C,Facts,Factors):-member(ft053w,Facts),!,
        write([the,vehicle,was,parked,in,private,parking]),nl,
        getPermittedDuration(C,Facts,Factors).
%rentedland
getPublicParking(C,Facts,Factors):-member(ft053r,Facts),!,
        write([the,vehicle,was,parked,in,private,parking]),nl,
        getPermittedDuration(C,Facts,Factors).
%Default: if moving then not parked
getPublicParking(C,Facts,Factors):-member(ft051m,Facts),!,
         write([accepted,that,vehicle,was,not,parked]),nl,
        getExigencyWhenApproached(C,Facts,Factors).
%Default: false
getPublicParking(C,Facts,Factors):-!,
        write([accepted,that,vehicle,parking,type,is,not,specified]),nl,
        getPermittedDuration(C,Facts,Factors).

% BF-05-4 PublicLocation
%Highway
getPublicLocation(C,Facts,Factors):-member(ft054hw,Facts),!,
        write([the,vehicle,was,in,public,location]),nl,
        getPublicParking(C,Facts,[bf054|Factors]).
%Downtown
getPublicLocation(C,Facts,Factors):-member(ft054d,Facts),!,
         write([the,vehicle,was,in,public,location]),nl,
        getPublicParking(C,Facts,[bf054|Factors]).
%Dwelling
getPublicLocation(C,Facts,Factors):-member(ft054dw,Facts),!,
        write([the,vehicle,was,in,private,location]),nl,
        getPublicParking(C,Facts,Factors).
%urbanResidential
getPublicLocation(C,Facts,Factors):-member(ft054ur,Facts),!,
        write([the,vehicle,was,in,private,location]),nl,
        getPublicParking(C,Facts,Factors).
%Suburban
getPublicLocation(C,Facts,Factors):-member(ft054s,Facts),!,
        write([the,vehicle,was,in,private,location]),nl,
        getPublicParking(C,Facts,Factors).
%rural
getPublicLocation(C,Facts,Factors):-member(ft054r,Facts),!,
        write([the,vehicle,was,in,private,location]),nl,
        getPublicParking(C,Facts,Factors).
%Default: false
getPublicLocation(C,Facts,Factors):-!,
```

```
        write([accepted,that,vehicle,location,is,not,specified]),nl,
        getPublicParking(C,Facts,Factors).

% BF-05-5 PermittedDuration
%ShortStay(ss)
getPermittedDuration(C,Facts,Factors):-member(ft055ss,Facts),!,
        write([the,vehicle,was,parked,for,short,time]),nl,
        getExigencyWhenApproached(C,Facts,[bf055|Factors]).
%Overnight(on)
getPermittedDuration(C,Facts,Factors):-member(ft055ss,Facts),!,
        write([the,vehicle,was,parked,for,one,overnight]),nl,
        getExigencyWhenApproached(C,Facts,[bf055|Factors]).
%LongStay(ls)
getPermittedDuration(C,Facts,Factors):-member(ft055ss,Facts),!,
        write([the,vehicle,was,parked,for,over,one,night,long,period]),nl,
        getExigencyWhenApproached(C,Facts,[bf055|Factors]).
%Default-unknown
getPermittedDuration(C,Facts,Factors):-!,
        write([accepted,that,the,vehicle,was,
          parked,for,unknown,period]),nl,
        getExigencyWhenApproached(C,Facts,Factors).

%AF1-05ExigencyWhenApproached
%UrgentStatus
getExigencyWhenApproached(C,Facts,Factors):-member(bf051,Factors),!,
        write([there,was,exigency,when,approached ]),nl,
        getInformation(C,Facts,[af105|Factors]).
%CapableToMove  and  public parking
getExigencyWhenApproached(C,Facts,Factors):-member(bf052,Factors),
         member(bf053,Factors),!,
        write([there,was,exigency,when,approached ]),nl,
        getInformation(C,Facts,[af105|Factors]).
%CapableToMove  and  public location
getExigencyWhenApproached(C,Facts,Factors):-member(bf052,Factors),
         member(bf054,Factors),!,
        write([there,was,exigency,when,approached ]),nl,
        getInformation(C,Facts,[af105|Factors]).
%Default-false
getExigencyWhenApproached(C,Facts,Factors):-!,
        write([accepted,that,there,was,no,exigency,
        when,approached ]),nl,
        getInformation(C,Facts,Factors).

% BF-21-1Information
%Public Informant
getInformation(C,Facts,Factors):-member(ft211pi,Facts),!,
        write([recieved,information,from,public,informant]),nl,
        getAuthorizedOriginOfProbableCause(C,Facts,[bf211|Factors]).
%agent officer
getInformation(C,Facts,Factors):-member(ft211ao,Facts),!,
        write([recieved,information,from,agent,officer]),nl,
        getAuthorizedOriginOfProbableCause(C,Facts,[bf211|Factors]).
%Default-false
getInformation(C,Facts,Factors):-!,
        write([accepted,that,the,origin,probable,
        cause,is,not,by,information,received]),nl,
        getObservation(C,Facts,Factors).

% BF-21-2       Observation
```

```
%Public
getObservation(C,Facts,Factors):-member(ft212po,Facts),!,
        write([observed,from,public,observer]),nl,
        getAuthorizedOriginOfProbableCause(C,Facts,[bf212|Factors]).
%agent officer
getObservation(C,Facts,Factors):-member(ft212ao,Facts),!,
        write([observed,from,agent,officer]),nl,
        getAuthorizedOriginOfProbableCause(C,Facts,[bf212|Factors]).
%Default-false
getObservation(C,Facts,Factors):-!,
        write([accepted,that,the,origin,probable,
        cause,is,not,by,observation]),nl,
        getProcedure(C,Facts,Factors).

% BF-21-3       Procedure
% incident to arrest
getProcedure(C,Facts,Factors):-member(ft213ia,Facts),!,
        write([search,incident,to,arrest,cite,harris,v,us,
        and,preston,v,us]),nl,
        getAuthorizedOriginOfProbableCause(C,Facts,[bf213|Factors]).
%multipe Parking
getProcedure(C,Facts,Factors):-member(ft213mp,Facts),!,
        write([multiple,parking,procedure]),nl,
        getAuthorizedOriginOfProbableCause(C,Facts,[bf213|Factors]).
%InspectionRegulation
getProcedure(C,Facts,Factors):-member(ft213ri,Facts),!,
        write([inspection,procedure,cite,harris,v,us,and,preston,v,us]),nl,
        getAuthorizedOriginOfProbableCause(C,Facts,[bf213|Factors]).
%Default-false
getProcedure(C,Facts,Factors):-!,
        write([accepted,that,the,origin,probable,cause,is,
        not,a,procedure,or,procedure,is,not,clarified]),nl,
        getAuthorizedOriginOfProbableCause(C,Facts,Factors).

%AF1-21AuthorizedOriginOfProbableCause
%information
getAuthorizedOriginOfProbableCause(C,Facts,Factors):-
         member(bf211,Factors),!,
        write([there,was,an,authorized,origin,of,probable,cause ]),nl,
        getPublicSafety(C,Facts,[af121|Factors]).
%observation
getAuthorizedOriginOfProbableCause(C,Facts,Factors):-
         member(bf212,Factors),!,
         write([there,was,an,authorized,origin,of,probable,cause ]),nl,
         getPublicSafety(C,Facts,[af121|Factors]).
%procedure
getAuthorizedOriginOfProbableCause(C,Facts,Factors):-
        member(bf213,Factors),!,
        write([there,was,an,authorized,origin,of,probable,cause ]),nl,
        getPublicSafety(C,Facts,[af121|Factors]).
%Default- not clarfiied or not authorized
getAuthorizedOriginOfProbableCause(C,Facts,Factors):-!,
         write([ accepted,that,origin,of,probable,cause,is,not,authorized,or,not,
         clarified ]),nl,
         getPublicSafety(C,Facts,[af121|Factors]).

% BF-22-1PublicSafety
%Weapon(w) and IllegalSubstances(is)
getPublicSafety(C,Facts,Factors):-member(ft221w,Facts),
```

```
        member(ft221is,Facts),!,
        write([main,reason,to,search,was,to,protect,the,public]),nl,
        getUrgentReasonToSearch(C,Facts,[bf221|Factors]).
%Weapon(w)
getPublicSafety(C,Facts,Factors):-member(ft221w,Facts),!,
        write([main,reason,to,search,was,to,protect,the,public,
        cite,harris,v,us,and,preston,v,us]),nl,
        getUrgentReasonToSearch(C,Facts,[bf221|Factors]).
%IllegalSubstances(is)
getPublicSafety(C,Facts,Factors):-member(ft221is,Facts),!,
        write([main,reason,to,search,was,to,protect,the,public,
        cite,carrol,v,us]),nl,
        getUrgentReasonToSearch(C,Facts,[bf221|Factors]).
%Default-true
getPublicSafety(C,Facts,Factors):-!,
        write([accepted,that,main,reason,to,
        search,was,to,protect,the,public]),nl,
        getCrime(C,Facts,[bf221|Factors]).

% BF-22-2Crime
%Smuggling
getCrime(C,Facts,Factors):-member(ft222s,Facts),!,
        write([main,reason,to,search,was,due,to,a,crime]),nl,
        getUrgentReasonToSearch(C,Facts,[bf222|Factors]).
%Dealing(d)
getCrime(C,Facts,Factors):-member(ft222d,Facts),!,
         write([main,reason,to,search,was,due,to,a,crime]),nl,
        getUrgentReasonToSearch(C,Facts,[bf222|Factors]).
%Murder(m)
getCrime(C,Facts,Factors):-member(ft222m,Facts),!,
        write([main,reason,to,search,was,due,to,a,crime]),nl,
        getUrgentReasonToSearch(C,Facts,[bf222|Factors]).
%Robbery(r)
getCrime(C,Facts,Factors):-member(ft222r,Facts),!,
         write([main,reason,to,search,was,due,to,a,crime,
         cite,chambers,v,maroney]),nl,
        getUrgentReasonToSearch(C,Facts,[bf222|Factors]).
%Default-false
getCrime(C,Facts,Factors):-!,
        write([accepted,that,main,reason,to,search,
         was,not,due,to,a,crime]),nl,
        getUrgentReasonToSearch(C,Facts,Factors).

%AF1-22UrgentReasonToSearch
%PublicSafety
getUrgentReasonToSearch(C,Facts,Factors):-member(bf221,Factors),!,
        write([the,main,reason,to,search,was,urgent ]),nl,
        getwholeVehcile(C,Facts,[af122|Factors]).
%Crime
getUrgentReasonToSearch(C,Facts,Factors):-member(bf222,Factors),!,
        write([the,main,reason,to,search,was,urgent ]),nl,
        getwholeVehcile(C,Facts,[af122|Factors]).
%Default-false
getUrgentReasonToSearch(C,Facts,Factors):-!,
        write([accepted,that,the,main,reason,to,
        immediate,search,is,not,clarified ]),nl,
         getwholeVehcile(C,Facts,Factors).
```

```
%BF-23-1wholeVehcile
%allParts(all)
getwholeVehcile(C,Facts,Factors):-member(ft231all,Facts),!,
        write([all,vehicle,parts,have,been,searched,cite,carrol,v,us,
         and,us,v,ross]),nl,
        getSearchPlace(C,Facts,[bf231|Factors]).
%Default-not clarified
getwholeVehcile(C,Facts,Factors):-!,
        write([accepted,that,it,is,not,clear,if,
         all,vehicle,parts,have,been,searched]),nl,
         getOnlyVehicleContainer(C,Facts,Factors).


%BF-23-2OnlyVehicleContainer
%CarTrunk(ct) and GloveCompartment(gc)
getOnlyVehicleContainer(C,Facts,Factors):-member(ft232ct,Facts),
         member(ft232gc,Facts),!,
        write([only,vehicle,containers,have,been,searched]),nl,
         getSearchPlace(C,Facts,[bf232|Factors]).
%CarTrunk(ct)
getOnlyVehicleContainer(C,Facts,Factors):-member(ft232ct,Facts),!,
        write([only,vehicle,containers,have,been,searched,
        cite,us,v,chadwick,and,arkansas,v,sandersl]),nl,
        getSearchPlace(C,Facts,[bf232|Factors]).
%GloveCompartment(gc)
getOnlyVehicleContainer(C,Facts,Factors):-member(ft232gc,Facts),!,
        write([only,vehicle,containers,have,been,searched]),nl,
        getSearchPlace(C,Facts,[bf232|Factors]).
%Default-not clarified
getOnlyVehicleContainer(C,Facts,Factors):-!,
        write([accepted,that,it,is,not,clear,
        which,part,of,vehicle,is,searched]),nl,
        getSearchPlace(C,Facts,Factors).

%BF-23-3LegalSearchPlace
%PoliceStation(ps) and AutomobilepublicLocation(al)
getSearchPlace(C,Facts,Factors):-member(ft233ps,Facts),
         member(ft233al,Facts),!,
        write([the,vehicle,was,searched,twice,at,the,same,automobile,location,
        and,at,police,station]),nl,
        getLegalSearchScope(C,Facts,[bf233|Factors]).
%PoliceStation(ps)
getSearchPlace(C,Facts,Factors):-member(ft233ps,Facts),!,
        write([the,vehicle,was,searched,at,police,station]),nl,
        getLegalSearchScope(C,Facts,Factors).
% Garage(g)
getSearchPlace(C,Facts,Factors):-member(ft233g,Facts),!,
        write([the,vehicle,was,searched,at,a,garage]),nl,
        getLegalSearchScope(C,Facts,Factors).
%AutomobilepublicLocation(al)
getSearchPlace(C,Facts,Factors):-member(ft233al,Facts),!,
        write([the,vehicle,was,searched,at,the,same,automobile,location]),nl,
        getLegalSearchScope(C,Facts,[bf233|Factors]).
%Default- not clarified
getSearchPlace(C,Facts,Factors):- !,
        write([accepted,that,the,vehicle,searching,
         location,is,not,clarified]),nl,
        getLegalSearchScope(C,Facts,Factors).
```

```
%AF1-23LegalSearchScope
%  WholeVehcile
getLegalSearchScope(C,Facts,Factors):-member(bf231,Factors),!,
         vwrite([the,search,scope,is,legal ]),nl,
         getProbableCauseSearchVehicle(C,Facts,[af123|Factors]).

% SearchPlace and OnlyVehicleContainer
getLegalSearchScope(C,Facts,Factors):-member(bf232,Factors),
         member(bf233,Factors),!,
         write([the,search,scope,is,illegal]),nl,
         getProbableCauseSearchVehicle(C,Facts,Factors).
%Default-False
getLegalSearchScope(C,Facts,Factors):-!,
        write([accepted,that,the,search,scope,is,illegal]),nl,
         getProbableCauseSearchVehicle(C,Facts,Factors).

%AF2-02ProbableCauseSearchVehicle
getProbableCauseSearchVehicle(C,Facts,Factors):-member(af123,Factors),
         member(af122,Factors),member(af121,Factors),!,
         write([there,is,a,probable,cause,to,search,vehicle ]),nl,
         getLicence(C,Facts,[af202|Factors]).
%urgent reason and authorized origion to search but illegal scope
getProbableCauseSearchVehicle(C,Facts,Factors):-member(af122,Factors),
          member(af121,Factors),!,
         write([there,is,a,probable,cause,to,search,vehicle,but,the,
         search,scope,was,illegal ]),nl,
         getLicence(C,Facts,Factors).
%Default-false
getProbableCauseSearchVehicle(C,Facts,Factors):-!,
         write([accepted,that,there,is,no,probable,cause,to,search,vehicle ]),nl,
         getLicence(C,Facts,Factors).

% BF-03-1Licence
%Vehicle
getLicence(C,Facts,Factors):-member(ft031vl,Facts),!,
         write([has,a,vehicle,licence]),nl,
         getRestrictedArea(C,Facts,[bf031|Factors]).
%Motorhome
getLicence(C,Facts,Factors):-member(ft031mh,Facts),!,
         write([has,a,special,motorhome,licence]),nl,
         getRestrictedArea(C,Facts,[bf031|Factors]).
%Default-if vehicle then licence
getLicence(C,Facts,Factors):-member(bf011,Factors),!,
         write([accepted,that,all,automobiles,are,registered]),nl,
         getRestrictedArea(C,Facts,[bf031|Factors]).

% BF-03-2RestrictedArea
%Airport
getRestrictedArea(C,Facts,Factors):-member(ft032ap,Facts),!,
         write([airport,is,a,restricted,area]),nl,
         getSubjectToInspection(C,Facts,[bf032|Factors]).
%Home
getRestrictedArea(C,Facts,Factors):-member(ft032h,Facts),!,
         write([private,home,is,a,restricted,area]),nl,
         getSubjectToInspection(C,Facts,[bf032|Factors]).
%PoliceStation
getRestrictedArea(C,Facts,Factors):-member(ft032ps,Facts),!,
         write([police,station,is,a,restricted,area]),nl,
         getSubjectToInspection(C,Facts,[bf032|Factors]).
```

```
%Default- false
getRestrictedArea(C,Facts,Factors):-!,
        write([accepted,that,not,restricted,area]),nl,
         getSubjectToInspection(C,Facts,Factors).

%AF1-03SubjectToInspection
%Licence and restricted area
getSubjectToInspection(C,Facts,Factors):-member(bf031,Factors),
         member(bf032,Factors),!,
         write([subject,to,regular,inspection,but,the,search,was,
         allocated,at,restricted,area]),nl,
         getOnPublicView(C,Facts,Factors).
%Licence
getSubjectToInspection(C,Facts,Factors):-member(bf031,Factors),!,
         write([subject,to,regular,inspection]),nl,
         getOnPublicView(C,Facts,[af103|Factors]).
%Default-false
getSubjectToInspection(C,Facts,Factors):-!,
        write([it,is,not,subject,to,regular,inspection]),nl,
         getOnPublicView(C,Facts,Factors).

% BF-04-1OnPublicView
%OnSeat
getOnPublicView(C,Facts,Factors):-member(ft041os,Facts),!,
         write([items,were,on,the,seat,it,is,on,public,view]),nl,
         getVisibilityOfItem(C,Facts,[bf041|Factors]).
%Default-false
getOnPublicView(C,Facts,Factors):-!,
        write([accepted,that,item,is,not,on,public,view,or,details,
         are,not,provided]),nl,
         getCanBeSeen(C,Facts,Factors).

% BF-04-2CanbeSeen
%Transparent
getCanBeSeen(C,Facts,Factors):-member(ft042t,Facts),!,
         write([items,can,be,seen,by,public]),nl,
         getVisibilityOfItem(C,Facts,[bf042|Factors]).
%OnFloor
getCanBeSeen(C,Facts,Factors):-member(ft042of,Facts),!,
         write([items,were,on,floor,it,can,be,seen,by,public]),nl,
         getVisibilityOfItem(C,Facts,[bf042|Factors]).
%Default-false
getCanBeSeen(C,Facts,Factors):-!,
        write([accepted,that,can,not,be,seen,
         by,public,or,details,are,not,provided]),nl,
         getCanNotBeSeen(C,Facts,Factors).

% BF-04-3CanNotbeSeen
%OpaqueContainer
getCanNotBeSeen(C,Facts,Factors):-member(ft043oc,Facts),!,
         write([items,were,in,an,opaque,container,it,can,not,be,
         seen,by,public]),nl,
         getVisibilityOfItem(C,Facts,[bf043|Factors]).
%GloveBox
getCanNotBeSeen(C,Facts,Factors):-member(ft043bg,Facts),!,
         write([items,were,inside,the,glove,box,it,can,not,be,
          seen,by,public]),nl,
         getVisibilityOfItem(C,Facts,[bf043|Factors]).
%Boot
```

```
getCanNotBeSeen(C,Facts,Factors):-member(ft043b,Facts),!,
         write([items,were,inside,the,boot,it,can,be,not,seen,by,public]),nl,
         getVisibilityOfItem(C,Facts,[bf043|Factors]).
%Default-false
getCanNotBeSeen(C,Facts,Factors):-!,
        write([accepted,that,it,is,not,clear,
         that,items,can,not,be,seen]),nl,
         getVisibilityOfItem(C,Facts,Factors).

%AF1-04VisibilityOfItem
getVisibilityOfItem(C,Facts,Factors):-member(bf041,Factors),!,
         write([item,is,visible,to,public]),nl,
         getGoodsCarried(C,Facts,[af104|Factors]).
getVisibilityOfItem(C,Facts,Factors):-member(bf042,Factors),!,
         write([item,is,visible,to,public]),nl,
         getGoodsCarried(C,Facts,[af104|Factors]).
%Default-false
getVisibilityOfItem(C,Facts,Factors):-!,
        write([accepted,that,it,is,not,visible,to,public]),nl,
         getGoodsCarried(C,Facts,Factors).

%BF-31-1GoodsCarried
%personalEffects(pe)
getGoodsCarried(C,Facts,Factors):-member(ft311pe,Facts),!,
         write([private,goods]),nl,
         getProtectionType(C,Facts,[bf311|Factors]).
%papers(p)
getGoodsCarried(C,Facts,Factors):-member(ft311p,Facts),!,
         write([private,goods]),nl,
         getProtectionType(C,Facts,[bf311|Factors]).
%CommercialItems(ci)
getGoodsCarried(C,Facts,Factors):-member(ft311ci,Facts),!,
         write([private,goods]),nl,
         getProtectionType(C,Facts,[bf311|Factors]).
%weapons(w)
getGoodsCarried(C,Facts,Factors):-member(ft311w,Facts),!,
         write([illegal,goods]),nl,
         getProtectionType(C,Facts,Factors).
%illegalSubstances(is)
getGoodsCarried(C,Facts,Factors):-member(ft311is,Facts),!,
         write([illegal,goods]),nl,
         getProtectionType(C,Facts,Factors).
%money(m)
getGoodsCarried(C,Facts,Factors):-member(ft311m,Facts),!,
         write([private,goods]),nl,
         getProtectionType(C,Facts,[bf311|Factors]).
%Default-unknown
getGoodsCarried(C,Facts,Factors):- !,
        write([ accepted,that,goods,
          carried,are,unknown ]),nl,
         getProtectionType(C,Facts,Factors).

%BF-31-2ProtectionType
%open(o)
getProtectionType(C,Facts,Factors):-member(ft312o,Facts),!,
         write([not,protected]),nl,
         getPrivateContentsCarriage(C,Facts,Factors).
%closed(c)
getProtectionType(C,Facts,Factors):-member(ft312c,Facts),!,
```

```
         write([just,closed,but,not,protected]),nl,
         getPrivateContentsCarriage(C,Facts,Factors).
%locked(l)
getProtectionType(C,Facts,Factors):-member(ft312l,Facts),!,
         write([locked,and,protected]),nl,
         getPrivateContentsCarriage(C,Facts,[bf312|Factors]).
%doublelocked(dl)
getProtectionType(C,Facts,Factors):-member(ft312dl,Facts),!,
         write([double,locked,and,protected]),nl,
         getPrivateContentsCarriage(C,Facts,[bf312|Factors]).
%Default- not clarified
getProtectionType(C,Facts,Factors):-!,
        write([accepted,that,protection,
         level,can,not,be,determined]),nl,
         getPrivateContentsCarriage(C,Facts,Factors).

%AF1-3-1PrivateContentsCarriage
%GoodsCarried and protected
getPrivateContentsCarriage(C,Facts,Factors):-member(bf312,Factors),
        member(bf311,Factors),!,
        write([private,contents]),nl,
        getConnectedServices(C,Facts,[af131|Factors]).
%GoodsCarried and not protected
getPrivateContentsCarriage(C,Facts,Factors):-member(bf311,Factors),!,
        write([private,contents,but,not,protected]),nl,
        getConnectedServices(C,Facts,Factors).
%Default-not private
getPrivateContentsCarriage(C,Facts,Factors):-!,
        write([accepted,that,contents,are,not,
        considered,private]),nl,
        getConnectedServices(C,Facts,Factors).

%BF-32-1ConnectedServices
%Gas(g) and water
getConnectedServices(C,Facts,Factors):-member(ft321g,Facts),
        member(ft321w,Facts),!,
        write([gas,and,water,services,were,connected]),nl,
        getResidence(C,Facts,[bf321|Factors]).
%Electricity(e)and water
getConnectedServices(C,Facts,Factors):-member(ft321e,Facts),
         member(ft321w,Facts),!,
        write([electtricity,and,water,services,were,connected]),nl,
        getResidence(C,Facts,[bf321|Factors]).
%Gas(g)
getConnectedServices(C,Facts,Factors):-member(ft321g,Facts),!,
        write([gas,service,was,connected]),nl,
        getResidence(C,Facts,[bf321|Factors]).
%Electricity(e)
getConnectedServices(C,Facts,Factors):-member(ft321e,Facts),!,
        write([electricity,service,was,connected]),nl,
        getResidence(C,Facts,[bf321|Factors]).
%Water(w)
getConnectedServices(C,Facts,Factors):-member(ft321w,Facts),!,
        write([water,service,was,connected]),nl,
        getResidence(C,Facts,[bf321|Factors]).
%default-not specified or not connected
getConnectedServices(C,Facts,Factors):-!,
        write([accepted,that,none,of,
         living,main,services,are,specified,or,connected]),nl,
```

```
        getResidence(C,Facts,Factors).

%AF1-3-2Residence
getResidence(C,Facts,Factors):-member(bf321,Factors),!,
        write([connected,to,one,or,more,main,living,services]),nl,
        getAccommodationSpaces(C,Facts,[af132|Factors]).
%default-false
getResidence(C,Facts,Factors):-!,
        write([accepted,that,not,connected,to,
         one,or,more,main,living,services]),nl,
        getAccommodationSpaces(C,Facts,Factors).

%BF-33-1AccommodationSpaces
%cab(c) and accommodation space
getAccommodationSpaces(C,Facts,Factors):-member(ft331c,Facts),
         member(ft331as,Facts),!,
        write([consists,of,a,cab,and,suitable,
        accommodation,space]),nl,
        getAccommodation(C,Facts,[bf331|Factors]).
%cab(c)
getAccommodationSpaces(C,Facts,Factors):-member(ft331c,Facts),!,
        write([consists,of,a,cab,only]),nl,
        getAccommodation(C,Facts,Factors).
%SuitableAccommodationSpace(as)
getAccommodationSpaces(C,Facts,Factors):-member(ft331as,Facts),!,
        write([consists,of,suitable,accommodation,space]),nl,
        getAccommodation(C,Facts,[bf331|Factors]).
%default-not clarified
getAccommodationSpaces(C,Facts,Factors):-!,
        write([accepted,that,vehicle,accommodation,
        spaces,are,not,clarified]),nl,
        getRoomsFunction(C,Facts,Factors).

%BF-33-2RoomsFunction
%bedroom(bd)
getRoomsFunction(C,Facts,Factors):-member(ft332bd,Facts),!,
        write([essential,room,for,accommodation]),nl,
        getAccommodation(C,Facts,[bf332|Factors]).
%bathroom(b)
getRoomsFunction(C,Facts,Factors):-member(ft332b,Facts),!,
        write([essential,room,for,accommodation]),nl,
        getAccommodation(C,Facts,[bf332|Factors]).
%Kitchen(k)
getRoomsFunction(C,Facts,Factors):-member(ft332k,Facts),!,
        write([essential,room,for,accommodation]),nl,
        getAccommodation(C,Facts,[bf332|Factors]).
%Living room(lr)
getRoomsFunction(C,Facts,Factors):-member(ft332lv,Facts),!,
        write([essential,room,for,accommodation]),nl,
        getAccommodation(C,Facts,[bf332|Factors]).
%default-not specified or no rooms
getRoomsFunction(C,Facts,Factors):-!,
        write([accepted,that,there,are,no,rooms,or,
        rooms,function,is,not,specified]),nl,
        getAccommodation(C,Facts,Factors).

%AF1-3-3Accommodation
%Accommodation
getAccommodation(C,Facts,Factors):-member(bf331,Factors),!,
```

```
        write([the,place,was,used,for,accommodation]),nl,
        getPrivacyRights(C,Facts,[af133|Factors]).
%RoomsFunction
getAccommodation(C,Facts,Factors):-member(bf332,Factors),!,
        write([the,place,was,used,for,accommodation]),nl,
        getPrivacyRights(C,Facts,[af133|Factors]).
%default-false
getAccommodation(C,Facts,Factors):-!,write([accepted,that,the,place,is,not,
         used,for,accommodation]),nl,
        getPrivacyRights(C,Facts,Factors).

%AF2-0-3ExpectationOfPrivacyInUseRequired-PrivacyRights
%residence and accommodation
getPrivacyRights(C,Facts,Factors):-member(af133,Factors),
        member(af132,Factors),!,
        write([the,is,a,high,expectation,of,privacy,in,use ]),nl,
        getRiskLosingEvidence(C,Facts,[af203|Factors]).
%privateContents
getPrivacyRights(C,Facts,Factors):- member(af131,Factors),!,
        write([the,is,a,high,expectation,of,privacy,in,use ]),nl,
        getRiskLosingEvidence(C,Facts,[af203|Factors]).
%default-false
getPrivacyRights(C,Facts,Factors):-
        write([default,low,expectation,of,privacy,in,use ]),nl,
        getRiskLosingEvidence(C,Facts,Factors).

% BF-02-2 RiskLostingEvidence
%Default-if Exigency when approached
getRiskLosingEvidence(C,Facts,Factors):-member(af105,Factors),!,
        write([there,is,risk,to,lose,evidence]),nl,
        getAvailabilityOfMagistrate(C,Facts,[bf022|Factors]).
%nearCourt
getRiskLosingEvidence(C,Facts,Factors):-member(ft022nc,Facts),!,
        write([there,was,no,risk,to,lose,evidence]),nl,
        getAvailabilityOfMagistrate(C,Facts,Factors).
%default
getRiskLosingEvidence(C,Facts,Factors):-!,write([accepted,that,there,was,
        no,risk,to,lose,evidence]),nl,
        getAvailabilityOfMagistrate(C,Facts,Factors).
% BF-02-3 AvailabilityofMagistrate
%WorkingTime
getAvailabilityOfMagistrate(C,Facts,Factors):-member(ft023wt,Facts),!,
        write([magistrate,available,during,working,hours]),nl,
        getAuthority(C,Facts,[bf023|Factors]).
%Overnight
getAvailabilityOfMagistrate(C,Facts,Factors):-member(ft023on,Facts),!,
        write([magistrate,are,not,available,overnight]),nl,
        getEaseObtainingWarrant(C,Facts,Factors).
%Default-False
getAvailabilityOfMagistrate(C,Facts,Factors):-!,
        write([accepted,that,magistrate,are,not,available]),nl,
        getAuthority(C,Facts,Factors).

% BF-02-1AuthorityofAvailableMagistrate
%authorized
getAuthority(C,Facts,Factors):-member(ft021a,Facts),!,
        write([neutral,and,detached,authorized,magistrate,are,available,
        cite,johnson,v,us]),nl,
        getEaseObtainingWarrant(C,Facts,[bf021|Factors]).
```

```
%notauthorized
getAuthority(C,Facts,Factors):-member(ft021na,Facts),!,
        write([warrant,issued,by,unauthorized,magistrate,cite,johnson,v,us]),nl,
        getEaseObtainingWarrant(C,Facts,Factors).
%Default -false
getAuthority(C,Facts,Factors):-!,
        write([accepted,that,authorized,magistrate,are,not,available]),nl,
        getEaseObtainingWarrant(C,Facts,Factors).

%AF1-02EaseObtainingWarrant
%AuthorityofAvailableMagistrate
getEaseObtainingWarrant(C,Facts,Factors):-member(bf021,Factors),
        member(bf023,Factors),!,
        write([it,is,easy,to,obtain,warrant]),nl,
        getExigency(C,Facts,[af102|Factors],[]).
% Default-if Exigency when approached then not easy to obtain warrant
% (Risk of losing Evidence)
getEaseObtainingWarrant(C,Facts,Factors):-member(bf022,Factors),!,
        write([it,is,not,easy,to,obtain,warrant]),nl,
        getExigency(C,Facts,Factors,[]).
```

% Generate the case decision after determining the Exigency and Privacy issues

```
% I1 Exigency
%Mobile and ExigencyWhenApproached and ProbableCauseToSearch
getExigency(C,Facts,Factors,Issues):-member(af101,Factors),
        member(af105,Factors),
        member(af202,Factors),!,
        write([justified,under,automobile,exception,cite,carroll,v,us]),nl,
        getPrivacy(C,Facts,Factors,[is1|Issues]).
%Mobile and ExigencyWhenApproached and
ProbableCauseToSearch and accommodation
getExigency(C,Facts,Factors,Issues):-member(af101,Factors),
        member(af133,Factors),!,
        write([reduce,expectation,of,exigency]),nl,
        getPrivacy(C,Facts,Factors,Issues).
%Easy to obtain warrant
getExigency(C,Facts,Factors,Issues):- member(af102,Factors),!,
        write([reduce,expectation,of,exigency]),nl,
        getPrivacy(C,Facts,Factors,Issues).
%Default-false
getExigency(C,Facts,Factors,Issues):- !,
        write([reduce,expectation,of,exigency]),nl,
        getPrivacy(C,Facts,Factors,Issues).

% I2 Privacy
%EnoughExpectationOfPrivacyInUse and (no InspectionRegulation)
getPrivacy(C,Facts,Factors,Issues):-member(af203,Factors),
        not(member(af103,Factors)),!,
        write([high,expectation,of,privacy,not,justified,under,
        automobile,exception]),nl,
        decide(C,[is2|Issues],Factors).
%EnoughExpectationOfPrivacyInUse and (not itemsVisibility)
getPrivacy(C,Facts,Factors,Issues):-member(af203,Factors),
        not(member(af104,Factors)),!,
        write([high,expectation,of,privacy,not,justified,under,
        automobile,exception]),nl,
        decide(C,[is2|Issues],Factors).
%AuthorityofAvailableMagistrate:
%Private location and warrant not authorized
```

```
getPrivacy(C,Facts,Factors,Issues):-not(member(bf053,Factors)),
         member(ft021na,Facts),!,
        write([high,expectation,of,privacy,obtained,warrant,was,issued,by,
         neutral,and,detached,magistrate,and,not,authorized]),nl,
        decide(C,[is2|Issues],Factors).
%Searching only container and (not items Visibility)
getPrivacy(C,Facts,Factors,Issues):-member(bf232,Factors),
        not(member(af104,Factors)),!,
        write([praivacy,is,not,justified,under,automobile,exception]),nl,
        decide(C,[is2|Issues],Factors).
%Default-false
getPrivacy(C,Facts,Factors,Issues):- !,
        write([reduced,expectation,of,privacy]),nl,
        decide(C,Issues,Factors).

% Decide Majority
decide(C,Issues,Factors):-member(is2,Issues),!,
        write([warrantless,search,violates,the,fourth,amendment]),nl,
         write([find,for,the,defendant]).
         gomin(C,Factors).

decide(C,Issues,Factors):-member(is1,Issues),!,
        write([warrantless,search,did,not,violate,the,fourth,amendment]),
        nl,write([find,for,the,plaintiff]).
        gomin(C,Factors).

decide(C,Issues,Factors):-write([wrong,decision]).
```

H.2 Dissenting Opinion Rules

The program presented above has been amended slightly to include dissenting opinions. New rules have been added for some factors, mainly for the issues to consider the different acceptance conditions for 5 cases in the domain as discussed in Chapter 6. After displaying the majority opinion, the program continues to evaluate the factors and issues according to the new rules, and decides the dissenting opinion in *decideMin*.

% Generate the case decision after determining the Exigency and Privacy issues

```
gomin(C,Factors):-case(C,Facts),
        write([********************]),nl,
        write([dissenting,argument]),nl,
             getAutomobileMin(C,Facts,Factors).


% BF-01-1 Automobile
%car(c)
getAutomobileMin(C,Facts,Factors):-member(ft011c,Facts),!,
         write([it,is,a,vehicle]),nl,
         getLargeContainerMin(C,Facts,[bf011|Factors]).
%mobilehome(mh)
getAutomobileMin(C,Facts,Factors):-member(ft011mh,Facts),!,
         write([it,is,a,mobileHome,vehicle]),nl,
         getLargeContainerMin(C,Facts,[bf011|Factors]).
%Default-false
getAutomobileMin(C,Facts,Factors):-!,
        write([accepted,that,it,is,not,an,automobile]),nl,
```

```
         getLargeContainerMin(C,Facts,Factors).


% BF-01-4 LargeContainer
%Footlocker
getLargeContainerMin(C,Facts,Factors):-member(ft014fl,Facts),!,
         write([footlocker,is,a,large,container]),nl,
         getMovableContainerMin(C,Facts,[bf014|Factors]).
%GoodsContainer
getLargeContainerMin(C,Facts,Factors):-member(ft014gc,Facts),!,
         write([large,goodsContainer]),nl,
         getMovableContainerMin(C,Facts,[bf014|Factors]).
%Default-false
getLargeContainerMin(C,Facts,Factors):-!,
        write([accepted,that,no,largeContainers]),nl,
         getMovableContainerMin(C,Facts,Factors).


% BF-01-5 MovableContainer
%Pouch
getMovableContainerMin(C,Facts,Factors):-member(ft015p,Facts),!,
         write([pouch,is,a,movableContainer]),nl,
         getMobileMin(C,Facts,[bf015|Factors]).
%Paprebag
getMovableContainerMin(C,Facts,Factors):-member(ft015pb,Facts),!,
         write([paper,bag,is,a,movableContainer]),nl,
         getMobileMin(C,Facts,[bf015|Factors]).
%Breifcase
getMovableContainerMin(C,Facts,Factors):-member(ft015bs,Facts),!,
         write([breifCase,is,a,movableContainer]),nl,
         getMobileMin(C,Facts,[bf015|Factors]).
%Suitcase
getMovableContainerMin(C,Facts,Factors):-member(ft015sc,Facts),!,
         write([suitcase,is,a,movableContainer]),nl,
         getMobileMin(C,Facts,[bf015|Factors]).
%Default-false
getMovableContainerMin(C,Facts,Factors):-!,
         write([accepted,that,it,is,not,a, movableContainer]),nl,
         getMobileMin(C,Facts,Factors).


%AF1-01- Mobile
%Mobile home ( this can be added as a base factor)
getMobileMin(C,Facts,Factors):- member(ft011mh,Facts),!,
         write([it,is,a,hybrid,with,mobility,features]),nl,
         getCapableToMoveMin(C,Facts,[af101|Factors]).
%Automobile
getMobileMin(C,Facts,Factors):- member(bf011,Factors),
         write([it,is,a,mobile]),nl,
         getCapableToMoveMin(C,Facts,[af101|Factors]).
%LargeContainer
getMobileMin(C,Facts,Factors):-member(bf014,Factors),
         write([any,large,container,is,a,mobile,but,not,like,
         automobile]),nl,
         getCapableToMoveMin(C,Facts,[af101|Factors]).
%MovableContainer
getMobileMin(C,Facts,Factors):-member(bf015,Factors),!,
         write([any,movable,container,is,a,mobile,but,not,like,
         automobile]),nl,
```

```
         getCapableToMoveMin(C,Facts,[af101|Factors]).
%Default-not mobile
getMobileMin(C,Facts,Factors):-!,write([accepted,that,it,is,not,
          a,mobile]),nl,
         getCapableToMoveMin(C,Facts,Factors).


% BF-05-2 CapableToMove
%if crashed then not capable to move
getCapableToMoveMin(C,Facts,Factors):-member(ft051c,Facts),!,
         write([the,vehicle,is,crashed,and,not,capable,to,move]),nl,
         deleteOne(Factors,bf052,DFactors),
         getExigencyWhenApproachedMin(C,Facts,DFactors).
getCapableToMoveMin(C,Facts,Factors):-!,write([accepted,that,the,
          vehicle,is,capable,to,move]),nl,
         getExigencyWhenApproachedMin(C,Facts,Factors).


%AF1-05 ExigencyWhenApproached
%UrgentStatus
getExigencyWhenApproachedMin(C,Facts,Factors):-
         member(bf051,Factors),!,
         write([there,was,exigency,when,approached]),nl,
         getRiskLosingEvidenceMin(C,Facts,[af105|Factors]).
%CapableToMove  and  public parking
getExigencyWhenApproachedMin(C,Facts,Factors):-
         member(bf052,Factors),
         member(bf053,Factors),!,
         write([there,was,exigency,when,approached]),nl,
         getRiskLosingEvidenceMin(C,Facts,[af105|Factors]).
%CapableToMove  and  public location
getExigencyWhenApproachedMin(C,Facts,Factors):-
         member(bf052,Factors),
         member(bf054,Factors),!,
         write([there,was,exigency,when,approached]),nl,
         getRiskLosingEvidenceMin(C,Facts,[af105|Factors]).
%Default-false
getExigencyWhenApproachedMin(C,Facts,Factors):-!,
        write([accepted,that,there,was,no,exigency,when,approached ]),nl,
         deleteOne(Factors,af105,DFactors),
         getRiskLosingEvidenceMin(C,Facts,DFactors).


% BF-02-2 RiskLostingEvidence
%if Exigency when approached
getRiskLosingEvidenceMin(C,Facts,Factors):-member(af105,Factors),!,
         write([there,is,risk,to,lose,evidence]),nl,
         getEaseObtainingWarrantMin(C,Facts,[bf022|Factors]).
%if no Exigency when approached
getRiskLosingEvidenceMin(C,Facts,Factors):-!,
         write([there,is,no,risk,to,lose,evidence]),nl,
         deleteOne(Factors,bf022,DFactors),
         getEaseObtainingWarrantMin(C,Facts,DFactors).


%AF1-02 EaseObtainingWarrant
%AuthorityofAvailableMagistrate
getEaseObtainingWarrantMin(C,Facts,Factors):-
         member(bf021,Factors),
```

```
          member(bf023,Factors),!,
         write([it,is,easy,to,obtain,warrant]),nl,
         getExigencyMin(C,Facts,[af102|Factors],[]).
% if Exigency when approached then not easy to obtain warrant
% (Risk of losing Evidence)
getEaseObtainingWarrantMin(C,Facts,Factors):-
         member(bf022,Factors),!,
         write([it,is,not,easy,to,obtain,warrant]),nl,
         getExigencyMin(C,Facts,Factors,[]).
%Default
getEaseObtainingWarrantMin(C,Facts,Factors):-!,
         write([it,is,easy,to,obtain,warrant]),nl,
         getExigencyMin(C,Facts,[af102|Factors],[]).


% I1 ExigencyMin
%Mobile and ExigencyWhenApproached and ProbableCauseToSearch
and no accommodation
getExigencyMin(C,Facts,Factors,Issues):-member(af101,Factors),
         member(af133,Factors),!,
         write([reduce,expectation,of,exigency]),nl,
         getPrivacyMin(C,Facts,Factors,Issues).

%Mobile and ExigencyWhenApproached and ProbableCauseToSearch
getExigencyMin(C,Facts,Factors,Issues):-member(af101,Factors),
         member(af105,Factors),
         member(af202,Factors),!,
         write([justified,under,automobile,exception]),nl,
         getPrivacyMin(C,Facts,Factors,[is1|Issues]).

%easy to obtain warrant
getExigencyMin(C,Facts,Factors,Issues):- member(af102,Factors),!,
         write([reduce,expectation,of,exigency]),nl,
         getPrivacyMin(C,Facts,Factors,Issues).

%default-false
getExigencyMin(C,Facts,Factors,Issues):- !,
        write([defaut,reduce,expectation,of,exigency]),nl,
         getPrivacyMin(C,Facts,Factors,Issues).

% I2 Privacy
% Mobile(mobilehome) and Accommodation gives priority to privacy
 getPrivacyMin(C,Facts,Factors,Issues):-member(ft011mh,Facts),
         member(af133,Factors),!,
         write([hybrid,with,mobility,and,home,features,shows,
         enough,expectation,of,privacy]),nl,
         decideMin(C,[is2|Issues],Factors).

% Easy to obtain warrant
 getPrivacyMin(C,Facts,Factors,Issues):-member(af102,Factors),!,
         write([privacy,required,it,is,not,justified,under,
         automobile,exception]),nl,
         decideMin(C,[is2|Issues],Factors).

% searching large or movable container inside automobile without
% probable cause assessment by neutral and deatached magistrate
getPrivacyMin(C,Facts,Factors,Issues):-member(bf014,Factors),
         not(member(bf021,Factors)),!,
         write([privacy,required,probable,cause,assessment,for,container,
```

```
        should,be,done,by,neutral,and,detached,magistrate]),nl,
        decideMin(C,[is2|Issues],Factors).

% searching large or movable container inside automobile without
%  assessment by neutral and detached magistrate
getPrivacyMin(C,Facts,Factors,Issues):-member(bf015,Factors),
        not(member(bf021,Factors)),!,
        write([privacy,required,probable,cause,assessment,for,
        container,should,be,done,by,neutral,and,detached,magistrate]),nl,
        decideMin(C,[is2|Issues],Factors).

%large Container inside automobile adds privacy interest->not mobile
 getPrivacyMin(C,Facts,Factors,Issues):-
       member(bf011,Factors),member(bf014,Factors),!,
        write([enlarge,scope,of,fourth,amendment,exception]),nl,
        write([large,container,inside,automobile,shows,enough,
        privacy,interest]),nl,
        decideMin(C,[is2|Issues],Factors).

%movable Container inside automobile
 getPrivacyMin(C,Facts,Factors,Issues):-member(bf011,Factors),
        member(bf015,Factors),!,
        write([enlarge,scope,of,fourth,amendment,exception]),nl,
        write([any,container,inside,automobile,shows,enough,
        privacy,interest]),nl,
        decideMin(C,[is2|Issues],Factors).

% searching large or movable container inside automobile without
% probable cause assessment by neutral and detached magistrate
getPrivacyMin(C,Facts,Factors,Issues):-member(bf014,Factors),
        not(member(bf021,Factors)),!,
        write([privacy,required,probable,cause,assessment,for,container,
        should,be,done,by,neutral,and,detached,magistrate]),nl,
        decideMin(C,[is2|Issues],Factors).

%EnoughExpectationOfPrivacyInUse and (no InspectionRegulation)
getPrivacyMin(C,Facts,Factors,Issues):-member(af203,Factors),
        not(member(af103,Factors)),!,
        write([high,expectation,of,privacy,not,justified,under,automobile,
        exception]),nl,
        decideMin(C,[is2|Issues],Factors).

%EnoughExpectationOfPrivacyInUse and (not itemsVisibility)
getPrivacyMin(C,Facts,Factors,Issues):-member(af203,Factors),
        not(member(af104,Factors)),!,
        write([high,expectation,of,privacy,not,justified,under,automobile,
        exception]),nl,
        decideMin(C,[is2|Issues],Factors).

%AuthorityofAvailableMagistrate: Private location and warrant not authorised
getPrivacyMin(C,Facts,Factors,Issues):-not(member(bf053,Factors)),
        member(ft021na,Facts),!,
        write([high,expectation,of,privacy,obtained,warrant,was,issued,by,
         neutral,and,detached,magistrate,and,not,authorised]),nl,
        decideMin(C,[is2|Issues],Factors).

%Searching only container and (not itemsVisibility)
getPrivacyMin(C,Facts,Factors,Issues):-member(bf232,Factors),
        not(member(af104,Factors)),!,
```

```
          write([privacy,required,it,is,not,justified,under,automobile,
          exception]),nl,
          decideMin(C,[is2|Issues],Factors).
%default-false
getPrivacyMin(C,Facts,Factors,Issues):- !,
         write([accepted,that,reduced,expectation,of,privacy]),nl,
          decideMin(C,Issues,Factors).


% Decide Minority
decideMin(C,Issues,Factors):-member(is2,Issues),!,
          write([warrantless,search,violates,the,fourth,amendment]).

decideMin(C,Issues,Factors):-member(is1,Issues),!,
          write([warrantless,search,did,not,violate,the,fourth,amendment]).


decideMin(C,Issues,Factors):-write([wrong,minority,decision]).


deleteOne([],_,[]).

deleteOne([H|Tail],H,Tail).

deleteOne([H|Tail],Y,[H|Z]):-
   deleteOne(Tail,Y,Z).
```

H.3 Results

To obtain the case decision, the user calls the case using *go(case name)*. The output from each case provides the decision justification on the basis of the base factors and abstract factor reports, in addition to the case decision: *find for plaintiff*,or, *find for defendant*. Considering a case represented as facts clearly provides more explanation of the case opinion compared to the output from the previous domains (see Appendix F, Appendix G) as discussed in Chapter 7. The list below illustrates the output from the 10 cases including the majority and dissenting opinions (five cases). As stated in Chapter 6, the output from the incorrect case, *Acevedo*, clarifies where the decision diverges from the actual decision when citing the *Chadwick* and *Sanders* cases.

California v. Acevedo

```
go(cva).
[it,is,a,vehicle,cite,carroll,v,us]
[accepted,that,no,largeContainers]
[paper,bag,is,a,movableContainer]
[it,is,a,mobile]
[there,was,an,urgent,status,when,vehicle,is,moving,cite,
carroll,v,us]
[the,vehicle,was,in,public,location]
[accepted,that,vehicle,was,not,parked]
```

```
[there,was,exigency,when,approached]
[received,information,from,public,informant]
[there,was,an,authorized,origin,of,probable,cause]
[main,reason,to,search,was,to,protect,the,public,
cite,carrol,v,us]
[the,main,reason,to,search,was,urgent]
[accepted,that,it,is,not,clear,if,all,vehicle,parts,
have,been,searched]
[only,vehicle,containers,have,been,searched,cite,
us,v,chadwick,and,arkansas,v,sandersl]
[the,vehicle,was,searched,at,the,same,automobile,
location]
[the,search,scope,is,illegal]
[there,is,a,probable,cause,to,search,vehicle,but,the,
search,scope,was
,illegal]
[accepted,that,all,automobiles,are,registered]
[police,station,is,a,restricted,area]
[subject,to,regular,inspection,but,the,search,was,allocated,
at,restricted,area]
[accepted,that,item,is,not,on,public,view,or,details,are,
not,provided]
[accepted,that,can,not,be,seen,by,public,or,details,are,
not,provided]
[items,were,inside,the,boot,it,can,be,not,seen,by,public]
[accepted,that,it,is,not,visible,to,public]
[illegal,goods]
[just,closed,but,not,protected]
[accepted,that,contents,are,not,considered,private]
[accepted,that,none,of,living,main,services,are,specified,
or,connected]
[accepted,that,not,connected,to,one,or,more,main,living,
services]
[accepted,that,vehicle,accommodation,spaces,are,not,
clarified]
[accepted,that,there,are,no,rooms,or,rooms,function,is,
not,specified]
[accepted,that,the,place,is,not,used,for,accommodation]
[accepted,that,low,expectation,of,privacy,in,use]
[there,is,risk,to,lose,evidence]
[accepted,that,magistrate,are,not,available]
```

```
[accepted,that,authorized,magistrate,are,not,available]
[it,is,not,easy,to,obtain,warrant]
[reduce,expectation,of,exigency]
[privacy,is,not,justified,under,automobile,exception]
[warrantless,search,violates,the,fourth,amendment]
[find,for,the,defendant]
[*******************]
[dissenting,argument]
[it,is,a,vehicle]
[accepted,that,no,largeContainers]
[paper,bag,is,a,movableContainer]
[it,is,a,mobile]
[accepted,that,the,vehicle,is,capable,to,move]
[there,was,exigency,when,approached]
[there,is,risk,to,lose,evidence]
[it,is,not,easy,to,obtain,warrant]
[accepted,that,reduce,expectation,of,exigency]
[enlarge,scope,of,fourth,amendment,exception]
[any,container,inside,automobile,shows,enough,privacy,
interest]
[warrantless,search,violates,the,fourth,amendment]
true .
```

California v. Carny

```
go(cvc).
[it,is,a,mobileHome,vehicle,cite,carny,v,california]
[accepted,that,no,largeContainers]
[paper,bag,is,a,movableContainer]
[it,is,a,mobile]
[there,was,no,urgent,status,automobile,was,parked]
[accepted,that,the,vehicle,is,capable,to,move]
[the,vehicle,was,in,public,location]
[the,vehicle,was,parked,in,public,parking]
[accepted,that,the,vehicle,was,parked,for,unknown,period]
[there,was,exigency,when,approached]
[received,information,from,public,informant]
[there,was,an,authorized,origin,of,probable,cause]
[main,reason,to,search,was,to,protect,the,public,cite,
carrol,v,us]
[the,main,reason,to,search,was,urgent]
[all,vehicle,parts,have,been,searched,cite,carrol,v,us,
```

```
and,us,v,ross]
[the,vehicle,was,searched,twice,at,the,same,automobile,
location,and, at,police,station]
[the,search,scope,is,legal]
[there,is,a,probable,cause,to,search,vehicle]
[has,a,special,motorhome,licence]
[police,station,is,a,restricted,area]
[subject,to,regular,inspection,but,the,search,was,allocated,
at,restricted,area]
[accepted,that,item,is,not,on,public,view,or,details,are,
not,provided]
[accepted,that,can,not,be,seen,by,public,or,details,are,
not,provided]
[accepted,that,it,is,not,clear,that,items,can,not,be,seen]
[accepted,that,it,is,not,visible,to,public]
[illegal,goods]
[just,closed,but,not,protected]
[accepted,that,contents,are,not,considered,private]
[accepted,that,none,of,living,main,services,are,specified,
or,connected]
[accepted,that,not,connected,to,one,or,more,main,living,
services]
[consists,of,a,cab,and,suitable,accommodation,space]
[the,place,was,used,for,accommodation]
[accepted,that,low,expectation,of,privacy,in,use]
[there,is,risk,to,lose,evidence]
[accepted,that,magistrate,are,not,available]
[accepted,that,authorized,magistrate,are,not,available]
[it,is,not,easy,to,obtain,warrant]
[justified,under,automobile,exception,cite,carroll,v,us]
[reduced,expectation,of,privacy]
[warrantless,search,did,not,violate,the,fourth,amendment]
[find,for,the,plaintiff]
[******************]
[dissenting,argument]
[it,is,a,mobileHome,vehicle]
[accepted,that,no,largeContainers]
[paper,bag,is,a,movableContainer]
[it,is,a,hybrid,with,mobility,features]
[accepted,that,the,vehicle,is,capable,to,move]
[there,was,exigency,when,approached]
```

```
[there,is,risk,to,lose,evidence]
[it,is,not,easy,to,obtain,warrant]
[reduce,expectation,of,exigency]
[hybrid,with,mobility,and,home,features,shows,enough,
expectation,of,privacy]
[warrantless,search,violates,the,fourth,amendment]
true.
```

US v. Ross

```
go(usvr).
[it,is,a,vehicle,cite,carroll,v,us]
[accepted,that,no,largeContainers]
[paper,bag,is,a,movableContainer]
[it,is,a,mobile]
[there,was,no,urgent,status,automobile,was,parked]
[accepted,that,the,vehicle,is,capable,to,move]
[accepted,that,vehicle,location,is,not,specified]
[the,vehicle,was,parked,in,public,parking]
[accepted,that,the,vehicle,was,parked,for,unknown,
period]
[there,was,exigency,when,approached]
[received,information,from,public,informant]
[there,was,an,authorized,origin,of,probable,cause]
[main,reason,to,search,was,to,protect,the,public,cite,
carrol,v,us]
[the,main,reason,to,search,was,urgent]
[all,vehicle,parts,have,been,searched,cite,carrol,v,us,
and,us,v,ross]
[the,vehicle,was,searched,twice,at,the,same,automobile,
location,and,at,police,station]
[the,search,scope,is,legal]
[there,is,a,probable,cause,to,search,vehicle]
[accepted,that,all,automobiles,are,registered]
[police,station,is,a,restricted,area]
[subject,to,regular,inspection,but,the,search,was,allocated,
at,restricted,area]
[accepted,that,item,is,not,on,public,view,or,details,are,not,
provided]
[accepted,that,can,not,be,seen,by,public,or,details,are,not,
provided]
[accepted,that,it,is,not,clear,that,items,can,not,be,seen]
```

```
[accepted,that,it,is,not,visible,to,public]
[illegal,goods]
[just,closed,but,not,protected]
[accepted,that,contents,are,not,considered,private]
[accepted,that,none,of,living,main,services,are,specified,or,
connected]
[accepted,that,not,connected,to,one,or,more,main,living,
services]
[accepted,that,vehicle,accommodation,spaces,are,not,clarified]
[accepted,that,there,are,no,rooms,or,rooms,function,is,
not,specified]
[accepted,that,the,place,is,not,used,for,accommodation]
[accepted,that,low,expectation,of,privacy,in,use]
[there,is,risk,to,lose,evidence]
[accepted,that,magistrate,are,not,available]
[accepted,that,authorized,magistrate,are,not,available]
[it,is,not,easy,to,obtain,warrant]
[justified,under,automobile,exception,cite,carroll,v,us]
[reduced,expectation,of,privacy]
[warrantless,search,did,not,violate,the,fourth,amendment]
[find,for,the,plaintiff]
[*******************]
[dissenting,argument]
[it,is,a,vehicle]
[accepted,that,no,largeContainers]
[paper,bag,is,a,movableContainer]
[it,is,a,mobile]
[accepted,that,the,vehicle,is,capable,to,move]
[there,was,exigency,when,approached]
[there,is,risk,to,lose,evidence]
[it,is,not,easy,to,obtain,warrant]
[justified,under,automobile,exception]
[privacy,required,probable,cause,assessment,for,container,
should,be,done,by,netural,and,detached,magistrate]
[warrantless,search,violates,the,fourth,amendment]
true
```

Arkansas v. Sanders

```
 go(avs).
[it,is,a,vehicle]
[large,goodsContainer]
```

```
[suitcase,is,a,movableContainer]
[it,is,a,mobile]
[there,was,an,urgent,status,when,vehicle,is,moving]
[accepted,that,vehicle,location,is,not,specified]
[accepted,that,vehicle,was,not,parked]
[there,was,exigency,when,approached]
[received,information,from,agent,officer]
[there,was,an,authorized,origin,of,probable,cause]
[main,reason,to,search,was,to,protect,the,public]
[the,main,reason,to,search,was,urgent]
[accepted,that,it,is,not,clear,if,all,vehicle,parts,have,been,
searched]
[only,vehicle,containers,have,been,searched]
[accepted,that,the,vehicle,searching,location,is,not,clarified]
[accepted,that,the,search,scope,is,illegal]
[there,is,a,probable,cause,to,search,vehicle,but,the,search,
scope,was,illegal]
[accepted,that,all,automobiles,are,registered]
[airport,is,a,restricted,area]
[subject,to,regular,inspection,but,the,search,was,allocated,
at,restricted,area]
[accepted,that,item,is,not,on,public,view,or,details,are,not,
provided]
[accepted,that,can,not,be,seen,by,public,or,details,are,not,
provided]
[items,were,inside,the,boot,it,can,be,not,seen,by,public]
[accepted,that,it,is,not,visible,to,public]
[illegal,goods]
[just,closed,but,not,protected]
[accepted,that,contents,are,not,considered,private]
[accepted,that,none,of,living,main,services,are,specified,
or,connected]
[accepted,that,not,connected,to,one,or,more,main,living,
services]
[accepted,that,vehicle,accommodation,spaces,are,not,
clarified]
[accepted,that,there,are,no,rooms,or,rooms,function,is,
not,specified]
[accepted,that,the,place,is,not,used,for,accommodation]
[accepted,that,low,expectation,of,privacy,in,use]
[accepted,that,there,is,risk,to,lose,evidence]
```

```
[accepted,that,magistrate,are,not,available]
[accepted,that,authorized,magistrate,are,not,available]
[it,is,not,easy,to,obtain,warrant]
[reduce,expectation,of,exigency]
[privacy,is,not,justified,under,automobile,exception]
[warrantless,search,violates,the,fourth,amendment]
[find,for,the,defendant]
true.
```

US v Chadwick

```
go(usvc).
[it,is,a,vehicle]
[footlocker,is,a,large,container]
[accepted,that,it,is,not,a,movableContainer]
[it,is,a,mobile]
[there,was,no,urgent,status,automobile,was,parked]
[accepted,that,the,vehicle,is,capable,to,move]
[accepted,that,vehicle,location,is,not,specified]
[the,vehicle,was,parked,in,public,parking]
[accepted,that,the,vehicle,was,parked,for,unknown,period]
[there,was,exigency,when,approached]
[received,information,from,public,informant]
[there,was,an,authorized,origin,of,probable,cause]
[main,reason,to,search,was,to,protect,the,public]
[the,main,reason,to,search,was,urgent]
[accepted,that,it,is,not,clear,if,all,vehicle,parts,have,been,
searched]
[only,vehicle,containers,have,been,searched]
[the,vehicle,was,searched,at,police,station]
[accepted,that,the,search,scope,is,illegal]
[there,is,a,probable,cause,to,search,vehicle,but,the,search,
scope,was,illegal]
[accepted,that,all,automobiles,are,registered]
[accepted,that,not,restricted,area]
[subject,to,regular,inspection]
[accepted,that,item,is,not,on,public,view,or,details,are,not,
provided]
[accepted,that,can,not,be,seen,by,public,or,details,are,not,
provided]
[items,were,inside,the,boot,it,can,be,not,seen,by,public]
[accepted,that,it,is,not,visible,to,public]
```

```
[illegal,goods]
[double,locked,and,protected]
[accepted,that,contents,are,not,considered,private]
[accepted,that,none,of,living,main,services,are,specified,
or,connected]
[accepted,that,not,connected,to,one,or,more,main,living,
services]
[accepted,that,vehicle,accommodation,spaces,are,not,
clarified]
[accepted,that,there,are,no,rooms,or,rooms,function,is,
not,specified]
[accepted,that,the,place,is,not,used,for,accommodation]
[accepted,that,low,expectation,of,privacy,in,use]
[accepted,that,there,is,risk,to,lose,evidence]
[accepted,that,magistrate,are,not,available]
[accepted,that,authorized,magistrate,are,not,available]
[it,is,not,easy,to,obtain,warrant]
[reduce,expectation,of,exigency]
[privacy,is,not,justified,under,automobile,exception]
[warrantless,search,violates,the,fourth,amendment]
[find,for,the,defendant]
true.
```

South dakota v. Opperman

```
go(sdvo).
[it,is,a,vehicle,cite,carroll,v,us]
[accepted,that,no,largeContainers]
[paper,bag,is,a,movableContainer]
[it,is,a,mobile]
[there,was,no,urgent,status,automobile,was,parked]
[accepted,that,the,vehicle,is,capable,to,move]
[accepted,that,vehicle,location,is,not,specified]
[the,vehicle,was,parked,in,public,parking]
[accepted,that,the,vehicle,was,parked,for,unknown,period]
[there,was,exigency,when,approached]
[accepted,that,the,origin,probable,cause,is,not,by,
information,received]
[accepted,that,the,origin,probable,cause,is,not,by,observation]
[accepted,that,the,origin,probable,cause,is,not,a,procedure,
or,procedure,is,not,clarified]
[accepted,that,origin,of,probable,cause,is,not,authorized,or,
not,clarified]
```

```
[main,reason,to,search,was,to,protect,the,public,cite,
carrol,v,us]
[the,main,reason,to,search,was,urgent]
[all,vehicle,parts,have,been,searched,cite,carrol,v,us,
and,us,v,ross]
[the,vehicle,was,searched,at,the,same,automobile,location]
[the,search,scope,is,legal]
[there,is,a,probable,cause,to,search,vehicle]
[accepted,that,all,automobiles,are,registered]
[accepted,that,not,restricted,area]
[subject,to,regular,inspection]
[accepted,that,item,is,not,on,public,view,or,details,are,
not,provided]
[accepted,that,can,not,be,seen,by,public,or,details,are
,not,provided]
[accepted,that,it,is,not,clear,that,items,can,not,be,seen]
[accepted,that,it,is,not,visible,to,public]
[illegal,goods]
[accepted,that,protection,level,can,not,be,determined]
[accepted,that,contents,are,not,considered,private]
[accepted,that,none,of,living,main,services,are,specified,
or,connected]
[accepted,that,not,connected,to,one,or,more,main,living,
services]
[accepted,that,vehicle,accommodation,spaces,are,not,
clarified]
[accepted,that,there,are,no,rooms,or,rooms,function,is,not,
specified]
[accepted,that,the,place,is,not,used,for,accommodation]
[accepted,that,low,expectation,of,privacy,in,use]
[there,is,risk,to,lose,evidence]
[accepted,that,magistrate,are,not,available]
[accepted,that,authorized,magistrate,are,not,available]
[it,is,not,easy,to,obtain,warrant]
[justified,under,automobile,exception,cite,carroll,v,us]
[reduced,expectation,of,privacy]
[warrantless,search,did,not,violate,the,fourth,amendment]
[find,for,the,plaintiff]
[******************]
[dissenting,argument]
[it,is,a,vehicle]
```

```
[accepted,that,no,largeContainers]
[paper,bag,is,a,movableContainer]
[it,is,a,mobile]
[accepted,that,the,vehicle,is,capable,to,move]
[there,was,exigency,when,approached]
[there,is,risk,to,lose,evidence]
[it,is,not,easy,to,obtain,warrant]
[justified,under,automobile,exception]
[enlarge,scope,of,fourth,amendment,exception]
[any,container,inside,automobile,shows,enough,privacy,
interest]
[warrantless,search,violates,the,fourth,amendment]
true .
```

Cady v. Dombrowski

```
[it,is,a,vehicle,cite,carroll,v,us]
[accepted,that,no,largeContainers]
[accepted,that,it,is,not,a,movableContainer]
[it,is,a,mobile]
[there,was,no,urgent,status,automobile,was,crashed]
[accepted,that,the,vehicle,is,capable,to,move]
[accepted,that,vehicle,location,is,not,specified]
[the,vehicle,was,parked,in,public,parking]
[accepted,that,the,vehicle,was,parked,for,unknown,period]
[there,was,exigency,when,approached]
[accepted,that,the,origin,probable,cause,is,not,by,
information,received]
[accepted,that,the,origin,probable,cause,is,not,by,observation]
[inspection,procedure,cite,harris,v,us,and,preston,v,us]
[there,was,an,authorized,origin,of,probable,cause]
[accepted,that,main,reason,to,search,was,to,protect,the,public]
[main,reason,to,search,was,due,to,a,crime]
[the,main,reason,to,search,was,urgent]
[all,vehicle,parts,have,been,searched,cite,carrol,v,us,
and,us,v,ross]
[the,vehicle,was,searched,at,a,garage]
[the,search,scope,is,legal]
[there,is,a,probable,cause,to,search,vehicle]
[accepted,that,all,automobiles,are,registered]
[accepted,that,not,restricted,area]
[subject,to,regular,inspection]
```

```
[accepted,that,item,is,not,on,public,view,or,details,are,
not,provided]
[items,can,be,seen,by,public]
[item,is,visible,to,public]
[accepted,that,goods,carried,are,unknown]
[accepted,that,protection,level,can,not,be,determined]
[accepted,that,contents,are,not,considered,private]
[accepted,that,none,of,living,main,services,are,specified,
or,connected]
[accepted,that,not,connected,to,one,or,more,main,living,
services]
[accepted,that,vehicle,accommodation,spaces,are,not,
clarified]
[accepted,that,there,are,no,rooms,or,rooms,function,is,
not,specified]
[accepted,that,the,place,is,not,used,for,accommodation]
[accepted,that,low,expectation,of,privacy,in,use]
[there,is,risk,to,lose,evidence]
[accepted,that,magistrate,are,not,available]
[accepted,that,authorized,magistrate,are,not,available]
[it,is,not,easy,to,obtain,warrant]
[justified,under,automobile,exception,cite,carroll,v,us]
[reduced,expectation,of,privacy]
[warrantless,search,did,not,violate,the,fourth,amendment]
[find,for,the,plaintiff]
[*******************]
[dissenting,argument]
[it,is,a,vehicle]
[accepted,that,no,largeContainers]
[accepted,that,it,is,not,a,movableContainer]
[it,is,a,mobile]
[the,vehicle,is,crashed,and,not,capable,to,move]
[accepted,that,there,was,no,exigency,when,approached]
[there,is,no,risk,to,lose,evidence]
[it,is,easy,to,obtain,warrant]
[reduce,expectation,of,exigency]
[privacy,required,it,is,not,justified,under,automobile,exception]
[warrantless,search,violates,the,fourth,amendment]
true .
```

Collidge v. New Hampshire

```
|    go(cvnh).
[it,is,a,vehicle]
[accepted,that,no,largeContainers]
[accepted,that,it,is,not,a,movableContainer]
[it,is,a,mobile]
[there,was,no,urgent,status,automobile,was,parked]
[accepted,that,the,vehicle,is,capable,to,move]
[the,vehicle,was,in,private,location]
[the,vehicle,was,parked,in,private,parking]
[accepted,that,the,vehicle,was,parked,for,unknown,period]
[accepted,that,there,was,no,exigency,when,approached]
[accepted,that,the,origin,probable,cause,is,not,by,
information,received]
[accepted,that,the,origin,probable,cause,is,not,by,observation]
[inspection,procedure]
[there,was,an,authorized,origin,of,probable,cause]
[accepted,that,main,reason,to,search,was,to,protect,the,public]
[the,main,reason,to,search,was,urgent]
[all,vehicle,parts,have,been,searched]
[accepted,that,the,vehicle,searching,location,is,not,clarified]
[the,search,scope,is,legal]
[there,is,a,probable,cause,to,search,vehicle]
[accepted,that,all,automobiles,are,registered]
[accepted,that,not,restricted,area]
[subject,to,regular,inspection]
[accepted,that,item,is,not,on,public,view,or,details,are,
not,provided]
[accepted,that,can,not,be,seen,by,public,or,details,are,
not,provided]
[accepted,that,it,is,not,clear,that,items,can,not,be,seen]
[accepted,that,it,is,not,visible,to,public]
[accepted,that,goods,carried,are,unknown]
[accepted,that,protection,level,can,not,be,determined]
[accepted,that,contents,are,not,considered,private]
[accepted,that,none,of,living,main,services,are,specified,
or,connected]
[accepted,that,not,connected,to,one,or,more,main,living,
services]
[accepted,that,vehicle,accommodation,spaces,are,not,
clarified]
[accepted,that,there,are,no,rooms,or,rooms,function,is,
```

```
not,specified]
[accepted,that,the,place,is,not,used,for,accommodation]
[accepted,that,low,expectation,of,privacy,in,use]
[there,was,no,risk,to,lose,evidence]
[accepted,that,magistrate,are,not,available]
[accepted,that,authorized,magistrate,are,not,available]
[warrant,obtained,but,not,authorized]
[reduce,expectation,of,exigency]
[high,expectation,of,privacy,obtained,warrant,was,issued,by,
neutral,and,
detached,magistrate,and,not,authorized]
[warrantless,search,violates,the,fourth,amendment]
[find,for,the,defendant]
true.
```

Chambers v. Maroney

```
|    go(cvm).
[it,is,a,vehicle]
[accepted,that,no,largeContainers]
[accepted,that,it,is,not,a,movableContainer]
[it,is,a,mobile]
[there,was,an,urgent,status,when,vehicle,is,moving]
[the,vehicle,was,in,public,location]
[accepted,that,vehicle,was,not,parked]
[there,was,exigency,when,approached]
[accepted,that,the,origin,probable,cause,is,not,by,
information,received]
[accepted,that,the,origin,probable,cause,is,not,by,observation]
[inspection,procedure]
[there,was,an,authorized,origin,of,probable,cause]
[accepted,that,main,reason,to,search,was,to,protect,
the,public]
[the,main,reason,to,search,was,urgent]
[all,vehicle,parts,have,been,searched]
[accepted,that,the,vehicle,searching,location,is,not,
clarified]
[the,search,scope,is,legal]
[there,is,a,probable,cause,to,search,vehicle]
[accepted,that,all,automobiles,are,registered]
[accepted,that,not,restricted,area]
[subject,to,regular,inspection]
```

```
[accepted,that,item,is,not,on,public,view,or,details,are,
not,provided]
[accepted,that,can,not,be,seen,by,public,or,details,are,
not,provided]
[accepted,that,it,is,not,clear,that,items,can,not,be,seen]
[accepted,that,it,is,not,visible,to,public]
[accepted,that,goods,carried,are,unknown]
[accepted,that,protection,level,can,not,be,determined]
[accepted,that,contents,are,not,considered,private]
[accepted,that,none,of,living,main,services,are,specified,
or,connected]
[accepted,that,not,connected,to,one,or,more,main,living,
services]
[accepted,that,vehicle,accommodation,spaces,are,not,
clarified]
[accepted,that,there,are,no,rooms,or,rooms,function,is,not,
specified]
[accepted,that,the,place,is,not,used,for,accommodation]
[accepted,that,low,expectation,of,privacy,in,use]
[accepted,that,there,is,risk,to,lose,evidence]
[accepted,that,magistrate,are,not,available]
[accepted,that,authorized,magistrate,are,not,available]
[it,is,not,easy,to,obtain,warrant]
[justified,under,automobile,exception]
[reduced,expectation,of,privacy]
[warrantless,search,did,not,violate,the,fourth,amendment]
[find,for,the,plaintiff]
true.
```

Carroll v. US

```
go(cvus).
[it,is,a,vehicle]
[accepted,that,no,largeContainers]
[accepted,that,it,is,not,a,movableContainer]
[it,is,a,mobile]
[there,was,an,urgent,status,when,vehicle,is,moving]
[accepted,that,vehicle,location,is,not,specified]
[accepted,that,vehicle,was,not,parked]
[there,was,exigency,when,approached]
[received,information,from,public,informant]
[there,was,an,authorized,origin,of,probable,cause]
```

```
[main,reason,to,search,was,to,protect,the,public]
[the,main,reason,to,search,was,urgent]
[all,vehicle,parts,have,been,searched]
[the,vehicle,was,searched,at,the,same,automobile,location]
[the,search,scope,is,legal]
[there,is,a,probable,cause,to,search,vehicle]
[accepted,that,all,automobiles,are,registered]
[accepted,that,not,restricted,area]
[subject,to,regular,inspection]
[accepted,that,item,is,not,on,public,view,or,details,are,not,
provided]
[accepted,that,can,not,be,seen,by,public,or,details,are,not,
provided]
[accepted,that,it,is,not,clear,that,items,can,not,be,seen]
[accepted,that,it,is,not,visible,to,public]
[illegal,goods]
[accepted,that,protection,level,can,not,be,determined]
[accepted,that,contents,are,not,considered,private]
[accepted,that,none,of,living,main,services,are,specified,or,
connected]
[accepted,that,not,connected,to,one,or,more,main,living,
services]
[accepted,that,vehicle,accommodation,spaces,are,not,
clarified]
[accepted,that,there,are,no,rooms,or,rooms,function,is,not,
specified]
[accepted,that,the,place,is,not,used,for,accommodation]
[accepted,that,low,expectation,of,privacy,in,use]
[accepted,that,there,is,risk,to,lose,evidence]
[accepted,that,magistrate,are,not,available]
[accepted,that,authorized,magistrate,are,not,available]
[it,is,not,easy,to,obtain,warrant]
[justified,under,automobile,exception]
[reduced,expectation,of,privacy]
[warrantless,search,did,not,violate,the,fourth,amendment]
[find,for,the,plaintiff]
true.
```

Bibliography

[1] Latifa Al-Abdulkarim, Katie Atkinson, and Trevor Bench-Capon, *Dialogues in US Supreme Court oral hearings*, Proceedings of CMNA 2013, 2013, In press, see: http://www.cmna.info/CMNA13/paper_8.pdf.

[2] Latifa Al-Abdulkarim, Katie Atkinson, and Trevor Bench-Capon, *From oral hearing to opinion in the US Supreme Court*, Proceedings of JURIX 2013, 2013, pp. 1–10.

[3] Latifa Al-Abdulkarim, Katie Atkinson, and Trevor Bench-Capon, *Abstract Dialectical Frameworks for legal reasoning*, Proceedings of Jurix 2014, 2014, pp. 61–70.

[4] Latifa Al-Abdulkarim, Katie Atkinson, and Trevor Bench-Capon, *Support for factor-based argumentation*, Computational Models of Argument - Proceedings of COMMA 2014, Atholl Palace Hotel, Scottish Highlands, UK, September 9-12, 2014, 2014, pp. 447–448.

[5] Latifa Al-Abdulkarim, Katie Atkinson, and Trevor Bench-Capon, *Evaluating the use of Abstract Dialectical Frameworks to represent case law*, Proceedings of the 15th International Conference on Artificial Intelligence and Law, ICAIL '15, ACM, 2015, pp. 156–160.

[6] Latifa Al-Abdulkarim, Katie Atkinson, and Trevor Bench-Capon, *Factors, issues and values: Revisiting reasoning with cases*, Proceedings of the 15th International Conference on Artificial Intelligence and Law, ICAIL '15, ACM, 2015, pp. 3–12.

[7] Latifa Al-Abdulkarim, Katie Atkinson, and Trevor Bench-Capon, *Using Abstract Dialectical Frameworks to argue about legal cases.*, D. Mohammed and M. Lewinski (eds.), Argumentation and Reasoned Action: Proceedings of the First European Conference on Argumentation, vol. II, College Publications, London, 2015, pp. 163–180.

[8] Latifa Al-Abdulkarim, Katie Atkinson, and Trevor Bench-Capon, *A methodology for designing systems to reason with legal cases using Abstract Dialectical Frameworks*, Artificial Intelligence and Law **24** (2016), no. 1, 1–49.

[9] Latifa Al-Abdulkarim, Katie Atkinson, and Trevor Bench-Capon, *ANGELIC secrets: Bridging from factors to facts in US trade secrets*, Legal Knowledge and Information Systems - JURIX 2016: The Twenty-Ninth Annual Conference, 2016, pp. 113–118.

[10] Vincent Aleven, *Teaching case-based argumentation through a model and examples*, Ph.D. thesis, University of Pittsburgh, 1997.

[11] Vincent Aleven and Kevin Ashley, *Doing things with factors*, Proceedings of the Fifth International Conference on Artificial Intelligence and Law, ICAIL '95, 1995, pp. 31–41.

[12] Leila Amgoud and Claudette Cayrol, *On the acceptability of arguments in preference-based argumentation*, Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, 1998, pp. 1–7.

[13] Michal Araszkiewicz, Agata Lopatkiewicz, Adam Zienkiewicz, and Tomasz Zurek, *Representation of an actual divorce dispute in the parenting plan support system*, Proceedings of the 15th International Conference on Artificial Intelligence and Law, ICAIL 2015, 2015, pp. 166–170.

[14] Kevin Ashley, *Modelling legal argument: Reasoning with cases and hypotheticals*, Bradford Books/MIT Press, Cambridge, MA, 1990.

[15] Kevin Ashley, *Teaching a process model of legal argument with hypotheticals*, Artificial Intelligence and Law **17** (2009), no. 4, 321–370.

[16] Kevin Ashley, *What a legal CBR ontology should provide*, Proceedings of the Twenty-Second International Florida Artificial Intelligence Research Society Conference, 2009.

[17] Kevin Ashley and Stefanie Brüninghaus, *A predictive role for intermediate legal concepts*, Proceedings of JURIX 2003, 2003, pp. 1–10.

[18] Kevin Ashley and Stefanie Brüninghaus, *Automatically classifying case texts and predicting outcomes.*, Artificial Intelligence and Law. **17** (2009), no. 2, 125–165.

[19] Kevin Ashley, Collin Lynch, Niels Pinkwart, and Vincent Aleven, *A process model of legal argument with hypotheticals*, JURIX 2008: The Twenty-First Annual Conference on Legal Knowledge and Information Systems, 2008, pp. 1–10.

[20] Katie Atkinson and Trevor Bench-Capon, *Legal case-based reasoning as practical reasoning*, Artificial Intelligence and Law **13** (2005), no. 1, 93–131.

[21] Katie Atkinson and Trevor Bench-Capon, *Argumentation and standards of proof*, ICAIL'07: Proceedings of the 11th International Conference on Artificial Intelligence and Law (Palo Alto, CA, USA), ACM Press: New York, NY, 2007, pp. 107–116.

[22] Katie Atkinson and Trevor Bench-Capon, *Practical reasoning as presumptive argumentation using action based alternating transition systems*, Artificial Intelligence **171** (2007), no. 10-15, 855–874.

[23] Katie Atkinson, Trevor Bench-Capon, and Peter McBurney, *Computational representation of practical argument*, Synthese **152** (2006), no. 2, 157–206.

[24] Katie Atkinson, Trevor Bench-Capon, Henry Prakken, and Adam Wyner, *Argumentation schemes for reasoning about factors with dimensions*, Proceedings of 26th International Conference on Legal Knowledge and Information Systems (JURIX 2013) (2013), 39–48.

[25] Katie Atkinson, Trevor Bench-Capon, and Douglas Walton, *Distinctive features of persuasion and deliberation dialogues*, Argument & Computation **4** (2013), no. 2, 105–127.

[26] John Langshaw Austin, *How to do things with words*, Harvard University Press, 1975.

[27] Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Giovanni Guida, *Encompassing attacks to attacks in abstract argumentation frameworks*, Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 10th European Conference, ECSQARU 2009 2009., 2009, pp. 83–94.

[28] Trevor Bench-Capon, *Persuasion in practical argument using value-based argumentation frameworks*, Journal of Logic and Computation **13** (2003), no. 3, 429–448.

[29] Trevor Bench-Capon, *Relating values in a series of supreme court decisions*, Proceedings of JURIX 2011, IOS Press, 2011, pp. 13–22.

[30] Trevor Bench-Capon, *Representing Popov v Hayashi with dimensions and factors*, Artificial Intelligence and Law **20** (2012), no. 1, 15–35.

[31] Trevor Bench-Capon, Michal Araszkiewicz, Kevin Ashley, Katie Atkinson, Floris Bex, Filipe Borges, Danièle Bourcier, Paul Bourgine, Jack Conrad, Enrico Francesconi, Thomas Gordon, Guido Governatori, Jochen Leidner, David Lewis, Ronald Prescott Loui, L. Thorne McCarty, Henry Prakken, Frank Schilder, Erich Schweighofer, Paul Thompson, Alex Tyrrell, Bart Verheij, Douglas Walton, and Adam Zachary Wyner, *A history of AI and law in 50 papers: 25 years of the international conference on AI and law*, Artificial Intelligence and Law **20** (2012), no. 3, 215–319.

[32] Trevor Bench-Capon and Floris Bex, *Cases and stories, dimensions and scripts*, Legal Knowledge and Information Systems - JURIX 2015: The Twenty-Eighth Annual Conference, Braga, Portual, December 10-11, 2015, 2015, pp. 11–20.

[33] Trevor Bench-Capon and Paul Dunne, *Argumentation in artificial intelligence*, Artificial Intelligence **171** (2007), no. 10, 619–641.

[34] Trevor Bench-Capon, T. Geldard, and Paul Leng, *A method for the computational modelling of dialectical argument with dialogue games*, Artificial Intelligence and Law **8** (2000), no. 2/3, 233–254.

[35] Trevor Bench-Capon and Sanjay Modgil, *Case law in extended argumentation frameworks*, In Proceedings of the 12th International Conference on Artificial Intelligence and Law, ACM Press, 2009, pp. 118–127.

[36] Trevor Bench-Capon, Henry Prakken, and Giovanni Sartor, *Argumentation in legal reasoning*, pp. 363–382, Springer US, Boston, MA, 2009.

[37] Trevor Bench-Capon and Edwina Rissland, *Back to the future: dimensions revisited*, Proceedings of JURIX 2001, IOS Press, 2001, pp. 41–52.

[38] Trevor Bench-Capon and Giovanni Sartor, *A model of legal reasoning with cases incorporating theories and values*, Artificial Intelligence **150** (2003), no. 1-2, 97–143.

[39] Trevor BenchCapon, Frans Coenen, and Paul Orton, *Argument Based Explanation of the British Nationality Act as a logic program*, Information & Communications Technology Law **2** (1993), no. 1, 53–66.

[40] Donald Berman and Carole Hafner, *Representing teleological structure in case-based legal reasoning: The missing link*, Proceedings of the Fourth International Conference on Artificial intelligence and Law, 1993, pp. 50–59.

[41] Donald Berman and Carole Hafner, *Understanding precedents in a temporal context of evolving legal doctrine*, Proceedings of the Fifth International Conference on Artificial Intelligence and Law, 1995, pp. 42–51.

[42] Floris Bex, *Arguments, stories and criminal evidence: A formal hybrid theory*, Springer, 2011.

[43] Floris Bex, *An integrated theory of causal stories and evidential arguments*, Proceedings of the 15th International Conference on Artificial Intelligence and Law, ICAIL 2015, 2015, pp. 13–22.

[44] Floris Bex, Henry Prakken, Chris Reed, and Douglas Walton, *Towards a formal account of reasoning about evidence: argumentation schemes and generalisations*, Artificial Intelligence and Law **11** (2003), no. 2, 125–165.

[45] Floris Bex and Chris Reed, *Dialogue templates for automatic argument processing*, Computational Models of Argument - Proceedings of COMMA 2012, 2012, pp. 366–377.

[46] Floris Bex and Bart Verheij, *Legal stories and the process of proof*, Artif. Intell. Law **21** (2013), no. 3, 253–278.

[47] L. Karl Branting, *Representing and reusing explanations of legal precedents*, Proceedings of the 2nd International Conference on Artificial Intelligence and Law, ICAIL '89, ACM, 1989, pp. 103–110.

[48] L. Karl Branting, *Reasoning with portions of precedents*, Proceedings of the 3rd International Conference on Artificial Intelligence and Law, ACM, 1991, pp. 145–154.

[49] L. Karl Branting, *An issue-oriented approach to judicial document assembly*, Proceedings of the 4th International Conference on Artificial Intelligence and Law, ACM, 1993, pp. 228–235.

[50] Joost Breuker and Nienke den Haan, *Separating world and regulation knowledge: Where is the logic*, Proceedings of the Third International Conference on Artificial Intelligence and Law, ACM Press, 1991, pp. 92–97.

[51] Gerhard Brewka, Paul Dunne, and Stefan Woltran, *Relating the semantics of Abstract Dialectical Frameworks and standard AFs*, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, 2011, pp. 780–785.

[52] Gerhard Brewka and Thomas Gordon, *Carneades and Abstract Dialectical Frameworks: A reconstruction*, Proceedings of the 2010 Conference on Computational Models of Argument: (COMMA) 2010 (Amsterdam, The Netherlands), IOS Press, 2010, pp. 3–12.

[53] Gerhard Brewka, Hannes Strass, Stefan Ellmauthaler, Johannes Peter Wallner, and Stefan Woltran, *Abstract Dialectical Frameworks Revisited*, 23rd International Joint Conference on Artificial Intelligence, 2013, pp. 803–809.

[54] Gerhard Brewka and Stefan Woltran, *Abstract Dialectical Frameworks*, Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, 2010, pp. 102–111.

[55] Stefanie Brüninghaus and Kevin Ashley, *Predicting outcomes of case-based legal arguments*, ICAIL'03: Proceedings of the 9th International Conference on Artificial Intelligence and Law, ACM Press, 2003, pp. 233–242.

[56] Claudette Cayrol and Marie-Christine Lagasquie-Schiex, *Bipolar abstract argumentation systems*, Guillermo Simari and Iyad Rahwan (Eds), Argumentation in Artificial Intelligence, Springer US, 2009, pp. pp 65–84.

[57] Alison Chorley and Trevor Bench-Capon, *AGATHA: using heuristic search to automate the construction of case law theories*, Artificial Intelligence and Law **13** (2005), no. 1, 9–51 (English).

[58] Alison Chorley and Trevor Bench-Capon, *An empirical investigation of reasoning with legal cases through theory construction and application*, Artificial Intelligence and Law **13** (2005), no. 3-4, 323–371.

[59] Keith Clark, *Negation as failure*, Herve Gallaire and Jack Minkler (Eds), Logic and data bases, Springer, 1978, pp. 293–322.

[60] Joe Collenette, *EFACTS automating annotations of US Supreme Court oral hearings*, 2015, University of Liverpool, BSc Dissertation.

[61] Phan Minh Dung, *On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming, and n-person games*, Artificial Intelligence **77** (1995), 321–357.

[62] Paul Dunne, Sylvie Doutre, and Trevor Bench-Capon, *Discovering inconsistency through examination dialogues*, IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, 2005, pp. 1680–1681.

[63] Stefan Ellmauthaler and Hannes Strass, *The DIAMOND system for computing with Abstract Dialectical Frameworks*, Computational Models of Argument - Proceedings of COMMA 2014, 2014, pp. 233–240.

[64] Arthur Farley and Kathleen Freeman, *Burden of proof in legal argumentation*, Proceedings of the 5th international conference on Artificial Intelligence and Law, ACM, 1995, pp. 156–164.

[65] Timothy Finin, Richard Fritzson, Donald McKay, and Robin McEntire, *KQML as an agent communication language*, Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94), 1994, pp. 456–463.

[66] TCC FIPA, *FIPA Communicative Act library Specification*, Available from the website `http://www.fipa.org`, 2001, Technical Report XC00037H, Foundation for Intelligent Physical Agents, 10th August 2001.

[67] James Fowler, Timothy Johnson, James Spriggs, Sangick Jeon, and Paul Wahlbeck, *Network analysis and the law: Measuring the legal importance of precedents at the u.s. supreme court*, Political Analysis **15** (2007), no. 3, 324–346.

[68] Anne von der Lieth Gardner, *An artificial intelligence approach to legal reasoning*, MIT Press, Cambridge, MA, 1984.

[69] Wilfred Glover, *Automating Annotation of Legal Documents*, 2016, University of Liverpool, BSc Dissertation.

[70] Thomas Gordon, *The Pleadings Game: Formalizing procedural justice*, Proceedings of the Fourth ICAIL, 1993, pp. 10–19.

[71] Thomas Gordon, Henry Prakken, and Douglas Walton, *The Carneades model of argument and burden of proof*, Artificial Intelligence **171** (2007), no. 10-15, 875–896.

[72] Thomas Gordon and Douglas Walton, *Pierson vs. post revisited - A reconstruction using the carneades argumentation framework*, Computational Models of Argument: Proceedings of COMMA 2006, September 11-12, 2006, Liverpool, UK, 2006, pp. 208–219.

[73] Thomas Gordon and Douglas Walton, *Legal reasoning with argumentation schemes*, Proceedings of the 12th International Conference on Artificial Intelligence and Law, ICAIL '09, ACM, 2009, pp. 137–146.

[74] Thomas Gordon and Douglas Walton, *Proof burdens and standards*, pp. 239–258, Springer US, Boston, MA, 2009.

[75] Thomas Gordon and Douglas Walton, *A carneades reconstruction of popov v hayashi*, Artificial Intelligence and Law **20** (2012), no. 1, 37–56.

[76] Matthias Grabmair and Kevin Ashley, *Argumentation with value judgments - an example of hypothetical reasoning*, Legal Knowledge and Information Systems - JURIX 2010: The Twenty-Third Annual Conference on Legal Knowledge and Information Systems, 2010, pp. 67–76.

[77] Thomas Gruber, *The role of common ontology in achieving sharable, reusable knowledge bases*, Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91). Cambridge, MA, USA, April 22-25, 1991., 1991, pp. 601–602.

[78] Carole Hafner, *Conceptual organization of case law knowledge bases*, Proceedings of the First International Conference on AI and Law, 1987, pp. 35–42.

[79] Charles Leonard Hamblin, *Fallacies*, London: Methuen, 1970.

[80] Herbert Lionel Adolphus Hart, *The Concept of Law, 3rd edition*, Oxford University Press, 2012.

[81] John Henderson and Trevor Bench-Capon, *Dynamic arguments in a case law domain*, Proceedings of the Eighth International Conference on Artificial Intelligence and Law, ICAIL 2001, 2001, pp. 60–69.

[82] Rinke Hoekstra, Joost Breuker, Marcello Di Bello, and Alexander Boer, *The LKIF core ontology of basic legal concepts.*, LOAIT **321** (2007), 43–63.

[83] John Horty and Trevor Bench-Capon, *A factor-based definition of precedential constraint*, Artificial Intelligence and Law **20** (2012), no. 2, 181–214.

[84] Michael Bommarito II, Daniel Martin Katz, and Jon Zelner, *Law as a seamless web?: comparison of various network representations of the united states supreme court corpus (1791-2005)*, The 12th International Conference on Artificial Intelligence and Law, Proceedings of the Conference, June 8-12, 2009, Barcelona, Spain, 2009, pp. 234–235.

[85] Peter Johnson and David Mead, *Legislative knowledge base systems for public administration: some practical issues*, Proceedings of the 3rd international conference on Artificial intelligence and law, ACM, 1991, pp. 108–117.

[86] Jeroen Keppens, *Towards qualitative approaches to Bayesian evidential reasoning*, Proceedings of the 11th International Conference on Artificial Intelligence and Law, ACM, 2007, pp. 17–25.

[87] Marc Lauritsen, *On balance*, Artificial Intelligence and Law **23** (2015), no. 1, 23–42.

[88] Edward Levi, *An introduction to legal reasoning*, The University of Chicago Law Review **15** (1948), no. 3, 501–574.

[89] Lars Lindahl and Jan Odelstad, *Intermediaries and intervenients in normative systems*, Journal of Applied Logic **6** (2008), no. 2, 229–250.

[90] Arno Lodder, *DiaLaw. on legal justification and dialogical models of argumentation*, Kluwer, 1999.

[91] Arno Lodder, *A simple model to structure the information of parties in online alternative dispute resolution*, Proceedings of the Eigths International Conference on Artificial Intelligence and Law, ICAIL 2001, 2001, p. 233.

[92] Arno Lodder and Aimée Herczog, *DiaLaw: A dialogical framework for modeling legal reasoning*, Proceedings of the Fifth International Conference on Artificial Intelligence and Law, ICAIL '95, 1995, pp. 146–155.

[93] Lee Loevinger, *Jurimetrics: The next step forward*, Minnesota Law Review **33** (1949), no. 5, 455 – 493.

[94] L. Lutomski, *The design of an attorney's statistical consultant*, Proceedings of the Second International Conference on Artificial Intelligence and Law, ICAIL '89, 1989, pp. 224–233.

[95] J.D. MacKenzie, *Question-begging in non-cumulative systems*, Journal of Philosophical Logic **8** (1979), 159–177.

[96] C. Marshall, *Representing the structure of a legal argument*, Proceedings of the Second International Conference on Artificial Intelligence and Law, ICAIL '89, 1989, pp. 121–127.

[97] L. Thorne McCarty, *An implementation of Eisner v. Macomber*, Proceedings of the Fifth International Conference on Artificial Intelligence and Law, ICAIL '95, College Park, Maryland, USA, May 21-24, 1995, 1995, pp. 276–286.

[98] Thorne McCarty, *Reflections on TAXMAN: An experiment in artificial intelligence and legal reasoning*, Harvard Law Review (1977), 837–893.

[99] Sanjay Modgil and Trevor Bench-Capon, *Metalevel argumentation*, Journal of Logic and Computation **21** (2011), no. 6, 959–1003.

[100] Sanjay Modgil and Henry Prakken, *Reasoning about preferences in structured extended argumentation frameworks*, Computational Models of Argument: Proceedings of COMMA 2010., 2010, pp. 347–358.

[101] Sanjay Modgil and Henry Prakken, *The ASPIC+ framework for structured argumentation: a tutorial*, Argument & Computation **5** (2014), no. 1, 31–62.

[102] Monica Palmirani, Guido Governatori, and Giuseppe Contissa, *Modelling temporal legal rules*, The 13th International Conference on Artificial Intelligence and Law, 2011, pp. 131–135.

[103] Henry Prakken, *From logic to dialectics in legal argument*, Proceedings of the 5th International Conference on Artificial Intelligence and Law (New York, NY, USA), ACM, 1995, pp. 165–174.

[104] Henry Prakken, *Formal systems for persuasion dialogue*, Knowledge Engineering Review **21** (2006), no. 2, 163–188.

[105] Henry Prakken, *An abstract framework for argumentation with structured arguments*, Argument & Computation **1** (2010), no. 2, 93–124.

[106] Henry Prakken, *Formalising a legal opinion on a legislative proposal in the ASPIC+ framework*, Legal Knowledge and Information Systems - JURIX 2012: The Twenty-Fifth Annual Conference, 2012, pp. 119–128.

[107] Henry Prakken and Giovanni Sartor, *A dialectical model of assessing conflicting arguments in legal reasoning*, Artificial Intelligence and Law **4** (1996), no. 3-4, 331–368.

[108] Henry Prakken and Giovanni Sartor, *Modelling reasoning with precedents in a formal dialogue game*, Artificial Intelligence and Law **6** (1998), no. 2-4, 231–287.

[109] Henry Prakken and Giovanni Sartor, *Formalising arguments about the burden of persuasion*, The 11th International Conference on Artificial Intelligence and Law, ACM, 2007, pp. 97–106.

[110] Henry Prakken and Giovanni Sartor, *Law andLogic: A review from an argumentation perspective*, Artificial Intelligence **227** (2015), 214–245.

[111] Henry Prakken, Adam Zachary Wyner, Trevor Bench-Capon, and Katie Atkinson, *A formalization of argumentation schemes for legal case-based reasoning in ASPIC+*, Journal of Logic and Computation **25** (2015), no. 5, 1141–1166.

[112] Roger Pressman, *Software engineering: a practitioner's approach*, Palgrave Macmillan, 2005.

[113] Chris Reed and Glenn Rowe, *Araucaria: Software for argument analysis, diagramming and representation*, International Journal on Artificial Intelligence Tools **13** (2004), no. 4, 983–.

[114] Adam Rigoni, *An improved factor based approach to precedential constraint*, Artificial Intelligence and Law **23** (2015), no. 2, 133–160.

[115] Edwina Rissland, *Dimension-based analysis of hypotheticals from Supreme Court oral argument*, Proceedings of the Second ICAIL, ACM, 1989, pp. 111–120.

[116] Edwina Rissland and Kevin Ashley, *A case-based system for trade secrets law*, Proceedings of the First International Conference on Artificial Intelligence and Law, ICAIL '87, 1987, pp. 60–66.

[117] Edwina Rissland, David Skalak, and M. Timur Friedman, *BankXX: Supporting legal arguments through heuristic retrieval*, Artificial Intelligence and Law **4** (1996), no. 1, 1–71.

[118] Edwina Rissland, David Skalak, and M. Timur Friedman, *Evaluating a legal argument program: The bankXX experiments*, Artificial Intelligence and Law **5** (1997), no. 1-2, 1–74.

[119] John Searle, *Speech acts: An essay in the philosophy of language*, Cambridge University Press, 1969.

[120] Marek Sergot, Fariba Sadri, Robert Kowalski, Frank Kriwaczek, Peter Hammond, and Terese Cory, *The British Nationality Act as a logic program*, Communications of the ACM **29** (1986), no. 5, 370–386.

[121] David. Skalak and Edwina Rissland, *Arguments and cases: An inevitable intertwining*, Artificial Intelligence and Law **1** (1992), no. 1, 3–44.

[122] J. C. Smith, Daphne Gelbart, Keith MacCrimmon, Bruce Atherton, John McClean, Michelle Shinehoft, and Lincoln Quintana, *Artificial intelligence and legal discourse: The flexlaw legal text management system*, Artificial Intelligence and Law **3** (1995), no. 1-2, 55–95.

[123] Sjoerd Timmer, John-Jules Meyer, Henry Prakken, Silja Renooij, and Bart Verheij, *A structure-guided approach to capturing bayesian reasoning about legal evidence in argumentation*, Proceedings of the 15th International Conference on Artificial Intelligence and Law (New York, NY, USA), ICAIL '15, ACM, 2015, pp. 109–118.

[124] Stephen Toulmin, *The uses of argument*, Cambridge University Press, 1958.

[125] Caroline Uyttendaele, Marie-Francine Moens, and Jos Dumortier, *Salomon: Automatic abstracting of legal cases for effective access to court decisions*, Artificial Intelligence and Law **6** (1998), no. 1, 59–79.

[126] A. Valente, *Legal knowledge engineering: A modelling approach*, Frontiers in artificial intelligence and applications, IOS Press, 1995.

[127] Bas van Gijzel and Henry Prakken, *Relating Carneades with abstract argumentation via the ASPIC+ framework for structured argumentation*, Argument & Computation **3** (2012), no. 1, 21–47.

[128] Pepijn R. S. Visser and Trevor Bench-Capon, *A comparison of four ontologies for the design of legal knowledge systems*, Artificial Intelligence and Law **6** (1998), no. 1, 27–57.

[129] P.R.S. Visser, *Knowledge specification for multiple legal tasks: a case study of the interaction problem in the legal domain*, Computer/law series, Kluwer Law International, 1995.

[130] Friedrich Waismann, *The Principles of Linguistic Philosophy*, Macmillan, 1965.

[131] Douglas Walton, *Types of dialogue and burdens of proof*, Proceedings of the 2010 Conference on Computational Models of Argument: Proceedings of COMMA 2010 (Amsterdam, The Netherlands), IOS Press, 2010, pp. 13–24.

[132] Douglas Walton, *Argumentation schemes for presumptive reasoning*, Taylor & Francis, 2013.

[133] Douglas Walton and Eric Krabbe, *Commitment in Dialogue: Basic concepts of interpersonal reasoning*, Suny Series, Logic & Language, State University of New York Press, 1995.

[134] Douglas Walton, Chris Reed, and Fabrizio Macagno, *Argumentation Schemes*, Cambridge University Press, 2008.

[135] Maya Wardeh, Trevor Bench-Capon, and Frans Coenen, *PADUA: a protocol for argumentation dialogue using association rules*, Artificial Intelligence and Law **17** (2009), no. 3, 183–215.

[136] Maya Wardeh, Frans Coenen, and Trevor Bench-Capon, *Multi-agent based classification using argumentation from experience*, Autonomous Agents and Multi-Agent Systems **25** (2012), no. 3, 447–474.

[137] Maya Wardeh, Frans Coenen, and Trevor Bench-Capon, *PISA: A framework for multiagent classification using argumentation*, Data Knowledge. Engineering **75** (2012), 34–57.

[138] Ludwig Wittgenstein, *Philosophical Investigations. translated by anscombe, G.E.M.*, Basil Blackwell, Oxford, 1953.

[139] Adam Wyner and Rinke Hoekstra, *A legal case OWL ontology with an instantiation of Popov v. Hayashi*, Artificial Intelligence and Law **20** (2012), no. 1, 83–107.

[140] Adam Wyner and Wim Peters, *Lexical semantics and expert legal knowledge towards the identification of legal case factors*, Legal Knowledge and Information Systems - JURIX 2010: The Twenty-Third Annual Conference on Legal Knowledge and Information Systems,, 2010, pp. 127–136.

[141] Adam Wyner and Wim Peters, *On rule extraction from regulations*, Legal Knowledge and Information Systems - JURIX 2011: The Twenty-Fourth Annual Conference, 2011, pp. 113–122.

[142] Adam Wyner, Wim Peters, and Daniel Katz, *A case study on legal case annotation*, Legal Knowledge and Information Systems - JURIX 2013: The Twenty-Sixth Annual Conference, 2013, pp. 165–174.

[143] Adam Zachary Wyner, *An ontology in OWL for legal case-based reasoning*, Artificial Intelligence and Law **16** (2008), no. 4, 361–387.

[144] Adam Zachary Wyner, Trevor Bench-Capon, and Katie Atkinson, *Arguments, values and baseballs: Representation of Popov v. Hayashi*, Legal Knowledge and Information Systems - JURIX 2007: The Twentieth Annual Conference on Legal Knowledge and Information Systems, 2007, pp. 151–160.

[145] Adam Zachary Wyner, Trevor Bench-Capon, and Katie Atkinson, *Towards formalising argumentation about legal cases*, The 13th International Conference on Artificial Intelligence and Law, Proceedings of the Conference, 2011, pp. 1–10.

[146] Tomasz Zurek and Michal Araszkiewicz, *Modeling teleological interpretation*, Proceedings of the 14th International Conference on Artificial Intelligence and Law, 2013, pp. 160–168.