Optimal Methods for Reasoning about Actions and Plans in Multi-agent Systems

Tiago de Lima

Institut de Recherche en Informatique de Toulouse

UNIVERSITÉ DE TOULOUSE
ÉCOLE DOCTORALE MATHÉMATIQUES, INFORMATIQUE ET
TÉLÉCOMMUNICATIONS DE TOULOUSE
INSTITUT DE RECHERCHE EN INFORMATIQUE DE TOULOUSE

OPTIMAL METHODS FOR REASONING ABOUT ACTIONS AND PLANS IN MULTI-AGENT SYSTEMS

Thesis presented and defended by

TIAGO DE LIMA

at Toulouse on the 22nd of October 2007 to obtain the degree of
Docteur en Informatique de l'Université de Toulouse

Supervisor: Andreas Herzig

Commetee:

Nicholas Asher (examiner)
Patrick Blackburn (reviewer)
Andreas Herzig (supervisor)
Lambèr Royakkers (examiner)
Philippe Balbiani (examiner)
Martin Cooper (president)
Gerhard Lakemeyer (reviewer)

Supported by the Programme Alßan,
the European Union Programme of High Level Scholarships for Latin America,
scholarship number E04D041703BR.

Acknowledgements

I worked along three years on this thesis. It is unlikely that I will be able to remember the names of all the people that I should acknowledge for that. If your name should be listed here but it is not, I am awfully sorry for that. And please, accept my sincere acknowledgments right now.

First, special thanks to my supervisor Andreas Herzig and also to Philippe Balbiani and Hans van Ditmarsch for co-authoring, extremely valuable discussions, important advising, support and help.

Thanks to the members of the cometee of my thesis defense, Nicholas Asher, Philipe Balbiani, Patrick Blackburn, Martin Cooper, Andreas Herzig, Gerhard Lakemeyer and Lambèr Royakkers for the comments and discussions during the defense. Thanks to Alexandru Baltag and Tomohiro Hoshi for co-authoring. Thanks to Barteld Kooi, Teo Kuipers and the Department of Theoretical Philosophy of the University of Groningen, for the invitation to Groningen, exiting discussions and very nice stay. Thanks to Olivier Roy, Joel Uckelman, Sara Uckelman and Jonathan Zvesper for the invitations and accommodation in Amsterdam. Thanks to Marcos Castilho, Alexandre Direne, and Michel Gagnon for the support, advising and help in the very beginning. Thanks to Guillaume Aucher, Megyn Bienvenu, Olivier Gasquet, Philippe Muller, Jérôme Lang, Emiliano Lorini, Bilal Said, Nicolas Troquard, and Ivan Varzinczak for some nice discussions and friendship. And special thanks to Mounira Kourjieh, Antonio de Lima, Marli de Lima, Tatiele de Lima and Thammi de Lima for other support. I would also like to thank the Programme Alßan for the financial support, and the Institut de Recherche en Informatique de Toulouse, where this thesis was developed.

And finally, thank you, the reader, for reading my thesis.

Tiago de Lima
Toulouse, October 2007

Contents

Chapter 1

Introduction

Conceived as a complex human activity, reasoning about actions and plans relies on the ability of relating cause and effect. The subject is vast and is an object of study of several science fields such as philosophy, computer science and logic. Results of these studies provide essential theoretical foundations for building automated systems. As good examples, we have systems for manufacturing, software engineering, robotics, logistics, education and entertainment. To better illustrate what we mean by the term 'reasoning' here, we give below a simple example inspired by a puzzle of Smullyan (1992).

EXAMPLE 1 (THE LADY OR THE TIGER)
The environment consists of an individual, called *agent*, that inhabits a room with two doors. These doors may be opened by the agent and, if so, behind each one the agent will either find the lady, or the tiger. If the agent opens a door and finds the lady, then she will marry him, and if he finds the tiger, then it will kill him. The available actions are:

- $listen_1$ and $listen_2$. By performing one of those, the agent listens to what happens behind the respective door, which results in hearing the tiger roaring if there is one behind the door; and
- $open_1$ and $open_2$. By performing it, the agent opens the respective door, which results in marrying the lady, or being killed by the tiger, depending on what is behind the door.

This example is used all along this thesis. It describes what we call a *dynamic system*. An example of *initial state* of this system could be: the agent is alive and not married, the lady is behind door 1 and the tiger is behind door 2. While an example of *goal* could be: the agent is alive and married. When a dynamic system is accompanied by an initial state and a goal, it is called a *planning problem*. A solution to the planning problem is a sequence of actions, or *plan*, whose execution leads to a state where the goal is satisfied. When a dynamic system is accompanied also by

a plan, it is called a *plan verification problem*. The plan verification *succeeds* when the given plan is a solution to the planning problem.

Note that the system of Example 1 has *knowledge-producing* actions $listen_k$. This kind of action does not necessarily change the physical state of the world, but is able to change the epistemic state of the agent. This means that we allow incomplete state descriptions: for example, it may be the case that the agent does not know what is behind each door. In this case, to avoid be killed he must perform listening actions and then, based on the information acquired at execution time, decides which door to open. Such scenarios thus require that plans be *conditional*, i.e., they branch on the result of the evaluation of some information acquired at execution time.

Our ultimate aim is to provide a formalism for dynamic system descriptions. This formalism should accommodate every significant element for reasoning about actions and plans in scenarios like the one of Example 1. We also intend that this formalism can be used for specification of automated systems. Therefore, we address the question whether an effective inference procedure exists, as well as how efficient it is. The term 'effective' here means that the procedure must be able to decide whether a solution to the problem exists and, if so, return it. And we use the term 'efficient' here to mean that the computational complexity of the procedure should be as lower as possible.

One of the first formalisms used to attain this objective is a dialect of second-order logic proposed by McCarthy (1968), named *situation calculus*. This formalism presented two main problems. First, the corresponding inference procedure is only semi-decidable. Second, it had no satisfactory solution to the frame problem.

The *representational frame problem* was pointed out by McCarthy and Hayes (1969). Roughly, it consists in the impossibility of giving a compact description of a dynamic system. More than twenty years later, Reiter (1991) provided a partial solution to this problem in the situation calculus. However, the associated inference procedure provided is too resource consuming. The problem of designing an efficient inference method for a formalism that solves the frame problem was named the *inferential frame problem* by Thielscher (1999).

One of the main contributions of this thesis is a solution to the inferential frame problem. First, we show that Reiter's solution to the representational frame problem can be encoded in the *dynamic epistemic logic* proposed by van Ditmarsch et al. (2005). We do so by providing a polynomial reduction from situation calculus to dynamic epistemic logic. Then, we provide a new proof method for the latter, whose computational complexity is much lower than the method proposed by Reiter. Moreover, we prove that this proof method is optimal.

Dynamic epistemic logic is very suitable for formalization of the plan verification problem, but not for the planning problem. The reason is that in this setting, plan verification amounts to validity checking, while planning demands the construction of the plan. In other words, it is more or less like if a part of the formula were lacking, and one should find the right piece for completing it. It is the reason why we propose here another logic in which one can, roughly speaking, quantify over epistemic

actions. This is the second main contribution of this thesis. This logic, called *arbitrary public announcement logic*, allows to formally state that 'there are actions that lead to the goal'. We also provide a proof method for it based on tableaux. Our hope is that future extensions of this method will allow planning in dynamic epistemic logic.

This thesis is organized as follows. The next chapter defines the scope of study of this work and presents a short state of the art of the subject. There we present the situation calculus, Reiter's solution to the frame problem and the regression method. In addition, we present a way of encoding that solution in modal logic and also show how to perform regression in modal logic.

The contributions of this thesis are presented from the third chapter onwards. Two results from (Herzig and de Lima, 2006) are presented in Chapter 3. The first one is the *decomposition theorem*. It states that one can encode every action into a sequence of two actions of specific kinds: the first one is purely epistemic and the second one is purely ontic. The second result proves the generality of a specific kind of purely epistemic action. We show that in single-agent scenarios, a simple kind of epistemic action called observation is enough to encode every possible purely epistemic action.

In the fourth chapter, we present the solution to the inferential frame problem that we have proposed in (van Ditmarsch et al., 2007a). We first show how to encode a decidable fragment of situation calculus in dynamic epistemic logic. After that, we extend the polynomial reduction method proposed by Lutz (2006). The method proceeds by a stepwise elimination of dynamic operators, thus reducing the original formula to an epistemic formula. We also consider multiple agents and the common knowledge operator. In all the cases the complexity of our method matches the complexity of the underling epistemic logic. It follows that validity checking is in coNP for the single-agent case, in PSPACE for multiple agents, and in EXPTIME when common knowledge is involved.

The fifth chapter presents an alternative theorem proving method for public announcement logic based on analytic tableaux. The method, proposed in (Balbiani et al., 2007b), is modular, allowing the definition of theorem proving methods for the latter logic where its epistemic logic is K, KT, S4 and S5. We also propose optimal strategies for the first two cases.

The sixth chapter presents arbitrary public announcement logic. This is a logic proposed in (Balbiani et al., 2007a), that allows for quantification over public announcements. A sound and complete axiomatisation is provided.

We draw conclusions and discuss future works in Chapter 7.

Chapter 2

Searching for an Adequate Formalism

2.1 Introduction

This chapter is a short description of the state of the art of the subject of this thesis. We start by an identification of distinct types of reasoning and some scenarios commonly addressed in the current literature on reasoning about actions. In the sequel, we present two different state of the art approaches to reasoning with incomplete information: situation calculus and epistemic dynamic logic. The two approaches turn out to be very similar. A short discussion about these similarities is in the end of the chapter.

2.2 Scope of This Work

2.2.1 Reasoning Tasks

At least three types of reasoning can be identified. Given a sequence of actions, *prediction* is the task of describing the world after their occurrence, and *explanation* is the task of describing the world before their occurrence.[1] For the third task, called *planning*, a current description of the world (the initial state) and a goal are also given. It then consists in finding the sequence of actions whose occurrence in the initial state satisfies the goal.

Here we also identify a fourth task, called *plan verification*: let an initial state, a goal and also a sequence of actions be given. This task consists in checking whether the sequence of actions is a solution to the planning problem.

Note that if one supposes that the set of all states and all action sequences are enumerable, then the possibility of verifying plans implies the possibility of performing

[1] One would perhaps prefer to call these two tasks respectively *deduction* and *abduction*.

the first three tasks. This is the reason why in great part of this work we are mainly concerned about this task. However, an efficient way of realizing the other tasks is commonly desirable. We address planning later on this work, in Chapter 6.

2.2.2 A Taxonomy of Dynamic Systems

We formally define a *dynamic system description* by the tuple $\langle S, A, T\rangle$, where S is a set of states, A is a set of actions, and T is a relation that models actions transitions. Therefore, T can be defined as a function that maps elements of $S \times A$ into S. In this deterministic setting, a *plan verification problem* is the tuple $\langle \Delta, s_0, G\rangle$, were Δ is a dynamic system description, $s_0 \in S$ is the initial state, and $G \subseteq S$ is a set of goal states. A *plan* is a sequence of actions $a_0, a_1, \ldots, a_{n-1}$ of A. If for all $0 \leq i < n$, $T(s_i, a_i) = s_{i+1}$ and $s_n \in G$, then this plan is a *solution* for the associated planning problem and in this case plan verification succeeds, else it fails.

This class of dynamic systems has been studied in several different ways. Notably, three approaches received great attention: state space search, such as described in (Penberthy and Weld, 1992), plan space search, such as in (Blum and Furst, 1997), and logical satisfiability, such as in (Kautz and Selman, 1992, 1999).

The definition of dynamic system description given above can be generalised to accommodate more realistic phenomena. For instance, the transition function T can be defined as a mapping from elements of $S \times A$ into probability distributions over S. Now, it accommodates non-deterministic actions. This means that action results are not predictable, but they are indeed observable. That is, once an action is executed, the agent receives complete and accurate feedback about its results and therefore knows in which state he is. Plan verification and planning problems are redefined accordingly, but solutions are no longer (simple) plans, because the agent must act based on the feedback received at execution time. In this setting, plans must be *conditional*, i.e., they have branches that correspond to decisions to be taken at execution time.

To the latter class, one can still add *partial observability*. That is, the agent does not necessarily perceive all changes caused by the actions. Formally, it can be modelled by providing a set of possible observations Ω and an observation function O. This function maps $A \times S$ into discrete probability distributions over Ω. As the agent cannot be sure in which state he is, the initial state is replaced by a *belief state*, i.e., by a set $b_0 \subseteq S$ of possible initial states. Partially observable planning and plan verification problems are redefined accordingly, and, in this setting, solutions must be conditional plans that map belief states to actions.

Systems where action results are non-deterministic are also called *uncertain*, and are frequently modelled by Markov decision process (MDP), and by partially observable Markov decision process (POMDP). Formalisations of planning problems with Markov decision processes are proposed, for example, by Kaelbling et al. (1998), Boutilier et al. (1999), and Bonet and Geffner (2001).

In this work we study a particular case of the latter class of systems in which actions are deterministic and the agent has *incomplete knowledge* about the initial state. Then the transition relation is defined as in the deterministic case, but the initial state

is a belief state. As for partially observable systems, solutions are conditional plans that map belief states to actions.

2.2.3 Extensional Versus Intensional Representations

Markov decision processes approaches have the disadvantage of requiring *extensional* representations. That is, one must keep in memory the entire belief state space. As this set is huge (the belief state space is exponentially larger than the state space), their representation is infeasible.

One way of avoiding huge representations is the use of *intentional* representations. It amounts to representing states and belief states with sets of multi-valued *fluents*. Let P be a finite set of fluents, each state of S is now labelled by the set of fluents that hold in it. This means that in this approach $S = \mathcal{P}(P)$. Going one step further, some approaches propose the representation of states by logical formulas. For example, suppose that the set of fluents contains $alive$ and $married$ (Example 1). The formula $alive \wedge \neg married$ represents all the states in which the agent is alive and not married. This approach also permits compact representations of the entire system by providing a set Δ of formulas. Allied to an inference procedure, one can address plan verification using theorem proving methods.

These ideas are not new. Logics for reasoning about actions and plans are proposed, for example, by McCarthy (1968), Sandewall (1992, 1995) and Gelfond and Lifschitz (1993). In addition, logical representations also motivated different approaches. For example, some of them replaced theorem proving by satisfiability checking, such as seen above and also in (Majercik and Littman, 2003), or by model checking, such as in (Cimatti et al., 1997).

Current state of the art logics still propose generalisations to multi-agent scenarios. As we use this approach in this work, we are therefore faced to the problem of representing a still larger set of states, since the initial state in this case is a set of belief states. To be compact, the representation of such sets is done using the ideas of Hintikka (1962), also followed by Moore (1980) and many others. They enrich the language with an epistemic operator 'K'. The formula $\mathrm{K}_i lady_1$ represents all the states where agent i "knows" (or "believes") that the lady is behind door 1.

2.2.4 Parsimony and the Frame Problem

The different proposed logics differ in expressivity. Thus, once the decision of using intensional representations is taken, one must choose the most adequate of them. The adequation of a logic is measured by its capability of expressing all desired phenomena and is counterbalanced by the *representational parsimony* of its language. This concept is extensively explored by Shanahan (1997). We outline his ideas here.

First, remember that we now suppose that a system description is a set of formulas Δ written in a logical language. Second, note that this language is defined in terms of its logical operators, the set A of actions, and the set P of fluents. Then, the representational parsimony criterion asserts that, to be considered acceptable, the language must permit complete descriptions that are no larger than $\mathcal{O}(|P| + |A|)$.

In addition, the effort required to add new information has to be proportional to the size of this information. For example, if one wants to add a new action to the domain description that directly affects n fluents, then it should require no more than $\mathcal{O}(n)$ new sentences.

The problem of designing a language that respects this criterion was named the *frame problem*,[2] by McCarthy and Hayes (1969). They showed that one of the most used formalisms for reasoning, the situation calculus, does not respect this criterion. The reason is that in principle domain descriptions must explicit the interaction between every action and fluent. Then such a description obviously has at least $\mathcal{O}(|P| \times |A|)$ formulas.

Since then, some formalisms that avoid the frame problem were proposed. One example is the very restrictive language STRIPS, proposed by Fikes and Nilsson (1971).

More than twenty years later, Reiter (1991) provided a solution to the frame problem in the situation calculus whose language is expressive enough to address most of the desired phenomena. This approach is based on the hypothesis that due to inertia, actions change only a small part of the world and the rest is left unchanged. Then the formalisation defined by Reiter permits that systems be entirely described with $\mathcal{O}(|P| + |A|)$ formulas. This approach was extended latter by Scherl and Levesque (1993) to deal with partially observable domains. However, the associated inference procedure, called *regression*, is too resource consuming. For suppose that φ is a formula whose validity is to be checked. Regression consists in reducing φ to an equivalent formula $\mathrm{reg}(\varphi)$ that does not mention actions. Then, a well-known and more efficient proof method can be used. But $\mathrm{reg}(\varphi)$ can be exponentially larger than φ. Therefore, the computational complexity of the entire method is too high. Other tentatives of solving the frame problem while maintaining high expressiveness were plagued by the same problem, which was named the *inferential frame problem* by Thielscher (1999).

But, what computational complexity should we expect? For example, Bylander (1994) showed that in STRIPS, planning is PSPACE-complete and therefore plan verification is in PTIME. Note however, that this was obtained for that specific language. In the way plan verification and planning were defined, their complexity depends on the complexity of validity, and consequence checking in the logic used. Then the best we can do is to search the least complex formalism that allows the description of the largest class of dynamic systems.

2.3 Situation Calculus

The dialect of second-order logic called *situation calculus* was one of the first formalisms used in reasoning about actions. Proposed by McCarthy (1968), it was not used in practical applications. The main reason of this restriction is the fact that

[2] Here we also call this problem *representational frame problem*, in contrast with the *inferential frame problem*, defined latter on.

there was no satisfactory solution to the representational frame problem, and the corresponding inference procedure is only semi-decidable.

After some decades, Reiter (1991) proposed a partial solution to the frame problem in situation calculus, as well as a procedure for reasoning called *regression*, which is decidable for a sub-class of problems. Since then, situation calculus has being largely used in specification of dynamic systems. Some programming languages, such as GOLOG (Levesque et al., 1997), were built, and the formalism also received a number of extentions such as for concurrent actions (Gelfond et al., 1991), probabilistic (noisy) actions (Bacchus et al., 1999), among many others.

In this section we present syntax and semantics of a decidable fragment of the situation calculus, as well as Reiter's solution to the frame problem and regression. Both were extended later by Scherl and Levesque (1993) to deal with incomplete knowledge. We thus present this extention here and also illustrate its use with the running example given in the introduction. The definitions given here follow those given in (Reiter, 2001a).

2.3.1 Syntax and Semantics

The full language of the situation calculus is a many sorted second-order predicate language with equality. In this work however, we drop functional symbols and restrict the arities of relational fluents to one.

DEFINITION 2 (LANGUAGE $\mathcal{L}_{\mathsf{Sit}}$)
The language $\mathcal{L}_{\mathsf{Sit}}$ is a many sorted second-order predicate language with equality. It has two disjoint sorts: one for situations (or states), denoted by S, one for actions, denoted by A, and one for everything else, denoted by O. Its alphabet contains the operators $\neg$, $\wedge$ and $\forall$, and also the following items:

- countably many predicate symbols, called *relational fluents*, of sort S;
- the functional symbol do of sort $(A \times S) \rightarrow S$;
- the predicate symbols:
  - Rdo of sort $A \times S \times S$,
  - $\sqsubset$ of sort $S \times S$,
  - $Poss$ of sort $A \times S$, and
  - R of sort $S \times S$;
- the constant symbol s_0 of sort S;
- countably many variable symbols of sorts S, A and O; and
- countably many variable symbols for relational fluents and for the predicates of the language.

The constant s_0 is used for designate the *actual* situation. All the other situations are directly or indirectly "linked" to s_0 by do or R. The function do models the transitions associated to actions. $do(a, s)$ is the result of executing action a in situation s. The predicate R is an accessibility relation that models the agent's knowledge. If $R(s', s)$ holds, then the agent considers that s' is possible at s.[3] The predicate $Poss$ models the executability preconditions of actions. If $Poss(a, s)$ holds, then the action a is executable in s.

To simplify notation, we use the classical abbreviations for the operators $\vee$, $\rightarrow$, $\leftrightarrow$ and $\exists$, and also the following one:

$$\mathrm{K}(\varphi^{-1}, s) \stackrel{\text{def}}{=} \forall s'(R(s', s) \rightarrow \varphi^{-1}[s'])$$

where the formula φ^{-1} is φ, but with the removal of the last position argument from all the fluents of φ, and the formula $\varphi[s]$ is φ, but with the introduction of s as the last argument on all its fluents. For example, $\mathrm{K}(lady_1, s)$ abbreviates the formula $\forall s'(R(s', s) \rightarrow lady_1(s'))$, which means that in every possible situation accessible from s, the lady is behind door 1. In other words, the agent knows that there is a lady behind door 1 in situation s.

DEFINITION 3 (Sit-MODEL)
A Sit-*model* is an ordinary Tarski model that satisfies all the formulas in the set Φ_{fdi} of *foundational domain independent axioms*, defined as follows:

$$\begin{aligned}\Phi_{\text{fdi}} = \{&(do(a_1, s_1) = do(a_2, s_2)) \rightarrow (a_1 = a_2 \wedge s_1 = s_2),\\ &\forall p(\forall s(Init(s) \rightarrow p(s)) \wedge \forall a, s(p(s) \rightarrow p(do(a, s))) \rightarrow \forall s(p(s))),\\ &\neg(s \sqsubset s_0),\\ &(s \sqsubset do(a, s')) \leftrightarrow (s \sqsubset s' \vee s = s'),\\ &R(s', s) \rightarrow (Init(s) \leftrightarrow Init(s'))\}\end{aligned}$$

where $Init(s)$ is an abbreviation of $\neg\exists a, s'(s = do(a, s'))$.

Note the similarity between these axioms and Peano Arithmetic. The first axiom eliminates finite cycles and confluence. The second axiom is a second-order induction which guarantees that every situation must be obtained by repeatedly applying the function do. The third and fourth axioms inductively define an ordering $\sqsubset$ in the set S. The second and fifth axioms together define the way situations are connected by the accessibility relation R.

These axioms guarantee that the set of situations form a forest. One of its trees is rooted in the initial situation s_0 and all other roots are related to it by R. Each root represents a possible initial situation according to the knowledge of the agent.

Depending on the definition of knowledge one wants to use, accessibility relations in Sit-models must moreover respect some *relational properties*. They can be among the following:

[3] Note that the "arrow" is from s to s', i.e., the positions of s and s' are inverted.

- Reflexivity (true knowledge):
$$\forall s.R(s,s)$$
- Transitivity (positive introspection):
$$\forall s,s',s''.(R(s',s) \wedge R(s'',s')) \rightarrow R(s'',s)$$
- Symmetry:
$$\forall s,s'.R(s',s) \rightarrow R(s,s')$$
- Euclidicity (negative introspection):
$$\forall s,s',s''.(R(s',s) \wedge R(s'',s)) \rightarrow R(s',s'')$$

From now on, we name Φ_R the set of formulas that describe all the relational properties of R.

The notions of satisfiability, validity and valid consequence are defined as for second-order logic. Let $\varphi \in \mathcal{L}_{\mathsf{Sit}}$, $\Gamma \subseteq \mathcal{L}_{\mathsf{Sit}}$ and M be a Sit-model, we note $M \models \varphi$ for M satisfies φ, $\models_{\mathsf{Sit}} \varphi$ for φ is valid in situation calculus, and $\Gamma \models_{\mathsf{Sit}} \varphi$ for φ is a valid consequence of Γ in situation calculus.

2.3.2 Conditional Plans

DEFINITION 4 (PLANNING TASK IN Sit)
Let a theory formed by the set Φ_{fdi} of foundational domain independent axioms, a system description Δ, and a description of the initial situation I be given. And let a goal formula $\varphi(s)$ with a single free variable s be given. Then the *simple planning task in situation calculus* is to find a sequence of actions $a_1, \ldots, a_n$ of A such that:

$$\Phi_{\text{fdi}} \cup \Delta \cup I \models_{\mathsf{Sit}} \varphi(do(a_n(\ldots(do(a_1, s_0))\ldots)))$$

This definition is not adequate for planning under incomplete knowledge. The reason can be explained using Example 1: in order to keep alive, the agent must avoid opening the tiger's door. But since he does not know in advance behind which door the tiger is, he can only decide what to do at execution time, after evaluating the result of a listening action. Therefore the plan must have some kind of branching.

Branching actions in situation calculus were proposed by Levesque (1996). In fact, he defines a general robot programming language that also contains looping actions. Programs written in this language include both ordinary and sensing actions, and are trivially executable by a machine.

DEFINITION 5 (ROBOT PROGRAM)
A *robot program* (or simply program) is inductively defined as follows:

- nil and $exit$ are robot programs;
- if $a \in A$ and π is a robot program, then $seq(a, \pi)$ is a robot program;

- if a is a binary sensing action and π and π' are both robot programs, then $branch(a, \pi, \pi')$ is a robot program; and
- if π and π' are robot programs, then $loop(\pi, \pi')$ is a robot program.

Similarly to the function do, which returns the result of performing a given action in some situation, it is necessary to say what is meant by performing a program in some situation. It leads to a definition of the predicate Rdo below.

DEFINITION 6 (PREDICATE Rdo)
The predicate $Rdo(\pi, s, s')$ means that π terminates legally when its execution starts in s and s' is the final situation. In other words, it is defined as:

$$Rdo(\pi, s, s') \stackrel{\text{def}}{=} \forall p(\varphi \rightarrow p(\pi, s, s', 1))$$

where φ is the conjunction of the universal closure of the following.

1. Termination normal case:
$$p(nil, s, s, 1)$$
2. Termination loop body:
$$p(exit, s, s, 0)$$
3. Ordinary actions:
$$Poss(a, s) \wedge p(\pi', do(a, s), s', x) \rightarrow p(seq(a, \pi'), s, s', x)$$
4. Sensing actions, true case:
$$Poss(a, s) \wedge sr(a, s) \wedge p(\pi', do(a, s), s', x) \rightarrow p(branch(a, \pi', \pi''), s, s', x)$$
5. Sensing actions, false case:
$$Poss(a, s) \wedge \neg sr(a, s) \wedge p(\pi'', do(a, s), s', x) \rightarrow p(branch(a, \pi', \pi''), s, s', x)$$
6. Loops, exit case:
$$p(\pi', s, s'', 0) \wedge p(\pi'', s'', s', x) \rightarrow p(loop(\pi', \pi''), s, s', x)$$
7. Loops, repeat case:
$$p(\pi', s, s'', 1) \wedge p(loop(\pi', \pi''), s'', s', x) \rightarrow p(loop(\pi', \pi''), s, s', x)$$

Intuitively, $p(\pi, s, s', 0)$ holds when π starts in s and ends in s' with $exit$, and $p(\pi, s, s', 1)$ holds when π starts in s and ends in s' with nil.

DEFINITION 7 (PLANNING TASK IN Sit WITH INCOMPLETE KNOWLEDGE)
Given a theory formed by the foundational domain independent axioms Φ_{fdi}, which includes the definition of Rdo given above, a domain description Δ, a description of the initial situation I, and a goal formula $\varphi(s)$ with a single free variable s, the *planning task in situation calculus under incomplete knowledge* is to find a robot program (or conditional plan) π such that:

$$\Phi_{\text{fdi}} \cup \Delta \cup I \models_{\mathsf{Sit}} \forall s(R(s, s_0) \rightarrow \exists s'(Rdo(\pi, s, s') \wedge \varphi(s')))$$

2.3.3 "Knowledge, Action, and the Frame Problem"

It was using situation calculus that McCarthy and Hayes (1969) pointed out the difficulty imposed by the representational frame problem in reasoning about actions. After that, several partial solutions were proposed, for example by (Haas, 1987), and by (Schubert, 1990). None of them however, is fully satisfactory. We present here yet another partial solution, proposed by Reiter (1991). We also incorporate the extention to incomplete knowledge proposed by Scherl and Levesque (1993) that was redesigned later by Scherl and Levesque (2003). We call this solution the *RSL's partial solution to the frame problem*, or RSL for short.

RSL relies on a number of simplifying hypothesis. The most important are listed below.

- H1 All actions are deterministic.
- H2 Each action is either a purely ontic (i.e., non-knowledge-producing) or a sensing action.
- H3 All the laws that define the behaviour of the actions are known by the agent.
- H4 All action occurrences are perceived by the agent.
- H5 *Action precondition completeness:* for each action constant a, it is possible to give a single formula $\psi_a(s)$ that characterises the condition under which a is executable, where the only free variable in this formula is s.
- H6 *Causal completeness:* for each fluent constant p, it is possible to give a finite set of action constants that may flip the truth value of p.
- H7 *Effect precondition completeness*: for each relational fluent $p(a, s)$, where p and a are constants, it is possible to give a single formula $\psi_p(a, s)$ that characterises all the conditions under which a flips the truth value of p to true (or to false) in the successor situation. Again, the only free variable in these formulas is s.
- H8 The length of the formula $\psi_p(a, s)$ in H7 is roughly proportional to the number of actions that affect the value of the fluent.
- H9 Relatively few actions affect a given fluent.
- H10 *Inertia*: The set of fluents affected by an action is much smaller than the entire set of fluents of the language.

Hypothesis H1 is about the nature of the world. This hypothesis is implemented by the fact that do is a function.

Hypothesis H2 is based on the distinction between ontic actions and epistemic (knowledge-producing) actions: the former modify the world while the latter makes the agent learn facts about the world. This hypothesis is motivated by the assertion that every action can be split in a purely ontic action and in an purely epistemic action. This assertion is folklore in the literature on reasoning about actions (for

example, see Shapiro et al. (2000)). We then suppose from now on, that $A = A_o \cup A_e$, where A_o contains only purely ontic actions and A_e contains only sensing actions.

Hypotheses H3 and H4 say that the agent's knowledge about actions types and about actions instances are accurate. In the case of purely ontic actions $o \in A_o$, they are implemented by the following *successor state axiom for knowledge*:

$$\text{OSSK.} \qquad \forall o, s, s'(R(s', do(o, s)) \leftrightarrow \exists s''(Poss(o, s'') \wedge (s' = do(o, s'')) \wedge R(s'', s)))$$

and in the case of purely epistemic actions, they are implemented by:

$$\text{ESSK.} \qquad \forall e, s, s'.R(s', do(e, s)) \leftrightarrow \exists s''.Poss(e, s'') \wedge (s' = do(e, s'')) \wedge R(s'', s) \wedge (sr(e, s) = sr(e, s''))$$

where the function sr, that ranges over $\{yes, no\}$, is the result of the sensing action e. This function must be defined for each sensing action of A_e. For example, suppose there is an action $sense_p$ whose execution makes the agent know whether a relational fluent p holds or not. Then sr can be defined by:

$$(sr(sense_p, s) = x) \leftrightarrow (((x = yes) \wedge p(s)) \vee ((x = no) \wedge \neg p(s)))$$

Intuitively, OSSK and ESSK ensure that: if after the occurrence of the action a in s the agent considers the situation s' possible, then there exists a situation s'' such that the agent considers possible in s and the occurrence of a in s'' results in s'. This property is called *perfect recall* in (Fagin et al., 1995). In other words, no action is able to make the agent forget facts. In the case of a sensing action e, ESSK moreover says that if the execution of e in s produces some epistemic result when executed in s, given by the function sr, then e produces the same epistemic result when executed in s''.

In addition, it is interesting to note that due to the latter two axioms, some relational properties of R, such as reflexivity, transitivity, symmetry and euclidicity, are true in the set of all situations if and only if they are true in all initial situations.

Scherl and Levesque (2003) also show that SSKs entail several desired properties. One of them is the *no-side-effect property* of epistemic actions, that corresponds to:

$$\forall e, s, p.p(s) \leftrightarrow p(do(e, s))$$

It says that sensing actions do not modify facts about the world.

RSL implements the other six hypotheses by requiring that the action preconditions and effects be described using a collection of formulas in some specific form. Among other things, it requires that these formulas have the property defined in the sequel.

DEFINITION 8 (UNIFORM FORMULAS)
Let s be a term of sort situation. The set of formulas in $\mathcal{L}_{\mathsf{Sit}}$ *uniform in* s, is recursively defined as follows:

- if p is a relational fluent, then $p(s)$ is uniform in s;
- if $\varphi \in \mathcal{L}_{\mathsf{Sit}}$ does not mention a term of sort S, then φ is uniform in s; and
- if φ and ψ are uniform in s, then so are $\neg\varphi$, $\varphi \wedge \psi$ and $\forall x.\varphi$, provided x is a variable not of the sort S.

That is, φ is uniform in s if: it does not mention the predicate $Poss$; it does not mention the predicate $\sqsubset$; it does not quantify over situations; it does not mention equality on situations; and if it mentions a term of the sort situation in the situation argument position of a fluent, then that term is s.

DEFINITION 9 (BASIC ACTION THEORY)
A *basic action theory* is a set of axioms:

$$\Theta = \Phi_{\text{fdi}} \cup \Phi_R \cup \{\text{OSSK}, \text{ESSK}\} \cup \Phi_{\text{una}} \cup \Phi_{\text{ap}} \cup \Phi_{\text{ss}} \cup \Phi_{\text{sf}} \cup \Phi_{s_0}$$

such that:

- Φ_{una} is a set of *unique-name axioms for actions*, that contains a formula of the form:
$$a_1 \neq a_2$$
for each pair of different action names a_1 and a_2;
- Φ_{ap} is a set of *action-precondition axioms*, that contains, for each $o \in A_o$, a single formula of the form
$$\forall a, s.Poss(a, s) \leftrightarrow \psi_a(s)$$
where ψ_a is uniform in s;
- Φ_{ss} is a set of *successor-state axioms*, that describe the effects of purely ontic actions. It contains, for each relational fluent p, a single formula of the form
$$\forall a, s.p(do(a, s)) \leftrightarrow \psi_p(a, s)$$
where $\psi_p(a, s)$ is uniform in s;
- Φ_{sf} is the set of *sensed fluent axioms* describing the effects of sensing actions. It contains, for each sensing action $e \in A_e$, a single formula of the form:
$$\forall s, x.(sr(e, s) = x) \leftrightarrow (((x = yes) \wedge p(s)) \vee ((x = no) \wedge \neg p(s)))$$
where p is the fluent whose truth value is known after the execution of e;[4] and
- Φ_{s_0} is the set of formulas that describes the initial situation. It only contains formulas that are uniform in s_0.

[4]Scherl and Levesque (2003) use an example to explain that it is possible to formalise complex sensing actions. That is, actions whose result is sensing the truth value of more complex formulas. However, a general form of sensed axioms is not provided. It thus remains unclear which are the constraints to be respected in these cases.

The uniformity of $\psi_a(s)$ and $\psi_p(a,s)$ ensures that action preconditions and action effects are completely determined by the current situation s. Moreover, under hypotheses H6–H10 ontic actions only change a small part of the world, leaving the rest unchanged. It follows that $|\Theta| = \mathcal{O}(|P| + |A|)$ (Reiter, 2001a). In addition, note that H9 implies that there is no action changing the truth value of an infinity of fluents. Therefore, RSL is a solution to the representational frame problem.

EXAMPLE 10
We illustrate RSL's solution by showing a formalisation of Example 1. We use the fluent $lady_1$ to mean that the lady is behind door 1. Thus the formula $\neg lady_1$ means that the lady is behind door 2 and the tiger is behind door 1. The first thing to be given is the unique-name axioms for actions:

$$\begin{aligned}\Phi_{\text{una}} = \{&listen_1 \neq listen_2,\\ &listen_1 \neq open_1,\\ &listen_1 \neq open_2,\\ &listen_2 \neq open_1,\\ &listen_2 \neq open_2,\\ &open_1 \neq open_2\ \}\end{aligned}$$

Next, the action-preconditions axioms:

$$\begin{aligned}\Phi_{\text{ap}} = \{&\forall s(Poss(listen_1, s) \leftrightarrow alive(s)),\\ &\forall s(Poss(listen_2, s) \leftrightarrow alive(s)),\\ &\forall s(Poss(open_1, s) \leftrightarrow alive(s)),\\ &\forall s(Poss(open_2, s) \leftrightarrow alive(s))\ \}\end{aligned}$$

Then, the successor-state axioms:

$$\begin{aligned}\Phi_{\text{ss}} = \{\ &\forall a, s(alive(do(a, s)) \leftrightarrow (alive(s) \wedge\\ &\qquad \neg(a = open_1 \wedge \neg lady_1(s)) \wedge\\ &\qquad \neg(a = open_2) \wedge lady_1(s)))),\\ &\forall a, s(married(do(a, s)) \leftrightarrow (((a = open_1) \wedge lady_1(s)) \vee\\ &\qquad ((a = open_2) \wedge \neg lady_1(s)) \vee\\ &\qquad married(s)))),\\ &\forall a, s(lady_1(do(a, s)) \leftrightarrow lady_1(s))\ \}\end{aligned}$$

The sensed fluent axioms:

$$\begin{aligned}\Phi_{\text{sf}} = \{\ &\forall s, x(sr(listen_1, s) = x) \leftrightarrow (((x = yes) \wedge lady_1(s)) \vee\\ &\qquad ((x = no) \wedge \neg lady_1(s))),\\ &\forall s, x(sr(listen_2, s) = x) \leftrightarrow (((x = yes) \wedge \neg lady_1(s)) \vee\\ &\qquad ((x = no) \wedge lady_1(s)))\qquad \}\end{aligned}$$

And finally, the initial situation.

$$\Phi_{s_0} = \{\mathrm{K}(alive \wedge \neg married, s_0)\}$$

Let the goal be described by $\varphi(s) = \mathrm{K}((alive \wedge married), s)$. Then a solution to the planning problem is a robot program π such that:

$$\Theta \models_{\mathsf{Sit}} \forall s(R(s, s_0) \rightarrow \exists s'(Rdo(\pi, s, s') \wedge \varphi^1[s']))$$

For example, the plan $branch(listen_1, seq(open_2, nil), seq(open_1, nil))$ is such a solution.

2.3.4 Situation Calculus Regression

In this section we present an effective procedure for reasoning using RSL's solution. It is based on goal regression, proposed by Waldinger (1977). Roughly, the idea is to start with the goal formula and reason backwards in order to reach the initial state. If the initial state is reachable and the path has been memorised, then one obtains the solution plan.

In situation calculus it corresponds to starting with a complex formula that represents the goal and applying successive "simplifications" in order to obtain an equivalent formula which does not mention actions and whose only situation term mentioned is the initial situation s_0. Once it is done, the foundational domain axioms are no longer necessary to prove that such a formula is entailed by the theory, which reduces the problem to a first-order theorem proving task.

Regression cannot be applied to any formula of $\mathcal{L}_{\mathsf{Sit}}$ though. The following definition ensures that a given goal can be regressed.

DEFINITION 11 (REGRESSABLE FORMULAS)
A formula $\varphi \in \mathcal{L}_{\mathsf{Sit}}$ is *regressable* if and only if:

- each term of the sort situation mentioned by φ has the syntactic form

  $$do(a_n, do(a_{n-1}, \ldots do(a_1, s_0) \ldots))$$

  for some $n \geq 0$, where $a_1, \ldots, a_n$ are terms of the sort action;
- φ does not quantify over situations; and
- φ does not mention the predicate symbol $\sqsubset$, nor does it mention any equality atom $s = s'$ for terms s, s' of sort situation.

In the sequel, we present a definition of the regression operator compiled from Reiter (2001a) and Scherl and Levesque (2003).

DEFINITION 12 (Sit REGRESSION OPERATOR)
Let an action theory Θ be given. The *regression operator* reg_Θ is defined inductively as follows:

1. If φ is a non-fluent atom, including equality atoms; or φ is a fluent atom, or a 'K' fluent whose situation argument is the situation constant s_0, then

$$\mathrm{reg}_\Theta(\varphi) = \varphi$$

2. If Φ_{ap} contains $\forall x(Poss(x) \leftrightarrow \psi_x)$, then

$$\mathrm{reg}_\Theta(Poss(a, s)) = \mathrm{reg}_\Theta(\psi_a(s))$$

3. If p is a fluent (other than 'K') and Φ_{ss} contains $\forall x, y(p(do(x, y)) \leftrightarrow \psi_p(x, y))$, then

$$\mathrm{reg}_\Theta(p(do(a, s))) = \mathrm{reg}_\Theta(\psi_p(a, s))$$

4. If Φ_{sf} contains $\forall y, z((sr(a, y) = z) \leftrightarrow \psi_a(z, y))$, then

$$\mathrm{reg}_\Theta(sr(a, s) = x) = \mathrm{reg}_\Theta(\psi_a(x, s))$$

5. If a is not a knowledge-producing action, then

$$\mathrm{reg}_\Theta(\mathrm{K}(\varphi, do(a, s))) = \mathrm{K}(Poss(a) \rightarrow \mathrm{reg}_\Theta(\varphi[do(a, s')])^{-1}, s)$$

   where s' is a new variable of sort situation.

6. If a is a knowledge-producing action, then

$$\begin{aligned}\mathrm{reg}_\Theta(\mathrm{K}(\varphi, do(a, s))) = \exists y((sr(a, s) = y) \wedge \\ \mathrm{K}((Poss(a) \wedge (sr(a) = y)) \rightarrow \mathrm{reg}_\Theta(\varphi[do(a)])^{-1}, s))\end{aligned}$$

7. If φ and ψ are formulas, then

   (a) $\mathrm{reg}_\Theta(\neg\varphi) \Rightarrow \neg\,\mathrm{reg}_\Theta(\varphi)$

   (b) $\mathrm{reg}_\Theta(\varphi \wedge \psi) \Rightarrow \mathrm{reg}_\Theta(\varphi) \wedge \mathrm{reg}_\Theta(\psi)$

   (c) $\mathrm{reg}_\Theta(\forall x(\varphi)) \Rightarrow \forall x(\mathrm{reg}_\Theta(\varphi))$

That the regression operator can be safely used for reasoning in situation calculus, is guaranteed by the following theorem.

THEOREM 13 (REGRESSION (Reiter, 2001a))
Let Θ be a basic action theory, let φ be a regressable sentence, and let Φ_R be any subset of the accessibility relation properties *reflexive*, *symmetric*, *transitive* and *euclidian*. Then:

$$\Theta \models_{\mathsf{Sit}} \varphi \quad \text{iff} \quad \Phi_{\mathrm{una}} \cup \Phi_R \cup \Phi_{s_0} \models_{\mathrm{FOL}} \mathrm{reg}_\Theta(\varphi)$$

EXAMPLE 14
We use the theory Θ defined in Example 10 to regress the goal:

$$\mathrm{K}((married \wedge alive), do(open_1, do(listen_1, s_0)))$$

For the first step we use the clause 5, that gives us:

$$\begin{aligned}&\mathrm{reg}_\Theta(\mathrm{K}((married \wedge alive), do(open_1, do(listen_1, s_0)))) = \\ &\mathrm{reg}_\Theta(\mathrm{K}(Poss(open_1) \rightarrow \mathrm{reg}_\Theta((married \wedge alive)[do(open_1, s)])^{-1}, do(listen_1, s_0)))\end{aligned}$$

Now, for simplicity, we perform the innermost regression separately:

$$\begin{aligned}&\mathrm{reg}_\Theta((married \wedge alive)[do(open_1, s)]) \\ &= \mathrm{reg}_\Theta(married(do(open_1, s))) \wedge \mathrm{reg}_\Theta(alive(do(open_1, s))) && \text{(by clause 7)} \\ &= \mathrm{reg}_\Theta((open_1 = open_1 \wedge lady_1(s)) \vee \\ &\qquad (open_1 = open_2 \wedge \neg lady_1(s)) \vee \\ &\qquad married(s)) \wedge \\ &\quad \mathrm{reg}_\Theta(alive(s) \wedge \\ &\qquad \neg(open_1 = open_1 \wedge \neg lady_1(s)) \\ &\qquad \neg(open_1 = open_2 \wedge lady_1(s))) && \text{(by clause 3)} \\ &= ((open_1 = open_1 \wedge lady_1(s)) \vee \\ &\quad (open_1 = open_2 \wedge \neg lady_1(s)) \vee \\ &\quad married(s)) \wedge \\ &\quad (alive(s) \wedge \\ &\quad \neg(open_1 = open_1 \wedge \neg lady_1(s)) \vee \\ &\quad \neg(open_1 = open_2 \wedge lady_1(s))) && \text{(by clauses 7, 1 and 3)}\end{aligned}$$

The resultant formula is equivalent to $alive(s) \wedge lady_1(s)$. For simplicity, we now use this shorter equivalent formula and name it ψ. We then continue performing the outermost regression. By clause 6 we have:

$$\begin{aligned}&\mathrm{reg}_\Theta(\mathrm{K}(Poss(open_1) \rightarrow \psi^{-1}, do(listen_1, s_0))) \\ &= \mathrm{reg}_\Theta(\exists y(sr(listen_1, s_0) = x \wedge \mathrm{K}((Poss(listen_1) \wedge sr(listen_1) = x) \rightarrow \\ &\qquad \mathrm{reg}_\Theta((Poss(open_1) \rightarrow \psi^{-1})[do(listen_1, s_0)])^{-1}, s_0)))\end{aligned}$$

Again, we perform the innermost regression separately:

$$
\begin{aligned}
&\mathrm{reg}_\Theta((Poss(open_1) \rightarrow \psi^{-1})[do(listen_1, s_0)]) = \\
&= \mathrm{reg}_\Theta(Poss(open_1, do(listen_1, s_0))) \ \rightarrow \\
&\quad (\mathrm{reg}_\Theta(alive(do(listen_1, s_0))) \wedge \\
&\quad \mathrm{reg}_\Theta(lady_1(do(listen_1, s_0)))) && \text{(by clause 7)} \\
&= alive(s_0) \ \rightarrow \\
&\quad ((alive(s_0) \wedge \\
&\quad \neg(listen_1 = open_1 \wedge lady_1) \wedge \\
&\quad \neg(listen_1 = open_2 \wedge \neg lady_1)) \wedge \\
&\quad lady_1(s_0)) && \text{(by clauses 2 and 3)}
\end{aligned}
$$

This is equivalent to $alive(s_0) \rightarrow lady_1(s_0)$. We continue with the outermost formula:

$$
\begin{aligned}
&\mathrm{reg}_\Theta(\exists x(sr(listen_1, s_0) = x \wedge \mathrm{K}((Poss(listen_1) \wedge sr(listen_1) = x) \ \rightarrow \\
&\qquad\qquad (alive \rightarrow lady_1), s_0))) \\
&= \exists x(\mathrm{reg}_\Theta(sr(listen_1, s_0) = x) \wedge \\
&\qquad \mathrm{reg}_\Theta(\mathrm{K}((Poss(listen_1) \wedge sr(listen_1) = x) \rightarrow (alive \rightarrow lady_1), s_0))) \\
&\text{(by clause 7)} \\
&= \exists x((((x = yes) \wedge lady_1(s_0)) \vee ((x = no) \wedge \neg lady_1(s_0))) \wedge \\
&\qquad \mathrm{K}((alive \wedge (sr(listen_1) = x)) \rightarrow (alive \rightarrow lady_1), s_0) \\
&\text{(by clauses 4, 7, 2 and 1)}
\end{aligned}
$$

As expected, the only situation mentioned by the resulting formula is s_0. This formula characterises a situation in which the execution of $listen_1$ and then of $open_1$ leads to the goal.

2.3.5 A Variant of Situation Calculus

Because situation calculus is defined axiomatically, properties about action theories that are not direct etailments are very hard to prove. An example of this difficulty is the very long proof given by Reiter (2001b) for the fact that if $\mathrm{K}\varphi$ entails $\mathrm{K}\psi \vee \mathrm{K}\chi$ in a theory Θ, then $\mathrm{K}\varphi$ entails $\mathrm{K}\psi$ in Θ, or $\mathrm{K}\varphi$ entails $\mathrm{K}\chi$ in Θ. Aiming at having a "more workable" semantics for situation calculus, Lakemeyer and Levesque (2004) (see also Lakemeyer and Levesque, 2005) proposed a variant of it called ES. This logic is not as expressive as the entire situation calculus, but it handles the action theories defined by Reiter.

The main difference between these two formalisms is the supression of situation terms from the language of ES. The predicate do is replaced by the dynamic operator $[\cdot]$. A formula of the form $[a]\varphi$ is read 'φ holds after action a'. The language of ES

also contains a modal operator 'K',[5] that was an abbreviation in situation calculus. A formula of the form $\mathrm{K}\varphi$ is read 'the agent knows that φ'. In addition, ES contains a new operator $\Box$. The formula $\Box\varphi$ is read 'φ holds after any sequence of actions'.[6]

ES formulas are evaluated in tuples of the form $\langle e, s, \alpha\rangle$ such that:

- $s \in S$ is a function from $P \times A^*$ to $\{0, 1\}$ and from $U \times A^*$ to $P \cup A$, where U is the set of variables of the language. It determines truth values of fluents and co-referring standard names for variables after any sequence of actions in A^*;
- $e \subseteq S$ is the epistemic state of the agent; and
- $\alpha \in A^*$ is a (possibly empty) sequence of actions.

The satisfaction relation is inductively defined by:

$\langle e, s, \alpha\rangle \models t_1 = t_2$	iff	$s(t_1, \alpha)$ is identical to $s(t_2, \alpha)$
$\langle e, s, \alpha\rangle \models p$	iff	$s(p, \alpha) = 1$
$\langle e, s, \alpha\rangle \models \neg\varphi$	iff	not $\langle e, s, \alpha\rangle \models \varphi$
$\langle e, s, \alpha\rangle \models \varphi \wedge \psi$	iff	$\langle e, s, \alpha\rangle \models \varphi$ and $\langle e, s, \alpha\rangle \models \psi$
$\langle e, s, \alpha\rangle \models \forall x.\varphi$	iff	for all std. names n of the right sort, $\langle e, s, \alpha\rangle \models \varphi[x/n]$
$\langle e, s, \alpha\rangle \models \mathrm{K}\varphi$	iff	for all $s' \in e$, $\langle e, s', \alpha\rangle \models \varphi$
$\langle e, s, \alpha\rangle \models [a]\varphi$	iff	$\langle e, s, \alpha \cdot s(a, \alpha)\rangle \models \varphi$
$\langle e, s, \alpha\rangle \models \Box\varphi$	iff	for all $\alpha' \in A^*$, $\langle e, s, \alpha \cdot \alpha'\rangle \models \varphi$

Lakemeyer & Levesque show that the same properties of knowledge as for situation calculus arise from this definition. For example, we have positive introspection, i.e., $\Box(\mathrm{K}\varphi \rightarrow \mathrm{KK}\varphi)$, negative introspection, i.e., $\Box(\neg\mathrm{K}\varphi \rightarrow \mathrm{K}\neg\mathrm{K}\varphi)$, and also the following epistemic successor state axiom for knowledge:

$$\text{SSK.} \qquad \Box([a]\mathrm{K}\varphi \leftrightarrow ((SF(a) \wedge \mathrm{K}(SF(a) \rightarrow [a]\varphi)) \vee (\neg SF(a) \wedge \mathrm{K}(\neg SF(a) \rightarrow [a]\varphi))))$$

where SF is a special rigid predicate of the language such that the truth value of $SF(a)$ is known by the agent after the execution of the action a.

Action theories for ES are defined analogously as for situation calculus. That is, a theory Θ is a set of formulas $\Phi_{\mathrm{una}} \cup \Phi_{\mathrm{ap}} \cup \Phi_{\mathrm{ss}} \cup \Phi_{\mathrm{sf}} \cup \Phi_{s_0}$, such that:

- Φ_{una} contains a formula of the form $a_1 \neq a_2$ for each pair of different action names a_1 and a_2;
- Φ_{ap} contains one formula of the form $\forall a.\Box(poss(a) \leftrightarrow \psi_a)$, where ψ_a is a fluent formula (i.e., a formula that does not mention $Poss$, SF, $\Box$, $[a]$ or 'K');

[5] ES also have the 'only knows' operator 'OK'. It allows, for instance, to infer more about the ignorance of the agent. However, this operator is not taken into account by regression and its translation to situation calculs is not provided. Therefore we decided to not include it in this short analysis.

[6] The logic ES also contains non-rigid predicates and second-order quantification. We do not pay attention to these here.

- Φ_{ss} contains formulas of the form $\forall a.\Box([a]p \leftrightarrow \psi_p(a))$, where each $\psi_p(a)$ is a fluent formula;
- Φ_{sf} contains formulas of the form $\forall a.\Box([a]SF(a) \leftrightarrow \chi_a)$, where each χ_a is a fluent formula; and
- Φ_{s_0} is a set of fluent formulas without free variables that describe the initial situation.

Then, using axiom SSK above, the definition of the regression procedure for ES is very similar to that for situation calculus (note that regressable formulas in ES do not mention the operator $\Box$). Let ϵ be an empty sequence of actions and let an action theory Θ be given. The regression operator reg_Θ is inductively defined on φ as follows:

1. $\text{reg}_\Theta(\alpha, t_1 = t_2) = (t_1 = t_2)$;
2. $\text{reg}_\Theta(\alpha, p)$ for a fluent p is inductively defined on α by:
   (a) $\text{reg}_\Theta(\epsilon, p) = p$,
   (b) $\text{reg}_\Theta(\alpha \cdot a, p) = \text{reg}_\Theta(\alpha, \psi_p(a))$;
3. $\text{reg}_\Theta(\alpha, SF(a)) = \text{reg}_\Theta(\alpha, \chi_a)$;
4. $\text{reg}_\Theta(\alpha, poss(a)) = \text{reg}_\Theta(\alpha, \psi_a)$;
5. $\text{reg}_\Theta(\alpha, \neg\varphi) = \neg\,\text{reg}_\Theta(\alpha, \varphi)$;
6. $\text{reg}_\Theta(\alpha, \varphi_1 \wedge \varphi_2) = \text{reg}_\Theta(\alpha, \varphi_1) \wedge \text{reg}_\Theta(\alpha, \varphi_2)$;
7. $\text{reg}_\Theta(\alpha, \forall x.\varphi) = \forall x.\,\text{reg}_\Theta(\alpha, \varphi)$;
8. $\text{reg}_\Theta(\alpha, \text{K}\varphi)$ is defined inductively on α by:
   (a) $\text{reg}_\Theta(\epsilon, \alpha) = \text{K}(\text{reg}_\Theta(\epsilon, \varphi))$,
   (b) $\text{reg}_\Theta(\alpha \cdot a, \text{K}\varphi) = \text{reg}_\Theta(\alpha, (SF(a) \wedge \text{K}(SF(a) \rightarrow [a]\varphi)) \vee (\neg SF(a) \rightarrow \text{K}(\neg SF(a) \rightarrow [a]\varphi)))$;
9. $\text{reg}_\Theta(\alpha, [a]\varphi) = \text{reg}_\Theta(\alpha \cdot a, \varphi)$.

Thus, ES uses quantification over actions to solve the representational frame problem (just as RSL's solution), but has not situation terms just as modal logics of actions and modal logics of knowledge.

2.4 Epistemic Logic

From this point on, all logics used in this thesis extend epistemic logics in a way or in another. We call *epistemic logics*, a family of modal logics that use possible worlds semantics to represent agent's knowledge or beliefs. This idea, originally proposed by Hintikka (1962), has been largely developed in more recent works (Fagin et al., 1995), (Meyer and van der Hoek, 1995) and (van Ditmarsch et al., 2007b).

DEFINITION 15 (LANGUAGES $\mathcal{L}_{\mathsf{ELC}}$, $\mathcal{L}_{\mathsf{EL}}$ AND $\mathcal{L}_{\mathsf{PL}}$)
Let P be a countable set of propositional letters, and let N be a finite set of agents. The language of epistemic logic with common knowledge $\mathcal{L}_{\mathsf{ELC}}$ is defined by the following BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathrm{K}_i\varphi \mid \mathrm{C}_G\varphi$$

where p ranges over P, i ranges over N, and G ranges over $\mathcal{P}(N)$. We also define the language of epistemic logic without common knowledge $\mathcal{L}_{\mathsf{EL}}$ as the language obtained from $\mathcal{L}_{\mathsf{ELC}}$ by dropping the operator 'C'; and the language of propositional logic $\mathcal{L}_{\mathsf{PL}}$ as the language obtained from $\mathcal{L}_{\mathsf{EL}}$ by dropping the operator 'K'.

The formula $\mathrm{K}_i\varphi$ is read 'agent i knows (or believes) that φ', and the formula $\mathrm{C}_G\varphi$ is read 'all agents in group G commonly know (or believe) that φ'. We use the common abbreviations for the operators $\top$, $\bot$, $\vee$, $\rightarrow$, $\leftrightarrow$ and '$\widehat{\mathrm{K}}$'. The later is the dual of 'K', defined by $\widehat{\mathrm{K}}_i\varphi \stackrel{\text{def}}{=} \neg\mathrm{K}_i\neg\varphi$. And we define the 'everybody knows' operator 'E' as:

$$\mathrm{E}_G\varphi \stackrel{\text{def}}{=} \bigwedge_{i\in G} \mathrm{K}_i\varphi$$

where G ranges over $\mathcal{P}(N)$. In addition, let 'M'be one of the operators 'K_i', 'E_G' or 'C_G'. We sometimes write $\mathrm{M}^\ell\varphi$ to abbreviate $\mathrm{M}\ldots\mathrm{M}\varphi$, were M is repreated ℓ times for $\ell \geq 0$, and $\mathrm{M}^0\varphi$ is simply φ.

The formula $\mathrm{E}_G\varphi$ is read 'every agent in group G knows (or believes) that φ'. We indeed need the operator 'C' of common knowledge (or common belief) to represent, for example, that something is a convention in a group of agents. In this sense, saying that something is known by everybody in the group of agents G is different from saying that something is commonly known in the group of agents G. As a classical example, consider the following, proposed by von Wright (1951):

Every driver must drive on the right side of the road. (2.1)

It was Lewis (1969) that first noted that even if every driver knows (2.1), it is still possible that some driver will rationally choose to drive on the left side of the road. To see this, suppose that there are only two drivers (agents), i.e., $N = \{i, j\}$. Also suppose that everybody knows (2.1), and let us note it $\mathrm{E}_N p$. Now, it may be the case that the agent i does not know that agent j knows p, i.e., it may be the case that $\neg\mathrm{K}_i\mathrm{K}_j p$. In this case i considers possible that j drives on the left side. Then, i may decide to drive on the left side, because he does not feel really safe about driving on the right side. Lewis goes further and also show that it is not enough that, in

addition, everybody knows that everybody knows p. That is, it is not enough to have $\mathrm{E}_N\mathrm{E}_N p$, because it may be the case that agent i does not know that agent j knows that i knows p, i.e., it may be the case that $\neg\mathrm{K}_i\mathrm{K}_j\mathrm{K}_i p$. In other words, i considers possible that j considers possible that i drives on the left side. Then i considers possible that j drives on the left side. Then i may decide to drive on the left side. Reasoning by induction, Lewis concludes that for any $k \in \mathbb{N}$, it is not enough having $\mathrm{E}_N p \wedge \mathrm{E}_N^2 p \wedge \cdots \wedge \mathrm{E}_N^k p$. To be a convention, we should have an infinite conjunction. That is, the formula $\mathrm{C}_G\varphi$ should be equivalent to $\mathrm{E}_G\varphi \wedge \mathrm{E}_G^2\varphi \wedge \ldots$ ad infinitum. The later is equivalent to the following recursive definition $\mathrm{C}_G\varphi \leftrightarrow \mathrm{C}_G(\varphi \wedge \mathrm{E}_G\varphi)$. It follows that the operator 'C' is a fix point operator that adds expressivity to the language. Its meaning can also be captured by the semantics given below.

DEFINITION 16 (EPISTEMIC MODEL)
An *epistemic model* is a tuple $\langle S, R, V\rangle$, where:

- S is a nonempty set of possible worlds;
- $R : N \to \mathcal{P}(S \times S)$ associates an accessibility relation R_i to each $i \in N$; and
- $V : P \to \mathcal{P}(S)$ associates an interpretation $V_p \subseteq S$ to each $p \in P$.

Let $M = \langle S, R, V\rangle$ be an epistemic model and let $s \in S$, we call the pair (M, s) a *pointed epistemic model*. For convenience, we define $R_i(s)$ as the set $\{s' \mid (s, s') \in R_i\}$.

DEFINITION 17 (SATISFACTION RELATION)
Let (M, s) be a pointed epistemic model. The *satisfaction relation* '$\models$' between pointed epistemic models and formulas in $\mathcal{L}_{\mathsf{ELC}}$ is inductively defined as follows:

$$
\begin{array}{lll}
M, s \models p & \text{iff} & s \in V_p \\
M, s \models \neg\varphi & \text{iff} & \text{not } M, s \models \varphi \\
M, s \models \varphi \wedge \psi & \text{iff} & M, s \models \varphi \text{ and } M, s \models \psi \\
M, s \models \mathrm{K}_i\varphi & \text{iff} & R_i(s) \subseteq [\![\varphi]\!]_M \\
M, s \models \mathrm{C}_G\varphi & \text{iff} & (\bigcup_{i \in G} R_i)^+(s) \subseteq [\![\varphi]\!]_M
\end{array}
$$

where $[\![\varphi]\!]_M = \{s \in S \mid M, s \models \varphi\}$ is the extension of φ in the model M, and the operator '$+$' in the last clause means transitive closure.

In the pointed epistemic model (M, s), the distinguished world s is interpreted as the *actual* world, and the set $R_i(s)$ is the set of worlds that agent i considers possible at s. Then a formula φ is known by agent i if and only if φ holds in all possible worlds that are accessible for agent i, i.e., iff φ holds in all worlds of $R_i(s)$. For the operator 'C' we use the transitive closure of relations R_i. Then a formula φ is commonly known by all agents, if and only if φ holds in all possible worlds that are "reachable" from s.

A formula $\varphi \in \mathcal{L}_{\mathsf{ELC}}$ is ELC-valid if and only if for all pointed epistemic models (M, s), $(M, s) \models \varphi$. All ELC-valid formulas can be derived in the Hilbert-style axiomatisation given below:

PL. all tautologies from classical propositional logic
K. $\mathrm{K}_i(\varphi \to \psi) \to (\mathrm{K}_i\varphi \to \mathrm{K}_i\psi)$
C. $\mathrm{C}_G\varphi \to \bigwedge_{i\in G} \mathrm{K}_i(\varphi \to \mathrm{C}_G\varphi)$
MP. From $\vdash_{\mathsf{ELC}} \varphi$ and $\vdash_{\mathsf{ELC}} \varphi \to \psi$, infer $\vdash_{\mathsf{ELC}} \psi$
N. From $\vdash_{\mathsf{ELC}} \varphi$, infer $\vdash_{\mathsf{ELC}} \mathrm{K}_i\varphi$
I. From $\vdash_{\mathsf{ELC}} \varphi \to \mathrm{E}_G(\varphi \wedge \psi)$, infer $\vdash_{\mathsf{ELC}} \varphi \to \mathrm{C}_G\psi$

Note that in the axiomatisation given above, from the validity $\mathrm{K}_i\varphi$, we cannot derive φ. In other words, when we say in the logic that an agent "knows" φ, φ does not necessarily hold in the actual world. That is, the operator 'K' does not respect the so-called 'true knowledge' property, and is therefore more accepted for the definition of belief. To have 'true knowledge', one should impose that for all formulas φ, $\mathrm{K}_i\varphi \to \varphi$ be valid in all epistemic models. This can be done by adding one more axiom scheme to the axiomatisation of the logic, or by requiring that each relation R_i be reflexive. Similarly, other properties can be required for knowledge, according to the definition of knowledge one wants to use. Below, we list the most common properties present in the recent literature of reasoning about knowledge:

- Reflexivity (or 'true knowledge'): for all $s \in S$, $s \in R_i(s)$. Axiom scheme:

$$\mathrm{T.} \quad \mathrm{K}_i\varphi \to \varphi$$

- Seriality: for all $s \in S$, there exists $s' \in R_i(s)$. Axiom scheme:

$$\mathrm{D.} \quad \mathrm{K}_i\neg\varphi \to \neg\mathrm{K}_i\varphi$$

- Symmetry: for all $s, s' \in S$, if $s' \in R_i(s)$, then $s \in R_i(s')$; Axiom scheme:

$$\mathrm{B.} \quad \mathrm{K}_i\varphi \to \mathrm{K}_i\neg\mathrm{K}_i\neg\varphi$$

- Transitivity (or 'positive introspection'): for all $s, s', s'' \in S$, if $s' \in R_i(s)$ and $s'' \in R_i(s')$, then $s'' \in R_i(s)$; Axiom scheme:

$$4. \quad \mathrm{K}_i\varphi \to \mathrm{K}_i\mathrm{K}_i\varphi$$

and

- Euclidicity (or 'negative introspection'): for all $s, s', s'' \in S$, if $s' \in R_i(s)$ and $s'' \in R_i(s)$, then $s'' \in R_i(s')$. Axiom scheme:

$$5. \quad \neg\mathrm{K}_i\varphi \to \mathrm{K}_i\neg\mathrm{K}_i\varphi$$

A *class of epistemic models*, is the set of all epistemic models that respect a subset of the properties above. Below, we list the classes that we address in this thesis and their respective names.

- K: no restrictions;

- KT: each R_i is reflexive;
- S4 (or KT4): each R_i is reflexive and transitive;
- KD45: each R_i is serial, transitive and euclidian; and
- S5 (or KT5): each R_i is reflexive and euclidian.

We note two things. First, K is also the class of all epistemic models. Second, we have the following relations between the above classes: S5 $\subset$ S4 $\subset$ KT $\subset$ K, and also S5 $\subset$ S4 $\subset$ KD45 $\subset$ K. From now on whenever an epistemic model is in K, we call it a K-model. Similarly, we also use the terms KT-, KD45-, S4-, and S5-model.

DEFINITION 18 (VALIDITY, SATISFIABILITY AND VALID CONSEQUENCE)
Let $\mathsf{C} \in \{\mathsf{K}, \mathsf{KT}, \mathsf{S4}, \mathsf{KD45}, \mathsf{S5}\}$, and let $\Psi \subseteq \mathcal{L}_{\mathsf{ELC}}$. A formula $\varphi \in \mathcal{L}_{\mathsf{ELC}}$ is:

- C-*valid* (noted: $\models_{\mathsf{C}} \varphi$) if and only if for all pointed C-models (M, s), $(M, s) \models \varphi$;
- C-*satisfiable* if and only if $\not\models_{\mathsf{C}} \neg\varphi$; and
- a *valid* C-*consequence* of Ψ (notation: $\Psi \models_{\mathsf{C}} \varphi$) if and only if for all pointed C-models (M, s), if for all $\psi \in \Psi$, $(M, s) \models \psi$, then $(M, s) \models \varphi$.

Because of the relations between classes of models mentioned above, we obviously have that an ELC-valid formula is also K-, KT-, S4-, KD45- and S5-valid. We therefore abuse notation and write $\models_{\mathsf{ELC}} \varphi$ (instead of $\models_{\mathsf{K}} \varphi$) to mean that φ is valid in all classes of epistemic models.

Before closing this section, we list several known complexity results for epistemic logics. All of them were shown by Halpern and Moses (1992). Without the operator 'C' (i.e., for $\mathcal{L}_{\mathsf{EL}}$), satisfiability checking is:

- NP-complete for $|N| = 1$ in KD45 and S5;
- PSPACE-complete for $|N| \geq 2$ in KD45 and S5; and
- PSPACE-complete for any number of agents in K, KT and S4.

With the operator 'C' (i.e., for $\mathcal{L}_{\mathsf{ELC}}$), satisfiability checking is:

- PSPACE-complete for $|N| = 1$ in S4, KD45 and S5;
- EXPTIME-complete for $|N| \geq 2$ in S4, KD45 and S5; and
- EXPTIME-complete for any number of agents in K and KT.

2.5 Epistemic Dynamic Logic

Epistemic dynamic logic (EDL) is a "union" of epistemic logic with propositional dynamic logic (PDL). The latter is presented, for example, in (Harel et al., 2000). EDL was proposed by Herzig et al. (2000a) as an alternative formalism for reasoning about actions and plans.[7] As situation calculus, this logic uses modal operators 'K' to represent agent's knowledge. But instead of the function do, EDL leaves situations implicit and uses modal operators $\langle\cdot\rangle$. The formula $\langle a\rangle\varphi$ means that φ holds after the execution of a.

In this section we show how EDL can be used to model dynamic systems by following the same "path" as in Section 2.3. That is, we start with its syntax and semantics, and then we define meaningful plans, a solution to the frame problem, and finish with the definition of a regression operator.

2.5.1 Syntax and Semantics

DEFINITION 19 (LANGUAGE $\mathcal{L}_{\mathsf{EDL}}$)
Let P be a countable set of propositional letters, let A be a countable set of (abstract) action letters and let $N = \{i\}$. The language of epistemic dynamic logic $\mathcal{L}_{\mathsf{EDL}}$ is the set of formulas φ defined by the following BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathrm{K}_i\varphi \mid [\alpha]\varphi$$
$$\alpha ::= a \mid ?\varphi \mid \alpha\,;\alpha \mid \alpha \cup \alpha$$

where p ranges over P, and a ranges over A.

We use the common abbreviations for the operators $\top$, $\bot$, $\vee$, $\rightarrow$, $\leftrightarrow$, and $\langle\cdot\rangle$. The latter is defined by $\langle a\rangle\varphi \stackrel{\text{def}}{=} \neg[a]\neg\varphi$.

In this logic, contrarily to situation calculus, situation terms are not part of the language, for they are left implicit. For instance, the intended meaning of a formula φ that does not mention the modal operators $[\cdot]$ is that φ holds in the initial situation. The formula $\mathrm{K}_i\varphi$ is read 'the agent knows (or believes) that φ'. In other words, φ holds in all situations that the agent considers possible at the initial one. That is, it has the same intuitive meaning as the formula $\mathrm{K}_i(\varphi^{-1}, s_0)$ of the situation calculus. The formula $[\alpha]\varphi$ is read 'φ holds after every possible execution of α'. Then the formula $\langle a\rangle\varphi$ has the same intuitive meaning as $\varphi^{-1}[do(a, s_0)]$ in situation calculus. Note however that determinism is not imposed by the language of EDL.

This language also defines *action operators*. The operator '?' is read 'if',[8] the operator ';' is read 'and then' (or 'sequence') and the operator $\cup$ is 'nondeterministic choice'. Note however that EDL does not provide a constructor for modelling loops. The PDL operator '$*$', normally used for that is not present in this logic. In some

[7] A similar (yet more expressive) logic was proposed by Koons and Asher (1994).
[8] In PDL literature, this action is normally called 'test' (Harel et al., 2000).

situations, the impossibility of this kind of construction can be seen as a strong limitation. But, as pointed out by Castilho et al. (1997), the version of PDL without '$*$' is strongly complete,[9] while full PDL is not.

DEFINITION 20 (EDL-MODEL)
An EDL-*model* is a tuple $\langle S, R, T, V\rangle$, such that:

- S is a nonempty set of possible worlds (also called states or situations);
- $R \subseteq (S \times S)$ is a reflexive and euclidian relation;
- $T : A \to (S \times S)$ associates a binary transition relation $T_a \subseteq (S \times S)$ to each $a \in A$;
- $V : P \to \mathcal{P}(S)$ associates an interpretation $V_p \subseteq S$ to each $p \in P$; and
- R and each T_a respect the *interaction constraint*: $(T_a \circ R) \subseteq (R \circ T_a)$.

The relation R models the agent's knowledge. For convenience we also define $R(s) = \{s' \mid (s, s') \in R\}$. Then, the set $R(s)$ is the set of worlds that the agent considers possible at s. Similarly, the relation T_a models the transition relation associated to the abstract action a. Letting $T_a(s) = \{s' \mid (s, s') \in T_a\}$, $T_a(s)$ is the set of possible results of the execution of a in s.

The interaction constraint ensures that the worlds that the agent considers possible after the execution of a, i.e., $(T_a \circ R)(s)$, are a subset of the outcomes of a in the worlds that the agent considers possible at s, i.e., $(R \circ T_a)(s)$. In other words, all worlds in $(T_a \circ R)(s)$ have an antecedent. This is the EDL version of perfect recall, named *no-forgetting* by its authors.

Let M be an EDL-model as defined above and let $s \in S$, we call the pair (M, s) a *pointed* EDL-*model*. The distinguished world s is intuitively interpreted as the *actual* world (or the initial situation).

DEFINITION 21 (SATISFACTION RELATION)
Let (M, s) be a pointed EDL-model, the *satisfaction relation* '$\models$' between pointed EDL-models and formulas in $\mathcal{L}_{\mathsf{EDL}}$ is inductively defined as follows:

$$
\begin{array}{lll}
M, s \models p & \text{iff} & s \in V_p \\
M, s \models \neg\varphi & \text{iff} & \text{not } M, s \models \varphi \\
M, s \models \varphi \wedge \psi & \text{iff} & M, s \models \varphi \text{ and } M, s \models \psi \\
M, s \models \mathrm{K}_i\varphi & \text{iff} & R(s) \subseteq [\![\varphi]\!]_M \\
M, s \models [a]\varphi & \text{iff} & T_a(s) \subseteq [\![\varphi]\!]_M
\end{array}
$$

where $[\![\varphi]\!]_M = \{s' \mid M, s' \models \varphi\}$ is the *extension* of φ in the model M.

[9] A proof system is (weakly) complete if and only if $\models \varphi$ implies $\vdash \varphi$. And it is strongly complete if and only if for all pairs (Ψ, φ), $\Psi \models \varphi$ implies $\Psi \vdash \varphi$.

DEFINITION 22 (VALIDITY, SATISFIABILITY AND VALID CONSEQUENCE)
Let $\Psi \subseteq \mathcal{L}_{\mathsf{EDL}}$. A formula $\varphi \in \mathcal{L}_{\mathsf{EDL}}$ is:

- EDL-*valid* (notation: $\models_{\mathsf{EDL}} \varphi$) if and only if for all pointed EDL-models (M, s), $(M, s) \models \varphi$;
- EDL-*satisfiable* if and only if $\not\models_{\mathsf{EDL}} \neg\varphi$; and
- a *valid* EDL-*consequence* of Ψ (notation: $\Psi \models_{\mathsf{EDL}} \varphi$) if and only if for all pointed EDL-models (M, s), if for all $\psi \in \Psi$, $(M, s) \models \psi$, then $(M, s) \models \varphi$.

Note that there is no clause for the action operators. Let α_1, α_2 be complex actions. Formulas containing complex actions are interpreted by means of the following abbreviations:

$$[?\varphi]\psi \stackrel{\text{def}}{=} \varphi \to \psi$$
$$[\alpha_1 ; \alpha_2]\varphi \stackrel{\text{def}}{=} [\alpha_1][\alpha_2]\varphi$$
$$[\alpha_1 \cup \alpha_2]\varphi \stackrel{\text{def}}{=} [\alpha_1]\varphi \wedge [\alpha_2]\varphi$$

For convenience we moreover define the following action operators:

$$\mathsf{skip} \stackrel{\text{def}}{=} ?\top$$
$$\mathsf{if}\ \varphi\ \mathsf{then}\ \alpha_1\ \mathsf{else}\ \alpha_2 \stackrel{\text{def}}{=} (?\varphi ; \alpha_1) \cup (?\neg\varphi ; \alpha_2)$$

The intuitive interpretation of the operator skip is 'do nothing', and the construction if φ then α_1 else α_2 means 'if φ holds, then execute α_1, else execute α_2'.

It follows from standard results in modal logic on Sahlqvist formulas (Blackburn et al., 2001a) that all EDL-valid formulas can be derived in the following Hilbert-style axiomatisation:

PL.	all tautologies from classical propositional logic
$\mathrm{K}(\mathrm{K}_i)$.	$\mathrm{K}_i(\varphi \to \psi) \to (\mathrm{K}_i\varphi \to \mathrm{K}_i\psi)$
$\mathrm{T}(\mathrm{K}_i)$.	$\mathrm{K}_i\varphi \to \varphi$
$5(\mathrm{K}_i)$.	$\neg\mathrm{K}_i\varphi \to \mathrm{K}_i\neg\mathrm{K}_i\varphi$
$\mathrm{K}([a])$.	$[a](\varphi \to \psi) \to ([a]\varphi \to [a]\psi)$
NF.	$\mathrm{K}_i[a]\varphi \to [a]\mathrm{K}_i\varphi$
MP.	From $\vdash_{\mathsf{EDL}} \varphi$ and $\vdash_{\mathsf{EDL}} \varphi \to \psi$, infer $\vdash_{\mathsf{EDL}} \psi$
$\mathrm{N}(\mathrm{K}_i)$.	From $\vdash_{\mathsf{EDL}} \varphi$, infer $\vdash_{\mathsf{EDL}} \mathrm{K}_i\varphi$
$\mathrm{N}([a])$.	From $\vdash_{\mathsf{EDL}} \varphi$, infer $\vdash_{\mathsf{EDL}} [a]\varphi$

Axioms $\mathrm{K}(\mathrm{K}_i)$ and $\mathrm{K}(a)$ are standard axioms for epistemic and dynamic logics respectively. Axiom $\mathrm{T}(\mathrm{K}_i)$ corresponds to reflexivity of relation R: everything that the agent knows is true. This is also called *true knowledge* in the literature of reasoning about knowledge. Axiom $5(\mathrm{K}_i)$ corresponds to euclidicity of R: if the agent does not know something, then he knows that he does not know it. It is also called *negative introspection*. Axioms $\mathrm{T}(\mathrm{K}_i)$ and $5(\mathrm{K}_i)$ together imply transitivity of R, which

corresponds to the scheme $\mathrm{K}_i\varphi \rightarrow \mathrm{K}_i\mathrm{K}_i\varphi$: if the agent knows something, then he knows that he knows it. This latter property is also called *positive introspection*. Finally, axiom NF corresponds to 'no-forgetting': if the agent knows in advance that after the execution of a φ holds, then he indeed knows φ after the execution of a.

THEOREM 23 (Herzig et al. (2000a))
Validity checking in EDL is PSPACE-hard and consequence checking is EXPTIME-hard.

This result follows from the fact that EDL extends epistemic logic, where validity checking is PSPACE-complete and consequence checking is EXPTIME-complete (cf. Section 2.4).

EXAMPLE 24
We reuse Example 1 to see how to write down a system description in EDL. The following formulas are examples of *effect laws*, i.e., they describe the result of the execution of the abstract actions in A.

$$lady_1 \rightarrow [listen_1]\mathrm{K}_i lady_1$$
$$lady_1 \rightarrow [open_1]married$$
$$\neg lady_1 \rightarrow [open_1]\neg alive$$

Below, some examples of *executability laws*. These formulas describe the conditions under which the abstract actions in A are executable.

$$\langle listen_1\rangle\top \leftrightarrow alive$$
$$\langle open_1\rangle\top \leftrightarrow alive$$

And now, some examples of so-called *frame axioms*. That is, formulas that describe everything that does not change by the execution of each action.

$$lady_1 \rightarrow [listen_1]lady_1 \qquad \neg lady_1 \rightarrow [listen_1]\neg lady_1$$
$$alive \rightarrow [listen_1]alive \qquad \neg alive \rightarrow [listen_1]\neg alive$$
$$married \rightarrow [listen_1]married \qquad \neg married \rightarrow [listen_1]\neg married$$
$$lady_1 \rightarrow [open_1]lady_1, \qquad \neg lady_1 \rightarrow [open_1]\neg lady_1$$
$$\neg alive \rightarrow [open_1]\neg alive \qquad married \rightarrow [open_1]married$$

2.5.2 Meaningful Plans

Let the set Δ be a system description containing formulas such as in Example 24. Let an initial situation be described by the formula φ_0 and a goal be described by the formula ψ be given. The simple plan verification task in EDL can be defined analogously as in situation calculus. Given a complex action α, this is the task of verifying whether the following holds:

$$\Delta \models_{\mathsf{EDL}} \varphi_0 \rightarrow \langle\alpha\rangle\psi$$

For instance, it holds for $\varphi_0 = alive \wedge \neg married$, $\psi = alive \wedge married$ and $\alpha = listen_1 ; ((?\neg lady_1 ; open_2) \cup (?lady_1 ; open_1))$. Note however, that similarly to situation calculus this definition of plan verification is not adequate. As claimed by Levesque (1996), "the agent needs to know how to execute the program". For example, the agent may not know whether $lady_1$ holds or not. Then, similarly to the conditional plans defined in Section 2.3.2 the complex actions in EDL must be restricted.

DEFINITION 25 (LANGUAGE $\mathcal{L}_{\mathsf{EDL}m}$)
Let P be a countable set of propositional letters, let A be a countable set of (abstract) action letters, and let $N = \{i\}$. The language of *epistemic dynamic logic with meaningful plans* $\mathcal{L}_{\mathsf{EDL}m}$ is the set of formulas φ defined by the following BNF:

$$\begin{aligned}\varphi &::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathrm{K}_i\varphi \mid [\pi]\varphi\\ \pi &::= a \mid \mathsf{skip} \mid \pi\,;\pi \mid \mathsf{if}\ \tau\ \mathsf{then}\ \pi\ \mathsf{else}\ \pi\\ \tau &::= \mathrm{K}_i\varphi \mid \neg\tau \mid \tau \wedge \tau \mid \mathrm{K}_i\tau\end{aligned}$$

where p ranges over P and a ranges over A.

In this definition the formulas τ are the *epistemically interpretable formulas*. That is, the formulas whose truth value can be evaluated by the agent. Then, the complex actions π are the *meaningful plans*. That is, the plans that can be executed by the agent.

In addition, to define plan verification correctly, the descriptions of the initial situation and the goal must also be redefined. Then, let Δ be a system description, φ_0 be an epistemically interpretable formula that describes the initial situation, $\psi \in \mathcal{L}_{\mathsf{PL}}$ be a goal description, and let π be a meaningful plan. Then *plan verification in* EDL is the task of verifying whether the following holds:

$$\Delta \models_{\mathsf{EDL}} \varphi_0 \rightarrow \langle\pi\rangle\psi$$

Validity checking in EDL is PSPACE-hard. Nevertheless, the computational complexity of plan verification is much lower.

THEOREM 26 (Herzig et al. (2000a))
Plan verification in EDL is Π_2^p-complete.

2.5.3 The Dependence Relation

The logic defined does not respect the parsimony criterion. As shown in Example 24, system descriptions require all the frame axioms. In order to decrease their size one can, for example, take the approach of Castilho et al. (1997), described in this section.

DEFINITION 27 (DEPENDENCE RELATION)
A *dependence relation* Dep is a binary relation between primitive actions and literals. That is, $Dep \subseteq (A \times L)$, where the set $L = P \cup \{\neg p \mid p \in P\}$ is the set of literals, and where P is the set of propositional letters of the language.

Note that Dep is not in the object language, but in the metalanguage. It is used to restrict the models of EDL.

DEFINITION 28 (Dep-MODEL)
Let Dep be a dependence relation, a Dep-model is an EDL-model such that for all $s, s' \in S$ and for all $a \in A$, if $s' \in T_a(s)$, then:

- $(a, p) \notin Dep$ and $s \notin V_p$ implies $s' \notin V_p$; and
- $(a, \neg p) \notin Dep$ and $s \in V_p$ implies $s' \in V_p$.

The satisfaction relation and also the corresponding notions of Dep-validity (notation: $\models_{Dep} \varphi$), valid Dep-consequence (notation: $\Psi \models_{Dep} \varphi$) and Dep-satisfiability are redefined for Dep-models similarly as for EDL-models (see Definition 21).

Syntactically, the additional constraint of the above definition corresponds to the following axiom scheme:

$$\text{Pre}([a]). \quad \neg l \rightarrow [a]\neg l \text{ if } (a, l) \notin Dep$$

EXAMPLE 29
For our running example, the dependence relation Dep can be defined as follows:

$$\{(open_1, married), (open_1, \neg alive), (open_2, married), (open_2, \neg alive)\}$$

Note that all the frame axioms of Example 24 are now valid. For suppose that, e.g., the pointed Dep-model (M, s) does not satisfy the formula $alive \rightarrow [listen_1]alive$. Then, we have that $(M, s) \models alive$ and also that there exists $s' \in T_{listen_1}(s)$ such that, $(M, s') \models \neg alive$. But, because $(listen_1, \neg alive) \notin Dep$, we have that for all $s' \in T_{listen_1}(s)$, $(M, s') \models alive$, which leads to a contradiction.

It is argued by Castilho et al. (1997) that these sets can be expected to be much smaller than $|P| \times |A|$. Therefore, the dependence relation can be used to solve the representational frame problem.

2.5.4 Modal Logic Regression

The partial solution to the frame problem in modal logics, proposed by Demolombe et al. (2003), is presented in this section. Similarly to RSL, it makes several simplifying assumptions that, in fact, correspond to almost all the assumptions made in Section 2.3.3. Nevertheless, some of them must be adapted to the present framework. Hypotheses H1, H3, H4, H9 and H10 of page 13 stay the same. The other ones are replaced by:

H2 All actions are purely ontic.

H5 *Action precondition completeness:* for each $a \in A$, it is possible to give a single formula $Poss(a) \in \mathcal{L}_{\mathsf{PL}}$ which describes the executability precondition of a. Moreover, it is supposed that this precondition is complete: if $Poss(a)$ does not hold, then the action a is not executable.

H6 *Causal completeness:* for each $p \in P$, it is possible to provide two finite sets $Cause^+(p) \subseteq A$ and $Cause^-(p) \subseteq A$, which contain the positive and negative possible "causes" of p respectively. Moreover, it is supposed that these two sets are complete: if $a \notin Cause^+(p)$, then the execution of a can never make p true. Symmetrically, if $a \notin Cause^-(p)$, then the execution of a can never make p false.

H7,H8 *Effect precondition completeness:* for each $p \in P$ and each $a \in Cause^+(p)$, it is possible to give a single formula $\gamma^+(a,p) \in \mathcal{L}_{\mathsf{PL}}$ which describes the *positive effect precondition of p by a*. Symmetrically for each $p \in P$ and each $a \in Cause^-(p)$, it is possible to give a single formula $\gamma^-(a,p) \in \mathcal{L}_{\mathsf{PL}}$ which describes the *negative effect precondition of p by a*. Moreover, it is supposed that these effect preconditions are complete: if $\gamma^+(a,p)$ does not hold, then the execution of a can never make p true. Symmetrically, if $\gamma^-(a,p)$ does not hold, then the execution of a can never make p false.

Hypothesis H1 is implemented by the following axiom scheme:

$$\text{Det.} \quad \neg[a]\varphi \rightarrow [a]\neg\varphi$$

Hypotheses H3, H4 are implemented by the successor state axiom scheme for knowledge below.

$$\text{OSSK.} \quad [a]\mathrm{K}_i\varphi \leftrightarrow (\langle a\rangle\top \rightarrow \mathrm{K}_i[a]\varphi)$$

EDL does not have an ESSK because of H2. Based on results of Herzig et al. (2000b), the authors of this solution argue that it does not represent "much loss of generality". In fact, we establish in Section 3.3 a stronger result that supports this assumption. Then we leave to that section the discussion about the generality of the axiom OSSK, and as a consequence, the generality of the regression method presented in this section.

The other hypotheses are implemented as follows. Suppose that all the $Poss(a)$, $Cause^+(p)$, $Cause^-(p)$, $\gamma^+(a,p)$, $\gamma^-(a,p)$ are given. Then we define the dependence relation and the set Δ_g of global axioms as follows:

- for every $a_i \in Cause^+(p)$ put (a_i, p) in Dep, and for every $a_j \in Cause^-(p)$ put $(a_j, \neg p)$ in Dep;
- for every $a \in A$, add the following executability axiom to Δ_g:

$$Poss(a) \leftrightarrow \langle a\rangle\top$$

- for every $p \in P$ and every $a_i \in Cause^+(p)$, add the following two effect axioms to Δ_g:

$$\gamma^+(a_i,p) \rightarrow [a_i]p$$
$$(\neg\gamma^+(a_i,p) \wedge \neg p) \rightarrow [a_i]\neg p$$

- for every $p \in P$ and every $a_j \in Cause^-(p)$ add the following two effect axioms to Δ_g:

$$\gamma^-(a_j, p) \rightarrow [a_j]\neg p$$
$$(\neg\gamma^-(a_j, p) \wedge p) \rightarrow [a_j]p$$

All together results in the following.

THEOREM 30 (Demolombe et al. (2003))
Let Dep and Δ_g be obtained from the sets $Poss(a)$, $Cause^+(p)$, $Cause^-(p)$, $\gamma^+(a,p)$ and $\gamma^-(a,p)$, then

- if $(a,p) \notin Dep$ and $(a,\neg p) \notin Dep$, then
  $\{\text{Det}, \text{OSSK}\} \cup \Delta_g \models_{Dep} [a]p \leftrightarrow (Poss(a) \rightarrow p)$.
- if $(a,p) \notin Dep$ and $(a,\neg p) \in Dep$, then
  $\{\text{Det}, \text{OSSK}\} \cup \Delta_g \models_{Dep} [a]p \leftrightarrow (Poss(a) \rightarrow (p \wedge \neg\gamma^-(a,p)))$.
- if $(a,p) \in Dep$ and $(a,\neg p) \notin Dep$, then
  $\{\text{Det}, \text{OSSK}\}\Delta_g \models_{Dep} [a]p \leftrightarrow (Poss(a) \rightarrow \gamma^+(a,p) \vee p)$.
- if $(a,p) \in Dep$ and $(a,\neg p) \in Dep$, then
  $\{\text{Det}, \text{OSSK}\} \cup \Delta_g \models_{Dep} [a]p \leftrightarrow (Poss(a) \rightarrow \gamma^+(a,p) \vee (p \wedge \neg\gamma^+(a,p)))$.

This result gives us what is necessary to define the regression operator for EDL.

DEFINITION 31 (EDL REGRESSION OPERATOR)
Let the dependence relation Dep be given. The *regression operator* reg_{Dep} is inductively defined as follows:

1. $\text{reg}_{Dep}(p) = p$;
2. $\text{reg}_{Dep}(\neg\varphi) = \neg\,\text{reg}_{Dep}(\varphi)$;
3. $\text{reg}_{Dep}(\varphi \wedge \psi) = \text{reg}_{Dep}(\varphi) \wedge \text{reg}_{Dep}(\psi)$;
4. $\text{reg}_{Dep}(\text{K}_i\varphi) = \text{K}_i\,\text{reg}_{Dep}(\varphi)$;
5. $\text{reg}_{Dep}([a]p) = Poss(a) \rightarrow p$,
   if $(a,p) \notin Dep$ and $(a,\neg p) \notin Dep$;
6. $\text{reg}_{Dep}([a]p) = Poss(a) \rightarrow (p \wedge \neg\gamma^-(a,p))$,
   if $(a,p) \notin Dep$ and $(a,\neg p) \in Dep$;
7. $\text{reg}_{Dep}([a]p) = Poss(a) \rightarrow (\gamma^+(a,p) \vee p)$,
   if $(a,p) \in Dep$ and $(a,\neg p) \notin Dep$;
8. $\text{reg}_{Dep}([a]p) = Poss(a) \rightarrow (\gamma^+(a,p) \vee (p \wedge \neg\gamma^-(a,p)))$,
   if $(a,p) \in Dep$ and $(a,\neg p) \in Dep$;

9. $\mathrm{reg}_{Dep}([a]\neg\varphi) = \neg(Poss(a) \rightarrow \mathrm{reg}_{Dep}(\varphi))$;
10. $\mathrm{reg}_{Dep}([a](\varphi \wedge \psi)) = \mathrm{reg}_{Dep}([a]\varphi) \wedge \mathrm{reg}_{Dep}([a]\psi)$;
11. $\mathrm{reg}_{Dep}([a]\mathrm{K}_i\varphi) = (Poss(a) \rightarrow \mathrm{K}_i\,\mathrm{reg}_{Dep}([a]\varphi))$.

THEOREM 32 (EDL REGRESSION)
Suppose that Δ_g and Dep are obtained from $Poss$, $Cause^+$, $Cause^-$, γ^+ and γ^- as defined above. Then $\mathrm{reg}_{Dep}(\varphi) \in \mathcal{L}_{\mathsf{EL}}$ and $\Delta_g \models_{Dep} \varphi$ if and only if $\models_{\mathsf{S5}} \mathrm{reg}_{Dep}(\varphi)$.

EXAMPLE 33
To illustrate regression we reuse Example 1 and apply the operator reg_{Dep} to a very simple formula $\varphi = [open_1]\mathrm{K}_i(alive \wedge married)$. First though, it is necessary to provide the system description Δ as required by the method. Note that, in fact, it is not necessary to provide details about every action, since there is only one action in the formula φ. But for the sake of the illustration the descriptions of all actions are given. We start by the action preconditions:

$$Poss(listen_1) = Poss(listen_2) = Poss(open_1) = Poss(open_2) = alive$$

Next, we provide the sets of possible causes:

$$Cause^+(married) = \{open_1, open_2\} \qquad Cause^-(married) = \emptyset$$
$$Cause^+(alive) = \emptyset \qquad Cause^-(alive) = \{open_1, open_2\}$$
$$Cause^+(lady_1) = \emptyset \qquad Cause^-(lady_1) = \emptyset$$

And then, the effect preconditions:

$$\gamma^+(open_1, married) = lady_1 \qquad \gamma^+(open_2, married) = \neg lady_1$$
$$\gamma^-(open_1, alive) = \neg lady_1 \qquad \gamma^-(open_2, alive) = lady_1$$

The dependence relation Dep is then generated as follows (cf. Example 29):

$$\{(open_1, married), (open_2, married), (open_1, \neg alive), (open_2, \neg alive)\}$$

Note that actions $listen_1$ and $listen_2$ are not purely ontic. That is, by performing, e.g., $listen_1$, the agent learns something. Therefore, the regression method just defined cannot be performed for this action.[10] But for the ontic actions $open_1$ and $open_2$ it works:

$$\begin{aligned}
&\mathrm{reg}_{Dep}([open_1]\mathrm{K}_i(alive \wedge married) = \\
=&Poss(open_1) \rightarrow \mathrm{K}_i\,\mathrm{reg}_{Dep}([open_1](alive \wedge married)) \\
=&Poss(open_1) \rightarrow \mathrm{K}_i(\mathrm{reg}_{Dep}([open_1]alive) \wedge \mathrm{reg}_{Dep}([open_1]married)) \\
=&Poss(open_1) \rightarrow \mathrm{K}_i((alive \wedge lady_1) \wedge (lady_1 \wedge married))
\end{aligned}$$

Note that the resulting formula holds in the states where the goal $(alive \wedge married)$ is reachable by only executing $open_1$, as expected.

[10] In fact, the listening actions of this example could be decomposed into two ontic actions. The way of doing this is explained in (Herzig et al., 2000b). We go into more details about this in Section 3.3. Therefore, we postpone the discussion until there.

2.6 Discussion and Conclusion

At this point, the similarities between the formalisms presented in Sections 2.3 and 2.5 became apparent. It shows that EDL can be used instead of situation calculus (or logic ES). Note in addition that although sometimes claimed as an "essential feature", quantification over actions is not really necessary, as we saw in Section 2.5.

Then the choice between all these logics seems to be a matter of taste. In the rest of the work however, we will see that it is not as simple as that. Being a modal logic, EDL is more easily related to other modal logics independently developed in the "dynamic logic tradition". These logics have a long tradition in the investigation of concepts such as common knowledge, relativised common knowledge, public and private observations, and others that are interesting in multi-agent scenarios.

This, added to the fact that we prefer the simpler syntax and semantics of EDL, without situation terms or quantification, made us choose the latter as the formalism in which we continue our investigations.

Chapter 3

A Framework for Epistemic and Ontic Change

3.1 Introduction

In this chapter, we address mono-agent environments where all action laws are known and events are public (hypotheses H3 and H4 of page 13). Our aim is to design a simple framework able to deal with scenarios where the agent does not have complete information about the world. This framework is in terms of epistemic dynamic logic (EDL), presented in Section 2.5.1.

We first show that in EDL every action can be decomposed into an epistemic action followed by an ontic action. We then show that epistemic actions can be reduced to sequences of observations. In the end of the chapter we show that the latter kind of action is closely related to public announcements of public announcement logic (Plaza, 1989).

3.2 The Separation Theorem

Roughly, purely ontic actions stand for actions that do not involve any perception: the agent only knows that action a has been performed, without learning about its (possibly nondeterministic or conditional) effects. On the other hand, purely epistemic actions cannot change facts about the world. Typical examples are sensing actions (testing whether a proposition is true or not) and observations (learning that a proposition is true). The definitions below make precise this distinction.

DEFINITION 34 (PURELY ONTIC ACTION)
The action $o \in A$ is *purely ontic* in an EDL-model $\langle S, R, T, V \rangle$, if and only if T_o satisfies the two properties below.

- *Epistemic determinism*: if $t, u \in T_o(s)$, then $R(t) = R(u)$.

- *No-learning*: if $t \in (R \circ T_o)(s)$ and $T_o(s) \neq \emptyset$, then $t \in (T_o \circ R)(s)$.

Epistemic determinism corresponds to the fact that the agent cannot distinguish between nondeterministic outcomes of an action: whether the coin falls heads or tails, the agent only knows that a coin has been tossed and that the disjunction holds. No-learning corresponds to the fact that if the agent considers that t is a possible outcome of execution of o in s, then the agent keeps on considering t to be a possible world after o. Syntactically, for a given system description Δ, o is purely ontic if and only if the two following properties hold for all $\varphi \in \mathcal{L}_{\mathsf{EDL}}$ (Herzig et al., 2000b):

$$\begin{array}{ll} \text{EDet.} & \Delta \models_{\mathsf{EDL}} \langle o \rangle \mathrm{K}_i \varphi \rightarrow [o] \mathrm{K}_i \varphi \\ \text{NL.} & \Delta \models_{\mathsf{EDL}} [o] \mathrm{K}_i \varphi \rightarrow ([o]\bot \vee \mathrm{K}_i [o] \varphi) \end{array}$$

DEFINITION 35 (PURELY EPISTEMIC ACTION)
The action $e \in A$ is *purely epistemic* in an EDL-model $\langle S, R, T, V \rangle$, if and only if it satisfies the following property.

- *Preservation*: if $t \in T_e(s)$, then for all $p \in P$, ($s \in V_p$ iff $t \in V_p$).

Preservation corresponds to the fact that epistemic actions do not change the world. Syntactically, e is purely epistemic in the models of a system description Δ if and only if the following hold for all $\varphi \in \mathcal{L}_{\mathsf{PL}}$ (Herzig et al., 2000b):

$$\text{Pre.} \quad \Delta \models_{\mathsf{EDL}} \varphi \rightarrow [e] \varphi$$

Now, we show that these two kinds of actions are all we need: every transition relation can be decomposed appropriately.

THEOREM 36 (SEPARATION)
Let $a \in A$ and let $\varphi \in \mathcal{L}_{\mathsf{EDL}}$. The formula φ is EDL-satisfiable if and only if there exist actions o and e such that: o is purely ontic, e is purely epistemic, and $\varphi[a/(e;o)]$ (the formula obtained by replacing a by $e;o$ in φ) is EDL-satisfiable.

PROOF SKETCH. From right to left is obvious.
From left to right is established by introducing intermediate worlds that correspond to the outcome of action o. ∎

It enables us to make a partition in the set A of abstract actions. From now on, it is formed by the union of two disjoint sets: A_e of purely epistemic and A_o of purely ontic actions. In addition, the conditions just given can become now axiom schemes in our framework. Therefore, in addition to the axioms schemes of EDL, given in Section 2.5.1, our framework also have the following ones. Let $\psi \in \mathcal{L}_{\mathsf{PL}}$, $\varphi \in \mathcal{L}_{\mathsf{EDL}}$, $e \in A_e$, and let $o \in A_o$:

$$\begin{array}{ll} \text{EDet}(o). & \langle o \rangle \mathrm{K}_i \varphi \rightarrow [o] \mathrm{K}_i \varphi \\ \text{NL}(o). & \mathrm{K}_i [o] \varphi \rightarrow ([o]\bot \vee \mathrm{K}_i [o] \varphi) \\ \text{Pre}(e). & \psi \rightarrow [e] \psi \end{array}$$

This decomposition permits a separated analysis of each of these two kinds of actions as we make in the next section.

3.3 How Many Kinds of Epistemic Actions Are There?

We start our analysis by considering the most basic kind of epistemic action: *observations*.[1] It can roughly be understood as an exogenous event that makes the agent observe that φ holds. It is noted $!\varphi$. Then the formula $[!\varphi]\psi$ is read 'ψ holds after all possible observations of φ'. We now consider a variant of EDL where observations is the only kind of epistemic actions allowed. We therefore restrict our attention to the following language.

DEFINITION 37 (LANGUAGE $\mathcal{L}_{\mathsf{EDL}o}$)
The language of the epistemic dynamic logic with observations $\mathcal{L}_{\mathsf{EDL}o}$ is the set of formulas φ defined by the following BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathrm{K}_i\varphi \mid [\alpha]\varphi$$
$$\alpha ::= o \mid !\varphi \mid ?\varphi \mid \alpha\,;\alpha \mid \alpha \cup \alpha$$

where p ranges over P and o ranges over A_o.

DEFINITION 38 (EDL*o*-MODEL)
An EDL*o-model* is an EDL-model such that every $T_{!\varphi}$ satisfies the following constraints:

1. *Preservation*: if $s' \in T_{!\varphi}(s)$, then for all $p \in P$, $s \in V_p$ iff $s' \in V_p$;
2. *Executability*: $M, s \not\models \varphi$ if and only if $T_{!\varphi}(s) = \emptyset$;
3. *Determinism*: if $M, s \models \varphi$, then $T_{!\varphi}(s)$ is a singleton; and
4. *No-forgetting and no-learning*: if $s' \in T_{!\varphi}(s)$, then $R(s') = (R \circ T_{!\varphi})(s)$.

The first constraint says that observations are purely epistemic actions. The second constraint says that truth of φ is a necessary condition for executability of $!\varphi$. The third condition says that observations are deterministic. And the fourth condition says that (according to Definition 34) observations are also purely ontic actions!

It can be shown that the following formulas involving observations are validities in this framework:

Pre(!).	$\psi \to [!\varphi]\psi$	for all $\psi \in \mathcal{L}_{\mathsf{PL}}$
Exe(!).	$\varphi \leftrightarrow \langle!\varphi\rangle\top$	
Det(!).	$\langle!\varphi\rangle\psi \to [!\varphi]\psi$	
NF(!).	$\mathrm{K}_i[!\varphi]\psi \to [!\varphi]\mathrm{K}_i\psi$	
NL(!).	$[!\varphi]\mathrm{K}_i\psi \to ([!\varphi]\bot \vee \mathrm{K}_i[!\varphi]\psi)$	

However, other kinds of epistemic actions exist. For instance, $listen_1$ (Example 1) is not an observation. It is what we call a *test*: given a formula φ it returns whether φ

[1] This action is also called *test that* in some different approaches.

holds or not. We note this kind of action $!!\varphi$, and the formula $[!!\varphi]\psi$ is read 'ψ holds after all possible tests of ψ'. For example, $listen_1$ can be written '$!!lady_1$'.[2]

In fact, the action $listen_1$ is conditional: the test depends on the context. It is noted, $!!\varphi \,\mathsf{cond}\, \psi$, where ψ is the condition for the test whether φ. The same may also be applied to observations. Then, the formula $[!\varphi \,\mathsf{cond}\, \psi]\chi$ is read 'under the condition ψ, χ holds after all possible observations of φ'. For example, $listen_1$ is $!!lady_1 \,\mathsf{cond}\, alive$. We can see these constructions as abbreviations:

$$\langle !\varphi \,\mathsf{cond}\, \psi\rangle\chi \stackrel{\text{def}}{=} \langle !\psi\rangle\langle !\varphi\rangle\chi$$
$$\langle !!\varphi\rangle\chi \stackrel{\text{def}}{=} \langle !\varphi\rangle\chi \vee \langle !\neg\varphi\rangle\chi$$
$$\langle !!\varphi \,\mathsf{cond}\, \psi\rangle\chi \stackrel{\text{def}}{=} \langle !\psi\rangle\langle !!\varphi\rangle\chi$$

We leave to the reader the confirmation that all these definitions match the intuitions behind the actions introduced above.

At least tests are definable in terms of observations. But other kinds of purely epistemic actions may be conceived. We want to have all of them in our logic. The theorem below gives what we want. It says that every purely epistemic action can be defined in terms of observations.

THEOREM 39 (OBSERVATIONS ARE GENERAL)
Suppose that P and A are finite. Let $\varphi \in \mathcal{L}_{\mathsf{EDL}}$ and let e be a deterministic purely epistemic action. The formula φ is satisfiable in finite models if and only if there exists a (complex) observation ϵ such that $\varphi[\epsilon/e]$ is satisfiable in finite models.

PROOF SKETCH. For a detailed proof the reader can see Appendix A.1.
From right to left: since ϵ is purely epistemic, take $T_e = T_\epsilon$.
From left to right: suppose $M, s \models \varphi$. To every $s \in S$, we can associate a characteristic formula $\delta(s)$ such that $M, s \models (s')$ iff s and s' satisfy the same formulas. For a proof the reader can refer to van Benthem (2006). Now, we show that $T_e = T_{\bigcup_{s\in S}(?\delta(s)\,;\gamma(s))}$, where $\gamma(s) = \bigvee_{t\in(T_e \circ R \circ T_e^{-1})(s)} \delta(t)$ by using the fact that: $\forall s, t, t' \in S$, if $t \in T_{!\gamma(s)}(s)$ and $t' \in T_e(s)$ then t and t' satisfy the same epistemic formulas. ∎

In other words, in EDL, sequences and nondeterministic compositions of observations suffice to express every kind of purely epistemic action.

In fact, the operator '!' is very closely related to public announcements of public announcement logic (PAL), originally proposed by Plaza (1989). Both PAL and EDLo have the same set of validities for announcements and '!' operators respectively. In other words, we have the following.

[2]This kind of action is also named *test if* or *sensing* in the literature. Do not confuse this operator with '?', that we call here 'if'.

THEOREM 40
The following schemes are valid in EDLo.

$$[!\varphi]p \leftrightarrow (\varphi \rightarrow p)$$
$$[!\varphi]\neg\psi \leftrightarrow (\varphi \rightarrow \neg[!\varphi]\psi)$$
$$[!\varphi](\varphi \wedge \chi) \leftrightarrow ([!\varphi]\psi \wedge [!\varphi]\chi)$$
$$[!\varphi]\mathrm{K}_i\psi \leftrightarrow (\varphi \rightarrow \mathrm{K}_i[!\varphi]\psi)$$

PROOF. It is in Appendix A.2. ∎

Some recent works (van Ditmarsch and Kooi, 2006a; van Ditmarsch et al., 2005) show that PAL is very suitable for modelling multi-agent communications. In addition, it is possible to incorporate other notions such as belief and common knowledge. PAL is itself a special case of other communication logics proposed, for example, by Baltag and Moss (2004); Gerbrandy (1999); van Ditmarsch et al. (2005). This suggests that EDLo can be extended to handle all these notions.

Chapter 4

Optimal Methods for Reasoning

4.1 Introduction

In this chapter we provide a solution to the inferential frame problem in the propositional case. Motivated by the results given in Section 3.3, among all possible epistemic actions we only consider *observations*: all agents observe *that* some proposition holds in the world, and update their epistemic state accordingly. We give a satisfiability-preserving polynomial transformation eliminating action operators from formulas. This provides an optimal procedure for reasoning about actions: both, in Reiter's approach case (without knowledge operators) and in the Scherl & Levesque's approach case, the decision procedure works in nondeterministic polynomial time. In the multi-agent case it works in polynomial space, and in the presence of common knowledge it works in exponential time. All these results are optimal because they match the complexity of the underlying epistemic logic.

Technically, our approach is built on recent progress in *dynamic epistemic logics* (DELC) (van Ditmarsch et al., 2005; Kooi, 2007). In this family of logics situation terms are left implicit, and there is no quantification over actions.[1] Thus the central device in Reiter's solution is not available. We show that nevertheless one can do without it, and recast this framework in DELC.[2] This logic has two kinds of actions: announcements and assignments. Announcements can be used to model observations, while assignments are enough for modelling world-altering (ontic) actions. DELC being an extension of Plaza's public announcement logic, we extend Lutz' op-

[1] If there were, a dynamic logic formulation of SSAs could be as in logic ES (see Section 2.3.5):

$$\forall x.([x]p \leftrightarrow (poss(x) \rightarrow (\bigvee_{\{a_i : p \in eff^+(a_i)\}} (x = a_i \wedge \gamma^+(a_i, p)) \vee (p \wedge \bigwedge_{\{a'_j : p \in eff^-(a'_j)\}} \neg(x = a'_j \wedge \gamma^-(a'_j, p))))))$$

[2] The same idea is outlined independently by van Benthem (2007). Another effort to bring together the situation calculus and modal logics was done by Blackburn et al. (2001b).

timal decision procedure for the latter (Lutz, 2006) to DELC, and show that we keep optimality: checking satisfiability in DELC is shown to have the same complexity as checking satisfiability in the underlying epistemic logic.

The remainder of the chapter is organised as follows: Sections 4.2 and 4.3 respectively extends the solution to the frame problem proposed in (Demolombe et al., 2003) to epistemic actions, and introduces dynamic epistemic logics. Section 4.4 contains the translation of RSL's approach into dynamic epistemic logic. Section 4.5 contains decision procedures for satisfiability checking in DELC for single-agent and multi-agent cases, as well as for the case of common knowledge.

4.2 Reiter-style Action Theories

In this section we extend the account of Reiter's solution proposed by (Demolombe et al., 2003), where Reiter-style action theories are formulated in a propositional dynamic logic (PDL) framework.

4.2.1 Action Descriptions

Let us make the same simplifying assumptions listed in Section 2.5.4. Also let A be a countable set of action letters (abstract atomic actions), and let a range over A.

DEFINITION 41 (ACTION DESCRIPTION)
An *action description* is a tuple $\langle poss, \mathit{eff}^+, \mathit{eff}^-, \gamma^+, \gamma^- \rangle$ such that:

- $poss : A \rightarrow \mathcal{L}_{\mathsf{ELC}}$ assigns an executability precondition to each action;
- $\mathit{eff}^+ : A \rightarrow \mathcal{P}(P)$ assigns a finite set of possible positive effects to each action;
- $\mathit{eff}^- : A \rightarrow \mathcal{P}(P)$ assigns a finite set of possible negative effects to each action;
- γ^+ is a family of functions $\gamma^+(a) : \mathit{eff}^+(a) \rightarrow \mathcal{L}_{\mathsf{ELC}}$. For each atom p in $\mathit{eff}^+(a)$, it assigns a precondition for the action a making p true; and
- γ^- is a family of functions $\gamma^-(a) : \mathit{eff}^-(a) \rightarrow \mathcal{L}_{\mathsf{ELC}}$. For each atom p in $\mathit{eff}^-(a)$, it assigns a precondition for the action a making p false.

If $\mathit{eff}^+(a) = \mathit{eff}^-(a) = \emptyset$, then a is a purely epistemic action. In the sequel, all epistemic actions are *observations*.

Note that: H3 and H4 make the functions in D not depend on agents; H1 ensures that for any action a, its effect can be characterised by $\gamma^+(a)$ and $\gamma^-(a)$; Finiteness of eff^+ and eff^- is due to H9; and H10 allows to claim that the representational frame problem is solved by such action descriptions.

In addition to H1–H10, we assume:

H11. All $\gamma^+(a,p) \wedge \gamma^-(a,p)$ are inconsistent in ELC.

Remark. Demolombe et al. (2003) restrict the ranges of $poss$, γ^+ and γ^- to formulas in $\mathcal{L}_{\mathsf{PL}}$. We have extended their range to formulas in $\mathcal{L}_{\mathsf{ELC}}$ (with common knowledge). This allows, for example, the description of actions such as 'make a phone call', whose precondition of execution is that the phone number is known.

To illustrate the definition, we introduce a very simple single-agent example involving an ontic action and two epistemic actions.

EXAMPLE 42
A robot does not know whether the light is on or not. The available ontic action is toggling a switch, with:

$$\begin{aligned} poss(toggle) &= \top \\ \mathit{eff}^+(toggle) &= \{light\} \\ \mathit{eff}^-(toggle) &= \{light\} \\ \gamma^+(toggle, light) &= \neg light \\ \gamma^-(toggle, light) &= light \end{aligned}$$

The observations are $oDark$ and $oBright$, with:

$$\begin{aligned} poss(oDark) &= \neg light & \qquad poss(oBright) &= light \\ \mathit{eff}^+(oDark) &= \emptyset & \mathit{eff}^-(oDark) &= \emptyset \\ \mathit{eff}^+(oBright) &= \emptyset & \mathit{eff}^-(oBright) &= \emptyset \end{aligned}$$

4.2.2 Models for an Action Description

Let D be an action description for the action letters in A. Models for D are obtained by adding transition relations to epistemic models (see Section 2.4).

DEFINITION 43 (D-MODEL)
A *D-model* is a tuple $\langle S, R, T, V\rangle$, where $\langle S, R, V\rangle$ is an epistemic model and

- $T : A \to \mathcal{P}(S \times S)$ associates a relation T_a to each $a \in A$.

Letting $T_a(s) = \{s' \mid (s, s') \in T_a\}$, D-models must moreover satisfy the following constraints:

C1. *No-forgetting:* $(T_a \circ R_i)(s) \subseteq (R_i \circ T_a)(s)$.

C2. *No-learning:* if $T_a(s) \neq \emptyset$ then $(R_i \circ T_a)(s) \subseteq (T_a \circ R_i)(s)$.

C3. *Determinism:* if $t_1, t_2 \in T_a(s)$ then $t_1 = t_2$.

C4. *Executability:* $T_a(s) \neq \emptyset$ iff $\langle S, R, V\rangle, s \models poss(a)$.

C5. *Postcondition (ontic):* if $t \in T_a(s)$, then

- $p \notin \mathit{eff}^+(a)$ and $s \notin V_p$ implies $t \notin V_p$;

- $p \notin \mathit{eff}^-(a)$ and $s \in V_p$ implies $t \in V_p$;
- $p \in \mathit{eff}^+(a)$ and $\langle S, R, V\rangle, s \models \gamma^+(a,p)$ implies $t \in V_p$;
- $p \in \mathit{eff}^-(a)$ and $\langle S, R, V\rangle, s \models \gamma^-(a,p)$ implies $t \notin V_p$;
- $p \in \mathit{eff}^+(a)$ and $\langle S, R, V\rangle, s \not\models \gamma^+(a,p)$ and $s \notin V_p$ implies $t \notin V_p$; and
- $p \in \mathit{eff}^-(a)$ and $\langle S, R, V\rangle, s \not\models \gamma^-(a,p)$ and $s \in V_p$ implies $t \in V_p$.

The relation T_a models the transition relation associated to the abstract action a. $T_a(s)$ is the set of possible results of the execution of action a at s. Concerning the constraints, C1 implements H3 and H4. It guarantees that every world in the set $(T_a \circ R_i)(s)$ has an antecedent. This is also called *perfect recall* in (Fagin et al., 1995). In other words, there is no action able to make agents forget facts. C2 is motivated by H3–H1. For epistemic actions learning about the mere occurrence of an observation is sufficient for each agent to make his epistemic state evolve: the execution of an observation action a eliminates the possible worlds where $\mathit{poss}(a)$ is false. C1 and C2 together correspond to Scherl & Levesque's SSA for knowledge in the case of an ontic action. C3 is motivated by H1. C4 defines the condition for an action be executable. C5 corresponds to Reiter's SSA for facts (as opposed to knowledge). Note that its consistency is guaranteed by H11: otherwise there could be a world s where both $\gamma^+(a,p)$ and $\gamma^-(a,p)$ hold, in which case we should have both $t \in V_p$ and $t \notin V_p$ for every $t \in T_a(w)$.

4.2.3 Validity in D-models

We now introduce a combination of epistemic logic and PDL which will be interpreted in D-models. The language $\mathcal{L}_D$ extends $\mathcal{L}_{\mathsf{ELC}}$ with dynamic operators. It is defined by the BNF:

$$\varphi \ ::= \ p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathrm{K}_i\varphi \mid \mathrm{C}_G\varphi \mid [a]\varphi$$

where p ranges over P, i over N, G over $\mathcal{P}(N)$, and a over A.

Remark. The solution presented in Section 2.5.4 supposes that there is only one agent. We do not make this restriction here and, in addition, consider common knowledge.

We define the *satisfaction relation* '$\models$' as for ELC, plus:

$$M, s \models [a]\varphi \quad \text{iff} \quad T_a(s) \subseteq [\![\varphi]\!]_M$$

A formula $\varphi \in \mathcal{L}_D$ is:

- *D-valid* ($\models_D \varphi$) if and only if for all pointed D-models (M, s), $(M, s) \models \varphi$; and
- *D-satisfiable* if and only if $\not\models_D \neg\varphi$.

$$[a]p \leftrightarrow (poss(a) \rightarrow p) \quad \text{if } p \notin \mathit{eff}^+(a) \cup \mathit{eff}^-(a) \tag{4.1}$$
$$[a]p \leftrightarrow (poss(a) \rightarrow (\gamma^+(a,p) \vee p)) \quad \text{if } p \in \mathit{eff}^+(a) \text{ and } p \notin \mathit{eff}^-(a) \tag{4.2}$$
$$[a]p \leftrightarrow (poss(a) \rightarrow (\neg\gamma^-(a,p) \wedge p)) \quad \text{if } p \notin \mathit{eff}^+(a) \text{ and } p \in \mathit{eff}^-(a) \tag{4.3}$$
$$[a]p \leftrightarrow (poss(a) \rightarrow (\gamma^+(a,p) \vee (\neg\gamma^-(a,p) \wedge p))) \quad \text{if } p \in \mathit{eff}^+(a) \cap \mathit{eff}^-(a) \tag{4.4}$$
$$[a]\neg\varphi \leftrightarrow (poss(a) \rightarrow \neg[a]\varphi) \tag{4.5}$$
$$[a](\varphi_1 \wedge \varphi_2) \leftrightarrow ([a]\varphi_1 \wedge [a]\varphi_2) \tag{4.6}$$
$$[a]\mathrm{K}_i\varphi \leftrightarrow (poss(a) \rightarrow \mathrm{K}_i[a]\varphi) \tag{4.7}$$

Table 4.1: Relevant D-validities (cf. Section 2.5.4, Theorem 30)

For our running example we have:

$$\not\models_D [toggle]\mathrm{K}_i light$$
$$\models_D [oDark][toggle]\mathrm{K}_i light$$
$$\models_D \neg\mathrm{K}_i\neg light \rightarrow [toggle]\neg\mathrm{K}_i light$$

Remark. Although epistemic actions do not change the world, note that $[a]poss(a)$ is not D-valid, even if a is an epistemic action. To see this, consider a such that $poss(a)$ is the so-called Moore-sentence: $p \wedge \neg\mathrm{K}_i p$. Then after learning that $p \wedge \neg\mathrm{K}_i p$ holds the agent will know that p, hence $\neg\mathrm{K}_i p$ does not hold any longer.

4.2.4 Modal Logic Regression

Let an action description D be given. Table 4.1 shows a number of valid equivalences. In each of those validities the complexity of the formula under the scope of the dynamic operator $[\cdot]$ decreases from the left to the right of the operator '$\leftrightarrow$'. For formulas without the common knowledge operator this allows for the definition of a procedure reg_D, called regression in (Reiter, 2001a), that repeatedly applies these equivalences until the resulting formula does not contain dynamic operators any more. It follows that for every domain description D and formula φ without operator 'C' we have:

$$\models_D \varphi \quad \text{iff} \quad \models_{\mathsf{ELC}} \mathrm{reg}_D(\varphi)$$

For example, $[toggle]\mathrm{K}_i light$ can be reduced to $poss(toggle) \rightarrow \mathrm{K}_i[toggle]light$ (by 4.7) and then to $\mathrm{K}_i\neg light$ (by 4.4). And the formula $[oDark]\mathrm{K}_i\neg light$ can be reduced to $poss(oDark) \rightarrow \mathrm{K}_i[oDark]\neg light$ (by 4.7) and then to $\neg light \rightarrow \mathrm{K}_i(\neg light \rightarrow \neg light)$ (by 4.1). The latter being ELC-valid, it follows that $\models_D [oDark][toggle]\mathrm{K}_i light$.

Unfortunately, reg_D is a suboptimal procedure because there are formulas such that $\mathrm{reg}_D(\varphi)$ is exponentially larger than φ (Reiter, 2001a, Section 4.6).

4.3 Dynamic Epistemic Logic

A different tradition in modelling knowledge and change has been followed in, for example, Plaza (1989), Baltag et al. (1998) and van Benthem (2006). Logics in this tradition are, e.g., those of van Ditmarsch et al. (2005) and Kooi (2007), which are based on public announcements and public assignments.

4.3.1 Syntax

DEFINITION 44 (LANGUAGES $\mathcal{L}_{\mathsf{DELC}}$ AND $\mathcal{L}_{\mathsf{DEL}}$)
The *language of dynamic epistemic logic with common knowledge* $\mathcal{L}_{\mathsf{DELC}}$ is the set of formulas φ defined by the following BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathrm{K}_i\varphi \mid \mathrm{C}_G\varphi \mid [\varphi]\varphi \mid [\sigma]\varphi$$
$$\sigma ::= \epsilon \mid p := \varphi, \sigma$$

where p ranges over P, i ranges over N, G ranges over $\mathcal{P}(N)$, and ϵ is an empty assignment. Similarly to ELC, we define the language of dynamic epistemic logic without common knowledge $\mathcal{L}_{\mathsf{DEL}}$ as the language obtained from $\mathcal{L}_{\mathsf{DELC}}$ by dropping the operator 'C'.

Again, the formula $[\alpha]\varphi$ is read 'φ holds after all possible executions of α'. When α is the formula φ, we say that it is the public announcement of φ. The action $p := \varphi$ is the public assignment of the truth value of φ to the atom p. For example, $p := \bot$ is a public assignment, and $\mathrm{K}_i[p := \bot]\neg p$ is a formula. When assignments are made in parallel, the same propositional letter can appear only once on the left hand side of the operator ':='. For convenience, we identify complex assignments of the form $(p_1 := \varphi_1, \ldots, p_n := \varphi_n)$ with sets of the form $\{p_1 := \varphi_1, \ldots, p_n := \varphi_n\}$, thus ϵ is identified with $\emptyset$.

The fragment of DELC without assignments is Plaza's public announcement logic with common knowledge (PALC) (Plaza, 1989), whose fragment without common knowledge we note PAL.

Announcements model epistemic actions, while assignments model ontic actions. For example, the epistemic action $oDark$ of Example 42 is modelled as $\neg light$, and the ontic action $toggle$ as the assignment $\sigma_{toggle} = (light := \neg light)$, i.e., the truth value of $light$ is toggled.

4.3.2 Semantics

DEFINITION 45 (SATISFACTION RELATION)
Formulas in $\mathcal{L}_{\mathsf{DELC}}$ are interpreted in epistemic models. The *satisfaction relation* '$\models$' is extended with the following two clauses:

$$M, s \models [\varphi]\psi \quad \text{iff} \quad M, s \models \varphi \text{ implies } M^{\varphi}, s \models \psi$$
$$M, s \models [\sigma]\varphi \quad \text{iff} \quad M^{\sigma}, s \models \varphi$$

$$
\begin{array}{l}
[\varphi]p \leftrightarrow (\varphi \rightarrow p) \\
[\varphi]\neg\psi \leftrightarrow (\varphi \rightarrow \neg[\varphi]\psi) \\
[\varphi](\psi_1 \wedge \psi_2) \leftrightarrow ([\varphi]\psi_1 \wedge [\varphi]\psi_2) \\
[\varphi]\mathrm{K}_i\psi \leftrightarrow (\varphi \rightarrow \mathrm{K}_i[\varphi]\psi) \\
[\sigma]p \leftrightarrow \sigma(p) \\
[\sigma]\neg\varphi \leftrightarrow \neg[\sigma]\varphi \\
[\sigma](\varphi \wedge \psi) \leftrightarrow ([\sigma]\varphi \wedge [\sigma]\psi) \\
[\sigma]\mathrm{K}_i\varphi \leftrightarrow \mathrm{K}_i[\sigma]\varphi
\end{array}
$$

Table 4.2: Relevant DELC-validities.

where M^φ and M^σ are updates of the epistemic model M that are defined as:

$$
\begin{aligned}
M^\varphi &= \langle S^\varphi, R^\varphi, V^\varphi \rangle \\
S^\varphi &= S \cap [\![\varphi]\!]_M \\
R_i^\varphi &= R_i \cap ([\![\varphi]\!]_M \times [\![\varphi]\!]_M) \\
V^\varphi(p) &= V_p \cap [\![\varphi]\!]_M
\end{aligned}
$$

and

$$
\begin{aligned}
M^\sigma &= \langle S, R, V^\sigma \rangle \\
V^\sigma(p) &= [\![\sigma(p)]\!]_M
\end{aligned}
$$

and where $\sigma(p)$ is the formula assigned to p in σ. If there is no such formula, i.e., if there is no $p := \varphi$ in σ, then $\sigma(p) = p$. (In particular $\epsilon(p) = p$ for all p.)

DEFINITION 46 (VALIDITY AND SATISFIABILITY)
A formula $\varphi \in \mathcal{L}_{\mathsf{DELC}}$ is:

- DELC-*valid* (notation: $\models_{\mathsf{DELC}} \varphi$) if and only if for all pointed epistemic models (M, s), $(M, s) \models \varphi$; and
- DELC-*satisfiable* if and only if $\not\models_{\mathsf{DELC}} \neg\varphi$.

For example, $\mathrm{K}_i p \rightarrow [q := p]\mathrm{K}_i q$ is DELC-valid. Some DELC-validities are listed in Table 4.2. When there are no operators 'C' then the equivalences in Table 4.2 obviously allow the definition of a regression procedure $\mathrm{reg}_{\mathsf{DEL}}$, that eliminates dynamic operators from an expression (van Ditmarsch et al., 2005):

$$\models_{\mathsf{DEL}} \varphi \quad \text{iff} \quad \models_{\mathsf{ELC}} \mathrm{reg}_{\mathsf{DEL}}(\varphi)$$

DEL-regression has the same problem of D-regression: the size of the formula $\mathrm{reg}_{\mathsf{DEL}}(\varphi)$ can be exponentially larger than that of φ. An example is the family of

formulas defined in (Lutz, 2006, Theorem 2):

$$\begin{aligned}\varphi_0 &= \top \\ \varphi_{n+1} &= \langle\langle\varphi_n\rangle\neg\mathrm{K}_i\neg\top\rangle\neg\mathrm{K}_j\neg\top\end{aligned}$$

Moreover, no such equivalences exist for the operator 'C' (Baltag et al., 1998).

In the next sections we provide a better solution. The first step is to formally link Reiter-style action descriptions D with DELC.

4.4 From Toronto to Amsterdam

The D-validities presented in Table 4.1 are similar to the DELC-validities presented in Table 4.2. We show in this section that:

- the executability preconditions $poss$ in D can be modelled in DELC as public announcements, because once an action is executed, all the agents now know that it was executable at the previous instant; and
- the changes brought about by actions can be modelled as public assignments.

DEFINITION 47 (TRANSLATION δ_D)
Let an action description D be given. We define the translation δ_D inductively as follows:

$$\begin{aligned}\delta_D(p) &= p \\ \delta_D(\neg\varphi) &= \neg\delta_D(\varphi) \\ \delta_D(\varphi\wedge\psi) &= \delta_D(\varphi)\wedge\delta_D(\psi) \\ \delta_D(\mathrm{K}_i\varphi) &= \mathrm{K}_i\delta_D(\varphi) \\ \delta_D(\mathrm{C}_G\varphi) &= \mathrm{C}_G\delta_D(\varphi) \\ \delta_D([a]\varphi) &= [poss(a)][\sigma_a]\delta_D(\varphi)\end{aligned}$$

where σ_a is the complex assignment:

$$\begin{aligned}&\{p := \gamma^+(a,p)\vee p \mid p\in \mathit{eff}^+(a) \text{ and } p\notin \mathit{eff}^-(a)\} \quad\cup \\ &\{p := \neg\gamma^-(a,p)\wedge p \mid p\notin \mathit{eff}^+(a) \text{ and } p\in \mathit{eff}^-(a)\} \quad\cup \\ &\{p := \gamma^+(a,p)\vee(\neg\gamma^-(a,p)\wedge p) \mid p\in \mathit{eff}^+(a)\cap \mathit{eff}^-(a)\}\end{aligned}$$

Note that $\delta_D(a)$ is well-defined because H9 guarantees that $\mathit{eff}^+(a)$ and $\mathit{eff}^-(a)$ are finite.

For example, $\delta_D([oDark]\neg light) = [\neg light][\epsilon]\neg light$, which is equivalent to $\top$ (remember that ϵ is the empty assignment); and also:

$$\delta_D([toggle]\neg light) = [\top][light := \neg light\vee(\neg light\wedge light)]\neg light$$

which is equivalent to $light$.

We now show that this translation is polynomial. First we define the function 'len' that returns the *length* of a given expression. In the case of sets and tuples, we count the length of each element and also the commas and delimeters. That is, the length of a given set X is:

$$\mathrm{len}(X) = 1 + \sum_{x \in X} (1 + \mathrm{len}(x))$$

while for a given tuple $Y = \langle y_1, \ldots, y_n \rangle$ it is

$$\mathrm{len}(Y) = 1 + \sum_{k=1}^{n} (1 + \mathrm{len}(y_k))$$

For formulas in $\mathcal{L}_{\mathsf{ELC}}$, we use the recursive definition that follows (note that G is a set):

$$\begin{aligned}
\mathrm{len}(p) &= 1 \\
\mathrm{len}(\neg\varphi) &= 1 + \mathrm{len}(\varphi) \\
\mathrm{len}(\varphi \wedge \psi) &= 1 + \mathrm{len}(\varphi) + \mathrm{len}(\psi) \\
\mathrm{len}(\mathrm{K}_i\varphi) &= 2 + \mathrm{len}(\varphi) \\
\mathrm{len}(\mathrm{C}_G\varphi) &= 1 + \mathrm{len}(G) + \mathrm{len}(\varphi)
\end{aligned}$$

For formulas in $\mathcal{L}_D$ we also use:

$$\mathrm{len}([a]\varphi) = 2 + \mathrm{len}(\varphi)$$

and for formulas in $\mathcal{L}_{\mathsf{DELC}}$ we also use (we consider σ as a set of assignments):

$$\begin{aligned}
\mathrm{len}([\varphi]\psi) &= 1 + \mathrm{len}(\varphi) + \mathrm{len}(\psi) \\
\mathrm{len}([\sigma]\varphi) &= 1 + \mathrm{len}(\sigma) + \mathrm{len}(\varphi) \\
\mathrm{len}(p := \varphi) &= 2 + \mathrm{len}(\varphi)
\end{aligned}$$

For example, $\mathrm{len}([\{p := q, q := p \wedge q\}]\mathrm{K}_i p = 1 + \mathrm{len}(\{p := q, q := p \wedge q\}) + \mathrm{len}(\mathrm{K}_i p) = 12 + 2 + 1 = 15$.

LEMMA 48 (POLYNOMIAL TRANSLATION)
Let D be a finite Reiter-style action description and let $\varphi \in \mathcal{L}_D$. Then $\mathrm{len}(\delta_D(\varphi)) \leq \mathcal{O}(\mathrm{len}(D) \times \mathrm{len}(\varphi))$.

PROOF. It is given in Appendix A.3. ∎

Now, the following correspondence holds (see also Tables 4.1 and 4.2).

THEOREM 49
Let D be a Reiter-style action description and let $\varphi \in \mathcal{L}_D$. Then φ is D-satisfiable if and only if $\delta_D(\varphi)$ is DELC-satisfiable.

PROOF. It is given in Appendix A.4. ∎

Hence D-satisfiability is polynomially reduced to DELC-satisfiability.

4.5 Optimal Regression

We now give a polynomial satisfiability-preserving reduction from DELC to ELC. The idea is first eliminate assignments, and then apply Lutz' reduction (Lutz, 2006) to eliminate announcements.

4.5.1 Eliminating Assignments

We apply a technique that is fairly standard in automated theorem proving (Nonnengart and Weidenbach, 2001).

THEOREM 50 (ASSIGNMENT ELIMINATION)
Let $[p_1 := \varphi_1, \ldots, p_n := \varphi_n]\psi$ be a subformula of a formula χ in $\mathcal{L}_{\mathsf{DELC}}$. Let ψ' be obtained from ψ by substituting every occurrence of p_k by x_{p_k}, where x_{p_k} is a new propositional letter not occurring in χ. Let χ' be obtained from χ by replacing $[p_1 := \varphi_1, \ldots, p_n := \varphi_n]\psi$ by ψ'. Let B abbreviate the conjunction of equivalences (biimplications) $\bigwedge_{1 \leq k \leq n}(x_{p_k} \leftrightarrow \varphi_k)$.

1. For $|N| = 1$. If $\chi \in \mathcal{L}_{\mathsf{DEL}}$, then χ is DEL-satisfiable if and only if

$$\chi' \wedge \bigwedge_{\ell \leq \mathrm{md}(\varphi)} \mathrm{K}_i^\ell B$$

   is DEL-satisfiable, where the modal depth $\mathrm{md}(\varphi)$ is the maximal number of nested modal operators of ψ.

2. For $|N| \geq 2$. If $\chi \in \mathcal{L}_{\mathsf{DEL}}$, then χ is DEL-satisfiable if and only if

$$\chi' \wedge \bigwedge_{\ell \leq \mathrm{md}(\varphi)} \mathrm{E}_N^\ell B$$

   is DEL-satisfiable.

3. If $\chi \in \mathcal{L}_{\mathsf{DELC}}$, then χ is DELC-satisfiable if and only if

$$\chi' \wedge \mathrm{C}_N B$$

   is DELC-satisfiable.

PROOF. To simplify suppose that the subformula of χ is $[p := \varphi]\psi$. We do the last case here. The other ones are analogous, and left to the reader.

From the left to the right. Suppose that $M = \langle S, R, V\rangle$ is an epistemic model such that $M, s \models \chi$. Then we construct an epistemic model $M_{x_p} = \langle S, R, V_{x_p}\rangle$, where

$$\begin{aligned} V_{x_p}(p) &= V_p \quad \text{for all } p \neq x_p \text{ and} \\ V_{x_p}(x_p) &= [\![\varphi]\!]_M \end{aligned}$$

First, note that $M_{x_p}, s \models \chi$ (because x_p does not appear in χ).
Second, note that $M_{x_p} \models x_p \leftrightarrow \varphi$ (because $[\![x_p]\!]_{M_{x_p}} = [\![\varphi]\!]_{M_{x_p}}$).
Therefore $M_{x_p}, s \models \mathrm{C}_N(x_p \leftrightarrow \varphi)$, i.e., $M_{x_p}, s \models \mathrm{C}_N B$.
Third, note that for every $t \in S$ we have that:
$M_{x_p}, t \models [p := \varphi]\psi$
iff $M_{x_p}^{p:=\varphi}, t \models \psi$
iff $M_{x_p}^{p:=\varphi}, t \models \psi'$ (because $V_{x_p}^{p:=\varphi}(p) = V_{x_p}^{p:=\varphi}(x_p)$).
Therefore $M_{x_p} \models [p := \varphi]\psi \leftrightarrow \psi'$
Therefore $M_{x_p}, s \models \chi' \wedge \mathrm{C}_N B$.

From the right to the left.
Suppose w.l.o.g. that M is generated from s.
Suppose that $M, s \models \chi' \wedge \mathrm{C}_N(x_p \leftrightarrow \varphi)$.
Then $M \models x_p \leftrightarrow \varphi$, i.e., $V(x_p) = [\![\varphi]\!]_M$.
Then, for all $t \in S$:
$M, t \models \psi'$ iff $M^{p:=\varphi}, t \models \psi$ (because $V(x_p) = [\![\varphi]\!]_M = V^{p:=\varphi}(p)$).
In other words, $M \models \psi' \leftrightarrow [p := \varphi]\psi$.
Therefore $M, s \models \chi$. ∎

Intuitively, the conjuncts $\mathrm{K}_i^{\ell}(x_{p_k} \leftrightarrow \varphi_k)$ set the value of the new letter x_{p_k} to that of φ_k. To guarantee that the equivalences hold everywhere in the model we need to use a master modality. In the case of DEL we use the 'everybody knows' operator, that has to be iterated up to the modal depth of the formula.

Renaming avoids exponential blow-up. This allows the definition of reduction operators $\mathrm{reg}_{\mathsf{DEL}}$, $\mathrm{reg}_{\mathsf{DELC}}$ that iteratively eliminate all assignments.

For example, consider the formula $\neg[\neg \mathit{light}][\mathit{light} := \neg \mathit{light}]\mathrm{K}_i \mathit{light}$. Its reduction is $\neg[\neg \mathit{light}]\mathrm{K}_i x_{\mathit{light}} \wedge \mathrm{K}_i(x_{\mathit{light}} \leftrightarrow \neg \mathit{light})$.

THEOREM 51
$\mathrm{reg}_{\mathsf{DEL}}$ and $\mathrm{reg}_{\mathsf{DELC}}$ are polynomial transformations, and preserve satisfiability in the respective logics.

PROOF. Satisfiability-equivalence follows from Theorem 50.

For the common-knowledge case we prove that the size of the reduction of χ is at most $\mathrm{len}(\chi) \times (\mathrm{len}(\chi) + 6)$, and for the case of DEL we prove that the size of the reduction of χ is at most $\mathrm{len}(\chi)^2 \times (\mathrm{len}(\chi) + 6)$. Indeed, in Theorem 50 the size of χ' is at most $\mathrm{len}(\chi)$, the size of each equivalence in B is at most $\mathrm{len}(\chi) + 4$, and the number of these equivalences is bound by the number of (atomic) assignments in χ, which is at most $\mathrm{len}(\chi)$. In the case of operators 'K' and 'E' the number of equivalences has to be multiplied by the modal depth of χ, which is at most $\mathrm{len}(\chi)$.

1. For $|N| = 1$. If $\chi \in \mathcal{L}_{\mathsf{DEL}}$, then χ is DEL-satisfiable if and only if $\mathrm{reg}_{\mathsf{DEL}}(\chi)$ is PAL-satisfiable;

2. For $|N| \geq 2$. If $\chi \in \mathcal{L}_{\mathsf{DEL}}$, then χ is DEL-satisfiable if and only if $\mathrm{reg}_{\mathsf{DEL}}(\chi)$ is PAL-satisfiable;

3. If $\chi \in \mathcal{L}_{\mathsf{DELC}}$, then χ is DELC-satisfiable if and only if $\mathrm{reg}_{\mathsf{DELC}}(\chi)$ is PALC-satisfiable.

∎

4.5.2 Eliminating Announcements

Once assignments are eliminated, we can eliminate announcements by Lutz' procedure. For simplicity we show only the case without common knowledge.

First we compute the set of contextual subformulas which are inductively defined as follows.

$$\begin{aligned}
\mathrm{Sub}(p) &= \{(\epsilon, p)\} \\
\mathrm{Sub}(\neg\varphi) &= \mathrm{Sub}(\varphi) \cup \{(\epsilon, \neg\varphi)\} \\
\mathrm{Sub}(\varphi \wedge \psi) &= \mathrm{Sub}(\varphi) \cup \mathrm{Sub}(\psi) \cup \{(\epsilon, \varphi \wedge \psi)\} \\
\mathrm{Sub}(\mathrm{K}_i\varphi) &= \mathrm{Sub}(\varphi) \cup \{(\epsilon, \mathrm{K}_i\varphi)\} \\
\mathrm{Sub}([\varphi]\psi) &= \mathrm{Sub}(\varphi) \cup \{(\varphi \cdot \tau, \chi) \mid (\tau, \chi) \in \mathrm{Sub}(\psi)\} \cup \{(\epsilon, [\varphi]\psi)\}
\end{aligned}$$

where τ denotes lists, ϵ is the empty list, and '$\cdot$' is concatenation.

Intuitively, $\mathrm{Sub}(\varphi)$ is the set of *relevant* subformulas of φ together with the sequence of announcements in the scope of which they occur. $(\tau, \psi) \in \mathrm{Sub}(\varphi)$ means that subformula ψ of φ is in the scope of the sequence τ of announcements. Now, let φ be a multi-agent formula whose DEL-satisfiability is to be decided. We introduce a set of fresh propositional letters $P_\varphi = \{x^\tau_\psi \mid (\tau, \psi) \in \mathrm{Sub}(\varphi)\}$. Then the reduction of φ is:

$$\mathrm{reg}_{\mathsf{DEL}}(\varphi) = x^\epsilon_\varphi \wedge \bigwedge_{\ell \leq \mathrm{md}(\varphi)} \bigwedge_{(\tau,\psi)\in\mathrm{Sub}(\varphi)} \mathrm{E}^\ell_N(B^\tau_\psi)$$

where $\mathrm{md}(\varphi)$ is the modal depth of φ, $\mathrm{E}^\ell_N\varphi$ abbreviates $\mathrm{E}_N \ldots \mathrm{E}_N\varphi$ (ℓ times), and the bi-implications B^τ_ψ are inductively defined as follows:

$$\begin{array}{lcl}
B^\tau_p & = & x^\tau_p \leftrightarrow p \\
B^\tau_{\neg\varphi} & = & x^\tau_{\neg\varphi} \leftrightarrow \neg x^\tau_\varphi \\
B^\tau_{\varphi\wedge\psi} & = & x^\tau_{\varphi\wedge\psi} \leftrightarrow (x^\tau_\varphi \wedge x^\tau_\psi) \\
B^\tau_{\mathrm{K}_i\varphi} & = & x^\tau_{\mathrm{K}_i\varphi} \leftrightarrow \mathrm{K}_i(\bigwedge_{\mu\in\mathrm{pre}(\tau)} x^\mu_{\mu/\tau} \rightarrow x^\tau_\varphi) \\
B^\tau_{[\varphi]\psi} & = & x^\tau_{[\varphi]\psi} \leftrightarrow (x^\tau_\varphi \rightarrow x^{\tau\cdot\varphi}_\psi)
\end{array}$$

and where $\mathrm{pre}(\tau)$ is the set of true prefixes of τ and μ/τ is the leftmost symbol of τ that is not in μ. When the sequence τ is empty, then the conjunction collapses to true. B^τ_ψ guarantees that x^τ_ψ is true exactly where ψ is true.

When applied to a formula in $\mathcal{L}_{\mathsf{DEL}}$ without assignments, $\mathrm{reg}_{\mathsf{DEL}}$ returns a formula in $\mathcal{L}_{\mathsf{EL}}$. For example, consider the formula $\neg[p]\mathrm{K}_i p$. The set of relevant bi-implications is $B = \{x^\epsilon_{\neg[p]\mathrm{K}_i p} \leftrightarrow \neg x^\epsilon_{[p]\mathrm{K}_i p}, x^\epsilon_{[p]\mathrm{K}_i p} \leftrightarrow (x^\epsilon_p \rightarrow x^{\epsilon\cdot p}_{\mathrm{K}_i p}), x^{\epsilon\cdot p}_{\mathrm{K}_i p} \leftrightarrow \mathrm{K}_i(x^\epsilon_p \rightarrow x^{\epsilon\cdot p}_p), x^{\epsilon\cdot p} \leftrightarrow p, x^\epsilon \leftrightarrow p\}$. Then $\mathrm{reg}_{\mathsf{DEL}}(\neg[p]\mathrm{K}_i p) = x^\epsilon_{\neg[p]\mathrm{K}_i p} \wedge \mathrm{K}_i \bigwedge B$, which successively implies x^ϵ_p, $\neg x^{\epsilon\cdot p}_{\mathrm{K}_i p}$, and $\neg\mathrm{K}_i(x^\epsilon_p \rightarrow x^{\epsilon\cdot p}_p)$. The latter is inconsistent with $\mathrm{K}_i(x^{\epsilon\cdot p} \leftrightarrow p)$ and $\mathrm{K}_i(x^\epsilon \leftrightarrow p)$ which are the last two bi-implications prefixed by K_i.

THEOREM 52 (Lutz (2006))
PAL-satisfiability has the same computational complexity as EL-satisfiability.

4.5.3 Eliminating Both

Via Theorem 49 one obtains:

COROLLARY 53
D-satisfiability has the same compuational complexity as EL-satisfiability.

In particular, when each R_i is an equivalence relation, D-satisfiability is:

1. NP-complete if $|N| = 1$;
2. PSPACE-complete if $|N| \geq 2$; and
3. EXPTIME-complete if common knowledge is involved.

4.6 Discussion and Conclusion

We have modelled the frame problem in dynamic epistemic logic by providing correspondents for situation calculus style ontic and observation actions, and we have given complexity results using that translation. As far as we know, this is the first optimal decision procedure for a Reiter-style solution to the frame problem.

Our results apply to the plan verification problem:[3] let an action description D, an $\mathcal{L}_{\mathsf{ELC}}$-formula φ describing the initial situation, an $\mathcal{L}_{\mathsf{ELC}}$-formula ψ describing the goal, and a sequence of actions (or plan) made up of the actions $a_1, \ldots, a_n \in A$ be given, we have to decide whether:

$$\models_D \varphi \rightarrow \langle a_1 \rangle \ldots \langle a_n \rangle \psi$$

Upper bounds follow from Corollary 53. Lower bounds are obtained because satisfiability of χ can be checked by putting $D = \emptyset$, $\varphi = \neg\chi$, $n = 0$ and $\psi = \bot$. Therefore, the plan verification problem inherits the complexity of the underlying logic.

Scherl & Levesque's epistemic extension of Reiter's solution allows for sensing actions $!!\varphi$, which test whether some formula φ is true. Such sensing actions can be viewed as abbreviating the nondeterministic composition of two announcements: $!!\varphi = \varphi \cup \neg\varphi$. The expansion of such abbreviations leads to exponential blow-up, which does not allow us to extend our approach and integrate primitive sensing actions: it is not clear how the associated successor state axiom (cf. axiom ESSK in Section 2.3.5)

$$[!!\varphi]\mathrm{K}_i\psi \leftrightarrow ((\varphi \rightarrow \mathrm{K}_i(\varphi \rightarrow [!!\varphi]\psi)) \wedge (\neg\varphi \rightarrow \mathrm{K}_i(\neg\varphi \rightarrow [!!\varphi]\psi)))$$

could be transformed into a polynomial transformation. Further evidence that the presence of sensing actions increases complexity is provided by the result in (Herzig et al., 2000a) that plan verification in this case is Π_2^p-complete. We leave integration of sensing actions as future work.

[3] Called "projection problem" in (Scherl and Levesque, 2003, p.22).

Chapter 5

Reasoning with Analytic Tableaux

5.1 Introduction

Traditionally, proof systems for dynamic epistemic logics are obtained by means of *reduction axioms*. In the particular case of PAL, they permit the translation of each formula in $\mathcal{L}_{\mathsf{PAL}}$ into an equivalent formula in $\mathcal{L}_{\mathsf{EL}}$. The well-known proof system for PAL is therefore obtained by just extending that of EL by the former's reduction axioms. It follows that both logics have the same expressivity. Nevertheless the translated formula can be exponentially larger than the original one. That is, PAL is strictly more succinct. This is the reason why PAL is considered to be more convenient for reasoning about knowledge (van Benthem et al., 2006). Curiously however, satisfiability checking in PAL is PSPACE-complete (Lutz, 2006), as well as in EL.

In this chapter, we present a tableau-calculus for PAL. The method decides satisfiability without reducing PAL-formulas to another language.

5.2 A Tableau Method for Public Announcement Logic

We present in this section a proof method for public announcement logic that uses tableaux. Exactly in the same way as all other tableau methods, given a formula φ, it systematically tries to construct a model for it. When it fails, φ is inconsistent and thus its negation is valid.

We present a modular method that can be used whenever the underling epistemic logic is any one of K, KT, S4 and S5. If no restriction is imposed over the accessibility relations in R, then we call the resultant logic K-PAL. If each R_i is reflexive, then the resultant logic is called KT-PAL. If each R_i is reflexive and transitive, then we call the resultant logic S4-PAL. And finally, if each R_i is reflexive, transitive and symmetric, then we call the resultant logic S5-PAL.

In our representation formulas are prefixed by a number that represents possible worlds in the model, similar to (Fitting, 1983, Chapter 8). Formulas are also prefixed by finite sequences of announcements corresponding to successive model restrictions, as in (Lutz, 2006). Given a finite sequence of formulas $\psi^k = (\psi_1 \dots \psi_k)$, for each $1 \leq i \leq k$, the sequence $(\psi_1 \dots \psi_i)$ is noted ψ^i whereas $\psi^0 = \epsilon$ denotes the empty sequence. In addition, we write $M|\psi$ for M^ψ, and $M|\psi^k$ for $M|\psi_1|\dots|\psi_k$.

DEFINITION 54 (LABELLED FORMULA)
A *labelled formula* is a triple $\lambda = (\psi^k, x, \varphi)$ where

- ψ^k is a finite sequence $(\psi_1 \dots \psi_k)$ of formulas in $\mathcal{L}_{\mathsf{PAL}}$;
- $x \in \mathbb{N}$; and
- $\varphi \in \mathcal{L}_{\mathsf{PAL}}$.

The pair ψ^k, x is the *label* of the formula φ. It represents the possible world named x in the epistemic model named ψ^k.

DEFINITION 55 (SKELETON)
A *skeleton* is a ternary relation $\Sigma \subseteq (N \times \mathbb{N} \times \mathbb{N})$ that represents the accessibility relations. A *branch* is a pair $b = (\Lambda, \Sigma)$ where Λ is a set of labelled formulas and Σ is a skeleton.

DEFINITION 56 (TABLEAU)
A *tableau* is a set $T^i = \{b_1^i, b_2^i, \dots\}$ of branches. A tableau T^{i+1} is *obtained from* a tableau T^i if and only if $T^{i+1} = (T^i \setminus \{b_j^i\}) \cup B$ for some $b_j^i = (\Lambda, \Sigma) \in T^i$ and some finite set B of branches generated from b_j^i by the application of one of the *tableau rules* defined below:

R$\neg$: if $(\psi^k, x, \neg\neg\varphi) \in \Lambda$, then $B = \{(\Lambda \cup \{(\psi^k, x, \varphi)\}, \Sigma)\}$.

R$\wedge$: if $(\psi^k, x, \varphi_1 \wedge \varphi_2) \in \Lambda$, then $B = \{(\Lambda \cup \{(\psi^k, x, \varphi_1), (\psi^k, x, \varphi_2)\}, \Sigma)\}$.

R$\vee$: if $(\psi^k, x, \neg(\varphi_1 \wedge \varphi_2)) \in \Lambda$, then $B = \{(\Lambda \cup \{(\psi^k, x, \neg\varphi_1)\}, \Sigma), (\Lambda \cup \{(\psi^k, x, \neg\varphi_2)\}, \Sigma)\}$.

RK: if $(\psi^k, x, \mathrm{K}_i\varphi) \in \Lambda$ and $(i, x, x') \in \Sigma$, then $B = \{(\Lambda_0, \Sigma), \dots (\Lambda_k, \Sigma)\}$, where

$$
\begin{array}{lll}
\Lambda_0 & = \Lambda \cup & \{(\psi^0, x', \neg\psi_1)\} \\
\Lambda_1 & = \Lambda \cup & \{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\} \\
\Lambda_2 & = \Lambda \cup & \{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3)\} \\
 & \vdots & \\
\Lambda_{k-1} & = \Lambda \cup & \{(\psi^0, x', \psi_1), \dots, (\psi^{k-2}, x', \psi_{k-1}), (\psi^{k-1}, x', \neg\psi_k)\} \\
\Lambda_k & = \Lambda \cup & \{(\psi^0, x', \psi_1), \dots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \varphi)\}.
\end{array}
$$

RT: if $(\psi^k, x, \mathrm{K}_i\varphi) \in \Lambda$, then $B = \{(\Lambda \cup \{(\psi^k, x, \varphi)\}, \Sigma)\}$.

R4: if $(\psi^k, x, \mathrm{K}_i\varphi) \in \Lambda$ and $(i, x, x') \in \Sigma$, then $B = \{(\Lambda_1, \Sigma), \ldots (\Lambda_{k+1}, \Sigma)\}$, where

$$\begin{array}{lll}
\Lambda_0 & = \Lambda \cup & \{(\psi^0, x', \neg\psi_1)\} \\
\Lambda_1 & = \Lambda \cup & \{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\} \\
\Lambda_2 & = \Lambda \cup & \{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3)\} \\
 & \vdots & \\
\Lambda_{k-1} & = \Lambda \cup & \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-2}, x', \psi_{k-1}), (\psi^{k-1}, x', \neg\psi_k)\} \\
\Lambda_k & = \Lambda \cup & \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \mathrm{K}_i\varphi)\}.
\end{array}$$

R5$_\uparrow$: if $(\psi^k, x, \mathrm{K}_i\varphi) \in \Lambda$ and $(i, x', x) \in \Sigma$, then $B = \{(\Lambda_1, \Sigma), \ldots (\Lambda_{k+1}, \Sigma)\}$, where

$$\begin{array}{lll}
\Lambda_0 & = \Lambda \cup & \{(\psi^0, x', \neg\psi_1)\} \\
\Lambda_1 & = \Lambda \cup & \{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\} \\
\Lambda_2 & = \Lambda \cup & \{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3)\} \\
 & \vdots & \\
\Lambda_{k-1} & = \Lambda \cup & \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-2}, x', \psi_{k-1}), (\psi^{k-1}, x', \neg\psi_k)\} \\
\Lambda_k & = \Lambda \cup & \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \mathrm{K}_i\varphi)\}.
\end{array}$$

R$\widehat{\mathrm{K}}$: if $(\psi^k, x, \neg\mathrm{K}_i\varphi) \in \Lambda$, then $B = \{(\Lambda \cup \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \neg\varphi)\}, \Sigma \cup \{(i, x, x')\})\}$ for some x' that does not appear in Λ.

R$[\cdot]$: if $(\psi^k, x, [\varphi_1]\varphi_2) \in \Lambda$, then $B = \{(\Lambda \cup \{(\psi^k, x, \neg\varphi_1)\}, \Sigma), (\Lambda \cup \{(\psi^k, x, \varphi_1), (\psi^k\varphi_1, x, \varphi_2)\}, \Sigma)\}$.

R$\langle\cdot\rangle$: if $(\psi^k, x, \neg[\varphi_1]\varphi_2) \in \Lambda$, then $B = \{(\Lambda \cup \{(\psi^k, x, \varphi_1), (\psi^k\varphi_1, x, \neg\varphi_2)\}, \Sigma)\}$.

Given a formula $\varphi \in \mathcal{L}_{\mathsf{PAL}}$, the tableau $T^0 \stackrel{\text{def}}{=} \{b_1^0\} \stackrel{\text{def}}{=} \{(\{(\epsilon, 0, \varphi)\}, \emptyset)\}$ is the *initial tableau* for φ. A *tableau for* φ is a tableau that can be obtained from the initial tableau for φ by successive applications of tableau rules.

The Rules R$\neg$, R$\wedge$ and R$\vee$ are standard. The intuition behind rules R$[\cdot]$ and R$\langle\cdot\rangle$ reflects the semantics of public announcements. The model (M, s) satisfies $[\psi]\varphi$ if and only if (M, s) satisfies $\neg\psi$ or it satisfies ψ and the restricted model satisfies φ. Rule R$\langle\cdot\rangle$ is the dual of rule R$[\cdot]$. The rules for the knowledge operators are quite different from their correspondent in EL. When a new world x' is created in $M|\psi_0|\psi_1|\ldots|\psi_k$ by rule R$\widehat{\mathrm{K}}$, we must be sure that this world can consistently belong to M and that it is not deleted by one of the announcements of the sequence. In the case of rule RK, the world x' was already created, but possibly in a model restricted by a different sequence of announcements. We therefore must be sure that x' would also be present in a model generated by the sequence of announcements we have in hand. This is also the case for rules R4 (transitivity) and R5$_\uparrow$ (symmetry), but not for rule RT (reflexivity), because in the latter we do not visit a different world.

The tableau method for K-PAL consists on rules R$\neg$, R$\wedge$, R$\vee$, RK, R$\widehat{\mathrm{K}}$, R$[\cdot]$ and R$\langle\cdot\rangle$. For KT-PAL, we also have rule RT. For S4-PAL, we have all rules for KT-PAL plus rule R4. And for S5-PAL, we have all rules for S4-PAL plus rule R5$_\uparrow$.

1.	$\epsilon, 0, \neg[p \wedge \neg \mathrm{K}_i p] \neg(p \wedge \neg \mathrm{K}_i p)$		
2.	$\epsilon, 0, p \wedge \neg \mathrm{K}_i p$		$(\mathrm{R}\langle\cdot\rangle : 1)$
3.	$p \wedge \neg \mathrm{K}_i p, 0, \neg\neg(p \wedge \neg \mathrm{K}_i p)$		$(\mathrm{R}\langle\cdot\rangle : 1)$
4.	$p \wedge \neg \mathrm{K}_i p, 0, p \wedge \neg \mathrm{K}_i p$		$(\mathrm{R}\neg : 3)$
5.	$p \wedge \neg \mathrm{K}_i p, 0, p$		$(\mathrm{R}\wedge : 4)$
6.	$p \wedge \neg \mathrm{K}_i p, 0, \neg \mathrm{K}_i p$		$(\mathrm{R}\wedge : 4)$
7.	$\epsilon, 1, p \wedge \neg \mathrm{K}_i p$	$(i, 0, 1) \in \Sigma$	$(\mathrm{R}\widehat{\mathrm{K}} : 6)$
8.	$p \wedge \neg \mathrm{K}_i p, 1, \neg p$		$(\mathrm{R}\widehat{\mathrm{K}} : 6)$
9.	$\epsilon, 1, p$		$(\mathrm{R}\wedge : 7)$
10.	$\epsilon, 1, \neg \mathrm{K}_i p$		$(\mathrm{R}\wedge : 7)$
	closed		$(8, 9)$

Figure 5.1: Closed tableau for the formula $[p \wedge \neg \mathrm{K}_i p] \neg(p \wedge \neg \mathrm{K}_i p)$.

DEFINITION 57
Let $b = (\Lambda, \Sigma)$ be a branch. The set of labelled formulas Λ is *blatantly inconsistent* if and only if $\{(\psi^k, x, \varphi), (\psi^k, x, \neg\varphi)\} \subseteq \Lambda$ or $\{(\psi^k, x, p), (\chi^\ell, x, \neg p)\} \subseteq \Lambda$. The branch b is *closed* if and only if Λ is blatantly inconsistent. The branch b is *open* if and only if it is not closed. A tableau is *closed* if and only if all its branches are closed. A tableau is *open* if and only if it has at least one open branch.

Note that (ψ^k, x, p) and $(\chi^\ell, x, \neg p)$ are inconsistent because boolean formulas are preserved through announcements.

EXAMPLE 58
Consider the formula $[p \wedge \neg \mathrm{K}_i p] \neg(p \wedge \neg \mathrm{K}_i p)$. In Figure 5.1 the tableau method is used to show its validity in K-PAL. Note that in this formula the announcement corresponds to the so-called Moore sentence (van Ditmarsch and Kooi, 2006a): "p is true and agent a does not know it". When it is true and publicly announced, all the agents, in particular agent i, become aware of it. Then the sentence becomes false just after being announced.

THEOREM 59 (SOUNDNESS AND COMPLETENESS)
For $\mathsf{C} \in \{\mathsf{K}, \mathsf{KT}, \mathsf{S4}, \mathsf{S5}\}$, there is a closed C-PAL-tableau for $\neg\varphi$ if and only if φ is C-PAL-valid.

PROOF. The proof is in Appendix A.5. ∎

5.3 Tableau Strategies

In the way the method is defined, redundant applications of tableau rules are allowed. In particular, they can be applied indefinitely often. Therefore it may never

stop. In this section we define strategies for the application of the tableau rules. They are inspired by the "tableau construction" defined in (Halpern and Moses, 1992).

When a set of labelled formulas Λ is not saturated under one or more tableau rules, we say that $\lambda \in \Lambda$ is a *witness* to this fact if the given rule, or rules, were still not applied to λ. For convenience, we further use notation $\Lambda(x)$ for the set of *labelled formulas of* x, defined by $\{(\psi^k, \varphi) \mid (\psi^k, x, \varphi) \in \Lambda\}$. And we also use notation $\Lambda(x, i)$ for the set of *labelled formulas of agent* i *in* x, defined by $\{(\psi^k, \mathrm{K}_i\varphi) \mid (\psi^k, x, \mathrm{K}_i\varphi) \in \Lambda\} \cup \{(\psi^k, \neg\mathrm{K}_i\varphi) \mid (\psi^k, x, \neg\mathrm{K}_i\varphi) \in \Lambda\}$.

The strategy defined below is for S5-PAL. It constructs a tree of nodes s whose labels are tableau branches $L(s) = (\Lambda_s, \Sigma_s)$ generated by the application of tableau rules to their antecedents.

STRATEGY 60
Let $\varphi_0 \in \mathcal{L}_{\mathsf{PAL}}$ be given. Construct a tree as follows.

1. Start with a single node s_0 (the root of the tree) whose label is the initial branch for φ_0, i.e., the pair $L(s_0) = (\Lambda_{s_0}, \Sigma_{s_0})$, where $\Lambda_{s_0} = \{(\epsilon, 0, \varphi_0)\}$ and $\Sigma_{s_0} = \emptyset$.

2. Repeat until neither step 2(a) nor step 2(b) below applies.

   (a) *World saturation:* if s is a leaf with label $L(s)$ such that $L(s)$ is open and not saturated under rules $\mathrm{R}\neg$, $\mathrm{R}\wedge$, RT, $\mathrm{R}\langle\cdot\rangle$, $\vee$, $\mathrm{R}[\cdot]$, RK, R4 and $\mathrm{R}5_\uparrow$, and $\lambda \in \Lambda_s$ is a witness to this fact, then do:

      i. if $\lambda = (\psi^k, x, \neg\neg\varphi)$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, x, \varphi)\}$ and $\Sigma_{s'} = \Sigma_s$. And then go to step 2.

      ii. if $\lambda = (\psi^k, x, \varphi_1 \wedge \varphi_2)$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, x, \varphi_1), (\psi^k, x, \varphi_2)\}$ and $\Sigma_{s'} = \Sigma_s$. And then go to step 2.

      iii. if $\lambda = (\psi^k, x, \mathrm{K}_i\varphi)$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, x, \varphi)\}$ and $\Sigma_{s'} = \Sigma_s$. And then go to step 2.

      iv. if $\lambda = (\psi^k, x, \neg[\varphi_1]\varphi_2)$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, x, \varphi_1), (\psi^k\varphi_1, x, \varphi_2)\}$ and $\Sigma_{s'} = \Sigma_s$. And then go to step 2.

      v. if $\lambda = (\psi^k, x, \neg(\varphi_1 \wedge \varphi_2))$ then create two successors s_1 and s_2 such that $\Lambda_{s_1} = \Lambda_s \cup \{(\psi^k, x, \neg\varphi_1)\}$ and $\Sigma_{s_1} = \Sigma_s$, and $\Lambda_{s_2} = \Lambda_s \cup \{(\psi^k, x, \neg\varphi_2)\}$ and $\Sigma_{s_2} = \Sigma_s$. And then go to step 2.

      vi. if $\lambda = (\psi^k, x, [\varphi_1]\varphi_2)$ then create two successors s_1 and s_2 such that $\Lambda_{s_1} = \Lambda_s \cup \{(\psi^k, x, \neg\varphi_1)\}$ and $\Sigma_{s_1} = \Sigma_s$, and $\Lambda_{s_2} = \Lambda_s \cup \{(\psi^k, x, \varphi_1), (\psi^k\varphi_1, x, \varphi_2)\}$ and $\Sigma_{s_2} = \Sigma_s$. And then go to step 2.

      vii. if $\lambda = (\psi^k, x, \mathrm{K}_i\varphi)$ and $(i, x, x') \in \Sigma$, then for each $i \in \{0, 1, \ldots, k-1\}$, create a successor s_i such that $\Lambda_{s_i} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < i\} \cup \{(\psi^i, x', \neg\psi_{i+1})\}$ and $\Sigma_{s_i} = \Sigma_s$, and also create a successor node s_k such that $\Lambda_{s_k} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, x', \varphi)\}$ and $\Sigma_{s_k} = \Sigma_s$. And then go to step 2.

      viii. if $\lambda = (\psi^k, x, \mathrm{K}_i\varphi)$ and $(i, x, x') \in \Sigma$, then for each $i \in \{0, 1, \ldots, k-1\}$, create a successor s_i such that $\Lambda_{s_i} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < i\} \cup \{(\psi^i, x', \neg\psi_{i+1})\}$ and $\Sigma_{s_i} = \Sigma_s$, and also create a successor node

s_k such that $\Lambda_{s_k} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, x', \mathrm{K}_i\varphi)\}$ and $\Sigma_{s_k} = \Sigma_s$. And then go to step 2.

ix. if $\lambda = (\psi^k, x, \mathrm{K}_i\varphi)$ and $(i, x', x) \in \Sigma$, then for each $i \in \{0, 1, \ldots, k-1\}$, create a successor s_i such that $\Lambda_{s_i} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < i\} \cup \{(\psi^i, x', \neg\psi_{i+1})\}$ and $\Sigma_{s_i} = \Sigma_s$, and also create a successor node s_k such that $\Lambda_{s_k} = \Lambda_s \cup \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, x', \mathrm{K}_i\varphi)\}$ and $\Sigma_{s_k} = \Sigma_s$. And then go to step 2.

(b) *Create a new world:* if s is a leaf with label $L(s)$ such that $L(s)$ is open, world-saturated and not saturated under rule $\mathrm{R}\widehat{\mathrm{K}}$ and $\lambda = (\psi^k, x, \neg\mathrm{K}_i\varphi)$ is a witness to this fact, then do steps i, ii and iii below. And then go to step 2.

i. generate a new natural number x' that does not appear in Σ_s and create a label $L' = (\Lambda', \Sigma')$, where $\Lambda' = \{(\psi^j, x', \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, x', \neg\varphi)\}$ and $\Sigma' = \{(i, x, x')\}$.

ii. if there is a sequence of natural numbers $y_0, y_1, \ldots, y_n$ such that $y_n = x$ and for all $0 \leq i < n$, $(i, y_i, y_{i+1}) \in \Sigma_s$ and $\Lambda_s(x, i) = \Lambda_s(y_0, i)$ and $\Lambda' \subseteq \Lambda_s(y_1)$, then create a successor s' such that $\Lambda_{s'} = \Lambda_s$ and $\Sigma_{s'} = \Sigma_s \cup \{(i, x, y_1)\}$.

iii. if step 2(b)ii does not apply, then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \Lambda'$ and $\Sigma_{s'} = \Sigma_s \cup \Sigma'$.

3. If s is a leaf and its label $L(s)$ is open, then return `true`, else return `false`.

Simple modifications of Strategy 60 above give us strategies for the other logics we consider here. A strategy for S4-PAL can be obtained by removing step 2(a)ix. By removing steps 2(a)viii and 2(a)ix, we obtain a strategy for KT-PAL. And by removing steps 2(a)iii, 2(a)viii and 2(a)ix we obtain a strategy for K-PAL.

Note that step 2(b)ii has a *loop test*. This is crucial to guarantee that the process halts for S4-PAL and S5-PAL. Before applying rule $\mathrm{R}\widehat{\mathrm{K}}$, which means that a new "world" x' will be created, it verifies that there is no loop.

We continue by proving termination. After that we prove soundness and completeness for S5-PAL only. Proofs for the other logics are similar and left to the reader. We first need a definition and a lemma.

DEFINITION 61 (LABELLED SUB-FORMULAS)
The *set of labelled sub-formulas of* φ, $\mathrm{Sub}(\varphi)$, and *the set of labelled sub-formulas of* φ *and its negations*, $\mathrm{Sub}^+(\varphi)$, are recursively defined as follows (cf. definition of contextual

sub-formulas on page 54):

$$\begin{aligned}
\mathrm{Sub}(p) &\stackrel{\text{def}}{=} \{(\epsilon, p)\} \\
\mathrm{Sub}(\neg\varphi) &\stackrel{\text{def}}{=} \mathrm{Sub}(\varphi) \cup \{(\epsilon, \neg\varphi)\} \\
\mathrm{Sub}(\varphi \wedge \psi) &\stackrel{\text{def}}{=} \mathrm{Sub}(\varphi) \cup \mathrm{Sub}(\psi) \cup \{(\epsilon, \varphi \wedge \psi)\} \\
\mathrm{Sub}(\mathrm{K}_i\varphi) &\stackrel{\text{def}}{=} \mathrm{Sub}(\varphi) \cup \{(\epsilon, \mathrm{K}_i\varphi)\} \\
\mathrm{Sub}([\psi]\varphi) &\stackrel{\text{def}}{=} \mathrm{Sub}(\psi) \cup \{(\psi\chi^k, \varphi') \mid (\chi^k, \varphi') \in \mathrm{Sub}(\varphi)\} \cup \{(\epsilon, [\psi]\varphi)\} \\
\\
\mathrm{Sub}^+(\varphi) &\stackrel{\text{def}}{=} \mathrm{Sub}(\varphi) \cup \{(\psi^k, \neg\varphi') \mid (\psi^k, \varphi') \in \mathrm{Sub}(\varphi)\}
\end{aligned}$$

LEMMA 62

1. $|\mathrm{Sub}(\varphi_0)| \leq \mathrm{len}(\varphi_0)$.
2. For all $(\psi^k, \varphi) \in \mathrm{Sub}(\epsilon, \varphi_0)$, $k \leq \mathrm{len}(\varphi_0)$.
3. $|\mathrm{Sub}^+(\varphi)| \leq 2 \times \mathrm{len}(\varphi)$.

Items 1 and 2 are proved in (Lutz, 2006) and 3 is an obvious consequence of them.

THEOREM 63
For all $\varphi \in \mathcal{L}_{\mathsf{PAL}}$, Strategy 60 creates a finite tree for φ.

PROOF. Let a $\mathcal{L}_{\mathsf{PAL}}$-formula φ be given. Because $\mathrm{Sub}^+(\varphi)$ is finite, each step generates a finite number of immediate successors. Then, by the fact that the initial tree for φ is a single node (and, in particular, it is finite), each step of the strategy generates a finite tree.

We now show that each step is applied finitely often. Let $\mathrm{len}(\varphi) = n$. By Lemma 62, the number of labelled sub-formulas of φ and its negations is bounded by $2n$. Then after $2n$ applications of step 2(a) all the leafs of the tree are world-saturated. This means that there can be at most $2n$ applications of step 2(a) between two subsequent applications of step 2(b).

Now, note that there exists at most 2^{2n} different subsets of $\mathrm{Sub}^+(\varphi)$. This means that the loop tests can fail at most 2^{2n} times. It immediately follows that step 2(b) can be applied at most 2^{2n} times. Therefore, Strategy 60 always creates a finite tree and thus always halts. ∎

THEOREM 64
For all $\varphi_0 \in \mathcal{L}_{\mathsf{PAL}}$, φ_0 is S5-PAL-satisfiable if and only if Strategy 60 for φ_0 returns `true`.

PROOF. From the left to the right. We show that if φ is S5-PAL-satisfiable, then the tree for φ generated by Strategy 60 will have at least one leaf whose label is an

open tableau branch. We do this by showing that all steps preserve satisfiability. This proof is along the lines of the first part of the proof of Theorem 59. The only remarkable difference is the step 2(b)ii: suppose that $(\psi^k, x, \neg \mathrm{K}_i\varphi) \in \Lambda_s$ and that the loop test succeeds. This means that there is a sequence y_0,y_1,...,y_n such that $y_n = x$ and for all $0 \leq i < n$, $(i, y_i, y_{i+1}) \in \Sigma_s$, and s has a successor s' such that $\Lambda_{s'} = \Lambda_s$ and $\Sigma_{s'} = \Sigma_s \cup \{(i, x, y_1)\}$. We then consider the (*unfolded*) tableau branch $L' = (\Lambda', \Sigma')$ such that $\Lambda' = \Lambda_{s'} \cup \{(\chi^\ell, x', \varphi') \mid (\chi^\ell, y_1, \varphi') \in \Lambda_{s'}\}$ and $\Sigma' = (\Sigma_s \setminus \{(i, x, y_1)\}) \cup \{(i, x, x'), (i, x', y_2)\}$. Clearly, L' is satisfiable if and only if $L(s')$ is satisfiable. By hypothesis, there is an epistemic structure $M = \langle S, R, V\rangle$ and a function $f : \mathbb{N} \to S$ that satisfy $L(s)$. Then there exists $s \in S^{\psi^k}$ such that $f(x) R_i^{\psi^k} s$. We thus consider the function $f' : \mathbb{N} \to S$ such that for all integer x that occur in Λ', $f'(x) \stackrel{\text{def}}{=} f(x)$ and $f'(x') \stackrel{\text{def}}{=} s$. Therefore $L(s')$ is satisfiable.

From the right to the left. If Strategy 60 for φ returns `true`, then the tree for φ has a leaf s such that $L(s)$ is open and saturated. Then we use this node to construct a model $M = \langle S, R, V\rangle$ that satisfies φ as follows. S contains all x that appear in Σ_s; R is the reflexive, transitive and symmetric closure of all triples $(i, x, x') \in \Sigma_s$; and each V_p contains all x such that $(\psi^k, x, p) \in \Lambda_s$ for some ψ^k. We then proceed by induction on the length of labelled formulas where the induction hypothesis is: if $L(s)$ is an open saturated branch that contains (ψ^k, x, φ') and $\mathrm{len}(\psi^k, x, \varphi') < n$, then $M|\psi^0, x \models \psi_1, \ldots, M|\psi^{k-1}, x \models \psi_k$, and $M|\psi^k, x \models \varphi'$. This is done along the lines of the second part of the proof of Theorem 59. The details are left to the reader. ∎

The depth of the tree created in Strategy 60 is exponential on the size of the input formula. However, S5-PAL is proven to be in PSPACE (Lutz, 2006), which means that this algorithm is not optimal. Below, we present optimal strategies for logics K-PAL and KT-PAL.

Similarly to the "tableau construction" defined in (Halpern and Moses, 1992), instead of labelling the nodes of the tree with entire tableau branches, in our next strategy, node labels contain formulas of only one world x. Hence, we now use pairs of the form $\lambda = (\psi^k, \varphi)$ that, for convenience, are called *labelled formulas* as well (note that x is no longer necessary). But our algorithm differs from that of (Halpern and Moses, 1992) in a crucial point: suppose that $L(s)$ contains the formula $(\psi^k, \mathrm{K}_i\varphi)$. When an i-successor node s' of s is created, one cannot immediately add the labelled formula (ψ^k, φ) to $L(s')$. The reason is that the world s' can have been deleted by some announcement in the sequence ψ^k (cf. tableau rules RK, R4 and R5$_\uparrow$). Then, in step 2(b), before adding this labelled formula to $L(s')$, the algorithm "verifies" that each ψ_i is true in s'. This is implemented with an auxiliary set $\Gamma_{s'}$.

STRATEGY 65
Let $\varphi_0 \in \mathcal{L}_{\mathsf{PAL}}$ be given. Construct a tree as follows.

1. Start with a single node s_0 (the root of the tree) whose label is the pair $L(s_0) = (\Lambda_{s_0}, \Gamma_{s_0})$, where $\Lambda_{s_0} = \{(\epsilon, \varphi_0)\}$ and $\Gamma_{s_0} = \emptyset$.

2. Repeat until neither 2(a) nor 2(b) below applies:

(a) *Local saturation:* if s is a leaf with label $L(s)$ such that $L(s)$ is open and not saturated under rules $\mathrm{R}\neg$, $\mathrm{R}\wedge$, $\mathrm{R}\vee$, RK and RT, and $\lambda \in \Lambda_s$ is a witness to this fact, then do:

i. if $\lambda = (\psi^k, \neg\neg\varphi) \in \Lambda_s$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, \varphi)\}$ and $\Gamma_{s'} = \Gamma_s$. And then go to step 2.

ii. if $\lambda = (\psi^k, \varphi_1 \wedge \varphi_2) \in \Lambda_s$ then create successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, \varphi_1), (\psi^k, \varphi_2)\}$ and $\Gamma_{s'} = \Gamma_s$. And then go to step 2.

iii. if $\lambda = (\psi^k, \mathrm{K}_i\varphi) \in \Lambda_s$ then create successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, \varphi)\}$ and $\Gamma_{s'} = \Gamma_s$. And then go to step 2.

iv. if $\lambda = (\psi^k, \neg[\varphi_1]\varphi_2) \in \Lambda_s$ then create a successor s' such that $\Lambda_{s'} = \Lambda_s \cup \{(\psi^k, \varphi_1), (\psi^k\varphi_1, \varphi_2)\}$ and $\Gamma_{s'} = \Gamma_s$. And then go to step 2.

v. if $\lambda = (\psi^k, \neg(\varphi_1 \wedge \varphi_2)) \in \Lambda_s$ then create two successors s_1 and s_2 such that $\Lambda_{s_1} = \Lambda_s \cup \{(\psi^k, \varphi_1)\}$ and $\Gamma_{s_1} = \Gamma_s$, and $\Lambda_{s_2} = \Lambda_s \cup \{(\psi^k, \varphi_2)\}$ and $\Gamma_{s_2} = \Gamma_s$. And then go to step 2.

vi. if $\lambda = (\psi^k, [\varphi_1]\varphi_2) \in \Lambda_s$ then create two successors s_1 and s_2 such that $\Lambda_{s_1} = \Lambda_s \cup \{(\psi^k, \neg\varphi_1)\}$ and $\Gamma_{s_1} = \Gamma_s$, and $\Lambda_{s_2} = \Lambda_s \cup \{(\psi^k, \varphi_1), (\psi^k\varphi_1, \varphi_2)\}$ and $\Gamma_{s_2} = \Gamma_s$. And then go to step 2.

vii. if $\lambda = (\psi^k, \varphi) \in \Gamma_s$ then for each $i \in \{0, 1, \ldots, k-1\}$, create a successor s_i such that $\Lambda_{s_i} = \Lambda_s \cup \{(\psi^j, \psi_{j+1}) \mid 0 \leq j < i\} \cup \{(\psi^i, \neg\psi_{i+1})\}$ and $\Gamma_{s_i} = \Gamma_s \setminus \{\lambda\}$, and also create a successor s_k such that $\Lambda_{s_k} = \Lambda_s \cup \{(\psi^j, \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, \varphi)\}$ and $\Gamma_{s_k} = \Gamma_s \setminus \{\lambda\}$. And then go to step 2.

(b) *Create new worlds:* if s is a leaf with label $L(s)$ which is local saturated and not saturated under rule $\mathrm{R}\widehat{\mathrm{K}}$, then for each labelled formula of the form $\lambda = (\psi^k, \neg\mathrm{K}_i\varphi) \in \Lambda_s$ that is witness to this, create an i-successor s' such that $\Lambda_{s'} = \{(\psi^j, \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, \neg\varphi)\}$ and $\Gamma_{s'} = \{(\chi^{k'}, \varphi') \mid \{(\chi^{k'}, \mathrm{K}_i\varphi')\} \in \Lambda_s\})$. And then go to step 2.

(c) *Mark nodes:* if the node s with label $L(s)$ is not marked `sat`, then mark it `sat` if either:

- Λ_s is not local saturated and one of its successor is marked `sat`;
- Λ_s is local saturated, it is not blatantly inconsistent and Λ_s does not contain labelled formulas of the form $(\psi^k, \neg\mathrm{K}_i\varphi)$; or
- Λ_s is local saturated and s has successors and all of them are marked `sat`.

3. If the root of the tree is marked `sat`, then return `true`, else return `false`.

The strategy above can be modified for other two logics we consider here. For K-PAL we remove step 2(a)iii, and for S4-PAL, we replace step 2(b) by the following:

2 (b′) Create new worlds: *if s is a leaf with label $L(s)$ which is local saturated and not saturated under rule $\mathrm{R}\widehat{\mathrm{K}}$, then for each labelled formula of the form $\lambda = (\psi^k, \neg\mathrm{K}_i\varphi) \in \Lambda_s$ that is a witness to this, then do steps i, ii and iii below. And then go to step 2.*

i. create a label $L' = (\Lambda', \Gamma')$*, where* $\Lambda' = \{(\psi^j, \psi_{j+1}) \mid 0 \leq j < k\} \cup \{(\psi^k, \neg\varphi)\}$ *and* $\Gamma' = \{(\chi^{k'}, \varphi'), (\chi^{k'}, \mathrm{K}_i\varphi') \mid (\chi^{k'}, \mathrm{K}_i\varphi') \in \Lambda_s\}$*).*

ii. if there is no node s'' *in the path from the root to* s *such that* $L_{s''} = L'$*, then create an* i*-successor node* s' *with label* $L(s') = L'$*.*

iii. if step 2(b)ii does not apply, then create an i*-arrow to the node* s'' *such that* $L(s'') = L'$*.*

Note that we also have a loop test in step 2(b')ii. The idea is the same as in Strategy 60, but here, we also compare the sets Γ. We also remark that it is not possible to use the same idea to define a strategy for S5-PAL. We address this question in Section 5.4. In the sequel, we prove termination, soundness and completeness for S4-PAL only. We leave other cases to the reader.

THEOREM 66
For all $\varphi \in \mathcal{L}_{\mathsf{PAL}}$, Strategy 65 creates a finite tree for φ.

PROOF. The proof is essentially the same as for Theorem 63. ∎

THEOREM 67
For all $\varphi \in \mathcal{L}_{\mathsf{PAL}}$, φ is S4-PAL-satisfiable if and only if Strategy 65 for φ returns `true`.

PROOF. From the left to the right. We show that if φ is S4-PAL-satisfiable, then the tree for φ generated by Strategy 65 has its root marked `sat`. We do this by showing that all steps preserve satisfiability. This proof is along the lines of the first part of proof of Theorem 59. The differences are steps 2(a)vii and 2(b'). Note that step 2(a)vii performs essentially the same task as steps 2(a)vii and 2(a)viii of Strategy 60. So their proof of soundness is very similar. For step 2(b'), note that it is similar to step 2(b) of Strategy 60. So its proof of soundness is also similar. We omit details here.

From the right to the left. If Strategy 65 for φ returns `true`, then the root s_0 is marked `sat`. Then, there is a sub-tree such that all its nodes are marked `sat`. We use this tree to construct a model $M = \langle S, R, V\rangle$ that satisfies φ in the following way. S contains all s in the sub-tree such that s is local saturated; each R_i is the reflexive and transitive closure of pairs (s, s') of nodes in S such that s' is a descendent of an i-successor of s; and each V_p contains all $s \in S$ such that $(\psi^k, p) \in \Lambda_s$ for some ψ^k. The proof continues along the lines of the second part of the proof of Theorem 59. We omit details here. ∎

We continue by showing computational complexity of Strategy 65 for K-PAL and KT-PAL. Remark that no optimal procedure is achieved for S4-PAL. We discuss this in Section 5.4.

THEOREM 68 (COMPLEXITY)
The tableau system for K-PAL and KT-PAL can be implemented in polynomial space.

PROOF. We first show that the height of the trees, generated by Strategy 65, are polynomial in $\mathrm{len}(\varphi)$. The tree construction starts with the root node s_0 whose label is $L(s_0) = (\{\lambda_0\}, \emptyset)$. Suppose that $\mathrm{len}(\lambda_0) = n$. Note that by Lemma 62 step 2(a) can be applied at most $2n$ times before generating a node s such that Λ_s is either closed or saturated. Also note that at this stage Γ_s is empty. Now, suppose that Λ_s is local saturated. Also suppose that s' is a local saturated descendent of an i-successor of s. Note that the K-modal depth of $\Lambda_{s'}$ is less than that of Λ_s. Because the number of K_i operators in φ is at most n, the root of the tree can have at most n i-descendents in the same branch of the tree. From this fact and the observation made before, it follows that the tree has height at most $\mathcal{O}(n^2)$.

We now prove that a depth first exploration of the trees can be made using a polynomial amount of memory. To see this, remember that by Lemma 62 we have that for all nodes s in the tree, $|L(s)| \leq 4n$. Then we can use a vector of $4n$ bits to encode the label of each node in the tree. We do this by setting to 1 the bit that corresponds to the formulas that are present in $L(s)$. Each step of Strategy 65 can produce at most $2n$ different immediate successors. Then, for each node, we can use a vector of $4n^2$ bits to memorise all the choices to be explored after the backtrack. It follows that we need at most $\mathcal{O}(n^5)$ bits of memory to explore the entire tree. ∎

5.4 Related Work and Discussion

We considered versions of PAL where the underlying epistemic logic obeys combination of principles T, 4 and 5. We did not consider the axiom D ($\mathrm{K}_i\varphi \rightarrow \neg\mathrm{K}_i\neg\varphi$) alone, i.e., epistemic logics such as KD and KD45. The reason is that in both systems the axiom T ($\mathrm{K}_i\varphi \rightarrow \varphi$) is derivable for any boolean formula φ. To see this, note that if we have axiom D, then $(\mathrm{K}_i p \wedge \neg p) \rightarrow \langle\neg p\rangle\bot$ is valid.

Recently, another tableau method for S5-PAL was proposed by de Boer (2006). Apart from some aesthetic differences, this method is very similar to ours. However, no proof of decidability is provided.

Strategy 65 is based on the optimal strategy for EL presented in (Halpern and Moses, 1992). Note however that instead of our rule $\mathrm{R5}_\uparrow$, Halpern and Moses use a rule that propagates all formulas prefixed by K_i and $\widehat{\mathrm{K}}_i$ operators to the i-successors. As this rule alone is not complete for S5, they also need to saturate the nodes under sub-formulas (which is called full propositional tableau). But note that such a rule would not be sound in our setting. For example, suppose that in node s with label $L(s)$ we have that $(\psi, \neg\mathrm{K}_i\varphi) \in \Lambda_s$. Because it may be the case that Λ_s also contains $(\epsilon, \neg\psi)$, we cannot add neither (ψ, φ) nor $(\psi, \neg\varphi)$ to Λ_s at the risk of making it blatantly inconsistent. Then, we cannot have our set of formulas saturated under sub-formulas in this way. Strategy 65 is not optimal for S4-PAL. An example is the formula $\neg\mathrm{K}_i p_0 \wedge \mathrm{K}_i[q_1]\mathrm{K}_i p_1 \wedge \mathrm{K}_i[q_2]\mathrm{K}_i p_2 \wedge \cdots \wedge \mathrm{K}_i[q_k]\mathrm{K}_i p_k$, for which Strategy 65 generates a tree containing a branch with 2^k different i-successors.

Chapter 6

Searching for Plans

6.1 Introduction

In the end of Chapter 4 we argue that plan verification can be formalised in dynamic epistemic logic. However, it seems that this logic is not suitable for addressing planning. The reason is that planning demands the construction of the plan. That is, it is more or less like if a part of the formula were lacking, and one should find the right piece for completing it. To try to address this issue, we propose a new logic. In this first attempt though, we restrict our attention to epistemic actions only.

The idea is to have a formalism wherein we can express what becomes true without explicit reference the actions realizing that. For example, when p is true, it becomes known by announcing it. Formally, in public announcement logic, $p \rightarrow [p]\mathrm{K}_i p$ is valid. This is equivalent to $\langle p \rangle \mathrm{K}_i p$ which stands for 'the announcement of p can be made and after that agent i knows p'. It follows that there is an announcement ψ, namely $\psi = p$, that makes the agent know that p, slightly more formally: $\exists \psi . \langle \psi \rangle \mathrm{K}_i p$. We introduce a modal operator that expresses exactly that:

$$\Diamond \mathrm{K}_i p$$

Obviously, the truth of this expression depends on the model: p has to be true. In case p is false, we have $\Diamond \mathrm{K}_i \neg p$ instead. Therefore the formula $\Diamond(\mathrm{K}_i p \vee \mathrm{K}_i \neg p)$ is valid. The corresponding logic is called *arbitrary public announcement logic* (APAL).

In section 6.2 we define the language $\mathcal{L}_{\mathsf{APAL}}$ and its semantics. In section 6.3 we prove some interesting validities and discuss the expressivity of the language, as well as its usefullness in addressing 'Fitch's knowability paradox' (van Benthem, 2004; Brogaard and Salerno, 2004). In section 6.4 we provide a Hilbert-style axiomatisation. In section 6.5 we expand the tableau calculus given in Chapter 5 to arbitrary public announcement logic. Section 6.6 discusses some possible future work and draws conclusions.

6.2 Syntax and Semantics

DEFINITION 69 (LANGUAGE $\mathcal{L}_{\mathsf{APAL}}$)
Let N be a finite set of agents, and P be a countable set of atoms. The language $\mathcal{L}_{\mathsf{APAL}}$ of *arbitrary public announcement logic* is inductively defined by the following BNF:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathrm{K}_i\varphi \mid [\varphi]\varphi \mid \Box\varphi$$

where p ranges over P and i ranges over N.

For $\Box\varphi$, read 'φ holds after any public announcement'. We also use the abbreviation for the operator $\Diamond$, defined as $\Diamond\varphi \stackrel{\text{def}}{=} \neg\Box\neg\varphi$. For $\Diamond\varphi$, read 'φ holds after some public announcement'.

DEFINITION 70 (SEMANTICS OF APAL)
Formulas in $\mathcal{L}_{\mathsf{APAL}}$ are interpreted in S5-models. Let M be an S5-model, the extra clause needed for the operator $\Box$ is as follows (note the restriction of ψ to $\mathcal{L}_{\mathsf{EL}}$):

$$M, s \models \Box\varphi \quad \text{iff} \quad \text{for all } \psi \in \mathcal{L}_{\mathsf{EL}},\ M, s \models [\psi]\varphi$$

For the semantics of the dual operator, we have that $M, s \models \Diamond\varphi$ if and only if there exists $\psi \in \mathcal{L}_{\mathsf{EL}}$ such that $M, s \models \langle\psi\rangle\varphi$.

DEFINITION 71 (VALIDITY AND SATISFIABILITY)
A formula φ in $\mathcal{L}_{\mathsf{APAL}}$ is:

- APAL-*valid* (notation: $\models_{\mathsf{APAL}} \varphi$) if and only if for all pointed S5-models (M, s), for all $\psi \in \mathcal{L}_{\mathsf{EL}}$, $(M, s) \models [\psi]\varphi$; and
- APAL-*satisfiable* if and only if $\not\models_{\mathsf{APAL}} \neg\varphi$.

EXAMPLE 72
A valid formula of the logic is $\Diamond(\mathrm{K}_i p \vee \mathrm{K}_i\neg p)$. To prove this, let (M, s) be arbitrary. Either $M, s \models p$ or $M, s \models \neg p$. In the first case, $M, s \models \Diamond(\mathrm{K}_i p \vee \mathrm{K}_i\neg p)$, because $M, s \models \langle p\rangle(\mathrm{K}_i p \vee \mathrm{K}_i\neg p)$ – the latter is true because $M, s \models p$ and $M^p, s \models \mathrm{K}_i p$; in the second case, we analogously derive $M, s \models \Diamond(\mathrm{K}_i p \vee \mathrm{K}_i\neg p)$ because $M, s \models \langle\neg p\rangle(\mathrm{K}_i p \vee \mathrm{K}_i\neg p)$.

This example also illustrates how to check validity of formulas with diamonds in APAL. The meaning of $\models_{\mathsf{APAL}} \Diamond\varphi$ is:

(*i*) for all (M, s), there exists $\psi \in \mathcal{L}_{\mathsf{EL}}$ such that $M, s \models \langle\psi\rangle\varphi$,

which is different from: (*ii*) there exists $\psi \in \mathcal{L}_{\mathsf{EL}}$ such that for all (M, s), $M, s \models \langle\psi\rangle\varphi$. The latter is **incorrect**. Note that if we were assuming (*ii*), then $\Diamond(\mathrm{K}_i p \vee \mathrm{K}_i\neg p)$ would be not valid.

We choose to restrict the quantification in $\Box$ over formulas of $\mathcal{L}_{\mathsf{EL}}$. Because the intended meaning of the operator $\Box$ is a quantification over announcements, one could also try to define its semantics by one of the following alternatives:

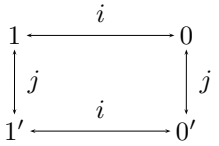
$$M, s \models \Box\varphi \quad \text{iff} \quad \text{for all } \psi \in \mathcal{L}_{\mathsf{APAL}},\ M, s \models [\psi]\varphi \tag{6.1}$$

$$M, s \models \Box\varphi \quad \text{iff} \quad \text{for all } S' \subseteq S \text{ containing } s,\ M|S', s \models \varphi \tag{6.2}$$

$$M, s \models \Box\varphi \quad \text{iff} \quad \text{for all } \psi \in \mathcal{L}_{\mathsf{PAL}},\ M, s \models [\psi]\varphi \tag{6.3}$$

The alternative (6.1) is not well-defined because $\Box\varphi$ can itself be such announcement ψ. Our semantics avoids this problem because the formula $[\psi]\varphi$ is less complex than $\Box\varphi$ in the lexicographic order on $\mathcal{L}_{\mathsf{APAL}}$.

The alternative (6.2) is similar to the proposition of Fine (1970) for quantification over propositional variables in modal logic. This version is not desirable because it permits some "strange" restrictions. For example, consider the model $M = \langle S, R, V\rangle$ where $S = \{1, 1', 0, 0'\}$, $R_i = \{(1,1), (1,0), (0,1), (0,0), (1',1'), (1',0'), (0',1'), (0',0')\}$, $R_j = \{(1,1), (1,1'), (1',1), (1',1'), (0,0), (0,0'), (0',0), (0',0')\}$, and $V_p = \{1, 1'\}$. This model is graphically represented in the figure below:



Now, assume that we are using alternative (6.2). One can restrict the model to $S' = \{1, 1', 0'\}$. However, because 0 and $0'$ are bisimilar, one cannot write a formula that correspond to such a restriction, which means that the restriction to S' should not correspond to a public announcement. One further evidence is the fact that $M, 1 \models \Diamond(\mathrm{K}_i p \wedge \neg \mathrm{K}_j \mathrm{K}_i p)$, because $M|S', 1 \models (\mathrm{K}_i p \wedge \neg \mathrm{K}_j \mathrm{K}_i p)$. In other words, it seems that one can restrict the model in such a way that only one agent perceives the occurrence of the action. And again, this would not be a public announcement.

Considering the alternative (6.3), one argument is that PAL has the same expressive power of EL. Then whatever model restriction that can result from a formula in $\mathcal{L}_{\mathsf{PAL}}$, it can also be obtained by some formula in $\mathcal{L}_{\mathsf{EL}}$.

6.3 Semantic Results

6.3.1 Validities

THEOREM 73
Let $\varphi, \psi \in \mathcal{L}_{\mathsf{APAL}}$. Then:

1. $\models_{\mathsf{APAL}} \Box p \leftrightarrow p$
2. $\models_{\mathsf{APAL}} \Box(\varphi \wedge \psi) \leftrightarrow (\Box\varphi \wedge \Box\psi)$
3. $\models_{\mathsf{APAL}} \Box\varphi \rightarrow \varphi$

4. $\models_{\mathsf{APAL}} \Box\varphi \to \Box\Box\varphi$
5. $\models_{\mathsf{APAL}} \mathrm{K}_i\Box\varphi \to \Box\mathrm{K}_i\varphi$ (but not in the other direction)
6. $\models_{\mathsf{APAL}} \varphi$ implies $\models_{\mathsf{APAL}} \Box\varphi$

PROOF. Let (M, s) be any pointed epistemic model, and let $p \in P$ and $\varphi \in \mathcal{L}_{\mathsf{APAL}}$ be arbitrary.

1. From the left to the right, see case 3 below.
   From the right to the left, suppose $M, s \models p$
   iff $s \in V_p$.
   Suppose that $M, s \models \psi$, for some $\psi \in \mathcal{L}_{\mathsf{EL}}$.
   Then $s \in V_p^\psi$
   iff $M^\psi, s \models p$.
   Then if $M, s \models \psi$, then $M^\psi, s \models p$.
   Then $M, s \models [\psi]p$.
2. Straightforward.
3. Suppose that $M, s \models \Box\varphi$.
   Then, in particular, $M, s \models [\top]\varphi$
   iff $M, s \models \varphi$ (because $M^\top = M$).
4. Suppose that $M, s \models \Diamond\Diamond\neg\varphi$.
   Then there exists $\chi, \chi' \in \mathcal{L}_{\mathsf{EL}}$ such that $M, s \models \langle\chi\rangle\langle\chi'\rangle\neg\varphi$.
   Then there exists $\chi, \chi' \in \mathcal{L}_{\mathsf{EL}}$ such that $M, s \models \langle\chi \wedge \langle\chi\rangle\chi'\rangle\neg\varphi$
   (because $\langle\chi \wedge \langle\chi\rangle\chi'\rangle\neg\varphi \leftrightarrow \langle\chi\rangle\langle\chi'\rangle\neg\varphi$ is valid in PAL).
   Then there exists $\psi \in \mathcal{L}_{\mathsf{EL}}$ such that ψ is equivalent to $\chi \wedge \langle\chi\rangle\chi'$ in PAL.
   Then there exists $\psi \in \mathcal{L}_{\mathsf{EL}}$ such that $M, s \models \langle\psi\rangle\neg\varphi$
   iff $M, s \models \Diamond\neg\varphi$.
5. Suppose that $M, s \models \mathrm{K}_i\Box\varphi$
   iff for all $t \in R_i(s)$, $M, t \models \Box\varphi$
   iff for all $t \in R_i(s)$, for all $\psi \in \mathcal{L}_{\mathsf{EL}}$, $M, t \models \psi$ implies $M^\psi, t \models \varphi$.
   Then for all $t \in R_i(s)$, for all $\psi \in \mathcal{L}_{\mathsf{EL}}$, if $M, s \models \psi$, then $M, t \models \psi$ implies $M^\psi, t \models \varphi$.
   Then for all $\psi \in \mathcal{L}_{\mathsf{EL}}$, if $M, s \models \psi$, then for all $t \in R_i(s)$, $M, t \models \psi$ implies $M^\psi, t \models \varphi$.
   Then for all $\psi \in \mathcal{L}_{\mathsf{EL}}$, if $M, s \models \psi$, then for all $t \in R_i^\psi(s)$, $M^\psi, t \models \varphi$
   iff for all $\psi \in \mathcal{L}_{\mathsf{EL}}$, if $M, s \models \psi$, then $M^\psi, t \models \mathrm{K}_i\varphi$
   iff for all $\psi \in \mathcal{L}_{\mathsf{EL}}$, $M, s \models [\psi]\mathrm{K}_i\varphi$
   iff $M, s \models \Box\mathrm{K}_i\varphi$.
6. Suppose that $\models_{\mathsf{APAL}} \varphi$.
   Then for all $\chi \in \mathcal{L}_{\mathsf{PAL}}$, $\models_{\mathsf{APAL}} [\chi]\varphi$ (by necessitation rule in PAL).
   Then for all $\psi \in \mathcal{L}_{\mathsf{EL}}$, $\models_{\mathsf{APAL}} [\psi]\varphi$.
   Then $\models_{\mathsf{APAL}} \Box\varphi$.

∎

The following theorem (for an arbitrary set of agents N) will be helpful to show that in the single-agent case every formula is equivalent to a formula in $\mathcal{L}_{\mathsf{EL}}$.

THEOREM 74
Let $\varphi, \varphi_0, \ldots, \varphi_n \in \mathcal{L}_{\mathsf{PL}}$, i.e., booleans, and $\psi \in \mathcal{L}_{\mathsf{APAL}}$.

1. $\models_{\mathsf{APAL}} \Box\varphi \leftrightarrow \varphi$
2. $\models_{\mathsf{APAL}} \Box\widehat{\mathrm{K}}_i\varphi \leftrightarrow \varphi$
3. $\models_{\mathsf{APAL}} \Box\mathrm{K}_i\varphi \leftrightarrow \mathrm{K}_i\varphi$
4. $\models_{\mathsf{APAL}} \Box(\varphi \vee \psi) \leftrightarrow (\varphi \vee \Box\psi)$
5. $\models_{\mathsf{APAL}} \Box(\widehat{\mathrm{K}}_i\varphi_0 \vee \mathrm{K}_i\varphi_1 \vee \cdots \vee \mathrm{K}_i\varphi_n) \leftrightarrow (\varphi_0 \vee \mathrm{K}_i(\varphi_0 \vee \varphi_1) \vee \ldots \vee \mathrm{K}_i(\varphi_0 \vee \varphi_n))$

PROOF.

1. We have that $\models_{\mathsf{PAL}} \langle\psi\rangle\varphi \leftrightarrow \varphi$, for all $\psi \in \mathcal{L}_{\mathsf{PAL}}$.
   Therefore, $\models_{\mathsf{APAL}} \Box\varphi \leftrightarrow \varphi$.
2. From the right to the left.
   It follows from the fact that for all $\varphi \in \mathcal{L}_{\mathsf{PL}}$, $\models_{\mathsf{PAL}} \varphi \rightarrow \langle\varphi\rangle\mathrm{K}_i\varphi$.
   From the left to the right. If follows from the fact that for all $\psi \in \mathcal{L}_{\mathsf{EL}}$, $\models_{\mathsf{PAL}} \langle\psi\rangle\mathrm{K}_i\varphi \rightarrow \langle\psi\rangle\varphi$ and that $\varphi \leftrightarrow \Diamond\varphi$ by point 1.
3. From the right to the left holds because axiom T applies for $\Diamond$.
   From the left to the right holds because $\langle\psi\rangle\widehat{\mathrm{K}}_i\varphi \leftrightarrow \widehat{\mathrm{K}}_i\varphi$ is valid in PAL.
4. From the right to the left: first, $\Diamond$ distributes over $\wedge$, and second, $\models_{\mathsf{APAL}} \Diamond\varphi \leftrightarrow \varphi$ by point 1.
   From the left to the right: first, $\varphi \wedge \Diamond\psi$ is equivalent (by point 1) to $\Box\varphi \wedge \Diamond\psi$.
   Now, let χ be an announcement realising ψ in a given pointed structure (M, s), i.e., $M, s \models \langle\chi\rangle\psi$.
   This implies also that $M, s \models \chi$.
   From that and $\Box\varphi$ it follows that $M, s \models \chi \wedge [\chi]\varphi$, i.e., $M, s \models \langle\chi\rangle\varphi$.
   Therefore $M, s \models \langle\chi\rangle(\varphi \wedge \psi)$ so also $M, s \models \Diamond(\varphi \wedge \psi)$.
5. We show this case for $n = 1$.
   From the right to the left.
   Let M, s be arbitrary and suppose $M, s \models \Box(\mathrm{K}_i\varphi_0 \wedge \widehat{\mathrm{K}}_i\varphi_1)$. Let ψ be the epistemic formula such that $M, s \models \langle\psi\rangle(\mathrm{K}_i\varphi_0 \wedge \mathrm{K}_i\varphi_1)$. In the model $M|\psi$ we now have that $M|\psi, s \models \widehat{\mathrm{K}}_i\varphi_1$. Let t be such that $t \in R_i s$, also $M|\psi, t \models \varphi_0$. Therefore $M|\psi, t \models \varphi_0 \wedge \varphi_1$, and therefore, $M|\psi, s \models \widehat{\mathrm{K}}_i(\varphi_0 \wedge \varphi_1)$. So $M|\psi, s \models \varphi_0 \wedge \mathrm{K}_i(\varphi_0 \wedge \varphi_1)$ and as φ_0 and φ_1 are booleans also $M, s \models \varphi_0 \wedge \mathrm{K}_i(\varphi_0 \wedge \varphi_1)$.[1]
   From the left to the right.

[1] Alternatively, one can use more straightforwardly the S5 validity $(\mathrm{K}_i\varphi_0 \wedge \widehat{\mathrm{K}}_i\varphi_1) \rightarrow (\mathrm{K}_i\varphi_0 \wedge \widehat{\mathrm{K}}_i(\varphi_0 \wedge \varphi_1))$.

Suppose that $M, s \models \varphi_0 \wedge \widehat{\mathrm{K}}_i(\varphi_0 \wedge \varphi_1)$. Consider the model $M|\varphi_0, t \models \varphi_1$. So $M|\varphi_0, s \models \widehat{\mathrm{K}}_i\varphi_1$. Also $M|\varphi_0, s \models \mathrm{K}_i\varphi_0$, because φ_0 is boolean. So $M|\varphi_0, s \models \mathrm{K}_i\varphi_0 \wedge \widehat{\mathrm{K}}_i\varphi_1$ and therefore $M, s \models \Diamond(\widehat{\mathrm{K}}_i\varphi_0 \wedge \widehat{\mathrm{K}}_i\varphi_1)$.

∎

6.3.2 Expressivity

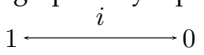
Let $N = \{i\}$. A formula is in *normal form* when it is a conjunction of disjunctions of the form $\varphi \vee \widehat{\mathrm{K}}_i\varphi_0 \vee \mathrm{K}_i\varphi_1 \vee \cdots \vee \mathrm{K}_i\varphi_n$. Every formula in single-agent epistemic logic (K45 and therefore also in) S5 is equivalent to a formula in normal form (Meyer and van der Hoek, 1995).

THEOREM 75
If there is only one agent, every formula in APAL is equivalent to a formula in S5.

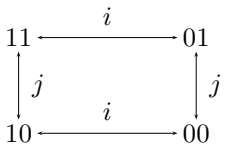
PROOF. By induction on the number of occurrences of $\Box$. Put the epistemic formula in the scope of an innermost $\Box$ in normal form. First, we distribute $\Box$ over the conjunction (by Theorem 73.2). We now get formulas of the form $\Box(\varphi \vee \widehat{\mathrm{K}}_i\varphi_0 \vee \mathrm{K}_i\varphi_1 \vee \cdots \vee \mathrm{K}_i\varphi_n)$. These are reduced by application of Theorem 74.4 and 74.5 to formulas of the form $\varphi_0 \vee \mathrm{K}_i(\varphi_0 \vee \varphi_1) \vee \cdots \vee \mathrm{K}_i(\varphi_0 \vee \varphi_n)$. ∎

THEOREM 76
APAL is strictly more expressive than PAL.

PROOF. Consider the formula $\varphi = \Diamond(\mathrm{K}_i p \wedge \neg\mathrm{K}_j\mathrm{K}_i p)$. Suppose, towards a contradiction, that there exists an epistemic formula ψ that is equivalent to φ. Then, ψ only contains a finite subset of atoms of P. Now, consider the model $M = \langle S, R, V\rangle$ such that $S = \{1, 0\}$, $R_i = \{(1,1),(1,0),(0,1),(0,0)\}$, $R_j = \{(1,1),(0,0)\}$, and $V_p = \{1\}$ for all atoms p in ψ. This model is graphically represented in the figure below:



Now consider the model $M' = \langle S', R', V'\rangle$ such that $S' = \{11, 01, 10, 00\}$, $R'_i = \{(11,11),(11,01),(01,11),(01,01),(10,10),(10,00),(00,10),(00,00)\}$, $R'_j = \{(11,11),(11,10),(10,11),(10,10),(01,01),(01,00),(00,01),(00,00)\}$, $V'_p = \{11, 10\}$ for all atom p in ψ and $V'_q = \{11, 01\}$ for all atom q not in ψ. This model is graphically represented in the figure below:



Clearly, $M, 1 \models \psi$ if and only if $M', 10 \models \psi$. And also $M, 1 \not\models \Diamond(\mathrm{K}_i p \wedge \neg\mathrm{K}_j\mathrm{K}_i p)$. But because $M', 10 \models \langle p \wedge \neg q\rangle(\mathrm{K}_i p \wedge \neg\mathrm{K}_j\mathrm{K}_i p)$, we have that $M', 10 \models \Diamond(\mathrm{K}_i p \wedge \neg\mathrm{K}_j\mathrm{K}_i p)$. Then ψ is not equivalent to φ, which is a contradiction. Therefore such a formula ψ does not exist. ∎

THEOREM 77
Arbitrary public announcement logic is not compact.

PROOF. Take the following infinite set of formulas:

$$\{[\psi](\mathrm{K}_i p \to \mathrm{K}_j \mathrm{K}_i p) \mid \psi \in \mathcal{L}_{\mathsf{EL}}\} \cup \{\neg\Box(\mathrm{K}_i p \to \mathrm{K}_j \mathrm{K}_i p)\}$$

By the semantics of $\Box$, this set is not satisfiable. Nevertheless we show that any finite subset is satisfiable (what contradicts compactness). Let

$$\{[\psi_k](\mathrm{K}_i p \to \mathrm{K}_j \mathrm{K}_i p) \mid 1 \leq k \leq n \text{ and } \psi_k \in \mathcal{L}_{\mathsf{EL}}\} \cup \{\neg\Box(\mathrm{K}_i p \to \mathrm{K}_j \mathrm{K}_i p)\}$$

be a finite subset, and let q be a proposition letter that is distinct from p and does not occur in any of the sentences ψ_k, where $1 \leq k \leq n$. Take now the pointed epistemic model $(M', 10)$ as in the proof of Theorem 76. As shown above, we have $M', 10 \models \Diamond(\mathrm{K}_i p \wedge \neg \mathrm{K}_j \mathrm{K}_i p)$, and thus $M', 10 \models \neg\Box(\mathrm{K}_i p \to \mathrm{K}_j \mathrm{K}_i p)$. On the other hand, for the pointed epistemic model $(M, 1)$ of the proof of Theorem 76, we have that $M, 1 \not\models \Diamond(\mathrm{K}_i p \wedge \neg \mathrm{K}_i \mathrm{K}_i p)$, i.e., $M, 1 \models \Box(\mathrm{K}_i p \to \mathrm{K}_j \mathrm{K}_i p)$. By the semantics of $\Box$, it follows that $M, 1 \models [\psi_k](\mathrm{K}_i p \to \mathrm{K}_j \mathrm{K}_i p)$ for all $1 \leq k \leq n$. But q does not occur in any of these formulas, so their truth-values must be the same at $(M', 10)$ and $(M, 1)$ (since as shown above, the two pointed epistemic models are bisimilar w.r.t. the language without q). Thus we have $M', 10 \models [\psi_k](\mathrm{K}_i p \to \mathrm{K}_j \mathrm{K}_i p)$ for all $1 \leq k \leq n$. Putting these together, we see that our finite set of formulas is satisfied at the state $(M', 10)$. ∎

6.3.3 Knowability

In (Balbiani et al., 2007a), some other interesting results are achieved. For instance, the paper presents the characterisation of interesting fragments of APAL. Namely, 'positive', 'preserved', 'successful' and 'knowable' formulas.

The *positive* $\mathcal{L}_{\mathsf{APAL}}$ *formulas* intuitively correspond to formulas that do not express ignorance, i.e., in epistemic logical ($\mathcal{L}_{\mathsf{EL}}$) terms: in which negations do not precede K_i operators. The fragment of positive formulas is defined by the following BNF:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathrm{K}_i \varphi \mid [\neg\varphi]\varphi \mid \Box\varphi$$

The negation in $[\neg\varphi]\varphi$ is there for technical reasons but unfortunately makes 'positive' somewhat of a misnomer.

The *preserved* $\mathcal{L}_{\mathsf{APAL}}$ *formulas* preserve truth under arbitrary (epistemically definable) model restrictions (relativisation). They are (semantically) defined as those φ for which $\models_{\mathsf{APAL}} \varphi \to \Box\varphi$. There is no corresponding semantic principle in public announcement logic that express truth preservation.

We will prove that positive formulas are preserved. Restricted to epistemic logic without common knowledge, this was observed by van Benthem (2006). van Ditmarsch and Kooi (2006a) extend this to public announcement logic, with clause $[\neg\varphi]\varphi$ (and with common knowledge operators; however without van Benthem's characterisation result). Note that the announcement must be true to be executable,

which, when seen as a disjunction, explains the negation in $[\neg\varphi]\varphi$. Surprisingly, we can expand this fragment with $\Box\varphi$ for arbitrary announcement logic.

THEOREM 78 (Balbiani et al. (2007a))
Positive formulas are preserved.

We do not know whether the preserved formulas are (logically equivalent to) positive. An answer to this question seems hard. It would extend van Benthem's results in (van Benthem, 2006).

Another semantic notion is that of 'success'. 'Successful formulas' are believed after their announcement, or, in other words, after 'revision' with that formula. This precisely corresponds to the postulate of 'success' in AGM belief revision (Alchourrón et al., 1985). Formally, φ is a *successful formula* iff $[\varphi]\varphi$ is valid (see van Ditmarsch and Kooi (2006a), elaborating an original but slightly different proposal made by Gerbrandy and Groeneveld (1997)). The validity of $[\varphi]\varphi$ corresponds to the validity of $\varphi \rightarrow [\varphi]\mathrm{K}_i\varphi$ (van Ditmarsch and Kooi, 2006a): announced formulas are believed after their true announcement.

THEOREM 79
Preserved formulas are successful.

PROOF. If $\models_{\mathsf{APAL}} \varphi \rightarrow \Box\varphi$, then $\models_{\mathsf{APAL}} \varphi \rightarrow [\varphi]\varphi$ which is equivalent to $\models_{\mathsf{APAL}} [\varphi]\varphi$. ∎

COROLLARY 80
Positive formulas are successful.

Fitch investigated the formulas that, if true, can become known (Brogaard and Salerno, 2004). Consider a multi-agent version. We define the *knowable formulas* as those for which, for all agents $i \in N$, $\models_{\mathsf{APAL}} \varphi \rightarrow \Diamond\mathrm{K}_i\varphi$. We can now observe that, e.g.:

THEOREM 81
Positive, preserved and successful formulas are knowable.

PROOF. Observe that $\varphi \rightarrow \Box\varphi$ implies $\varphi \rightarrow [\varphi]\mathrm{K}_i\varphi$; and this implies $\varphi \rightarrow \langle\varphi\rangle\mathrm{K}_i\varphi$; and this implies $\varphi \rightarrow \Diamond\mathrm{K}_i\varphi$. ∎

We did not investigate the knowable formulas in depth. Some knowable formulas are not positive, for example $\neg\mathrm{K}_i p$: if true, announce $\top$ and $\mathrm{K}_i\neg\mathrm{K}_i p$ (still!) holds. Therefore, $\neg\mathrm{K}_i p \rightarrow \Diamond\mathrm{K}_i\neg\mathrm{K}_i p$.

6.4 Axiomatisation

In this section we propose an axiomatisation for $\mathcal{L}_{\mathsf{APAL}}$. Let $\sharp$ be a new symbol and let $\varphi \in \mathcal{L}_{\mathsf{APAL}}$. Following the technique suggested by Goldblatt (1982), we define *necessity forms* inductively as follows:

- $\sharp$ is a necessity form;
- if $\boldsymbol{\psi}$ is a necessity form, then $\varphi \to \boldsymbol{\psi}$ is a necessity form;
- if $\boldsymbol{\psi}$ is a necessity form, then $[\varphi]\boldsymbol{\psi}$ is a necessity form; and
- if $\boldsymbol{\psi}$ is a necessity form,then $\mathrm{K}_i\boldsymbol{\psi}$ is a necessity form.

Note that each necessity form has a unique occurrence of $\sharp$. If $\boldsymbol{\psi}$ is a necessity form and $\varphi \in \mathcal{L}_{\mathsf{APAL}}$, then $\boldsymbol{\psi}(\varphi)$ is obtained from $\boldsymbol{\psi}$ by replacing $\sharp$ in $\boldsymbol{\psi}$ with φ. Necessity forms are used in the derivation rule $\mathrm{R}(\Box)$ of the axiomatisation, shown below.

PL.	all tautologies from classical propositional logic
K(K).	$\mathrm{K}_i(\varphi \to \psi) \to (\mathrm{K}_i\varphi \to \mathrm{K}_i\psi)$
T(K).	$\mathrm{K}_i\varphi \to \varphi$
4(K).	$\mathrm{K}_i\varphi \to \mathrm{K}_i\mathrm{K}_i\varphi$
5(K).	$\neg\mathrm{K}_i\varphi \to \mathrm{K}_i\neg\mathrm{K}_i\varphi$
PA1.	$[\varphi]p \leftrightarrow (\varphi \to p)$
PA2.	$[\varphi]\neg\psi \leftrightarrow (\varphi \to \neg[\varphi]\psi)$
PA3.	$[\varphi](\psi \wedge \chi) \leftrightarrow ([\varphi]\psi \wedge [\varphi]\chi)$
PA4.	$[\varphi]\mathrm{K}_i\psi \leftrightarrow (\varphi \to \mathrm{K}_i[\varphi]\psi)$
PA5.	$[\varphi][\psi]\chi \leftrightarrow [(\varphi \wedge [\varphi]\psi)]\chi$
APA.	$\Box\varphi \to [\psi]\varphi$ for $\psi \in \mathcal{L}_{\mathsf{EL}}$
MP.	From $\vdash_{\mathsf{APAL}} \varphi$ and $\vdash_{\mathsf{APAL}} \varphi \to \psi$, infer $\vdash_{\mathsf{APAL}} \psi$
N(K).	From $\vdash_{\mathsf{APAL}} \varphi$, infer $\vdash_{\mathsf{APAL}} \mathrm{K}_i\varphi$
N([·]).	From $\vdash_{\mathsf{APAL}} \varphi$, infer $\vdash_{\mathsf{APAL}} [\psi]\varphi$
N($\Box$).	From $\vdash_{\mathsf{APAL}} \varphi$, infer $\vdash_{\mathsf{APAL}} \Box\varphi$
R($\Box$).	From $\vdash_{\mathsf{APAL}} \boldsymbol{\varphi}([\chi]\psi)$ for all $\chi \in \mathcal{L}_{\mathsf{EL}}$, infer $\vdash_{\mathsf{APAL}} \boldsymbol{\varphi}(\Box\psi)$

DEFINITION 82
A formula $\varphi \in \mathcal{L}_{\mathsf{APAL}}$ is a *theorem* (notation: $\vdash_{\mathsf{APAL}} \varphi$) if and only if it belongs to the least set of formulas containing all axioms and closed under the derivation rules.

All axioms and rules are sound. In particular, the rule $\mathrm{R}(\Box)$ is correct with respect to the semantics, i.e., if $\models_{\mathsf{APAL}} \boldsymbol{\varphi}([\chi]\psi)$ for all $\chi \in \mathcal{L}_{\mathsf{EL}}$, then $\models_{\mathsf{APAL}} \boldsymbol{\varphi}(\Box\psi)$.

In the above formulation, the rule $\mathrm{R}(\Box)$ is infinitary. In (Balbiani et al., 2007a), it is shown that this rule can be replaced by the following:

$\mathrm{R}'(\Box)$	From $\vdash_{\mathsf{APAL}} \boldsymbol{\varphi}([p]\psi)$, infer $\vdash_{\mathsf{APAL}} \boldsymbol{\varphi}(\Box\psi)$ where p does not occur in $\boldsymbol{\varphi}$ nor in ψ

EXAMPLE 83
For an example of derivation in the above axiomatisation using the axiom and rule for $\Box$, we show that the valid formula $\Box\varphi \rightarrow \Box\Box\varphi$ is also a theorem. Note that in step 4 of the derivation we use that $\Box p \rightarrow [q]\sharp$ is a necessity form and that in step 5 of the derivation we use that $\Box p \rightarrow \sharp$ is a necessity form.

1.	$\vdash_{\mathsf{APAL}} [(q \wedge [q]r)]p \leftrightarrow [q][r]p$	by PA5
2.	$\vdash_{\mathsf{APAL}} \Box p \rightarrow [(q \wedge [q]r)]p$	by APA
3.	$\vdash_{\mathsf{APAL}} \Box p \rightarrow [q][r]p$	by 1, 2, PL and MP
4.	$\vdash_{\mathsf{APAL}} \Box p \rightarrow [q]\Box p$	by 3 and $\mathrm{R}'(\Box)$
5.	$\vdash_{\mathsf{APAL}} \Box p \rightarrow \Box\Box p$	by 4 and $\mathrm{R}'(\Box)$

The main effect of rule $\mathrm{R}(\Box)$ is that it makes the canonical model (consisting of all maximal consistent sets of formulas closed under the rule) standard for $\Box$. Let us see how. In the remainder of this section, most proof details are omitted. A complete proof can be found in (Balbiani et al., 2007a).

A set x of formulas is called a theory if it satisfies the following conditions:

- x contains the set of all theorems of $\mathcal{L}_{\mathsf{APAL}}$; and
- x is closed under modus ponens and $\mathrm{R}(\Box)$.

Obviously the least theory is the set of all theorems and the largest theory is the set of all formulas. The latter theory is called the trivial theory. A theory x is said to be consistent if $\bot \notin x$. We shall say that a theory x is maximal if for all formulas φ, $\varphi \in x$ or $\neg\varphi \in x$. Let x be a set of formulas. For all formulas φ, let $x + \varphi = \{\psi \mid \varphi \rightarrow \psi \in x\}$. For all agents i, let $\mathrm{K}_i x = \{\varphi \mid \mathrm{K}_i\varphi \in x\}$. For all formulas φ, let $[\varphi]x = \{\psi\colon [\varphi]\psi \in x\}$.

LEMMA 84
Let x be a theory, φ be a formula, and i be an agent. Then $x + \varphi$, $\mathrm{K}_i x$ and $[\varphi]x$ are theories. Moreover $x + \varphi$ is consistent if and only if $\neg\varphi \notin x$.

PROOF. The proof is based on the fact that x is closed under modus ponens and $\mathrm{R}(\Box)$. ∎

LEMMA 85
Let x be a consistent theory. There exists a maximal consistent theory y such that $x \subseteq y$.

PROOF. This is the Lindenbaum Lemma for the arbitrary announcement logic. The proof can be done as in (Goldblatt, 1982). ∎

DEFINITION 86 (CANONICAL MODEL)
The canonical model of $\mathcal{L}_{\mathsf{APAL}}$ is the structure $M_c = \langle S, R, V\rangle$ defined as follows:

- S is the set of all maximal consistent theories;

- For all agents i, R_i is the binary (equivalence) relation on S defined by xR_iy iff $\mathrm{K}_i x = \mathrm{K}_i y$;
- For all atoms p, V_p is the subset of S defined by $x \in V_p$ iff $p \in x$.

LEMMA 87 (TRUTH)
Let φ be a formula in $\mathcal{L}_{\mathsf{APAL}}$. Then for all maximal consistent theories x and for all finite sequences $\psi^k = (\psi_1, \ldots, \psi_k)$ of formulas in $\mathcal{L}_{\mathsf{APAL}}$ such that $\psi_1 \in x$, $[\psi_1]\psi_2 \in x$, $\ldots$, $[\psi_1] \ldots [\psi_{k-1}]\psi_k \in x$:

$$M_c^{\psi^k}, x \models \varphi \Leftrightarrow [\psi_1] \ldots [\psi_k]\varphi \in x$$

PROOF. The proof is by induction on φ. ∎

As a result we have:

THEOREM 88 (SOUNDNESS AND COMPLETENESS)
Let φ be a formula. Then φ is a theorem iff φ is valid.

PROOF. Soundness is immediate, following the observations at the beginning of this section. Completeness follows from Lemmas 84, 85, and 87. ∎

6.5 A Tableau Method for Arbitrary Announcements

Now, we generalise the method given in section 5.2 to arbitrary announcements. We reuse labelled formulas for $\mathcal{L}_{\mathsf{APAL}}$, as introduced in definition 54.

DEFINITION 89 (TABLEAU (CONTINUATION))
A *tableau* for the formula $\varphi \in \mathcal{L}_{\mathsf{APAL}}$ is defined as in Definition 56. The tableau rules are the same, plus the following ones:

R$\Box$: If $(\psi^k, n, \Box\varphi) \in \Lambda$, then $B = \{\langle \Lambda \cup \{(\psi^k : n : [\chi]\varphi)\}, \Sigma\rangle\}$ for any $\chi \in \mathcal{L}_{\mathsf{EL}}$.

R$\Diamond$: If $(\psi^k, n, \neg\Box\varphi) \in \Lambda$, then $B = \{\langle \Lambda \cup \{(\psi^k : n : \neg[p]\varphi)\}, \Sigma\rangle\}$ for some $p \in P$ that does not occur in Λ.

These rules are similar to Smullyan's tableau rules for closed first-order formulas (Smullyan, 1968; Letz, 1999). They reflect that the operator $\Box$ quantifies over announcements. In tableau rule $\Box$, this operator is eliminated by replacing it by an arbitrary formula in $\mathcal{L}_{\mathsf{EL}}$. Tableau rule R$\Diamond$ is more curious though: instead of replacing the operator by an announcement of a formula in $\psi \in \mathcal{L}_{\mathsf{EL}}$, we replace it by an announcement of a new propositional letter. The intuitive argument here is the following. Since this new propositional letter does not occur in the branch, we are free to give it an arbitrary interpretation to represent a specific restriction in the model. In this way, we make the calculus simpler because it is not necessary to make a 'good choice' at the moment of the application of rule R$\Diamond$.

1.	$\epsilon, 0, \neg[\neg\Box\neg \mathrm{K}_i p]\mathrm{K}_i p$		
2.	$\epsilon, 0, \neg\Box\neg \mathrm{K}_i p$		$(\mathrm{R}\langle\cdot\rangle : 1)$
3.	$\neg\Box\neg \mathrm{K}_i p, 0, \neg \mathrm{K}_i p$		$(\mathrm{R}\langle\cdot\rangle : 1)$
4.	$\epsilon, 0, \neg[q]\neg \mathrm{K}_i p$		$(\mathrm{R}\Diamond : 2)$
5.	$\epsilon, 0, q$		$(\mathrm{R}\langle\cdot\rangle : 4)$
6.	$q, 0, \neg\neg \mathrm{K}_i p$		$(\mathrm{R}\langle\cdot\rangle : 4)$
7.	$q, 0, \mathrm{K}_i p$		$(\mathrm{R}\neg : 6)$
8.	$\epsilon, 1, \neg\Box\neg \mathrm{K}_i p$	$(i, 0, 1) \in \Sigma$	$(\mathrm{R}\widehat{\mathrm{K}} : 3)$
9.	$\neg\Box\neg \mathrm{K}_i p, 1, \neg p$		$(\mathrm{R}\widehat{\mathrm{K}} : 3)$
10.	$q, 1, p$		$(\mathrm{RK} : 7)$
	closed		$(9, 10)$

Figure 6.1: Closed tableau for the formula $[\Diamond \mathrm{K}_i p]\mathrm{K}_i p$.

EXAMPLE 90
Consider the formula $[\Diamond \mathrm{K}_i p]\mathrm{K}_i p$. Note that it is valid in APAL since its announcement corresponds to the sentence: 'there is an announcement after which agent i knows that p'. That is, it is publicly announced that p can be known. This means that p is true and thus now agent i knows it. In Figure 6.1 we use the tableau method to show that this formula is APAL-valid.

THEOREM 91 (SOUNDNESS AND COMPLETENESS (CONTINUATION))

There is a closed APAL-tableau for $\neg\varphi$ if and only if φ is APAL-valid.

PROOF. This is an easy extension of the proof of theorem 59. The details are in Appendix A.5. ∎

This tableau method can be used for giving us a proof of semi-decidability of this logic.

THEOREM 92
The set of APAL-valid formulas of $\mathcal{L}_{\mathsf{APAL}}$ is recursively enumerable.

PROOF. First note that the same argument used in the proof of Theorem 63 can be used to show that each tableau rule generates a finite tableau. Then, by completeness, we have that for all formulas φ, all closed tableaux for φ are finite. Then, consider a procedure that enumerates all pairs (φ, T) such that T is a closed tableau. For each pair, the procedure verifies if T is a tableau for $\neg\varphi$. When the checking is finished, it generates another pair and performs another round of checking, and so on ad infinitum. ∎

6.6 Discussion, Further Work and Conclusion

We proposed an extension of public announcement logic with an operator $\Box$ that expresses what is true after arbitrary announcements. We proved several validities involving that operator, gave a sound and complete infinitary axiomatisation, and a labelled tableau calculus.

At a first glance, it may seem that the operator $\Box$ of APAL is the same as the one of the logic ES (Lakemeyer and Levesque, 2004) (see also Lakemeyer and Levesque, 2005), dicussed in Section 2.3.5. There are some differences between them though. In ES this operator quantifies over atomic actions, while in APAL it quantifies over formulas. In addition, all epistemic actions are sensing actions in ES, while in APAL they are public announcements. And finally, and most importantly, the ES operator also quantifies over ontic actions. Nevertheless, it is worth noting that we were able to show in this chapter a reduction (or regression) procedure for the APAL operator $\Box$ in single-agent settings. We leave as an open question, whether this procedure can be exteded for the operator $\Box$ of logic ES.

Some further issues in the logic APAL and in its tableau calculus might be interesting and fruitful to pursue. For instance, we know that the model checking problem in APAL is decidable under the assumption that models are finite and that only a finite number of atoms changes value in a given model. This result is not trivial because of the quantification implicit in the operator $\Box$.

In addition, we hope that if the logic is decidable, an adapted version of the tableau calculus proposed here could be used for planning. This is possible along the same idea proposed by Castilho et al. (1999). Suppose that we have a closed tableau for the formula:

$$\varphi \wedge \Box\neg\psi$$

where φ describes an initial situation and ψ a goal. This means that the formula $\varphi \rightarrow \Diamond\psi$ is valid, i.e., the goal ψ is reachable. Then we take all the branches of the closed tableau and remove those branches that are closed without involving a subformula of $\Box\neg\psi$. For the remaining branches we extract finite sequences $\psi^k = (\psi_1 \ldots \psi_k)$ of announcements. Then the (nondeterministic) plan associated to the tableau is the nondeterministic composition of all these sequences of announcements. For example, suppose that $\varphi = \top$ and $\psi = \mathrm{K}_i p \vee \mathrm{K}_i \neg p$. In this case, the nondeterministic plan would be $p \cup \neg p$.

Another interesting research track would be the expansion of the language of APAL with additional modal operators. Notably, common knowledge and public assignments. However, if we apply a recent unpublished result of van Ditmarsch and Kooi (2006b), we have that the introduction of public assignments in APAL leads to a trivialisation of the problem: in finite models all satisfiable formulas are realizable in such a logic, or, more formally, given arbitrary finite (M, s) and satisfiable φ, there is always a sequence of announcements and assignments α such that $(M, s) \models \langle\alpha\rangle\varphi$. In other words, every (satisfiable) goal is reachable!

Then it seems that the idea of quantifying over actions does not lead to a good formalisation of planning. Remember though, that the operator $\Box$ quantifies over the

whole set of actions. In a realistic setting, each agent should have a set of available actions and the quantification should be done over these sets. For example, a blind agent should not be *able* to announce that the light is on, where the concept of *ability* used here is the one defined by van der Hoek et al. (1994) (see also van der Hoek et al., 2000). In addition, even if the action is available for the agent, it must be executable *by the agent*. To use the same terminology of Section 2.5.2, the action must be meaningful. For example, the agent should not be able to announce that the light is on if he does not know wheather the light is on or not. We believe that with such restrictions the problem will not be trivial any more.

Chapter 7

Conclusion

The research performed in this thesis started from the idea of extending the approach of Demolombe et al. (2003) to account for epistemic actions. In this framework, the Reiter (1991) solution to the frame problem – which addresses only factual change – is encoded into the modal logic EDL. One of the restrictions imposed is that all actions should respect the no-learning principle. At first glance, it seemed improbable that a genuine (or even meaningful) epistemic action could respect this principle. However, we find a kind of epistemic action that does: observations (Chapter 3) respect no-learning, and are indeed able to make agents' epistemic states evolve.

Observations are an interesting kind of epistemic action, but do not seem to be expressive enough. For instance, they are different from sensing actions, formalised for example in the framework of Scherl and Levesque (2003). This is why we needed to demonstrate that all epistemic actions can be encoded by "complex" observations. This result is stated as the 'observations are general' theorem in Section 3.3.

After that, we showed that observations were nothing but public announcements, and that factual change in EDL were very similar to public assignments. Public announcements and public assignments are actions present in dynamic epistemic logic (DEL). DEL is a family of logics studied in the "Dutch tradition" of reasoning about actions and knowledge. In DEL, actions are viewed as model transformations. By trying to understand the relations between all these notions, we ended up by realizing that both approaches – EDL and DEL – could, so to speak, do the same job. The technical result is the polynomial translation from EDL to DEL of Section 4.4.

The existing reasoning method available for DEL, made up of reduction axioms, were too resource consuming. The next step then was to find an efficient method allowing to automatically reasoning in this setting. That is, we started to look for a solution to the inferential frame problem: the problem of finding an efficient reasoning method for a formalism that solves the representational frame problem. At this point, we splitted the work into two research tracks. In one of them, we extended an existing proof method, proposed by Lutz (2006), that accounts for a fragment of DEL. In the other track, we developed a tableau method for DEL.

The first track turned out to be more fruitful and we ended up with an optimal

proof method for the full logic DEL as we wanted. The extension of Lutz' method proposed here, also accounts for ontic change, and is the only optimal proof method for this logic up to now.

The second track did not give us a proof method for the full logic DEL, but resulted in a novel proof method for one of its fragments: public announcement logic (PAL). We also provided optimal tableau strategies for deciding satisfiability in two special cases: K-PAL and KT-PAL. And in addition, we have reasons to believe that optimal strategies for S4-PAL and S5-PAL exist.

Note that these results also contribute to bringing together two different research communities of reasoning about actions and knowledge. The "Toronto community", whose adepts developed several extensions to the situation calculus, and the "Amsterdam community", whose adepts use dynamic epistemic logic formalisations of, essentially, the same problems. Some ideas about the similarities between situation calculus and dynamic epistemic logic were independently outlined by van Benthem (2007). Here we go further, and formally prove the connections between them.

The last objective pursued in this work was a proper treatment of the task of planning. Note that plan verification, the problem that we address using DEL, can be seen as validity checking. However, this is not the case for planning, because the latter demands the construction of a plan. We therefore designed another formalism called arbitrary public announcement logic (APAL). This is our first effort towards a proper treatment of the planning task using modal logic. In fact, we do not really address planning using APAL, but a still different problem that lies between plan verification and planning: the *plan existence problem*. Even for this task though, APAL is not ideal. The first reason is that APAL does not account for ontic change. The second reason is that if we permit that the diamond operator of APAL quantifies over plans, then they must be meaningful plans, as defined in Section 2.5.2. The third, and main, reason is that APAL quantifies over the set of all epistemic actions, and not over the set of "possible epistemic actions for each agent", as would be more adequate. To exemplify why this distinction is important, suppose a blind agent in a room. This agent should not be able to observe (by means of its own capabilities) whether the light is on or off. Because each agent has limited capabilities and cannot execute whatever observation, the quantification over all of them is not realistic. Many results for APAL were achieved however. We provided a Hilbert-style axiomatisation and a tableau method. And we also provided some results concerning its expressivity.

A straightforward extension of the tableau method defined in Chapter 5 should be enough to permit the definition of decision procedures for DEL with public assignments (ontic change). The rules to be added are simple. Note that if $[p := \psi]\varphi$ is satisfiable, then either both ψ and p are false after the assignment, or both ψ and p are true after the assignment. This corresponds to a kind of cut rule, as the following one:

$\mathrm{R} :=$ if $(\psi^k, x, [p := \psi]\varphi) \in \Lambda$, then
$B = \{(\Lambda \cup \{(\psi^k, x, \neg\psi), (\psi^k(p := \psi), x, \neg p), (\psi^k(p := \psi), x, \varphi)\}, \Sigma), (\Lambda \cup \{(\psi^k, x, \psi),$
$(\psi^k, (p := \psi), x, p), (\psi^k(p := \psi), x, \varphi)\}, \Sigma)\}$.

It is not clear however, whether this extension can lead to optimal decision procedures.

One of the most interesting questions raised by this work was already mentioned in Section 4.6. Our proof method would not be optimal if sensing actions $!!\varphi$ were defined as primitive operators in DEL. It is not clear how the associated reduction axiom:

$$\mathrm{K}_i[!!\varphi]\psi \leftrightarrow ((\varphi \rightarrow \mathrm{K}_i(\varphi \rightarrow [!!\varphi]\psi)) \wedge (\neg\varphi \rightarrow \mathrm{K}_i(\neg\varphi \rightarrow [!!\varphi]\psi)))$$

can be "efficiently" integrated in Lutz' reduction. We know that satisfiability in DEL with sensing actions is PSPACE-hard. The form of the reduction axioms implies that this problem is in EXPSPACE. We leave as an open question the precise complexity class of the satisfiability problem of the logic with sensing actions.

One could also introduce other kinds of actions into the present framework. An (almost) straightforward extension is the integration of non-public actions. This can be done by enriching the action descriptions defined in Section 4.2 with binary accessibility relations $R : N \rightarrow (A \times A)$. Then action descriptions take a form that is similar to action models proposed by Baltag et al. (1998). In Baltag et al.'s approach, actions in A are linked by arrows that are labelled by agents in N. We interpret this as follows: if $(a, b) \in R_i$, then whenever action a occurs, agent i believes that action b occurs. In particular, when $b = \mathsf{skip}$, the occurrence of a is not perceived by agent i. Therefore, this action is not public. By using a slightly different version of DEL, Baltag et al. define model transformations for this kind of actions. Reduction axioms are also provided. From the latter and the results achieved by this thesis, it follows that non-public actions can also be easily introduced in the framework of situation calculus.

Bibliography

Carlos Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50:510–530, 1985.

Fahiem Bacchus, Joseph Halpern, and Hector Levesque. Reasoning about noisy sensors and effectors in the situation calculus. *Artificial Intelligence*, 111(1–2):171–208, 1999.

Philippe Balbiani, Alexandru Baltag, Hans van Ditmarsch, Andreas Herzig, Tomohiro Hoshi, and Tiago de Lima. What can we achieve by arbitrary announcements? A dynamic take on Fitch's knowability. In Dov Samet, editor, *Proceedings of the eleventh Theoretical Aspects of Rationality and Knowledge conference (TARK)*, pages 42–51. Presses universitaires de Louvain, 2007a.

Philippe Balbiani, Hans van Ditmarsch, Andreas Herzig, and Tiago de Lima. A tableau method for public announcement logics. In N. Olivetti, editor, *Proceedings of the International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX)*, volume 4548 of *Lecture Notes in Artificial Intelligence (Lecture Notes in Computer Science)*, pages 43–59. Springer-Verlag, 2007b.

Alexandru Baltag and Lawrence Moss. Logics for epistemic programs. *Synthese*, 139 (2):165–224, 2004.

Alexandru Baltag, Lowrence Moss, and Slawomir Solecki. The logic of common knowledge, public announcements, and private suspicions. In *Proceedings of the seventh Theoretical Aspects of Rationality and Knowledge conferene (TARK)*, pages 43–46. Morgan Kaufmann Publishers Inc., 1998.

Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic*. Cambridge University Press, 2001a.

Patrick Blackburn, Jaap Kamps, and Maarten Marx. Situation calculus as hybrid logic: First steps. In P. Brazdil and A. Jorge, editors, *Proceedings of the Tenth Portuguese Conference on Artificial Intelligence (EPIA)*, volume 2258 of *Lecture Notes in Computer science*, pages 253–260. Springer-Verlag, 2001b.

Avrim Blum and Merrick Furst. Fast planning through planning graph analysis. *Artificial Intelligence*, 90:281–300, 1997.

Blai Bonet and Hector Geffner. Planning and control in artificial intelligence: A unifying perspective. *Applied Intelligence*, 14(3):237–252, 2001.

Craig Boutilier, Thomas Dean, and Steve Hanks. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11:1–94, 1999.

Berit Brogaard and Joe Salerno. Fitch's paradox of knowability. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. The Metaphysics Research Lab, Center for the Study of Language and Information, Stanford University, Summer 2004. URL `http://plato.stanford.edu/archives/sum2004/entries/fitch-paradox/`.

Tom Bylander. The computational complexity of propositional strips planning. *Artificial Intelligence*, 69(1–2):165–204, 1994.

Marcos Castilho, Luis Fariñas del Cerro, Olivier Gasquet, and Andreas Herzig. Modal tableaux with propagation rules and structural rules. *Fundamenta Informaticae*, 32(3/4):1–17, 1997.

Marcos Castilho, Olivier Gasquet, and Andreas Herzig. Formalizing action and change in modal logic I: the frame problem. *Journal of Logic and Computation*, 9 (5):701–735, 1999.

Alessandro Cimatti, Enrico Giunchiglia, Fausto Giunchiglia, and Paolo Traverso. Planning via model checking: a decision procedure for AR. In S. Steel and R. Alami, editors, *Proceedings of the Forth European Conference on Planning (ECP)*, Lecture Notes in Computer Science, Volume 1348, pages 130–142. Springer Verlag, September 24–26 1997. ISBN 3-540-63912-8.

Mathijs de Boer. Praktische bewijzen in public announcement logica (Practical proofs in public announcement logic). Master's thesis, Department of Artificial Intelligence, University of Groningen, 2006. in Dutch.

Robert Demolombe, Andreas Herzig, and Ivan Varzinczak. Regression in modal logic. *Journal of Applied Non-Classical Logics*, 13(2):165–185, 2003.

Ronald Fagin, Joseph Halpern, Yoram Moses, and Moshe Vardi. *Reasoning about Knowledge*. The MIT Press, 1995.

Richard Fikes and Nils Nilsson. STRIPS: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3–4):189–208, 1971.

Kit Fine. Propositional quantifiers in modal logic. *Theoria*, 36(3):336–346, 1970.

Melvin Fitting. *Proof Methods for Modal and Intuitionistic Logics*. Reidel Publishing Company, 1983.

Michael Gelfond and Vladimir Lifschitz. Representing action and change by logic programs. *Journal of Logig Programming*, 17(2/3&4):301–321, 1993.

Michael Gelfond, Vladimir Lifschitz, and Arkady Rabinov. What are the limitations of situation calculus. In Robert Boyer, editor, *Essays in Honor of Woody Bledsoe*, pages 167–180. Kluwer Academic Publishers Group, 1991.

Jelle Gerbrandy. *Bisimulations on Planet Kripke*. PhD thesis, ILLC, University of Amsterdam, 1999.

Jelle Gerbrandy and Willem Groeneveld. Reasoning about information change. *Journal of Logic, Language and Information*, 6(2):147–169, 1997.

Robert Goldblatt. *Axiomatising the Logic of Computer Programming*, volume 130 of *Lecture Notes in Computer Science*. Springer, 1982.

Andrew Haas. The case for domain-specific frame axioms. In F. M. Brown, editor, *The Frame Problem in Artificial Intelligence*. Morgan Kaufmann, 1987.

Joseph Halpern and Yoram Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54:311–379, 1992.

David Harel, Dexter Kozen, and Jerzy Tiuryn. *Dynamic Logic*. Foundations of Computing Sereies. The MIT Press, 2000.

Andreas Herzig and Tiago de Lima. Epistemic actions and ontic actions: A unified logical framework. In Jaime Simão Sichman, Helder Coelho, and Solange Oliveira Rezende, editors, *Advances in Artificial Intelligence – IBERAMIA-SBIA 2006*, volume 4140 of *Lecture Notes in Artificial Intelligence (Lecture Notes in Computer Science)*, pages 409–418. Springer-Verlag, 2006.

Andreas Herzig, Jérôme Lang, Dominique Longin, and Thomas Polacsek. A logic for planning under partial observability. In *Proceedings of the Seventeenth Conference on Artificial Intelligence (AAAI) and the Twelfth Conference on Innovative Applications of Artificial Intelligence (IAAI)*, pages 768–773. The AAAI Press, 2000a.

Andreas Herzig, Jérôme Lang, and Thomas Polacsek. A modal logic for epistemic tests. In Werner Horn, editor, *Proceedings of the Fourteenth European Conference on Artificial Intelligence (ECAI)*, pages 553–557. IOS Pres, 2000b.

Jaakko Hintikka. *Knowledge and Belief*. Cornell University Press, 1962.

Leslie Kaelbling, Michael Littman, and Anthony Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, 1998.

Henry Kautz and Bart Selman. Planning as satisfiability. In J. LLoyd, editor, *Proceedings of the Tenth European Conference on Artificial Intelligence (ECAI)*, pages 359–379, 1992.

Henry Kautz and Bart Selman. Unifying SAT-based and graph-based planning. In Thomas Dean, editor, *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 318–325, San Francisco, 1999. Morgan Kaufmann.

Barteld Kooi. Expressivity and completeness for public update logic via reduction axioms. *Journal of Applied Non-Classical Logics*, 17(2):231–253, 2007.

Robert Koons and Nicholas Asher. Belief revision in a changing world. In *Proceedings of the eleventh Theoretical Aspects of Rationality and Knowledge conference (TARK)*, pages 321–340. ACM, 1994.

Gerhard Lakemeyer and Hector Levesque. Situations, si! Situation terms, no! In *Proceedings of the International Conference on Knowledge Representation and Reasoning (KR)*, pages 516–526. AAAI Press, 2004.

Gerhard Lakemeyer and Hector Levesque. Semantics for a useful fragment of the situation calculus. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 490,496. Professional Book Center, 2005.

Reinhold Letz. Tableau methods for modal and temporal logics. In M. D'Agostino, D. Gabbay, R. Hähnle, and J. Possega, editors, *Handbook of Tableau Methods*, pages 125–196. Kluwer Academic Publishers, 1999.

Hector Levesque. What is planning in presence of sensing? In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI) and the Eighth Annual Conference on Innovative Applications of Artificial Intelligence (IAAI)*, volume 2, pages 1139–1146. The AAAI Press, 1996.

Hector Levesque, Reymond Reiter, Yves Lespérance, Fangzhen Lin, and Richard Scherl. GOLOG: A logic programming language for dynamic domains. *Journal of Logic Programming*, 31(1–2):59–83, 1997.

David Lewis. *Convention, A Philosophical Study*. Harvard University Press, 1969.

Carsten Lutz. Complexity and succintness of public announcement logic. In P. Stone and G. Weiss, editors, *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 137–144, 2006.

Stephen Majercik and Michael Littman. Contingent planning under uncertainty via stochastic satisfiability. *Artificial Intelligence*, 147(1-2):119–162, 2003.

John McCarthy. Situations, actions and causual laws. In M. Minsky, editor, *Semantic Information Processing*, pages 410–417. The MIT Press, 1968.

John McCarthy and Patrick Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, 1969.

John-Jules Meyer and Wiebe van der Hoek. *Epistemic Logic for AI and Computer Science*. Number 41 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1995.

Robert Moore. Reasoning about knowledge and action. Technical Note 191, SRI International, 1980.

Andreas Nonnengart and Christoph Weidenbach. Computing small clause normal forms. In *Handbook of Automated Reasoning*, pages 335–367. North Holland, 2001.

J. Scott Penberthy and Daniel Weld. UCPOP: A sound, complete, partial order planner for ADL. In *Proceedings of the International Conference on Knowledge Representation and Reasoning (KR)*, pages 103–114, 1992.

Jan Plaza. Logics of public communications. In M. L. Emrich, M. Hadzikadic, M. S. Pfeifer, and Z. W. Ras, editors, *Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems (ISMIS)*, pages 201–216, 1989.

Reymond Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In V. Lifschitz, editor, *Papers in Honor of John McCarthy*, pages 359–380. Academic Press Professional Inc., 1991.

Reymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press, 2001a.

Reymond Reiter. On knowledge-based programming with sensing in the situation calculus. *ACM Transactions on Computational Logic*, pages 433–437, 2001b.

Erik Sandewall. Features and fluents. A systematic approach to the representation of knowledge about dynamical systems. Technical Report LiTH-IDA-R-92-30, IDA, Linköping Univesity, 1992.

Erik Sandewall. *Features and Fluents. The Representation of Knowledge about Dynamical Systems*, volume 1. Oxford University Press, 1995.

Richard Scherl and Hector Levesque. The frame problem and knowledge-producing actions. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI)*, pages 689–695. The AAAI Press, 1993.

Richard Scherl and Hector Levesque. Knowledge, action and the frame problem. *Artificial Intelligence*, 144(1–2):1–39, 2003.

Lenhart Schubert. Monotonic solution of the frame problem in the situation calculus: An efficient method for worlds with fully specified actions. In H. Kynburg, R. Loui, and G. Carlson, editors, *Knowledge Representation and Defeasible Reasoning*, pages 23–67. Kluwer Academic Publishing, 1990.

Murray Shanahan. *Solving the Frame Problem*. The MIT Press, 1997.

Steven Shapiro, Maurice Pagnucco, Yves Lespérance, and Hector Levesque. Iterated belief change in the situation calculus. In *Proceedings of the Seventh International Conference on Principles and Knowledge Representation and Reasoning (KR)*, pages 527–538, 2000.

Raymond Smullyan. *First-Order Logic*. Springer-Verlag, 1968.

Raymond Smullyan. *The Lady or the Tiger? and Other Logic Puzzles Including a Mathematical Novel That Features Godel's Great Discovery*. Random House Puzzles & Games, 1992.

Michael Thielscher. From situation calculus to fluent calculus: State update axioms as a solution to the inferential frame problem. *Artificial Intelligence*, 111(1-2):277–299, 1999.

Johan van Benthem. Dynamical odds and ends. Technical Report ML-1998-08, ILLC, University of Amsterdam, 1998.

Johan van Benthem. What one may come to know. *Analysis*, 64(2):95–105, 2004.

Johan van Benthem. "One is a Lonely Number": logic and communication. In Z. Chatzidakis, P. Koepke, and W. Pohlers, editors, *Logic Colloquium'02*, volume 27 of *Lecture Notes in Logic*, pages 96–129. ASL & A.K. Peters, 2006.

Johan van Benthem. Modal logic meets situation calculus. Manuscript, 2007.

Johan van Benthem, Jan van Eijck, and Barteld Kooi. Logics of communication and change. *Information and Computation*, 204(11):1620–1662, 2006.

Wiebe van der Hoek, Bernd van Linder, and John-Jules Meyer. A logic of capabilities. In Anil Nerode and Yuri Matiyasevich, editors, *Proceedings of the Third International Symposium Logical Foundations of Computer Science*, volume 813 of *Lecture Notes in Computer Science*, pages 366–378. Springer, 1994.

Wiebe van der Hoek, Bernd van Linder, and John-Jules Meyer. On agents that have the ability to chose. *Studia Logica*, 65:79–119, 2000.

Hans van Ditmarsch and Barteld Kooi. The secret of my success. *Synthese*, 151(2): 201–232, 2006a.

Hans van Ditmarsch and Barteld Kooi. The logic of ontic and epistemic change. URL `http://arxiv.org/abs/cs.LO/0610093v1`. Manuscript, 2006b.

Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. Dynamic epistemic logic with assignment. In F. Dignum, V. Dignum, S. Koenig, S. Kraus, M. Singh, and M. Wooldridge, editors, *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 141–148. ACM, 2005.

Hans van Ditmarsch, Andreas Herzig, and Tiago de Lima. Optimal regression for reasoning about knowledge and actions. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, pages 1070–1075. AAAI Press, 2007a.

Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic*, volume 337 of *Synthese Library*. Springer, 2007b.

Georg von Wright. *An Essay in Modal Logic*. North-Holland, 1951.

Richard Waldinger. Achieving several goals simutaneously. In E. Elock and D. Michi, editors, *Machine Intelligence*, volume 8, pages 94–136. Ellis Harwood, 1977.

Appendix A

Long Proofs

A.1 Observations Are General

Here we provide a detailed proof for Theorem 39 of Section 3.3. For simplicity, we sometimes use infix notation for relations in this section. For example, sT_es' means that $(s, s') \in T_e$.

DEFINITION 93
Let M be an EDLo-model, $s, t \in S$ are *epistemicaly bisimilar* (notation: $s \overset{\text{epi}}{\leftrightarrows} t$) if and only if there exists a non-empty relation $Z \subset S \times S$ such that:

1. s and t satisfy the same propositional letters of P;
2. The forth condition: if $(sZt \ \ \& \ \ sRs')$, then $\exists t'(s'Zt' \ \ \& \ \ tRt')$; and
3. The back condition: if $(sZt \ \ \& \ \ tRt')$, then $\exists s'(s'Zt' \ \ \& \ \ sRs')$.

Now, we present a theorem of (van Benthem, 2006) (It is also in (van Benthem, 1998)). It establishes that epistemicaly bisimilar worlds can be characterised by epistemic formulae.

THEOREM 94
(van Benthem, 2006) Consider any finite multi-S5 model (M, s). There exists a finite set of formulae $\delta(s_i)$ $(1 \leq i \leq k)$ such that (a) each world satisfies one of them, (b) no world satisfies two of them (i.e., they define a partition of the model), and (c) if two worlds satisfy the same formula $\delta(s_i)$, then they agree on all epistemic formulae.

Since we can characterise epistemicaly bisimilar worlds syntactically, then we have the following result.

LEMMA 95
Let M be an EDLo-model.

$$\forall s, t, t' \in S((t \in T_{!_{\gamma(s)}}(s) \ \ \& \ \ t' \in T_e(s)) \Rightarrow t \overset{\text{epi}}{\leftrightarrows} t')$$

where $\gamma(s) = \bigvee_{t \in (T_e \circ R \circ T_e^{-1})(s)} \delta(t)$

PROOF. Let M and s be given. Let $Z = \{(t, t') \in S \times S \mid \exists v, v T_{!\gamma(s)} t \ \& \ v T_e t'\}$. Now, suppose that $(t, t') \in Z$, we prove that Z is an epistemic bisimulation:

1. t and t' satisfy the same propositional letters.
   It follows from the fact that $\forall p \in P(t \in V(p) \Leftrightarrow s \in V(p))$, by Definition 38, and that $\forall p \in P(s \in V(p) \Leftrightarrow t' \in V(p))$, by Definition 35.

2. The forth condition: $(tZt' \ \& \ tRu) \Rightarrow \exists u'(uZu' \ \& \ t'Ru')$.
   suppose that $(tZt' \ \& \ tRu)$,
   then $s(T_{!\gamma(s)} \circ R)u$,
   then $s(R \circ T_{!\gamma(s)})u$, by No-forgetting,
   then $\exists v(sRv \ \& \ vT_{!\gamma(s)}u)$,
   then $T_{!\gamma(s)}(v) \neq \emptyset$,
   then $M, v \models \gamma(s)$, by condition 2 of Definition 38,
   then $\delta(v)$ is in the disjunction $\gamma(s)$, by Theorem 94,
   then $s(T_e \circ R \circ T_e^{-1})v$, by the definition of $\gamma(s)$,
   then $\exists u'(s(T_e \circ R)u' \ \& \ vT_e u')$,
   then uZu', because $vT_{!\gamma(s)}u$ and $vT_e u'$
   and also $t'Ru'$, because $sT_e t'$ and $s(T_e \circ R)u'$ and e is deterministic.

3. The back condition: $(tZt' \ \& \ t'Ru') \Rightarrow \exists u(uZu' \ \& \ tRu)$.
   suppose that $(tZt' \ \& \ t'Ru')$,
   then $s(T_e \circ R)u'$,
   then $s(R \circ T_e)u'$, by No-forgetting,
   then $\exists v(sRv \ \& \ vT_e u')$,
   then $s(T_e \circ R \circ T_e^{-1})v$,
   then $\delta(v)$ is in the disjunction $\gamma(s)$, by the definition of $\gamma(s)$,
   then $M, v \models \gamma(s)$, by Theorem 94,
   then $T_{!\gamma(s)}(v) \neq \emptyset$, by condition 2 of Definition 38,
   then $\exists u.vT_{!\gamma(s)}u$,
   then uZu', because $vT_{!\gamma(s)}u$ and $vT_e u'$
   and also $R(u) = (R \circ T_{!\gamma(s)})(v)$, by condition 4 of Definition 38,
   then tRu, because sRv and $sT_{!\gamma(s)}t$.

∎

In the sequel, we show that the action e and its corresponding complex observation ϵ are exactly the same. We start by proving it for epistemic formulae.

LEMMA 96
Let $\varphi \in \mathcal{L}_{\mathsf{EL}}$ and let $\epsilon = \bigcup_{s \in S}(?\delta(s')\,; !\gamma(s')\delta(s))$. $M, s \models [e]\varphi$ iff $M, s \models [\epsilon]\varphi$.

PROOF.

- From left to right: suppose that $M, s \models [e]\varphi$,
  iff $\forall s' \in T_e(s).M, s' \models \varphi$.

- Suppose that $T_e(s) = \emptyset$,
  then $(T_e \circ R \circ T_e^{-1})(s) = \emptyset$,
  then $\gamma(s) = \bot$,
  then not $M, s \models \gamma(s)$,
  then $T_{!\gamma(s)}(s) = \emptyset$, by condition 2 of Definition 38,
  then $\forall s' \in T_{!\gamma(s)}(s).M, s' \models \varphi$,
  then $M, s \models [!\gamma(s)]\varphi$,
  then $M, s \models \delta(s) \to [!\gamma(s)]\varphi$,
  then $M, s \models [?\delta(s') ; !\gamma(s')]\varphi$.
- Suppose that $t \in T_e(s)$,
  then $M, t \models \varphi$
  and $s \in (T_e \circ R \circ T_e^{-1})(s)$, because R is reflexive,
  then $\delta(s)$ is in the disjunction $\gamma(s)$,
  then $M, s \models \gamma(s)$,
  then $\exists t'.T_{!\gamma(s)}(s) = \{t'\}$, by condition 3 of Definition 38,
  then $t \overset{\text{epi}}{\leftrightarrows} t'$, by Lemma 95,
  then $M, t' \models \varphi$,
  then $\forall u \in T_{!\gamma(s)}(s)(M, u \models \varphi)$,
  then $M, s \models [!\gamma(s)]\varphi$,
  then $M, s \models \delta(s) \to [!\gamma(s)]\varphi$,
  then $M, s \models [?\delta(s') ; !\gamma(s')]\varphi$.

We also have that $\forall s' \overset{\text{epi}}{\not\leftrightarrows} s$(not $M, s \models \delta(s')$), by Theorem 94,
then $\forall s' \overset{\text{epi}}{\not\leftrightarrows} s(M, s \models \delta(s') \to [!\gamma(s')]\varphi)$,
then $\forall s' \overset{\text{epi}}{\not\leftrightarrows} s(M, s \models [?\delta(s') ; !\gamma(s')]\varphi)$,
then $\forall s'(M, s \models [?\delta(s') ; !\gamma(s')]\varphi)$,
then $M, s \models \bigwedge_{s' \in S}[?\delta(s') ; !\gamma(s')]\varphi$,
then $M, s \models [\bigcup_{s' \in S}(?\delta(s') ; !\gamma(s')]\varphi$.

- From right to left: suppose that $M, s \models [\bigcup_{s' \in S}(?\delta(s') ; !\gamma(s')]\varphi$,
  iff $M, s \models \bigwedge_{s' \in S}[?\delta(s') ; !\gamma(s')]\varphi$,
  iff $\forall s' \in S.M, s \models [?\delta(s') ; !\gamma(s')]\varphi$,
  iff $\forall s' \in S.M, s \models \delta(s') \to [!\gamma(s)]\varphi$,
  iff $\forall s' \in S.$ if $M, s \models \delta(s')$ then $M, s \models [!\gamma(s')]\varphi$,
  iff $\forall s' \in S.$ if $M, s \models \delta(s')$ then $\forall s'' \in T_{!\gamma(s')}(s).M, s'' \models \varphi$.

  Suppose t such that $M, s \models \delta(t)$: iff $s \overset{\text{epi}}{\leftrightarrows} t$, by Theorem 94,
  Then $\forall s' \overset{\text{epi}}{\leftrightarrows} s(\forall s'' \in T_{!\gamma(s')}(s)(M, s'' \models \varphi))$.

  - Suppose t such that $t \overset{\text{epi}}{\leftrightarrows} s$ and $T_{!\gamma(t)}(s) = \emptyset$:
    then not $M, s \models \gamma(t)$, by condition 3 of Definition 38,

then not $M, t \models \gamma(t)$, because $\gamma(t) \in \mathcal{L}_{\mathsf{EL}}$,
then $\delta(t)$ is not in the disjunction $\gamma(t)$, by Theorem 94,
then $t \notin (T_e \circ R \circ T_e^{-1})(t)$,
then $T_e(t) = \emptyset$, because R is reflexive,
then $\forall t' \in T_e(t)(M, t' \models \varphi)$,
then $M, t \models [e]\varphi$,
then $M, s \models [e]\varphi$, because $s \overset{\text{epi}}{\leftrightarrows} s$.

– Suppose t, u such that $t \overset{\text{epi}}{\leftrightarrows} s$ and $u \in T_{!\gamma(t)}(s)$:
then $M, s \models \gamma(t)$, by condition 2 of Definition 38
then $M, t \models \gamma(t)$, because $\gamma(t) \in \mathcal{L}_{\mathsf{EL}}$,
then $\delta(t)$ is in the disjunction $\gamma(t)$, by Theorem 94,
then $t \in (T_e \circ R \circ T_e^{-1})(t)$,
then $T_e(t) \neq \emptyset$,
  * Suppose v' such that $v' \in T_e(t)$:
  then $v' \overset{\text{epi}}{\leftrightarrows} v$, by Lemma 95,
  then $M, v' \models \varphi$.

  Then $\forall v' \in T_e(t)(M, v' \models \varphi)$,
  then $M, t \models [e]\varphi$,
  then $M, s \models [e]\varphi$, because $s \overset{\text{epi}}{\leftrightarrows} s$.

∎

COROLLARY 97
Let $\varphi \in \mathcal{L}_{\mathsf{EDL}}$ and let e and ϵ be as above.

$$M, s \models \varphi \text{ iff } M, s \models \varphi[\epsilon/e]$$

PROOF. Straightforward by induction on φ. The base case is Lemma 96. ∎

THEOREM 39 (OBSERVATIONS ARE GENERAL) Suppose that P and A are finite. Let $\varphi \in \mathcal{L}_{\mathsf{EDL}}$ and let e be a deterministic purely epistemic action. The formula φ is satisfiable in finite models if and only if there exists a (complex) observation ϵ such that $\varphi[\epsilon/e]$ is satisfiable in finite models.

PROOF. From right to left: since ϵ is purely epistemic, take $T_e = T_\epsilon$. From left to right: straightforward by Corollary 97. ∎

A.2 Relation Between PAL and EDL*o*

In this section, we show that '$!\varphi$' is in fact a public announcement of φ. We do this by proving that all axioms involving the public announcements in PAL are derivable in EDL*o* for the operator '$!$'.

LEMMA 98
$\models_{\mathsf{EDL}o} \langle !\varphi\rangle\psi \leftrightarrow (\varphi \wedge [!\varphi])$

PROOF. For all $\varphi, \psi \in \mathcal{L}_{\mathsf{EDL}o}$:

1.	$\bot \rightarrow \neg\psi$	PL
2.	$[!\varphi](\bot \rightarrow \neg\psi)$	by Generalisation in 1
3.	$[!\varphi]\bot \rightarrow [!\varphi]\neg\psi$	by Modus ponens in K and 2
4.	$\neg\varphi \leftrightarrow [!\varphi]\bot$	Det(!)
5.	$\neg\varphi \rightarrow [!\varphi]\neg\psi$	by Subst. of equivalents in 3 and 4
6.	$\langle !\varphi\rangle\psi \rightarrow \varphi$	by PL in 4 and 5
7.	$\langle !\varphi\rangle\psi \rightarrow [!\varphi]\psi$	Det(!)
8.	$\langle !\varphi\rangle\psi \rightarrow (\varphi \wedge [!\varphi]\psi)$	by PL in 6 and 7
9.	$\psi \rightarrow (\neg\psi \rightarrow \bot)$	PL
10.	$[!\varphi](\psi \rightarrow (\neg\psi \rightarrow \bot))$	by Generalisation in 9
11.	$[!\varphi]\psi \rightarrow [!\varphi](\neg\psi \rightarrow \bot)$	by Modus ponens in K and 10
12.	$[!\varphi]\psi \rightarrow ([!\varphi]\neg\psi \rightarrow [!\varphi]\bot)$	by Modus ponens in K and 11
13.	$[!\varphi]\psi \rightarrow (\langle !\varphi\rangle\top \rightarrow \langle !\varphi\rangle\psi)$	PL in 12
14.	$\varphi \leftrightarrow \langle !\varphi\rangle\top$	Exe(!)
15.	$[!\varphi]\psi \rightarrow (\varphi \rightarrow \langle !\varphi\rangle\psi)$	by Subst. of equivalents in 13 and 14
16.	$([!\varphi]\psi \wedge \varphi) \rightarrow \langle !\varphi\rangle\psi$	by PL in 15
17.	$\langle !\varphi\rangle\psi \leftrightarrow (\varphi \wedge [!\varphi]\psi)$	by PL in 8 and 16

∎

THEOREM 99
The following schemes are valid in EDLo.

1. $[!\varphi]p \leftrightarrow (\varphi \rightarrow p)$
2. $[!\varphi]\neg\psi \leftrightarrow (\varphi \rightarrow \neg[!\varphi]\psi)$
3. $[!\varphi](\varphi \wedge \chi) \leftrightarrow ([!\varphi]\psi \wedge [!\varphi]\chi)$
4. $[!\varphi]\mathrm{K}\psi \leftrightarrow (\varphi \rightarrow \mathrm{K}[!\varphi]\psi)$

PROOF. Scheme 2 is equivalent to Lemma 98. Scheme 3 is valid because of the axiom K. Scheme 4 corresponds to both axioms NL and NF. Below, we prove scheme 1.

For all $p \in P$ and all $\varphi \in \mathcal{L}_{\mathsf{EDL}o}$:

1.	$\neg p \to [!\varphi]\neg p$	Pre(!)
2.	$(\neg p \wedge \varphi) \to ([!\varphi]\neg p \wedge \varphi)$	by PL in 1
3.	$([!\varphi]\neg p \wedge \varphi) \leftrightarrow \langle !\varphi\rangle \neg p$	Lemma 98
4.	$(\neg p \wedge \varphi) \to \langle !\varphi\rangle \neg p$	by Subst. of equivalents in 2 and 3
5.	$[!\varphi]p \to (\varphi \to p)$	by PL in 4
6.	$\bot \to p$	PL
7.	$[!\varphi](\bot \to p)$	by Generalisation in 6
8.	$[!\varphi](\bot \to p) \to ([!\varphi]\bot \to [!\varphi]p)$	K(!)
9.	$[!\varphi]\bot \to [!\varphi]p$	by Modus ponens in 7 and 8
10.	$\neg\varphi \leftrightarrow [!\varphi]\bot$	Exe(!)
11.	$\neg\varphi \to [!\varphi]p$	by Subst. of equivalents in 9 and 10
12.	$p \to [!\varphi]p$	Pre(!)
13.	$(\varphi \to p) \to [!\varphi]p$	by PL in 11 and 12
14.	$([!\varphi]p \leftrightarrow (\varphi \to p)$	by PL in 5 and 13

∎

And finally, we prove that the axioms of EDL*o* are sound in PAL.

THEOREM 100
The following schemes are valid in PAL.

1. $\psi \to [\varphi]\psi \qquad (\text{for } \varphi \in \mathcal{L}_{\mathsf{PL}})$
2. $\varphi \leftrightarrow \langle\varphi\rangle\top$
3. $\langle\varphi\rangle\psi \to [\varphi]\psi$
4. $[\varphi]\mathrm{K}\psi \to ([\varphi]\bot \vee \mathrm{K}[\varphi]\psi)$

PROOF. Scheme 1 is valid because $p \to [\varphi]p$ is valid. Schemes 2 and 3 are straightforward by the semantics of '!' in PAL. Scheme 4 is valid because it follows from scheme 2 and the axiom $[\varphi]\mathrm{K}\psi \leftrightarrow (\varphi \to \mathrm{K}[\varphi]\psi)$ of the proof system of PAL. ∎

A.3 Polynomial Translation

LEMMA 48 Let D be a finite Reiter-style action description and let $\varphi \in \mathcal{L}_D$. Then $\mathrm{len}(\delta_D(\varphi)) \leq \mathcal{O}(\mathrm{len}(D) \times \mathrm{len}(\varphi))$.

PROOF. First, we recall that each $\mathit{eff}^+(a)$, $\mathit{eff}^-(a)$, $\gamma^+(a)$ and $\gamma^-(a)$ are finite. In addition, $\mathrm{len}(D) = 6 + \mathrm{len}(\mathit{poss}) + \mathrm{len}(\mathit{eff}^+) + \mathrm{len}(\mathit{eff}^-) + \mathrm{len}(\gamma^+) + \mathrm{len}(\gamma^-)$, and $\mathrm{len}(\gamma^+(a)) = 1 + \sum_{p \in \mathit{eff}^+(a)} (1 + \mathrm{len}(\gamma^+(a,p)))$, and analogously for $\mathrm{len}(\gamma^-(a))$. We prove by induction on the length of φ that $\mathrm{len}(\delta_D(\varphi)) \leq 8 \times \mathrm{len}(D) \times \mathrm{len}(\varphi)$:

Induction base: $\mathrm{len}(\varphi) = 1$; hence $\varphi = p$ for some atomic p.
Then $\mathrm{len}(\delta_D(p)) = \mathrm{len}(p) = 1 \leq 8 \times \mathrm{len}(D)$.
Induction hypothesis: if $\mathrm{len}(\varphi) \leq n$, then $\mathrm{len}(\delta_D(\varphi)) \leq 8 \times \mathrm{len}(D) \times \mathrm{len}(\varphi)$.
Induction step:

1. $\mathrm{len}(\delta_D(\neg\varphi))$
   $= \mathrm{len}(\neg\delta_D(\varphi))$
   $= 1 + \mathrm{len}(\delta_D(\varphi))$
   $< 8 \times \mathrm{len}(D) + \mathrm{len}(\delta_D(\varphi))$ (because $1 < 8 \times \mathrm{len}(D)$)
   $\leq 8 \times \mathrm{len}(D) + 8 \times \mathrm{len}(D) \times \mathrm{len}(\varphi)$ (by induction hypothesis)
   $= 8 \times \mathrm{len}(D) \times (1 + \mathrm{len}(\varphi))$
   $= 8 \times \mathrm{len}(D) \times \mathrm{len}(\neg\varphi)$
2. Analogously to point 1, we prove that $\mathrm{len}(\delta_D(\varphi \wedge \psi)) \leq 8 \times \mathrm{len}(D) \times \mathrm{len}(\varphi \wedge \psi)$.
3. Analogously to point 1, we prove that $\mathrm{len}(\delta_D(\mathrm{K}_i\varphi)) \leq 8 \times \mathrm{len}(D) \times \mathrm{len}(\mathrm{K}_i\varphi)$.
4. $\mathrm{len}(\delta_D([a]\varphi)) = \mathrm{len}([\mathit{poss}(a)][\sigma_a]\delta_D(\varphi))$ (by Definition 47)
   $= 2 + \mathrm{len}(\mathit{poss}(a)) + \mathrm{len}(\sigma_a) + \mathrm{len}(\varphi)$

   We have that

$$\begin{aligned}
\mathrm{len}(\sigma_a) = & \sum_{p \in \mathit{eff}^+(a)\ \&\ p \notin \mathit{eff}^-(a)} 7 + \mathrm{len}(\gamma^+(a,p)) \quad + \\
& \sum_{p \in \mathit{eff}^-(a)\ \&\ p \notin \mathit{eff}^+(a)} 8 + \mathrm{len}(\gamma^-(a,p)) \quad + \\
& \sum_{p \in \mathit{eff}^+(a) \cap \mathit{eff}^-(a)} 9 + \mathrm{len}(\gamma^+(a,p)) + \mathrm{len}(\gamma^-(a,p)) \\
< & \sum_{p \in \mathit{eff}^+(a)\ \&\ p \notin \mathit{eff}^-(a)} 7 + 1 + \mathrm{len}(\gamma^+(a,p)) \quad + \\
& \sum_{p \in \mathit{eff}^-(a)\ \&\ p \notin \mathit{eff}^+(a)} 7 + 1 + \mathrm{len}(\gamma^-(a,p)) \quad + \\
& \sum_{p \in \mathit{eff}^+(a) \cap \mathit{eff}^-(a)} 7 + (1 + \mathrm{len}(\gamma^+(a,p))) + (1 + \mathrm{len}(\gamma^-(a,p))) \\
= & \sum_{p \in \mathit{eff}^+(a)} 7 \quad + \sum_{p \in \mathit{eff}^-(a)} 7 \quad + \\
& \sum_{p \in \mathit{eff}^+(a)} 1 + \mathrm{len}(\gamma^+(a,p)) \quad + \\
& \sum_{p \in \mathit{eff}^-(a)} 1 + \mathrm{len}(\gamma^-(a,p))
\end{aligned}$$

$$\leq 7 \times \mathrm{len}(\mathit{eff}^+(a)) + 7 \times \mathrm{len}(\mathit{eff}^-(a)) + \mathrm{len}(\gamma^+(a)) + \mathrm{len}(\gamma^-(a))$$

Then

$$\begin{aligned}
&\mathrm{len}(\delta_D([a]\varphi)) \\
&\leq 2 + 7 \times \mathrm{len}(D) + \mathrm{len}(\delta_D(\varphi)) \\
&< 8 \times \mathrm{len}(D) + \mathrm{len}(\delta_D(\varphi)) && \text{(because } 2 < \mathrm{len}(D)\text{)} \\
&\leq 8 \times \mathrm{len}(D) + 8 \times \mathrm{len}(D) \times \mathrm{len}(\varphi) && \text{(by induction hypothesis)} \\
&= 8 \times \mathrm{len}(D) \times (1 + \mathrm{len}(\varphi)) \\
&\leq 8 \times \mathrm{len}(D) \times (2 + \mathrm{len}(\varphi)) \\
&= 8 \times \mathrm{len}(D) \times (\mathrm{len}([a]\varphi))
\end{aligned}$$

∎

A.4 From D-models to DEL-models

THEOREM 49 Let D be a Reiter-style action description and let $\varphi \in \mathcal{L}_D$. Then φ is D-satisfiable if and only if $\delta_D(\varphi)$ is DELC-satisfiable.

PROOF. Let an action description D be given, and let A be the set of actions described in D. For each epistemic model N and each $a \in A$, we define the structure $N^a = \langle S^a, R^a, V^a \rangle$ such that

$$\begin{aligned}
s^a \in S^a \quad &\text{iff} \quad N, s \models \mathit{poss}(a) \\
(s_1^a, s_2^a) \in R_i^a \quad &\text{iff} \quad s_1^a, s_2^a \in S^a \text{ and } (s_1, s_2) \in R_i \\
s^a \in V_p^a \quad &\text{iff} \quad s^a \in S^a \text{ and } N, s \models \gamma^+(a,p) \vee (\neg\gamma^+(a,p) \wedge p)
\end{aligned}$$

We define the set $\mathrm{seq}(A)$ as the set of all action sequences of A by the following BNF:

$$\tau ::= \epsilon \mid \tau a$$

where ϵ denotes the empty sequence and a ranges over A. We also extend the definition of $\mathrm{len}(\cdot)$ to sequences by stipulating $\mathrm{len}(\epsilon) = 0$ and $\mathrm{len}(\tau a) = 1 + \mathrm{len}(\tau)$.

Now, let $N^\epsilon = N$ and let $N^{\tau a} = (N^\tau)^a$. We define the structure:

$$N^D = \langle S^D, R^D, T^D, V^D \rangle$$

such that

$$\begin{aligned} S^D &= \bigcup_{\tau \in \mathrm{seq}(A)} S^\tau \\ R_i^D &= \bigcup_{\tau \in \mathrm{seq}(A)} R_i^\tau \\ V_p^D &= \bigcup_{\tau \in \mathrm{seq}(A)} V_p^\tau \\ T_a^D &= \bigcup_{\tau \in \mathrm{seq}(A)} \{(s^\tau, s^{\tau a}) \mid a \in A \text{ and } s^\tau \in S^\tau \text{ and } s^{\tau a} \in S^{\tau a}\} \end{aligned}$$

LEMMA 101
N^D is a D-model.

PROOF. We must show that N^D satisfies all the five constraints of the definition of D-models in page 45. We do only the first two cases here; the other ones are easier and left to the reader.

No-forgetting.
Suppose that $t \in (T_a^D \circ R_i^D)(s)$.
Then there exists $s' \in T_a^D(s)$ such that $t \in R_i^D(s')$.
Then there exists $\tau \in \mathrm{seq}(A)$ such that $s \in S^\tau$ and $s', t \in S^{\tau a}$
and there also exists $t' \in S^\tau$ such that $t \in T_a^D(t')$
(by the definition of N^D).
then $t \in R_i^{\tau a}(s')$ iff $t' \in R_i^\tau(s)$
(by the definition of N^D).

No-learning.
Suppose that $T_a^D(s) \neq \emptyset$ and $t \in (R_i^D \circ T_a^D)(s)$.
Then there exists $s' \in T_a^D(s)$
and there also exists $t' \in R_i^D(s)$ such that $t \in T_a^D(t')$.
Then there exists $\tau \in \mathrm{seq}(A)$ such that $s, t' \in S^\tau$ and $s', t \in S^{\tau a}$
(by the definition of N^D).
Then $t \in R_i^{\tau a}(s')$ iff $t' \in R_i^\tau(s)$.
(by the definition of N^D). ∎

LEMMA 102
Let $M = \langle S, R, T, V \rangle$ be a D-model. Let $N = \langle S, R, V \rangle$ be the epistemic model generated from M. Then (N^D, s) is bisimilar to (M, s) for every $s \in S$.

PROOF. We define the relation $Z \subseteq S^D \times S$ as the union $\bigcup_{\tau \in \mathrm{seq}(A)} Z^\tau$ where:

$$\begin{aligned} s \in Z^\epsilon(t^\epsilon) \quad &\text{iff} \quad t = s \\ s \in Z^{\tau a}(t^{\tau a}) \quad &\text{iff} \quad \text{there are } s' \text{ and } t^\tau \text{ such that} \\ &\qquad s \in T_a(s') \text{ and } t^{\tau a} \in T_a^D(t^\tau) \text{ and } s' \in Z^\tau(t^\tau) \end{aligned}$$

We first show that each Z^τ is an *epistemic bisimulation* between N^τ and M. That is, that the following holds.

if $s \in Z^\tau(t^\tau)$, then:

1. $t^\tau \in V_p^\tau$ iff $s \in V_p$;
2. if $t' \in R_i^\tau(t^\tau)$, then there exists s' such that $s' \in R_i(s)$ and $s' \in Z^\tau(t')$; and
3. if $s' \in R_i(s)$, then there exists t' such that $t' \in R_i^\tau(t^\tau)$ and $s' \in Z^\tau(t')$.

The proof is by induction on $\mathrm{len}(\tau)$.

Induction base.
Suppose that $\mathrm{len}(\tau) = 0$.
Then $s \in Z^\epsilon(t^\epsilon)$
iff $s = t$, from what the three cases follow.

Induction hypothesis.
If $\mathrm{len}(\tau) \leq n$, then $s \in Z^\tau(t^\tau)$ implies that Z^τ is an epistemic bisimulation between (N^τ, t^τ) and (M, s).

Note that the induction hypothesis implies that if $\varphi \in \mathcal{L}_{\mathsf{ELC}}$, then $(N^\tau, s^\tau) \models \varphi$ iff $(M, s) \models \varphi$.

Induction step.
Suppose that $\tau a \in \mathrm{seq}(A)$, and $\mathrm{len}(\tau) = n$, and $s \in Z^{\tau a}(t^{\tau a})$.

1. Suppose that $t^{\tau a} \in V_p^{\tau a}$
   iff there are s' and t^τ such that $s \in T_a(s')$ and $t^{\tau a} \in T_a^D(t^\tau)$ and $s' \in Z^\tau(t^\tau)$
   (by the definition of $Z^{\tau a}$),
   and $t^{\tau a} \in S^{\tau a}$ and $(N^\tau, t^\tau) \models \gamma^+(a, p) \vee (\neg\gamma^+(a, p) \wedge p)$
   (by the definition of $N^{\tau a}$)
   iff $(M, s') \models \gamma^+(a, p) \vee (\neg\gamma^+(a, p) \wedge p)$
   (by the induction hypothesis)
   iff $s \in V_p$.

2. Suppose that $u \in R_i^{\tau a}(t^{\tau a})$
   iff there are s' and t^τ such that $s \in T_a(s')$ and $t^{\tau a} \in T_a^D(t^\tau)$ and $s' \in Z^\tau(t^\tau)$
   (by the definition of $Z^{\tau a}$).
   Then there exists u' such that $u' \in R_i^\tau(t^\tau)$ and $u \in T_a^D(u')$
   (because N^D respects No-forgetting).
   Then there exists v' such that $v' \in R_a gent(s')$ and $v' \in Z^\tau(u')$
   (by the induction hypothesis)
   and $(N^\tau, u') \models poss(a)$
   (because $T(u') \neq \emptyset$ and N^D respects Determinism).
   Then $(M, v') \models poss(a)$
   (by the induction hypothesis)
   iff there exists v such that $v \in T_a(v')$
   (because M respects Executability).

Then $v \in R_i(s)$
(because M respects No-learning)
and $u \in Z^{\tau a}(t^{\tau a})$
(by the definition of $Z^{\tau a}$).

3. Suppose that $v \in R_i(t)$.
Analogously to the previous point, we show that there exists u such that $u \in R_i^{\tau a}(t^{\tau i})$ and $v \in Z^{\tau a}(u)$.

Now, note that because Z contains Z_τ for all $\tau \in \mathrm{seq}(A)$, Z is an epistemic bisimulation between N^D and M. As a consequence, we have that if $s \in Z(t)$ and $\varphi \in \mathcal{L}_{\mathsf{ELC}}$, then $N^D s \models \varphi$ iff $M, t \models \varphi$.

We now do one more step and show that Z is a full bisimulation between N^D and M.

Suppose that $s \in Z(t)$.
Then there exists $\tau \in \mathrm{seq}(A)$ such that $t = t^\tau$ and $s \in Z^\tau(t^\tau)$
(by the definition of Z).

1. Suppose that $t' \in T_a^D(t^\tau)$
iff $t' = t^{\tau a} \in S^{\tau a}$
(by the definition of T_a^D)
iff $N^\tau, t^\tau \models \mathit{poss}(a)$
(by the definition of $S^{\tau a}$)
iff $M, s \models \mathit{poss}(a)$
(because Z^τ is an epistemic bisimulation between (N^τ, t^τ) and (M, s))
iff there exists s' such that $s' \in T_a(s)$
(because M respects Executability)
iff $s' \in Z^{\tau a}(t^{\tau a})$
(by the definition of $Z^{\tau a}$).
Then $s' \in Z(t^{\tau a})$
(by the definition of Z).

2. Suppose that $s' \in T_a(s)$
iff $M, s \models \mathit{poss}(a)$
(because M respects Executability)
iff $N^\tau, t^\tau \models \mathit{poss}(a)$
(because Z^τ is an epistemic bisimulation between (N^τ, t^τ) and (M, s))
iff there exists $t^{\tau a}$ such that $t^{\tau a} \in S^{\tau a}$
(by the definition of $S^{\tau a}$)
iff $t^{\tau a} \in T_a^D(t^\tau)$
(by the definition of T_a^D)
iff $s' \in Z^{\tau a}(t^{\tau a})$
(by the definition of $Z^{\tau a}$).
Then $s' \in Z(t^{\tau a})$
(by the definition of Z).

∎

LEMMA 103
Let N be an epistemic model. Then $(N, s) \models \delta_D(\varphi)$ if and only if $(N^D, s) \models \varphi$

PROOF. In this proof we use the fact that if $(N, s) \models poss(a)$, then $(N^a, s^a) \leftrightarrows ((N^{poss(a)})^{\sigma_a}, s)$, where '$\leftrightarrows$' denotes the bissimilarity relation defined in (Blackburn et al., 2001a).

The proof is by induction on the number of connectives of φ.

Induction base. Suppose that $\varphi = p$ for some $p \in P$.
Then $\delta_D(\varphi) = p$, and we obviouslly have that $(N, s) \models p$ iff $(N^D, s) \models p$.

Induction hypothesis. If the number of connectives of φ is less than n, then $(N, s) \models \delta_D(\varphi)$ iff $(N^D, s) \models \varphi$.

Induction step. We consider four cases:

1. Suppose that $\varphi = \neg\psi$ and $N, s \models \delta_D(\neg\psi)$
   iff $N, s \models \neg\delta_D(\psi)$
   iff $N, s \not\models \delta_D(\psi)$
   iff $N^D, s \not\models \psi$ (by the induction hypothesis)
   iff $N^D, s \models \neg\psi$.

2. $N, s \models \delta_D(\psi \wedge \chi)$
   iff $N, s \models \delta_D(\psi) \wedge \delta_D(\chi)$
   iff $N, s \models \delta_D(\psi)$ and $N, s \models \delta_D(\chi)$
   iff $N^D, s \models \psi$ and $N^D, s \models \chi$ (by the induction hypothesis)
   iff $N^D, s \models \psi \wedge \chi$.

3. $N, s \models \delta_D(\mathrm{K}_i\psi)$
   iff $N, s \models \mathrm{K}_i\delta_D(\psi)$
   iff for all $s' \in R_i(s)$, $N, s' \models \delta_D(\psi)$
   iff for all $s' \in R_i^D(s)$, $N, s' \models \delta_D(\psi)$
   (because $R_i(s) = R_i^\epsilon(s) = R_i^D(s)$ — by the definition of R_i^D)
   iff for all $s' \in R_i^D(s)$, $N^D, s' \models \psi$
   (by the induction hypothesis)
   Iff $N^D, s \models \mathrm{K}_i\psi$.

4. $N, s \models \delta_D([a]\psi)$
   iff $N, s \models [poss(a)][\sigma_a]\delta_D(\psi)$
   iff $N, s \models poss(a)$ implies $N^{poss(a)}, s \models [\sigma_a]\delta_D(\psi)$
   iff $N, s \models poss(a)$ implies $(N^{poss(a)})^{\sigma_a}, s \models \delta_D(\psi)$
   iff $N^D, s \models poss(a)$ implies $(N^{poss(a)})^{\sigma_a}, s \models \delta_D(\psi)$
   (because $poss(a) \in \mathcal{L}_{\mathsf{ELC}}$ and by Lemma 102)
   iff $N^D, s \models poss(a)$ implies $N^a, s^a \models \delta_D(\psi)$
   (because $(N^{poss(a)})^{\sigma_a}, s \leftrightarrows N^a, s^a$)
   iff $N^D, s \models poss(a)$ implies $(N^a)^D, s^a \models \psi$
   (by the induction hypothesis)
   iff $N^D, s \models poss(a)$ implies $N^D, s^a \models \psi$

(by the fact that $(N^a)^D$ is a sub-tree of N^D)
iff $N^D, s \models [a]\psi$.
(by the fact that $T_a^D(s) = \{s^a\}$).

∎

Now, the main theorem follows straightforwardly:

From the left to the right. Suppose that M is a D-model such that $(M, s) \models \varphi$
iff $(N^D, s^\epsilon) \models \varphi$ (by Lemma 102)
iff $(N, s^\epsilon) \models \delta_D(\varphi)$ (by Lemma 103).

From the right to the left.
Suppose that N is an epistemic model such that $(N, s) \models \delta(\varphi)$.
Then M^D is a D-model (by Lemma 101)
and $(M^D, s^\epsilon) \models \varphi$ (by Lemma 103). ∎

A.5 Soundness and Completeness of the Tableau Method

Proof of Theorem 91 is an extension of proof of Theorem 59. We thus put both together here.

THEOREM 59 AND 91 (SOUNDNESS AND COMPLETENESS)
For $\mathsf{C} \in \{\mathsf{K}, \mathsf{KT}, \mathsf{S4}, \mathsf{S5}\}$, there is a closed C-APAL-tableau for $\neg\varphi$ if and only if φ is C-APAL-valid.

PROOF. From the left to the right. We prove that if φ is satisfiable, then there is no closed tableau for φ. We do this by showing that all tableau rules preserve satisfiability. In other words, let T^i be a tableau for a given formula that contains a branch $b = (\Lambda, \Sigma)$, we show that if b is satisfiable, then the set of branches B generated by any tableau rule has also at least one satisfiable branch. We therefore need to define what is meant by satisfiability of branches.

DEFINITION 104
The branch b is *satisfiable* if and only if there exists an epistemic structure $M = \langle S, R, V\rangle$ and a function f from $\mathbb{N}$ to S such that for all $(i, x, x') \in \Sigma$, $f(x)R_i f(x')$ and for all $(\psi^k, x, \varphi) \in \Lambda$:

$$M|\psi^0, f(x) \models \psi_1,\ M|\psi^1, f(x) \models \psi_2, \ldots, M|\psi^{k-1}, f(x) \models \psi_k,\ M|\psi^k, f(x) \models \varphi$$

Suppose that the branch $b = (\Lambda, \Sigma)$ is satisfiable. The proofs for rules $\neg$, $\wedge$ and $\vee$ are straightforward and left to the reader. We then prove that rule K is sound. If $(\psi^k, x, \mathrm{K}_i\varphi) \in \Lambda$ and $(i, x, x') \in \Sigma$, then the application of rule K generates all the branches $b_j = (\Lambda_j, \Sigma)$ for $1 \leq j \leq k + 1$ where

$$
\begin{array}{lll}
\Lambda_1 & = \Lambda \cup & \{(\psi^0, x', \neg\psi_1)\} \\
\Lambda_2 & = \Lambda \cup & \{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\} \\
\Lambda_3 & = \Lambda \cup & \{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3)\} \\
 & \vdots & \\
\Lambda_k & = \Lambda \cup & \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-2}, x', \psi_{k-1}), (\psi^{k-1}, x', \neg\psi_k)\} \\
\Lambda_{k+1} & = \Lambda \cup & \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \varphi)\}.
\end{array}
$$

By hypothesis, there exists an epistemic structure $M = \langle S, R, V\rangle$ and a function f from $\mathbb{N}$ to S such that for all $(i, x, x') \in \Sigma$, $f(x)R_i f(x')$ and $M|\psi^0, f(x) \models \psi_1$, …, $M|\psi^{k-1}, f(x) \models \psi_k$ and $M|\psi^k, f(x) \models \mathrm{K}_i\varphi$. Then, $M|\psi^k, f(x') \models \varphi$. Then, one of the following conditions holds:

$M|\psi^0, f(x') \models \neg\psi_1$ or
$M|\psi^0, f(x') \models \varphi_1, M|\psi^1, f(x') \models \neg\psi_2$ or
$M|\psi^0, f(x') \models \varphi_1, M|\psi^1, f(x') \models \psi_2, M|\psi^2, f(x') \models \neg\psi_3$ or
$\vdots$
$M|\psi^0, f(x') \models \varphi_1, \ldots, M|\psi^{k-2}, f(x') \models \psi_{k-1}, M|\psi^{k-1}, f(x') \models \neg\psi_k$ or
$M|\psi^0, f(x') \models \varphi_1, \ldots, M|\psi^{k-1}, f(x') \models \psi_k, M|\psi^k, f(x') \models \varphi$.

Therefore one of the branches b_j is satisfiable.

Rules T, 4 and 5 are proved to be sound in a similar way. In these cases we also use the fact that R_i is respectively reflexive, transitive and symmetric. We omit the details here.

For rule $\widehat{\mathrm{K}}$ suppose that $(\psi^k, x, \neg\mathrm{K}_i\varphi) \in \Lambda$. Then the application of rule $\widehat{\mathrm{K}}$ generates only one branch $b_1 = (\Lambda_1, \Sigma_1)$ such that $\Lambda_1 = \Lambda \cup \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \neg\varphi)\}$ and $\Sigma_1 = \Sigma \cup \{(i, x, x')\}$ for some x' that does not occur in Λ. By hypothesis, there exists an epistemic structure $M = \langle S, R, V\rangle$ and a function f from $\mathbb{N}$ to S such that $M|\psi^0, f(x) \models \psi_1$, …, $M|\psi^{k-1}, f(x) \models \psi_k$ and $M|\psi^k, f(x) \models \neg\mathrm{K}_i\varphi$. Then, there exists $s \in S^{\psi^k}$ such that $f(x)R_i^{\psi^k}s$ and $M|\psi^0, s \models \psi_1$, …, $M|\psi^k, w \models \psi_k$ and $M|\psi^k, s \models \neg\varphi$. We thus consider the function $f' : \mathbb{N} \to S$ such that for all x that occur in Λ, $f'(x) := f(x)$ and $f'(x') := s$. Therefore, b_1 is satisfiable.

For rule $\mathrm{R}[\cdot]$ suppose that $(\psi^k, x, [\varphi_1]\varphi_2) \in \Lambda$. Then the application of the rule $\mathrm{R}[\cdot]$ generates the branches $b_1 = (\Lambda \cup \{(\psi^k, x, \neg\varphi_1)\}, \Sigma)$ and $b_2 = (\Lambda \cup \{(\psi^k, x, \varphi_1), (\psi^k\varphi_1, x, \varphi_2)\}, \Sigma)$. Seeing that $M|\psi^k, f(x) \models [\varphi_1]\varphi_2$ iff either $M|\psi^k, f(x) \models \neg\varphi_1$ or $M|\psi^k, f(x) \models \varphi_1$ and $M|\psi^k|\varphi_1, f(x) \models \varphi_2$, thus b_1 is satisfiable or b_2 is satisfiable.

For rule $\mathrm{R}\langle\cdot\rangle$ suppose that $(\psi^k, x, \neg[\varphi_1]\varphi_2) \in \Lambda$. Then the application of rule $\mathrm{R}\langle\cdot\rangle$ generates only one branch $b_1 = (\Lambda \cup \{(\psi^k, x, \varphi_1), (\psi^k, \varphi_1, x, \neg\varphi_2)\}, \Sigma)$. Seeing that $M|\psi^k, f(x) \models \neg[\varphi_1]\varphi_2$ iff $M|\psi^k, f(x) \models \varphi_1$ and $M|\psi^k|\varphi_1, f(x) \models \neg\varphi_2$, thus b_1 is satisfiable.

For rule $\Box$ suppose that $(\psi^k, x, \Box\varphi) \in \Lambda$. Then B contains only one branch $b_1 = \langle\Lambda \cup \{(\psi^k, x, [\chi]\varphi)\}, \Sigma\rangle$ for some $\chi \in \mathcal{L}_{\mathsf{PAL}}$. Seeing that $M|\psi^k, f(x) \models \Box\varphi$ iff $M|\psi^k, f(x) \models [\chi]\varphi$ for all $\chi \in \mathcal{L}_{\mathsf{PAL}}$, thus b_1 is satisfiable.

For rule $\Diamond$ suppose that $(\psi^k, x, \neg\Box\varphi) \in \Lambda$. Then B contains only one branch $b_1 = \langle\Lambda \cup \{(\psi^k, x, \neg[p]\varphi)\}, \Sigma\rangle$ for some $p \in P$ that does not occur in Λ. Since $M|\psi^k, f(n) \models \neg\Box\varphi$, then there exists a formula $\chi \in \mathcal{L}_{\mathsf{PAL}}$ such that $M|\psi^k, f(n) \models \neg[\chi]\varphi$. Let $M' = \langle S, R, V'\rangle$ be the epistemic structure defined as follows.

- $V'(p') := V(p')$ if p' is different from p; and
- $V'(p) := \{s \in S : M|\psi^k, s \models \chi\}$.

Now we have that $M|\psi^k, f(n) \models \neg[p]\varphi$. Therefore, b_1 is satisfiable.

It finishes the first part of the proof.

From the right to the left. We do it for S5-PAL only. The other cases are similar and left to the reader. We first need the following lemma.

LEMMA 105
Let (Λ, Σ) be a branch of a tableau T^i. if $(\psi^k, x, \varphi) \in \Lambda$, then $(\psi^{j-1}, x, \psi_j) \in \Lambda$ for all $j \leq k$.

We show Lemma 105 by induction on i. The base case is $i = 0$. This follows immediately by the definition of initial tableau. Now consider the branch b in the tableau T^{i+1}. If it contains a formula of the form (ψ^k, x, φ). Then either it is in a branch of T^i, and in this case the induction hypothesis applies, or it is added by an application of one of the tableau rules in T^i. If it is added by the application of rules $\neg$, $\wedge$, $\vee$ and T, then T^i contains a formula of the form (ψ^k, x', φ') and the induction hypothesis applies. Now, for rules K, $\widehat{\mathrm{K}}$, 4 and 5, note that whenever a new formula of the form (ψ^k, x, φ) is added to the branch b, then all the formulas (ψ^{j-1}, x, ψ_j) are also added to the same branch. Finally, for the rules $\mathrm{R}[\cdot]$ and $\mathrm{R}\langle\cdot\rangle$, note that whenever a new labelled formula of the form (ψ^k, x, φ) is added, then we have one of two cases: (i) a formula of the form (ψ^k, x, φ) is present in T^i or (ii) a formula of the form (ψ^{k-1}, x, φ) is present in T^i and the formula (ψ^{k-1}, x, ψ_k) is added to T^{i+1}. In both cases, induction hypothesis applies. It finishes the demonstration of Lemma 105.

The proof continues by induction on the structure of labelled formulas in the tableau. We first need more three definitions.

DEFINITION 106
The *length* of labelled formulas is recursively defined as follows:

$$\begin{array}{lcl}
\mathrm{len}(p) & = & 1 \\
\mathrm{len}(\neg\varphi) & = & 1 + \mathrm{len}(\varphi) \\
\mathrm{len}(\varphi_1 \wedge \varphi_2) & = & 1 + \mathrm{len}(\varphi_1) + \mathrm{len}(\varphi_2) \\
\mathrm{len}(\mathrm{K}_i\varphi) & = & 1 + \mathrm{len}(\varphi) \\
\mathrm{len}([\varphi_1]\varphi_2) & = & 1 + \mathrm{len}(\varphi_1) + \mathrm{len}(\varphi_2) \\
\mathrm{len}(\Box\varphi) & = & 2 + \mathrm{len}(\varphi) \\
\mathrm{len}(\psi^k) & = & \mathrm{len}(\psi_1) + \cdots + \mathrm{len}(\psi_k) \\
\mathrm{len}(\psi^k, x, \varphi) & = & 1 + \mathrm{len}(\psi^k) + \mathrm{len}(\varphi)
\end{array}$$

DEFINITION 107
The *size* $\mathrm{size}(\lambda)$ of a labelled formula $\lambda = (\psi^k, x, \varphi)$ is defined as a pair $(\mathrm{wei}(\lambda), \mathrm{len}(\lambda))$ where $\mathrm{wei}(\lambda)$ is the *weight* of λ defined as the number of occurrences of the operator $\Box$ in ψ^k and φ. In addition, we define that $\mathrm{size}(\lambda_1) < \mathrm{size}(\lambda_2)$ if and only if either $\mathrm{wei}(\lambda_1) < \mathrm{wei}(\lambda_2)$ or both $\mathrm{wei}(\lambda_1) = \mathrm{wei}(\lambda_2)$ and $\mathrm{len}(\lambda_1) < \mathrm{len}(\lambda_2)$.

It is clear that the relation '$<$' for sizes is a well-founded order.

DEFINITION 108
Let T be a tableau.

1. T is *saturated under rule* $\neg$ if and only if for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, \neg\neg\varphi) \in \Lambda$, then $(\psi^k, x, \varphi) \in \Lambda$.

2. T is *saturated under rule* $\wedge$ if and only if for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, \varphi_1 \wedge \varphi_2) \in \Lambda$, then $(\psi^k, x, \varphi_1) \in \Lambda$ and $(\psi^k, x, \varphi_2) \in \Lambda$.

3. T is *saturated under rule* $\vee$ if and only if for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, \neg(\varphi_1 \wedge \varphi_2)) \in \Lambda$, then $(\psi^k, x, \neg\varphi_1) \in \Lambda$ or $(\psi^k, x, \neg\varphi_2) \in \Lambda$.

4. T is *saturated under rule* K if and only if for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, \mathrm{K}_i\varphi) \in \Lambda$ and $(i, x, x') \in \Sigma$, then
   $\{(\psi^0, x', \neg\psi_1)\} \subseteq \Lambda$ or
   $\{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\} \subseteq \Lambda$ or
   $\{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3\} \subseteq \Lambda$ or
   $\vdots$
   $\{(\psi^0, x', \psi_1), \ldots, (\psi^{k-2}, x', \neg\psi_{k-1}), (\psi^{k-1}, x', \psi_k)\} \subseteq \Lambda$ or
   $\{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \varphi\} \subseteq \Lambda$

5. T is *saturated under rule* $\widehat{\mathrm{K}}$ if and only if for all $b = (\Lambda, \Sigma) \in T$, if $(psi^k, x, \neg\mathrm{K}_i\varphi \in \Lambda$, then $\{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \neg\varphi)\} \subseteq \Lambda$ and $(i, x, x') \in \Sigma$.

6. T is *saturated under rule T* if and only if for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, \mathrm{K}_i\varphi) \in \Lambda$, then $(\psi^k, x, \varphi) \in \Lambda$.

7. T is *saturated under rule 4* if and only if for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, \mathrm{K}_i\varphi) \in \Lambda$ and $(i, x, x') \in \Sigma$, then
   $\{(\psi^0, x', \neg\psi_1)\} \subseteq \Lambda$ or
   $\{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\} \subseteq \Lambda$ or
   $\{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3\} \subseteq \Lambda$ or
   $\vdots$
   $\{(\psi^0, x', \psi_1), \ldots, (\psi^{k-2}, x', \neg\psi_{k-1}), (\psi^{k-1}, x', \psi_k)\} \subseteq \Lambda$ or
   $\{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \mathrm{K}_i\varphi\} \subseteq \Lambda$

8. T is *saturated under rule 5* if and only if for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, \mathrm{K}_i\varphi) \in \Lambda$ and $(i, x', x) \in \Sigma$, then
   $\{(\psi^0, x', \neg\psi_1)\} \subseteq \Lambda$ or
   $\{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\} \subseteq \Lambda$ or
   $\{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3\} \subseteq \Lambda$ or
   $\vdots$
   $\{(\psi^0, x', \psi_1), \ldots, (\psi^{k-2}, x', \neg\psi_{k-1}), (\psi^{k-1}, x', \psi_k)\} \subseteq \Lambda$ or
   $\{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \mathrm{K}_i\varphi\} \subseteq \Lambda$

9. T is *saturated under rule* $\mathrm{R}[\cdot]$ if and only if:
for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, [\varphi_1]\varphi_2) \in \Lambda$, then either $\{(\psi^k, x, \neg\varphi_1)\} \subseteq \Lambda$ or $\{(\psi^k, x, \varphi_1), ((\psi^k, \varphi_1), x, \varphi_2)\} \subseteq \Lambda$.

10. T is *saturated under rule* $\mathrm{R}\langle\cdot\rangle$ if and only if:
for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, \langle\varphi_1\rangle\varphi_2) \in \Lambda$, then $\{(\psi^k, x, \varphi_1), ((\psi^k, \varphi_1), x, \varphi_2)\} \subseteq \Lambda$.

11. T is *saturated under rule* $\Box$ if and only if for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, \Box\varphi) \in \Lambda$, then $(\psi^k, x, [\chi]\varphi) \in \Lambda$ for all $\chi \in \mathcal{L}_{\mathsf{PAL}}$.

12. T is *saturated under rule* $\Diamond$ if and only if for all $b = (\Lambda, \Sigma) \in T$, if $(\psi^k, x, \Diamond\varphi) \in \Lambda$, then $(\psi^k, x, [\chi]\varphi) \in \Lambda$ for some $\chi \in \mathcal{L}_{\mathsf{PAL}}$.

A tableau T is *saturated* if and only if it is saturated under all tableau rules.

We now prove that if a saturated tableau for a given formula φ is open, then φ is satisfiable. Suppose that T^∞ is an open saturated tableau for φ. Then, it contains at least one open branch $b = \langle\Lambda, \Sigma\rangle$. We use this branch to construct an epistemic structure $M = \langle S, R, V\rangle$ that satisfies φ as follows:

- $S = \{x \in \mathbb{N} \mid x \text{ occurs in } \Lambda\}$;
- $R_i =$ reflexive, transitive and symmetric closure of $\{(x, x') \mid (i, x, x') \in \Sigma\}$;
- $V(p) = \{x \mid (\psi^k, x, p) \in \Lambda \text{ for some } \psi^k\}$;

And we also define a function $f(x) = x$ for all x occurring in Λ.

Clearly, S is a non-empty set, R_i is an equivalence relation, $V(p)$ assigns a subset of S to each proposition that appears on the tableau and if $(i, x, x') \in \Sigma$, then $f(x')R_i f(x)$. Thus, we now show that for all labelled formulas $\lambda = (\psi^k, x, \varphi) \in \Lambda$, we have $\mathcal{P}(\lambda)$ defined as follows:

$$\mathcal{P}(\lambda) = \begin{cases} M|\psi^0, f(x) \models \psi_1 & \text{and} \\ \quad\vdots & \\ M|\psi^{k-1}, f(x) \models \psi_k & \text{and} \\ M|\psi^k, f(x) \models \varphi. & \end{cases}$$

We do this by induction on $\mathrm{size}(\lambda)$. The base case, when $\mathrm{size}(\lambda) = (0, 1)$, as well as the cases where $\lambda = (\psi^k, x, \neg\neg\varphi)$, $\lambda = (\psi^k, x, \varphi_1 \wedge \varphi_2)$ and $\lambda = (\psi^k, x, \neg(\varphi_1 \wedge \varphi_2))$, are straightforward. They are left to the reader.

Now, let $\lambda = (\psi^k, x, \mathrm{K}_i\varphi)$. By Lemma 105 and by the fact that the branch is open we have that (ψ^0, x, ψ_1), …, $(\psi^{k-1}, x, \psi_k) \in \Lambda$. The length of each of these labelled formulas is less than $\mathrm{len}(\lambda)$. By induction hypothesis we have that $M|\psi^0, f(x) \models \psi_1$, …, $M|\psi^{k-1}, f(x) \models \psi_k$. Now, by the fact that T^∞ is saturated under rules K, T, 4 and 5, for all x' such that xR_ix', Λ contains at least one of the following sets of labelled formulas:

$\{(\psi^0, x', \neg\psi_1)\}$;
$\{(\psi^0, x', \psi_1), (\psi^1, x', \neg\psi_2)\}$;
$\{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), (\psi^2, x', \neg\psi_3)\}$;
$\vdots$
$\{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \neg\psi_k)\}$;
$\{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \varphi)\}$.
The length of each of these labelled formulas is also less than $\mathrm{len}(\lambda)$. Again by induction hypothesis, for all x' such that xR_ix',
$M|\psi^0, f(x') \models \neg\psi_1$ or
$M|\psi^0, f(x') \models \psi_1$ and $M|\psi^1, f(x') \models \neg\psi_2$ or
$M|\psi^0, f(x') \models \psi_1$, $M|\psi^1, f(x') \models \psi_2$ and $M|\psi^2, f(x') \models \neg\psi_3$ or
$\vdots$
$M|\psi^0, f(x') \models \psi_1, \ldots, M|\psi_{k-1}, f(x') \models \neg\psi_k$ or
$M|\psi^0, f(x') \models \psi_1, \ldots, M|\psi^{k-1}, f(x') \models \psi_k$ and $M|\psi^k, f(x') \models \varphi$.
Then $M|\psi^k, f(x) \models \mathrm{K}_i\varphi$. Therefore, $\mathcal{P}(\lambda)$ holds.

Let $\lambda = (\psi^k, x, \neg\mathrm{K}_i\varphi)$. By Lemma 105 and by the fact that the branch is open we have that $(\psi^0, x, \psi_1), \ldots, (\psi^{k-1}, x, \psi_k) \in \Lambda$. The length of each of these labelled formulas is less than $\mathrm{len}(\lambda)$. By induction hypothesis we have $M|\psi^0, f(x) \models \psi_1, \ldots, M|\psi^{k-1}, f(x) \models \psi_k$. Now, by the fact that T^∞ is saturated under rule $\widehat{\mathrm{K}}$, $(i, x, x') \in \Sigma$ and that $\{(\psi^0, x', \psi_1), (\psi^1, x', \psi_2), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \neg\varphi)\} \subseteq \Lambda$. The length of each of these labelled formulas is less than $\mathrm{len}(\lambda)$. By induction hypothesis we have $M|\psi^0, f(x') \models \psi_1, \ldots M|\psi^{k-1}, f(x') \models \psi_k$ and $M|\psi^k, f(x') \models \neg\varphi$. Then $M|\psi^k, f(x) \models \neg\mathrm{K}_i\varphi$. Therefore, $\mathcal{P}(\lambda)$ holds.

Let $\lambda(\psi^k, x, [\varphi_1]\varphi_2)$. By Lemma 105 and by the fact that the branch is open we have that $(\psi^0, x, \psi_1), \ldots, (\psi^{k-1}, x, \psi_k) \in \Lambda$. The length of each of these labelled formulas is less than $\mathrm{len}(\lambda)$. By induction hypothesis we have $M|\psi^0, f(x) \models \psi_1, \ldots, M|\psi^{k-1}, f(x) \models \psi_k$. Now, because T^∞ is saturated under rule $\mathrm{R}[\cdot]$, either $\{(\psi^k, x, \neg\varphi_1)\} \subseteq \Lambda$ or $\{(\psi^k, x, \varphi_1), ((\psi^k, \varphi_1), x, \varphi_2)\} \subseteq \Lambda$. The length of each of these labelled formulas is less than $\mathrm{len}(\lambda)$. By induction hypothesis, either $M|\psi^k, f(x) \models \neg\varphi_1$ or both $M|\psi^k, f(x) \models \varphi_1$ and $M|\psi^k|\varphi_1, f(x) \models \varphi_2$. Then $M|\psi^k, f(x) \models [\varphi_1]\varphi_2$. Therefore, $\mathcal{P}(\lambda)$ holds.

Let $\lambda(\psi^k, x, \neg[\varphi_1]\varphi_2)$: By Lemma 105 and by the fact that the branch is open we have that $(\psi^0, x, \psi_1), \ldots, (\psi^{k-1}, x, \psi_k) \in \Lambda$. The length of each of these labelled formulas is less than $\mathrm{len}(\lambda)$. By induction hypothesis we have $M|\psi^0, f(x) \models \psi_1, \ldots, M|\psi^{k-1}, f(x) \models \psi_k$. Now, by the fact that T^∞ is saturated under rule $\mathrm{R}\langle\cdot\rangle$, $\{(\psi^k, x, \varphi_1), ((\psi^k, \varphi_1), x, \varphi_2)\} \subseteq \Lambda$. The length of each of these labelled formulas is less than $\mathrm{len}(\lambda)$. By induction hypothesis, $M|\psi^k, f(x) \models \varphi_1$ and $M|\psi^k|\varphi_1, f(x) \models \neg\varphi_2$. Then, $M|\psi^k, f(x) \models \varphi$. Therefore, $\mathcal{P}(\lambda)$ holds.

Let $\lambda = (\psi^k, x, \Box\varphi)$. Then, by the fact that T^∞ is saturated under rule $\Box$, Λ contains $(\psi^k, x, [\chi]\varphi)$ for all $\chi \in \mathcal{L}_{\mathsf{PAL}}$. As $\chi \in \mathcal{L}_{\mathsf{PAL}}$, the size of each of these labelled formulas is less than the size of λ. By Lemma 105 and the induction hypothesis, $M|\psi^k, f(n) \models [\chi]\varphi_1$ for all $\chi \in \mathcal{L}_{\mathsf{PAL}}$. Then $M|\psi^k, f(n) \models \varphi$. Therefore, $\mathcal{P}(\lambda)$ holds.

Now let $\Lambda = (\psi^k, x, \neg\Box\varphi)$ Then, by the fact that T^∞ is saturated under rule $\Diamond$, Λ contains $(\psi^k, x, \neg[p]\varphi)$ for some $p \in P$. The size of this labelled formula is less than

the size of λ. By Lemma 105 and the induction hypothesis, $M|\psi^k, f(n) \models \neg[p]\varphi_1$ for some $p \in P$. Then $M|\psi^k, f(n) \models \varphi$. Therefore, $\mathcal{P}(\lambda)$ holds. ∎

Abstract

In the 60's John McCarthy and Patrick Hayes pointed out the impossibility of giving compact descriptions for dynamical systems in first-order logic. This problem, that seemed to be intrinsic to every logical formalism for reasoning about actions, became known as *the frame problem*. More than twenty years later, Raymond Reiter proposed his famous (partial) solution to the frame problem. However, the associated inference method designed by Reiter turned out to be of high computational complexity. The problem of designing an efficient inference method for a formalism that solves the frame problem was named *the inferential frame problem* by Thielscher (1999).

The main contribution of this thesis is a solution to the inferential frame problem. Here, we first show that one can recast Reiter's solution in another formalism called *dynamic epistemic logic* (DEL). We do so by providing a polynomial reduction from a fragment of situation calculus to DEL. Then, we propose a novel proof method for DEL, whose computational complexity is much lower than that of the method proposed by Reiter. We also show that the computational complexity of our proof method is optimal. It follows that the whole method can be considered as a solution to the inferential frame problem.

A second important contribution of this work is the proposition of a novel formalism for reasoning about actions. In spite of its capacity in formalising *plan verification*, DEL is not suitable for formalising *planification*. The reason is that plan verification is as validity problem in DEL, while planning demands the construction of a plan. As a first effort towards the proper treatment of planning using this kind of logic, we propose here a novel formalism wherein one can quantify over actions. This formalism, the *arbitrary public announcement logic* (APAL), allows to formally expressing that "there exists a sequence of actions that leads to the goal". The idea is that, with quantification over actions, planning can become a validity problem. We provide a Hilbert-style axiomatisation of APAL, some expressivity results and also a proof method based on semantic tableaux.

Resumé

Chapitre 1 : Introduction

Conçu comme une activité humaine complexe, le raisonnement sur actions et plans est fortement relié à la capacité de raisonner sur les causes et les effets. Le sujet est vaste et, en même temps, est un des objets d'étude de plusieurs domaines de recherche comme la philosophie, l'informatique et la logique. Les résultats de ces études nous donnent de fondements théoriques essentiels pour la conception des systèmes automatiques. Comme bons exemples, nous pouvons citer la génie logicielle, la robotique, la logistique, l'éducation et les jeux. Pour bien illustrer ce que nous voulons dire ici par le terme 'raisonnement', nous utilisons l'exemple cidessous qui a été inspiré d'un puzzle logique de Smullyan (1992).

EXEMPLE 109 (LA DEMOISELLE OU LE TIGRE)
L'environnement est constitué d'un individu, aussi appelé *agent*, qui habite dans une salle ayant deux portes. L'agent peut ouvrir une des portes et derrière la porte ouverte l'agent trouvera ou bien la demoiselle ou bien le tigre. Si l'agent trouvera la demoiselle, alors il se mariera avec elle, et si l'agent trouvera le tigre, il sera tué par lui. Les actions disponibles sont :

- $listen_1$ et $listen_2$. En exécutant une de ces actions l'agent écoute ce qui se passe derrière la porte respective, ceci lui permettra d'entendre le bruit du tigre dans le cas où il en a un derrière la porte ; et
- $open_1$ et $open_2$. En exécutant une de ces actions, l'agent ouvre la porte respective, ceci lui permettra de se marier avec la demoiselle, ou de se faire tuer par le tigre, selon ce qu'il trouvera derrière la porte.

Cet exemple décrit un exemple de ce que nous appelons *système dynamique*. Un exemple d'*état initial* du système pourrait être : l'agent est vivant et non marié, la demoiselle est derrière la porte 1 et le tigre derrière la porte 2. Et un exemple de *but* pourrait être : l'agent est vivant et marié. Quand un système dynamique est accompagné d'un état initial et d'un but, il est appelé *problème de planification*. Une solution du problème de planification est une séquence d'actions, ou *plan*, tel que son exécution conduit à un état où le but est satisfait. Quand un système dynamique est accompagné aussi d'un plan, il est appelé *problème de vérification de plan*. La

vérification d'un plan *réussit* quand le plan donné est une solution du problème de planification.

Nous remarquons que le système de l'Exemple 109 a des actions *épistémiques* $listen_k$. Ce type d'action ne change pas nécessairement l'état physique du monde, mais il est capable de changer l'état épistémique de l'agent. Cela veut dire que nous permettons des descriptions d'état incomplètes : par exemple, il peut être le cas que l'agent ne sais pas ce que se trouve derrière chaque porte. Dans ce cas, pour éviter d'être tué par le tigre, il doit exécuter les actions d'écouter et en suite, basé sur l'information acquise en temps d'exécution, décider laquelle des portes ouvrir. Alors, ces scénarios impliquent que les plans soient *conditionnels*, i.e., ils se divisent en deux branches selon l'évaluation de l'information acquise en temps d'exécution.

Notre objectif ultime est de proposer un formalisme pour décrire des systèmes dynamiques. Ce formalisme doit être capable d'incorporer touts les éléments importants du raisonnement sur actions et plans dans des scénarios comme celui donné dans l'Exemple 109. Nous voulons aussi que ce formalisme puisse être utilisé dans la spécification des systèmes automatiques. Donc, nous adressons la question de la décidabilité de la méthode de démonstration, ainsi que son efficacité. Le terme 'efficace' ici veut dire que la complexité du calcul de telle procédure doit être la moins élevée possible.

Un des premiers formalismes utilisé pour atteindre ce but est un dialecte de la logique du second ordre proposé par McCarthy (1968), nommé *calcul de situations*. Ce formalisme présente deux problèmes principaux. En premier terme, la méthode d'inférence est seulement semi-décidable. En deuxième terme, il n'avait pas une solution pour le problème du décor.

Le *problème du décor représentationel* a été relevé par McCarthy and Hayes (1969). Grossièrement, il consiste en l'impossibilité de donner une description compacte à un système dynamique. Plus de vingt ans plus tard, Reiter (1991) a donné une solution partielle pour ce problème dans le calcul de situations. Pourtant, la procédure d'inférence donnée par Reiter n'est pas optimale. Le problème de concevoir une méthode d'inférence efficace pour un formalisme qui résout le problème du décor a été nommé le *problème du décor inférenciel* par Thielscher (1999).

Une des principales contributions de cette thèse est une solution au problème du décor inférenciel. Premièrement, nous montrons que la solution au problème du décor représentationel de Reiter peut être traduite dans la *logique épistémique dynamique* proposée par van Ditmarsch et al. (2005). Nous proposons une réduction polynomiale du calcul de situations vers cette logique. Ensuite, nous donnons une méthode de démonstration pour la logique épistémique dynamique, tel que la complexité est beaucoup moins élevée que celle de la méthode de Reiter. En plus, nous démontrons que cette méthode est optimale.

La logique épistémique dynamique est très appropriée pour la formalisation du problème de vérification de plans, mais pas pour le problème de la planification. La raison est que dans ce scénario, la vérification de plan se réduit à une vérification de validité de formules, tandis que pour la planification il est nécessaire de construire

un plan. Pour cette raison, nous proposons une autre logique dans laquelle il est possible de quantifier sur les actions épistémiques. Cette logique, appelée *logique des annonces publiques arbitraries*, permet la formalisation de 'il existe une action qui conduit au but'. Nous donnons aussi une méthode de démonstration basée sur les tableaux. Nous avons l'espoir que de futures extensions de cette méthode nous permettrons la planification en logique épistémique dynamique.

Chapitre 2 : À la recherche d'un formalisme approprié

Le langage du calcul des situations est un dialecte de la logique de second ordre qui comporte quelques éléments spéciaux :

- une constante s_0 pour représenter la situation initiale ;
- un symbole de fonction do pour modéliser les transitions des actions ; et
- un fluent spécial K pour modéliser la connaissance de l'agent. Ce fluent est défini par l'abréviation suivante :

$$\mathrm{K}(\varphi^{-1}, s) \stackrel{\text{def}}{=} \forall s'(R(s', s) \rightarrow \varphi^{-1}[s'])$$

  où R relie la situation actuelle aux situations que l'agent considère possibles.

Dans ce travail, nous restreindrons notre attention au fragment propositionnel de cette logique.

Les modèles du calcul de situations sont des modèles standard de Tarski qui respectent un ensemble de propriétés appelé *Axiomes Fondationels du Domaine*. Grossièrement, ces axiomes sont essentiels pour garantir que l'ensemble des situations forment une forêt. La racine d'une des ces arbres est la situation initiale s_0 et toutes les autres racines lui sont reliées par R. Chaque racine représente une situation initiale que l'agent considère possible, basé sur sa connaissance.

La solution de Reiter pour le problème du décor inférenciel, étendue par Scherl and Levesque (1993) pour le cas avec connaissance, est basée sur plusieurs hypothèses dont nous citons les suivantes:

1. Toutes les actions sont déterministes.
2. Toutes les occurrences des actions sont publiquement aperçues.
3. Les préconditions d'exécution des actions données sont complètes.
4. L'ensemble des effets des actions donné est complet et beaucoup plus petit que l'ensemble des fluents du langage.

Plusieurs propriétés sont impliquées par ces hypothèses. En particulier, il est possible de décrire un système dynamique par une *théorie basique d'actions* Θ qui comporte les formules suivantes.

- L'ensemble des "*unique-name axioms*" Φ_{una} avec des formules de la forme $a_1 \neq a_2$.
- L'ensemble des "*action-precondition axioms*" Φ_{ap} avec des formules de la forme $Poss(a, s) \leftrightarrow \psi_a(s)$.
- L'ensemble des "*successor-state axioms*" Φ_{ss} avec des formules de la forme $p(do(a, s)) \leftrightarrow \psi_p(a, s)$.
- L'ensemble des "*sensed fluent axioms*" Φ_{sf} avec des formules de la forme $sr(a, s = x) \leftrightarrow (((x = yes) \wedge p(s)) \vee ((x = no) \wedge \neg p(s)))$.
- Et l'ensemble Φ_{s_0} avec des formules qui décrivent la situation initiale.

Une théorie basique d'actions permet la définition d'une procédure de décision pour le problème de satisfiabilité de formules en calcul de situations. Cette procédure est définie par deux étapes. La première étape est la régression (l'opérateur reg_Θ) définie par Reiter. Cet opérateur, appliquée à une formule qui décrit un but en calcul des situations, retourne une formule équivalente mais qui ne contient pas d'actions. Ensuite, comme la formule retournée par la régression est "plus simple", i.e., elle ne contient que des fluents, opérateurs booléen et des fluents 'K', alors il est possible d'utiliser une méthode de preuve standard de vérification de satisfiabilité pour la logique S5.

Le problème du décor représentationel est donc résolu. Néanmoins, la formule retournée par la régression peut être exponentiellement plus grande que la formule de départ. C'est-à-dire, la méthode n'est pas optimale et donc le problème du décor inférenciel reste sans solution dans cette approche.

L'approche de Demolombe et al. (2003) utilise les mêmes idées pour proposer une méthode de régression pour une logique modale appelée EDL. Cette logique présente quelques différences syntaxiques et une sémantique plus attractive. En particulier, les situations sont implicites car ils ne font pas parti du langage et la fonction do est remplacée par l'opérateur $\langle\cdot\rangle$. Par exemple, la formule $\langle a\rangle\varphi$ veut dire 'φ est vraie après l'exécution de a'. La régression pour EDL est très similaire à la régression pour le calcul de situations. En plus, avec le même desavantage en concernant le problème du décor inférenciel.

Chapitre 3 : Une logique pour le changement épistémique et ontique

Les actions purement ontiques sont les actions qui n'impliquent aucune perception de la part de l'agent. Formellement, dans le cadre de la logique EDL, il s'agît des actions qui respectent les deux principes suivants :

- *Déterminisme épistémique* : si $t, u \in T_o(s)$, alors $R(t) = R(u)$.
- *"No-learning"* : si $t \in (R \circ T_o)(s)$ et $T_o(s) \neq \emptyset$, alors $t \in (T_o \circ R)(s)$.

Les actions purement épistémiques sont celles qui n'ont pas des effets sur le monde physique. Formellement, elles respectent le principe suivant :

- *Préservation* : si $t \in T_e(s)$, alors pour tout $p \in P$, ($s \in V_p$ ssi $t \in V_p$).

Le premier résultat de ce chapitre est le Théorème de la 'Séparation'. Nous montrons qu'en EDL, toutes les actions sont décomposables en deux actions que se suivent. La première est purement épistémique et la seconde purement ontique.

Ensuite, nous faisons une analyse des actions épistémiques. Comme deuxième résultat de ce chapitre nous montrons le Théorème de la 'Généralisation'. Ce théorème dit que toutes les actions épistémiques peuvent être simulées par des observations complexes. Les observations sont un type d'action qui ont la curieuse propriété d'être ontique et épistémique à la fois. En plus, elles peuvent être tenue comme le type d'actions épistémique le plus simple. En particulier, le théorème de la généralisation dit que les actions "sensing" peuvent être simulée par des compositions non déterministes des observations.

Un troisième résultat est achevé dans ce chapitre. Nous montrons formellement que les observations en EDL sont exactement les mêmes actions que les annonces publiques de la logique des annonces publiques PAL. Entre autres, cela veut dire que EDL et DEL ont une forte relation, qui est exploitée dans le chapitre suivant.

Chapitre 4 : Méthodes optimales pour le raisonnement

Dans ce chapitre nous reprenons les hypothèses de la solution du problème du décor pour le calcul de situations et étendrons la solution de Demolombe et al. (2003). Notre solution utilise les *descriptions d'actions* définie comme étant une structure de la forme :

$$D = \langle poss, \mathit{eff}^+, \mathit{eff}^-, \gamma^+, \gamma^- \rangle$$

tel que :

- $poss : A \rightarrow \mathcal{L}_{\mathsf{ELC}}$ attribue une formule à chaque action qui décrit sa précondition d'exécutabilité ;
- $\mathit{eff}^+ : A \rightarrow \wp(P)$ attribue un ensemble fini d'effets positifs possibles à chaque action ;
- $\mathit{eff}^- : A \rightarrow \wp(P)$ attribue un ensemble fini d'effets négatifs possibles à chaque action ;
- γ^+ est une famille de fonctions $\gamma^+(a) : \mathit{eff}^+(a) \rightarrow \mathcal{L}_{\mathsf{ELC}}$. Elle attribue une formule à chaque pair (a, p) qui décrit la précondition pour que l'action a rende p vrai ; et
- γ^- est une famille de fonctions $\gamma^-(a) : \mathit{eff}^-(a) \rightarrow \mathcal{L}_{\mathsf{ELC}}$. Elle attribue une formule à chaque pair (a, p) qui décrit la précondition pour que l'action a rende p faux.

Notons que Scherl et Levesque (2003) restreint le codomaine de $poss$, γ^+ et γ^- aux formules propositionnelles. Nous avons étendu ce codomaine aux formules dans $\mathcal{L}_{\mathsf{ELC}}$. Ceci permet la formalisation des actions comme 'faire un appel téléphonique', dont la précondition d'exécution est de connaître le numéro de téléphone de l'interlocuteur.

Les modèles pour D sont obtenus en rajoutant des relations de transition aux modèles de la logique épistémique.

En plus, les D-modèles doivent satisfaire les restrictions suivantes :

1. *"No-Forgetting" :* $(T_a \circ R_i)(s) \subseteq (R_i \circ T_a)(s)$.
2. *"No-Learning" :* si $T_a(s) \neq \emptyset$, alors $(R_i \circ T_a)(s) \subseteq (T_a \circ R_i)(s)$.
3. *Déterminisme :* si $t_1, t_2 \in T_a(s)$, alors $t_1 = t_2$.
4. *Exécutabilité :* $T_a(s) \neq \emptyset$ ssi $\langle S, R, V\rangle, s \models (a)$.
5. *Préservation (épistémique) :* si $t \in T_e(s)$, alors

$$t \in V(p) \text{ ssi } s \in V(p) \quad \text{pour tout } p \in P$$

6. *Pos-condition (ontique) :* si $t \in T_o(s)$, alors
   - $p \notin \mathit{eff}^+(o)$ et $s \notin V(p)$ implique $t \notin V(p)$;
   - $p \notin \mathit{eff}^-(o)$ et $s \in V(p)$ implique $t \in V(p)$;
   - $p \in \mathit{eff}^+(o)$ et $\langle S, R, V\rangle, s \models (o, p)$ implique $t \in V(p)$;
   - $p \in \mathit{eff}^-(o)$ et $\langle S, R, V\rangle, s \models (o, p)$ implique $t \notin V(p)$;
   - $p \in \mathit{eff}^+(o)$ et $\langle S, R, V\rangle, s \not\models (o, p)$ et $s \notin V(p)$ implique $t \notin V(p)$;
   - $p \in \mathit{eff}^-(o)$ et $\langle S, R, V\rangle, s \not\models (o, p)$ et $s \in V(p)$ implique $t \in V(p)$.

Soit D une description d'actions. Il est possible de définir une procédure $\mathrm{reg}_D(\varphi)$ tel que :

$$\models_D \varphi \quad \text{ssi} \quad \models_{\mathsf{ELC}} \mathrm{reg}_D(\varphi)$$

Notons que reg_D est sous-optimal, puisqu'il y a des formules tel que $\mathrm{reg}_D(\varphi)$ est exponentiellement plus long que φ, exactement comme dans le calcul des situations.

Une tradition différente dans la modélisation de connaissance et changement a été suivi par, par exemple, Plaza (1989), Baltag et al. (1998) et van Benthem (2006). La logique DEL de van Ditmarsch et al. (2005) et Kooi (2007) se situe dans cette tradition. Elle est basée sur les annonces publiques et les affectations publiques.

Le langage de DEL remplace les opérateurs $[a]$ par les opérateurs $[\varphi]$ et $[\sigma]$ où σ est de la forme $p := \varphi$. Le premier est dit annonce publique de φ et le deuxième est dit affectation publique de la valeur de vérité de φ à l'atome p. Les modèles de DEL sont des modèles épistémiques et les actions sont définies comme des operations de mise à jours.

Les annonces modélisent les actions épistémiques, tandis que les affectations modélisent les actions ontiques. La traduction δ_D de $\mathcal{L}_D$ dans $\mathcal{L}_{\mathsf{DELC}}$ est donc évidente. Soit D une description d'actions, nous avons :

$$\delta_D([a]\varphi) = [poss(a)][\sigma_a]\varphi$$

où σ_a est l'affectation complexe :

$$\{p := \gamma^+(a,p) \vee (\neg\gamma^-(a,p) \wedge p) \mid p \in \mathit{eff}^+(a) \cup \mathit{eff}^-(a)\}$$

THÉORÈME 110
Soit D une description finie d'actions à la Reiter, et soit $\varphi \in \mathcal{L}_D$. Alors φ est D-satisfiable si et seulement si $\delta_D(\varphi)$ est DELC-satisfiable, et $|\delta_D(\varphi)| \leq |\varphi| \times |D|$.

La preuve est par induction sur la structure de φ. Donc δ_D est polynomial, et le problème de décider si étant donné D et φ, φ est D-satisfiable peut être transformé d'une façon polynomiale dans un problème de DELC-satisfiabilité.

Ensuite, nous proposons une extension de la réduction polynomiale de Lutz (2006). Notre réduction est étendue à toute la logique épistémique dynamique. L'extension est basée sur le théorème suivant.

THÉORÈME 111 (ÉLIMINATION DES AFFECTATIONS)
Soit
$[p_1 := \varphi_1, \ldots, p_n := \varphi_n]\psi$ une sous-formule de la formule χ in $\mathcal{L}_{\mathsf{DELC}}$. Soit ψ' obtenu de ψ par la substitution de chaque occurrence de p_k par x_{p_k}, où x_{p_k} est une nouvelle proposition que n'apparaît pas dans χ. Soit χ' obtenu à partir de χ en remplaçant $[p_1 := \varphi_1, \ldots, p_n := \varphi_n]\psi$ par ψ'. Soit B l'abréviation de la conjonction des équivalences (bi-implications) $\bigwedge_{1\leq k\leq n}(x_{p_k} \leftrightarrow \varphi_k)$.

1. Pour $|N| = 1$. Si $\chi \in \mathcal{L}_{\mathsf{DEL}}$, alors χ est DEL-satisfiable si et seulement si

$$\chi' \wedge \bigwedge_{\ell \leq \mathrm{md}(\varphi)} \mathrm{K}_i^\ell B$$

   est DEL-satisfiable, où la profondé modale $\mathrm{md}(\varphi)$ est le nombre maximal des opérateurs modaux imbriqués dans ψ.

2. Pour $|N| \geq 2$. Si $\chi \in \mathcal{L}_{\mathsf{DEL}}$, alors χ est DEL-satisfiable si et seulement si

$$\chi' \wedge \bigwedge_{\ell \leq \mathrm{md}(\varphi)} \mathrm{E}_N^\ell B$$

   est DEL-satisfiable.

3. Si $\chi \in \mathcal{L}_{\mathsf{DELC}}$, alors χ est DELC-satisfiable si et seulement si

$$\chi' \wedge \mathrm{C}_N B$$

   est DELC-satisfiable.

Ceci implique que dans le cas particulier où chaque R_i est une relation d'équivalence, le problème de la D-satisfiabilité sans l'opérateur C_G est NP-complet si $N = 1$, et PSPACE-complet si $N \geq 2$. Le problème de la D-satisfiabilité avec l'opérateur C_G est EXPTIME-complet.

Chapitre 5 : Le raisonnement avec tableaux analytiques

Dans ce chapitre nous présentons une méthode de démonstration pour la logique des annonces publiques qu'utilise les tableaux. Étant donné une formule φ, cette méthode essaie de construire un modèle pour φ de manière systématique. Quand la méthode échoue, cela veut dire que φ est inconsistante et donc sa négation est valide.

La méthode présentée est modulaire et peut être utilisée aussi dans les cas où la logique épistémique de base est K, KT, S4 ou S5.

Dans le tableau, nous utilisons des formules étiquetées. Cela veut dire que les formules sont préfixées par un nombre qui correspond à un monde possible, d'une façon similaire à celle utilisée dans la méthode de tableaux de Fitting (1983, Chapitre 8). En plus, les formules de notre méthode de tableaux sont aussi préfixées par des séquences finies d'annonces qui correspondent aux restrictions successives du modèle. Par exemple, $(p \cdot \mathrm{K}_i q, 1, [p]p \wedge \mathrm{K}_i \neg p)$ est une formule étiquetée où $p \cdot \mathrm{K}_i q$ est une séquence d'annonce et 1 est un monde possible.

Les règles de notre méthode étendent les règles utilisées pour la logique épistémique. Ci-dessous, nous montrons un exemple de règle dans cette méthode.

$\mathrm{R}\widehat{\mathrm{K}}$: si $(\psi^k, x, \neg \mathrm{K}_i \varphi) \in \Lambda$, alors $B = \{(\Lambda \cup \{(\psi^0, x', \psi_1), \ldots, (\psi^{k-1}, x', \psi_k), (\psi^k, x', \neg\varphi)\}, \Sigma \cup \{(i, x, x')\})\}$ pour un x' qui n'apparaît pas dans Λ.

Il s'agît de la règle pour le dual de l'opérateur 'K'. Comme dans la méthode standard pour la logique modale, cette règle construit un nouveau monde possible x'. Mais dans ce contexte, pour que la règle soit correcte, il est nécessaire d'assurer que le monde x' fait parti des modèles qui précèdent le modèle ψ^k dans la suite de mises à jours.

Nous démontrons l'adéquation et la complètude de la méthode ainsi que la terminaison. Ensuite, nous présentons une stratégie optimale pour la logique des annonces publiques dont la logique épistémique de base est K ou KT. Dans ces deux cas, la méthode travaille en espace polynomial.

Chapitre 6 : À la recherche des plans

Dans ce chapitre nous développons un formalisme dans lequel nous pouvons exprimer ce qui devient vrai après l'exécution d'une action sans avoir besoin de faire référence à l'action elle-même. Formellement, dans la logique des annonces publiques, $p \rightarrow [p]\mathrm{K}_i p$ est valide. Cela est équivalent à $\langle p \rangle \mathrm{K}_i p$ qui peut être lue 'l'annonce de p est exécutable et après son exécution l'agent i sait p'. Ceci implique qu'il existe une annonce ψ, ($\psi = p$), qui fait que l'agent sache p, d'une façon un

peu plus formelle : $\exists\psi.\langle\psi\rangle \mathrm{K}_i p$. Nous introduisons un opérateur modal qui exprime exactement cette notion :

$$\Diamond \mathrm{K}_i p$$

Évidament, la valeur de vérité de cette expression dépend du modèle : p doit être vrai. Dans le cas où p est faux, nous avons plutôt $\Diamond \mathrm{K}_i \neg p$. Donc la formule $\Diamond(\mathrm{K}_i p \vee \mathrm{K}_i \neg p)$ est valide. La logique qui correspond à ces idées est nommée *logique des annonces publiques arbitraires* (APAL).

La sémantique du nouvel opérateur est définie de la façon suivante :

$$M, s \models \Box\varphi \quad \text{ssi} \quad \text{pour toute } \psi \in \mathcal{L}_{\mathsf{EL}},\ M, s \models [\psi]\varphi$$

Quelques résultats intéressants sont achevés. Par exemple en utilisant les validités suivantes :

1. $\Box p \leftrightarrow p$
2. $\Box(\varphi \wedge \psi) \leftrightarrow (\Box\varphi \wedge \Box\psi)$
3. $\Box(\varphi \wedge \psi) \leftrightarrow (\Box\varphi \wedge \Box\psi)$ où $\varphi \in \mathcal{L}_{\mathsf{PL}}$
4. $\models_{\mathsf{APAL}} \Box(\widehat{\mathrm{K}}_i\varphi_0 \vee \mathrm{K}_i\varphi_1 \vee \cdots \vee \mathrm{K}_i\varphi_n) \leftrightarrow (\varphi_0 \vee \mathrm{K}_i(\varphi_0 \vee \varphi_1) \vee \ldots \vee \mathrm{K}_i(\varphi_0 \vee \varphi_n))$

nous avons démontré que dans le cas où $|N| = 1$, la logique APAL est réductible à la logique épistémique. Poutant, APAL multi-agents est strictement plus expressive que la logique épistémique.

Parmis les autres résultats importants, nous avons démontré que APAL n'est pas compacte et que la vérification de modèle en APAL est décidable. Nous avons aussi proposé une axiomatisation infinitaire et une méthode de tableaux pour APAL. Malheureusement pourtant, nous avons pas pu démontré la décidabilité d'APAL.

Chapitre 7 : Conclusion

Le travail de recherche de cette thèse a commencé avec l'idée d'améliorer l'approche de Demolombe et al. (2003) pour s'adresser aux actions épistémiques. Dans cette approche, la solution de Reiter pour le problème du décor – qui ne s'adresse qu'aux changement ontiques – est traduite en logique modale. Une des restrictions imposée est que toutes les actions doivent respecter le principe "no-learning". Au premier regard, il semble improbable qu'une vraie action épistémique puisse respecter ce principe. Pourtant, nous avons trouvé un type d'action épistémique qui le respecte : les *observations* (Chapitre 3) respectent "no-learning", et en plus sont capables de faire évoluer l'état épistémique des agents.

Les observations sont un type d'action intéressant, mais elles ne semblent pas très expressives. Par exemple, elles sont différentes des actions appelées "sensing", formalisées par exemple dans l'approche de Scherl and Levesque (2003). C'est pourquoi nous avons besoins de montrer que toutes les actions épistémiques peuvent être exprimées par une observation "complexe". Ce résultat est établi comme le Théorème 'Les observations sont généralles' dans la Section 3.3.

Ensuite, nous avons montré que les observations n'étaient rien d'autre que des annonces publiques, et que les changements ontiques en EDL étaient très proches des affectations publiques. Annonces publiques et affectations publiques sont présentes dans la logique épistémique dynamique (DEL). DEL est une famille de logiques étudiée dans la "tradition Néerlandaise" de raisonnement sur les actions et la connaissance. En essayant de comprendre les relations entre toutes ces notions, nous avons fini pour découvrir que les deux approches – EDL et DEL – pourraient réaliser le même travail.

La méthode de raisonnement déjà existante pour DEL, construite avec des axiomes de réduction, n'est pas optimale. Donc le prochain pas était de trouver une méthode efficace pour raisonner dans ce scénario. C'est-à-dire, nous avons commencé à chercher une solution pour le problème du décor inférenciel : le problème de trouver une méthode de raisonnement efficace pour un formalisme qui résout le problème du décor représentationel. En ce moment, le travail s'est divisé en deux pistes différentes. Dans une des deux, nous avons étendu une méthode de preuve, proposée par Lutz (2006), pour un fragment de DEL. Dans l'autre piste, nous avons développé une méthode de tableaux pour DEL.

La première piste s'est avérée plus productive en terminant avec une méthode de preuve optimale pour la logique DEL comme nous avons désiré. L'extension de la méthode de Lutz proposée dans cette thèse s'adresse aussi aux changements ontiques, et est la seule méthode optimale pour cette logique en ce moment.

La seconde piste ne nous a pas donné une méthode de preuve pour toute la logique DEL, mais a abouti à une nouvelle méthode de preuve pour un des ses fragments : la logique des annonces publiques (PAL). Nous avons proposé aussi des stratégies pour décider la satisfiabilité des formules en deux cas spéciaux : K-PAL et KT-PAL. En plus, nous avons des raisons pour croire que des stratégies optimales pour S4-PAL et S5-PAL existent.

Nous remarquons que ces résultats contribuent aussi pour mettre ensemble deux communautés scientifiques différentes. La "communauté de Toronto", dont les adeptes ont développé des extensions pour le calcul de situations de McCarthy, et la ' "communauté d'Amsterdam", dont les adeptes utilisent des formalisations en logique épistémique dynamique, essentiellement, des mêmes problèmes. Quelques idées sur les similarités entre calcul de situations et logique épistémique dynamique ont été déjà relevées indépendamment par van Benthem (2007). Ici, nous allons au delà, et démontrons formellement les connexions entre les deux approches.

Le dernier but poursuivi dans ce travail était un traitement approprié pour la tache de planification. Nous remarquons que la vérification de plan, le problème que nous adressons en utilisant DEL, peut être vu comme vérification de validité de formules. Pourtant cela n'est pas le cas pour la planification, parce que pour cette dernière, nous avons besoin de construire le plan. Donc nous avons développé un nouveau formalisme appelé logique des annonces publiques arbitraires (APAL). Il s'agît de notre premier effort vers un traitement approprié de la tâche de planification en logique modale. En fait, nous n'adressons pas vraiment la planification avec APAL, mais encore un autre problème entre la vérification de plans et la planification appelée vérification d'existence de plan. Même pour cette tâche pourtant, APAL

n'est pas idéal. La première raison : APAL ne traite pas les changements ontiques. La seconde raison : si nous permettons que l'opérateur diamant d'APAL quantifie sur les plans, alors ils doivent être des plans significatifs, comme défini dans la Section 2.5.2. La troisième et la plus importante raison : APAL quantifie sur l'ensemble de toutes les actions épistémiques, et non pas seulement sur les "actions épistémiques possibles pour l'agent", ce qui serait plus adéquat. Pour expliquer pourqui cela est important, supposons la présence d'un agent aveugle dans une salle. Cet agent ne devrait pas être capable d'observer (parmi ses propres moyens) si la lumière est allumée ou pas. Parce que chaque agent a des capacités limitées et ne peut pas tout observer. La quantification sur toutes les observations n'est pas réaliste. Plusieurs résultats sur APAL ont été achevés pourtant. Nous proposons une axiomatisation hilbertienne et une méthode de tableaux. Et nous avons démontré aussi quelques résultats d'expressivité.

Une simple extension de la méthode de tableaux définie dans le Chapitre 5 doit permettre la définition d'une procédure qui décide la satisfiabilité des formules en DEL avec des affectations publiques (changement ontique). Les règles qui doivent être ajouter sont simples. Nous remarquons que si $[p := \psi]\varphi$ est satisfiable, alors ou bien les deux ψ et p sont faux après l'affectation, ou bien les deux ψ et p sont vrais après l'affectation. Cela correspond à la règle suivante :

$\mathrm{R} :=$ if $(\psi^k, x, [p := \psi]\varphi) \in \Lambda$, alors
$B = \{(\Lambda \cup \{(\psi^k, x, \neg\psi), (\psi^k(p := \psi), x, \neg p), (\psi^k(p := \psi), x, \varphi)\}, \Sigma), (\Lambda \cup \{(\psi^k, x, \psi), (\psi^k, (p := \psi), x, p), (\psi^k(p := \psi), x, \varphi)\}, \Sigma)\}$.

Il n'est pas claire, poutant, si cette extension est optimale.

Une des plus intéressantes questions relevée par ce travail a été déjà mentionnée dans la Section 4.6. Notre méthode de démonstration n'est pas optimale si les actions "sensing" $!!\varphi$ sont définies comme des operations primitives en DEL. Il n'est pas claire comment l'axiome de réduction associé :

$$\mathrm{K}_i[!!\varphi]\psi \leftrightarrow ((\varphi \rightarrow \mathrm{K}_i(\varphi \rightarrow [!!\varphi]\psi)) \wedge (\neg\varphi \rightarrow \mathrm{K}_i(\neg\varphi \rightarrow [!!\varphi]\psi)))$$

peut être intégré "efficacement" dans la réduction de Lutz. Nous savons que la satisfiabilité en DEL avec des actions de "sensing" est PSPACE-dûr. La forme des axiomes de réduction implique que ce problème est dans EXPSPACE. Nous laissons comme une question ouverte la classe de complexité du problème de satisfiabilité de DEL avec actions de "sensing".

Il est possible d'introduire d'autre types d'actions dans l'approche présentée ici. Une extension possible est l'addition des actions non publiques. Ça peut être fait en enrichissant les descriptions d'action de la Section 4.2 avec des relations d'accessibilité $R : N \rightarrow (A \times A)$. Donc les descriptions d'actions ont une forme similaire aux modèles d'actions proposés par Baltag et al. (1998). Dans l'approche de Baltag et al., les actions dans A sont reliées par des flèches étiquetées avec des agents dans N. Nous les interprétons de cette manière : si $(a, b) \in R_i$, alors quand l'action a est exécutée, l'agent i croit que l'action b a été exécutée. En particulier, quand

$b = \mathsf{skip}$, l'occurrence de a n'est pas aperçue par l'agent i. Donc, cette action n'est pas publique. En utilisant une version légèrement différente de DEL, Baltag et al. définissent des transformations de modèles pour ce type d'action. Les axiomes de réduction sont proposés aussi. À partir de cette dernière observation et les résultats achevés par cette thèse, nous concluons que les actions non publiques peuvent facilement être introduites aussi dans le calcul de situations.

Resumo

Amo-te assim, desconhecida e obscura,
Tuba de alto clangor, lira singela,
Que tens o tom e o silvo da procela
E o arrolo da saudade e da ternura!

– Olavo Bilac

Nos anos 60, John McCarthy e Patrick Hayes apontaram a impossibilidade de obter-se representações compactas de sistemas dinâmicos em lógica de primeira ordem. Este problema, que parecia intrínsico a todo formalismo lógico decidado ao racioncíno sobre ações, recebeu o nome de *problema do quadro*. Mais de vinte anos mais tarde, Raymond Reiter propos sua famosa solução (parcial) para o problema do quadro. No entanto, o método de inferência associado concebido por Reiter possui complexidade computacional elevada. O problema de obter-se um método de inferência eficiente para um formalismo que dispõe de uma solução para o problema do quadro recebeu de (Thielscher, 1999) o nome de *problema do quadro inferencial*.

A principal contribuição desta tese é uma solução ao problema do quadro inferencial. Nós primeiramente mostramos que é possível modelar-se a solução de Reiter em outro formalismo chamado *lógica epistêmica dinâmica* (DEL). Nós o demonstramos através de uma redução polinomial de um fragmento do cálculo de situações à DEL. Em seguida, nós propomos um novo método de prova para DEL, o qual possui complexidade computacional muito inferior à do método proposto por Reiter. Nós também mostramos que nosso método é optmial. Segue que o método como um todo pode ser considerado como uma solução do problema do quadro inferencial.

Uma segunda contribuição importante deste trabalho é a proposição de um novo formalismo para raciocínio sobre ações. A despeito da sua capacidade em formalizar *verificação de planos*, DEL não é apropriada para formalização de *planejamento*. A razão é o fato de que verificação de plano é um problema de validade em DEL, enquanto que planejamento exige a construção de um plano. Como uma primeira tentativa na direção de um tratamento adequado de planejamento neste tipo de lógica, nós propomos aqui um novo formalismo no qual pode-se quantificar sobre ações. Este formalismo, a *lógica de anúncios públicos arbitrarios* (APAL), permite a expressão formal de que "existe uma seqüência de ações que conduz ao objetivo". A idéia é a de que, com quantificação sobre ações, planejamento pode ser visto como um problema de validade. Nos propomos uma axiomatização ao estilo de Hilbert

para APAL, alguns resultados de expressividade e um método de prova baseado em tablôs semânticos.

Glossary

$:=$	public assignment operator	43
δ	translation from D to DEL	45
$[\![\ ]\!]$	extention in the model	23, 26
$\Box$	arbitrary announcement operator	63
$\Box$	arbitrary plan operator	19
$\Diamond$	dual of the arbitrary announcement operator	64
$[\,]$	dynamic operator	25, 37, 43, 63
$\langle\,\rangle$	dual of the dynamic operator	25, 38
!	observation operator	37
!!	test (sensing) operator	37, 38, 50
APAL	arbitrary public announcement logic	63
C	common knowledge operator	21, 43
D	action description	40
DEL	dynamic epistemic logic	43
DELC	dynamic epistemic logic with common knowledge	43
Dep	dependence relation	29
EDL	epistemic dynamic logic	25
EDLm	epistemic dynamic logic with meaningful plans	29
EDLo	epistemic dynamic logic with observations	37
EL	epistemic logic	21
ELC	epistemic logic with common knowledge	21
ES	logic ES	19
$\widehat{\mathrm{K}}$	dual of the knowledge operator	21
K	knowledge operator	9, 19, 21, 25, 29, 37, 43, 63
$\mathcal{L}_x$	the language of the logic x	9, 21, 25, 29, 37, 42, 43, 63
PDL	propositional dynamic logic	25

PL	propositional logic	21
reg	regression operator	16, 32
Sit	situation calculus	9

Méthodes optimales pour le raisonnement sur actions et plans dans des systèmes multi-agents

Tiago de Lima

Cet travail présente une solution au problème du décor inférenciel. Nous réalisons cela en donnant une réduction polynomiale d'un fragment du calcul des situations vers la logique épistémique dynamique (DEL). En suite, une nouvelle méthode de preuve pour DEL, dont la complexité algorithmique est inférieure à celle de la méthode de Reiter pour le calcul de situations, est proposée. Ce travail présente aussi une nouvelle logique pour raisonner sur les actions. Cette logique permet d'exprimer formellement qu''il existe une suite d'action conduisant au but'. L'idée étant que, avec la quantification sur les actions, la planification devient un problème de validité. Une axiomatisation et quelques résultats d'expressivité sont donnés, ainsi qu'une méthode de preuve basée sur les tableaux sémantiques.

Mots clés : raisonnement sur actions et plans, raisonnement sur la connaissance, logique épistémique dynamique, calcul des situations, problème du décor.

Cette thèse, présentée et soutenue à Toulouse le 22 octobre 2007, a été réalisée sous la direction d'Andreas Herzig. L'auteur a obtenu le grade de Docteur en Informatique de l'Université de Toulouse.

Université de Toulouse – Institut de Recherche en Informatique de Toulouse
118 route de Narbonne, 31062 TOULOUSE CEDEX 9, France
http://www.irit.fr

Optimal Methods for Reasoning about Actions and Plans in Multi-agent Systems

Tiago de Lima

This work presents a solution to the inferential frame problem. We do so by providing a polynomial reduction from a fragment of situation calculus to espistemic dynamic logic (DEL). Then, a novel proof method for DEL, such that the computational complexity is much lower than that of Retier's proof method for situation caluculs, is proposed. This work also presents a new logic for reasoning about actions. This logic allows to formally express that 'there exists a sequence of actions that leads to the goal'. The idea is that, with quantification over actions, planning can become a validity problem. An axiomatisation and some expressivity results are provided, as well as a proof method based on sematic tableaux.

Keywords: reasoning about actions and plans, reasoning about knowledge, dynamic epistemic logic, situation calculus, frame problem.

This thesis, presented and defended at Toulouse on the 22nd October 2007, was performed under the supervision of Andreas Herzig. The author obtained the degree of Docteur en Informatique de l'Université de Toulouse.

Université de Toulouse – Institut de Recherche en Informatique de Toulouse
118 route de Narbonne, 31062 TOULOUSE CEDEX 9, France
http://www.irit.fr