ON THE APPLICATION OF FORMAL METHODS TO CLINICAL GUIDELINES: AN ARTIFICIAL INTELLIGENCE PERSPECTIVE

Een wetenschappelijke proeve op het gebied van de
Natuurwetenschappen, Wiskunde en Informatica

Proefschrift

ter verkrijging van de graad van doctor
aan de Radboud Universiteit Nijmegen
op gezag van de rector magnificus prof. mr. S.C.J.J. Kortmann,
volgens besluit van de College van Decanen
in het openbaar te verdedigen op vrijdag 18 april 2008
om 13.30 uur precies

door

Aart Jan Hommersom
geboren op 25 februari 1980
te Rheden

Promotor: Prof. dr. ir. Th.P. van der Weide

Copromotor: Dr. P.J.F. Lucas

Manuscriptcommissie:
Prof. dr. F.W. Vaandrager
Prof. dr. J-J.Ch. Meyer (Universiteit Utrecht)
Prof. dr. ir. L.C. van der Gaag (Universiteit Utrecht)
Prof. dr. F.A.H. van Harmelen (Vrije Universiteit Amsterdam)
Dr. M. Balser (Universiteit Augsburg, Duitsland)

SIKS Dissertation Series No. 2008-06
The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

This work has been partially supported by the European Commission's IST program, under contract number IST-FP6-508794 (PROTOCURE II).

ISBN: 978-90-9022824-2

To my parents

Preface

It was during the preparation of my master thesis that I realised how much fun research can be. In particular, to study people's ideas and to come up with new ideas, even though the problem may be too difficult to solve in its full generality. It also made me realise that this attitude is less suitable for working on a commercial product or at a "scary software company" as my supervisor at that time, John-Jules Meyer, so strikingly put it. Luckily, he put me into contact with Peter Lucas, who had a vacancy for a PhD position at that time. As a PhD student, I joined the Protocure project, which was a European project that dealt with the topic of this thesis, namely, formal methods in context of clinical guidelines. The focus on research topics in the area of artificial intelligence with a clear application and high potential impact was and still is very appealing.

As in many theses, I would like to devote the rest of this preface on acknowledging the people that supported me in one way or the other the last four years. First of all, I want to thank my co-promotor and daily supervisor for the last four years, Peter Lucas, who has taught me a tremendous amount about research. For a young researcher, the broadness and the depth of work that is being published in the various areas related to the work in this thesis is overwhelming. Peter has guided me through this and has given me insight and confidence in the process of doing research. Peter has not only been a formal supervisor, we also had a lot of fun. Our trips to meetings and conferences have always been most enjoyable and always filled with the essentials, such as art and wine! I am looking forward to continuing working in his group.

After I worked for a year in Nijmegen, Perry Groot joined our group. He gave me a renewed motivation to continue with the project and has since then taught me important lessons about research. His opinions are always honest and do not beat around the bush, which I have always considered very valuable. Moreover, several chapters are the result of a collaboration with him.

Besides Peter and Perry, several other people have been a co-author on the papers that formed the basis of this thesis: my promotor Theo van der Weide, Patrick van Bommel, Michael Balser, Jonathan Schmitt, Annette ten

Teije, Radu Serban, Frank van Harmelen, Mar Marcos, and Begoña Martínez-Salvador. Also, I am very grateful to the rest of the Protocure members that have taught me many things about clinical guidelines. Furthermore, various chapters have been improved by careful reading of Julien Schmaltz, Perry Groot, Nivea de Carvallo Ferreira and by the members of the manuscript committee.

The former IRIS group in Nijmegen where I have spent the last four years has been a social and friendly group. For this I would like to thank the various members that have come and gone the last four years. It is special that so many of you have become more than just colleagues. My roommates in 'the old building', Marcel and Ildikó, have been very influential in my thinking about research and life in general. Also, I would like to mention the regular coffee breaks with Henriëtte and Ildikó, which have been most enjoyable!

In my personal life, Tjarko, Luuk, and Tristan have been great friends whom I should call more often. Last, but not least, my family is thanked for their endless support and putting up with me while I was staring at the screen during the weekends. Thank you.

Arjen Hommersom
February 2008

Contents

Chapter 1

Introduction

Many systems, both natural and artificial, are very complex, making their understanding and analysis challenging. There are various reasons why the analysis of systems is useful. For example, an analysis of a biological system can be used to predict the behaviour of the system, whereas an analysis of a computer system can be used to find out if it satisfies its requirements.

In order to support the analysis by means of computers, symbolic calculation can be used, which allows one to draw conclusions expressed in symbols according to certain rules of inference. These techniques differ from other types of mathematical techniques in the sense that, instead of simulating a system or establishing a numerical value through a numerical analysis, theorems are proved about the behaviour of a system. This is what is generally understood by the term *formal verification*. The formal notation and techniques for the formal specification and verification that are used in the development of digital systems are referred to by the term *formal methods*.

In the context of computer systems, there is a great deal of experience in applying formal methods, e.g., modelling and verification of hardware components or of security-critical applications. Although medicine is a field in which mistakes made have a major bearing on the health and life expectancy of people, and thus can be seen as a prototypical safety-critical area, only very few researchers have considered using formal methods in this context. It was the aim of this thesis to investigate the potential of using formal methods to analyse medical problems. In particular, this thesis discusses applications of formal methods to so-called clinical guidelines, which are documents advising healthcare professionals and patients in this clinical decision making.

Contrary to what the name suggests, "formal methods" are not methodologies, i.e., they do not specify how the tools and notation should be applied to a given problem. It was the main objective of the research described in this thesis to explore the use of formal methods as we know from computer science to the verification of clinical guidelines.

In this chapter, we describe the development of symbolic reasoning to its current state and how this could be relevant to clinical guidelines. On the basis of a discussion of related work that has been done in this area, several research questions are formulated that further define the scope of this thesis.

1.1 Development of Symbolic Reasoning

Already at the end of the 17th century, Gottfried Leibniz discussed his idea for the development of a *calculus ratiocinator*, a sort of "calculating machine", which can determine the truth value of a proposition. He anticipated that this could settle philosophical problems by purely mechanical means and "when there are disputes among persons, we can simply say: Let us calculate, without further ado, and see who is right" (*The Art of Discovery (1685)*) [Stanford Encyclopedia, 2002]. As natural language is inherently ambiguous, Leibniz understood that this required a clean universal language, which he called *characteristica universalis*, to express scientific problems in, which he started to develop. At the beginning of the 19th century, it was not uncommon to express basic mathematical notions in symbols, for example in algebra; yet, basic logical reasoning itself had still not been formalised. Around 1850, Boole and others introduced symbols for the logical connectives and a few decades later Frege and Peirce introduced symbols for predicate logic. As the language is then described purely formal, i.e., it has a mathematically defined form and meaning, it is possible to describe reasoning as moving around symbols according to certain rules of inference. This subject was called 'mathematical logic' by Peano, whereas Venn coined the term 'symbolic logic', to stress the difference with the informal logics which were then prevalent in philosophy, e.g., by Kant, Hegel, and others [Gabbay and Woods, 2004].

These formalisations of logic in the 19th century lead to the idea that (symbolic) logic and mathematics are in fact the same thing. Even stronger, Frege (*Begriffschrift (1879)*) and Russell made themselves famous by insisting that mathematics can be expressed without relevant loss in a logical framework. This resulted in Whitehead and Russell's *Principia Mathematica (1910-1913)*, which formalises large parts of mathematics. So Leibniz' ideas seemed to become reality, as least for mathematics. However, the question was, asked by the mathematician David Hilbert (1928) how to mechanically discriminate between statements that are true or false, which is known as the *entscheidungsproblem* (decision problem). A solution of the this problem is some procedure that, given a logical sentence, performs a finite number of operations to determine its validity[1]. Gödel proved that the entscheidungsproblem cannot be solved by finding a finite number of postulates (axioms) that describe the mathematical

[1] *Das Entscheidungsproblem ist gelöst, wenn man ein Verfahren kennt das bei einem vorgelegten logischen Ausdruck durch endlich viele Operationen die Entscheidung über die Allgemeingültigkeit (...) erlaubt.* [Hilbert and Ackermann, 1928, p. 73], italics in the original.

truths, as these formal systems, besides very simple ones, are necessarily incomplete, i.e., some truths then remain unprovable. In 1936, the problem was proved to be unsolvable by Church [Church, 1936] and Turing [Turing, 1936], i.e., there is no such procedure for arithmetic if one adopts either of the two (equivalent) notions of computability proposed by Church and Turing. Even though the highly general problem of answering any mathematical question is unsolvable, there are all kinds of procedures imaginable that are capable of solving specific classes of problems, even in mathematics. And if it is true that the mental function can be described as a machine, as Turing believed [Turing, 1950], reasoning can be mechanised up to the intelligence of humans or at least to the level of intelligence indistinguishable from humans.

The further development of digital computers brought about new and more specific questions that required such symbolic reasoning, for example, as Turing did in 1949: "How can one check a routine in the sense of making sure that it is right?" [Turing, 1949]. After Goldstine and von Neumann [Goldstine and von Neumann, 1947] a few years earlier, he was one of the first to ask such questions, though it would take another 20 years before these first ideas would be made somewhat practical [Morris and Jones, 1984].

In the 1950s a new discipline emerged, for which the term artificial intelligence (AI) was introduced, that aimed at developing computerised intelligent systems. In 1956, Newell, Shaw, and Simon introduced one of the first AI program (The Logic Theorist) that was capable of proving theorems of the Principia Mathematica, which showed the feasibility of such an approach. Another important influence in this time was system theory that aimed at describing systems (which could be living organism, an organisation, a computer, etc.) not by its internal structure, but through the mathematical laws which govern its observable behaviour. It consists of several sub-disciplines, such as control theory which describes systems by differential equations. Another building-block of this field is the theory of finite-state machines [Gill, 1962], which was increasingly more important due to the widening use of digital computers. At the end of the 1950s and early 1960s, the scientific discipline of computer science was established, and finite-state machines was a well-known theory for describing computer systems. However, it was considered by some far too impracticable to reason about these systems mechanically using this theory. For example, McCarthy writes in 1962 that [McCarthy, 1962]:

> Much of the work on the theory of finite automata has been motivated by the hope of applying it to computation. I think this hope is mostly in vain because the fact of finiteness is used to show that the automaton will eventually repeat a state. However, anyone who waits for an IBM 7090 to repeat a state, solely because it is a finite automaton, is in for a very long wait.

Rather than the low-level view on computation that finite-state machines provide, McCarthy proposed to formalise computation in terms of more abstract terms that were found in the programming languages ALGOL 60 and LISP.

[McCarthy, 1962, McCarthy, 1963], so that one could use logic to reason about programs. This resulted in a theorem prover for a LISP-like logic, the Boyer-Moore theorem prover [Boyer and Moore, 1975, Boyer et al., 1995]. A breakthrough in program verification came at the end of the 1960s, when Floyd, Hoare and others developed the theory now known as Floyd-Hoare Logic [Floyd, 1967, Hoare, 1969] that allows program verification by inserting logical assertions in a program. Hence, a program could then be seen as a special logical theory that one can reason about. This again gave an abstraction of what a program *does*, to a more manageable description, in this case, what can be *proved* about a program, which is sometimes called a *deductive* approach. On the basis of this work, many systems were developed, such as automated verification systems (e.g., [King, 1969, Good, 1970]).

In order to make intelligent systems that are capable of advising users, AI researchers became interested in (human) knowledge representation. They had to be efficient to reason about, sufficiently expressive to encode relevant knowledge, and inference should not only be symbolic, but conclusions that are drawn should also be justified and understandable to the user. In particular, in the 1970s, new knowledge representation formalism were developed, such as rules-based formalisms and the frame formalism [Minsky, 1975]. Systems using such formalisms were called knowledge-based systems or expert systems, of which early examples were mostly concerned with medical diagnosis. For example, MYCIN [Shortliffe, 1976] aimed at advising about diagnosis and treatment of a number of infectious diseases and is the one of the most famous expert systems.

In the area of program verification, the deductive approach of Floyd-Hoare logic and related approaches such as dynamic logic [Pratt, 1976] and predicate transformers [Dijkstra, 1975] were successful on sequential programs; however, it was very difficult to apply to programs that contain concurrency. Work was done in order to verify specific classes of properties [Ashcroft, 1975, Owicki and Gries, 1976], such as deadlock freedom and data integrity, but few suggestions were put forward with respect to other properties dealing with issues such as termination, responsiveness, and liveness [Pnuelli, 1981]. As a consequence, new approaches were required. One direction was formed by algebraic approaches, most notably the calculus of concurrent systems (CCS) [Milner, 1980] to describe processes and reason about these processes explicitly. Since then, many more of these process calculi have been developed, such as the algebra of communicating processes (ACP) [Bergstra and Klop, 1987] and more recently the π-calculus [Milner, 1999]. Further development in this direction is described in [Baeten, 2004].

At around the same time as Milner introduced his CCS, Pnueli proposed to give programs a temporal semantics such that temporal logic can be used to reason about them [Pnueli, 1977, Pnuelli, 1981]. Many of the properties that were difficult to express in the deductive approaches for concurrent programs are naturally described in temporal logic, which takes into account the operational nature of programs. However, reasoning using temporal logic was still

tedious and a good deal of ingenuity was required to construct proofs such that they can be managed. Not much later, Clarke and Emerson proposed to model a concurrent program as a finite transition system, which can be interpreted as a model of temporal logic. Verification then involves checking that the concurrent program is a model of the property that one is interested in, a procedure they called model checking [Clarke et al., 1986, Queille and Sifakis, 1982]. This made it possible verify concurrent programs automatically by an exhaustive search of the state space "when the number of global states is not excessive (i.e., not more than a few thousand)" [Clarke et al., 1986]. In some sense, the early work of the 1950s in finite state transition systems had become modern again, as the computers that were available at the beginning of the 1980s had much more computing power than the machine McCarthy referred to, and the explicit reasoning about finite state transition system had become feasible for relatively small systems.

In AI, logic had also gained popularity as a knowledge representation formalism. Even though, compared to other knowledge representation mechanisms, they were computationally hard to reason with, a logical formalisation helps to understand reasoning. Therefore, logical formalisms can act as a solid base for implementing solutions. For example, temporal logics were used in AI as one of the candidates for formalisation of reasoning about time, action, and change [Cohen and Levesque, 1990].

Model checking has further improved significantly since then. McMillan [McMillan, 1993, Burch et al., 1990] discovered that the state transition system could be represented much more succinctly using a different symbolic representation based on ordered binary decision diagrams [Bryant, 1986], so that systems up to 10^{20} states could be model checked. Furthermore, support for modularising [Grumberg and Long, 1994, Clarke et al., 1989] and abstracting [Clarke et al., 1994, Dams et al., 1993, Wolper, 1986] systems has been added. Also, in AI, model checking has been used, e.g., in order to verify models of reasoning actors (multi-agent systems) (e.g., [Benerecetti et al., 1998]).

The success of model checking systems of the last years has reduced the use of theorem proving as an alternative for the verification of software and hardware systems, but cannot be avoided when the number of states of the system is beyond the capabilities of model checkers. This is the case when, for example, infinite data types are modelled or when one is interested in finding out constraints on parameters that ensure correctness of a system. A notable success in verification of hardware is an application of the ACL2 system, a successor of the aforementioned Boyer-Moore theorem prover, which was used to prove correctness of the AMD5K86 microprocessor. Moreover, theorem proving has had numerous applications in mathematics and in AI. Many heuristics have been developed in order to reason efficiently, such as resolution [Robinson, 1965b]. For example, the automated theorem proving system EQP proved a long standing conjecture of Robbins in the area of Boolean algebra in 1993 [McCune, 1997]. In AI, theorem provers have for example been used to verify knowledge-based systems (e.g., [Fensel et al., 1996]), as the knowledge

representation is often based on logic. Because of a clear separation between the knowledge base and a reasoning or problem solving method, the domain knowledge for example, can be validated and verified independently from the implementation.

For a long time, the tools that were available were difficult to use. As sketched in [Vaandrager, 2006], the situation is now improving, as many of the tools that are available have drastically improved in terms of scalability, accessibility, convenience, and realisability. Even with a limited amount of knowledge about the underlying techniques, model checking can be used. Moreover, they are being integrated with existing software practice, e.g., by model checking UML models [Knapp et al., 2002]. The developments of the last decade have made it possible to now apply verification techniques industrially. As an illustration of this, some of these application are discussed in the next section.

1.2 Applications of Formal Methods

It would be unwise to apply formal methods to all system development. The successful application of these techniques depend on a number of conditions, such as the *efficacy* of applying formal methods. As the costs of using formal methods are typically quite high, a balance has to be made between the these costs and their benefits. In hardware design, formal methods are well-established as errors can lead to significant economic losses. In software development, formal methods are generally applied to safety-critical and security-critical systems.

A second issue is that the application of formal methods requires appropriate *people support*. Though many of the formal methods aim at being fully automatic, for a number of techniques formal methods experts are required. As we will see in this thesis, it is often useful to abstract away from details of a system in order to get a useful specification. As it is not always straightforward to define what the unneeded details are, human intelligence is required. Furthermore, techniques such as theorem proving may be too computationally complex to be done by a computer fully automatically.

An example of a safety-critical system is the movable storm surge barrier, called the Maeslant Kering, that protects the low, western part of the Netherlands. The aim of the dams protecting these areas is to make sure that flooding does not occur more than once every 10,000 years, which is a failure probability that cannot be reached if the barrier is controlled manually. Therefore, it is controlled by a fully automatic system containing 450.000 lines of (C++) code of which 200.000 lines are operational code; the rest deals with simulating and testing. The software has been operational since 1997 and has been verified using formal methods [Chaudron et al., 1999, Tretmans et al., 2001]. The correct operation of the barrier is of great importance. If it does not close in time, then this could result in the flooding of Rotterdam, as large parts

of Rotterdam are situated below sea level. If, on the other hand, it does not open in time, this could damage the barrier as well as result in great economic losses, as Rotterdam's harbour cannot be reached when the barrier is closed.

In order to, for example, prevent fraud, many systems contain security components. Even when cryptographic algorithms are strong, as many of them are, a security applications may be broken if the algorithm is used improperly, i.e., if the protocol in which it is used does not adequately protect the secret. In these cases, formal methods can be useful. For example, a security-critical system that has been verified using formal methods is the Universal Electronic Payment System (UEPS) [Anderson, 1999]. This is an electronic funds transfer product well-suited for developing country environments. This protocol has been analysed using a logic, BAN logic [Burrows et al., 1989], designed for reasoning about security protocols. Other approaches for performing such analysis exists, such as strand spaces [Fabrega et al., 1998] or by the use of epistemic logic [Hommersom et al., 2004b].

1.3 Clinical Guidelines

A promising and challenging application area for the application of formal methods is clinical decision making, as is vital that clinical decisions are sound. In fact, ensuring safety is the primary preoccupation of medical regulatory agencies. Nonetheless, mistakes are made in hospitals: it was estimated that every year, in the Netherlands, 30,000 people are harmed and 1,700 people die in hospitals due to causes that can be avoided [Bruijne et al., 2007]. As medical errors may have such far reaching consequences for patients, there is good reason to be extremely careful.

Clinical guidelines are documents that include recommendations, advice, and management instructions aimed at supporting the decision-making in healthcare. For example, after a patient is diagnosed with hypertension, it is recommended to measure the blood pressure regularly, to inform the patients of risk factors, to prescribe appropriate medication, etc. Technology that is based on the recommendations given by the guideline, for example decision-support systems, should make sure that these decisions are sound with respect to the guideline. Therefore, technologies that are being developed aim to ensure that these technologies are designed to be safe [Fox and Das, 2000] by, for example, ensuring there is proper facilitation for simulation of new technologies, so that harmful side effects or adverse consequences of use are kept to an absolute minimum.

Another question is if the recommendations given by the guideline are correct or to which extent they are correct. Medicine is getting increasingly more complicated and specialised, so problems may occur. Even though the use of clinical guidelines in practice is still growing, problems concerning guidelines are not of academic nature. For example, in [Boyd et al., 2005], it was found that adhering to current clinical guidelines in caring for an older person with

several comorbidities may have undesirable effects. The researchers note that the focus of clinical guidelines to certain aspects of a disease "could create perverse incentives that emphasise the wrong aspects of care for this population and diminish the quality of their care". Furthermore, the researchers concluded that if the relevant guidelines were followed, the hypothetical patient would be prescribed 12 medications and a complicated non-pharmacological regimen. Adverse interactions between drugs and diseases could result.

The problems mentioned in this study are due to the complexity of dealing with several comorbidities; however, there are other possible harms of guidelines [Woolf et al., 1999]. First, scientific evidence may be interpreted incorrectly, be misleading, or be simply absent. Second, recommendations could be constructed on the basis of personal views and misconceptions that are not consistent with the available evidence. Third, recommendations could be suboptimal from a *patient perspective* due to conflicting interests such as cutting costs or protecting the interests of others, for example, doctors or politicians. Formal methods have the potential to make assumptions and design decisions that justify the construction of the recommendations more explicit. This makes it possible to inspect whether or not the evidence has been interpreted well and whether recommendations have been based upon sound medical principles, rather than built on personal views.

1.4 Guideline Representation

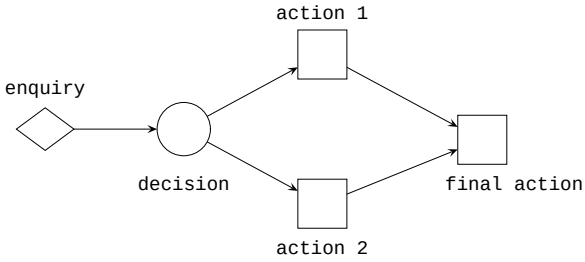
Although guidelines are documents in natural language, several guideline representation languages have been developed in the past decade to represent essential fragments of guidelines in a computer-interpretable fashion. An older, but still influential modelling language based on rules is the *Arden Syntax* [Arden Syntax Technical Committee of HL7, 1999, Hripcsak et al., 1994]. An example of an Arden Syntax fragment from a guideline which advises a physician on impaired kidney function based on the levels of creatine and blood urea nitrogen (BUN) levels is the following rule[2]:

if creatine_level $>$ 1.5 or BUN $>$ 30
then alert_text := "Consider impaired kidney function (...)";

The temporal relations between actions that have to be performed is typically not made explicit in such rule formalisms and is determined by means of an inference mechanism, which establishes the order in which the rules are processed. In order to explicitly formalise complex decisions and care pathways, more expressive formalisms were developed such as EON [Tu and Musen, 1999], PRO*forma* [Fox and Das, 2000], GLIF3 [Peleg et al., 2000], and Asbru [Shahar et al., 1998]. These languages have been named "task-based networks" in [Peleg et al., 2003], as they consist of

[2] Based on `http://www.dbmi.columbia.edu/homepages/wandong/KR/krarden.html`, accessed August 1, 2007.

a number of tasks with temporal and other relations between them. This is accomplished by defining tasks and sub-tasks that may be executed in a certain order. The main difference with rule-based languages is that they allow one to explicitly model the control flow. For example, in PRO*forma* this is represented graphically as:



where each of the nodes represent some "task" and the arrows between them form a partial order, which represent that a successor task can be started when the current task is finished. Additional constructs exist to refine such temporal relations, for example, starting of a task may be delayed by pre-conditions that refer to a specific time after a previous task was completed or conditioned on the result of a decision. Each of the tasks is further parameterised with elements such as conditions, triggers, cycles, etc. How these languages relate to the more well-known workflow systems, which contain similar primitives, is an interesting question, which the guideline and workflow communities have started to address recently [Mulyar et al., 2007, Fox et al., 2008].

Those languages are typically very expressive and can be considered as providing the current paradigm for modelling clinical guidelines. However, the suitability of these languages for the use of verification of clinical guidelines is questionable. A crucial problem is that only a few of those languages have a formal semantics. The exceptions are PRO*forma* [Sutton and Fox, 2003, Fox and Das, 2000] and a subset of Asbru [Balser et al., 2002a, Balser et al., 2006] (sometimes referred to as Asbru Light), for which operational semantics exist. A more fundamental point in the underlying assumption of some of those languages is that a guideline describes a series of tasks that have to be executed in a certain order to the extent that it includes every detail of clinical practice. This is of course something one wishes to achieve if the goal is to use these models in decision support systems that have to cover the whole clinical pathway.

In this thesis, we will look at these guidelines somewhat differently. Instead of seeing guidelines as a rather complete description of medical practice, we look at guidelines as only putting constraints on this practice. This is justified if one considers the type of knowledge that is present in the guideline (cf. Chapter 3). Moreover, other types of knowledge, such as knowledge concerning the patient, pathophysiological processes, and common-sense medical management is something which also has to be taken into account if one aims

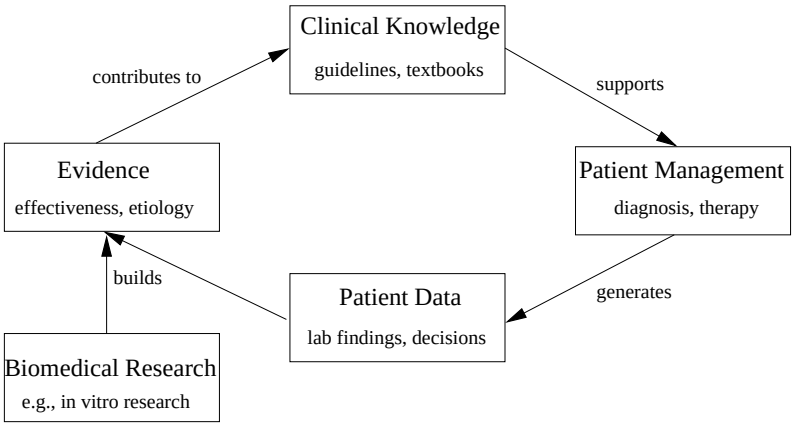


Figure 1.1: Cyclic development in healthcare, based on introductory chapter of [van Bemmel and Musen, 2002] and [Wyatt, 2002].

at verifying relevant properties. In this thesis, this knowledge is referred to as *background knowledge*. As such knowledge cannot be specified in the aforementioned guideline representation languages, they fail to be sufficiently complete for the type of verification that is proposed and studied in this thesis.

1.5 Formal Methods and Clinical Guidelines

Nowadays, computer science plays an important role in the healthcare process. As an illustration, Figure 1.1 provides an overview of the development of healthcare. Starting from the right, the patient management generates data, which is stored in some way. The analysis of this data, together with results obtained from biomedical research, yields evidential knowledge, which is exploited to formulate clinically useful knowledge, for example in the form of a guideline. The clinical knowledge is then again employed for the management of patients. In all four corners, computer science plays a role. For example, decision support systems are used to support the patient management process on the basis of clinical knowledge, electronic patient records are used to store the data that is being generated by the management, data mining techniques are used to analyse the data, and guideline authoring software supports the construction of clinical knowledge, i.e., electronic guidelines. Systems designed in this cycle could be verified using formal methods. For example, the control program for a radiation therapy machine at the University of Washington has been developed using formal methods [Bowen and Hinchey, 1997], as incorrect decisions of the machine may lead to harmful situations. If there is too little radiation, the tumour will not be eliminated, whereas when there is too much radiation, the treatment will lead to unnecessary side-effects.

In this thesis, we apply formal methods to clinical knowledge as described in clinical guidelines. Early work in verification of clinical guidelines mostly dealt with checking that the guideline is unambiguous, complete, and consistent, for example by representing guidelines by a decision table [Shiffman and Greenes, 1994, Shiffman, 1997] or as a set of rules [Miller et al., 1999]. In [Duftschmid and Miksch, 1999, Duftschmid et al., 1998], Asbru models are translated to first-order logic and rich structural properties are investigated in order to check the coherence of the model, for example, by a reachability analysis of each of the specified actions in the model. Similar to the structure, the coherence of temporal constraints that have been put on the (Asbru) model is discussed in [Duftschmid et al., 2002]. If problems with coherence can be traced back to the guideline, this can be considered a form of verification. However, this technique is mostly aimed at validating the formal model rather than checking the correctness of the original guideline.

A major influence on the use of formal methods for the verification of clinical guidelines came from the European Protocure project[3], which funded part of the research in this thesis. The premise of this project was that, since the task-based models resemble the structure of software programs to a large extent, verification can take place, as if they are a special type of programs, i.e., using program verification techniques. The Asbru language was taken as a starting point and translations were made to a suitable language for a theorem prover, based on algebra and logic. The left part of Figure 1.2 describes an overview of this approach. First, formalisation of the guideline and properties takes place by modelling the guideline in Asbru, after which this can be translated to the logic of the theorem prover KIV [Balser et al., 2000], designed for the verification of software. If done correctly, proving the formal properties with respect to the formal model ensures that the informal guideline satisfies the original properties.

This thesis addresses a number of issues surrounding this process. The first question is the following:

> *Which knowledge is required to investigate properties of clinical guidelines?*

In using formal methods for the verification of systems, relevant properties are often derived from the requirements used during the development of software. This raises the question what constitutes the requirements in the design of guidelines. From Figure 1.1, it is clear that the content of guidelines is determined by available medical evidence as well as by ideas about the proper management of disease in patients, which is "background knowledge" with respect to the guideline. However, guideline recommendations must also be closely connected to available underlying knowledge about physiological and clinical processes, otherwise they cannot be effective. As a result, the quality

[3] See `http://www.protocure.org`

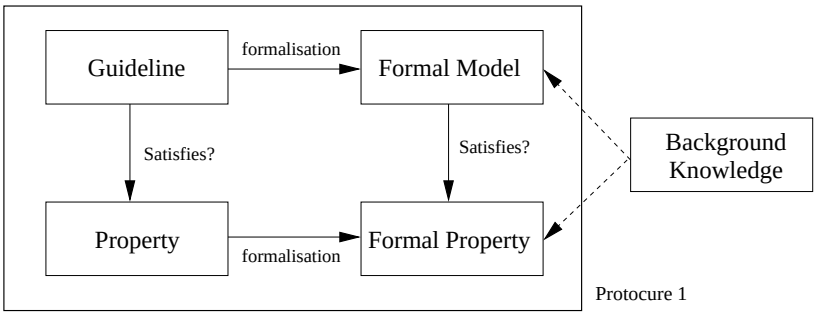


Figure 1.2: In order to find out whether a property is satisfied by the guideline, a formal property is checked on the formal model of the guideline. Additional background knowledge can be used to improve this idea.

of guidelines must be considered in this light as well (see Figure 1.2). Thus, in order to apply formal methods effectively, all relevant knowledge must be identified. In Chapter 3, various types of knowledge are identified by means of a number of examples taken from a real-world guideline. This thesis then focuses on some of these aspects that are considered more relevant, and these are subsequently formalised and verified or checked in Chapters 4, 5, and 6. The actual verification shows that this allows one to answer relevant questions with respect to an actual guideline. In order to use symbolic reasoning techniques, this knowledge has to be represented well, resulting in the next question:

> *What is a suitable language for representing clinical guidelines in order to apply formal methods?*

As mentioned before in this chapter, guideline representation languages do not allow formalising background knowledge in a suitable fashion. Furthermore, questions were raised about the appropriateness of using only the ideas of tasks and task decompositions for the modelling of guidelines. Besides these issues, an important step in using formal methods is choosing the right level of abstraction of the system that one is interested in. This also puts additional restrictions on a suitable knowledge representation formalism. This thesis provides evidence that a logical approach is feasible and has several of the desirable aspects of a guideline formalisation language. However, a suitable language raises the question how reasoning should commence in order to derive useful results. This could be summarised into the following question:

> *How do formal methods contribute to ensuring the quality of guidelines?*

Previous work, as outlined in Section 1.5 has given a partial answer to this question. However, the additional approaches with associated techniques that

are considered in this thesis have the potential to provide a more complete answer. From an AI perspective, issues such as understandability of the results are important and representation and inference should take this into account. On the other hand, this has something to do with the type of reasoning that is involved and how such reasoning can be mapped to specific systems. The results of the experiments reported in this thesis show that appropriate reasoning about guidelines is amenable to the verification techniques that are available. As one can choose from several verification techniques the question then becomes:

> *Which techniques are most suitable for answering questions about clinical guidelines?*

In this thesis, a number of techniques are explored, such as interactive and automated theorem proving and model checking. As discussed in this introductory chapter, verification serves several purposes. As a result, it is also of interest to learn more about which technique is most appropriate in a given situation. For example, as guidelines are considered the golden standard of medical care, they could also be taken as the requirements that other systems should comply to, e.g., in the development of local protocols, or the patient management itself. Formal methods can then be used to verify whether these systems comply with the guideline. This is a topic that is investigated in this thesis as well.

1.6 Overview of this Thesis

This thesis deals with different facets of formal methods in relation to clinical guidelines, i.e., requirements, specification, and verification.

Chapter 2 offers the necessary preliminaries with respect to formal methods. In particular, we will focus on the use of temporal logics and associated techniques, in particular theorem proving and model checking. This chapter more or less sets the scene for most of the technical work underlying this thesis.

Chapter 3 introduces the concepts of clinical guidelines and protocols and offers an analysis of the current guideline-development process. As formal methods are employed in the development of systems, it is crucial to understand how a system is developed and what its requirements are. Case studies are introduced to illustrate this; the case studies are used in subsequent chapters.

In **Chapter 4** we investigate ways to verify clinical guidelines using automated theorem proving techniques. It appears that important requirements about the quality of clinical guidelines can be represented by schemata borrowed from the theory of abductive diagnosis, using temporal logic to model

the time-oriented aspects expressed in a guideline. It is investigated how this approach can be mapped to the facilities of a resolution-based theorem prover, OTTER, and a complementary program that searches for finite models of first-order statements, MACE-2. It is shown that the reasoning required for checking the quality of a guideline can be mapped to such fully automated theorem-proving facilities. Parts of this chapter were published in [Hommersom et al., 2005a].

Chapter 5 takes a similar approach as Chapter 4. Clinical guidelines are now represented by a standard guideline representation language, Asbru. In this chapter, it is proposed to include medical background knowledge into such task networks. It is investigated how the reasoning as introduced in Chapter 4 can be applied to this setting and be mechanised using an interactive theorem prover. The yields insight into the question regarding the appropriateness of techniques for the verification of guidelines. Parts of this chapter were published in [Hommersom et al., 2004a, Hommersom et al., 2007].

Chapter 6 is concerned with verification using the guideline as a gold standard. More specifically, we look at the problem of making sure that adaptations of guidelines for particular purposes are of sufficiently good quality. Furthermore, we investigate how to ensure by means of the use of formal methods that clinical decisions made by medical doctors are sound with respect to the guideline. In order to do this automatically, a model checking technique is investigated in this application area. Parts of this chapter were published in [Hommersom et al., 2006, Groot et al., 2007].

Chapter 7 deals with more fundamental issues of specifying clinical guidelines in logic, and in particular we consider several loose ends of the research not touched upon in the earlier chapters. Although logical methods are not commonly used in medical research about clinical guidelines, the results of the earlier chapters suggest that adopting a logical view allows one to look at guidelines in a more critical fashion. In this chapter we study two aspects of guidelines, the time-oriented decision making underlying the structure of guidelines and the clinical consideration of choosing alternative actions based on the results obtained by previous actions, more thoroughly. The resulting insights from these studies could help in designing a special-purpose clinical guideline logic. Parts of this chapter were published in [Hommersom et al., 2005b, Hommersom and Lucas, 2007].

Finally, in **Chapter 8** we bring the results of the research described in this thesis into perspective and consider the questions that drove the research once again, but now in light of what we have achieved. Also, some future lines of research that can be seen as natural follow-ups to the research described in this thesis are discussed.

Chapter 2

Preliminaries

In this chapter, preliminaries with respect to the languages and techniques that are used throughout this thesis are introduced. As guidelines are temporal systems, the main type of logic that we use is temporal logic, which is the main topic of this chapter. Necessary concepts with respect to reasoning and techniques that are employed for formal verification are introduced. Furthermore, related work with respect to formal systems in biomedicine is mentioned. Finally, it is summarised how these languages and techniques will be used in the subsequent chapters.

Familiarity with propositional and first-order logic is assumed throughout this chapter. Furthermore, we assume the reader is familiar with a few basic notions related to automata and complexity theory.

2.1 Temporal Logic

Temporal logic refers to a class of logics for reasoning about time. An important subclass of these logics is tense logic, which originally dealt with reasoning about the past, present, and future. Tense logic was first introduced in 1957 by Arthur Prior [Prior, 1957] as a branch of modal logic, which are logics that deal with relational structures. Since then many variants have been developed, for example, for use in computer science. In this chapter, we do not give a complete overview of all of these logics. Instead, we focus on those the logics that are used in subsequent chapters of this thesis.

This chapter is organised as follows. We first introduce the semantics of temporal logics. On the basis of these models, we will define a number of logics that are used, starting with future-time temporal logics as they play such an important role in computer science for verifying digital systems. After this, we briefly discuss a comparison, in terms of expressiveness, between the logics that have been introduced.

2.1.1 Temporal Model

As said, the meaning of modal logics can be described in terms of relational structures. A general model of such a relational structure is a Kripke structure. Such a structure, over a set of atomic propositions $\mathcal{P}$, is formally defined as a three tuple $M = \langle S, R, L \rangle$ such that [Clarke et al., 2001]:

- S is a set of states;
- $R \subseteq S \times S$ is a binary relation;
- $L : S \rightarrow 2^{\mathcal{P}}$ is a function that labels each state with the set of atomic propositions true in that state.

In the context of temporal logic, R defines a transition relation. A *path* in the model M from a state $s_0 \in S$ is then a (non-empty) sequence $\pi = s_0 s_1 s_2 \ldots$ such that $R(s_i, s_{i+1})$ holds for all $i \geq 0$. With π^i we denote the suffix of π starting at s_i and with $\pi^{[i,j]}$ we denote the sub-path of π from s_i to (and including) s_j with $i \leq j$. Finally, we denote $|\pi|$ for one less than the length of the sequence, i.e., the number of transitions of the path, which is either ∞ if the sequence is infinite (i.e., the sequence does not have a last element) or some natural number.

In practice, several assumptions are made with respect to the model. In programming literature, usually a set of initial states $S_0 \subseteq S$ is defined. Otherwise, each $s \in S$ can be considered an initial state. A second assumption that is often made is that S is a finite set. Then, M could be seen as a non-deterministic finite-state machine, which can be used for modelling many digital systems. It is non-deterministic as from each state it could be possible to move to several successor states. Finiteness ensures that computing many types of properties of such a model is, in principle, possible. Finally, R is often assumed to be serial, i.e., for all s there exists s' such that $R(s, s')$. In this thesis, this assumption is not made, unless otherwise stated.

One might observe that to each state a propositional model (a truth table modelled by the function L) is associated which is used to describe propositional temporal logics. In order to acquire a first-order temporal logic instead, a first-order structure is associated with each state, usually with additional restrictions on the relationship between successor structures, e.g., that their underlying sets are equal. This poses many difficult philosophical and technical issues that go beyond the scope of this thesis. Relevant details will be mentioned in subsequent chapters if necessary. For a survey of such logics, we refer to [Cocchiarella, 2002].

2.1.2 Future-time Temporal Logics

The logics that we discuss here use atomic propositions and Boolean connectives (e.g., $\neg, \vee, \wedge$) to build up more complex expressions for describing properties of states. There are two types of modalities used, namely, *path quantifiers*

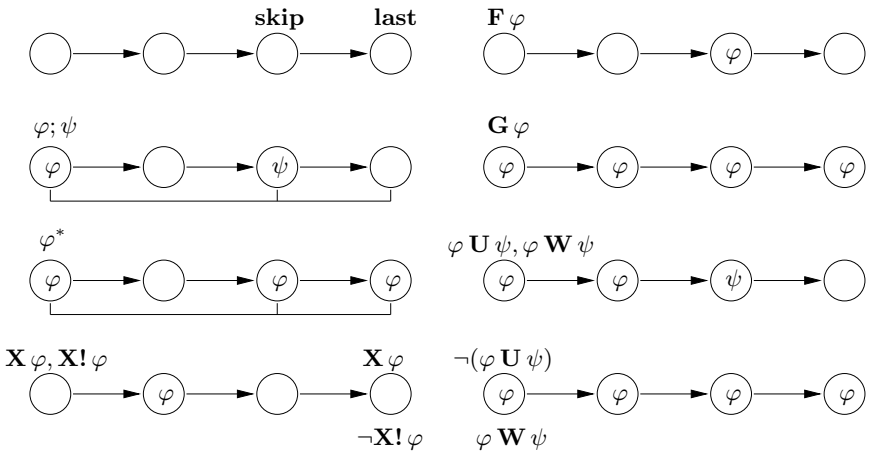


Figure 2.1: Some example sequences consisting of four states that provide interpretations for the temporal operators used, where absence of a formula in a state implies that the formula does not hold there. The bar under the sequence for the chop and star operator indicate that their sub-formulas are evaluated on these sub-intervals. For all other temporal operators, sub-formulas can refer to the complete sequence.

and *temporal operators*. The path quantifiers are **A** and **E** to specify that all or some of the paths, starting at a specific state, have some property, respectively. The temporal operators describe properties of a path through the model, some of which specify a property of the whole path and some split up the path into several sub-intervals. Temporal operators of the first type that are used are **X**, **X!**, **G**, **F**, **U**, and **W**, with $\mathbf{X}\varphi$ being true if φ holds in the next state if there is one, **X!** is true if $\mathbf{X}\varphi$ holds and there is indeed a next state, $\mathbf{G}\varphi$ if φ holds in the current state and all future states, $\mathbf{F}\varphi$ if φ holds in some state in the future (or is true in the current state), $\varphi\mathbf{U}\psi$ if φ holds until ψ holds, i.e., there is a state on the path where ψ holds and in every preceding state φ holds, and $\varphi\mathbf{W}\psi$ if φ holds along the path up to and including the first state where ψ holds, however ψ is not required to hold eventually. Three basic interval temporal operators are used here, namely **skip**, ';' (*chop*), and '$*$' (*star*), with **skip** being true exactly if the length of the interval is 1, $\varphi;\psi$ holds if the path can be "chopped" into two parts, such that φ holds in the first part and ψ holds in the second part, and φ^* holds if it is possible to decompose the interval into a (possibly infinite) number of finite intervals in which φ holds. Example interpretations of these operators on short sequences of states are illustrated in Figure 2.1.

The first logic we discuss is Computational Tree Logic (CTL). In CTL there are two types of formulas: *state formulas*, which are true in a specific state, and *path formulas*, which are true along a specific path. The syntax of state

and path formulas is defined as follows:

1. Each atomic proposition, denoted by $p, q, r, \ldots$ is a state formula.
2. If φ and ψ are state formulas, then $\neg\varphi$, $\varphi \vee \psi$, and $\varphi \wedge \psi$ are state formulas.
3. If φ is a path formula, then $\mathbf{E}\varphi$ and $\mathbf{A}\varphi$ are state formulas.
4. If φ and ψ are state formulas, then $\mathbf{X}\varphi$, $\mathbf{G}\varphi$, $\mathbf{F}\varphi$, $\varphi\mathbf{U}\psi$, and $\varphi\mathbf{W}\psi$ are path formulas.

CTL consists of the set of all state formulas. Note that in this definition, path quantifiers are always directly followed by a temporal operator, e.g., $\mathbf{E}\,\mathbf{G}\,p$. If we drop this restriction on the order of operators, i.e., by building up more complex expressions of path formulas using temporal and propositional operators, then the logic it generates is CTL* (see [Clarke et al., 2001] for more details). We can then, for example, express $\mathbf{A}(\mathbf{G}p \vee \mathbf{G}q)$, i.e., for all paths always p or always q holds, something which cannot be expressed in CTL. The semantics of CTL* (and the sub-logic CTL) is defined with respect to a Kripke structure M. Given a state formula φ, the notation $M, s \models \varphi$ denotes that φ holds in state s of the Kripke structure M. A formula is valid if for all models M and all initial states s of this M it holds $M, s \models \varphi$. Assuming that φ_1 and φ_2 are state formulas and ψ_1 and ψ_2 are path formulas, the relation $\models$ can be defined inductively as shown in Figure 2.2.

Linear Temporal Logic (LTL) is a logic which is built up by the temporal operators, i.e., all the formulas can be seen as path formulas, so the language is defined as follows:

1. Each atomic proposition is a formula.
2. If φ and ψ are formulas, then $\neg\varphi$, $\varphi \vee \psi$, and $\varphi \wedge \psi$ are formulas.
3. If φ and ψ are formulas, then $\mathbf{X}\varphi$, $\mathbf{G}\varphi$, $\mathbf{F}\varphi$, $\varphi\mathbf{U}\psi$, and $\varphi\mathbf{W}\psi$ are formulas.

The semantics is defined with respect to paths of a Kripke model. Given a formula φ, $M, \pi \models \varphi$ denotes that φ holds on path π of the Kripke structure M. Similar to CTL, a formula φ is valid if for models M, and all paths π starting from an initial state of M holds $M, \pi \models \varphi$. It is then obvious that an LTL formula φ is equivalent to the CTL* formula $\mathbf{A}\varphi$.

Many extensions have been proposed in this linear framework of logics, mainly by the addition of special and sometimes more powerful operators. One such proposal is Interval Temporal Logic (ITL), i.e., the logic that is defined by adding the interval temporal logical operators **skip**, ';', and '$*$', as we have introduced above to LTL, of which the last two cannot easily be defined in terms of the other operators.

The main difference between the linear and branching logics is that in the branching framework one can talk about uncertainty in time, e.g., one

$$
\begin{array}{lcl}
M, s \models p & \Leftrightarrow & p \in L(s) \\
M, s \models \neg\varphi & \Leftrightarrow & M, s \not\models \varphi \\
M, s \models \varphi_1 \vee \varphi_2 & \Leftrightarrow & M, s \models \varphi_1 \text{ or } M, s \models \varphi_2 \\
M, s \models \varphi_1 \wedge \varphi_2 & \Leftrightarrow & M, s \models \varphi_1 \text{ and } M, s \models \varphi_2 \\
M, s \models \mathbf{E}\,\psi & \Leftrightarrow & \text{there is a path } \pi \text{ from } s \text{ such that } M, \pi \models \psi \\
M, s \models \mathbf{A}\,\psi & \Leftrightarrow & \text{for every path } \pi \text{ starting from } s \text{ such that } M, \pi \models \psi \\
M, \pi \models \varphi & \Leftrightarrow & s \text{ is the first state of } \pi \text{ and } M, s \models \varphi \\
M, \pi \models \neg\psi & \Leftrightarrow & M, \pi \not\models \psi \\
M, \pi \models \psi_1 \vee \psi_2 & \Leftrightarrow & M, \pi \models \psi_1 \text{or } M, \pi \models \psi_2 \\
M, \pi \models \psi_1 \wedge \psi_2 & \Leftrightarrow & M, \pi \models \psi_1 \text{ and } M, s \models \psi_2 \\
M, \pi \models \mathbf{X!}\,\psi & \Leftrightarrow & |\pi| \neq 0 \text{ and } M, \pi^1 \models \psi \\
M, \pi \models \mathbf{X}\,\psi & \Leftrightarrow & |\pi| \neq 0 \text{ implies } M, \pi^1 \models \psi \\
M, \pi \models \mathbf{F}\,\psi & \Leftrightarrow & \text{there exists a } k \geq 0 \text{ such that } M, \pi^k \models \psi \\
M, \pi \models \mathbf{G}\,\psi & \Leftrightarrow & \text{for all } k \geq 0, M, \pi^k \models \psi \\
M, \pi \models \psi_1\,\mathbf{U}\,\psi_2 & \Leftrightarrow & \text{there exists a } k \geq 0 \text{ such that } M, \pi^k \models \psi_2 \text{ and} \\
 & & \text{for all } 0 \leq j < k, M, \pi^j \models \psi_1 \\
M, \pi \models \psi_1 \mathbf{W} \psi_2 & \Leftrightarrow & \text{there exists a } k \geq 0 \text{ such that } M, \pi^k \models \psi_1 \text{ and} \\
 & & \text{for all } 0 \leq j < k, M, \pi^j \models \psi_2 \\
 & & \text{or for all } k \geq 0, M, \pi^k \models \psi_2 \\
M, \pi \models \mathbf{skip} & \Leftrightarrow & |\pi| = 1 \\
M, \pi \models \psi_1; \psi_2 & \Leftrightarrow & |\pi| = \infty \text{ and } \pi \models \psi_1, \text{ or} \\
 & & \text{there exists } n \leq |\pi| \text{ with } \pi^{[0,n]} \models \psi_1 \text{ and } \pi^{[n,|\pi|]} \models \psi_2 \\
M, \pi \models \psi^* & \Leftrightarrow & |\pi| = 0 \\
 & & \text{or there exists } 0 = n_0 < n_1 < \ldots < n_m < |\pi| \\
 & & \quad \text{with } \pi^{[n_i, n_{i+1}]} \models \psi \text{ for all } 0 \leq i < m \\
 & & \quad \text{and } \pi^{[n_m, |\pi|]} \models \psi \\
 & & \text{or there exists infinite many } 0 = n_0 < n_1 < \ldots \\
 & & \quad \text{with } \pi^{[n_i, n_{i+1}]} \models \psi \text{ for all } 0 \leq i
\end{array}
$$

Figure 2.2: Semantics of CTL* with φ representing a state formula and ψ representing a path formula.

can say "a patient *may* recover". Note that you should not necessarily read this uncertainty epistemologically, as the world (or system) could behave non-deterministically, i.e., it is not necessarily a lack of knowledge that creates this uncertainty.

2.1.3 Past-time Temporal Logics

As future-time logics allows one to reason about the future, some tense logics also allows one to reason about the past. Originally, these logics were developed from philosophical interest, i.e., to investigate the relationship between tense and modality; however, it has since then been recognised in the area of artificial intelligence. In context of formal methods, these operators have not been used much as past-time operators are typically not part of program specifications. Furthermore, from a technical point of view, in LTL, these modalities can be omitted and properties can be rephrased as an equivalent future-only formula if one focuses on one initial state [Gabbay, 1987]. Nonetheless, past-time temporal logics are useful for knowledge representation as formulas using these additional operators may be more intuitive and are typically more succinct (in fact, exponentially more succinct for LTL [Markey, 2003]).

The semantics of such logics is again described by a Kripke model as shown above in Section 2.1.1. Modalities can be defined by considering the inverse relation of R of a Kripke model M and paths now have a past as well as a future, i.e., $\pi = \ldots, s_{-1}, s_0, s_1, \ldots$, so that, for example the $\mathbf{S}$ (*since*) operator can be defined (NB: as there is no unique initial state, the state is added as an additional parameter to the model):

$$M, \pi, s_0 \models \psi_1 \,\mathbf{S}\, \psi_2 \quad \Leftrightarrow \quad \begin{array}{l}\text{there exists a } k < 0 \text{ such that } M, \pi, s_k \models \psi_2 \text{ and} \\ \text{for all } k < j < 0, M, \pi, s_j \models \psi_1\end{array}$$

i.e., $\varphi \,\mathbf{S}\, \psi$ means somewhere in the past ψ held and after that until now φ held. Similar to the future-time modalities, linear modalities for the history can be introduced (which is often called Ockhamist past, after William of Ockham [Zanardo and Carmo, 1993]), yielding modalities such as $\mathbf{S}$ (since), $\mathbf{X}^{-1}$ (one moment ago), $\mathbf{P}$ (in the past), $\mathbf{H}$ (always in the past), etc. Branching variants of past-time modalities are less common, e.g., "yesterday, the patient may have been given a treatment" cannot be expressed as it either *did* or *did not* happen. Of course, there can be uncertainty whether the patient has been given the treatment; however, again, standard temporal logic is not concerned with epistemology. For such purposes other logics have been suggested, such as alternating-time temporal logic (ATL) [Alur et al., 1998]. It is possible to combine a branching logic with a linear past (PCTL), which has also been proposed [Laroussinie and Schnoebelen, 2000]. While for a subset of this logic there exists a translation to CTL, the PCTL specification can be more natural and succinct.

$\mathbf{X!}\,\varphi$	$\equiv$	$\mathbf{skip};\varphi$	in the next state φ
$\mathbf{X}\,\varphi$	$\equiv$	$\neg\mathbf{X!}\,\neg\varphi$	if there is a next state, then in the next state φ
last	$\equiv$	$\mathbf{X}\,\bot$	this is the last state of the interval
finite	$\equiv$	$\neg(\top;\bot)$	the interval is finite
$\mathbf{F}\,\varphi$	$\equiv$	$\top\,\mathbf{U}\,\varphi$	eventually φ
$\mathbf{F}\,\varphi$	$\equiv$	$\mathbf{finite};\varphi$	eventually φ
$\mathbf{G}\,\varphi$	$\equiv$	$\neg\mathbf{F}\,\neg\varphi$	always φ
$\mathbf{P}\,\varphi$	$\equiv$	$\top\,\mathbf{S}\,\varphi$	somewhere in the past φ
$\mathbf{H}\,\varphi$	$\equiv$	$\neg\,\mathbf{P}\,\neg\varphi$	always in the past φ

Figure 2.3: Some example definition of temporal modalities in terms of other modalities.

2.1.4 Expressiveness

With respect to expressivity of these logics, many results are known. For example, Kamp [Kamp, 1968] has shown that the linear temporal logic with the **S** and **U** modal operator is *expressively complete* for linear continuous time lines, i.e., every possible temporal operator can be define by this language. Other first-order theories, such as the ones using the successor function requires for example the **X** modality. For a more in-depth study with respect to completeness results of various temporal logics, we refer to [Gabbay et al., 1994, Goldblatt, 1987, Venema, 2001]. Minimal sets of modalities can be acquired by omitting the modalities that can be defined in terms of other modalities, similar to the usual propositional abbreviations such as $\top \equiv \varphi \vee \neg\varphi$, $\bot = \neg\top$, $\varphi \wedge \psi \equiv \neg(\neg\varphi \vee \neg\psi)$, etc. In Figure 2.3, we provide an (incomplete) set of such equivalences.

2.2 Reasoning

In order to reason with the logics introduced above, a distinction is made between the object and meta language. The object language is the language that is "talked about", whereas the meta-language is the language in which we "talk about" the object language. This distinction was made clear by Tarski in his work to formalise the concept of truth [Tarski, 1944], where he used an appropriate meta-language for defining truth in the object level. Similar to the distinction between object-language and meta-language, facts that cannot be derived from within a theory, but that can be asserted about the theory are called meta-knowledge. In order to discuss some of the reasoning that is employed, we discuss object-level reasoning and meta-level reasoning and introduce their notation. This discussion is required for introducing the type of reasoning that is employed in Chapters 4 and 5.

2.2.1 Object-level Reasoning

In the previous section, we have used the $\models$ symbol in order to denote what a formula means in a specific model. Throughout this thesis, we will also use the $\models$ symbol in order to denote entailment, i.e., $\varphi \models \psi$ means that in all models where φ holds, also ψ holds. This is a slight abuse of notation as φ is a sentence in the object language, whereas M is described in some meta-language. In order to avoid reasoning with models, axioms and inference rules are defined that preserve truth according to the semantics. In order to describe logical consequences on the basis of such axioms and inference rules, we write $\varphi \vdash \psi$, where $\vdash$ is defined according to a set of inference rules, i.e., a proof procedure. As a convenience, for sets of formulas Γ and Δ, we write $\Gamma \vdash \Delta$ and $\Gamma \models \Delta$ as shorthand for $\bigwedge \Gamma \vdash \bigvee \Delta$ and $\bigwedge \Gamma \models \bigvee \Delta$, respectively.

A proof procedure is called sound if for all formulas φ holds that if $\vdash \varphi$ then $\models \varphi$, i.e., everything that can be derived is true. Conversely, we call the proof procedure complete if for all φ it holds that if $\models \varphi$ then $\vdash \varphi$, i.e., everything that is true can be derived.

For first order logics, a plethora of proof systems have been developed (e.g., natural deduction, sequent calculi, semantic tableaux, etc.). In general, there is a trade-off between the amount of axioms and the amount of inference rules. Systems with many axioms and little inference rules (typically just modus ponens) are called Hilbert-style systems. Such axiomatisations for temporal logic have been studied extensively, for example:

$$\mathbf{F}\,\mathbf{F}\,\varphi \rightarrow \mathbf{F}\,\varphi$$

is an axiom of linear temporal logic. A great amount of sound and complete axiomatisations, i.e., a set of axioms that allows for the derivation of all truths and no more, have been developed. A systematic overview of these axiomatisations is given in [Kuhn and Portner, 2002].

Such axiomatisations are interesting from a theoretical perspective; however, they are inefficient for computerised reasoning. In Chapter 4, resolution [Robinson, 1965b] is used, which is based on exactly one inference rule:

$$\frac{p \vee \psi \qquad \neg p \vee \phi}{\psi \vee \phi}$$

and a procedure (*unification*), which replaces variables by terms, in order to obtain complementary literals which are then used in the resolution rule. The resolution procedure is sound and refutation-complete, where refutation-completeness means that if $\Gamma \models \bot$ (note that an empty disjunction equals $\bot$), then $\Gamma \vdash \bot$ [Robinson, 1965b]. In practice, this does not pose a restriction on the applicability of resolution to solve problems as in order to establish that $\Gamma \models \varphi$, it suffices to show that $\Gamma \cup \{\neg\varphi\} \models \bot$. Similar to first-order logic, inference rules have been proposed for temporal logic, such as temporal resolution

[Fisher et al., 2001] and verification diagrams [Manna and Pnueli, 1994].

2.2.2 Meta-level Reasoning

Meta-knowledge and meta-reasoning are important topics for intelligent systems for obtaining better results. For example, as in mathematics, meta-knowledge can be used to better understand some system under investigation by reasoning about it on a meta-level. More specific to artificial intelligence is that meta-knowledge can be used as heuristics for automated deduction systems and inference control in problem solving (e.g., [Bundy, 1988]). The idea there is that the search space is much smaller on a meta-level and results found there can help finding solutions on the object-level. Finally, as meta-theorems often cannot be represented appropriately in the object language, describing reasoning on a meta-level can be useful for modelling for various systems. An example of the latter are argumentation frameworks [Dung, 1995], where arguments are constructed in a sufficiently expressive object language and are reasoned about in the meta-language, which deals with notions such as defeat and aggregation of arguments. A second example of meta reasoning, which is used in this thesis, is abduction, which aims at finding the best explanation given certain findings.

2.3 Techniques

There is a large range of formal methods ranging from (computationally) cheap and very incomplete methods to very expensive and less incomplete methods. Complete correctness of a system cannot be guaranteed, which is often stated in literature (e.g., [McMillan, 1993]), as it is impossible to enumerate all the correctness criteria for any non-trivial system. Even though completeness is not feasible, formal methods are more than just "proofs of correctness" as many problems can occur with respect to ambiguity, incompleteness, and misunderstandings. They force one to make things explicit with respect to the requirements of the system as well as the assumptions that one relies on.

In order to use formal methods for the verification of systems, Kripke models are looked upon as a state transition system, i.e., all the states of the model are considered states of a system. There are two main traditions in formal verification, namely theorem proving and model checking. The first deals with establishing logical consequences of logical formulas, i.e., to establish whether a formula provably follows from a set of premises:

$$\Gamma \vdash \varphi$$

Model checking on the other hand takes a specification of a model M and decides whether or not a formula holds, i.e., whether or not:

$$M \models \varphi$$

Intuitively, and formally as we will see below, model checking is easier as finding the logical consequences of Γ as done in theorem proving requires one to take into account all models that satisfy Γ, while one specific model is given in the case of model checking. Even though this limits the applicability of model checking (as one needs to be able to specify a model), many systems can be described in terms of such a finite Kripke model, which makes model checking a powerful technique.

Some basic notions with respect to these techniques are discussed next.

2.3.1 Theorem Proving

In this thesis, we mean by theorem proving the computer-supported reasoning using a set of inference rules, such as the resolution rule that was introduced in Section 2.2.1. A distinction can be made between automated theorem proving and interactive theorem proving. Classical automated theorem proving aims at proving theorems completely automatically, whereas interactive theorem proving may require additional user interaction. Systems of the former type include OTTER [McCune, 2003], Prover9 [McCune, 2007], Vampire [Riazanov and Voronkov, 2002], and E [Schulz, 2002]. Interactive theorem provers that are used in computer science are systems such as HOL [Slind, 1991], PVS [Owre et al., 1992], Isabelle [Paulson, 1989], and a system that is used in this thesis: KIV [Balser et al., 2000].

If one wants to do actual reasoning with temporal logic, issues such as completeness, decidability, and complexity are relevant. For first-order logic, the resolution rule is refutation-complete, i.e., $\Gamma \models \bot$, then $\Gamma \vdash \bot$, which, combined with the logical equivalence of $\Gamma \models \varphi$ and $\Gamma \wedge \neg\varphi \models \bot$ yields completeness. Completeness of proof procedures of temporal logic vary, e.g., temporal resolution is complete [Dixon et al., 1998], whereas the completeness of the temporal strategy of KIV is an open issue [Balser, 2005, p. 143]. For automated reasoning, one would like decidability, i.e., to be able to distinguish between theorems and non-theorems. While first-order logic is undecidable (though still semi-decidable, i.e., valid theorems will eventually be found), the propositional temporal logics we have discussed in this chapter are all decidable, though logical reasoning is typically a hard problem. For future-time temporal logics, for example, checking validity of a formula in both LTL [Sistla and Clarke, 1985] and ITL [Aaby and Narayana, 1988] is PSPACE-complete, while for CTL this problem is EXPTIME-complete [Emerson and Halpern, 1982]. Nonetheless, smart heuristics make it possible to reason about interesting problems, even for problems stated in first-order logic (e.g., [Phillips and Vojtěchovskiý, 2005, Jech, 1995]).

2.3.2 Model Checking

In terms of complexity, model checking is an easier task. While theorem proving using CTL is EXPTIME-complete, model checking can be done in com-

putational complexity which is linear to the size of the model and formula [Clarke et al., 1986]. LTL model checking, given a branching structure, on the other hand, is still PSPACE-complete, although also polynomial in size of the model. The former is straightforward to see when considering that all linear models can be described in the branching structure, making it no easier than theorem proving. In this sense, it is unfair to call this "model checking" as a model of an LTL formula is a linear structure. As such linear structures have no use in modern program verification, such type of model checking is not a topic of interest.

The discussion on whether to use CTL or LTL for model checking is far from over, as LTL is usually more intuitive and better suited as a specification language. For example, recently, Vardi [Vardi, 2001] revived this discussion by summing up the advantages of linear-time frameworks in terms of expressiveness, compositionality, property-specific abstractions, and uniformity.

As also discussed in Chapter 1, symbolic model checking is based on the use of compact representations of sets of states using clever data structures, for instance Binary Decision Diagrams (BDD). This approach has been very successful in the last decade, which can be observed by the fact that this technique is used on a wide industrial scale. Nonetheless, BDDs may grow exponentially, which restricts the size of the model that can be verified efficiently. Hence, other techniques have also been used such as Bounded Model Checking (BMC), which is based on propositional SAT solving [Biere et al., 2003], which was first introduced in [Biere et al., 1999]. So far, it has been shown that BMC can be used for verifying systems that could not be verified with BDD techniques. However, at the same time, problems exist that can be solved more efficiently using BDD techniques. The idea of BMC is to search for a counter-model in executions whose length is bounded by some constant k. Typically, the model checker iterates from 0 to the pre-defined upper bound k and terminates once it has found a counter example. As a consequence, this method is incomplete as the smallest counter-example may be beyond the upper bound provided by the user. This is a promising technique for clinical guidelines, usually only a small number of treatments are discussed (cf. Chapter 3), whereas there is a large amount of non-determinism present.

Finally, another technique, which is employed in this thesis, is so called *modular verification*, i.e., to verify a restricted part of the system (cf. [Kupferman and Vardi, 1998]). In the assumed-guarantee paradigm, the specification of a module consists of a specification of guaranteed behaviour assuming that the system behaves in a certain way. This is called the assumed behaviour. In Chapter 7, the assumed behaviour is written down in a linear temporal logic and the guaranteed behaviour in branching temporal logic. The assume-guarantee assertions are written down as $[\varphi]M\langle\psi\rangle$, meaning that the CTL formula ψ holds in the computation tree consisting of all computations of the program, described by M, that satisfy the LTL formula φ.

2.4 Formal Systems in Biomedicine

One important area where formal calculi have been applied is systems biology, that aims at modelling the complex interactions in biological systems. In particular the modelling of metabolic pathways, i.e., series of chemical reactions occurring within a cell, has been a topic of interest. Several formalisms have been employed. First, process calculi, such as the well-known π-calculus [Milner, 1999, Kuttler, 2006], which are used for modelling concurrent systems, have been applied to this area, e.g., using a stochastic process calculus in [Errampalli et al., 2004]. Second, Petri-nets have been proposed for modelling such networks in [Hofestädt and Thelen, 1998]. Finally, there are several papers which aim at modelling biological pathways using specialised languages in order to apply model checking algorithms for investigating relevant properties of such a pathway. For example, the rule-based language Pathway Logic [Eker et al., 2002] and the BIOCHAM language [Chabrier and Fages, 2003], which can be used to build models that can be queried using CTL in the SMV model checking tool. Probabilistic modelling and model checking of biological pathways has also been proposed [Pronk et al., 2007].

With respect to medical reasoning, various representation languages for formalising and reasoning about medical knowledge have been used and developed, e.g., (heuristical) rule-based systems [Shortliffe, 1974] and causal representations [Patil, 1981]. There has not been done much work in logical reasoning about medical knowledge. An exception is [Lucas, 1993], which studied the use of logic modelling for building expert systems. Predicate logic was used to formalise models of medical knowledge and could be reasoned with using classical automated theorem proving. In more recent years, probabilistic models have gained popularity in this area. For example, statistical knowledge is succinctly represented in Bayesian networks [Pearl, 1988], and can be used to represent medical knowledge (e.g., [Lucas et al., 1998] and [van Gerven, 2007] for recent results).

Less research has been done to exploit formal methods for reasoning about the correctness of the clinical decisions. Of course, the modelling of clinical reasoning is a well-known topic in the area of decision support and expert systems [Lucas and van der Gaag, 1991] resulting in several medical systems as mentioned in Chapter 1 (e.g., MYCIN [Shortliffe, 1976]) and could be taken as a basis for such an approach. Despite that, little has been done in order to study and verify quality of the decisions themselves, besides what was already mentioned in context of clinical guidelines (cf. Section 1.3).

2.5 Conclusions

In this chapter, we have presented several languages that allows us to formalise the necessary medical and guideline knowledge. These languages will be explored in a number of chapters. First, in Chapter 4, linear temporal logic is

used for reasoning about medical knowledge and the guideline. Then, in Chapter 5, the medical knowledge formalised in linear temporal logic is combined with the guideline formalised in a guideline representation language. Formal aspects of this approach are discussed. In both Chapters 4 and 5, the focus will be on the use of theorem proving for reasoning about the quality of clinical guidelines. In Chapter 6, we employ computational tree logic for modelling parts of a guideline and investigate how this logic can be used for reasoning about so-called guideline adaption and for investigating the quality of clinical management. In the latter study, we compare this to linear temporal logic. The technique that is used here is model checking. Finally, Chapter 7 proposes to use interval temporal logic for modelling parts of the guideline and we extend this logic to certain aspects that are underlying clinical guidelines. We investigate how such an approach can be used to verify parts of the guideline and compare this to the other approaches of Chapters 4 and 5.

Chapter 3

Guidelines and Protocols

In order to obtain useful results by applying formal methods to a system, one must first have a clear understanding of the system under consideration. Compared to clinical guidelines, technical systems are often quite well understood. For guidelines, it is not at all clear what an appropriate model of such a guideline entails, e.g., in this thesis, it is argued that physiological knowledge should be part of a model, whereas many of the guideline modelling languages ignore this aspect of the guideline. Moreover, the modelling itself is difficult as it requires of a combination of technical knowledge as well as medical expertise. For example, in the Protocure project, it was found that most errors in the models were introduced during the modelling of the guideline. Finally, requirements with respect to guidelines should be made explicit. If it is unknown what a guideline is supposed to accomplish, or if the requirements are simply not useful, then any results derived from these requirements are equally useless. As a result, it is of importance to obtain insight into the nature of clinical guidelines, which is the goal of this chapter. Even though this will not solve the inherent difficulty in modelling a guideline, a problem that in its full generality is beyond the scope of this thesis, it provides a direction for investigating the formal aspects of a guideline.

The systems that are of interest in this thesis are clinical guidelines, also sometimes referred to as medical guidelines or clinical practice guidelines (CPG). These are documents which include recommendations, advice and management instructions aimed at supporting the decision-making process of healthcare professionals and patients, based on the results of scientific research and subsequent professional opinion. Their aim is to establish good medical practice in healthcare [van Everdingen et al., 2004]. A secondary aim is to make the care process more effective [CBO, 2002]. Research in the last few years has shown that clinical guidelines can indeed improve healthcare outcomes [Woolf et al., 1999] and may even reduce the costs of care up to 25% [Clayton and Hripsak, 1995]. Worldwide, a number of organisations, such as

the Dutch Institute for Health Care Improvement (CBO) in the Netherlands and Scottish Intercollegiate Guidelines Network (SIGN) in Scotland, have been founded to assist specialist groups in the development of guidelines. Other organisations, such as the Dutch General Practitioners (NHG) carry out similar activities for general practitioners. In 2002, the Guidelines International Network (GIN) was founded to promote systematic development of clinical practice guidelines through international collaboration [Ollenschläger et al., 2004].

A related type of document is a clinical protocol, which is seen as a local version of a guideline, meant to be used to support effective daily clinical care. The need for a protocol in conjunction with a guideline is twofold: firstly, a guideline is an extensive document (e.g., the Dutch breast-cancer guideline is 121 pages), and, therefore, it is not easy to locate relevant information; secondly, detailed recommendations about duration, dose, or actual procedure have been omitted from the guideline ensuring that it is generic, and, thus, these details are included in a protocol to complement the information that is in the guideline. Hence, basically, a clinical protocol is a summary of the most important sections that are in the guideline with respect to additional local management information.

In this chapter, we introduce the necessary preliminaries with respect to guidelines and protocols. In the first part, we concern ourself with the question how guidelines are being developed at the moment, and how protocols are derived from guidelines. In the second part of this chapter we look at two example guidelines: a guideline about the treatment of diabetes mellitus type 2 and breast cancer. Finally, we address the issue of quality of clinical guidelines. The description of guidelines and protocols that is presented here essentially yields an interpretation of guidelines and protocols that is taken as a starting point for the rest of this thesis.

3.1 Development of Guidelines and Protocols

This section is based on documentation provided by the Dutch Institute for Health care Improvement (CBO)[1].

3.1.1 Summary of the Process

Figure 3.1 gives an overview according to the guideline-development process in the view of CBO. The development process goes through the following phases:

(1) Selection of the guideline topic. This is based on perceived needs within the healthcare community with regard to setting standards of care for a particular patient group.

(2) Installation of the guideline-development team. The people involved in the development of a guideline have different backgrounds and expertise.

[1] http://www.cbo.nl

For example, the team could include medical specialists, clinical pharmacists, patient group representatives, guideline developers, so that they, from their own field of expertise, can contribute insights and principles. The chairperson of the group normally has considerable authority in the topic concerned.

(3) Design. This concerns the actual development of the document, based on the gathering of relevant information and discussion among the members of the guideline-development team.

(4) Comment phase. The national community to which the guideline is of concern is given the opportunity to offer comments in a national guideline meeting. In addition to this, the scientific organisations (such as the Netherlands Heart Foundation) may offer comments.

(5) Dissemination. The guideline is sent to various organisations, published in a journal (often the Netherlands Journal of Medicine, NTVG) and on the CBO website.

(6) Evaluation. In the end, guideline developers wish to learn whether implementation of the guideline within a healthcare organisation has had an impact on the quality of care. If this is the case, the effort put into the guideline development is seen as having been justified. Evaluation is done by considering *indicators*, i.e. measurable elements, such as number of years of survival following treatment.

In this thesis, we are in particularly interested in the main product of the guideline-development process, i.e., the guideline document, and the way the knowledge represented in this document changes during the development process. This is why in the remainder of this section the focus is only on one of the phases mentioned above: the *design phase*.

3.1.2 Design of a Guideline

The identification of the key questions which need to be addressed by a guideline is of major importance, because if a guideline does not address these it is unlikely it will ever be employed by the targeted community. These questions are answered by selecting relevant scientific evidence from literature, which is then summarised for inclusion in the guideline, and forms the basis for the recommendations and included advice. In order to better understand this process, we will go into somewhat more detail.

The design phase of guideline development focuses on the process of drafting the actual document. It consists of the phases from the problem analysis to the writing of the actual guideline, which is depicted in Figure 3.1. The most important phases are discussed below.

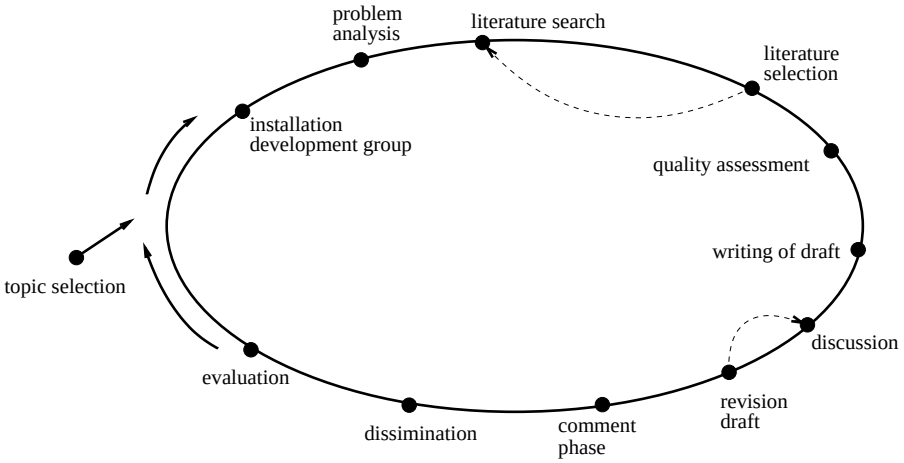


Figure 3.1: Adapted CBO development cycle [CBO, 2005] highlighting some of the main steps in the development process.

Problem analysis

In the preparatory phase a problem analysis is performed. The goal of the analysis is to collect a set of key questions for which the following criteria hold:

- providing an answer is seen as highly relevant by the community of healthcare professional working in the areas as well as by patient groups,
- there exist no consensus among experts about the appropriate answer,
- answers can be provided in the form of scientific evidence and argumentation by inspecting the scientific literature.

The relevance of problems may be established by sending out surveys to clinicians and healthcare organisations and by having discussion in the working group. This is currently not done on a regular basis. Surveys are also vital in order to obtain support for the guideline in the medical community and to make sure that the key questions are seen as being relevant *in practice*.

Collecting evidence and the role and significance of PICO

The guidelines developed by the CBO are evidence-based, which means that in practice, when a healthcare professional or patient need to make a decision, he or she should be aware of the evidence and its strength that support this decision. The strength of the evidence is judged based on the quality of the studies that have been performed. For example, strong evidence comes from

randomised controlled trials (RCTs), which is a scientific method for comparing the effectiveness of medical procedures, while opinions based on clinical experience are considered to be much weaker.

The first step in collecting relevant evidence from literature consists of drafting an appropriate question, which is done in a structured fashion. A question always consists of four subcomponents, which is expressed by the acronym PICO; *PICO* stands for:

- **P**atient group, i.e., the group of patients having particular characteristics (have a particular disease, e.g. patients with breast cancer, or have undergone a particular treatment);
- **I**ntervention, i.e., an action such as treatment;
- **C**omparison with other interventions, e.g., one treatment against another one, or one diagnostic test against another one;
- **O**utcome: the result of the intervention, for example, diagnosis, cure, survival or death.

The utilisation of the PICO method is done for a number of reasons: it forces one to focus on the issues that really matter with regard to the scope of the guideline, producing questions which can be answered in principle if enough literature is available. As a result it creates an upper bound on the number of papers that have to be considered and examined.

The medical literature databases are then searched by reformulated the PICO into a list of keywords and by doing database searches based on this. The best way to do this heavily depends on the capabilities of the database used. Databases which allow the user to search for literature based on methodology are preferred. In general the results of this search generates a huge amount of noise, so a successive selection of literature has to be done manually. Popular literature databases are: MEDLINE/PubMed[2], the Cochrane library[3] and EMBASE (Excerpta Medica DataBASE) from Elsevier[4].

The quality of the selected literature is then established. Several 'checklists' have been developed which can be used to verify the existence of possible *bias* in studies. Examples of possible biases are performance bias (the physician that treats the patient or the patient itself is aware of the assigned treatment) and detection bias (the physician that has to determine the outcome is aware of the assigned treatment). It is essential that several members of the group criticise the literature independently. Afterwards the members argue about their judgement to reach a consensus.

A *systematic review* is a structured process that consists of finding relevant literature as described above, extracting the results from it and by combining

[2] `http://www.nlm.nih.gov` [3] `http://www.cochrane.nl` [4] `http://www.embase.com`

it using the technique of *meta-analysis* into a structured document. Thus, the result is a sort of summary of all the evidence available in the literature. The highest level of evidence is a systematic review of randomised controlled trials and it is therefore important these exist when formulating recommendations from this evidence. An example of a site where systematic reviews are being offered is EMB Online (Evidence-based Medicine Online)[5].

From evidence to recommendation

Evidence from literature is organised in a so-called *evidence table*. Although these tables do not have a standard structure or content, they typically list a number of criteria that is considered important to establish the weight of this evidence. Examples are the type of study, the number of patients, statistical relevance, clinical relevance, etc.

From these evidence tables a preliminary text can be composed consisting of at least the following elements: (1) a clear starting question, which is answered in a certain section of the guideline, (2) the method used to find literature, (3) a summary of the literature (results and evidence tables), (4) an examination and commentary on the results, (5) conclusions based on the literature, (6) a level of evidence for each conclusion, and (7) the recommendation based on the conclusion that can be used in practice.

A number of conclusions are formulated from the evidence found in the literature. A conclusion is a 2-tuple of a statement and an associated level of evidence. The CBO distinguishes five levels of evidence for *treatments*:

(A1) Systematic reviews with at least some studies of A2 level, where the results of the individual studies are consistent.

(A2) Randomised comparative clinical research of good quality and sufficient size and consistency.

(B) Randomised clinical trials of poor quality of insufficient size or other comparative research (non-randomised, comparative cohort-research)

(C) Non-comparative research

(D) Opinion of an expert

Diagnostic conclusions are rated with the same evidence levels, but each of the evidence levels has a different definition associated with it. These definitions are omitted in this thesis.

It is stressed, however, that the levels of evidence are not a measure of the quality of a guideline, nor can they be used as an excuse to diverge from a conclusion. They may be used as a tool for the reader and possibly as a consideration when formulating recommendations.

[5] `http://ebm.bmjjournals.com`

The construction of recommendations is performed starting with the evidence taking into account a number of other considerations. The considerations have been organised into 10 '*domains*'.

- *Domain 1: Clinical relevance.* The working group establishes whether evidence will be useful and practical for physicians.
- *Domain 2: Safety.* Interventions should not cause unnecessary side-effects. Furthermore, risks on short and longterm should be made clear to patients.
- *Domain 3: Patient perspective.* Patient satisfaction is important in the medical field. Expectations of patients about their treatment and the accessibility to a certain treatment should be considered.
- *Domain 4: Perspective of professionals.* Risks for physicians and the amount of time it takes to implement a recommendation.
- *Domain 5: Availability of resources.* As stated before: the recommendations must be practically feasible. This means that sufficient human and non-human resources must be available.
- *Domain 6: Costs and effectiveness.* Costs and the effectiveness of an intervention must be in a reasonable balance.
- *Domain 7: Organisational.* It should be possible to implement recommendations in existing organisations. Typical items here try to capture the practical possibilities in terms of necessary change to an organisation and existing infrastructure for the implementation of a particular intervention to be.
- *Domain 8: Judicial.* Local judicial consequences for the professional and specific laws due to implementation of the intervention.
- *Domain 9: Ethical.* General ethical values have to be applied, such as truthfulness and honesty, which has, for example, lead to the concept of informed consent in medicine, which states that a patient has to be give consent based upon an appreciation and understanding of the facts and implications of an intervention.
- *Domain 10: Industrial.* In some cases it can be relevant to question the commercial interests industry can have in implementing an intervention.

The recommendations are the outcome of discussions carried out in the working groups. Clearly, given all these considerations, recommendations are heavily dependent on the opinions of the experts (the working group). This means that, typically, recommendations on the same subject vary between countries, even though the evidence is often similar. A few objective criteria

can be given to judge the quality of recommendations; they should be (1) relevant and practical, (2) clear and unequivocal, (3) prescriptive (not descriptive), (4) optional, not obligatory and (5) when a consensus cannot be reached, either recommendations should be omitted or the different considerations should be denoted.

3.1.3 Protocol Development

One way to reduce the costs and time for developing guideline and protocols, and avoid unnecessary duplication of effort of guideline development is by relying on the local adaptation of guidelines developed at the (inter)national level by expert groups [Groot et al., 2008]. In this context 'guideline adaptation' is a process in which existing guidelines are adapted so that they can be used within a different care setting. A local adaptation is called a *(clinical) protocol*. As explain earlier, a protocol typically provides detailed information about duration, dose, or procedure, suited to the local context. Often, such detailed information is lacking in guidelines. Although a clinical protocol is a summary of the most important sections that are in a guideline, mostly recommendations, certain recommendations may be changed if they do not fit the local context.

In this thesis, we will not go into detail about the exact procedure; however, guideline adaptation should follow similar procedures as used in guideline development, including making transparent any decisions and key factors that influence the modifications. In recent years, several of such procedures have been proposed [Graham et al., 2003, Graham and Harrison, 2005, Graham et al., 2002, Graham et al., 2005].

3.2 Examples of Guidelines

In this section, we will discuss two case studies that act as a running example throughout this thesis. In Section 3.2.1, we will discuss the guideline on treatment of diabetes mellitus type 2 (DM2). In Section 3.2.2, the guideline on the treatment of breast cancer is discussed.

These guidelines are considerably distinct as the diabetes guideline is aimed at the general practitioner, whereas the breast cancer guideline is developed for medical specialists. As a result, the latter is more extensive in its justifications, whereas the diabetes guideline contains more detail.

3.2.1 Diabetes Mellitus type 2

In 2003, about 36 per 1000 men and 39 per 1000 women were diagnosed with diabetes mellitus type 2 in the Netherlands. Worldwide, the prevalence of diabetes is rising due to population growth, aging, urbanisation, and increasing prevalence of obesity, and physical inactivity [Wild et al., 2004]. In this section, we introduce the recommended management of this disease.

The guideline that acts as a case study here is the Dutch general practitioner's (NHG) guideline of 2003 [NHG, 2003]. An example of a part of a guideline is the following (translated) text:

> 1. refer to a dietician; check blood glucose after 3 months
> 2. in case (1) fails and Quetelet Index (QI) ≤ 27, then administer a sulfonylureum derivate (e.g., tolbutamide, 500 mg 1 time per day, max. 1000 mg 2 per day) and in case of Quetelet Index (QI) > 27 biguanide (500 mg 1 per day, max. 1000 mg 3 times per day); start with lowest dosage, increase each 2-4 weeks if necessary

This guideline is particularly concise (about 3 A4 pages). While modern guidelines can be as large as 100 pages, the number of recommendations they include are typically few. In complicated diseases, each type of disease is typically described in different sections of a guideline, which provides ways to modularise the guideline in a natural fashion.

At the start of the 2006, this guideline was updated. The most prominent change compared to the 2003 version is that the biguanide drugs are now indicated for all patients from step 2 onwards. This is due to new evidence that suggests that these drugs can reduce the mortality rate, while the sulfonylurea drugs do not. Subsequently, in case they are is not sufficient, thiazolidinediones are suggested as a replacement for sulfonylurea drugs for obese patients with cardiovascular problems and without heart failure.

3.2.2 Breast Cancer

In the Netherlands only, as many as 10,000 women are diagnosed with breast cancer every year. For women, the chance of ever being diagnosed with this disease is 10% [CBO, 2002]. Changes in DNA, in particular the genes that control the instructions for cells to grow, divide, and die, may cause cancer; however, little is known under which circumstances this actually happens [American Cancer Society, 2006].

The guideline that we have used was the 2004 version of the Dutch CBO guideline on the treatment of breast cancer. This guideline is considerably more complex than the diabetes guideline due to the fact that it was developed more systematically described by the methodology in Section 3.1. This warrants a more comprehensive analysis of the guideline, which is described next.

3.3 Analysis of Guidelines

In this section, we will attempt to give an indication of how a guideline is organised, where in particular the difference in semantics of the various parts or fragments is given attention. The structure of the guideline is the product of the development process. Therefore, by looking at these structures, we will gain insight into the underlying meaning of the guideline text. As the breast cancer guideline is much more extensive compared to the diabetes guideline,

1.2 Diagnostic and treatment of operable invasive breast cancer

In this chapter, operable invasive breast cancer is used to describe: T1-2 N0-1 M0 breast cancer (UICC 2002).

1.2.1 Diagnostic procedures for invasive breast cancer T1-2 N0-1
Please refer to the CBO-guideline 'Diagnostic procedures for breast cancer' (Spring 2000).
There are extensive options for investigating dissemination in patients with breast cancer. The value of carrying out extensive diagnostic procedures in patients with localised disease is questionable since metastases, if present, cannot be detected.[21]
(...)
Conclusion

Level 3 For patients with T1-2 N0-1 breast cancer, preoperative investigations to detect metastases are not beneficial.

C Samant,[23] *Ciatto,*[24] *van der Hoeven*[27]

Recommendations
For T1-2 N0-1 breast cancer, preoperative investigations to detect metastases are not recommended. Symptoms which may be indicative of metastases should be evaluated. In the case of a high postoperative stage, investigations to detect metastases may be considered.

Figure 3.2: Structure of the CBO guideline on breast cancer (p. 13)

we use the breast cancer guideline as an example. All the page numbers that are mentioned in the footnotes refer to pages of the breast cancer guideline.

Global structure

A guideline is divided in several chapters. The first chapter is generally the introduction. The other chapters have a specific topic related to the primary topic which do not overlap with other chapters. All chapters are divided in subsections that contain:

- *summary text,* which normally serves as an introduction to the issues that follow so that the reader is able to understand the arguments underlying recommendations and conclusions;
- *conclusions*: these are short summary statements of the important insight from the literature, introduced in the preceding guideline text.
- *recommendations:* these are statements pertaining to (medical) management actions.

See for example Figure 3.2. The structure of these chapters obviously depends on the questions they want to answer (cf. Section 3.1). For each question the primary literature is listed together with additional considerations. From the primary literature the most important conclusions are given a 'grade of evidence' and are put in a separate box, which is based on the level of evidence of the individual studies (cf. Section 3.1). Finally, note that the recommendations follow from the primary literature and the additional considerations, i.e., not merely from the conclusions that are highlighted.

Structure in narrative text

The text primarily consists of outcomes of PICOs (e.g., *Adjuvant hormone therapy for locally advanced breast cancer results in improved survival in the long-term.*[6]), sometimes combined with the type of research (RCT,...) and an evaluation of the paper (e.g., *In addition, in a number of the above-mentioned studies only the responders to neoadjuvant chemotherapy were analysed, which has resulted in a bias when comparing locoregional treatment studies.*[7]). Many other arguments can be found in the guideline that are not directly related to the main effects and the appraisal of the evidence, for example:

- Advice of treatment in scientific literature:
  Some authors therefore recommend just chemotherapy and radiotherapy for this group of patient.[8]

- Factors that are important in decisions making, including contra indications:
  In addition, the following conditions are important to ensure an optimum treatment result with breast-conserving therapy for DCIS: (..).[9]

- Prioritising of factors that are important for making decisions:
  The ultimate choice of treatment (...) lies with the patient.[10]

- Side-effects of a treatment, although usually considered to be common knowledge:
  The possible occurrence of cancer specific problems is indicated, such as: limited mobility of the arm, lymphoedema, skin problems, coping problems, early menopause, pain as a result of damage to the nervus intercostobrachialis, or of irradiation of the breast.[11]

- Relevant tests to reach the goal:
  It may be beneficial to select patients for the sentinel node procedure

[6] p. 48 [7] p. 47 [8] p. 49 [9] p. 12 [10] p. 12 [11] p .60

by carrying out axillary staging using ultrasound of the axilla and ultrasound-guided fine needle aspiration biopsy.[12]

Besides these scientific statements, we can also collect a number of non-scientific arguments. Examples of this vary in structure: from appealing to common sense (e.g., *As long as these questions remain unanswered, the approach should be based on common sense.*[13]) to stating the acceptable outcome (e.g., *It is generally assumed that a maximum (cumulative) risk of local recurrence of 1% per year is acceptable for BCT.*[14]). Finally, we sum up some other aspects of a guideline:

- Definitions:
  - Definition of treatments:
    BCT implies ample local excision of the tumour, an axillary staging procedure and radiotherapy of the breast.[15]
  - Definition of diseases:
    Definition: Locoregionally advanced breast cancer is used to describe breast cancer which is unresectable on the basis of the classic unresectability criteria: oedema of the skin (peau d'orange), ulceration, satellite skin nodules, inflammatory carcinoma, infiltration of the chest wall (T4), lymph nodes fixed to one another and/or to deeper structures (N2), or palpable internal mammary, parasternal, infraclavicular and/or supraclavicular lymph nodes (N3). In addition, large primary tumours (> 5 cm; T3) are also included in this category (T3, T4, all N classes, M0; all T classes, N2 or 3, M0).[16]
- Pros and cons on the basis of scientific literature, situation in the Netherlands, bias, etc:
  In the groups without radiotherapy in the two Danish studies, locoregional recurrence was found in approximately 30% of cases after a 12-year follow-up period. This suggests that the surgical treatment was inadequate in many cases. In Denmark axillary dissections were not carried out, in contrast to the standard procedure in the Netherlands, and instead level I and II samplings were carried out.[17]
- Temporal aspects:
  - Control flow is usually not provided explicitly, but may be deduced from advice and evidence that is given:
    (...) good communication between those making the diagnoses and

[12] p. 24 [13] p. 24 [14] p. 15 [15] p. 14 [16] p. 45 [17] p. 18

those treating the patient in the various disciplines is required, preferably in the form of structured weekly discussions.[18]

– Order of treatment is usually not explicitly mentioned. Sometimes optimal order is given, or explicitly stated to be unknown:
The optimal sequence for adjuvant chemotherapy and radiotherapy is not known.[19]

- Goals are usually mentioned in the introduction to chapters:
The main objective within the framework of the treatment of metastasised breast cancer is maintaining or improving the quality of life by treating complaints.[20]

Characterisation of Knowledge

Time is used in a guideline to describe the changes in the state of the patient and its environment. In many cases, an imprecise characterisation of temporal information is used, e.g., 'before treatment' and 'after treatment'. Sometimes, guidelines are more specific and actually give reasonably precise time frames, such as minutes, days, weeks, or months, but only in a limited number of cases medical science is as precise as in physics, such as the length of time in which radiotherapy is applied. In the past, time has been thoroughly researched in the context of logic [Prior, 1957], information modelling, real-time systems [Chaochen, 1993, Fidge et al., 1998], and other areas. Several logical ontologies can be taken as a starting point for modelling time, which was described formally in Chapter 2.

A second important aspect of guidelines is the description of patients and patient groups. Typical elements in the state of a patient are symptoms, signs and other measurable elements. Finally, important are the actions that have to be taken place. In the context of guidelines, we will refer to these as *interventions*, i.e., all kinds of medical actions that influence the condition of a patient or the environment of that patient. A formalisation of these aspects will be introduced in subsequent chapters in several ways depending on the purpose for which the formalisation is introduced.

3.4 Quality Criteria

With respect to quality of clinical guidelines, several quality measures have been proposed. This section will deal with the methods for quality assurance of guidelines. In the next section we will give a short impression of how so-called indicators are used, which are formulated during the evaluation phase of the guideline development. Then, we will present the AGREE instrument that is often used during the guideline development process. We discuss the

[18] p. 52 [19] p. 40 [20] p. 71

aspects of these measures for the use of applying formal methods. We conclude that they are not sufficient and propose a framework of quality measures for clinical guidelines.

3.4.1 Quality of Health care: Medical Indicators

Indicators are quantitative measures that can be used to monitor and evaluate the quality of, in this case, healthcare organisations. Three types of indicators can be distinguished:

1. *Structural indicators* give insight in the side conditions that have to be fulfilled to meet healthcare needs.

2. *Process indicators* assess what a provider did for a patient and how well he or she did it.

3. *Outcome indicators* can be used to monitor an outcome variable.

Furthermore, indicators can be *internal* or *external*. Internal indicators are not made public because of their subjectiveness; they can be used to measure quality inside one's own healthcare organisation. In contrast, external indicators are used to give objective advice holding providers of healthcare accountable for their quality.

It should be noted that guidelines are meant to promote the quality of healthcare organisations and, therefore, the quality of guidelines is related to indicators. For example in [ten Teije et al., 2006], process indicators are taken as quality measures. The main advantage is that these process indicators are relatively simple to obtain as the external indicators are made public. Moreover, it is conceptually clear that they can be matched to recommendations given in the guideline.

There are also disadvantages to this approach. The first is that indicators are directly derived from recommendations of guidelines. In such a case, indicators are trivially consistent with recommendations. Therefore, in [ten Teije et al., 2006], indicators are used that are derived from other guidelines. As guidelines are based on national circumstances, such as the demography, it is unlikely that such process indicators should necessarily hold. Moreover, weighing various considerations in the guideline development may result in a different treatment processes that are of equally high quality. Finally, indicators are limited to particular aspects of clinical practice, as they are developed for parts of the clinical pathway that is likely to be done differently than what is recommended.

3.4.2 Quality of Guideline Development: AGREE

The AGREE instrument [Cluzeau et al., 2003] is a tool to establish the quality of guidelines developed by the guideline development community. It consists of a 23 items divided into 6 '*domains*' dealing with (1) clear scope and purpose

of the guideline, (2) stakeholder involvement, (3) methodology, (4) clarity and presentation, (5) possibility to apply the guideline in practice, and (6) editorial independence.

Even though the AGREE instrument can be used to assess a guideline after its development, it is typically used as a quality guide *during* the development of a guideline. As a result, virtually all the items refer to the rigour of this development process. The underlying assumption is that if the development process was of high quality, then the guideline must be of high quality as well.

In case of formal verification, most of these items cannot be exploited. Exceptions to this are the clarity and presentation, which are important to properly model a guideline. Furthermore, the possibility to apply the guideline in practice has a decision-theoretical element to it, as this includes the cost effect of implementation. Nonetheless, few of the remaining items can be of use.

3.4.3 Quality Criteria for Formal Verification

In related work with respect to verification of guidelines as described in Section 1.5, very similar to software engineering practice, ad-hoc decisions are made to decide which property is being checked. As a consequence, the specification of properties has focused on 'programmatic' type of properties, in the sense that properties should be expressed in terms of the syntactical part of the structure of the guideline. As we are interested in investigating properties of clinical guidelines, this type of approach is alluring for computer scientists, because with such a program-like structure it allows them to use program verification techniques almost directly. However, it is not obvious that these of properties are indeed the type of properties that are most relevant to quality checking of a guideline.

From the development of guidelines, it can be seen that evidence alone does not provide enough means to construct concrete recommendations. First, it is of importance to assess the nature of the evidence. For example, evidence based on randomised control trials (RCTs) should have more influence on decision making than expert opinions alone. Second, the applicability of the evidence to the population of interest, i.e., its generalisability, should be considered. Finally, the costs of the treatment and available knowledge about a particular healthcare system should be taken into account.

Focussing on the nature of the evidence, it is clear that the opinion of the expert is based on certain trade-offs between wanted and unwanted effects of interventions. Recommendations based on evidence alone are likely to be correct. In contrast, trade-offs are much more complex and therefore it can be expected that problems related to these trade-offs might occur in guidelines. In this section, we propose a systematic approach for finding quality criteria based on the idea that guidelines are based heavily on these trade-offs. Note that it is not always obvious what the relevant trade-offs are; however, in many cases there is some idea of what the important points of discussion are

for prescribing a certain treatment. As a consequence, medical expertise is needed to identify the trade-offs.

In the next section we will sketch the possible approaches there are for a systematic methodology for finding quality criteria. After this, we will focus on one specific methodology for finding these quality criteria based on medical background knowledge.

Approaches

There are three main approaches to identify the goals of a specific high-quality guideline, namely the so-called knowledge-oriented approach, the guideline-oriented approach, and an approach based on literature.

In the literature approach one tries to identify goals from external literature. One possibility is by using indicators, which are measures of quality of healthcare as described earlier. Besides process indicators, outcome indicators could also be used. For example, an indicator for many types of breast cancer is the 5 year survival rate. Such indicator can then be interpreted as a goal to maximise the survival of all individual patients. Similarly to process indicators, the scope for such quality measures is limited.

In the guideline oriented approach one tries to extract medical goals from the guideline text directly. In [Serban et al., 2004], this was done in two ways using pattern recognition software. If the goals, intentions and measures of outcome were mentioned explicitly these were matched with specific patterns. Goals that are implicit in the guideline text are much harder to find using patterns. Possibly, background knowledge could be exploited for improving the search to find such patterns in a highly automated manner. However, it is a difficult and error-prone process which needs further improving to apply this in practice.

In this section, we will focus on the knowledge-oriented approach where one tries to identify important goals based on background knowledge, i.e., knowledge that is present during the design of the guideline but is not necessarily recognised as such. In the next section we will present a hierarchy of goals and will provide examples to motivate the approach.

A knowledge-oriented approach towards finding proof goals

Many of the goals that can be formulated are related to each other. In particular, many goals are more specific or more general instances of other goals. As a consequence, we present a hierarchy of objectives and goals, based on medical expert knowledge derived from guideline developers.

In this approach, we distinguish between the main objective of an intervention and a number of optimisation criteria. The objective can for example be diagnostic or therapeutic and is independent of the question whether or not it is the ‘best’ intervention given some preference relation. Of course, preferences of patients can differ, but in practice there are a number of restrictions

to these preferences which we will refer to as optimality criteria. For example, we expect that all patients prefer the least amount of side-effects.

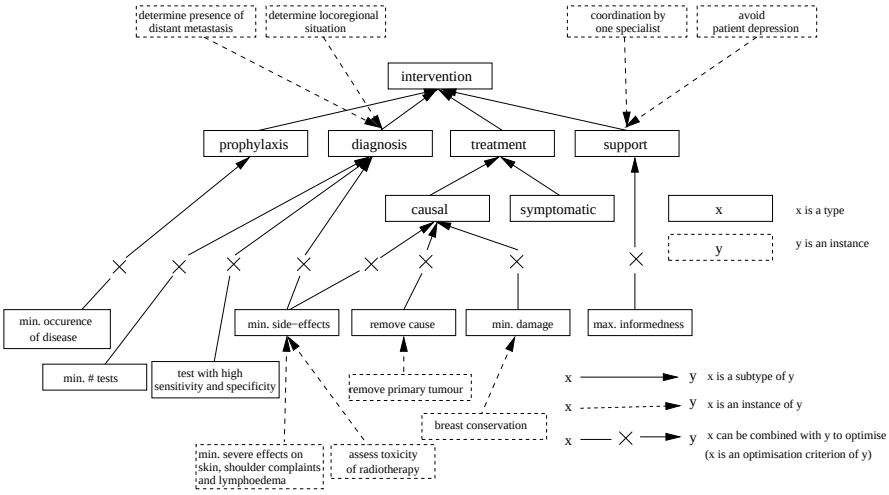


Figure 3.3: A hierarchy of types of interventions and goals, separated by the product operator, with instances related to the breast cancer guideline.

Schematically, we have outlined such an approach in Figure 3.3. At the top, a hierarchy of types of interventions is described, e.g., treatments are intervention and a symptomatic treatment is a treatment. Instances of such interventions for a specific disease are visualised by a dashed line, e.g., determining the locoregional situation is a diagnostic intervention. Each of these types of interventions, which describes the *objective* of the intervention, is associated with certain general goals, e.g., to minimise the number of tests is a general goal related to diagnostic interventions. We have visualised this by relating the objective with a certain goal. It is also possible that it is independent of a specific objective and is therefore associated with all objectives, hence the most general 'intervention' type. For example, minimising the discomfort for a patient is a goal in any intervention that is performed. Finally, for each of the goals, disease specific instances can be added, e.g., to remove the primary tumour is an instance of the general goal of removing the cause in causal breast cancer treatments. Note that this figure is not complete and other goals and interventions are relevant in the treatment of breast cancer.

Examples of medical proof goals

To evaluate the approach, we study a number of examples from the national Dutch breast cancer guideline developed under the supervision of CBO and a protocol from the Dutch Integral Cancer Centre East (IKO) and discuss how the decisions that are being made are related to the hierarchy discussed in the

previous section. In particular, it is the intention of this section to make it plausible that in practice (1) trade-offs are made between certain goals which affect the decision making process, (2) that these goals have a relation to the primary objective, and (3) that there are disease-specific goals related to more general goals that one tries to achieve.

Evidence levels

One particular optimisation in the highest level of the goal hierarchy are evidence levels. This could be viewed as a optimisation of the general concept 'intervention', or more precisely, of every objective that an intervention could have. This means that interventions may be replaced under the condition that their effects have the same evidence levels, taking into account that the other criteria for choosing the intervention are equal. This explains why in the IKO protocol certain therapies recommended by the national guideline are replaced by others in case the recommendation was based on evidence with a low evidence level, e.g., when the original recommendation was based on an expert opinion. An example of such a choice is given in the next subsection.

Trade-offs in treatment selection

Selection of the best possible treatment in medicine can be seen as the process of trading of pros of treatments (e.g., curation) versus cons (e.g., side effects). Below we will give two examples of such trade-offs.

The first example deals with the choice of curative or palliative (i.e., with the aim to relieve or sooth symptoms of the disease) treatment. In case there is a metastasis (secondary cancerous growth elsewhere) of the breast cancer tumour, the guideline recommends a palliative treatment. Although it is not explicitly mentioned in the guideline, there is a strong relationship between this particular choice and a specific intervention. For example, a dissection of the lymph node, in order to reduce the tumour load clearly has a curative goal. Furthermore, even though some treatments are applied for both a curative as well as a palliative objective, such as chemotherapy, it is to be expected that, for instance, quality of life is a more important factor of a palliative treatment. In other words, the (optimality) criteria for selecting certain treatments depend on the objective of the treatment. This explains why we associate the optimality criteria to certain primary objectives.

As a second example, we consider the choice of chemotherapy for a curative treatment and compare the national guideline with the IKO protocol to uncover trade-offs made. There are two leading goals that one tries to achieve, namely to remove the primary tumour as well as minimising the comorbidity. To achieve the highest survival rate, the most important factor is the removal of the tumour cells. However, this induces additional risks because of the toxicity of the treatment which needs to be taken into account. Ideally, one wants to treat high-risk patients with the most effective chemotherapy regimen, in

contrast to low-risk patients where the long-term survival rate benefits from a less toxic therapy. Because the risks are high and the less toxic therapies have been found to be less effective to treat the disease, the guideline recommends a more aggressive regimen (FEC or similar anthracycline containing therapies, which are particular types of chemotherapy). In case this treatment cannot be applied for certain patient groups because of comorbidity, the guideline recommends that other chemotherapies, such as AC or CMF, may be applied. A second indication for choosing AC is pregnancy. However, no evidence exists that FEC is more harmful for the unborn child than AC and for this reason and the superior effectiveness of FEC, the IKO protocol excludes the AC option and recommends to treat pregnant women with FEC. However, in case of a locoregional advanced breast cancer, a type of breast cancer that is irresectable, the IKO protocol considers this an option in context of neo-adjuvant treatment (i.e., to reduce the tumour load to make surgery possible) if after 2 cycles of treatment with FEC the disease is stable or progresses. The slight variations between recommendations of the CBO guideline and the IKO protocol show that there may be differences in recommendation depending on the amount of risks (in terms of direct toxicity) that one wants to take to possibly cure a patient. Moreover, it seems that the lack of evidence for toxicity of FEC for the unborn child is interpreted differently in the two cases, as the guideline chooses a more conservative approach.

Ordering criteria

Alongside the more high-level optimality criteria such as the minimisation of side-effects, there are certain disease-specific optimality criteria, e.g., the ordering of elements of a treatment and diagnostic interventions. There are two reasons for such an ordering. Firstly, an order is needed because certain medical actions render other actions useless. Secondly, because there is evidence that a certain 'optimal' order exists between the medical interventions, i.e., one order yields better results than other orders. For both cases we provide an example.

The first example we look at a diagnostic action, a core biopsy, with a therapeutic action, chemotherapy. A core biopsy provides histological data if the patient did not receive chemotherapy and because of this the core biopsy is performed before chemotherapy. In principle, histological data and information about the receptor status is obtained after (or during) surgery, by examining the resected tumour mass. If the patient received chemotherapy before surgery, then this is no longer possible. In other words, sequences of interventions, i.e., plans or protocols, that perform chemotherapy before a core biopsy have to be considered suboptimal.

In the second example, we see that the reason why a certain order is optimal is in some cases unknown. Medical research has given evidence that postmenopausal women with node-positive, estrogen-receptor-positive breast cancer which receive tamoxifen, a hormonal therapy, after chemotherapy have a higher chance to survive than patients which had received hormonal therapy

before chemotherapy.[21] More studies are needed to find out exactly how these therapies interact and why these they affect each other the way they do, but it is nonetheless important to find out that the guideline complies with this order.

3.5 Conclusions

This chapter was used to introduce some important aspects of clinical guidelines. We have discussed the route from the development of the guideline to the end-product. Several conclusions can be drawn from this. First, modern guidelines are well-structured, making a clear distinction between types of knowledge that are present. The knowledge that we have tried to identify in the breast cancer guideline is extremely diverse, e.g., descriptive knowledge concerning development of the disease, important aspects of interventions, etc. Some of the relevant knowledge discussed is contained in the guideline, some of the medical knowledge is derived from the experts in the working group and partly it is based on subjective criteria, such as the trade-offs that are made in the selection of treatments.

The particular view on quality of guidelines will be formalised in subsequent chapters, in particular Chapters 4 and 5. Trade-offs between treatments will be discussed in Chapter 4; Chapter 5 will focus on ordering criteria between treatments. In Chapter 6, we will look at spotting differences between guidelines and protocols, which was described informally in this chapter. Finally, the analysis of the breast cancer guideline acts as a basis for modelling the concepts underlying clinical guidelines in a logical language. Some of these concepts are further investigated in Chapter 7.

[21] See `http://www.breastcancer.org`

Chapter 4

Verification of Guidelines using Automated Theorem Proving

This chapter explores the use of *automated deduction* for the verification of clinical guidelines. For the rapid development of good quality guidelines it is required that guidelines can be at least partially verified automatically; unfortunately, there is still no verification method that can be readily used by guideline developers. Previously, it was shown that for reasoning about models of medical knowledge, for example in the context of medical expert systems [Lucas, 1993], classical automated reasoning techniques (e.g., [Robinson, 1965a, Wos et al., 1984]) are a practical option. Important for the reasoning about knowledge in clinical guidelines is its temporal nature; time plays a part in the physiological mechanisms as well as in the exploration of possible treatment plans. As far as we know, the application of automated reasoning techniques to guideline knowledge has not been investigated. The guideline we studied to this purpose has a structure similar to other guidelines and the verification principles used have sufficient generality. Thus, the results of the study go beyond the actual guideline studied.

There are two approaches to checking the quality of clinical guidelines using formal methods: (1) the *object-level* approach amounts to the translation of a guideline to a formal language, such as Asbru [Shahar et al., 1998], and subsequently applying program verification or logical methods to analyse the resulting representation for establishing whether certain domain-specific properties hold; (2) the *meta-level* approach, which consists of formalising general requirements to which a guideline should comply, and then investigating whether this is the case for a specific domain. Here we are concerned with the meta-level approach to guideline-quality checking. For example, a good-quality clinical guideline regarding treatment of a disorder should preclude the prescription of redundant drugs, or advise against the prescription of treatment that is less effective than some alternative. An additional goal of this chapter

is to establish how feasible it is to implement such meta-reasoning techniques in existing tools for automated deduction for the purpose of quality checking of a clinical guideline.

The verification approach of this chapter is to translate the modelling formalism, a restricted part of temporal logic, to standard first-order logic. Furthermore, the quality criteria are interpreted in such a way that they can be stated in terms of a monotonic entailment relation. We show that, because of the restricted language needed for the formalisation of the guideline knowledge, the translation is a relatively simple fragment of first-order logic which is amended to fully automated verification. Thus, we show that it is indeed possible, while not easy, to cover the route from informal medical knowledge to a logical formalisation and automated verification.

The meta-level approach that is used here is particularly important for the *design* of clinical guidelines, because it corresponds to a type of reasoning that occurs during the guideline development process. Clearly, quality checks are useful during this process; however, the design of a guideline can be seen as a very complex process where formulation of knowledge and construction of conclusions and corresponding recommendations are intermingled. This makes it cumbersome to do *interactive* (non-automatic) verification of hypotheses concerning the optimal recommendation during the construction of such a guideline, because guideline developers do not generally have the necessary background in formal methods to construct such proofs interactively. Automated theorem proving could therefore be potentially more beneficial for supporting the guideline development process.

In the next section, we start by explaining what clinical guidelines are, and a method for formalising guidelines by temporal logic is briefly reviewed. Then, the formalisation of guideline quality using a meta-level scheme that comes from the theory of abductive diagnosis is described. The guideline on the management of diabetes mellitus type 2 that has been used in the case study is given attention, and its formalisation is given as well. This part is based on [Lucas, 2003]. The main topic of this chapter, an approach to checking the quality of this guideline using the reasoning machinery offered by automated reasoning tools, is then presented. Finally, we discuss what has been achieved, and the advantages and limitations of this approach are brought into perspective. In particular, we will discuss the role of automated reasoning in quality checking guidelines in comparison to alternative techniques such interactive verification.

4.1 Modelling Clinical Guidelines

In Section 3.2.1, we have briefly introduced the Dutch guideline for the treatment of diabetes mellitus type 2. Part of this description includes details about dosage of drugs at specific time periods. As we want to reason about the general structure of the guideline, rather than about dosages or specific time periods,

- Step 1: diet
- Step 2: if Quetelet Index (QI) ≤ 27, prescribe a sulfonylurea drug; otherwise, prescribe a biguanide drug
- Step 3: combine a sulfonylurea drug and biguanide (replace one of these by a α-glucosidase inhibitor if side-effects occur)
- Step 4: one of the following:
  - oral anti-diabetics and insulin
  - only insulin

Figure 4.1: Tiny fragment of a clinical guideline on the management of diabetes mellitus type 2. If one of the steps s_k where $k = 1, 2, 3$ is ineffective, the management moves to step $k + 1$.

we have made an abstraction as shown in Figure 4.1. This guideline fragment is used here as a running example.

Guidelines can be as large as 100 pages; however, the number of recommendations they include are typically few. In complicated diseases, each type of disease is typically described in different sections of a guideline, which provides ways to modularise the formalisation in a natural fashion. For example, consider the breast cancer guideline described in Section 3.3, which is an extensive guideline about breast cancer treatment. However, each of the recommendation, for example, as shown in Figure 3.2, is similar in nature and structure to the abstraction shown in Figure 4.1. In fact, the recommendation shown in Figure 3.2 is much more simple in this specific case. In this sense, the fragment in Figure 4.1 can be lookup upon as one of the recommendations in any guideline whatever its size. On the other hand, clinical protocols are normally more detailed, and the abstraction used here will not be appropriate if one wishes to consider such details in the verification process. For example, in the Protocure project, work was also carried on the verification of a clinical protocol about the management of neonatal jaundice, where there was focus on the levels of a substance in the blood (bilirubin) [ten Teije et al., 2006]. Clearly, in this case abstracting away from substance levels would be inappropriate. Thus, where development of an abstraction of a medical document will be necessary for any verification task, the way it is done is dependent on what is being verified and the nature of the document. Clinical guidelines are typically less specific than protocols, and thus already relatively abstract.

Clinical guidelines regarding a disorder often contain particular parts dealing with the diagnosis, treatment and follow-up of a disorder. In this study, we

restrict ourselves to the treatment part; similar principles apply to the other parts of a guideline.

As discussed in Chapter 3, one way to use formal methods in the context of guidelines is to automatically verify whether or not a clinical guideline fulfils particular properties, such as whether it complies with quality indicators as proposed by healthcare professionals [Marcos et al., 2002]. For example, using particular patient assumptions such as that after treatment the levels of a substance are dangerously high or low, it is possible to check whether this situation does or does not violate the guideline. However, verifying the effects of treatment as well as examining whether a developed clinical guideline complies with global criteria, such as that it avoids the prescription of redundant drugs, or the request of tests that are superfluous, is difficult to impossible if only the guideline text is available. Thus, the capability to check whether a guideline fulfils particular medical objectives may require the availability of more medical knowledge than is actually specified in a clinical guideline. How much additional knowledge is required may vary from guideline to guideline. In the development of the theory below it is assumed that at least some medical background knowledge is required; the extent and the purpose of that background knowledge is subsequently established using the diabetes mellitus type 2 guideline. The development, logical implementation, and evaluation of a formal method that supports this process is the topic of the remainder of this chapter.

The logic we use in this chapter is Linear Temporal Logic (LTL). For syntax and semantics, see Section 2.1.2. Alternative formal languages for modelling medical knowledge are possible. For example, differential equations describing compartment models that are used to predict changes in physiological variables in individual patients have been shown to be useful (e.g., [Magni et al., 2000, Lehmann, 1998]). In the context of clinical reasoning they are less useful, as they essentially concern levels of substances as a function of time and, thus, do not offer the right level of abstraction that we are after.

4.2 Application of Logic to Medical Knowledge

In order to represent the medical knowledge, a specific language is defined in this section. We restrict ourselves to the knowledge which concerns itself with the primary aim of a guideline, which is to have a certain positive effect on a patient. To establish that this is indeed the case, knowledge concerning the physiology of a patient is required. Therefore, it is assumed that two types of knowledge are involved in detecting the violation of good medical practice:

- Knowledge concerning the (patho)physiological mechanisms underlying the disease, and the way treatment influences these mechanisms. The knowledge involved could be causal in nature, and is an example of *object-knowledge*.

- Knowledge concerning good practice in treatment selection; this is *meta-knowledge.*

Below we present some ideas on how such knowledge may be formalised using temporal logic (cf. [Lucas, 1995] for earlier work in the area of formal modelling of medical knowledge).

We are interested in the prescription of drugs, taking into account their mode of action. Abstracting from the dynamics of their pharmacokinetics, this can be formalised in logic as follows:

$$(\mathbf{G}\, d \wedge r) \rightarrow \mathbf{G}(m_1 \wedge \cdots \wedge m_n) \tag{4.1}$$

where d is the name of a drug, r is a (possibly negative or empty) *requirement* for the drug to take effect, and m_k is a mode of action, such as decrease of release of glucose from the liver, which holds at all future times. Note that we assume that drugs are applied for a long period of time, here formalised as 'always'. This is reasonable if we think of the models as finite structures that describe a somewhat longer period of time, allowing the drugs to take effect. Synergistic effects and interactions amongst drugs can also be formalised along those lines, as required by the guideline under consideration. This can be done either by combining their joint mode of action, by replacing d in the formula above by a conjunction of drugs, or by reasoning about modes of actions. As we do not require this feature for the clinical guideline considered in this chapter, we will not go into details. In addition, it is possible to reason about such effects using special purpose temporal logics with abstraction and constraints, such as developed by Allen [Allen, 1983] and Terenziani [Terenziani, 2000] without a connection to a specific field, and by Shahar [Shahar, 1997] for the field of medicine. Thus, temporal logics are expressive enough to cope with extensions to the formalisation as used in this chapter.

The modes of action m_k can be combined, together with an *intention* n (achieving normoglycaemia, i.e., normal blood glucose levels, for example), a particular patient *condition* c, and *requirements* r_j for the modes of action to be effective:

$$(\mathbf{G}m_{i_1} \wedge \cdots \wedge \mathbf{G}m_{i_m} \wedge r_1 \wedge \cdots \wedge r_p \wedge \mathbf{H}c) \rightarrow \mathbf{G}n \tag{4.2}$$

For example, if the mode describes that there is a stimulus to secrete more insulin and the requirement that sufficient capacity to provide this insulin is fulfilled, then the amount of glucose in the blood will decrease.

Good practice medicine can then be formalised as follows. Let $\mathcal{B}$ be background knowledge, $T \subseteq \{d_1, \ldots, d_p\}$ a set of drugs, C a collection of patient conditions, R a collection of requirements and N a collection of intentions which the physician has to achieve. As an abbreviation, the union of C and R, i.e., the variables describing the patient, will be referred to as P, i.e., $P = C \cup R$. Finding an acceptable treatment given such knowledge amounts to finding an explanation, in terms of a treatment, that the intention will be achieved. Find-

ing the best possible explanation given a number of findings is called *abductive* reasoning [Console and Torasso, 1991, Poole, 1990]. We say that a set of drugs T is a *treatment* according to the theory of abductive reasoning if [Lucas, 2003]:

(T1) $\mathcal{B} \cup \mathbf{G}T \cup P \not\models \bot$ (the drugs do not have contradictory effects), and

(T2) $\mathcal{B} \cup \mathbf{G}T \cup P \models N$ (the drugs handle all the patient problems intended to be managed).

One could think of the formula $\mathcal{B} \cup \mathbf{G}T \cup P$ as simulating a particular patient P given a particular treatment T. For each relevant patient groups, these properties can be investigated. If in addition to **(T1)** and **(T2)** condition

(T3) $O_\varphi(T)$ holds, where O_φ is a meta-predicate standing for an optimality criterion or combination of optimality criteria φ,

then the treatment is said to be *in accordance with good-practice medicine*, denoted by $\mathrm{Good}_\varphi(T, P)$.

A typical example of this is subset minimality $O_\subset$:

$$O_\subset(T) \equiv \forall T' \subset T : T' \text{ is not a treatment according to (1) and (2)} \quad (4.3)$$

i.e., the minimum number of effective drugs are being prescribed. For example, if $\{d_1, d_2, d_3\}$ is a treatment that satisfies condition (3) in addition to (1) and (2), then the subsets $\{d_1, d_2\}$, $\{d_2, d_3\}$, $\{d_1\}$, and so on, do not satisfy conditions (1) and (2). In the context of abductive reasoning, subset minimality is often used in order to distinguish between various solutions; it is also referred to in literature as *Occam's razor*. Another definition of the meta-predicate O_φ is in terms of minimal cost O_c:

$$O_c(T) \equiv \forall T', \text{with } T' \text{ a treatment: } c(T') \geq c(T) \quad (4.4)$$

where $c(T) = \sum_{d \in T} cost(d)$; combining the two definitions also makes sense. For example, one could come up with a definition of $O_{\subset,c}$ that among two subset-minimal treatments selects the one that is the cheapest in financial or ethical sense.

The quality criteria that we have presented in this section could also be taken as starting points for *critiquing* (cf. Section 6.2), i.e., criticising clinical actions performed and recorded by a physician (cf. [Miller, 1984] for an early critiquing system), especially if we consider the formalisation of the background knowledge a model for simulating a patient receiving a specific treatment. However, in this chapter, we look at means to criticise the recommendations given by the guideline, which is also the reason why we do not make use of a dataset with patient data, but only of information of hypothetical patients.

4.3 Management of Diabetes Mellitus Type 2

In this section, the medical knowledge with respect to the (patho)physiology of diabetes is formalised. To determine the global quality of the guideline, the background knowledge itself was only formalised so far as required for investigating the usefulness of the theory of quality checking introduced above. The knowledge that is presented here was acquired with the help of a physician, though this knowledge can be found in many standard textbooks on physiology (e.g., [Ganong, 2005, Guyton and Hall, 2000]).

4.3.1 Initial Analysis

It is well known that diabetes type 2 is a very complicated disease: various metabolic control mechanisms are deranged and many different organ systems, such as the cardiovascular and renal system, may be affected by the disorder. Here we focus on the derangement of glucose metabolism in diabetic patients, and even that is nontrivial. To support non-expert medical doctors in the management of this complicated disease in patients, access to a guideline is really essential.

One would expect that as this disorder is so complicated, the diabetes mellitus type 2 guideline is also complicated. This, however, is not the case, as may already be apparent from the guideline fragment shown in Figure 4.1. This indicates that much of the knowledge concerning diabetes mellitus type 2 is missing from the guideline, and that without this background knowledge it will be impossible to spot the sort of flaws we are after. Hence, the conclusion is that a deeper analysis is required; the results of such an analysis are discussed next.

4.3.2 Diabetes Type 2 Background Knowledge

Pathophysiologically, there are two main phenomena, namely, insufficient secretion of the hormone insulin due to a decreased production of insulin by *B cells* in the Langerhans islets of the *pancreas*, and insulin resistance in liver, muscle, and fat tissue. The latter phenomenon sets it apart from other types of diabetes, e.g., type 1 is due to an autoimmune destruction of the B cells. Insulin resistance has been linked to factors such as obesity, in particular around the waste, hypertension, higher amounts of triglycerides, and lower amounts of HDL cholesterol. All overweight individuals have insulin resistance, but only those with an inability to increase B cell production of insulin develop diabetes type 2. A hypothesis is that due to the insulin resistance, the production of insulin by the B cells starts to raise. After some time, the B cells become exhausted, and they are no longer capable of meeting the demands for insulin. As a consequence, the level of glucose in the blood is too high (hyperglycaemia).

Figure 4.2 summarises the most important mechanisms and drugs involved in the control of the blood level of glucose. The protein hormone insulin, which

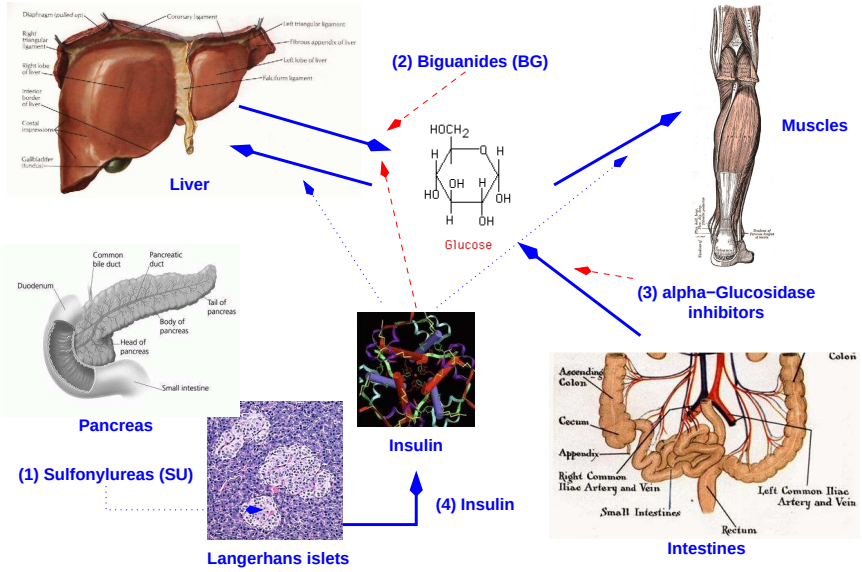


Figure 4.2: Summary of drugs and mechanisms controlling the blood level of glucose; – – →: inhibition, · · · · · →: stimulation.

is produced by the *B cells* in the Langerhans islets of the *pancreas*, has the following major effects:

- it increases the uptake of glucose by the liver, where it is stored as glycogen, and inhibits the release of glucose from the liver;
- it increases the uptake of glucose by insulin-dependent tissues, such as muscle and adipose tissue.

Treatment of diabetes type 2 consists of:

- Use of *sulfonylurea* (SU) drugs, such as tolbutamid. These drugs stimulate the B cells in producing more insulin, and if the cells are not completely exhausted, the hyperglycaemia can thus be reverted to normoglycaemia (normal blood glucose levels).
- Use of *biguanides* (BG), such as metformin. These drugs inhibit the release of glucose from the liver.
- Use of α-*glucosidase inhibitors*. These drugs inhibit (or delay) the absorption of glucose from the intestines.
- Injection of *insulin*. This is the ultimate, causal treatment.

As insulin is typically administered by injection, in contrast to the other drugs, which are normally taken orally, doctors prefer to delay prescribing insulin as

long as possible. Thus, the treatment part of the diabetes type 2 guideline mentions that one should start with prescribing oral antidiabetics (SU or BG, cf. Figure 4.1). Two of these can also be combined if taking only one has insufficient glucose-level lowering effect. If treatment is still unsatisfactory, the guideline suggests to: (1) either add insulin, or (2) stop with the oral antidiabetics entirely and to start with insulin.

From a medical point of view, advice (1) above is somewhat curious. If the oral antidiabetics are no longer effective enough, the B cells could be completely exhausted. Under these circumstances, it does not make a lot of sense to prescribe an SU drug. The guideline here assumes that the B cells are always somewhat active, which may limit the amount of insulin that has to be prescribed. Similarly, prescription of a BG (or a α-glucosidase inhibitor) is justified, as by adding such an oral antidiabetic to insulin, the number of necessary injections can be reduced from twice a day to once a day. It should be noted that, when on insulin treatment, patients run the risk of getting hypoglycaemia (i.e., the level of glucose in the blood is too low), which is a side effect of insulin treatment not mentioned explicitly in the guideline.

The background knowledge concerning the (patho)physiology of the glucose metabolism as described above is formalised using temporal logic, and kept as simple as possible. The specification is denoted by $\mathcal{B}_{\mathrm{DM2}}$:

(1) $\mathbf{G}\,\mathrm{Drug}(\mathit{insulin}) \rightarrow \mathbf{G}(\mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up} \wedge \mathit{uptake}(\mathit{peripheral\text{-}tissues}, \mathit{glucose}) = \mathit{up})$

Insulin increases the uptake of glucose into the liver and into peripheral tissues.

(2) $\mathbf{G}(\mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up} \rightarrow \mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down})$

An increased uptake of glucose into the liver inhibits the release of glucose from the liver.

(3) $(\mathbf{G}\,\mathrm{Drug}(\mathrm{SU}) \wedge \neg \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{exhausted}) \rightarrow \mathbf{G}\mathit{secretion}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{up}$

If the production of insulin is still possible, then a sulfonylurea drug increases the secretion of insulin.

(4) $\mathbf{G}\,\mathrm{Drug}(\mathrm{BG}) \rightarrow \mathbf{G}\mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down}$

Biguanide drugs inhibit the release of glucose from the liver.

(5) $(\mathbf{G}\mathit{secretion}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{up} \wedge \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{subnormal} \wedge \mathrm{QI} \leq 27 \wedge \mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia})) \rightarrow \mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia})$

Diabetic patients with a lower Quetelet-index having a subnormal capacity to produce insulin, will be cured if the B cells are stimulated to produce more insulin.

(6) $(\mathbf{G}\mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down} \wedge$

$$\begin{aligned}&\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{subnormal} \wedge \\ &\mathrm{QI} > 27 \wedge \mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia})) \\ &\rightarrow \mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia})\end{aligned}$$

Diabetic patients with a higher Quetelet-index having a subnormal capacity to produce insulin, will be cured if the release of glucose from the liver is inhibited.

$$\begin{aligned}(7)\ &((\mathbf{G}\mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down} \vee \\ &\quad \mathbf{G}\mathit{uptake}(\mathit{peripheral\text{-}tissues}, \mathit{glucose}) = \mathit{up}) \wedge \\ &\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{nearly\text{-}exhausted} \wedge \\ &\mathbf{G}\mathit{secretion}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{up} \wedge \\ &\mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia})) \\ &\rightarrow \mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia})\end{aligned}$$

Diabetic patients for which the B cells are nearly exhausted will be cured if the B cells are stimulated to secrete more insulin and either the release of glucose from the liver is inhibited or the increase of glucose by peripheral tissues is stimulated.

$$\begin{aligned}(8)\ &(\mathbf{G}\mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up} \wedge \\ &\quad \mathbf{G}\mathit{uptake}(\mathit{peripheral\text{-}tissues}, \mathit{glucose}) = \mathit{up}) \wedge \\ &\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{exhausted} \wedge \\ &\mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia})) \\ &\rightarrow \mathbf{G}(\mathrm{Condition}(\mathit{normoglycaemia}) \vee \mathrm{Condition}(\mathit{hypoglycaemia}))\end{aligned}$$

If both the uptake of glucose into the liver as well as the peripheral tissues is stimulated in diabetic patients for which the B cells exhausted, then the condition will be normo- or hypoglyceamia.

$$\begin{aligned}(9)\ &(\mathrm{Condition}(\mathit{normoglycaemia}) \oplus \mathrm{Condition}(\mathit{hypoglycaemia}) \oplus \\ &\mathrm{Condition}(\mathit{hyperglycaemia})) \wedge \neg(\mathrm{Condition}(\mathit{normoglycaemia}) \wedge \\ &\mathrm{Condition}(\mathit{hypoglycaemia}) \wedge \mathrm{Condition}(\mathit{hyperglycaemia}))\end{aligned}$$

Possible values for the condition of the patient are mutually exclusive.

where $\oplus$ stands for the exclusive OR. Note that when the B cells are exhausted, increased uptake of glucose by the tissues may result not only in normoglycaemia but also in hypoglycaemia. Note that this background knowledge was originally developed for reasoning about the application of an individual treatment. It requires some modification in order to reason about the whole guideline fragment (see Section 4.4.5).

4.3.3 Quality Check

The consequences of various treatment options can be examined using the method introduced in Section 4.2. Hypothetical patients for whom it is the intention to reach a normal level of glucose in the blood (normoglycaemia) and one of the steps in the guideline is applicable in the guideline fragment given in Figure 4.1, are considered, for example:

- Consider a patient with hyperglycaemia due to nearly exhausted B cells. For these patients, the third step of Figure 4.1 is applicable, so we check that:

$$\mathcal{B}_{\mathrm{DM2}} \cup \mathbf{G}\,T \cup \{\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{nearly\text{-}exhausted}\} \cup \{\mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia})\} \models \mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia})$$

holds for $T = \{\mathrm{Drug(SU)}, \mathrm{Drug(BG)}\}$, which also satisfies the minimality condition $O_{\subset}(T)$.

- Prescription of treatment $T = \{\mathrm{Drug(SU)}, \mathrm{Drug(BG)}, \mathrm{Drug(insulin)}\}$ for a patient with exhausted B cells, for which the intended treatment regime is described in the fourth step of Figure 4.1, yields:

$$\mathcal{B}_{\mathrm{DM2}} \cup \mathbf{G}\,T \cup \{\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{exhausted}\} \cup \{\mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia})\} \models \mathbf{G}(\mathrm{Condition}(\mathit{normoglycaemia}) \vee \mathrm{Condition}(\mathit{hypoglycaemia}))$$

In the last case, it appears that it is possible that a patient develops hypoglycaemia due to treatment; if this possibility is excluded from axiom (8) in the background knowledge, then the minimality condition $O_{\subset}(T)$, and also $O_c(T)$, does not hold since insulin by itself is enough to reach normoglycaemia. In either case, good practice medicine is violated, which is to prescribe as few drugs as possible, taking into account costs and side-effects of drugs. Here, three drugs are prescribed whereas only two should have been prescribed (BG and insulin, assuming that insulin alone is too costly), and the possible occurrence of hypoglycaemia should have been prevented.

4.4 Automated Quality Checking

As mentioned in the introduction, we have explored the feasibility of using automated reasoning tools to check the quality of guidelines, in the sense described above.

4.4.1 Motivation for using Automated Reasoning

Several techniques are available for reasoning with temporal logic. Firstly, an automated theorem prover aims at proving theorems without any interaction from the user. This is a problem with high complexity; e.g., for first-order logic, this problem is only semi-decidable. For this reason, *interactive theorem proving* has been used as an alternative, where it is possible and sometimes necessary to give hints to the system. As a consequence, more complicated problems can be handled; however, in the worst case every step of the proof has to be performed manually.

In this chapter, we are concerned with obtaining insight how much of the proof effort can be automated as this would clearly improve the practical usefulness of employing formal methods in the process of guideline development.

In Chapter 5, we use interactive theorem proving for quality-checking guidelines. This was a successful experiment; however, the number of interactions that were needed were still high and a lot of expertise in the area of theorem proving is required for carrying out this task. Furthermore, there has been considerable progress in terms of speed and the size of problems that theorem provers can handle [Pelletier et al., 2002]. In our opinion, these developments provide enough justification to explore the use of automated reasoning techniques in combination with specific strategies.

One of the most important application areas of model finders and theorem provers is program verification. In programs there is a clear beginning of the execution, which makes it intuitive to think about properties that occur after the start of the program. Therefore, it is not surprising that much work that has been done in the context of model finding and theorem proving only deals with the future time modality. However, it is more natural to model medical knowledge with past time operators, i.e., what happened to the patient in the past. It is well-known that formulas with a past-time modality can be mapped to a logical formula with only future time modalities such that both formulas are equivalent for some initial state [Gabbay, 1989]. The main drawback of this approach is that formulas will get much larger in size [Markey, 2003] and as a consequence become much harder to verify in a theorem prover designed for modal logics.

For this reason, we have chosen to use an alternative approach which uses a *relational translation* to map the temporal logic formulas to first-order logic. As primary tools we used the resolution-based theorem prover OTTER [McCune, 2003] and the finite model searcher MACE-2 [McCune, 2001], which take first-order logic with equality as their input. These systems have been optimised for reasoning with first-order logical formulas and offer various reasoning strategies to do this efficiently. For example, OTTER offers the set-of-support strategy and hyperresolution as efficient reasoning methods. There are alternative systems that could have been used; however, it is not the aim of this thesis to compare these systems. OTTER has been proven to be robust and efficient, and has been successfully applied to solve problems of high complexity, for example in the area of algebra [Phillips and Vojtěchovskiý, 2005] and logic [Jech, 1995].

There has been work done to improve the speed of resolution-based theorem provers on modal formulas [Areces et al., 2000], but again, converse modalities such as the past-time operators are not considered. We found that the general heuristics applicable to full first-order logic are sufficient here.

4.4.2 Translation

In order to prove the meta-level quality criteria as defined in Section 4.2, it is necessary to formulate the requirements in the object-language. This seems to contradict the general approach of meta-level verification. However, object-level properties typically do not contain background knowledge concerning the

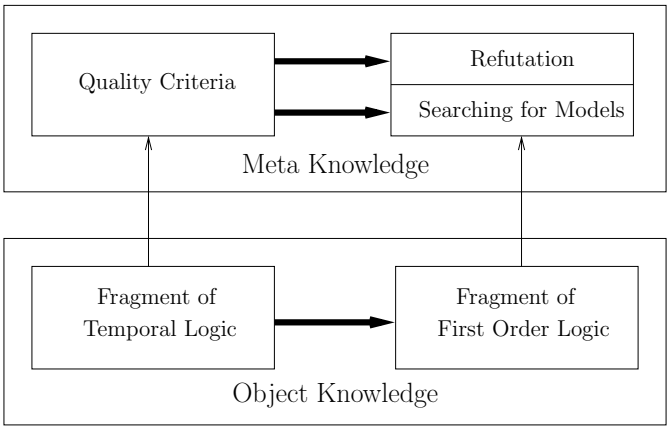


Figure 4.3: Translation of medical knowledge.

validity of what is being verified. For example, the **(T2)** property of Section 4.2 has a clear meaning in terms of the aim of the clinical guideline, which would be lost if stated as an object-level property. Moreover, it is not (directly) possible to deductively derive that something does *not* follow, which is one of the quality requirements.

Figure 4.3 summarises the general approach. We will first give a definition for translating the object knowledge to standard logic and then the translation of the meta-level knowledge will follow.

Translation of Object Knowledge

The background knowledge, as defined in Section 4.3.2, is translated into first order logic. For every function f with two elements in the co-domain, call these $\{c_1, c_2\}$, we introduce a fresh variable p for every element a in the domain such that $f(a) = c_1$ holds iff p holds, and $f(a) = c_2$ holds iff $\neg p$ holds. For example, axiom (2) of $\mathcal{B}_{DM2}$ in Section 4.3.2 is represented by defining '*uptake*(*liver*, *glucose*) = *up*' and '*release*(*liver*, *glucose*) = *up*' as propositions and rewriting this axiom as:

$$\mathbf{G}(\text{`}\mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up}\text{'} \rightarrow \neg(\text{`}\mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{up}\text{'}))$$

where the quotes pieces of text are propositional variables. In order to accomplish this, we do the following. For the *capacity* function, a function with three elements in its co-domain, we add a proposition p_x for each expression *capacity*(*b-cells*, *insulin*) = x and an axiom saying that each pair of these propositions are mutually exclusive. Finally, the term QI $>$ 27 is interpreted as a proposition as well, i.e., we do not reason about the numerical value of QI.

Technically, this translation is not required, since we could extend the trans-

lation below to full first-order temporal logic. In practice, however, we would like to avoid additional complexity from first-order formulas during the automated reasoning.

The relational translation (e.g., [Moore, 1979, Areces et al., 2000, Schmidt and Hustadt, 2003]) $\mathsf{ST}_t(\varphi)$, also referred to as the *standard translation*, translates a propositional temporal logical formula φ into a formula in a first-order logic with (time-indexed) unary predicate symbols P for every propositional variable p and one binary predicate $>$. It is defined as follows, where t is an individual variable standing for time:

$$\begin{array}{lll}
\mathsf{ST}_t(p) & \text{iff} & P(t) \\
\mathsf{ST}_t(\neg\varphi) & \text{iff} & \neg\mathsf{ST}_t(\varphi) \\
\mathsf{ST}_t(\varphi \wedge \psi) & \text{iff} & \mathsf{ST}_t(\varphi) \wedge \mathsf{ST}_t(\psi) \\
\mathsf{ST}_t(\mathbf{G}\varphi) & \text{iff} & \forall t' \; (t \not> t' \rightarrow \mathsf{ST}_{t'}(\varphi)) \\
\mathsf{ST}_t(\mathbf{H}\varphi) & \text{iff} & \forall t' \; (t > t' \rightarrow \mathsf{ST}_{t'}(\varphi))
\end{array}$$

Note that the last two elements of the definition give the meaning of the **G** modality and its converse, the **H** modality. For example, the formula $\mathbf{G}(p \rightarrow \mathbf{P}p)$ translates to $\forall t_2 \; (t \not> t_2 \rightarrow (P(t_2) \rightarrow \exists t_3 \; (t_2 > t_3 \wedge P(t_3))))$. A listing of the translated knowledge can be found in Appendix B.1. It is straightforward to show that a formula in temporal logic is satisfiable if and only if its relational translation is. Also, recall that we use set union to denote conjunction, thus $\mathsf{ST}_t(\Gamma \cup \Delta)$ is defined as $\mathsf{ST}_t(\Gamma) \wedge \mathsf{ST}_t(\Delta)$.

In the literature a functional approach to translating modal logic has appeared as well [Ohlbach, 1988], which relies on a non-standard interpretation of modal logic and could be taken as an alternative to this translation.

Translation of Meta-level Knowledge

Again, we consider the criteria for good practice medicine and make them suitable for use with the automated reasoning tools. In order to stress that we deal with provability in these tools, we use the '$\vdash$' symbol instead of the '$\models$' (validity) symbol. We say that a treatment T is a treatment complying with the requirements of good practice medicine iff:

(T1′) $\mathsf{ST}_t(\mathcal{B} \cup \mathbf{G}T \cup C \cup R) \nvdash \bot$

(T2′) $\mathsf{ST}_t(\mathcal{B} \cup \mathbf{G}T \cup C \cup R \cup \neg N) \vdash \bot$

(T3′) $\forall T' \subset T : T'$ is not a treatment according to (1) and (2)

Criterion **(T3′)** is a specific instance of **(T3)**, i.e., subset minimality as explained in Section 4.2 (Equation 4.3). As the relational translation preserves satisfiability, these quality requirements are equivalent to their unprimed counterparts in Section 4.2. To automate this reasoning process we use MACE-2 to verify **(T1′)**, OTTER to verify **(T2′)**, and **(T3′)** can be seen as a combination of both for all subsets of the given treatment.

4.4.3 Results

In this subsection we will discuss the actual implementation in OTTER and some results obtained by using particular heuristics.

Resolution Strategies

An advantage that one gains from using a standard theorem prover that a whole range of different resolution rules and search strategies are available and can be varied depending on the problem. OTTER uses the set-of-support strategy [Wos et al., 1965] as a standard strategy. In this strategy the original set of clauses is divided into a *set-of-support* and a *usable* set such that in every resolution step at least one of the parent clauses has to be a member of the set-of-support and each resulting resolvent is added to the set-of-support.

Looking at the structure of the formulas in Section 4.3, one can see that formulas are of the form $L_1 \wedge \cdots \wedge L_n \rightarrow L_{n+1}$, where almost all L_i, $i = 1, \ldots, n+1$, are positive literals. Hence, we expect the occurrence of mainly negative literals in our clauses, which can be exploited by using negative hyperresolution (neg_hyper for short) [Robinson, 1965a] in OTTER. With this strategy a clause with at least one positive literal is resolved with one or more clauses only containing negative literals (i.e., negative clauses), provided that the resolvent is a negative clause. The parent clause with at least one positive literal is called the *nucleus*, and the other, negative, clauses that resolve at least one literal in the nucleus are referred to as the *satellites*.

Verification of Treatments

The ordering predicate $>$ that was introduced in Section 4.4.2 was defined by adding axioms of irreflexivity, anti-symmetry, and transitivity. We did not find any cases where the axiom of transitivity was required to construct the proof, which can be explained by the low modal depth of our formulas. As a consequence, the axiom was omitted with the aim to improve the speed of theorem proving. Furthermore, because we lack the next step modality, we did not need to axiomatise a subsequent time point. Experiments showed that this greatly reduces the amount of effort for the theorem prover.

We used OTTER to perform the two proofs which are instantiations of **(T2′)**. First, we consider a patient with hyperglycaemia due to nearly exhausted B cells and prove:

$$\begin{aligned}\mathsf{ST}_0(&\mathcal{B}_{\mathrm{DM2}} \cup \mathbf{G}\,T \cup \{\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{nearly\text{-}exhausted}\} \\ &\cup \{\mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia})\} \\ &\cup \{\neg\mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia})\}) \vdash \bot\end{aligned}$$

where $T = \{\mathrm{Drug(SU)}, \mathrm{Drug(BG)}\}$, i.e., step 3 of the guideline (see Figure 4.1). Note that we use '0' here to represent the current time point. This property was proven using OTTER in 62 resolution steps with the use of the negative

hyperresolution strategy for which roughly 7000 candidate clauses were generated. A summary of this proof can be found in Appendix A.1.

Similarly, we could prove that given a treatment T such that $T = \{\text{Drug(SU)}, \text{Drug(BG)}, \text{Drug(insulin)}\}$ for a patient with exhausted B cells, as suggested by the guideline in step 4, it follows that:

$$\begin{aligned}&\mathsf{ST}_0(\mathcal{B}_{\text{DM2}} \cup \mathbf{G}\,T \cup \{\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{exhausted}\} \cup \\ &\quad \{\mathbf{H}\text{Condition}(\mathit{hyperglycaemia})\} \cup \\ &\quad \{\neg(\mathbf{G}(\text{Condition}(\mathit{normoglycaemia}) \vee \text{Condition}(\mathit{hypoglycaemia})))\}) \\ &\quad \vdash \bot\end{aligned}$$

However, if we take $T = \{\text{Drug(insulin}\}$, the same holds, which shows that, as already mentioned in Section 4.3.3, that even if we ignore the fact that the patient may develop *hypoglycaemia*, the treatment is not minimal. Compared to the previous property, a similar magnitude of complexity in the proof was observed, resulting in 52 resolution steps. In both cases, the amount of CPU time is within a few minutes on a modern pc.

Using Weighting

One possibility to improve the performance is by using term ordering strategies. This will be explained below. We first give a motivating example why this is particularly useful for this class of problems. Consider the following example taken from [Areces et al., 2000]. Suppose we have the formula $\mathbf{G}(p \rightarrow \mathbf{F}p)$. Proving that this is satisfiable amounts to proving that the following two clauses are satisfiable (where variables t_i are universally quantified and function f arises due to Skolemnisation):

1. $0 > t_1 \vee \neg P(t_1) \vee t_1 \not> f(t_1)$
2. $0 > t_2 \vee \neg P(t_2) \vee P(f(t_2))$

which is the relational translation of this formula. It can be observed, that although we have two possibilities to resolve these two clauses, for example on the P literal, this is useless because the negative P literal is only bound by the **G**-operator while the positive P literal comes from a formula at a deeper modal depth under the **F**-operator. For example, suppose we resolve these $\neg P(t_1)$ and $P(f(t_2))$ and rename t_2 to t, which generates the clause:

$$0 > f(t) \vee f(t) \not> f(f(t)) \vee 0 > t \vee \neg P(t)$$

and with (2) again we have:

$$0 > f(f(t)) \vee f(f(t)) \not> f(f(f(t))) \vee 0 > f(t) \vee c > t \vee \neg P(t)$$

etc. In this way, we can generate many new increasingly lengthy clauses. Clearly, these nestings of the Skolem functions will not help to find a contra-

Weights	binary resolution	negative hyperresolution
$(0, 1)$	17729	6994
$(1, 0)$	13255	6805
$(1, 1)$	39444	7001
$(1, -1)$	13907	6836
$(2, -2)$	40548	7001
$(2, -3)$	16606	6805
$(3, -4)$	40356	7095
$(3, -5)$	27478	7001

Figure 4.4: Generated clauses to prove an instance of property **T2′** depending on weights (x, y) for the ordering relation on time.

diction more quickly if the depth of the modalities in the formulas that we have is small, as the new clauses are similar to previous clauses, except that they describe a more complex temporal structure.

In OTTER the weight of the clauses determines which clauses are chosen from the set-of-support and usable list to become parents in a resolution step. In case the weight of two clauses is the same, there is a syntactical ordering to determine which clause has precedence, which is called the Knuth-Bendix Ordering (KBO) [Knuth and Bendix, 1970]. As the goal of resolution is to find an empty clause, lighter clauses are preferred. By default, the weight of a clause is the sum of all occurring symbols (i.e., all symbols have weight 1) in the literals. As we have argued, since the temporal structure of our background knowledge is relatively simple, nesting Skolem functions will not help to find such an empty clause. Therefore it can be of use to manually change the weight of the ordering symbol, which is done in OTTER by a tuple (x, y) for each predicate, where x is multiplied by the sum of the weight of its arguments and is added to y to calculate the new weight of this predicate. For example, if $x = 2$ and $y = -3$, then $v > w$ has a total weight of $2 + 2 - 3 = 1$, and $f(f(c)) > f(d)$ has a weight of $2 * 3 + 2 * 2 - 3 = 7$.

See Figure 4.4 where we show results when we applied this for some small values for x and y for both binary and negative hyperresolution. What these numbers show (similar results were obtained for the other properties) is that the total weight of the ordering predicate should be smaller than the weight of other, unary, predicates. Possibly somewhat surprisingly, the factor x should not be increased too much. Furthermore, in the case of a negative hyperresolution strategy the effect is minimal.

4.4.4 Disproofs

MACE-2 (Models And CounterExamples) is a program that searches for small finite models of first-order statements using a Davis-Putman-Loveland-Logemann decision procedure [Davis and Putman, 1969, Davis et al., 1962] as

```
> :                          Condition(hyperglycaemia) :
    t | 0 1                          t 0 1
    ---+----                         -------
    0 | F T                            T T
    1 | F F

Drug(SU):                          Drug(BG) :
   t 0 1                              t  0 1
 -------                              -------
     F F                                T F

capacity(b-cells, insulin) = nearly-exhausted :
      t | 0 1
      -------
          T T
```

Figure 4.5: Snippet from a MACE-2 generated model. It lists the truth value of all the unary predicates given each element of the domain (i.e., the time points '0' and '1') and every combination of domain elements for the binary predicate $<$. Truth values are denoted by T (true) and F (false).

its core. Because of the relative simplicity of our temporal formulas, it is to be expected that counterexamples can be found rapidly, exploring only few states. Hence, it could be expected that models are of the same magnitude of complexity as in the propositional case and this was indeed the case. In fact, the countermodels that MACE-2 found consist of only two elements in the domain of the model.

The first property we checked corresponded to checking whether the background knowledge augmented with patient data and a therapy was consistent, i.e., criterion **(T1′)**. Consider a patient with hyperglycaemia due to nearly exhausted B cells. We used MACE-2 to verify consistency:

$$\mathsf{ST}_0(\mathcal{B}_{\mathrm{DM2}} \cup \mathbf{G}\, T \cup \{\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{nearly\text{-}exhausted}\} \cup \{\mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia})\}) \nvdash \bot$$

for $T = \{\mathrm{Drug(SU)}, \mathrm{Drug(BG)}, \mathrm{Drug(insulin)}\}$. From this it follows that there is a model if $T = \{\mathrm{Drug(SU)}, \mathrm{Drug(BG)}\}$ and consequently we have verified **(T1′)**.

Similarly, we found that for all $T \subset \{\mathrm{Drug(SU)}, \mathrm{Drug(BG)}\}$, it holds that:

$$\mathsf{ST}_0(\mathcal{B}_{\mathrm{DM2}} \cup \mathbf{G}\, T \cup \{\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{nearly\text{-}exhausted}\} \cup \{\mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia})\} \cup \{\neg\mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia})\}) \nvdash \bot$$

i.e., it is consistent to believe the patient will not have normoglycaemia if

less drugs are applied, which violates **(T2)** for these subsets. So indeed the conclusion is that the treatment complies with **(T3′)** and thus complies with the criteria of good practice medicine. See for example Figure 4.5, which contains a small sample of the output that MACE-2 generated. The output consists of a first-order model with two elements in the domain, named '0' and '1', and an interpretation of all predicates and functions in this domain. It shows that it is consistent with the background knowledge to believe that the patient will continue to suffer from hyperglycaemia if one of the drugs is not applied. Note that the model specifies that biguanide is applied at the first time instance, as this is not prohibited by the assumptions.

Finally, consider the treatment $T = \{\text{Drug(SU)}, \text{Drug(BG)}, \text{Drug(insulin)}\}$ for a patient with exhausted B cells, and suppose we exclude the patient developing hypoglycaemia from the background knowledge, we can show that:

$$\begin{aligned}&\mathsf{ST}_0(\mathcal{B}_{\text{DM2}} \cup \mathbf{G}\,T \cup \{\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{exhausted}\} \cup \\ &\quad \{\mathbf{H}\text{Condition}(\mathit{hyperglycaemia})\} \cup \\ &\quad \{\mathbf{G}(\text{Condition}(\mathit{normoglycaemia}))\})) \nvdash \bot\end{aligned}$$

so the patient may be cured with insulin treatment, even though this is not guaranteed as Condition(*normoglycaemia*) does not deductively follow from the premises. However, as discussed in Section 4.4.3, it is possible to prove the property **(T1)** when $T = \{\text{Drug(insulin)}\}$ and thus **(T3′)** does not hold in this case and as a consequence the guideline does not comply with the quality requirements as discussed in Section 4.3.3.

4.4.5 Plan Structure

So far, we have not considered the order in which treatments are being considered and executed. In this subsection, we look at the problem of reasoning about the order of treatments described in the treatment plan listed in Figure 4.1.

Formalisation

In order to reason about a sequence of treatments, additional formalisation is required. The background knowledge was developed for reasoning about an individual treatment, and therefore, is parameterised for the treatment that is being applied. We postulate $\mathcal{B}_{\text{DM2}}$, parameterised by s, where s is a certain step in the protocol, i.e., $s = 1, 2, 3, 4$ (cf. Figure 4.1; for example $s = 1$ corresponds to diet). The first axiom is then described by:

$$\forall s\ (\mathbf{G}\,\text{Drug}(\mathit{insulin}, s) \rightarrow \mathbf{G}(\mathit{uptake}(\mathit{liver}, \mathit{glucose}, s) = \mathit{up}))$$

The complete description of this background knowledge is denoted by $\mathcal{B}'_{\text{DM2}}$ The reason for this is that the '**G**' modality ranges over the time period of an individual treatment, rather than the complete time frame. Similarly, the

patient can be described, assuming the description of the patient description does not change, by $\forall s\ P(s)$, where P is a parameterised description of the patient. For example, in diabetes, it may be assumed that the Quetelet index does not change; however, the condition generally does change due to the application of a treatment.

The guideline as shown in Figure 4.1 is modelled in two parts. First, we need to specify which treatment is administered in each step of the protocol. Second, the transition of one step to the next has to be specified. The former is modelled as a conjunction of treatments for each step of the guideline. For example, in the initial treatment step (i.e., step 1) only 'diet' is applied, hence, the following is specified:

$$\mathbf{G}\ \mathit{diet}(1)$$

In general, for treatment $T(s)$ in step s, we write $\mathbf{G}T(s)$. Here s is a meta-variable standing for the step in the protocol, i.e., it is a ground atom in the concrete specification of the protocol. Object-level variables can be recognised by the fact that they are bounded by quantification. For example, $T(s)$ is a ground term in the actual specification, while $\forall s\ T(s)$ is not. In this notation, we will refer to the set of treatment prescriptions for each step and all patient groups $P(s)$ as $\mathcal{D} = \bigcup_s P(s) \rightarrow \mathbf{G}\,T(s)$.

The second part of the formalisation concern the change of treatments, which is formalised by a predicate $\mathit{control}(s)$ that describes which step of the guideline is reached. Recall from Figure 4.1, that treatments are stopped in case they fail, i.e., when they do not result in the desired effect. This change of control can be described in the meta-language as:

$$\mathcal{B} \cup \mathbf{G}\,T(s) \cup P(s) \not\models N(s) \Rightarrow \mathit{control}(s+1) \tag{4.5}$$

for all steps s, i.e., if the intention cannot be deduced, then we move to a subsequent step. We will refer to this axiom as the *control axiom* $\mathcal{C}$. Note that $\neg N(s)$ cannot be deduced from the background knowledge, due to its causal nature. However, clearly, in the context of automatic reasoning, it is useful to reason about the theory deductively. To be able to do this, one can use the so-called completed theory, denoted as COMP(Γ), where Γ is some first-order theory. The COMP function is formally defined in [Clark, 1978] for general first-order theories. For propositional theories one can think of this function as replacing implication with bi-implications, for example, COMP($p \rightarrow q$) $= p \leftrightarrow q$ and COMP($\{p \rightarrow q,\ p \rightarrow r\}$) $= p \leftrightarrow (q \vee r)$. By the fact that the temporal formulas can be interpreted as first-order sentences, we have for example:

$$\begin{aligned}&\mathrm{COMP}(\mathbf{G}\,\mathrm{Drug}(\mathit{insulin}) \rightarrow \mathbf{G}(\mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up})\\ &= \mathbf{G}\,\mathrm{Drug}(\mathit{insulin}) \leftrightarrow \mathbf{G}(\mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up}\end{aligned}$$

This can be extended for the whole set of axioms of diabetes. The relevance

of this operator for this chapter, is that abductive reasoning can be seen as deductive reasoning in this completed theory [Console et al., 1991]. In the following section, we introduce an extension to this idea for the restricted part of temporal logic described in Section 4.2. These results are based on a direct application of work done by Stärk [Stärk, 1994]. Then, we will apply those results to the above formalisation.

Completion

An important resolution strategy is SLD resolution which is linear resolution with a selection function for Horn clauses, i.e., clauses with at most one positive literal (for a definition see for example [Lucas and van der Gaag, 2005]). SLD resolution is sound and refutation complete for Horn clause logic. It is refutation complete in the sense that if one would use a breadth-first strategy through the tree of all SLD derivations, a finite SLD refutation will be found if the set of Horn clauses is unsatisfiable. Below, as a convenience, we will write that we derive ψ from φ using SLD resolution iff there is an SLD refutation from $\varphi \wedge \neg\psi$.

SLDNF resolution augments SLD resolution with a so-called 'negation as failure' (NAF) rule [Clark, 1978]. The idea is in order to prove $\neg A$, try proving A; if the proof succeeds, then the evaluation of $\neg A$ fails; otherwise, if A fails on every evaluation path, then $\neg A$ succeeds. The latter part of this strategy is not a standard logical rule and could be described formally as, given some theory Γ, if $\Gamma \nvdash A$ then $\Gamma \vdash \neg A$ is concluded. It must be noted that the query A must be grounded. This type of inference is featured in logic programming languages such as PROLOG, although most implementations also infer the negation as failure for non-ground goal clauses.

This type of resolution is used here to show that a completed theory can be used in a deductive setting to reason about the meta-theory. In particular, in [Stärk, 1994], this is used to show that a certain class of programs have the property that if a proposition deductively follows from that program, then there is a successful SLDNF derivation. This is shown by so-called input/output specifications, which are given by a set of mode specifications for every predicate. A mode specification for a predicate says which arguments are input arguments and which arguments are output arguments; other arguments are called normal arguments. Given an input/output specification a program must be written in such a way that in a computed answer the free variables of the output terms are contained in the free variables in the input terms. Furthermore, the free variables of a negative literal must be instantiated to ground terms during a computation. For example, the following well-known logic program

$$\mathsf{append}([], L, L).$$
$$\mathsf{append}(L_1, L_2, L_3) \rightarrow \mathsf{append}([X|L_1], L_2, [X|L_3]).$$

has two mode specifications. Either the first two arguments are input arguments resulting in a concatenation of the two lists in the output argument, or, the first two arguments can act as output arguments resulting in the decomposition of the third argument into two lists.

In the following, we will write all ground atoms without arguments, e.g., we denote A when we mean $A(c)$, where c is some constant, unless the constant is relevant. We then prove the following lemma.

Lemma 4.1. *If* $\mathrm{COMP}(\Gamma) \models \neg A_g$*, where* Γ *is a formula of the form:*

$$\forall s \forall t\ (A_0(s) \wedge \cdots \wedge A_n(s) \wedge A_{n+1}(t,s) \wedge \cdots \wedge A_m(t,s) \rightarrow A_k(t,s))$$

where A_i *are all positive atoms and* A_g *is any ground atom, then there exists an SLDNF derivation of* $\neg A_g$ *for theory* Γ*.*

A proof can be found in Appendix A.2. Note here that Γ only contains Horn clauses. Further note that the relation between the completed theory and SLDNF derivation holds for a much more elaborate class of formulas [Stärk, 1994]. Hence, this result could be generalised to a more elaborate temporal descriptions. However, the fact that we are dealing with Horn clauses yields the following property, which is the main result of this section.

Theorem 4.1. *If* Γ *is in the form assumed in Lemma 4.1 and* A *is again any ground atom, then* $\mathrm{COMP}(\Gamma) \models \neg A$*, then* $\Gamma \not\models A$*.*

Proof. Suppose $\mathrm{COMP}(\Gamma) \models \neg A$. Then by Lemma 4.1 it holds that $\neg A$ is derived by SLDNF resolution from Γ. From the definition of SLDNF derivation either $\neg A$ holds by SLD resolution or a derivation for A fails. In either way, it follows from the soundness of SLD resolution that deriving A from Γ using SLD resolution will fail. Since each of the clauses is Horn and SLD resolution is complete for these Horn clauses, it follows that $\Gamma \not\models A$. □

Implementation

The result of Theorem 4.1 is used to investigate the completion of a restricted subset of temporal logic. To simplify matters, we introduce the following assumptions. First, the **H** operator is omitted. In this case, this is justified as this operator only plays a role to denote the fact that the patient suffers from hyperglycaemia and plays no role in the temporal reasoning. Hence, we have a (propositional) variable that expresses exactly the fact that in the past the condition was hyperglycaemic. Second, as there is no reasoning about the past, we may translate $\mathbf{G}\varphi$ to $\forall t\ \varphi(t)$. Finally, we only make a distinction between whether the glucose level is decreasing or not, i.e., we abstract from the difference between normo- and hypoglycaemia. Furthermore, we assume that the mutual exclusion of values for capacity is omitted and part of the description of the patient, i.e., a patient with $\mathrm{QI} > 27$ is now described by $\{\mathrm{QI} > 27, \neg(\mathrm{QI} \leq 27)\}$. We will refer to these translation assumptions in

Temporal Logic	First-order Logic
$A_1 \wedge \cdots \wedge A_n \wedge \mathbf{G}\, A_{n+1} \wedge$ $\cdots \wedge \mathbf{G}\, A_m \rightarrow \mathbf{G}\, A_i$	$\forall t\ (A_1 \wedge \cdots \wedge A_n \wedge A_{n+1} \wedge$ $\cdots \wedge A_m \rightarrow A_i(t))$
$\mathbf{G}(A_1 \wedge \cdots \wedge A_n \rightarrow A_i)$	$\forall t\ (A_1(t) \wedge \cdots \wedge A_n(t) \rightarrow A_i(t))$
$\mathbf{G}\, A_i,\ A_i$	$A_i(t),\ A_i$
$\neg \mathbf{G}\, A_i$	$\neg A_i$

Figure 4.6: The type of temporal formulas and their translation, where the Skolem constants describing time instances are omitted.

addition to the translation to first-order logic described in Section 4.4.2 as ST'_t. Furthermore, let COMP(Γ) be understood as the formula which is equivalent according to ST to COMP($\mathsf{ST}'_t(\Gamma)$) whenever Γ is a theory in temporal logic. Note that this abstraction is sound, in the sense that anything that is proven with respect to the condition of the patient by the abstracted formulas can be proven from the original specification.

Let p_i be a patient characteristic, d a drug, and l_i either a patient characteristic or drug. The temporal formulas that are allowed are listed in Figure 4.6. We claim that each temporal formula is an instance of a temporal formula mentioned in Figure 4.6, universally quantified by a step s, except for the last goal clause which is grounded. The background knowledge can be written in terms of the first and second clause, taken into account that axiom (7) can be rephrased to two clauses of the first type and we need to make sure that each literal is coded as a positive atom. This is a standard translation procedure that can be done for many theories and is described in e.g., [Shepherdson, 1987, p. 23]. Axiom (3) needs to be rewritten for each of the cases of capacity implied by the negated sub-formula. For each drug and patient characteristic in the hypothesis, the third clause of Figure 4.6 applies. A goal is an instance of the fourth clause of Figure 4.6. As the first three clauses are Horn, Theorem 4.1 can be instantiated for the background knowledge, which yields:

Theorem 4.2. COMP($\mathcal{B}'_{DM2} \cup \boldsymbol{G}T(s) \cup P(s)$) $\models \neg N(s)$ *implies* $\mathcal{B}'_{DM2} \cup P(s) \cup \boldsymbol{G}T(s) \not\models N(s)$.

This states that, if the completed theory implies that the patient will not have normoglycaemia, then this is consistent conclusion with respect to the original specification, for any specific step described by s. Therefore, there is no reason to assume that T is the correct treatment in step s. This result is applied to the control axiom $\mathcal{C}$ as described in Section 4.4.5, i.e., formula 4.5. If we were to deduce that

$$\text{COMP}(\mathcal{B} \cup \mathbf{G}\, T(s) \cup P(s)) \models \neg N(s)$$

then, assuming the literals are in a proper form required by Theorem 4.2, this

implies that

$$\mathcal{B} \cup \mathbf{G}\, T(s) \cup P(s) \not\models N(s)$$

Thus, we postulate the following axiom describing the change of control, denoted by $\mathcal{C}'$

$$\mathrm{COMP}(\mathcal{B} \wedge \mathbf{G}\, T(s) \wedge P(s)) \wedge \neg N(s) \rightarrow \mathit{control}(s+1)$$

The axioms $\mathcal{D}$ (cf. Section 4.4.5) and $\mathcal{C}'$ are added to the guideline formalisation in order to reason about the structure of the guideline.

To investigate the quality of the treatment sequence, a choice of quality criteria has to be chosen. Similarly to individual treatments, notions of optimality could be studied. Here, we investigate the property that for each patient group, the intention should be reached at some point in the guideline. For the diabetes guideline, this is formalised as follows:

$$\mathcal{B}'_{\mathrm{DM2}} \cup \mathcal{D} \cup \forall s\ P(s) \models \exists s\ N(s)$$

As we restrict ourselves to a particular treatment described in step s, this property is similar to the property proven in Section 4.4.3. However, it is possible that the control never reaches s for a certain patient group, hence, using the knowledge described in $\mathcal{C}$, it is also important to verify that this step is indeed reachable, i.e.,

$$\mathcal{B}'_{\mathrm{DM2}} \cup \mathcal{D} \cup \forall s\ P(s)) \cup \mathcal{C}' \models \exists s\ (N(s) \wedge \mathit{control}(s))$$

We experimented with this idea and verified a number of properties for different groups. For example, assume $P(s) = \{\mathit{capacity}(\mathit{liver}, \mathit{glucose}, s) = \mathit{exhausted}, \mathrm{QI}(s) \leq 27, \mathbf{H}\,\mathrm{Condition}(\mathit{normoglycaemia})\}$ (note the $\mathbf{H}$ operator is abstracted from the specification) then:

$$\mathcal{B}'_{\mathrm{DM2}} \cup \mathcal{D} \cup \forall_s\ P(s) \cup \mathcal{C}' \models \mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia}, 3) \wedge \mathit{control}(3)$$

i.e., the third step will be reached and in this step the patient will be cured. This was implemented in OTTER using the translation as discussed in the previous subsection. As the temporal reasoning is easier due to the abstraction that was made, the proofs are reasonably short. For example, in the example above, the proof has length 25 and was found immediately.

4.5 Conclusions

The quality of guideline design is for the largest part based on its compliance with specific treatment aims and global requirements. We have made use of a logical meta-level characterisation of such requirements, and with respect to the requirements use was made of the theory of abductive,

diagnostic reasoning, i.e., to diagnose potential problems with a guideline [Lucas, 1997, Lucas, 2003, Poole, 1990]. In particular, what was diagnosed were problems in the relationship between medical knowledge, and suggested treatment actions in the guideline text and treatment effects; this is different from traditional abductive diagnosis, where observed findings are explained in terms of diagnostic hypotheses. Moreover, we were able to reason about the necessity of applying certain interventions using the optimality criterion. This method allowed us to examine fragments of a guideline and to prove properties of those fragments. Furthermore, we have succeeded in proving a property using the structure of the guideline, namely that the blood glucose will go down eventually for all patients if the guideline is followed (however, patients run the risk of developing hypoglycaemia, which should be avoided).

In Chapter 5, we will use a tool for interactive program verification, (KIV [Reif, 1995]) for the purpose of quality checking of the diabetes type 2 guideline and further develop quality criteria for clinical guidelines. The main advantage of the use of interactive theorem proving is that the resulting proofs were relatively elegant as compared to the solutions obtained by automated resolution-based theorem proving. This may be important if one wishes to convince the medical community that a guideline complies with their medical quality requirements and to promote the implementation of such a guideline. However, to support the *design* of guidelines, this argument is of less importance. A push-button technique would there be more appropriate. The work that needs to be done to construct a proof in an interactive theorem prover would severely slow down the development process as people with specialised knowledge are required.

Chapter 5

Verification of Guidelines using Interactive Theorem Proving

In this chapter, we explore ways to verify clinical guidelines using *interactive theorem proving*. Interactive theorem proving is a technique for gradually proving properties by means of computerised reasoning under the guidance of a human, i.e., usually interactions between the system and the user are mandatory; these may ultimately lead to a proof if the interactions guide the system into the right direction. As mentioned in the conclusions of the previous chapter, a significant advantage of the use of interactive theorem proving is that the resulting proofs are relatively understandable, by means of a proof tree, compared to the solutions obtained by automated resolution-based theorem proving. This is important if one wishes to convince the medical community that a guideline does or does not comply to their quality requirements. The second advantage is that interactive theorem proving can be applied to problems of almost arbitrary complexity, provided one is prepared to invest significant amounts of time into the manual work.

In this chapter, we employ the methodology of the previous chapter with respect to checking the quality of individual treatments, which may consist of multiple simultaneous actions, such as the prescription of multiple drugs, using interactive theorem proving. Furthermore, the semantics of a guideline representation language, Asbru, are reviewed, and the methodology is extended to be able to handle Asbru-based guideline models. We specifically focus on the modelling of quality requirements for such types of model, which is an extension of the ideas presented in Chapter 4, as we need to integrate guideline knowledge as represented in a guideline representation language with medical background knowledge. Finally, the complete diabetes mellitus type 2 guideline is verified on the basis of these quality criteria.

5.1 Checking the Quality of Individual Treatments

In this section, the quality of individual treatments is investigated using the interactive theorem prover KIV. We first review the techniques used in this chapter, and, subsequently, we discuss some of the peculiarities that arise using these techniques in comparison to the fully automated techniques presented in Chapter 4. This implies that we will reconsider some of the material of the previous chapter but now look upon it in the context of interactive theorem proving. This makes it easier for the reader to compare the different approaches.

5.1.1 Introduction to KIV

KIV is an integrated software environment that supports the process of software development using formal methods [Balser et al., 2000]. It has been used, and found suitable, for the verification of large software systems, such as a PROLOG compiler [Schellhorn and Ahrendt, 1997] or an electronic purse [Schellhorn et al., 2006]. The specification language of KIV is based on higher-order algebraic specifications. Reactive systems can be described in KIV by means of either state-charts or parallel programs; here we use parallel programs. Parallel programs are modelled as follows. Let e denote an arbitrary (first-order) expression and v_d a dynamic variable (see below), then constructs for parallel programs include: $v_d := e$ (*assignments*), **if** ψ **then** φ_1 **else** φ_2 (*conditionals*), **while** ψ **do** φ (*loops*), **var** $v_d = e$ **in** φ (*local variables*), **patom** φ **end** (*atomic execution*), $\phi_1 \,||\, \varphi_2$ (*interleaved execution*), and $[p\#(e; v_d)]$ (*call to procedure p with value parameters e and reference parameters* v_d). The semantics of this extended language are defined in [Balser, 2005].

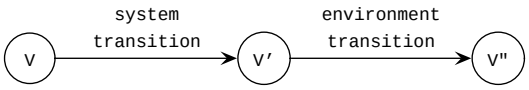


Figure 5.1: The relation between unprimed and primed variables as two distinct transitions: the system transition and the environment transition.

The correctness of systems is ensured by reasoning about the parallel program using future-time linear temporal logic [Balser et al., 2002b]. In KIV, linear temporal logic as described in Chapter 2 is extended with static variables v_s, which are variables that are mapped to the same element in the universe of discourse at each time point. Dynamic variables v_d, such as program variables, may have different interpretations at different time points. In the upcoming sections, the use of static variables will be explicitly mentioned. A speciality of KIV is the use of primed and double-primed variables: a primed variable v_d' represents the value of this variable after a system transition, the double-primed variable v_d'' is interpreted as the value after an environment

Specification	Data elements
capacity	exhausted, nearly-exhausted, subnormal
condition	hyperglycaemia, hypoglycaemia, normoglycaemia
updown	up, down
drug	SU, BG, glucosidase, insulin
setdrugs	set of elements of sort drug
setsetdrugs	set of elements of sort setdrugs

Table 5.1: Data specifications.

transition. System and environment transitions alternate, with v_d'' being equal to v_d in the successive state (cf. Figure 5.1 and Section 5.5.1).

5.1.2 Specification in KIV

In KIV, data types are expressed in a many-sorted algebra with possibilities for parameterisation, allowing the creation of specific sorts by defining constraints on the parameters. The sorts with associated data elements required to create a specification of the domain of diabetes mellitus type 2 are listed in Table 5.1.2.

In KIV, functions and predicates are static, i.e. they do not change over time. Therefore, for the formalisation in KIV functions and predicates were mapped to dynamic variables. For example, *secretion*(*B-cells*, *insulin*) was mapped to a dynamic variable named 'BsecretionI'. Since variables in axioms of algebraic specifications are universally quantified, a procedure with name 'patient' was used to bind these variables. This gives each relevant variable a context and prohibits instantiations of axioms with variables that have different names. In order to make formulas more readable, we will continue to write the variables as in their original notation.

KIV does not support past-time operators; however, as Gabbay has shown [Gabbay, 1989], it is possible to translate any temporal formula with past-time operators to an equivalent temporal formula with only future-time operators that includes 'until'. This implies that after translation it is possible, at least in principle, to verify the temporal formulas introduced in sections 4.2 and 4.3. All of the axioms with past-time operators are of the following fixed form:

$$\varphi \wedge \mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia}) \rightarrow \psi$$

We can rewrite this, through the semantical definitions, and obtain a pure future-time formula, i.e., a formula with only future-time operators:

$$\neg\,(\mathrm{Condition}(\mathit{hyperglycaemia})\ \mathbf{U}\ \neg\,(\phi \rightarrow \psi))$$

For more details, see [Hommersom et al., 2004a]. In the next sections, we continue to use the past-time modality, however. This makes the relation to Chapter 4 easier to comprehend.

5.1.3 Proofs

Again, consider a patient with hyperglycaemia due to nearly exhausted B-cells and $T = \{\text{Drug(SU)}, \text{Drug(BG)}\}$. Then the following sequent **(T2)** of Section 4.2, was proven by KIV in about 50 steps:

$$\mathcal{B}_{\text{DM2}} \cup \mathbf{G}\, T \cup \{\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{nearly\text{-}exhausted}\} \cup \{\mathbf{H}\, \text{Condition}(\mathit{hyperglycaemia})\} \models \mathbf{G}\, \text{Condition}(\mathit{normoglycaemia})$$

We can consider this an implication in the form:

$$\models \varphi \wedge \mathbf{H}\, \text{Condition}(\mathit{hyperglycaemia}) \rightarrow \psi$$

as mentioned in the previous section, where φ contains the background knowledge, treatment, and patient and ψ contains the intention. Hence, in future-time temporal logic, we need to prove a negated until formula. An outline of this proof follows. The proof obligation $\Gamma \vdash \Delta,\ \neg(\varphi\, \mathbf{until}\, \psi)$ is equivalent to $\Gamma,\ \varphi\, \mathbf{until}\, \psi \vdash \Delta$. The sequent is proved by induction over the number of steps it takes to satisfy ψ. For this, introduce a fresh dynamic variable N and generalise the sequent to $(N = N'' + 1 \wedge \phi)\, \mathbf{until}\, \psi,\ \Gamma \vdash \Delta$. The equation $N = N'' + 1$ ensures that N decreases in each step. Now, we can perform induction with induction term N which yields

$$(N = N'' + 1 \wedge \phi)\, \mathbf{until}\, \psi,\ \Gamma,\ N = n,\ \Box(N < n \rightarrow \mathsf{IndHyp}) \vdash \Delta$$

where $\mathsf{IndHyp} = ((N = N'' + 1 \wedge \phi)\, \mathbf{until}\, \psi) \wedge \bigwedge \Gamma \rightarrow \bigvee \Delta$ and n is a new static variable. We move to the next state by symbolically executing the temporal formulae. For example,

$$\phi\, \mathbf{until}\, \psi \Leftrightarrow \psi \vee (\phi \wedge \circ(\phi\, \mathbf{until}\, \psi))$$

is used to execute the **until** operator. In this case, the induction hypothesis can be applied in all possible successive states.

Other of such proofs have a similar structure and length.

5.1.4 Disproofs

The final part of this section we will show disproofs of properties that do not follow from the background knowledge by using program verification techniques. In the previous section we reasoned with the background knowledge; here we use a more extensive implementation of the '*patient*' procedure as shown in Figure 5.2, which implements part of the therapeutic reasoning.

Define the following theory which defined the implements of the background knowledge:

$$I = \{[\texttt{patient(...)}]\} \cup \bigcup_{x \neq \text{Drugs}} \{\mathbf{G}\ x' = x''\}$$

```
patient(var Drugs, Condition, UptakeLG, UptakePG,
            ReleaseLG, BcapacityI, BsecretionI, QI)
begin
  var oncebcapi = false, hchyper = true, nownormal = false
    in while true do
      patom
        if SU ∈ Drugs ∧ (BcapacityI ≠ exhausted ∨ oncebcapi)
        then begin
          BsecretionI := up; oncebcapi := true
        end;
        if BG ∈ Drugs then ReleaseLG := down;
        if  (ReleaseLG = down ∨ UptakePG = up)
          ∧ BsecretionI = up
          ∧( (Bcapacity = nearly-exhausted ∧ hchyper)
            ∨ nownormal)
        then begin
          nownormal := true; hchyper := false;
          Condition := normoglycaemia
        end
      end
end
```

Figure 5.2: Declaration of the patient procedure.

where the last term denotes that variables, except for 'Drugs', are not altered by the environment, but only by the program itself. In about 400 steps using KIV it was proved that $I \vdash \mathcal{B}_{\mathrm{DM2}}$, which implies $I \vDash \mathcal{B}_{\mathrm{DM2}}$ assuming KIV is sound. From this and the fact that I is consistent (since a program is consistent and the environment is not altered), we have shown that $\mathcal{B}_{\mathrm{DM2}} \nvDash \bot$ and therefore condition **(T1)** (cf. Section 4.2). The number of steps shows that this proof was significantly harder. The reason is that in many cases an invariant could only be defined after an initial symbolic execution. This caused an explosion of states that had to be considered. Furthermore, the invariants that had to be formulated were less straightforward.

Now showing that this set of drugs is a minimal treatment (condition **(T3)**), as discussed in Section 4.3, we construct a specific patient as follows:

$$\begin{aligned} I' = I \cup \{&\mathit{secretion}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{down},\\ &\mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down},\\ &\mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{down}\} \end{aligned}$$

Again, I' is consistent by its construction. It was proved in about 25 steps

with KIV that for all $T_s \subset \{\mathrm{SU}, \mathrm{BG}\}$:

$$I' \cup \mathbf{G}\, T_s \cup \{\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{nearly\text{-}exhausted}\} \cup \{\mathrm{Condition}(\mathit{hyperglycaemia})\} \models \neg\mathbf{G}\, \mathrm{Condition}(\mathit{normoglycaemia})$$

Because of monotonicity of temporal logic and $I \models \mathcal{B}_{\mathrm{DM2}}$, we have $I' \models \mathcal{B}_{\mathrm{DM2}}$. Since I' is consistent, we can conclude:

$$\mathcal{B}_{\mathrm{DM2}} \cup \mathbf{G}\, T_s \cup \{\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{nearly\text{-}exhausted}\} \cup \{\mathrm{Condition}(\mathit{hyperglycaemia})\} \not\models \mathbf{G}\, \mathrm{Condition}(\mathit{normoglycaemia})$$

Hence, $T = \{\mathrm{Drug}(\mathrm{SU}), \mathrm{Drug}(\mathrm{BG})\}$ is a minimal treatment. As one might expect, it shows that after the construction of the appropriate countermodel, disproofs are fairly easy.

5.2 Checking the Quality of Clinical Guidelines

In this section, we lay the framework for checking the quality of the clinical guidelines. First, we introduce the background knowledge that is used here. Then, quality criteria of individual treatments are extended to the complete guideline.

5.2.1 Formalisation of the Background Knowledge

The formalisation of the background knowledge of diabetes mellitus type 2 described here is based work described in Chapter 4, which formalised background knowledge for the purpose of verifying quality requirements of individual treatments (cf. Section 4.2). An example of a formula from Chapter 4 is the following:

$$\begin{aligned}
&(\ \mathbf{G}\ \mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up}\\
&\wedge\ \mathbf{G}\,(\ \mathit{uptake}(\mathit{peripheral\text{-}tissues}, \mathit{glucose}) = \mathit{up}\\
&\quad\ \wedge\ \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{exhausted})\\
&\wedge\ \mathbf{H}\,\mathrm{Condition}(\mathit{hyperglycaemia})\\
&)\ \rightarrow \mathbf{G}\ \ (\ \mathrm{Condition}(\mathit{normoglycaemia})\\
&\qquad\ \vee\ \mathrm{Condition}(\mathit{hypoglycaemia}))
\end{aligned}$$

Unfortunately, when combining this background knowledge formula with the logical theory of a fully formalised guideline, a theory results that is unsuitable for verification purpose as the time frame in which we have to verify quality requirements has changed. In large parts of Chapter 4, the time frame was restricted to the start and end of the application of one, *individual* treatment, whereas now we are interested in verifying quality requirements in the time frame of an *entire guideline* in which *multiple* treatments may be started and ended at any time. Hence, the assumption made in the formalisation of the background knowledge that a certain drug should *always* be applied to give

rise to its effects, no longer holds when verifying quality requirements in the time frame of an entire guideline.

In order to pass through subsequent possible treatments in Chapter 4, the background knowledge was parameterised with a particular step taken in the guideline. However, in this chapter, a different approach is taken. It should be noted that this adaptation does not render the previous method invalid; rather the formalisation proposed in this chapter provides a different interpretation to the temporal model, which makes it more appropriate to be used in this new context. We have employed the same logic as used for formalising the guidelines, i.e., future-time linear logic (cf. Section 2.1.2), with the time frame starting at the beginning of the guideline and continuing to either infinity or the end of the guideline. It is assumed that any additional knowledge needed about the patient history is available at the start of the guideline, allowing us to omit the converse modality **H** from [Lucas, 2003, Hommersom et al., 2004a], again similar to the abstraction performed in Chapter 4. An alternative to this approach is to translate formulas to equivalent temporal formulas with only future-time operators. However, we do not compare to Chapter 4 here, this provides little additional value in this case.

Furthermore, we interpreted the time period between causes (e.g., drug administration) and effects (e.g., drug effects) as the time period between the current state and the next state. In summary, the **G** operator used in [Lucas, 2003, Hommersom et al., 2004a] to model global qualitative behaviour is replaced by the **X!** operator to model cause-effect relationships. Hence, the above formula is replaced by the following formula:

$$
\begin{aligned}
&(\ \mathbf{X!}\ \mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up}\\
&\wedge \mathbf{X!}\,(\ \mathit{uptake}(\mathit{peripheral\text{-}tissues}, \mathit{glucose}) = \mathit{up}\\
&\qquad \wedge\ \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{exhausted})\\
&\wedge\ \mathrm{Condition}(\mathit{hyperglycaemia})\\
&) \rightarrow \mathbf{X!}\ (\ \mathrm{Condition}(\mathit{normoglycaemia})\\
&\qquad \vee\ \mathrm{Condition}(\mathit{hypoglycaemia}))
\end{aligned}
$$

The final specification is denoted by $\mathcal{B}_{\mathrm{DM2}}$ and is shown in Figure 5.3. One additional axiom for reasoning about diet is added (axiom (5)), which was not part of the original specification. As an illustration on how to read these formulas, consider Formula (6). This formula denotes that in case the B cells are being stimulated to secrete more insulin, i.e., the secretion is *up* after some unspecified time period, the capacity of these B cells is subnormal, and the QI index is less or equal than 27, then we expect that the condition will change from *hyperglycaemia* to *normoglycaemia*.

5.2.2 Quality Requirements of Clinical Guidelines

In Chapter 4, we have given a formalisation of the diabetes guideline using temporal logic. Here, we go one step further and investigate to which extend

(1) $\text{Drug}(\mathit{insulin}) \rightarrow \mathbf{X!}\,(\mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up} \wedge \mathit{uptake}(\mathit{peripheral\text{-}tissues}, \mathit{glucose}) = \mathit{up})$

(2) $\mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up} \rightarrow \mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down}$

(3) $(\text{Drug}(\text{SU}) \wedge \neg \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{exhausted}) \rightarrow \mathbf{X!}\ \mathit{secretion}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{up}$

(4) $\text{Drug}(\text{BG}) \rightarrow \mathbf{X!}\ \mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down}$

(5) $\mathit{diet} \wedge \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{normal} \rightarrow \mathbf{X!}\,\text{Condition}(\mathit{normoglycaemia})$

(6) $(\mathbf{X!}\ \mathit{secretion}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{up} \wedge \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{subnormal} \wedge \text{QI} \leq 27 \wedge \text{Condition}(\mathit{hyperglycaemia})) \rightarrow \mathbf{X!}\,\text{Condition}(\mathit{normoglycaemia})$

(7) $(\mathbf{X!}\ \mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down} \wedge \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{subnormal} \wedge \text{QI} > 27 \wedge \text{Condition}(\mathit{hyperglycaemia})) \rightarrow \mathbf{X!}\,\text{Condition}(\mathit{normoglycaemia})$

(8) $((\mathbf{X!}\ \mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down} \vee \mathbf{X!}\ \mathit{uptake}(\mathit{peripheral\text{-}tissues}, \mathit{glucose}) = \mathit{up}) \wedge \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{nearly\text{-}exhausted} \wedge \mathbf{X!}\ \mathit{secretion}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{up} \wedge \text{Condition}(\mathit{hyperglycaemia})) \rightarrow \mathbf{X!}\,\text{Condition}(\mathit{normoglycaemia})$

(9) $(\mathbf{X!}\ \mathit{uptake}(\mathit{liver}, \mathit{glucose}) = \mathit{up} \wedge \mathbf{X!}\ \mathit{uptake}(\mathit{peripheral\text{-}tissues}, \mathit{glucose}) = \mathit{up} \wedge \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{exhausted} \wedge \text{Condition}(\mathit{hyperglycaemia})) \rightarrow \mathbf{X!}\,(\text{Condition}(\mathit{normoglycaemia}) \vee \text{Condition}(\mathit{hypoglycaemia}))$

(10) $(\text{Condition}(\mathit{normoglycaemia}) \oplus \text{Condition}(\mathit{hypoglycaemia}) \oplus \text{Condition}(\mathit{hyperglycaemia})) \wedge \neg\,(\text{Condition}(\mathit{normoglycaemia}) \wedge \text{Condition}(\mathit{hypoglycaemia}) \wedge \text{Condition}(\mathit{hyperglycaemia}))$

Figure 5.3: Background knowledge $\mathcal{B}_{\text{DM2}}$ of diabetes mellitus type 2. Drug(x) holds iff drug x is being administered at that moment in time. The $\oplus$ operator denotes the exclusive OR operator.

it is possible to use a guideline representation language for this purpose. We also look in detail at possible guideline quality criteria.

Clinical guidelines consist, besides a description of treatments, of a control structure that uses patient information to decide on a particular treatment plan, i.e., the order (sequentially or in parallel) of treatments. Quality requirements for guidelines should extend the quality requirements of treatments as shown in the previous section to include requirements on the control structure.

Analogous to the previous section, good practice medicine of clinical guidelines can be formalised as follows. Let $\mathcal{B}$ be background knowledge, T be a treatment, P be a patient group, N be a collection of intentions, and M a clinical guideline in the form of a plan hierarchy. The structure M is composite and entails at certain points in time a treatment T that corresponds to a particular stage in the treatment process as described by the guideline. At all other time points M entails the empty treatment, i.e., $T = \varnothing$. A clinical guideline M is called a *proper* guideline according to the theory of abductive reasoning, i.e., $M \in \mathrm{Pr}_P$, if:

(M1) $\mathcal{B} \cup M \cup P \not\models \bot$ (the guideline does not have contradictory effects), and

(M2) $\mathcal{B} \cup M \cup P \models \mathbf{F}\, N$ (at some future state, the guideline satisfies all intentions)

In contrast to meta-axioms **(T1)** and **(T2)** (see Chapter 4), we focus here on treatment plans M rather than separate treatments T. Note that in contrast to **(T2)**, in **(M2)** we state $\mathbf{F}\, N$, i.e., eventually N, as the guideline may consist of multiple interventions and diagnoses that may not directly result in reaching each intention. Note that we here assume a broad notion of intention in the sense that intentions may hold for a longer period of time, e.g., after two weeks the intention may be satisfied that a certain drug is administered within the first week. This persistence of satisfied intentions explains why we require that at some point all the intentions should all be satisfied at once.

If, in addition to **(M1)** and **(M2)**, condition

(M3) $O_\varphi(M)$ holds, where O_φ is a meta-predicate standing for an optimality criterion or a combination of optimality criteria φ, which is defined as: $O_\varphi(M) \equiv \forall M' \in \mathrm{Pr}_P : \neg(M \prec_\varphi M')$,

then the guideline is said to be *in accordance with good-practice medicine*, again denoted as $\mathrm{Good}_\varphi(M, P)$.

A particular instance of condition **(M3)** and the preference relation can be constructed using a predicate. This predicate divides the possible guideline models into two groups, namely one group in which all models satisfy the predicate and one in which all models do not satisfy the predicate. The preference relation is then the preference over one of these two groups. For example, some predicates that should hold for a proper guideline are the completion of treatments as soon as the patient is cured, or consistency of the order of prescribed treatments with the preference order between treatments.

Formally, one can represent this by specifying, in addition to **(M1)** and **(M2)**, additional proof obligations. For example:

- The plan never prescribes treatments that are less preferred than the treatment that is in accordance with good practice medicine (see Section 4.2 for the definition of $\text{Good}_\varphi(T, P)$):

$$\mathcal{B} \cup M \cup P \models \mathbf{G}\ (\forall_{T'} : \text{Good}_\varphi(T', P) \wedge T \rightarrow \neg(T \prec_\varphi T'))$$

  where the preference relation $\prec_\varphi$ is defined on treatments, i.e., $T \prec_\varphi T'$ denotes that T is less preferred than T' given some criteria φ.

- If the patient is successfully treated, i.e., assuming no new conditions or comorbidities, the plan should not start new treatments, but unboundedly long treatments (like insulin) are allowed:

$$\mathcal{B} \cup M \cup P \models \mathbf{G}\ (N \wedge T \rightarrow \mathbf{X!}\,(T \vee \mathbf{last}))$$

- The order of prescribed treatments is consistent with the preference relation:

$$\mathcal{B} \cup M \cup P \models \mathbf{G}\ (T \wedge \mathbf{F}\ T' \rightarrow \neg(T \preceq_\varphi T'))$$

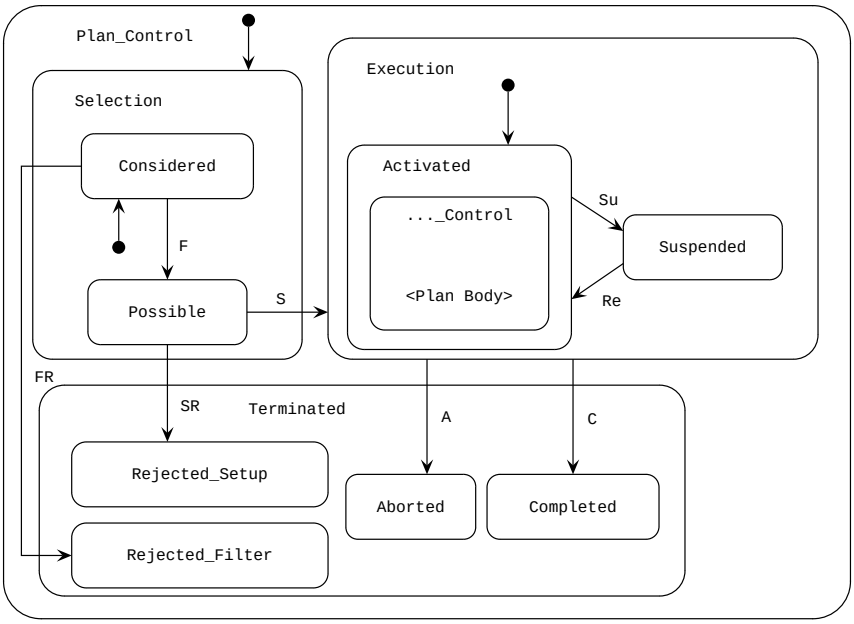
These quality requirements are examples that we consider realistic in the case study and in some sense corresponds to the idea that many quality criteria can be traced back to orderings between treatments (cf. Section 3.4). We believe that these examples have sufficient generality and can be used to illustrate the techniques that we employ. Nonetheless, further research will have to decide which quality requirements are most suitable for which guidelines as quality requirements should always be considered in the context of a specific guideline. In Section 5.5 we describe the actual verification process of these proof obligations for the guideline for the management of diabetes.

5.3 Clinical Guidelines in Asbru

Much research has already been devoted to the development of representation languages for clinical guidelines. Most of them look at guidelines consisting of a composition of actions, whose execution is controlled by conditions [Miksch, 1999]. However, most of them are not formal enough for the purpose of our research as they often incorporate free-text elements which do not have a clear semantics. Exceptions to this are PRO*forma* [Fox et al., 1996, Fox et al., 1997] and Asbru [Seyfang et al., 2002, Shahar et al., 1998]. The latter has been chosen in this chapter as a basis to implement the clinical guideline M mentioned in the previous section.

5.3.1 Introduction to Asbru

A clinical guideline is considered in Asbru as a hierarchical *plan*. The main components of an Asbru plan are intentions, conditions, plan-body, and time annotations. Furthermore, a plan can have arguments and can return a value.



S: [Satisfied(*setup_cond*)] SR: [¬Satisfiable(*setup_cond*)] A: [Satisfied(*abort_cond*)]
F: [Satisfied(*filter_cond*)] FR: [¬Satisfied(*filter_cond*)] C: [Satisfied(*complete_cond*)]
Su: [Satisfied(*suspend_cond*)] Re: [Satisfied(*reactivate_cond*)]

Figure 5.4: The plan state model, where Satisfied(*cond*) denotes that the environment satisfied the condition *cond* whereas Satisfiable(*cond*) denotes that, theoretically, the environment could still satisfy the condition *cond*, i.e., that no deadline has passed in case of time constraints.

The *intentions* are the high-level goals of a plan. Intentions can be expressed in terms of achieving, maintaining, or avoiding a certain state or action. The states or actions to which intentions refer can be intermediate or final (overall). In total there are twelve possible forms of intentions built up by combining elements from the sets {achieve, maintain, avoid}, {intermediate, overall}, and {state, action}.

Conditions can be associated to a plan to define different aspects of its execution. The most important types of condition are: (1) filter and setup conditions,[1] which must be true before a plan can start, (2) abort conditions,

[1] filter conditions are conditions about values that cannot change value, e.g., $sex = male$,

which define when a plan must abort, and (3) complete conditions, which define when a started plan finishes successfully. Conditions can be 'overridable' (i.e., healthcare personnel can manually satisfy the condition) or 'require confirmation' (i.e., conditions must be explicitly confirmed before they are satisfied).

The *plan-body* contains the actions, sub-plans, or both to be executed as part of the plan. The main types of plan-body are: (1) user-performed: an action has to be performed by a user, which requires interaction, which is not further modelled, (2) single-step: an action which can be either an activation of a sub-plan, an assignment of a variable, a request for an input value, or an if-then-else statement, (3) sub-plans: a set of plans to be performed in a given order, either sequentially, in parallel, in any-order, or unordered, and (4) cyclical plans: a repetition of actions over a time period. In case of sub-plans, it is also required to specify a waiting strategy to describe which of the sub-plans must be completed for the super plan to complete, e.g., all sub-plans should be executed (**wait-for all**).

Time annotations can be associated to various Asbru elements, e.g., intentions, conditions, plan activations. A time annotation specifies (1) in which interval things must start, (2) in which interval things must end, (3) their minimal and maximal duration, and (4) a reference time point.

5.3.2 The Semantics of Asbru

To help in the understanding of Asbru we review here the semantics of Asbru in a semi-formal statechart notation [Balser et al., 2006]. In Asbru, plans are organised in a hierarchy, where a plan may include a number of sub-plans. The semantics of Asbru is defined in [Balser et al., 2002a] by flattening the hierarchy of plans and using one top level control to execute all plans synchronously. Within each top level step, a step of every plan is executed. Whether a plan is able to progress depends on its conditions. The plan state model shown in Figure 5.4 defines the semantics of the main plan hierarchy. The 'Plan_Control' is divided into a selection phase, an execution phase, and a termination phase. Each plan goes into the 'Considered' state when it receives a *consider* signal. In this state its *filter condition* is checked. If it evaluates to true, control advances to the state 'Possible'. Then the setup condition is checked and if it is passed, control advances to the execution phase. If the filter condition is not satisfied or the setup condition is not satisfiable anymore (i.e., it is not possible to satisfy the condition in the future, because a deadline has passed), the plan is rejected. The same happens, if the super-plan terminates. In the execution phase the plan waits for an external signal *activate*, to be sent by its super-plan.

In state 'Activated', the sub-plans are executed, which can be sequentially, in parallel, unordered, or in any order, and each order determines a different controlling statechart [Balser et al., 2002a]. A plan can synchronise its sub-plans using the signals *consider* and *activate*. Additional control to propagate

whereas setup conditions are conditions about values that may change, e.g., glucose level.

execution states of a sub-plan to its parent and vice versa is also present, e.g., the abortion of a mandatory sub-plan enforces the parent-plan also to abort. Sub-plans can either be completed successfully or aborted, e.g., in the case of emergency patient readings.

The complete technical definitions, in addition to the semantics of the other constructs not shown here, can be found in [Balser et al., 2006].

5.3.3 Asbru Model of the Diabetes Mellitus Type 2 Guideline

The overall structure of the Asbru model of the guideline fragment (Figure 4.1) of the treatment of diabetes mellitus type 2 is shown in Figure 5.5. The Asbru model consists of a hierarchy of seven plans. The top level plan 'Treatments_and_Control' sequentially executes the four sub-plans 'Diet', 'SU_or_BG', 'SU_and_BG', and 'Insulin_Treatments', which correspond to the four steps of the guideline fragment in Figure 4.1. The fourth sub-plan 'Insulin_Treatments' is further refined by the two sub-plans 'Insulin_and_Antidiabetics' and 'Insulin', which can be executed in any order.

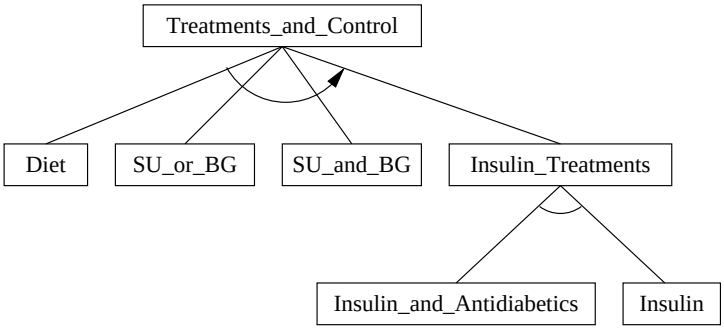


Figure 5.5: Asbru plan hierarchy of the diabetes mellitus type 2 guideline.

The Asbru specifications of two plans in the hierarchy, namely 'SU_or_BG' and 'Insulin_Treatments' are shown in Figure 5.6. Independent from the intentions, which are high level goals, one can describe the expected behaviour of plans using the **effects** attribute. In the case of 'SU_or_BG' there is a relationship between the Quetelet index (QI) and the drug administered. If the Quetelet index is less than or equal to 27, then SU is administered, else BG is administered. The plan 'SU_or_BG' corresponds to step 2 in the guideline fragment of Figure 4.1, which completes if the patient condition improves, i.e., the patient no longer has hyperglycaemia. In Figure 5.6 this is represented by the **complete condition**. The plan 'SU_or_BG' aborts when the condition of the patient does not improve. In Figure 5.6 this is represented by the **abort condition**, which requires a manual confirmation to ensure that some time passes for the drugs to have an impact on the patient condition.

The plan 'Insulin_Treatments' consists of two sub-plans, which correspond to the two options of step 4 in the guideline fragment of Figure 4.1, i.e., either insulin is administered or insulin and anti-diabetics are administered. Sub-plans are represented using the **body** attribute (Figure 5.6). In this case, the sub-plans are executed in any order and execution completes if one of the two sub-plans successfully completes (**wait-for one**).

plan SU_or_BG
  effects
    (QI $\leq$ 27 $\rightarrow$ SU $\in$ Drugs) $\wedge$
    (QI $>$ 27 $\rightarrow$ BG $\in$ Drugs)
  abort condition
    condition = hyperglycaemia **confirmation required**
  complete condition
    condition = hypoglycaemia $\vee$
    condition = normoglycaemia

plan Insulin_Treatments
  body anyorder wait for one
    Insulin_and_Antidiabetics
    Insulin

Figure 5.6: Asbru specifications of two treatments recommended in the diabetes mellitus type 2 guideline.

5.4 Specification in KIV

Previous sections have given the temporal logic formalisation of the background knowledge of diabetes mellitus type 2, the quality requirements, and the Asbru model of the medical guideline for diabetes mellitus type 2. In this section we discuss how these elements can be translated into KIV representations, so that they become amendable to verification.

5.4.1 Specification Methodology in KIV

The guideline and patient can be looked upon as a system (guideline) that interacts with the environment (patient). KIV allows a clear distinction between system and environment transitions by using primed and double-primed variables. Therefore, the Asbru model is only allowed to map variables into primed variables, whereas the environment is only allowed to map primed variables into double primed variables. System and environment transitions alternate (Figure 5.1).

However, system transitions in Asbru may involve a large number of steps (e.g., signals, plan state changes) before the model reaches a stable state from

which no further step can be made unless time progresses or the environment changes. Asbru is mainly a control oriented language and many control steps are not considered to take any real time at all. In an interactive theorem prover like KIV, this behaviour can be modelled by the introduction of two transition types, *micro-steps* and *macro-steps* [Schmitt et al., 2006a]. Micro-steps are technical Asbru steps where time and environment are not allowed to change. Macro-steps are temporal steps in which interaction can occur with the environment (e.g., plan activations) and are only executed when there are no micro-steps possible. The variable 'Tick', controlled by the symbolic execution of the Asbru semantics, holds when a macro-step occurs.

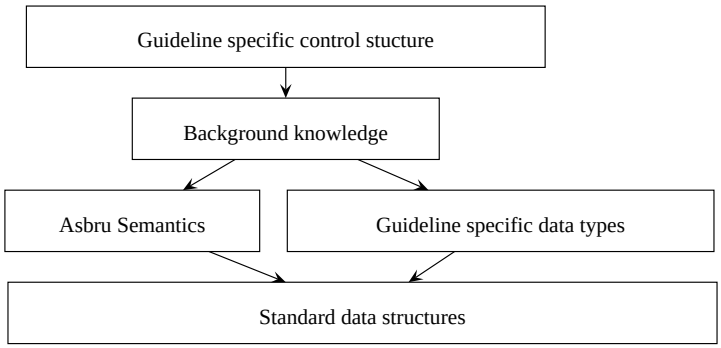


Figure 5.7: Dependency structure of Asbru specifications with $A \rightarrow B$ denoting that A depends on B.

In KIV, system descriptions are represented by means of a set of algebraic specifications. These algebraic specifications can be enriched with additional algebraic structures, which form a dependency structure between the different specifications. To maximise re-usability, several layers are used for representing our framework in KIV. The lowest layer in this dependency structure consists of standard data structures like Booleans and sets, which are typically obtained from libraries in KIV. On top of that, all data structures required to obtain a full definition of the semantics of Asbru were provided. The remaining layers consist of the structures dependent on the specific guideline under study. On top of the standard data structures, additional data structures are represented. For the diabetes case study, the data types are modelled as enumeration types. On top of the asbru semantics and data structures the background knowledge is represented. The top layer consists of the control structure of the guideline, which is the structure of Figure 5.5 in the diabetes case study (cf. Figure 5.7).

5.4.2 Specification of Background Knowledge in KIV

The background knowledge, i.e., axioms (1)–(9) in Figure 5.3, has been reformulated in terms of preconditions and postconditions, for reasons which will

become clear in Section 5.5.1. Every element that refers to the current point in time is interpreted as a precondition and each element that refers to the next point in time is interpreted as a postcondition. The values of these elements are stored in a data structure, denoted by 'Patient'. The patient is modelled by a sequence of pairs $[v, c]$, where v is the name of a variable and c a constant denoting the value of that variable, depending on the point in time. Updates to the patient record are done by appending a pair to the end of the sequence. Moreover, the most recent value of a variable v in a sequence s is given by the term $s[v]$. An example of the final translation can be found in Figure 5.8.

predicates
Knowledge : $patient \times patient$;
axioms
BDM2-1:
Knowledge(pre, $post$) $\rightarrow$ ($insulin \in pre[treatment] \rightarrow$
$post[uptake(liver,glucose)] = up \wedge$
$post[uptake(peripheral\text{-}tissues, glucose)] = up$)
BDM2-8:
Knowledge(pre, $post$) $\rightarrow$ ($post[uptake(liver,glucose)] = up$
$\wedge\ post[uptake(peripheral\text{-}tissues,glucose)] = up$)
$\wedge\ pre[capacity(b\text{-}cells,insulin)] = exhausted$
$\wedge\ pre[condition] = hyperglycaemia \rightarrow$
$post[condition] = normoglycaemia$)

Figure 5.8: Background knowledge in KIV as a first order predicate using pre- and postconditions, i.e., *pre* and *post* are shorthand notations for patient data structures with $pre[v] = c$ and $post[v] = c$ referring to the condition $v = c$ of the patient in the current and next state respectively. The use of *pre* and *post* variables is necessary to parameterise the background knowledge for arbitrary patient data structures. In addition, two translated rules from the background formalisation in Figure 5.3 are shown with BDM2-i representing rule (i).

5.4.3 Specification of Asbru in KIV

As each Asbru plan has a strict format, an algebraic function 'mk-asbru-def' has been defined for the translation of Asbru plans into KIV specifications. By calling 'mk-asbru-def' with the parameters that constitute a plan, translation of any guideline in Asbru becomes straightforward. The parameters consist of the various conditions that control plan state changes, the control type of sub-plans, a list of sub-plans, a retry value (for aborted plans), a wait-for condition (for mandatory sub-plans), and an optional wait-for flag (whether to wait for sub-plans). As there are quite a number of parameters, default values are provided to ease specification.

The Asbru semantics is implemented as a parallel program, parametrised with a given Asbru model. Temporal properties are proven using symbolic execution and induction [Balser, 2005].

5.4.4 Specification of Quality Requirements in KIV

With the help of KIV, we have verified that the diabetes guideline is proper, i.e., that the guideline satisfies conditions **(M1)** and **(M2)**, which is discussed in detail in Section 5.5.1 and 5.5.2. Meta-level quality requirements are verified in KIV using a sequent $\Gamma \vdash \Sigma$ where the conclusion Σ is some instantiation of **(M3)** and the antecedent Γ is a fixed structure that consists of the initial state of the patient and the Asbru model, the Asbru model, the effects of treatments, the background knowledge, and the environment assumptions. The sequent in Figure 5.9 is the specification in KIV of the quality requirement mentioned in Section 5.2.2, i.e., each patient is eventually cured from hyperglycaemia.

$$
\begin{array}{l}
\quad /\!* \textit{ Initial state of patient } *\!/ \\
\quad \mathrm{Patient}[\textit{condition}] = \textit{hyperglycaemia}, \\
\quad /\!* \textit{ Initial state of guideline } *\!/ \\
\quad \mathrm{AS}[\mathrm{Treatments\_and\_Control}] = \textit{inactive}, \ldots, \\
\quad /\!* \textit{ Asbru model } *\!/ \\
\quad [\textit{asbru}\#(\mathrm{Treatments\_and\_Control}; \mathrm{AS}, P)], \\
\quad /\!* \textit{ Effects } *\!/ \\
\quad \mathbf{G}\,(\mathrm{AS}[\mathrm{SU\_or\_BG}] = \textit{activated} \leftrightarrow \\
\qquad \mathrm{BG} \in \mathrm{Patient}'[\textit{treatment}] \wedge \ldots), \\
\quad /\!* \textit{ Background knowledge } *\!/ \\
\quad \mathbf{G}\,\mathrm{Knowledge}(\mathrm{Patient}', \mathrm{Patient}'') \\
\quad /\!* \textit{ Environment assumption } *\!/ \\
\quad \mathbf{G}\,(\mathrm{AS}''[\mathrm{Treatments\_and\_Control}] = \\
\qquad \mathrm{AS}'[\mathrm{Treatments\_and\_Control}] \wedge \ldots) \\
\vdash \\
\quad /\!* \textit{ Property } *\!/ \\
\quad \mathbf{F}\,(\mathrm{Patient}[\textit{condition}] = \textit{hypoglycaemia} \vee \\
\qquad \mathrm{Patient}[\textit{condition}] = \textit{normoglycaemia})
\end{array}
$$

Figure 5.9: Specification in KIV of the quality requirement that each patient is eventually cured from hyperglycaemia.

The *initial state* of the *patient* and the *Asbru model* are represented using additional data structures [Schmitt et al., 2005]. The patient data is represented in a data structure 'patient-data-history', which in Figure 5.9 is set to the patient group {Condition(*hyperglycaemia*)}. The initial state of the Asbru model is represented using a data structure 'AS' of type 'asbru-state', which keeps track of all plan states over time, and in which initially each plan is set to inactive. The *Asbru model* of the guideline describes the control structure,

and its specification in KIV has already been discussed in Section 5.4.3. The *effects of treatments* specify in KIV the behaviour of plans in the Asbru model. This is a direct translation of the **effects** attribute used in the Asbru model (cf. Section 5.3.3). In our diabetes case study the effects of plans are the administration of a certain drug as soon as the plan becomes activated, which may depend on the value of other variables like the Quetelet index (cf. Figure 5.6). The *background knowledge* is represented in the sequent using the first-order predicate 'Knowledge' and has already been discussed in Section 5.4.2. The environment is in principle allowed to change every variable arbitrarily. The *environment assumptions* restrict the behaviour of the environment. These restrictions (1) forbid the environment to change any variable, (2) force the environment to deterministically change a variable (e.g., advancing a clock), and (3) guarantee certain variable assignments in a nondeterministic way (e.g., the existence of a value when a signal is sent).

5.5 Verification using KIV

This section discusses in detail the verification of the quality requirements that we have defined in Section 5.2.2 for the guideline for the management of diabetes mellitus type 2 using KIV.

5.5.1 Consistency of the Formal Model

Property **(M1)** ensures that the formal model, including the Asbru guideline and the background knowledge, is consistent. Verifying property **(M1)** corresponds to verifying that the preconditions of the sequent in Figure 5.9 are not contradictory. KIV is a theorem prover, as such, it can only derive theorems and cannot be used to directly show that a given set of formulas is consistent. Nonetheless, the KIV system can be used to derive a theorem that creates a strong argument to show that our model is consistent, which we will illustrate here.

The initial state is – in our case – described as a set of equations and it has been trivial to see that they are consistent, as they do not contain any logical operators. The guideline is given as an Asbru plan. The semantics of any Asbru plan is defined in a programming language where every program construct ensures that the resulting reactive system is consistent: in every step, the program either terminates or calculates a consistent output for arbitrary input values. The Asbru plan thus defines a total function from unprimed to primed variables in every step (Figure 5.1). The formula defining the effects maps the output variables of the guideline to input variables of the patient model, which cannot violate the consistency of the resulting model.

The background knowledge defines our patient model. We consider the patient to be part of the environment which is the relation between the primed and the double primed variables in every step, i.e., respectively the states before and after an environment transition (cf. Figure 5.1). If the patient

model ensures that for an arbitrary primed state there exists a double primed state, the overall system of alternating guideline and environment transitions is consistent: given an initial (unprimed) state, the guideline calculates an output (primed) state; the effects define a link between the variables of the guideline and the variables of the patient model; the patient model reacts to the (primed) output state and yields a (double primed) state which acts again as input to the Asbru guideline in the next step. The additional environment assumption of Figure 5.9 does not destroy consistency, as the set of restricted variables of the environment assumption is disjunct to the set of variables of the patient model.

$$\begin{aligned} post = pre[&uptake(peripheral\text{-}tissues, glucose), up] \\ &[uptake(liver, glucose), up] \\ &[release(liver, glucose), down] \\ &[secretion(b\text{-}cells, insulin), up] \\ &[condition, normoglycaemia] \end{aligned}$$

Figure 5.10: Example patient adhering to background knowledge, with *pre* and *post* denoting patient data structures. Using the algebraic sequence notation for patient data structures described in Section 5.4.2, *pre* denotes an arbitrary patient data structure and *post* denotes the patient data structure equal to the patient data structure *pre* in which certain variables are updated.

It remains to ensure consistency of the background knowledge, which is defined in terms of a predicate 'Knowledge'. An additional property

$$\forall\ pre.\ \exists\ post.\ \text{Knowledge}(pre, post)$$

ensures that the relation is total. In order to verify that the properties of Figure 5.3 together with the property above is consistent, we have assumed a specific patient (see Figure 5.10), for whom all possible physiological effects described by the background knowledge occur. The patient reacts to an arbitrary input state *pre* with raised uptake of glucose in liver and peripheral-tissues, raised secretion of insulin from the B cells, and lowered release of glucose from the liver. Furthermore, the condition of the patient always improves to normal. Verifying that the example patient satisfies all of the properties of Figure 5.3 has been fully automatic.

5.5.2 Successful Treatment

In order to verify property **(M2)**, i.e., the guideline eventually manages all patient problems, a proof for the sequent of Figure 5.9 must be constructed. The verification strategy in KIV is symbolic execution with induction [Balser, 2005, Balser et al., 2002b]. The plan state model of Figure 5.4

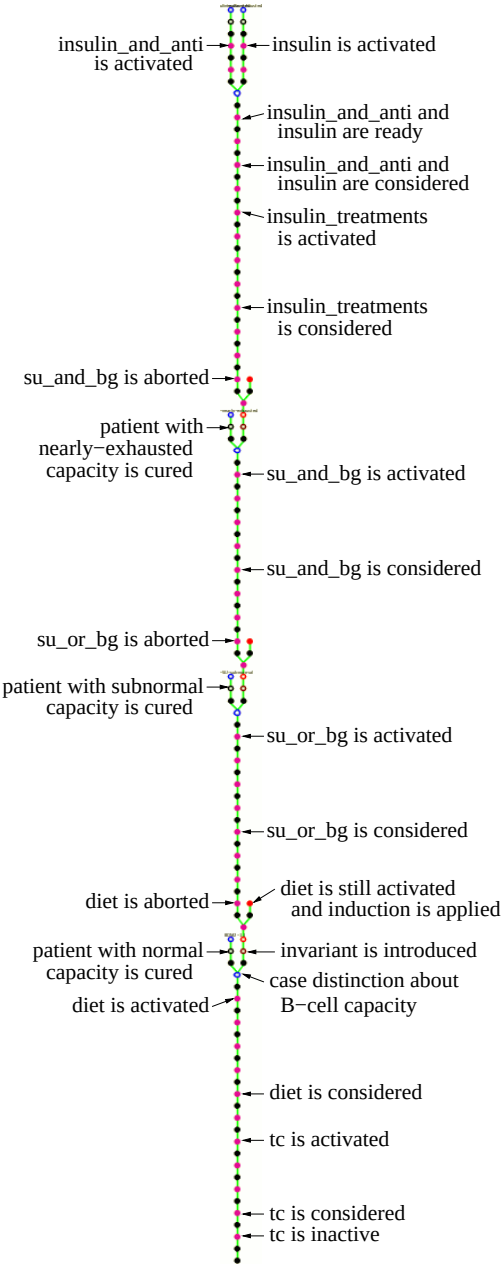


Figure 5.11: Annotated proof tree for property **(M2)**, i.e., the guideline eventually manages all patient problems. Proofs are started at the bottom with each subsequent bullet representing a proof step. Different branches represent case distinctions. The term 'tc' stands for 'Treatments_and_Control'.

is implemented by a procedure called 'asbru'. This procedure is symbolically executed. In the initial state, the top level plan 'Treatments_and_Control' is in 'inactive' state. After executing the first step, the plan is 'considered'. Execution continues as described by the plan state model of Figure 5.4 and produces the proof tree of Figure 5.11. Starting at the bottom, the proof consists of a number of step execution rules followed by simplification rules to simplify the first order formulas describing the current state. Each proof step is represented graphically with a bullet. The final proof tree contains all of the possible execution paths of the guideline.

After the 'Treatments_and_Control' plan is activated, the first sub-plan 'diet' is considered. Execution continues until the 'diet' plan is activated. The axioms of Figure 5.3 do not contain any knowledge about how diet effects a patient. We have therefore added the axiom

(10) $\mathit{diet} \wedge \text{Patient}[\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin})] = \mathit{normal}$
$\rightarrow \text{Patient}[\mathit{condition}] = \mathit{normoglycaemia}$

Patients whose capacity of the B cells is *normal* are cured with diet (left branch of case distinction). For other patients, diet may not be sufficient (right branch of case distinction). In this case, we assume that the doctor eventually aborts the diet treatment. We use induction to reason about the unspecified time period that a diet is followed by the patient. As an invariant,

$$\text{Patient}[\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin})] \neq \mathit{normal}$$

is used. In the next step, the doctor has either aborted 'diet' (left branch) or 'diet' is still active (right branch). In the second case, induction can be applied. If the first treatment is 'aborted', the second treatment 'SU_or_BG' is 'considered' and after some steps is 'activated'. In this case, either SU or BG is prescribed, depending on the Quetelet index QI. For a patient whose B cell capacity is subnormal, the background knowledge ensures that the condition of the patient improves (properties (3), (4), (5), and (6) of Figure 5.3). Thus, for the rest of the proof we can additionally assume that

$$\text{Patient}[\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin})] \neq \mathit{subnormal}$$

The third treatment ('SU_and_BG') is similarly executed and because of properties (3), (4), and (7) of the background knowledge patients with nearly exhausted B cell capacity are cured. Thus, the precondition concerning the capacity of the B cells can be strengthened to

$$\begin{aligned} &\text{Patient}[\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin})] \neq \mathit{normal} \\ \wedge\ &\text{Patient}[\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin})] \neq \mathit{subnormal} \\ \wedge\ &\text{Patient}[\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin})] \neq \mathit{nearly\text{-}exhausted} \end{aligned}$$

Here, we require an additional axiom which says that $\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin})$

is a function and therefore can only obtain one of the values from the set

$$\{\mathit{normal}, \mathit{subnormal}, \mathit{nearly\text{-}exhausted}, \mathit{exhausted}\}$$

to conclude

$$\mathrm{Patient}[\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin})] = \mathit{exhausted}$$

This axiom together with properties (1) and (8) of the background knowledge ensure that the prescription of insulin finally cures the patient.

5.5.3 Optimality of Treatment

With respect to property **(M3)**, an optimality criterion of the guideline is that no treatments are prescribed that are not in accordance with good practice medicine (Section 4.2), i.e., some preference relation $\preceq$ between treatments exists and the guideline never prescribes a treatment T, with $T \preceq T'$, *and* T' being sufficient for reaching the physicians' intentions for the patient group in question.

In our case study the preference for treatments is based on the minimisation of (1) the number of insulin injections, and (2) the number of drugs involved (cf. Section 4.2). We have defined this using a reflexive, transitive order $\leq$ such that for all treatments T, it holds that $\{\mathit{insulin}\} \leq T$ and $T \leq \{\mathit{diet}\}$. Furthermore, the treatments prescribing the oral anti-diabetics sulfonylurea and biguanide are incomparable. The proof obligation is then as follows:

$$\begin{array}{l}
\quad /\!* \textit{ Initial state of guideline } *\!/ \\
\quad \mathrm{AS}[\mathrm{Treatments_and_Control}] = \mathit{inactive}, \ldots, \\
\quad /\!* \textit{ Asbru plan } *\!/ \\
\quad [\mathit{asbru}\#(\mathrm{Treatments_and_Control}, \mathit{st}; \mathrm{AS}, \mathrm{Patient})], \\
\quad /\!* \textit{ Effects } *\!/ \\
\quad \mathbf{G}\,(\mathrm{AS}[\mathrm{SU_or_BG}] = \mathit{activated} \leftrightarrow \\
\qquad \mathrm{BG} \in \mathrm{Patient}'[\mathit{treatment}] \wedge \ldots), \\
\quad /\!* \textit{ Background knowledge } *\!/ \\
\quad \mathbf{G}\,\mathrm{Knowledge}(\mathrm{Patient}', \mathrm{Patient}'') \\
\quad /\!* \textit{ Environment assumption } *\!/ \\
\quad \mathbf{G}\,(\mathrm{AS}''[\mathrm{Treatments_and_Control}] = \\
\qquad \mathrm{AS}'[\mathrm{Treatments_and_Control}] \wedge \ldots) \\
\vdash \\
\quad /\!* \textit{ Property } *\!/ \\
\quad \mathbf{G}\,(\forall_T \colon \mathrm{Good}_{\leq}(T, \mathrm{Patient}) \rightarrow \neg\,(\mathrm{Patient}[\mathit{treatment}] < T))
\end{array}$$

Furthermore, we needed to add the following axiom to our system:

$$\mathbf{G}\ \mathrm{Patient}[\mathrm{QI}] = \mathrm{Patient}''[\mathrm{QI}]$$

i.e., the Quetelet index QI does not change during the execution of the guideline. This axiom is needed, because the decision to prescribe a treatment is not exactly made at the same time as the actual application of the treatment, and, therefore, the decision to prescribe this treatment could be based on a patient with a Quetelet index different from the patient that takes the drugs.

Proving this property in KIV was done in approximately one day using particular heuristics for the straightforward parts, e.g., propositional simplification and symbolic execution. The theorem was proven using two lemmas for two specific patient groups. In total, it took approximately 500 steps to verify this property with a degree of automation of approximately 90%. The verification process yields insight in the inferences needed to construct the proof, which provides the opportunity to construct case-specific heuristics. This improves the level of automation, however, verification of other properties will not necessarily benefit from the additional heuristics.

5.5.4 No New Treatments

The previous property does not rule out that the guideline for diabetes mellitus type 2 prescribes additional treatments in case the patient is cured. This can be formalised as follows:

$$\begin{aligned}&\mathbf{G}\,\forall_{T_s}(\mathrm{Tick} \wedge T_s = \mathrm{Patient}[\mathit{treatment}] \wedge \\ &\quad \mathbf{X!}\,\mathrm{Patient}[\mathit{condition}] \neq \mathit{hyperglycaemia} \rightarrow \\ &\quad \mathbf{X}\,\mathbf{G}\,((\neg\,\mathbf{last} \wedge \mathrm{Tick}) \rightarrow \mathrm{Patient}[\mathit{treatment}] = T_s)))\end{aligned}$$

To compare the current treatment with any future treatment a static variable T_s is used to store the current treatment administered to patient 'Patient' as 'Patient' may dynamically change. The variable T_s only needs to be compared with future treatments in case the patient is cured. Because curing the patient requires one time step in the formalisation of the background knowledge, to check if treatment T_s has cured the patient we need to check whether the patient's condition is different from hyperglycaemia in the *next* state using the **X!** modal operator (cf. Figure 2.2). When both conditions hold then either the execution of the guideline should complete or, when it does not, only treatment T_s should be administered. The 'Tick' variable is introduced to restrict the property to macro-steps (cf. Section 5.4.1). As we are only interested in the temporal behaviour of plan activations, the property would trivially be violated when one would allow micro-steps as micro-steps do not allow temporal behaviour, i.e., plans are never activated in micro-steps.

The effort to prove this property was of the same order as proving that the treatment is successful. The difficult part was to find the right formalisation, taking into account that the execution of the guideline completes and that internal (micro) steps can violate the proof obligation.

5.5.5 Order of Treatments

Finally, it was proven that the order of any two treatments in the guideline is consistent with the order relation as we have defined in Section 5.5.3. The formalisation of the property in KIV was done as follows:

$$\begin{aligned}&\mathbf{G}\,\forall_{T_s}(\mathrm{Tick} \wedge T_s = \mathrm{Patient}[\mathit{treatment}]\\&\quad \rightarrow \mathbf{G}\,(\mathbf{last} \vee (\mathrm{Tick} \rightarrow \neg(T_s \leq \mathrm{Patient}[\mathit{treatment}]))))\end{aligned}$$

At each time the current treatment is bound to a static variable T_s, which can be used to compare against subsequent steps in the guideline. For any future steps, we require that either the guideline completes (**last** holds) or that activated treatments are not more preferred than T_s, i.e., the property states that less preferred treatments should not be applied first. The formalisation represents the property introduced in Section 5.2.2, by using convenient KIV features. Again, the variable 'Tick' is needed in the formalisation to abstract from technical system steps (cf. Section 5.5.4).

This property also had a high degree of automation with roughly 800 steps in total. The reason for this slightly higher number of steps is due to nested temporal operators. However, the steps were straightforward, as the different branches have a similar structure. The only knowledge used was the axiom that states that the Quetelet index QI is constant during the run of the guideline. Without this assumption it is possible that the treatment switches from 'SU' to 'BG' or from 'BG' to 'SU' during the activation of the plan 'SU_or_BG'. This unwanted behaviour would lead to a counter-example as these two drugs are assumed to be incomparable. This shows that additional assumptions about patients may be necessary even for properties that only state something about the structure of the guideline.

5.6 Conclusions

In this chapter we have set up a general framework for the verification of clinical guidelines, consisting of a clinical guideline, medical background knowledge, and quality requirements. A model for the background knowledge of glucose level control in diabetes mellitus type 2 patients was developed based on a general temporal logic formalisation of (patho)physiological mechanisms and treatment information. Furthermore, we developed a theory for quality requirements of good practice medicine based on the theory of abductive diagnosis. This theory of quality requirements and the model of background knowledge were then used in a case study in which we verified several quality criteria of the diabetes mellitus type 2 guideline used by the Dutch general practitioners. In the case study we used Asbru to model the guideline as a network of tasks and KIV for the formal verification.

In the course of our study we have shown that setting up a general framework for the formal verification of clinical guidelines with medical background

knowledge is feasible and that actual verification of quality criteria can be done with a high degree of automation. We believe that both the inclusion of medical background knowledge and semi-automatic decision support are essential elements for adequately supporting the development and management of clinical guidelines. Our approach allows one to reason about the guideline in terms of the effects treatments have on patients and, consequently, it is possible to specify general requirements in terms of the *outcome* of a guideline, rather than in terms of the fairly arbitrary document structure of the guideline.

Although, the presented case study is small compared to some other guidelines, the framework has been setup to be scalable for larger verification studies. First, a large number of algebraic specifications in KIV dealing with the Asbru semantics and data types are re-usable. Second, translation of Asbru models into KIV can be done automatic. Third, KIV has an integrated proof maintenance system that keeps track of invalidated proofs in case of changes to specifications and even tries to correct invalidated proofs automatically. However, KIV is an expressive tool, which may result in an additional overhead when verifying quality criteria of clinical guidelines. Additional research could focus on other techniques like model checking, which may be less expressive, but require less overhead for verifying quality criteria. Such techniques could be part of a process of quality checking guidelines in which our approach would be at the far end of the spectrum of possible techniques while simpler techniques can be used as early as the modelling of the guideline itself to remove errors and ambiguities in the guideline. This would improve scalability even further.

Chapter 6

Applying Model Checking to Formal Models of Guidelines

Clinical guidelines are often underspecified, which is sometimes seen as an anomaly in the guideline [Shiffman, 1997, Miller et al., 1999]. As discussed in Chapter 3, the development of guidelines is based more and more on answering particular key questions while ignoring others, which makes such incompleteness to be expected. This poses a problem for applying model checking technology: if a system is largely underspecified, then there are a large number of possible behaviours that have to be taken into consideration. This was one of the reasons for the use of theorem proving facilities in the previous chapters. Nonetheless, we may expect that at least parts of the guideline contain detailed information, making model checking of those parts a feasible option. In addition, one may also expect that clinical protocols contain less underspecification as they are seen as local adaptation of clinical guidelines. The hypothesis is that protocols therefore also contain more detailed information. Furthermore, in using a guideline to support the actual medical management, a detailed model of the care process is needed. In such a situation, we also expect that the model includes additional knowledge.

One way to look upon a patient and a patient's disease logically is as a state machine, i.e., as a system described in terms of states and state transitions in time. Model checking technology offers methods that allow one to analyse concurrent systems for their consistency. For this purpose, one can rely on an extensive collection of tools and techniques readily available. Model checking is a well-investigated technique for the verification of systems modelled by a finite transition system. However, model checking has been applied mainly to technical systems, such as hardware, software-based communication protocols, concurrent programs, etc. This global view on the representation of disease process, patient conditions, and disease management actions raises the question whether model checking can be used as a basis for checking properties of

guidelines. It is this question that is being explored in detail in this chapter.

This chapter consists of two parts. In the first part, we concern ourselves with protocols and how consistency can be checked on the basis of guidelines. Model checking is employed to actually perform these checks. In the second part of this chapter, we look at so-called critiquing which aims at spotting and analysing differences between the proposed actions taken by a medical doctor, and a set of 'ideal' actions as prescribed by the computerised guideline. As such systems are supposed to be deployed in actual practice, we assume that the underlying model is relatively complete. Again, this raises the question how we can use model checking techniques to find these differences.

6.1 Protocol Refinement

In the past, protocol conformance to guidelines has only been looked at from an informal angle [Marcos et al., 2006]. In this section we address the problem of protocol conformance to guidelines using formal methods. This is done by interpreting guidelines as defining (logical) constraints on the medical management of patients performed in practice, whereas protocols are interpreted as more or less executable models. The constraint-based approach of looking at guidelines was inspired by a statement by Wiersma and Burgers that "recommendations in guidelines should not only be based on evidence extracted from scientific literature, but take into account the context of daily medical practice as well" [van Everdingen et al., 2004]. In principle this approach would allow one to discover flaws or suboptimal management actions in the medical management in practice, assuming that a given protocol and guideline are correct, or to find incorrect or suboptimal medical management decisions in a protocol or guideline, assuming that the medical management in practice is correct and optimal. In this chapter, we assume that the guideline augmented with information from general medical practice is correct and use model checking techniques to find flaws in the protocol.

6.1.1 Approach

The premise in this work is that clinical practice guidelines and protocols provide *necessary*, but not *sufficient* conditions for making medical decisions. In particular, the guideline requires interpretation of medical doctors to apply the advice in practice. Thus, treatments which are acceptable given a strict interpretation of the protocol, might not be acceptable in practice. For example, the Dutch breast cancer guideline (cf. Section 3.2.2) does not exclude the possibility of informing the patient *after* treatment about the different treatment possibilities. Clearly, such incompleteness of the guideline or protocol will not cause problems in practice as the advice is not mechanically followed by the physician. Hence, differences related to such a (lack of) advice can be considered irrelevant in practice. This is abstractly represented in Figure 6.1, which shows several paths of a protocol compared with all paths allowed by

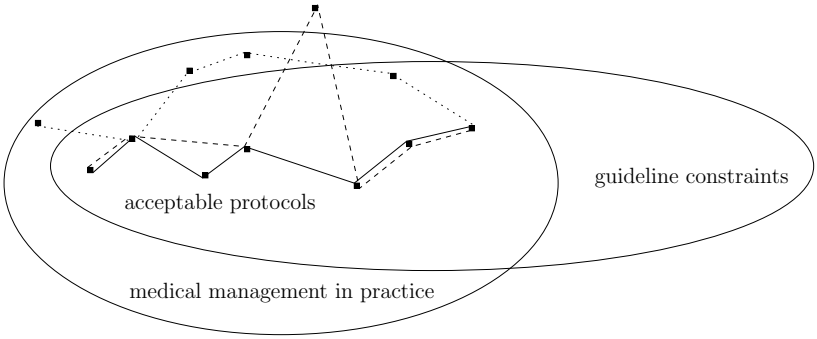


Figure 6.1: Sketch of medical management paths occurring in a protocol.

a guideline and the medical management in practice. In this case, violations occurring in the dotted line are most interesting, as the violation in the dashed line will not occur in practice. For example, physicians have been trained to inform a patient properly and to avoid doing unnecessary tests when a diagnosis has been established. As guidelines are not meant as textbooks for teaching medical students, such common sense medical reasoning is typically ignored in a guideline.

Thus, the question is whether or not a protocol, restricted to medical management in practice, conforms to the constraints that the guideline imposes. The problem of this approach is that this medical management in practice cannot easily be assessed, because it is based on general and abstract medical knowledge, which is difficult to articulate and express directly, which makes it difficult to elicitate and formalise. Therefore, the approach we have taken in this chapter is that we have formulated a concrete description of treatment paths that are taken in practice, extracted from a medical textbook. This does not formalise the *general* medical practice, but provides a good estimation of what is usually done in practice for a specific domain. Then, the approach consists of the following steps. Firstly, concrete treatment paths are formulated, which are known to be part of medical management in practice. Secondly, this information is weakened up to a point it is consistent with the protocol, i.e., we investigate which paths of medical practice are consistent with the protocol. Finally, the protocol restricted by medical practice is compared with the constraints imposed by the guideline.

In other words, the guideline restricted to medical practice, i.e., the intersection of the two ellipses in Figure 6.1, is defined as the gold standard. Finding a deviation from this gold standard in a protocol might indicate a problem in the protocol, although it is possible that it only indicates an error in the medical textbook or guideline. Nonetheless, we believe that such information is more valuable to guideline and protocol developers than deviations which are not even part of common sense medical practice.

6.1.2 Medical Management of Breast Cancer

First, we give an informal description on the medical management as stated in the CBO guideline, the IKO protocol, and the specialised textbook of Roses [Roses, 2005] that deals with locoregional treatment of operable breast cancer, i.e., T1-2 N0-1 M0 breast cancer according to the TNM classification system [Green et al., 2002]. Thereafter, we give formalisations according to the approach described in the previous section.

6.1.3 Informal Description of Medical Management

According to the CBO guideline there are only two options for local treatment of operable invasive breast cancer: breast-conserving therapy (BCT) or modified radical mastectomy (MRM). BCT implies ample local excision of the tumour, an axillary staging procedure, and radiotherapy of the breast. MRM involves a total resection of the breast (mastectomy) and dissection of the axillary nodes (AND). The aim of BCT is to achieve a survival rate comparable to that following MRM with an optimal cosmetic result in terms of the treated breast. BCT is usually the preferred treatment unless the patient has a clear preference for MRM or there are contra indications for BCT, i.e., there is either (1) multicentricity (two or more tumour foci in different quadrants), (2) diffuse malignant microcalcifications, or (3) previous radiotherapy of the breast. Whereas these three contra indications are obtained *before* surgery, one other contra indication for BCT is obtained *during* surgery, i.e., (4) the margins of the local excision remain tumour-positive after repeated local excision attempts. In this case, local excision attempts are unsuccessful in removing the primary tumour and treatment therefore switches to MRM.

Treatment of the axillary nodes is also part of the treatment of breast cancer as the pathologic assessment of axillary lymph nodes remains the most important prognostic variable for the invasive breast cancer patient. An optimal assessment would be achievable by means of a complete axillary node dissection. However, AND may lead to morbidity, e.g., pain, limited shoulder movement, and lymphoedema An alternative for axillary staging is the sentinel node procedure (SNP), which only dissects the sentinel nodes, i.e., those nodes that drain the area of the breast where the primary tumour is located and thus are most likely to contain metastasis. The SNP is currently the standard procedure for axillary staging in breast cancer provided that the contra indications do not hold, where contra indications of SNP are defined as (1) suspected or proven malignancy in the axillary nodes, (2) tumour $>$ T2, (3) multiple tumour foci, or (4) potentially disrupted lymph drainage due to recent axillary surgery or a large biopsy cavity following tumour excision. When the SNP is not possible, complete axillary node dissection should be carried out. Furthermore, treatment of the axilla is indicated (i.e., dissection, radiotherapy) for all forms of lymph node metastasis.[1]

[1] The CBO guideline differs at this point with the IKO protocol as it makes an exception

Whereas the CBO guideline and IKO protocol lack many details about treatment order, [Roses, 2005] provides a much more detailed description. In addition, according to [Roses, 2005], the sentinel node procedure (SNP) is started before segmental excision (i.e., used in BCT) or mastectomy. The sentinel nodes (SNs) are then immediately sent to the pathology lab, where they are examined during surgery. If the SNs are found to be positive, axillary dissection can be completed during the primary breast surgery in one setting. Furthermore, [Roses, 2005] differs with the CBO guideline and IKO protocol in the case of recurrent tumour positive resection margins in the BCT treatment. Whereas CBO and IKO recommend to switch the treatment to MRM, which includes axillary dissection, [Roses, 2005] only recommends a mastectomy with axillary dissection dependent on sentinel node histopathology.

6.1.4 Formalisation of Medical Management

Here, we introduce the constraint-based representation of the guideline, an executable model of the protocol, and the model of the medical management performed in practice.

Constraint-Based Representation of the CBO Guideline

The language we use for atomic propositions consists of medical actions *Actions*, medical plans *Plans* (composite actions), and data structures *Data* (patient characteristics):

$$\begin{aligned} \mathit{Actions}: &\ \{\text{tumour-excision}, \text{mastectomy}, \text{AND}, \text{SNP}\} \\ \mathit{Plans}: &\ \{\text{TREATMENT}, \text{BCT}, \text{MRM}, \text{AXILLA-STAGING}\} \\ \mathit{Data}: &\ \{\text{CI-BCT}, \text{CI-SNP}, \text{TF}, \text{SN}, \text{ITC}\} \end{aligned}$$

with CI-BCT, CI-SNP $\in \{\top, \bot\}$ (true, false) denoting the contra indications for BCT and SNP respectively, SN $\in \{\mathit{unknown}, \mathit{neg}, \mathit{pos}\}$ denotes whether there is a metastasis found in the lymph nodes after performing the SN procedure, TF $\in \{\mathit{unknown}, \top, \bot\}$ denotes whether the re-section margins are tumour free, and ITC $\in \{\mathit{unknown}, \top, \bot\}$ denotes whether the tumour cells are isolated when the sentinel node is found positive. In formulas, we write the variable name if it is meant that the variable is equal to $\top$, and the negated variable name is used to denote the respective variable is equal to $\bot$. As the guideline does not refer to a variable as if it were *unknown*, apparently assuming that appropriate diagnostic tests have been performed, this value is not used in the formulas. An action or a plan is true the moment it is started, e.g., activated according to the Asbru semantics (cf. Section 5.3). The final representation in temporal logic of the medical management in the CBO guideline is shown in Figure 6.2.

---

for isolated tumour cells.

Constraints related to control structure
(1) **AG**(TREATMENT → **AF**(BCT ∨ MRM))
(2) **AG**(CI-BCT → ¬BCT)
(3) **AG**(BCT → **AF**(AXILLA-STAGING ∨ MRM) ∧ **AF** tumour-excision)
(4) **AG**(MRM → **AF** AND ∧ **AF** mastectomy)
(5) **AG**(AXILLA-STAGING → **AF** (AND ∨ SNP))
(6) **AG**(CI-SNP → ¬SNP)
(7) **AG**(tumour-excision → ((¬TF → **AF** MRM) ∧ (TF → **AG** ¬MRM))
(8) **AG**(SNP → (SN = *pos* ∧ ¬ITC → **AF** AND))
(9) **AG**((**AG**¬MRM) → **AG**(SNP → **AG**(ITC → **AG**¬AND)))

Figure 6.2: Constraint-based representation of the CBO guideline. BCT = breast conserving treatment, MRM = modified radical mastectomy, CI-BCT = contra indications for BCT, SN = result of sentinel node procedure, CI-SNP = contra indications for SNP, TF = tumour free resection margins.

Some constraints given by the guideline are not easily expressible in temporal logic as other modalities than treatment order are involved. For example, the preference for BCT over MRM and the preference for the SNP over axillary dissection for staging the axilla. Furthermore, certain assumptions regarding the patient data are implicit in the guideline. For example, the status of the resection margins (tumour free (TF) or not (¬TF)) becomes known after tumour excision and the existence of metastasis (SN=pos or SN=neg) becomes known after the SNP. Here we have chosen not to consider these more implicit constraints, as they are not directly related to the recommendations given in the guideline.

Asbru Representation of the IKO Protocol

To transform the IKO protocol into a more or less executable model, which can be verified with respect to the constraints set by the CBO guideline, we have chosen to use the guideline representation language Asbru [Shahar et al., 1998] as intermediate representation. We use Asbru, because its semantics has been defined precisely in previous research [Balser et al., 2002a] and can be translated automatically into SMV [McMillan, 1993] for model checking purposes [Bäumler et al., 2006]. An example fragment of this SMV model can be consulted in Appendix B.3.

The Asbru model constructed (Figure 6.3) consists of nine plans ordered in a hierarchy. Arrows indicate sequentially executed sub-plans, dashed lines unordered executed sub-plans. The latter is in particular used when the protocol lacks any information about treatment order, e.g., `MRM` with sub-plans `AND` and `mastectomy`. The top level plan `treatment` first executes `BCT` unless there are contra indications (filter condition). If `BCT` successfully completes, so will `treatment` (wait-for condition), else `MRM` will be executed. The execution of

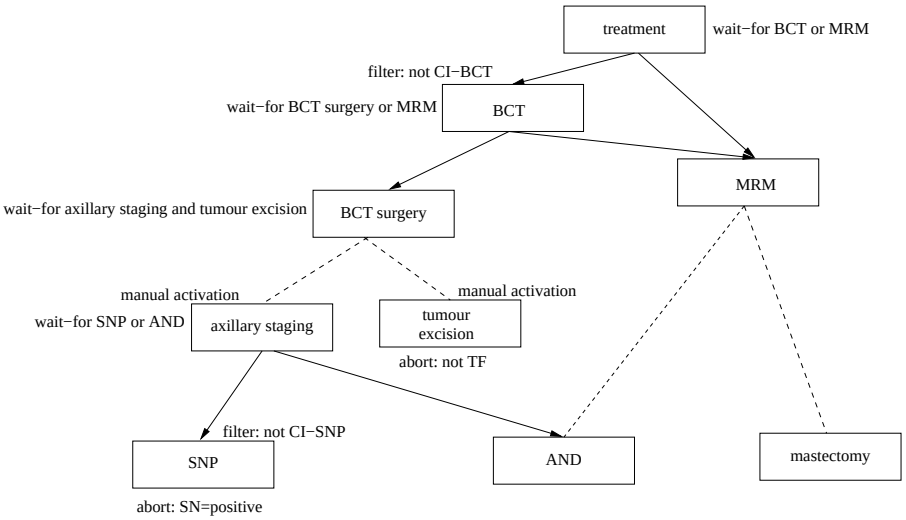


Figure 6.3: Asbru interpretation of the IKO protocol. Arrows represent sequential plans, dashed lines represent unordered sub-plans.

`BCT surgery` and `MRM` by `BCT` is analogous. `BCT surgery` may execute its sub-plans in any order. To allow for a particular order we use a manual activation which we assume to occur eventually. The `SNP` and `tumour excision` may abort (abort condition) in case of positive SNs or not tumour free resection margins respectively, which is then propagated up the hierarchy, resulting in `BCT` executing `MRM`. (cf. Section 5.3 and [Balser et al., 2002a] for details about the Asbru semantics.)

In the SMV model, which is automatically constructed from the Asbru model, most variables dealing with patient data are initialised as *unknown* and receive an non-deterministic value in the second step to make sure there is only one root of the model. Furthermore, we assume that they do not change during the treatment. The only variables that are initialised at a later stage are the status of the sentinel node, which becomes known during the SNP and whether or not the re-section margins are tumour free, which becomes known after excising the tumour. Furthermore, fairness constraints have been added to ensure that the manual activation of both the axillary staging and the tumour excision eventually occurs. In other words, the patient will not wait indefinitely for the treatments to start.

Decision Tree of Medical Management in Practice

Information from [Roses, 2005] can be represented in a decision tree as shown in Figure 6.4, which deals with the ordering of medical actions treating the

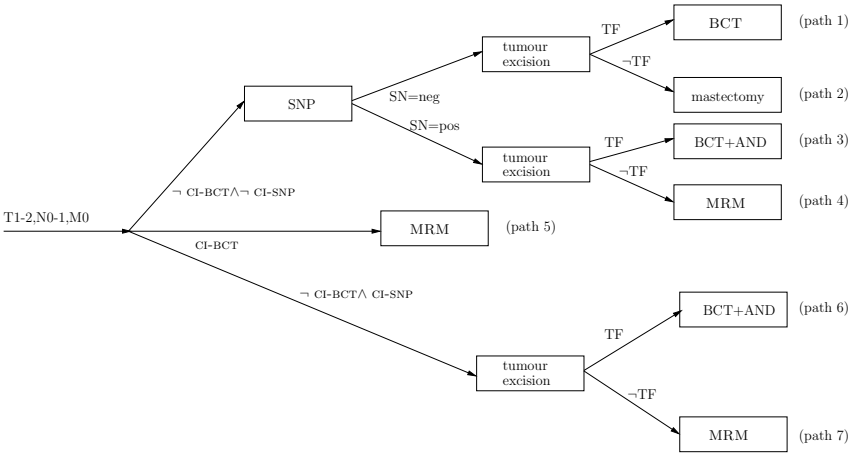


Figure 6.4: Background knowledge: possible treatment paths for surgery of operable invasive breast cancer. CI-BCT = contra indications BCT, CI-SNP = contra indications SNP, TF = tumour free resection margins, AND = axillary node dissection.

primary tumour (BCT and MRM) and the axilla (SNP and/or AND).[2] Nodes represent medical actions or plans, arcs represent constraints. A path from the root node to a leaf node represents a treatment path, which defines the order of medical actions when the constraints on the path are satisfied. Leaf nodes labelled with BCT or MRM denotes the summary of the medical strategy followed in that treatment path.

The formal interpretation of this decision tree is two-fold. First, we can give a CTL characterisation of these paths in terms of an existential path operator, which allows us to verify whether each of these paths exists in the protocol. This set of formulas is denoted by Δ. Each action that occurs in the decision tree has been interpreted as a plan activation in the execution of the Asbru model, although the technical details of this have been omitted here for clarity. The first two paths are described as follows:

$$
\begin{array}{ll}
(1) & \mathbf{EX}(\neg\text{CI-BCT} \wedge \neg\text{CI-SNP} \wedge \\
& \quad \mathbf{EF}(\text{SNP} \wedge \text{SN} = neg \wedge \\
& \quad\quad \mathbf{EF}(\text{tumour-excision} \wedge \text{TF} \wedge \\
& \quad\quad\quad \mathbf{AG}(\neg\text{mastectomy} \wedge \neg\text{AND})))) \\
(2) & \mathbf{EX}(\neg\text{CI-BCT} \wedge \neg\text{CI-SNP} \wedge \\
& \quad \mathbf{EF}(\text{SNP} \wedge \text{SN} = neg \wedge \\
& \quad\quad \mathbf{EF}(\text{tumour-excision} \wedge \neg\text{TF} \wedge \\
& \quad\quad\quad \mathbf{EF}(\text{mastectomy}) \wedge \mathbf{AG}(\neg\text{AND}))))
\end{array}
$$

[2] In this section, we abstract from radiotherapy and isolated tumour cells.

On the other hand, we may view the decision tree as a number of constraints on medical management in practice. To this end, we extract a number of assumptions from the decision tree about the normal implementation of the advice given in the protocol. It is quite obvious that it can be guaranteed that such statements are sound with respect to the decision tree. In this case study, we consider the following LTL assumptions referred to by Γ.

(1) $(\neg\text{CI-BCT} \wedge \neg\text{CI-SNP}) \leftrightarrow \mathbf{F}\ \text{SNP}$
(2) $(\mathbf{F}\ \text{SNP}) \rightarrow ((\neg\text{tumour-excision}\ \mathbf{U}\ \text{SNP}) \wedge \mathbf{F}\ \text{tumour-excision})$
(3) $((\mathbf{F}\ \text{SN} = neg) \wedge (\mathbf{F}\ \text{TF})) \rightarrow (\neg(\mathbf{F}\ \text{AND}) \wedge \neg(\mathbf{F}\ \text{MRM}))$
(4) $((\mathbf{F}\ \text{SN} = neg) \wedge (\mathbf{F}\ \neg\text{TF})) \rightarrow ((\mathbf{F}\ \text{mastectomy}) \wedge \neg\mathbf{F}\ \text{AND})$
(5) $((\mathbf{F}\ \text{SN} = pos) \wedge (\mathbf{F}\ \text{TF})) \rightarrow ((\mathbf{F}\ \text{AND}) \wedge \neg(\mathbf{F}\ \text{MRM}))$
(6) $((\mathbf{F}\ \text{SN} = pos) \wedge (\mathbf{F}\ \neg\text{TF})) \rightarrow \mathbf{F}\ \text{MRM}$
(7) $(\text{CI-BCT} \rightarrow (\neg(\mathbf{F}\ \text{tumour-excision}) \wedge \mathbf{F}\ \text{MRM}))$
(8) $(\neg\text{CI-BCT} \wedge \text{CI-SNP}) \rightarrow \mathbf{F}\ \text{tumour-excision}$
(9) $(\neg\text{CI-BCT} \wedge \text{CI-SNP} \wedge (\mathbf{F}\ \text{TF})) \rightarrow ((\mathbf{F}\ \text{AND}) \wedge \neg(\mathbf{F}\ \text{MRM}))$
(10) $(\neg\text{CI-BCT} \wedge \text{CI-SNP} \wedge (\mathbf{F}\ \neg\text{TF})) \rightarrow \mathbf{F}\ \text{MRM}$

Assumption (1) and (2) deals with the use of sentinel node procedure and the order between this and the excision of the tumour. Assumptions (3) to (6) are concerned with paths (1) to (4). Assumption (7) deals with path (5). Finally assumptions (8) and (9) deals with path (6) and (7).

6.1.5 Model Checking Results

As explained in Section 6.1.3, the medical management stated by the IKO protocol is less precise than the medical management performed in practice. Typically, one would expect the medical management in the protocol to be underconstrained when compared to the medical management in practice. With the Cadence SMV model checker we were able to verify that all paths described by Δ, except (2), can occur in the IKO protocol. Path (2) does not hold in the IKO protocol because it recommends a MRM whereas [Roses, 2005] recommends a mastectomy, i.e., axillary dissection is included in the medical management according to the protocol, but not according to [Roses, 2005]. The guideline does not provide specific evidence related to this advice, which suggests that both possibilities are acceptable. Nonetheless, such differences could be discussed with medical experts to find out whether the protocol or textbook is incomplete or incorrect.

Because treatment path (2) from the medical textbook is not part of the protocol, it follows that sentence (4) of Γ can not be coherent with the model (i.e., from (4) it follows the antecedent of (4) is false), so in this form it is not usable. We could therefore either adapt the assumption so that it corresponds to the guideline or omit it. Here, we have omitted it. Let Γ' be Γ without (4), then we verify each guideline constraint φ by model checking the modular statement (cf. Section 2.3.2) $[\Gamma']M\langle\varphi\rangle$ using SMV on the Asbru model

of the IKO protocol. This shows that constraint (9) does not hold in the Asbru model of the IKO protocol, indicating a difference between protocol and guideline with respect to medical management in practice. The reason for this difference can be tracked back to preliminary evidence stated in the guideline, which has a low certainty degree, i.e., a low *level of evidence*. Although, in this case the difference between protocol and guideline is clear and could also have more easily been found through an informal analysis, this is largely because the protocol and guideline have a very similar structure and their recommendations are almost identical. However, the approach taken is independent of the underlying structure of the protocol and guideline. Therefore, this case study shows that formal techniques can be used to compare guideline and protocol independent of their underlying document structure.

The resources to check that paths from the decision tree exists were minimal. The number of nodes in the binary decision diagram (BDD) was only 100,000, and terminated within 3 seconds on a modern pc. The number of BDD nodes allocated for checking that the protocol conforms with the guideline was 250,000, and time spent on the verification of all the constraints took 175 seconds. While it is to be expected that more resources will be needed for larger case-studies, we have not used advanced techniques such as bounded model checking, variable ordering, or proof decomposition to reduce the computational complexity.

6.1.6 Discussion

The aim of this work was to use formal methods for obtaining insight into the differences and similarities between guidelines and protocols, based on the assumption that protocols should be looked upon as local modifications of guidelines. In the work presented in this section, the view was taken that clinical guidelines and protocols usually only specify necessary but not sufficient constraints on medical management.

In our study we have set up a modular model checking approach for checking the conformance of a protocol to the guideline from which it has been adapted. In this approach, clinical guidelines and protocols are considered to be constraints on medical management. Medical guidelines are representable as temporal logic formulas whereas protocols are interpretable as executable models. Furthermore, medical management as used in practice was used as additional background knowledge for restricting the guideline and protocol thereby rejecting treatment paths which are illogical for medical management in practice. In this chapter we have applied the modular model checking approach to the CBO breast cancer guideline, the IKO protocol, and used [Roses, 2005] for additional background knowledge. We have shown that with this approach interesting differences between guidelines and protocols with respect to background knowledge can be obtained. These differences can then be communicated to healthcare professionals for further clarification. In our case, one of the medical oncologists responsible for the IKO protocol confirmed that some

differences found could be traced back to low levels of evidence, suggesting that such differences may be ignored.

The research presented in this chapter is a novel approach in locating differences between protocols and guidelines giving a promising starting point for further investigating the relations between guidelines, protocols, and medical management in practice. Some questions still remain for further research. Firstly, a limitation of this research is that it was only based on one reference protocol on breast cancer treatment, selected at the start of this research. A second protocol for this type of breast cancer by the NKI (the Netherlands Cancer Institute) is available; however, this protocol is very different from the first IKO protocol and the guideline, resulting in other challenges than those discussed in this chapter. Secondly, a question that emerged during the course of our research was whether the level of evidence as indicated by the oncologist can be incorporated in our approach as differences based on low levels of evidence can be ignored. Thirdly, formally obtaining differences between protocol and guideline may be useful for protocol designers. In this research, we have only been able to find end point protocols; as a consequence, the transformation process could only be described as consisting of a single step, which in reality may be a more iterative process.

6.2 Critiquing

As a second application area for the use of model checking formal models of clinical guidelines or protocols, we focus on spotting and analysing differences between the proposed actions taken by a medical doctor, and a set of, 'ideal' actions as prescribed by the computerised guideline, which is called *critiquing*.

Model checking takes domain knowledge, called a system description, and sequences of actions as input. In this case, a formalised guideline is taken as a system description; the actions that have been performed on a specific patient are represented as a temporal formula. Model checking then involves investigating the consistency of the formalised guideline and actual treatment. The exploration of the use of model checking in the analysis of medical knowledge (guidelines and patient data) with the purpose of critiquing, is the innovative part of this work.

6.2.1 Approach

The common feature of a critiquing system is that the user of the system provides as input (1) a problem description (e.g., patient symptoms), and (2) a proposed solution (e.g., a treatment plan). This second input is what distinguishes critiquing systems from the more traditional expert systems, which only take a problem description as input [Silverman, 1992, Gertner, 1995]. The second input to a critiquing system, i.e., a proposed solution, is typically the output of an expert system.

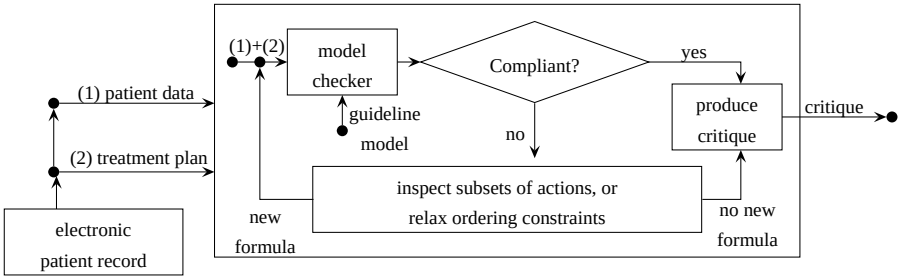


Figure 6.5: Critiquing approach using model checking. Given patient data and a treatment plan as input (temporal specifications), the critiquing system uses a model checker to verify consistency with respect to a guideline model (state transition system) to generate a critique (empty in case of compliance).

In our approach of critiquing medical treatment plans using model checking, the input to the system consists of patient data and a treatment plan (cf. Figure 6.5). Patient data consists of patient symptoms and test outcomes measured for the patient, whereas the treatment plan consists of all actions (to be) performed by the practitioner. As the critiquing process is difficult to accept by practitioners when they are continually interrupted to provide input to the system, both patient data and treatment plan are typically provided by electronic records. We will assume that these are given to the system as temporal logic formulas.

The critiquing system uses the patient data and treatment plan as specifications that need to be checked against a formal model of the guideline, i.e., a state transition system. When the specifications are consistent with the guideline model, no critique needs to be generated as the proposed treatment plan conforms with the guideline. In case an inconsistency is found between the specification and the guideline model, the specification is weakened to get insight to which extent the treatment plan is consistent with the guideline. There are two possible reasons for the incompatibility:

Non-compliant order: It is possible that each of the actions in the treatment plan can be applied to this patient, but only in a different order than the treatment plan proposes. This can be established by removing the order between some of the actions in the treatment plan.

Non-compliant actions: Another possibility is that, according to the guideline, some of the actions cannot be prescribed at all for the patient in question. This can be investigated by considering a subset of the actions in the treatment plan.

The approaches can be combined and lead to further insight into the nature of the detected inconsistency allowing the system to exploit these insights into a critique, which is then given to the practitioner.

6.2.2 Temporal Logic and Critiquing

Each path in the state transition system can be considered a patient who is given a treatment that is consistent with the recommendation described by the guideline. Global properties of the guideline can be checked using LTL formulas or CTL formulas starting with **A**, for example, '**AF** radio-therapy', denotes that in each possible treatment, somewhere in the future radio-therapy is applied.

In the context of critiquing, CTL properties always start with an **E**, i.e., it is established that *some* treatment path exists in the guideline where the proposed treatment is described. For example, abstracting from the patient, a treatment given by a sequence of actions $\alpha_1, \alpha_2, \ldots$ can then be represented as:

$$\mathbf{EF}(\alpha_1 \wedge \mathbf{EX}\,\mathbf{EF}\,(\alpha_2 \wedge \mathbf{EX}\,\mathbf{EF}\,(\ldots))) \tag{6.1}$$

i.e., in some treatment α_1 is done, and after that α_2, etc. In general, CTL model checking is more efficient than LTL model checking, however, in case we do not know the order between the actions, a CTL formula consists of a disjunction of each possible order of actions and considers the existence of each order. In case of n actions, with all order unknown, this leads to formulas of size $O(n \times 2^n)$. Similarly, when global properties of the treatment path are introduced, for example the state of the patient or the fact that some action *never* occurs, such knowledge becomes difficult to express. Assume for example a global property described by β, then Formula (6.1) must be rephrased to the rather complicated formula:

$$\mathbf{E}(\beta\,\mathbf{U}\,(\alpha_1 \wedge \beta \wedge \mathbf{EX}\,\mathbf{E}(\beta\,\mathbf{U}\,(\alpha_2 \wedge \beta \wedge \mathbf{EX}\,\mathbf{E}(\ldots \wedge\ \mathbf{EG}\,\beta))))) \tag{6.2}$$

i.e., β holds until at some point α_1 and β (still) holds, after which β holds, etc.

Usually, the knowledge is reasonably complete and the global information is sparse, however, for a more succinct representation we can either use a more expressive logic such as CTL* (cf. Section 2.1.1) or consider LTL model checking. In the latter approach, one possibility is again modular model checking (cf. Section 2.3.2), where the model is restricted using an LTL formula to those traces where the formula is valid. Thus, to prove the *existence* of a treatment in this approach, it is required to verify that the model restricted to a specification of a certain patient and treatment is not empty. Let φ be an LTL formula and $[\varphi]M\langle\bot\rangle$ denote that the set of LTL assertions φ leads to an empty model, i.e., φ describes a trace not present in the model. In contrast, if $[\varphi]M\langle\bot\rangle$ is shown to be false, then M can not be empty when restricted to φ proving that the trace described by φ exists in the model M. Formula (6.2) can thus be verified by showing that

$$[\mathbf{G}\beta \wedge \mathbf{F}(\alpha_1 \mathbf{XF}(\alpha_2 \wedge \ldots))]M\langle\bot\rangle \tag{6.3}$$

is *false*, i.e., there is no such trace where always β holds and at some point α_1 is followed by α_2. An additional benefit of this presentation is that when order information is absent, the property is typically more intuitively specified. Nonetheless, when there are few actions involved and much of the order information is present, CTL formulas are expected to be more efficient to verify.

6.2.3 Application of the Methodology

Design and choice of case studies

The clinical guideline used is the Dutch breast cancer guideline (cf. Section 3.2.2). The models used here were developed as part of the Protocure project, and is somewhat more elaborate than the model of the protocol as used in Section 6.1. Patient data were obtained from the Dutch Comprehensive Cancer Centre South (CCC), a registry in the Netherlands used for cancer research, planning of services, and evaluation and implementation of guidelines. The data collected concerns breast cancer patients treated in the period January 2003 - June 2004, when the guideline was applicable, and therefore suitable for compliance checks with the guideline. Each patient record consists of 269 variables, which includes information about the diagnosis and treatment.

The patient data from the registry could, in principle, directly be used for critiquing with respect to the guideline. However, matching such data records to the terminology of the guideline is hard [Marcos et al., 2001] and differs from the course commonly followed in medicine. In medical literature, specific patient cases, called *casuistics*, are frequently discussed in detail to gain insight into the way the patient's disease was managed. These papers follow a long standing tradition and are seen as part of the 'education permanente' of the medical profession. Critiquing in this chapter was therefore done casuistically by having the CCC patient data interpreted by medical experts who provided a direct mapping from the patient data in the registry to the guideline. Section 6.2.3 presents in more detail a case-study in critiquing using the casuistic interpretation of the CCC data.

A second case-study is presented in Section 6.2.3, which was obtained from the New South Wales Breast Cancer Institute, Australia.[3] These studies have been developed from the casuistic point of view to "allow clinicians, healthcare professionals and members of the public to examine and understand some of the controversial and difficult aspects of breast cancer management". They are therefore more detailed than the patient data collected by the registry and are more suitable for an investigation of critiquing from a clinical point of view.

Case study 1: ductal carcinoma in situ

The steps of critiquing on one specific patient derived from the data, and subsequently interpreted by medical experts, is illustrated here. The diagnosis

[3] `http://www.bci.org.au/medical/caseindex.htm`

Medical condition: 79 years-old woman. Lesion of right breast: carcinoma in-situ with size between 1 and 2 cm. Two lymph nodes investigated and none positive.
Treatment: sentinel node biopsy + breast-conserving surgery without axillary clearance.

Figure 6.6: Description of patient in conjunction with the prescribed treatment.

and treatment is summarised in Figure 6.6. It can be said that this is a rather typical patient as it is a patient with one of the most frequent diagnoses in the data records. The following property describes the treatment sequence that our example patient has undergone. *"For a patient with diagnosis Ductal Carcinoma In Situ (DCIS), the following sequence of states is possible: the treatment starts, then axillary staging by sentinel node is activated, after which breast conserving therapy is activated"*. To specify and then verify that breast conserving therapy (denoted bct) can take place after axillary staging by sentinel node procedure (denoted asbSN), the following CTL formula is used:

$$\mathbf{EF}(\mathrm{DCIS} \wedge \mathbf{EX}\,\mathbf{EF}(\mathrm{asbSN} \wedge \mathbf{EX}\,\mathbf{EF}\,\mathrm{bct}))$$

A more strict formula could be obtained by assuming that the diagnosis DCIS, holds up to the moment of breast conserving therapy. However, this property stated above turns out to be false as it is, i.e., this treatment is non-compliant with respect to the guideline. In other words, according to the model of the guideline describing the treatment of DCIS, the sequence of actions performed by the doctor is incorrect for this patient. It could also be explained by the fact that, according to the model, at least one of the two actions in patient treatment should not be started, or they should be started in a different sequence. To identify this inconsistency, we reduce the actions that are being performed. If we reduce the sequence to only one action, then both actions are found possible, as shown by the following property (corresponding to the case when only 'bct' is activated as part of the DCIS treatment):

$$\mathbf{EF}(\mathrm{DCIS} \wedge \mathbf{EX}\,\mathbf{EF}\,\mathrm{bct})$$

The new conclusion is that under these circumstances the two actions cannot be activated in this sequence, or the ordering should be reversed.

In the experiment on the seven fairly prototypical patient-cases that can be found in the Dutch CCC data-set, some deviation was found between the guideline and each of the seven prototypical cases. Interestingly, for three of these, some differences could indeed be explained by looking at the new 2004 revision of the guideline. For example, the model of the 2004 guideline con-

tained necessary references to chemotherapy for certain patients as described by the data [Serban et al., 2006].

Case study 2: infiltrating ductal carcinoma

For the second case study we have more elaborate information available. It concerns a patient who is a female with a lump in the 3 o'clock position of the right breast and a second lump just above this. No palpable axillary nodes or other abnormalities were found. The mammography revealed no focal mass, grouped microcalcifications, or anatomic distortion. Finally, the histopathology showed two lesions: both infiltrating duct carcinoma, 20mm in size, and with similar morphology. The sentinel nodes were mapped using lymphoscintigraphy and a biopsy was taken of a right axillary lymph node and an internal mammary node (the sentinel node procedure). In the right axillary lymph node, no malignancy was found. However, in the internal mammary node, metastatic carcinoma was identified. The treatment consisted of a total mastectomy of the right breast with immediate reconstruction. The axilla was treated by means of an axillary clearance and re-section of two further internal mammary nodes at higher levels (these were sampled partly because of the original pathology finding and partly because of ready access to the IMC).

The vocabulary of the guideline does not include the term 'infiltrating ductal carcinoma', but rather discusses 'operable invasive breast cancer' (OIBC). According to the guideline, operable invasive breast cancer is defined as T1-2 N0-1 M0, i.e., a tumour smaller than 5cm, with maximally one lymph node positive, and no distant metastasis. On basis of information provided by the diagnostic tests, the patient can be considered part of this patient group. Each of the three interventions (sentinel node procedure, mastectomy, and axillary clearance) can be mapped to terms found in the guideline. This can be done with reasonable confidence, however, some details have to be ignored such as the re-section of the internal mammary nodes as part of the axillary clearance, as this part of the treatment is not mentioned in the guideline. With respect to the order between interventions, it is only clear that the sentinel node procedure (asbSN) is performed before the other two interventions.

The treatment can again be critiqued using a CTL proof obligation, but as some of the information is missing here, we illustrate critiquing using modular model checking. The proof obligation is then described by $[\varphi]M\langle\bot\rangle$ where

$$\varphi = \mathbf{G}\,\mathrm{OIBC} \wedge \mathbf{F}\,(\mathrm{asbSN} \wedge \mathbf{X}(\mathbf{F}\,\text{axillary-clearance} \wedge \mathbf{F}\,\text{mastectomy}))$$

The proof obligation $[\varphi]M\langle\bot\rangle$ is true, showing that this combination of interventions is *not* possible (cf. Section 6.2.2). The reason for this can be further analysed by removing one of the order constraints between the two actions yielding:

$$\varphi' = \mathbf{G}\,\mathrm{OIBC} \wedge \mathbf{F}\,\mathrm{asbSN} \wedge \mathbf{F}\,\text{axillary-clearance} \wedge \mathbf{F}\,\text{mastectomy}$$

As $[\varphi']M\langle\bot\rangle$ is true, the formula φ' is further weakened by removing one of the interventions from the conjunct, i.e., removing one of the three interventions provides a new proof obligation. The results of model checking these proof obligations show that the guideline model does not contain a trace with both a sentinel node procedure and axillary clearance for this patient, while all other combinations appear to be possible. Thus, the conclusion is that the combination of actions that are being prescribed is non-compliant with respect to the guideline.

6.2.4 Related Work

The use of the term *critiquing* to describe a system that criticises the solution provided by a human can be attributed to Miller [Miller, 1984], who developed his ATTENDING system for critiquing anaesthesia management. Although critiquing has first been used for evaluating medical treatment plans, since then it has been applied to a wide variety of problems such as engineering design, decision making, word processing, knowledge base acquisition, and software engineering [Silverman, 1992]. At the end of the 1990s, when several guideline representation languages were introduced, critiquing using guidelines became a topic of interest, e.g., the approach of Shahar et al. [Shahar et al., 1997]. In contrast with previous work, in this approach the patient states are considered for critiquing, besides the physician's actions. Advani et al. [Advani et al., 1998] argued that a critiquing system should adjust its critique for cases when the physician's actions are following the spirit and overall goals or intentions of the guideline designers, even though the actions deviate from the guideline. However, in [Marcos et al., 2001], a case study showed that intentions of the protocol are often implicit and moreover, the intentions reported by experts almost always differ, which makes it hard to model. Recently, there was some progress to overcome this difficulty [Sips et al., 2006], which might be interesting to integrate in our proposed methodology.

6.2.5 Discussion

The main conclusion of this work is that it is, in principle, possible to use model checking on formalised models in order to critique medical guidelines against patient data. We have shown how critiquing can be characterised in temporal logic and have applied this to a case study on the treatment of breast cancer. The strong aspect of this technology is the high degree of automation, making it suitable for deployment in a critiquing system.

Model checking provides additional value to a simulation-based critiquing of an operational version of the guideline. Such critiquing based on running the operational guideline model through an interpreter only checks the consistency of a patient record against a single trace through the guideline (namely, the one chosen by the interpreter), while model checking compares the patient record

against all possible traces through the guideline. This difference is crucial when the guideline is underspecified, which is usually the case, and therefore contains non-deterministic choices between treatments.

The fully automated nature of model checking also brings a weakness with it: model checking only *detects* inconsistencies, but does not contribute to the interpretation of the inconsistency. In general, model checking can construct a counter-example illustrating the inconsistency, which is often a very good guide towards tracing its source. However, this only works when model checking global properties, i.e., properties dealing with all possible treatment paths, while in Section 6.2.2 we argue that critiquing deals with formulas that establish the existence of an individual treatment, thereby making it impossible for the model checker to generate a counter-example. In this chapter, we have proposed some general strategies to deal with this (repeated experiments with weaker specifications by relaxing order constraints and by removing actions).

A general conclusion with respect to the breast cancer case study that can be drawn is that a closer correspondence is needed between the processes of guideline construction and data-collection. In fact, this is currently already being partially implemented by the Dutch Institute of Healthcare Improvement: newly constructed guidelines are currently being equipped with a data-collection dictionary, which will ensure the correspondence between collected data and guideline terminology.

Even though the steps in the analysis of the case studies was done manually, it is not difficult to see how to automate this process since the temporal formulas could be generated mechanically. A more challenging question is how to use the result of this process for the construction of a human readable critique. In evidence-based guidelines, explanation and references are often provided, however, formal models of guidelines often abstract from this information making it difficult to provide elaborate information to the practitioner. This is an interesting topic for future research.

6.3 Conclusions

General conclusions with respect to model checking in context of clinical guidelines are speculative due to its preliminary state of research. What is shown in this chapter is that in certain circumstances, when there is sufficient knowledge available about the possible courses of treatment, model checking is a feasible technique. Moreover, [Bäumler et al., 2006] showed that, when focusing on a single chapter of the breast cancer treatment, it is often possible to do model checking automatically. For some chapters, however, model checking was problematic though it may be different for other clinical guidelines. While further empirical research could shed more light on this issues, modelling a single guideline takes many man-months. Moreover, the nature of a guideline highly depends on the guideline development method, making results of one guideline development organisation difficult to compare to others.

Chapter 7

Language Fragments for Guideline Formalisation

The results obtained in the previous chapters have offered motivation for the further investigation of the use of logical languages for guideline formalisation. In this chapter, we zoom in on two aspects of clinical guidelines and these are then studied in some detail from a formal, logical point of view. To start, we reconsider some of the reasons why we thought the alternative methods to be worth studying.

In Chapter 5, use was made of the task-network paradigm for modelling clinical guidelines. The task-network languages are mainly developed to *execute* guidelines and therefore either lack formal semantics or the formal semantics is extremely intricate [Schmitt et al., 2006b]. Furthermore, guidelines often lack sufficient detail, which makes the idea of describing guideline as tasks and, thus, in terms of task decomposition, less suitable. A third reason for looking at alternative methods for the formalisation of guidelines is abstraction. For the successful application of these formal methods it is necessary to describe a guideline at an appropriate level of abstraction, which is difficult to accomplish in task-based networks. In Chapter 4, we employ standard temporal logic in order to reach this goal; however, on further reflection, the question is raised if a more specialised language would not be more appropriate for this purpose.

Many textbooks on logic are reductionist in nature as they provide the bare minimum of concepts in order to represent all what is part of the logic. For example, in temporal logic, the 'until' operator may be used to define many of the standard operators such as 'always' and 'eventually'. However, in order to represent knowledge, languages which only provide primitive operators have little practical value as a richer language may make modelling easier. Therefore, the aim of this chapter is not to introduce a new 'minimal' type of logic. Instead, the aim is to provide a number of ontological considerations with

respect to the formalisation of guidelines. The aim is then to provide ways to formally represent certain phenomena which point out the underlying notions of guidelines. The great variety in knowledge that is present in guidelines makes it impossible to do this in full generality. Task-network languages, such as Asbru and PRO*forma*, focus on the management of the guideline, whereas, other important types of knowledge are ignored. The results of Chapters 4 and 5 show that this cannot always be justified when checking the quality of a guideline.

First, we propose a simple language defined in terms of set theory for guideline formalisation, inspired by the insights provided by our analysis of the development of clinical guidelines summarised in Chapter 3. Then, we study how a logical language can used to accomplish similar goals and compare the result to the former language. Second, we study a natural interpretation of clinical guidelines as a sort of action planning, where execution of one action may be a consequence of failure to succeed of another. It appears that the concept of failure, as used in exception handling in programming languages, offers this natural interpretation. It is, subsequently, formalised as a first-class citizen of a temporal logic.

7.1 A History-based Formalisation of Medical Guidelines

We further elaborate on the analysis given in Section 3.3. Additional examples to motivate the formalisation are provided in the text.

To be able to verify quality criteria of clinical guidelines using formal methods, we need to have a language that can be used to express quality criteria that can be related to the key elements in a guideline. In this thesis, it was stated several times that the key elements in clinical guidelines are (at least) order in time, patients, and interventions. Here, we discuss our choices for a language for the formal representation of those key elements, used in the remainder of the section. These elements may be particularly useful in the development of a guideline and allows comparing the available evidence against recommendations given to physicians.

7.1.1 Histories

A clinical guideline contains descriptions of processes concerning the disease, medical management and recommendations. Typical elements in the description of a patient are symptoms, signs and other *observable* elements. Because many of these elements are unknown and often irrelevant we have chosen to define the state space as a many-sorted first-order logic including equality; the set of all well-formed expressions in this logic is denoted by State, as basically we use the logic to describe the state of the patient (group) under consideration at a particular time point. Note that the word 'state' is not used here in a technical sense (i.e., a *complete* description of the patient). Instead, the

state refers to a description of the patient that can be found in, e.g., a patient record.

Let there be a structure $\mathcal{A}$ consisting of a domain for every sort σ and an interpretation I of every constant c^σ of a given sort to the domain of this sort such that $I(c_i^\sigma) \neq I(c_j^{\sigma'})$ if $i \neq j$, i.e., we assume unique names. Let State be a language built up inductively consisting of terms and propositional connectives in the usual manner such that elements of State can be interpreted on the structure. For example, 'temperature $= 37 \vee$ systolic-blood-presure $= 120$' is a typical element of State.

Interventions include all kinds of medical actions that influence the condition of a patient or the environment of that patient. We formalise this as a countable set of interventions $\mathsf{Interventions}$. The interpretation of a set of interventions $I \subseteq \mathsf{Interventions}$ is that the interventions in I occur in parallel. For example, the use of a number of drugs during a longer period of time.

Unlike for states, we assume a closed-world assumption for the interventions: if it is described that a patient received a treatment I and there is an i s.t. $i \notin I$, then i was not applied. This corresponds to the use of medical files, where a medical doctor does not expect an exhaustive description of a patient, but if the medical record does not list a certain treatment, then it is assumed, by default, that this treatment has not been administered to this patient.

Let $\wp(X)$ denote the powerset of X and let $[V \to W]$ denote the function space of functions $f : V \to W$. We then define a history as follows:

Definition 7.1. *A history is defined as an element of the set* $\mathsf{History}$ *such that:*

$$\mathsf{History} = [\mathsf{Time} \to ((\mathsf{State} \times \wp(\mathsf{Intervention})) \cup \{\epsilon\})]$$

such that Time *is a totally ordered set and* ϵ *has the interpretation 'undefined'.*

The assumption that Time is totally ordered is not strictly necessary, though it is convenient in upcoming paragraphs. Note that these elements of Time do not express anything about the absolute difference between the time points. So the distance between t_0 and t_1 is not necessarily the same distance as the distance between t_1 and t_2. We denote the total order as $\prec$ and as a convention, it holds that $t_i \prec t_j$, whenever $i < j$. In addition to being imprecise about certain patients it also allows us to 'instantiate' patients of a certain patient-group by adding patient-specific information to this history.

Consider the following example from the CBO breast cancer guideline (cf. Section 3.2.2) :

> After a mastectomy or breast-conserving treatment, there is an increased risk of movement problems, impaired sensation, pain, and lymphoedema. Adjuvant radiotherapy increases the risk of limited movement of the shoulder and of lymphoedema. Physiotherapeutic intervention can have a positive effect on the recovery of mobility and functionality of the shoulder joint. Early initiation of intensive

> remedial therapy (in other words, during the first postoperative week) has an unfavourable effect on the wound drainage volume and duration.

There are several possible ways to formalise this excerpt depending on the focus of the modeller. One possibility is to pick some patient-group, for example the patient-group which receives physiotherapy too early after the mastectomy. Then, a lymphoedema develops resulting in high drainage, i.e.,

$$\{(t_0, \text{breast cancer}, \varnothing), (t_1, \text{breast cancer}, \{\text{mastectomy}\}),\\ (t_2, \text{lymphoedema}, \{\text{physiotherapy}\}), (t_3, \text{high drainage}, \varnothing)\}$$

7.1.2 Expectations

When dealing with guidelines, we are concerned with the dynamic aspect, for example, the description of how a history is *expected* to continue. As a consequence, this means that the history is extended with new information. A typical example is an expectation of a treatment, i.e., the expected effects of a treatment. First, some notation is introduced: let $dom(h)$ denote the domain of the function h, i.e., $\{t \in \mathsf{Time} \mid h(t) \neq \epsilon\}$, and max and min the maximum and minimum, respectively, of a set S that is totally ordered. Note that if a minimum or maximum does not exist then their maximum and minimum are undefined.

Definition 7.2. *Given a history h, then (1) $maxt(h) = max(dom(h))$, (2) $mint(h) = min(dom(h))$.*

Note that we assumed that the time is totally ordered, so there is at most one minimum and maximum, and otherwise they are undefined.

Definition 7.3. *Given a history h and h' then h' is an extension of h iff (1) $dom(h) \subseteq dom(h')$ and (2) for all $t \in \mathsf{Time}$: $h(t) \neq \epsilon$ implies $h(t) = h'(t)$. The extension is strict if $h \neq h'$.*

Definition 7.4. *The projection of a history h to two elements $t, t' \in \mathsf{Time}$, denoted as $\langle h \rangle_{(t,t')}$, is defined as the history h' such that: (1) $dom(h') \subseteq dom(h)$, (2) for all $t \in \mathsf{Time}$: $h(t) \neq \epsilon$ implies $h(t) = h'(t)$ and, (3) $t'' \in dom(h') \Rightarrow t \prec t'' \prec t'$.*

Extensions extend a description of the patient at points in time that were undefined. Projections restrict a history to certain time points. Obviously, a history is always an extension of a projection on itself.

Definition 7.5. *The right projection of a history h to the elements $t \in \mathsf{Time}$, denoted as $\langle h \rangle_{|t}$, is defined as the history h' such that: (1) $dom(h') \subseteq dom(h)$, (2) for all $t \in \mathsf{Time}$: $h(t) \neq \epsilon$ implies $h(t) = h'(t)$ and, (3) $t' \in dom(h') \Rightarrow t' \prec t$.*

The right projection describes a projection to a certain upper limit in time. With these definitions, we define the expectation function as follows:

Definition 7.6. *The expected continuation of a given history is the function space:*

$$E = [MHistory \to \wp(History)]$$

where $h \in MHistory$ *iff* $h \in History$ *and* $max t(h)$ *is defined, such that for each* $e \in E, h \in MHistory$ *it holds:*

1. $e(h) \neq \varnothing$
2. $h' \in e(h) \Rightarrow h'$ *is a strict extension of* h
3. *Let* $M = \max(dom(h)), i \geq M$*: if* $h' \in e(h)$ *and* $i \in dom(h')$*, then there exists* $h'' \in e(\langle h' \rangle_{|i})$ *such that* h'' *is an extension of* h'*.*

Condition (1) expresses that if we have an expectation about a history, then it introduces new information. The second condition says that expectation functions only *extend* histories, i.e., no information is lost. Finally, the third condition makes sure that an expectation function is consistent with itself. Informally, it means the expectation does not contradict expectations of expected histories, i.e., if we have expectations of expected histories, then these expectations do not violate our original expectations. For example, if we expect that it will rain tomorrow morning after which it will get sunny, then just on the basis of it raining tomorrow morning (and no additional information), we should still consider it possible that it will become sunny. Otherwise, our original expectation seems unjustified.

Example 7.1. Consider a patient p with breast cancer:

$$p = \{(t_1, \text{breast cancer}, \{\text{chemotherapy}\})\}$$

The use of chemotherapy can cause an infection, which we can describe as an expectation $e(p) = \{h\}$ where

$$h = \{(t_1, \text{breast cancer}, \{\text{chemotherapy}\}), (t_2, \text{infection}, \varnothing)\}$$

Note that this is of course a rather naive example, because in this case there is only one expectation of future events, whereas in a more realistic setting, more alternatives would be listed.

This is not the only dynamic aspect concerning clinical guidelines. For example, in the context of so-called living guidelines, the guideline is revised based on new evidence. In practice, this means that either a new patient-group is described by a new history, or the description of the patient-group is narrowed to a more detailed description. This idea is formalised as follows.

Definition 7.7. *Given an expectation function e, a history h, and $h' \in e(h)$, a proper expansion of e is the expectation function e' if and only if there exists an h'' in $e'(h)$ such that for all $t \in$ Time: if $t \in dom(h')$, $h'(t) = (s', I')$ and, $h''(t) = (s'', I'')$ then s'' implies s'.*

So intuitively, this means that the expansion of an expectation function details information about situations that the original expectation function considered and additionally consists of new situations that were not considered before.

Example 7.2. Again consider the patient described by p from Example 7.1 and assume we have a expanded expectation function e' based on new evidence which says:

$$e'(p) = \{(t_1, \text{breast cancer}, \{\text{chemotherapy}\}), (t_2, \text{hairloss} \wedge \text{infection}, \varnothing)\}$$

Clearly, by Definition 7.7, this is a proper expansion of the guideline.

7.1.3 A Logical Perspective on Histories

As large parts of this thesis deal with logic, we will look at histories from a logical perspective. We will assume that the semantics of a suitable logic has to be able to deal with the issues that were described in the previous section, i.e., it has to be able to describe time, states, and interventions. With respect to time, we note that natural language, and in particular clinical guidelines, is tensed. Therefore, a tensed formal language for the formalisation of clinical guidelines might well be regarded as natural by domain experts.

Histories

Consider the history of Section 7.1:

$$\{(t_0, \text{breast cancer}, \varnothing), (t_1, \text{breast cancer}, \{\text{mastectomy}\}),$$
$$(t_2, \text{lymphoedema}, \{\text{physiotherapy}\}), (t_3, \text{high drainage}, \varnothing)\}$$

Given the explanation of the meaning of such a history, this history could be described by the following formula in linear temporal logic (assuming for a moment that the time instances above may overlap):

$$\mathbf{F}\ (\text{breast cancer} \wedge \text{Drugs} = \varnothing \wedge$$
$$(\mathbf{F}\ \text{breast cancer} \wedge \text{Drugs} = \{\text{mastectomy}\} \wedge (\ldots))$$

From a knowledge representation point of view, there are some issues with this formalisation. The first issue is with all the nestings in the formula, which makes adding additional information tedious, i.e., the formula is not modular. This can be improved by using the interval temporal logic operators

(cf. Section 2.1.2), i.e.,

$$\top; \text{breast cancer} \wedge \text{Drugs} = \varnothing; \text{breast cancer} \wedge \text{Drugs} = \{\text{mastectomy}\}; ...$$

which is arguably easier to comprehend. The second issue with this formalisation is that the closed world assumption for the interventions is non-monotonic, so this either requires additional specification, i.e., not applying a certain drug should be specified explicitly or a non-monotonic mechanism such as abduction is required.

Expectations

From a logical point of view, expectations describe beliefs about the future. In literature, a distinction was made between two notions of expectations. The first being the more serious beliefs about the future that may not, ever, under any circumstances, be shown to be false. Typically, these beliefs contain propositions that cannot be proven to be false, such as religious beliefs. The second is less strict and can be described as a 'readiness to bet', such as the belief that an aspirin will help to reduce a headache. In [Kraus and Lehmann, 1988], this was investigated by combining belief ($\mathbf{B}$) and linear temporal modalities. The axiom that was proposed for the former type of belief is:

$$\mathbf{B}\ \mathbf{X}\varphi \rightarrow \mathbf{X}\ \mathbf{B}\varphi$$

i.e., if φ is believed to happen next, then in the next moment, φ will be believed. For the second interpretation, the following is proposed:

$$\mathbf{B}\ \mathbf{X}\varphi \rightarrow \mathbf{B}\ \mathbf{X}\ \mathbf{B}\varphi$$

i.e., if φ is believed to happen next, then it is believed that in the next state, φ is believed, i.e., not necessarily φ will happen. In [Kraus and Lehmann, 1988], it is mentioned that finding a natural model for these axioms is an open problem. While this axiom focuses on the belief of an agent, it does provide any means to reason about the actual expectations, which makes the second axiom somewhat weak, whereas the first axiom does not capture the intuitive notion of an expectation that we have in mind in the medical context.

Here we take a different approach by interpreting path quantifiers of branching temporal logic as a type of belief or knowledge operator, i.e., $\mathbf{A}$ stands for all possible futures that an agent considers possible. Pursuing this idea, what is clear is that in CTL the path quantifiers could be interpreted as a type of knowledge operators, as $\mathbf{A}\varphi$ means that φ will necessarily become true, whatever the behaviour of the system. However, this is different for expectations, where there is uncertainty about the future.

The idea of giving a specific interpretation to the path quantifiers is not new; for example, in [Ågotnes et al., 2007], these operators are given a deontic interpretation. Such a logic is different from the logic

CTLK [Fagin et al., 1995], in which CTL is combined with an additional **K** operator for expressing knowledge. Similarly, the logic introduced in [van der Meyden, 1994] is LTL is combined with such a knowledge operator. Such logics allow arbitrary reasoning about knowledge orthogonal to the temporal reasoning, while we want to explicitly focus on reasoning about temporal knowledge. So, for example, we will not be concerned with knowledge *about* knowledge, yielding a conceptually simpler logic. For example, the logic of [van der Meyden, 1994] is not recursively axiomisable, whereas, standard CTL is [Reynolds, 2005]. Of course, we cannot express the axiom by Kraus and Lehmann, yet, such beliefs do not seem of much relevance in this context.

To illustrate this approach, we define semantics of the CTL operators in terms of histories and expectations and provide some insight in its logical properties. Let h be a non-empty path, which represents the 'current' state and e an expectation function. Furthermore, assume that State is a propositional language, by only considering finite data-types and translating the sentences as described in Section 4.4.2. The semantics of this logical language can then described as follows, where $\mathcal{P}$ is a set of propositional variables, where each model is described by an $h \in \mathsf{MHistory}$ and $e \in \mathsf{E}$:

$$
\begin{array}{lll}
h \models_e p & \text{iff} & (s, I) = h(maxt(h)) \text{ and } s \text{ implies } p \quad (p \in \mathcal{P}) \\
h \models_e i & \text{iff} & (s, I) = h(maxt(h)) \text{ and } i \in I \quad (i \in \mathsf{Intervention}) \\
h \models_e \neg\varphi & \text{iff} & h \not\models_e \varphi \\
h \models_e \varphi \wedge \psi & \text{iff} & h \models_e \varphi \text{ and } h \models_e \psi \\
h \models_e \mathbf{EX!}\,\varphi & \text{iff} & M = maxt(h) \text{ and there exists } h' \in e(h) \text{ s.t. } h'_{|M+1} \models \varphi \\
h \models_e \mathbf{E}\,\varphi\,\mathbf{U}\,\psi & \text{iff} & M = maxt(h) \text{ and there exists } h' \in e(h) \text{ and } t \in dom(h'), \\
 & & t > M \text{ s.t. } h'_{|t} \models_e \psi \text{ and } h'_{|i} \models_e \varphi \text{ for all } i \in dom(h) \text{ and} \\
 & & M \leq i < t \\
h \models_e \mathbf{A}\,\varphi\,\mathbf{U}\,\psi & \text{iff} & M = maxt(h) \text{ and for all } h' \in e(h), \text{ there exists a} \\
 & & t \in dom(h'), t > M \text{ s.t. } h'_{|t} \models_e \psi \text{ and } h'_{|i} \models_e \varphi \\
 & & \text{for all } i \in dom(h) \text{ and } M \leq i < t \\
h \models_e \mathbf{X}^{-1}\,\varphi & \text{iff} & h \neq \varnothing \text{ and } h_{|maxt(h)-1} \models \varphi \\
h \models_e \varphi\,\mathbf{S}\,\psi & \text{iff} & M = maxt(h) \text{ and there exists } t \in dom(h) \text{ s.t. } t < M \\
 & & \text{and } h_{|t} \models_e \psi \text{ and for all } i \in dom(h) \colon h_{|i} \models_e \varphi \\
 & & \text{where } t < i < M
\end{array}
$$

where $(+1)$ and (-1) denote the successor and predecessor, respectively, in the total order $\prec$ for which this time point is defined. As every $h' \in e(h)$ is a strict extension, the weak next operator coincides with the strong next operator (**X!**). Note that with the equivalences as defined in Figure 2.3, the other usual temporal operators can be defined. The main difference between this logic and standard branching temporal logic is that in this logic the history changes as the history is growing. The reason is that, even though we have certain expectations about the future, our expectations do not necessarily have to become true. For example, we may expect that a drug cures the patient, but this does not necessarily mean that the patient will be cured. For example,

the following is a consistent formula:

$$\mathbf{P}\,\mathbf{AG}p \wedge \mathbf{AG}\neg p$$

meaning that in the past we have expected p, yet now p is not expected anymore, in fact, $\neg p$ is now to be expected. On the other hand, we do have:

$$\mathbf{EF}\,p \rightarrow \mathbf{EF}\,\mathbf{EF}\,p$$

i.e., if it is considered possible that p will happen, then this possibility is considered possible. This is also true in standard branching temporal logic, which is not a coincidence. In fact, we have the following relation.

Proposition 7.1. *Let $\models$ be defined on the basis of the semantics as defined in Chapter 2 for serial Kripke models, then it holds that for any formula φ: $\models_e \varphi \Rightarrow \models \varphi$*

Proof. (*sketch*) Observe that Kripke model M can be defined in terms of an expectation function, in particular by interpreting the accessibility relationship as expectations of the next state. We restrict our histories to paths in M, i.e., a path $s_0, s_1, \ldots$ yields a history $\{(t_0, l_0, \varnothing), (t_1, l_1, \varnothing, \ldots\}$, where $l_i = \bigwedge L(s_i)$ (cf. Section 2.1.2 for the definition of L). We define a function e such that $e(h)$ are all paths in M that start with the sequence h. It remains to be shown that this is an expectation function. Due to seriality of M, clearly conditions (1) and (2) hold. Furthermore, condition (3) holds because all expectations of the extension of a history were already in the expectations of the original history.

Now given that $h \models_e \varphi$, it in particular holds for the constructed h, e derived from the Kripke model. Finally, the claim is that there is a one-to-one mapping between the semantics under this interpretation. This can be proven by induction on the structure of the formulae. For example, take $\mathbf{A}\,\varphi\,\mathbf{U}\,\psi$. Each possible path corresponds to each expectation, by construction. Given such a path constructed path π, we have $\varphi\,\mathbf{U}\,\psi$. This implies that for some t, we have $\pi^t \models \psi$, which corresponds to the condition that $h_{|t} \models_e \psi$. Similarly, for each $t' \leq t$ we have $\pi^{t'} \models \psi$ iff $h_{|t'} \models_e \varphi$. The other cases are similar. □

It is not difficult to see that the converse does not hold. For example,

$$\mathbf{AG}p \rightarrow \mathbf{AX!}\ \mathbf{AG}p$$

is a theorem of CTL, but does not follow from the semantics as defined above, as in the next time step there may be other expectations where p does not hold. Expectation functions can be strengthened to accomplish this. At the far end of this spectrum, there are expectations that coincide exactly with serial Kripke and then it holds that $\models_e \varphi$ iff $\models \varphi$. Of course, in such a case, there is little reason to use expectation functions. However, it is useful to consider special types of expectation function, for example, it may be reasonable to assume

that you have actual expectations about the near future. This suggests the following definition.

Definition 7.8. *Let e be an expectation function, then we say that e has expectation lookahead if for all mhistories $h \neq \varnothing$, we have $h' \in e(h_{|maxt(h)-1})$ implies that $h'_{|maxt(h)} = h$.*

This definition expresses that your expectations are consistent about the next time point. Moreover, call h all-knowing, if for all $t \in \mathsf{Time}$, we have $h(t) \neq \epsilon$. These types of expectations can then be characterised by the following proposition.

Proposition 7.2. *If an expectation function e has a lookahead and is defined on all-knowing histories, then it holds*

$$h \models_e \boldsymbol{X}^{-1}\, \boldsymbol{AX!}\varphi \rightarrow \varphi$$

Proof. Suppose $h \models_e \mathbf{X}^{-1}\, \mathbf{AX!}\, \varphi$. Then we have $h_{|maxt(h)-1} \models_e \mathbf{AX!}\, \varphi$. Then for all $h' \in e(h_{|maxt(h)-1})$, we have $h'_{|maxt(h)-1+1} \models \varphi$. So $h'_{|max(h)} \models \varphi$. By lookahead $h_{|max(h)} \models \varphi$. So $h \models \varphi$. □

A similar looking formula is $\mathbf{AX!}\, \mathbf{X}^{-1}\, \varphi \rightarrow \varphi$. However, it is not difficult to see that this is true for any history and expectation function, as anyone would expect that in the next moment, the previous moment is the same as it is right now. This is different from saying that what was expected in the previous moment has come true.

7.1.4 Consistency of Histories

While a logical analysis provides ways to consider what can be derived from the given information, consistency is a more tangible subject in the context of guidelines. Some useful notions of consistency are introduced.

Given $s, s' \in \mathsf{State}$, s and s' are *state-consistent* iff there exists a model $\mathcal{A}$ such that $\mathcal{A} \models s$ and $\mathcal{A} \models s'$. Given $I, J \subseteq \mathsf{Intervention}$, I and J are called *intervention-consistent* iff $I \neq J$. The tuples (s, I) and (s', I') are called *SI-consistent* iff s and s' are consistent and I and I' are consistent. Finally, given two histories h and h'. We call h and h' *history-consistent* iff for all $t \in \mathsf{Time}$, $h(t)$ is consistent with $h(t')$ or if one of them equals ϵ.

Because of the commitment of guideline developers to produce high quality guidelines, we do not expect to find unambiguous inconsistencies within a guideline. This was the reason for focusing on other types of issues related to quality of guidelines in Chapters 4 and 5, i.e., from the perspective of what happens to a patient when prescribing a certain treatment. Nonetheless, it is expected that during the process of developing guidelines different views on how the patient should be treated are considered. Clearly, in many cases these views will be inconsistent and it is therefore of use to detect such inconsistencies.

Every country typically develops their own version of a guideline about similar subjects. Therefore, we have the possibility to simulate the process as described above by comparing recommendations of different guidelines. As an illustration, we compare the Dutch CBO guideline with the Scottish SIGN guideline for breast cancer [SIGN, 1998]. Consider chapter 13.2 of this guideline concerning local recurrence in the axilla after mastectomy.

> nodule(s)/nodes should be excised (...) and if not previously irradiated, locoregional radiotherapy should be given.

Hence, the SIGN guideline describes the following patient-group:

$$h = \{(t_0, \text{breast cancer}, \{\text{mastectomy}\}), (t_1, \text{breast cancer}, \{\text{radiotherapy}\}), (t_2, \neg\text{breast cancer}, \varnothing), (t_3, \text{breast cancer}, I)\}$$

such that $t_0 \prec t_1 \prec t_2 \prec t_3$ and radiotherapy $\notin I$.

The more recent CBO guideline discusses the local treatment of local recurrence following modified radical mastectomy.

> If an isolated local recurrence occurs in a previously irradiated area, high-dose radiotherapy is not an option. In that case, low-dose re-radiation with thermotherapy is the treatment of choice.

Hence, in this case we find that there are patient-groups which are treated taking this guideline into account described by:

$$h' = \{(t_0, \text{breast cancer}, \{\text{mastectomy}\}), (t_1, \text{breast cancer}, \{\text{radiotherapy}\}), (t_2, \neg\text{breast cancer}, \varnothing), (t_3, \text{breast cancer}, \{\text{radiotherapy}, \text{thermotherapy}\})\}$$

such that $t_0 \prec t_1 \prec t_2 \prec t_3$. Hence, we find by definition that h and h' are inconsistent. In particular, we find that on time t_3 this fragment is intervention-inconsistent.

Whereas logic provides means to find an inconsistency between two specifications, it is more difficult to define the notions of types of inconsistency, as done above, for arbitrary formulas. The formalisation of intervention-inconsistency requires an explicit match between time points, e.g.,

$$(h; I) \wedge (h; J)$$

where h is a description of the patient and $I \neq J$ may be consistent if the time of h has not been fixed to a certain length.

7.1.5 Discussion

In this section, we have investigated an interpretation of guidelines as sequences of time-state-intervention tuples, called histories. The history-based approach

offered a special-purpose framework that, as we see it, allows one to express the most important aspects of the *clinical management* of patients. One would, thus, expect that it is also suitable as a knowledge-representation formalism for *clinical guidelines*. The examples given in this section more or less confirm this impression.

In addition, we have provided an logical analysis of the history-based languages. In particular, we have studied some logical results with respect to the idea of expectations. Further work in line of what has been described in this thesis might the addition of more meta-level properties, which would one allow to develop a logic of guideline quality, possibly related to ideas studied in Chapter 4 and 5 of this thesis in terms of logical abduction. Given the fact that we use standard logic CTL to model expectation, though with different logical properties, provides means to exploit standard techniques, e.g., model checking. Furthermore, the language is simple enough to allow one to formally describe the *evolution* of clinical guidelines as they are being developed. This is an application of the theory developed above we think may be worth investigating further, both from a theoretical and application point of view.

7.2 Interpretation of Task Execution using Failures

The rest of this chapter is focused on a specific concept of guidelines that was ignored until now, failures, which provides an interpretation of the application of tasks in practice.

7.2.1 Introduction

For agents that do not have a complete model of their environment or lack certain control over it, it is unavoidable that failures to perform tasks occur. Many systems require some type of robustness against these failures, e.g., robots need to make sure that their task will be accomplished, aviation systems need to make sure that the plane does not crash, etc. For example, in June 2007, there was a crisis in the International Space Station (ISS) as the computer systems failed, even though they were supposedly built robustly using a triply redundant control computer complex[1]. In agent literature, the semantics of failures have been investigated in a logical sense [Rao and Georgeff, 1991] and have been incorporated in agent programming languages [Hindriks et al., 1998]. Similarly, in software engineering, the use of exceptions as first-class citizens in programming languages is wide-spread.

Besides the internal aspects of a system, i.e., a program state or *mental* state of an agent, an important aspect of systems is *behaviour*, i.e., how it acts and reacts in a dynamic environment. To reason about this behaviour, mechanisms that go beyond the scope of classical predicate logic are employed. Since the late seventies, several temporal logics have been proposed to deal

[1] `http://www.spectrum.ieee.org/oct07/5598`. Accessed: October 25, 2007.

with specification and verification of hardware and software systems. In artificial intelligence, many of the logics dealing with actions usually contain some temporal component. In systems where failures heavily determine the final behaviour, modelling of this behaviour is more natural when failures are part of the modelling language. Moreover, we will argue that, since temporal logical formulas can be used to describe behaviour, the failure of a behaviour is best described using a sentential operator, i.e., as a property of a (temporal) logical sentence. This contrasts with other approaches, where failure is seen as a property of primitive events and corresponds to the major contribution of this chapter.

Suppose we are modelling an agent, typically a physician, who treats a patient. As almost all drugs may result in side-effects, it is of great importance that the agent does not over-medicate the patient. Therefore, if the disease is not directly life-threatening, management of a disease should start with a non-invasive treatment where one expects as little side-effects as possible. It is not always possible to measure beforehand if the effects of the treatment will be desirable, as this could require a test that is considered to be too invasive or because it is not known which physiological variable should be measured. As a result, a failure to treat the patient may occur, which means that subsequent actions are required.

Medical treatments are performed in sequence or in parallel. Sequential actions are typically done in case an earlier treatment fails or when a certain physiological state should be reached before a subsequent state can be effective. In such a case, failure to perform a treatment will result in a failure of the whole protocol, as it will block the successful administering of subsequent treatments. Parallel treatments occur for example when multiple drugs are prescribed at the same time. If the effects of these drugs are combined, then the combination of drugs will fail if one of the individual actions fails. If failures are not handled appropriately, it may lead to medical mismanagement, e.g., in case drugs become ineffective due to failure of other treatment components, continuing to administer these drugs is considered bad medical practice. As a consequence, failure handling plays an important role in maintaining the quality of medical management.

This idea of an implicit mechanism that "propagates" the failures throughout the management of a disease leads to the idea that such failures could be seen as exceptions that need to be handled appropriately. This idea is pursued in the next section.

7.2.2 Exception Handling

The idea of handling failures while performing a task is well-known in the context of programming languages by means of exception handling mechanisms. An exception is a failure of an operation that cannot be resolved by the operation itself [Tucker and Noonan, 2007]. Exception handling mechanisms provides a way for a program to deal with them. Many programming languages

(C++, Ada, Java, etc) now incorporate such extensive exception mechanism in order to facilitate robust applications. Typically, such a mechanism consists of two parts. There is a mechanism to *throw* an exception, which sends a signal that an exception has occurred. Second, *catching* an exception transfers control to the exception handler that defines the response that the program takes when the exception occurs. Looking at it slightly differently, one could say that the program determines the plan that is being executed, while the exception handler is able to revise this plan in case an failure occurs.

For the purpose of this section, it is useful to summarise the semantics of exception handling mechanisms. A formal semantic model of exceptions in Java based on denotational semantics [Alves-Foss and Lam, 1999] as well as operational semantics [Oheimb and Nipkow, 1999] exists. The complete mathematical description of these mechanisms is too extensive to be discussed here, as only a small part of the semantics deals with failures. Instead, we give a more general description of the operational semantics of the exception mechanism. A state, here denoted by σ, consists of the heap, values of the local variables, and optionally an exception. Evaluation rules describe how statements change the state, typically in the form $\sigma_0 \xrightarrow{s} \sigma_1$ which denotes that the execution of statement s starting in state σ_0 can terminate in state σ_1. For exception handling, the state is extended with an exception, i.e., we then deal with assertions $\sigma_0 \xrightarrow{s} \sigma_1^e$ which means that the execution of s in σ_0 can terminate in σ_1 throwing an exception denoted by the superscript e[2]. The operational semantics is then also extended with these assertions, e.g., for sequential composition this yields the following two rules depending on whether or not a failure has occurred in the first statement:

$$\frac{\Gamma \vdash \sigma_0 \xrightarrow{s_1} \sigma_1 \quad \Gamma \vdash \sigma_1 \xrightarrow{s_2} \sigma_2}{\Gamma \vdash \sigma_0 \xrightarrow{s_1;s_2} \sigma_2} \qquad \frac{\Gamma \vdash \sigma_0 \xrightarrow{s_1} \sigma_1^e}{\Gamma \vdash \sigma_0 \xrightarrow{s_1;s_2} \sigma_1^e}$$

where Γ defines the context of the rule. Logically speaking, what we see here is that failures are propagated through the semantics of each programming structure. We will show how to incorporate this idea in terms of temporal logic in the next two sections.

7.2.3 Interval Temporal Action Logic with Failure

In this section, we extend the logic of ITL with actions and introduce an operator that denotes failure of the formula. We will refer to this extended logic as ITALF.

Let $\mathcal{A}$ be a set of actions, and $\mathcal{P}$ a set of atomic propositions, which describe a part of the state and is disjunct with $\mathcal{A}$. Models σ we will be working with consists of a (possible infinite) sequence of states $\sigma_0, \ldots$. Each σ_i is defined as $\langle \pi_i, \alpha_i \rangle$, where π_i is a function $\mathcal{P} \rightarrow \{\top, \bot\}$ and α_i a function $\mathcal{A} \rightarrow \{\mathsf{inactive}, \mathsf{active}, \mathsf{failed}\}$. When discussing a σ', we will write α_i' and

[2] abstracting from the different types of exceptions

π'_i such that $\sigma'_i = \langle \alpha'_i, \pi'_i \rangle$. Let the language be extended with actions and an operator **fail**. All semantics given by the language of ITL remains the same. Entailment of ITL will be denoted as $\models_{\text{ITL}}$ from now on and $\models$ will be understood as entailment for ITALF. Actions are interpreted as activations, hence, negations of actions are understood as actions that are not active (i.e., inactive or failed). This is formalised as follows:

$$\sigma \models a \quad \Leftrightarrow \quad \alpha_0(a) = \mathsf{active}$$

For the definition of failure, we need to consider models where we abstract from the difference between inactive and activation, but instead only consider the difference between failures and non-failures. In order to accomplish this, we use the following models, that we denote as $\mathsf{failmodel}(\sigma)$:

Definition 7.9. *For all σ, $\mathsf{failmodel}(\sigma) = \sigma'$ if:*

- $|\sigma| = |\sigma'|$
- *for all i such that $0 \leq i \leq |\sigma|$:*
  - *for all $p \in \mathcal{P}$: $\pi_i(p) = \pi'_i(p)$*
  - *for all $a \in \mathcal{A}$: if $\alpha_i(a) = \mathsf{failed}$ then $\alpha'_i(a) = \mathsf{failed}$, otherwise $\alpha'_i(a) = \mathsf{active}$*

So $\mathsf{failmodel}(\sigma)$ describes σ where non-failures (in particular inactive actions) are interpreted as activations. We can then consider failure as a type of negation in the definition of **fail**, as follows:

$$\sigma \models \mathbf{fail}\,\varphi \quad \Leftrightarrow \quad \sigma \not\models \varphi \text{ and } \mathsf{failmodel}(\sigma) \not\models \varphi$$

To understand this definition, consider φ as a formula that implies that certain propositions and actions are true or false at certain moments in time, even though for some formulas, there is a choice to be made on which point in time. For atomic propositions, the definition is clear and is equivalent to the negation as $\mathsf{failmodel}(\sigma)$ does not evaluate propositions differently than σ. For actions, the situation is more complicated. First, if an action a is implied by φ at a certain moment in time, then φ fails if the action fails on that point in time, which is exactly given looking at $\neg a$ on $\mathsf{failmodel}(\sigma)$. This seems sufficient; however, consider the converse, i.e., that φ implies $\neg a$ at a certain point in time. Then, the formula fails if in fact the action is activated at that point, which corresponds to the first part of the definition. Note that by just looking at $\mathsf{failmodel}(\sigma)$, we can only derive that it must be active or inactive; however, a failure not to do an action does not correspond to this idea.

To get more feeling for this definition, suppose we know that 'surgery', at some point, fails, i.e., **fail** $\Diamond$surgery. A model σ is then a model of this failure if we have two conditions. First, $\sigma \not\models \Diamond$surgery, hence, we know that surgery is never activated. Second, we know that in $\mathsf{failmodel}(\sigma) \not\models \Diamond$surgery. In this

1.	$\mathbf{fail}(\varphi) \rightarrow \mathbf{fail}(\varphi \wedge \psi)$
2.	$\mathbf{fail}(\varphi \vee \psi) \rightarrow \mathbf{fail}(\varphi) \vee \mathbf{fail}(\psi)$
3.	$\mathbf{fail}(\varphi) \rightarrow \mathbf{fail}(\varphi; \psi)$, where φ is objective
4.	$\mathbf{fail}(\varphi) \rightarrow \mathbf{fail}(\varphi^*)$, where φ is objective

Figure 7.1: Propagation of failures in ITALF, where objective formulas are formulas that do not contain any temporal operators.

failmodel, we make a distinction between failures and non-failures, so, if in this failmodel surgery is false everywhere, we know that the $\alpha_i(\text{surgery}) = \mathsf{failed}$ for all i. Hence, in other words, $\mathbf{fail} \Diamond \text{surgery} = \Box \mathbf{fail}\, \text{surgery},$.

We now discussed some properties of this logic. As already mentioned in the previous section, with respect to atomic propositions p, it follows:

$$\mathbf{fail}\, p \equiv \neg p$$

i.e., failure to accomplish p simply means it is not true. So, the formalisation considers failure as a kind of negation. Typically, in the formalisation of medical management, we are interested in formulas such as:

$$\mathbf{fail}\, (p \rightarrow a)$$

i.e., in situation described by p, the action a must be activated. According to the semantics, this is equivalent to $p \wedge \mathbf{fail}\, a$, i.e., if the implication fails, then in the situation described by p, the action a indeed fails. As argued in the previous subsection, failure not to do an action a, i.e., $\mathbf{fail}\, \neg a$ means that a is in fact done. Conversely $\neg \mathbf{fail}\, a$ means that a is either active or inactive. Hence $\mathbf{fail}\, \neg\varphi \neq \neg \mathbf{fail}\, \varphi$.

In general, the definition of **fail** is such as to propagate to larger formulas. This is summarised in Figure 7.1. To prove the first one, for example, consider some model $\sigma \models \mathbf{fail}\, \varphi$. Then $\sigma \not\models \varphi$, so $\sigma \not\models \varphi \wedge \psi$. Moreover, $\mathsf{failmodel}(\sigma) \not\models \varphi$, so also $\mathsf{failmodel}(\sigma) \not\models \varphi \wedge \psi$. Thus $\sigma \models \mathbf{fail}(\varphi \wedge \psi)$. The third one can be proved as follows. Consider $\sigma \models \mathbf{fail}\, \varphi$, where φ is objective. Then clearly for any $n \leq |\sigma|$ we have $\sigma^{[0,n]} \not\models \varphi$. Hence, it follows that $\sigma \not\models \varphi; \psi$. This is similar for the failmodel, hence $\sigma \models \mathbf{fail}(\varphi; \psi)$. The other two cases have a similar proof.

What is interesting is that the calculus rules of the operational semantics of an imperative programming language described in Section 7.2.2 can now be understood in terms of failure inside the logic. For example, for sequential composition, the following calculus rule is sound with respect to the semantics:

$$\frac{\Gamma \vdash \mathbf{fail}(\varphi)}{\Gamma \vdash \mathbf{fail}(\varphi; \psi)}$$

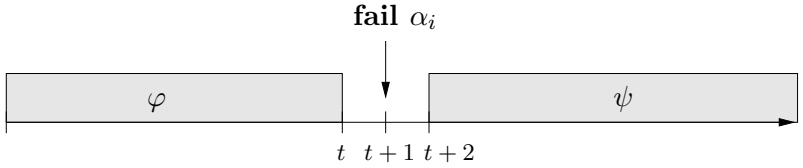


Figure 7.2: Sketch of φ **orelse**$_{\{\alpha_i\}}$ ψ in case of failure of α_i.

where φ is objective, which follow directly from item (3) of Figure 7.1 and modus ponens. In order to describe acting on the basis of failure, we define an additional operator:

$$\varphi \ \mathbf{orelse}_A \ \psi \quad \triangleq \quad \varphi \wedge \neg \mathbf{last}; \mathbf{X!}\,(\mathbf{failstate}_A \wedge \mathbf{X!}\,\psi)$$

where

$$\mathbf{failstate}_A = \bigvee_{a_i \in A} \mathbf{fail}\ a_i \wedge \bigwedge_{a_i \in \mathcal{A}} \neg a_i$$

i.e., φ holds forever, or, an action fails at some point after which ψ holds. We assume that φ is true in at least a unit interval, which prevents failures to occur right away. In Figure 7.2, a model where a failure occurs and is handled is sketched. The definition of the **failstate** ensures that during this time no action can be active, and thus no additional failures may occur. In some sense, this operator may be read as an exception handling mechanism where failures of 'type' A are caught in the execution of φ, such that ψ is executed when this occurs.

7.2.4 Reduction to ITL

In this subsection, we will show how ITALF can be translated to ITL. We thereby give means to exploit the proof techniques that were developed for ITL. To accomplish a reduction to ITL, additional propositional variables are required. We assume we have an infinite number of propositional variables such that we have a (unique) fresh proposition f_a, standing for failure, for each $a \in \mathcal{A}$.

Definition 7.10. *Given a formula φ, define $\Phi(\varphi)$ as φ where every occurrence of some action $a \in \mathcal{A}$ has been replaced with $\neg f_a$.*

Definition 7.11. *The reduction of a formula φ is defined on the structure of*

φ as follows:

$$\begin{array}{lcl}
\mathit{reduce}(p) & = & p \\
\mathit{reduce}(a) & = & a \\
\mathit{reduce}(\neg\varphi) & = & \neg\mathit{reduce}(\varphi) \\
\mathit{reduce}(\varphi \wedge \psi) & = & \mathit{reduce}(\varphi) \wedge \mathit{reduce}(\psi) \\
\mathit{reduce}(\mathit{skip}) & = & \mathit{skip} \\
\mathit{reduce}(\varphi;\psi) & = & \mathit{reduce}(\varphi);\mathit{reduce}(\psi) \\
\mathit{reduce}(\varphi^*) & = & \mathit{reduce}(\varphi)^* \\
\mathit{reduce}(\boldsymbol{\mathit{fail}}\,\varphi) & = & \neg\mathit{reduce}(\varphi) \wedge \neg\Phi(\mathit{reduce}(\varphi))
\end{array}$$

Below, we refer to $\mathsf{reduct}(\varphi)$ as the ITALF formula that is found by applying the definition exhaustively from left to right. The main result of this subsection which provides the connection between ITL and ITALF follows.

Definition 7.12. *Given a ITALF formula φ, intended meaning of the failure propositions is defined as follows:*

$$\mathcal{I}(\varphi) = \boldsymbol{G}\,(a_0 \rightarrow \neg f_{a_0} \wedge \cdots \wedge a_n \rightarrow \neg f_{a_n})$$

where $\{a_0, \ldots, a_n\} \subseteq \mathit{actions}(\varphi)$, such that $\mathit{actions}(\varphi)$ is defined as the set of those $a \in \mathcal{A}$ that is a sub-formula of φ.

Note that $\mathcal{I}(\varphi)$ is finitely bounded by actions in a formula, which is important as the total number of actions in the language may be infinite. Then, we have the following result:

Theorem 7.1. $\models \varphi$ *iff* $\mathcal{I}(\varphi) \models_{\mathrm{ITL}} \mathit{reduct}(\varphi)$

Therefore, reasoning in ITALF can be encoded in ITL. Proof of this theorem can be found in Appendix A.3.

7.2.5 Related Work

Failure has received little attention in formal theories of action. Of course, reasoning of actions had always taken into account the notion of failure, as illustrated by the logic of [Rao and Georgeff, 1991]. However, it is assumed that failure can be added in a relatively straightforward manner. One notable example of where the notion of failure is part of both the syntax and semantics is the approach of Giunchiglia et al. [Giunchiglia et al., 1994]. Its primitive syntactic structure is:

iffail α **then** β **else** γ

And from this, abbreviations are defined such that it allows one to reason conveniently about failures. The semantics is defined in terms of *behaviours* where it said that some behaviours have *failed*, while others are *successful*. Behaviours are defined technically in terms of linear models.

What this language lacks is the notion of time, as behaviours are simply considered a sequence of actions which either fail or do not fail. For medical management, this poses a problem, as failure may occur after a longer period of time. This means that the notion of failure needs a richer structure, so that it is possible to interact between time and failure.

Another important shortcoming for using this language in the context of medical management is that failures is a property of a *behaviour*. As said before, in medical management, actions are often performed in parallel, for example, the administering of a combination of drugs. In such cases, some drugs may fail to reach the required effects, while others may be successful. Hence, in the language decisions need to be made on, not only *if* a failure has occurred, but also *what* action has failed.

7.2.6 Discussion

In this section, we have introduced semantics of failures in interval temporal logic inspired by the exception mechanism that can be found in many programming languages. The practical usefulness of our approach in context of clinical guidelines is illustrated by verifying a fragment of diabetes mellitus type 2 using this logic. However, we think that the ideas and results could be applied in a much wider context. First, the reasoning about failures can have its applications in agent-based systems. Failures to perform tasks are an important aspect for decision making by agents, so having a reasonably rich language for modelling these failures seems justified. Second, in the context of program refinement, the process of (high-level) specifications to implementations of systems, exceptions are introduced at some point to model failure of components. The results of this section makes it possible to abstract of concrete programming construct to describe how control of flow should change in case exceptions occur.

The logic that is proposed here can be seen as a three-valued logic, i.e., formulas are true, false, or failed. Some work has been done to link three-valued logics idea to temporal reasoning [Konikowska, 1998], which is based on Kleen's three-valued calculus that deals with 'unknown' values. This results in different logical properties compared to ITALF, e.g., unknown values propagate over a disjunctions, while failures do not.

7.3 Application to a Medical Guideline

In this section, we consider the modelling of the diabetes mellitus type 2 (DM2) guideline (cf. Figure 4.1) using the failure interpretation of the guideline that was discussed in the previous section. The knowledge in this fragment concerns information about order and time of treatment (e.g., sulfonylurea in step 2), about patients and their environment (e.g., Quetelet index lower than or equal to 27), and finally which drugs are to be administered to the patient (e.g., a sulfonylurea drug).

7.3.1 Modelling of DM2

The physiological mechanisms were originally modelled in temporal logic, which is described in Figure 5.3. The next state of the model is interpreted as a state after some *unknown* time period, which is important, as we will see below, for deciding when to move to a new treatment.

In this Chapter, we mainly focus on the modelling of the guideline fragment of Figure 4.1. The possible actions that can be performed is the set $\mathcal{A}$ consisting of $\{diet, SU, BG, insulin\}$. Each treatment A is a subset of $\mathcal{A}$. Treatment changes if a treatment has failed, which can be conveniently be formalised in ITALF. The main structure of the guideline, denoted by $\mathcal{M}$, is then:

$$\begin{array}{l}\mathbf{G}\ \text{treatment} = \{\text{diet}\} \\ \mathbf{orelse}_{\{\text{diet}\}}\ (\mathbf{if}\ \text{QI} < 27\ \ \mathbf{then}\ (\mathbf{G}\,\text{treatment} = \{\text{SU}\}) \\ \qquad\qquad\qquad\qquad\qquad\ \ \mathbf{else}\ (\mathbf{G}\,\text{treatment} = \{\text{BG}\}) \\ \qquad\qquad\quad \mathbf{orelse}_{\{\text{SU,BG}\}}\ \ (\mathbf{G}\,\text{treatment} = \{\text{SU}, \text{BG}\} \\ \qquad\qquad\qquad\qquad\qquad\ \ \mathbf{orelse}_{\{\text{SU,BG}\}}\ \mathbf{G}\,\text{treatment} = \{\text{insulin}\}))\end{array}$$

where each term treatment $= A$ is an abbreviation for:

$$\bigwedge (\{\alpha \mid \alpha \in A\} \cup \{\neg\alpha, \neg\mathbf{fail}\ \alpha \mid \alpha \in (\mathcal{A} \setminus A)\})$$

i.e., the actions in A are activated, and all other actions are inactive (i.e., false and have not failed). This formalisation includes the handling of the failures in some sense; however, we also need to define in which cases these failures occur. One can think of this as 'throwing' the exceptions during the management of the disease. Define an abbreviation for this as follows:

$$\mathbf{fails}\,\varphi \triangleq \mathbf{X!}\,\mathbf{fail}\,\varphi$$

The guideline does not specify what amount of time is allowed to pass before it can be concluded that the treatment is not effective. Clearly, if a failure occurs immediately, then patients will all receive insulin treatment. Here, we assume the property of the background knowledge that relevant effects with respect to the condition of the patient are known in the next state. Hence, decisions whether the treatment fails can be taken after one step in the execution. These failure axioms are denoted as $\mathcal{F}$ and formalised as follows:

$$\mathbf{G}\,(\alpha_i \rightarrow \mathbf{X!}\,((\alpha_i \wedge \text{Condition}(\textit{hyperglycaemia})) \leftrightarrow \mathbf{fails}\,\alpha_i))$$

for all $\alpha \in \mathcal{A}$.

7.3.2 Verification

Several tools for ITL have been developed, such as the interpreter Tempura [Moszkowski, 1996] and support for ITL in the theorem prover PVS [Cau and B., 1996]. For our experiments, we have used the KIV system, an

interactive theorem prover, as described in Section 5.1.1. Below, we will write sequents $\Gamma \vdash \Delta$ to denote $\mathcal{I}(\Gamma \cup \Delta) \vdash_{\mathrm{KIV}} \mathsf{reduce}(\bigwedge \Gamma \rightarrow \bigvee \Delta)$, where $\vdash_{\mathrm{KIV}}$ denotes the deductibility relation defined by the sound (propositional and temporal) inference rules implemented in KIV.

In the specification of properties presented, we made use of algebraic specification to specify the variables in the background knowledge, though it could be translated to propositional logic if necessary. Furthermore, we made use of some additional variables to represent each treatment (e.g., '*treatmentdiet*' defined as 'treatment = {diet}'), and both failure-states. In practice, this makes the proofs more manageable. The relationship between the actions and these additional variables are defined appropriately in the system, i.e., all the additional propositional variables could be replaced by actions and failure of actions.

Example 1: Diet may be applied indefinitely

The first example is the following property. Let $\mathcal{B}_{DM2}$ be the background knowledge, $\mathcal{M}$ be the guideline, and $\mathcal{F}$ failure axioms, then:

$$\mathcal{B}_{DM2},\ \mathcal{M}, \mathcal{F},\ \mathbf{G}\ \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{normal} \\ \vdash \mathbf{G}\,\mathbf{X}\,\mathrm{Condition}(\mathit{normoglycaemia})$$

i.e., in case the patient has B cells with sufficient capacity to produce insulin, then diet is sufficient for lowering the level of glucose in the blood. As only the failure of diet is relevant in the proof, $\mathcal{M}$ can be weakened to:

$$(\mathbf{G}\ \mathit{treatmentdiet}) \wedge \neg\, \mathbf{last};\ f_{\mathrm{diet}}$$

Symbolic execution, in the context of the background knowledge, leads to the situation where:

$$(\mathbf{G}\ \mathit{treatmentdiet};\ f_{\mathrm{diet}}) \wedge \mathrm{Condition}(\mathit{normoglycaemia})$$

Since we have Condition(*normoglycaemia*), it can be derived that *diet* does not fail, thus in the next step it can be derived that the condition is still normoglycaemia, which is exactly the same situation as we had before. By induction, we can then reason that this will *always* be the case. A more detailed proof can be found in Appendix A.4.

Example 2: Reasoning about the patient in case of failure

Guidelines are not applied blindly by physicians, as the physician has to make a decision for an individual patient on the basis of all known information. As a consequence, a physician might be interested in reasons of failure. Suppose we have an arbitrary patient, then we can prove the following:

$$\mathcal{B}_{DM2},\ \mathcal{M}, \mathcal{F} \vdash \mathbf{fail}(\mathbf{G}\ \mathit{diet}) \rightarrow \mathbf{F}\ \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) \neq \mathit{normal}$$

i.e., if *always* applying diet fails, then apparently the patient has non-normal capacity of its B cells at a certain moment in time. $\mathcal{M}$ is needed here to derive that in case diet stops, a failure has occurred rather than a non-failing termination of diet. Proving this in KIV is similar as the previous example.

Example 3: Level of sugar in the blood will decrease

As a third example, we use one of the quality criteria for the diabetes guideline defined in Section 5.2.2, i.e., the guideline reaches its intention, namely, the level of sugar in the blood will be lowered for any patient group. This property is formalised as follows:

$$\mathcal{B}_{DM2},\ \mathcal{M},\ \mathcal{F}, \mathbf{G}\,(\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin})'') \wedge \\ \mathbf{G}\,\mathrm{QI} = \mathrm{QI}'' \vdash \mathbf{F} \neg \mathrm{Condition}(\mathit{hyperglycaemia})$$

where V'' denotes the value of the variable V in the next step. Our proof strategy consisted of splitting the patient group into groups which are cured by the same treatment, e.g., similar to the previous example, when the capacity is normal, then diet is sufficient.

Consider the example where the capacity of insulin in the B cells is nearly-exhausted. KIV derives from the failure axioms that:

$$\mathbf{G}\,(\alpha_i \rightarrow \mathbf{X!}\,(\alpha_i \leftrightarrow \mathbf{X!}\,(\neg\alpha_i \wedge f_{\alpha_i})))$$

as we may assume that $\mathbf{G} \neg \mathrm{Condition}(\mathit{hyperglycaemia})$, because the negation of this formula immediately proves the property. Furthermore, reasoning with the background knowledge, we can derive that proving $\mathbf{F}\,(SU \wedge BG)$ is sufficient to prove this property, because for this patient group a treatment consisting of SU and BG is sufficient to conclude $\mathrm{Condition}(\mathit{normoglycaemia})$. It is then easy to see how to complete this proof as the failure axioms specify that all the treatments will fail (after two steps), hence symbolic execution shows that eventually the third step will be activated.

7.3.3 Discussion

Compared to the results of Chapter 5, the verification of the investigated properties required significantly less effort. This is mainly due to the fact that in Chapter 5 the guideline was formalised in the guideline representation language Asbru, which yields overhead in complexity due to a complicated semantics. On the other hand, many of the steps that are required in ITALF were done manually, as it is not obvious to predict the correct next step in the proof. For example, it is important during verification to 'weaken' the irrelevant parts of the guideline, which is described by calculus rules such as:

$$\frac{\Gamma, \varphi; \mathsf{true} \vdash \Delta}{\Gamma, \varphi; \psi \vdash \Delta}\ \textsc{weaken chop}$$

making the symbolic execution more efficient. Moreover, failure propositions on the sequent introduce additional complexity, as the human needs to remember the semantics of these propositions in order to apply the relevant axioms. These facts combined makes it interesting to consider more automatic techniques, such as automated theorem proving or model checking.

7.4 Conclusions

In this chapter, we have aimed to uncover some elements of guideline formalisation in order to formally reason guidelines. In the first part of this chapter, we studied possible formalisations of medical knowledge for designing a guideline. The history-based approach has the advantage that it can easily be extended with relevant structure. For example, in [Lucas et al., 2005], the guideline development process was described abstractly using this approach. The logical approach, however, provides direct means to formally reason about the guideline. Moreover, it provides intuitive means to investigate properties of concepts that are of importance to formally modelling of guidelines. In this context, we discussed the concept of expectations. Furthermore, we discussed the failure concept in modelling clinical guidelines, and show how this can be made practical in terms of clinical guidelines. In this case, we were inspired by failures in programming languages; however, other semantics of can be imagined. This is a possible topic for future work.

Chapter 8

Conclusions

The goal of this thesis was to explore the use of formal methods for the verification of clinical guidelines. First, we will summarise the results of this thesis given the research questions that were presented in the first chapter of this thesis. Then, the results are placed in the broader context of the use of formal methods for the development of clinical guidelines.

8.1 Summary of Results

Each of the chapter conclusions describes the contribution of that respective chapter. Here, we reconsider the results on the basis of the research questions that were posed in Chapter 1.

> *Which knowledge is required to investigate properties of clinical guidelines?*

In Chapter 3, we reviewed some of the thoughts from the medical community on the quality of clinical guidelines. The AGREE instrument has been developed by the guideline development community as the principal tool for the assessment of the quality of guidelines. However, this tool focuses primarily on the guideline development process, such as on the quality of the evidence used as a basis for the guideline, and does not really address the quality of a guideline itself. We discussed several aspects of the knowledge underlying guidelines that can be used to study their quality, and an explicit choice was made to focus on properties that deal with the knowledge-based goals of the use of a guideline, i.e., we decided to investigate whether or not the goals of the use of guidelines are reached, taking into account medical background knowledge.

To realise this aim, various types of medical background knowledge were formalised in the subsequent chapters; examples were: knowledge concerning (patho)physiological processes, the ordering of treatments and quality requirements. In Chapter 4, this idea was pursued for individual treatments, whereas

in Chapter 5, this was further refined for complete treatment paths. In Chapter 6, that deals with the refinement of guidelines to protocols, knowledge concerning medical management is added. The rationale is that guidelines should be seen in the context of an actual clinical treatment process, rather than as constraints in isolation.

> *What is a suitable language for representing clinical guidelines in order to apply formal methods?*

In this thesis, doubts were expressed regarding the suitability of current guidelines representation languages for use in conjunction with formal methods. The first reason for this is that standard logical languages already appear appropriate for expressing various abstractions of guidelines, as illustrated by the formalisation presented in Chapter 4. Second, current guideline representation languages have a complicated semantics, which renders the verification process hard (e.g., see the semantics of time-annotations in Asbru [Schmitt et al., 2006b]). For example, the examination of some properties of an Asbru model, verified in Chapter 5, took significantly more effort than the verification of properties on the basis of a logical model, as described in Chapter 7. Nonetheless, guideline representation languages are not likely to disappear soon; therefore, the question how current state-of-the-art guideline representation languages, such as Asbru, might become more amenable to verification remains valid. Some insight into this is given in Chapter 5 through the application of theorem proving to a concrete Asbru model of a guideline.

> *How do formal methods contribute to ensuring the quality of guidelines?*

The process of modelling a guideline often already reveals most of the flaws included in a guideline, which typically have to do with issues such as incompleteness of descriptions and confusing terminology. Actually, some of the flaws may be introduced by the modellers themselves, in translating the document's text into a formal model. As formal methods people usually have had no medical training, they may quite easily come up with the wrong interpretations of guideline text, which thus results into a incorrect formal model. Nonetheless, in the research several flaws were found that may give rise to an ambiguous interpretation of the text.

The second contribution of formal methods is that they make assumptions about the recommendations included in the clinical guideline explicit. In the studies described in Chapters 4 and 5, the quality of guidelines was the main topic of concern and requirements concerning the quality of recommendations was made explicit. One might argue that the ones proposed there are incomplete requirements from a medical point of view; yet, most alternative quality measures can be expressed along similar lines. Unfortunately, such quality measures are not mentioned in current clinical guidelines, as they are more or less taken for granted.

Verification challenges guideline developers to make their underlying assumptions about medical knowledge and their notion of quality explicit. One such challenge is presented for the prescription of insulin in combination with oral antidiabetics for treating diabetics. If one accepts the medical principle of limiting the number of drugs to a patient to the minimum, then this recommendation seems unacceptable. This issue was discovered before [Lucas, 2003]; however, we have shown that such errors can be detected fully automatic given a formal model of a guideline. Moreover, interactive techniques can be applied to find more complex flaws in guidelines, as discussed in Chapter 5.

Which techniques are most suitable for answering questions about clinical guidelines?

In this thesis, we explore the use of three of the most frequently applied techniques, i.e., automated theorem proving, interactive theorem proving, and model checking. For investigating quality, we have made use of theorem proving as the required background knowledge cannot be represented succinctly as a branching model. Even though linear temporal logical formulas can be represented as an automaton (e.g., [Somenzi and Bloem, 2000]), such a model can have up to $2^{O(n)}$ states where n is the number of subformulas [Vardi and Wolper, 1994]. Moreover, the underspecification of guidelines creates serious challenges for applying standard model checking technology [Bäumler et al., 2006]. Further abstractions were needed to keep the state space under control, which raises doubts whether model checking provides a useful technique for quality-checking. However, in cases where guidelines are applied in practice, and thus reasonably well-specified, and where background knowledge is not required, model checking is a suitable choice. This was demonstrated by experiments of which the results are reported in Chapter 6.

Similar to the question whether or not to use model checking or theorem proving, a trade-off has to be made between automated or interactive theorem proving. Automated theorem proving provides a push-button technique for verification, which makes it particularly suitable during the design of a guideline if the complexity of the model can be kept under control. Interactive verification was used in the research described in this thesis in order to overcome the complexity of the Asbru specification. Moreover, an advantage of the interactive approach is that its yields proofs that can be inspected. The proof tree it generates is intuitive and in case a property does not hold, the reason for this is clear by the structure of the proof.

8.2 Limitations and Future Directions

8.2.1 Formalisation

One of the main challenges of applying formal methods to clinical guidelines remains bridging the gap between guideline developers and knowledge about

formal methods needed to carry out verification. The practical usefulness of the work that is presented in this thesis depends on such developments, although there is evidence that these developments will be initiated in the near future. Some progress in this area have already been made, for example, in visualisation [Kosara and Miksch, 2001], interactive execution of guideline representation languages, and into the development of semi-formal modelling languages [Seyfang et al., 2006], although these developments are still fairly preliminary.

The logical language that we have employed in Chapter 4 is conceptually relatively simple compared to language used for the representation of guidelines and complex temporal knowledge as discussed in [Shahar and Cheng, 2000]; however, in principle all these mechanisms could be formalised in first-order logic and could be incorporated in our approach. The suitability of employing a logical languages was illustrated by the investigation in the appropriateness of using similar mechanisms as described in Chapter 7. Finally, we are currently moving into an era where guidelines are evolving into highly structured documents and are constructed more and more using information technology. It is not unlikely that the knowledge itself will be stored using a more formal language. A method to assist guideline developers for looking into the quality of clinical guidelines using, for example, automated verification will then be practically useful.

8.2.2 Reasoning

With respect to the reasoning in guidelines, more general frameworks are imaginable. We have shown that treatment selection in guidelines can be described abductively, which raises the question to which other areas of medicine this may be applied. While abductive reasoning is seen as an essential aspect of theories of model-based diagnosis, it remains to be seen how it relates to reasoning about preventive and supportive interventions. Furthermore, abductive reasoning can be seen as part of more general frameworks of non-monotonic reasoning, such as argumentation frameworks [Prakken and Vreeswijk, 2002], which suggests that in some circumstances a more powerful reasoning method may be required.

Certain characteristic aspects of clinical reasoning, such as reasoning with uncertain knowledge, were ignored in this research. The guidelines under considerations did not contain sufficient amounts of probabilistic information to warrant its consideration. Nonetheless, uncertainty reasoning does play a part in clinical guidelines, even though this is hard to make explicit quantitatively. Qualitative uncertainty can be easily incorporated into the abductive framework as suggested by [Console and Torasso, 1991]. For example, the background knowledge could be extended to incorporate expressions of uncertainty, e.g., instead of writing

$$\mathrm{Drug(BG)} \rightarrow \mathbf{X!}\ \mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down}$$

we could have written:

$$\text{Drug(BG)} \wedge \alpha \rightarrow \mathbf{X!}\, \mathit{release}(\mathit{liver}, \mathit{glucose}) = \mathit{down}$$

where α is a literal expressing the uncertainty of the effects that the drug BG has on the patient (it is called an *incompleteness-assumption* literal in the terminology of [Console and Torasso, 1991]). For the successful application of BG, it requires the physician to assume that α is true, i.e., that it is a 'normal' patient, in order to derive the desired effects. An optimality criterion could then take into account such uncertainty on the basis of, for example, an ordering over these uncertainty literals. However, as we did not include such knowledge in the case study, these uncertainty literals have not played a role in the pathophysiological models. In some guidelines, quantitative uncertainty is present in the guideline, which may require probabilistic representation and inference. Moreover, it could be useful to combine quantitative knowledge from data with the qualitative knowledge of guidelines. In both cases, probabilistic logics may be applicable (e.g. [Poole, 1993, Kersting and De Raedt, 2000, Richardson and Domingos, 2006]).

8.2.3 Quality

One of the limitations of this thesis is that part of the quality of a guideline, and thus of medical decisions in general, was defined as an extremely general proposition, i.e., "optimal with respect to cost", whether it be financial, ethical cost or cost related to the patient preferences. Of course, many cost functions would be silly and medically unacceptable. However, to make this more precise, what is needed is a rational theory of medicine. Unfortunately, it is fair to say that medicine does not have one. As discussed in Chapter 3, decisions are based on a wide range of 'heuristic' principles related to e.g., physiological, ethical, and psychological frameworks and there does not seem to be an articulated theory that ties these considerations together. As for example [Hoey and Todkill, 2001] mentions, the quip "That's all very well in practice, but will it work in theory?" poses a valid question for medicine. Moreover, even if we would be able to define the correct cost function, the cost function would contain many attributes compared to, for example digital systems. Although multi-attribute utility functions (e.g., [Keeney and Raiffa, 1976]) are well-known in decision-analysis, it will be challenging to combine these ideas with the use of formal methods.

8.3 Final Thoughts

In the artificial intelligence community, there has been a trend to move to machine learning methods, as more and more data become available. In this sense, this thesis goes against the tide, as it is mainly concerned with knowledge representation and reasoning, though there are obvious differences between this

work in comparison to the work on knowledge representation at the end of the 1980s, namely the use of modal logics and powerful verification techniques, which have only become more practically useful during the last decade.

There is some initial work on the use of machine learning techniques for the construction of guidelines [Mani and Aliferis, 2007], but a rational theory for decision making is then imperative. First, machine learning techniques are extremely good for building predictive and descriptive models of the data, but cannot be prescriptive. This is related to Hume's is-ought problem, i.e., just by knowing what *is* the case, you cannot know what you *ought* to do. Data-mining techniques are essential to construct medical knowledge, but a gap remains for constructing clinical knowledge. What ought to be the case requires at least knowledge about patients' preferences and some measure of quality. Several machine-learning proposals deal with this problem by assuming utility functions and subsequently maximising the expected utility of a particular treatment; however, it is a question how feasible it is to encode qualitatively described quality criteria in a utility function as these utility functions must contain large amounts of attributes which makes knowledge elicitation difficult.

In order to bridge the gap between biomedical scientific and clinical knowledge, the reasons for studying logical representations are as valid as they were in the early days of artificial intelligence. Large parts of the guideline can be represented using temporal logic, logical sentences are fairly easy to understand (in a sense that they can often be intelligibly phrased in natural language), and a logical justification can be provided. Nonetheless, some of the knowledge in guidelines has been ignored, as it occurs so sporadically. Such knowledge may be integrated with the logical parts using one of the probabilistic logics that have appeared the last couple of years. This thesis can act as a basis for such further investigations.

Appendix A

Proofs

A.1 Proof of Meta-level Property (T2) in Chapter 4

In the formulas below, each literal is augmented with a time-index. These implicitly universally quantified variables are denoted as t and t'. Recall that $g(x, y) = \mathit{down}$ is implemented as $\neg(g(x, y) = \mathit{up})$ and functions f and f' are Skolem functions introduced by OTTER. Both Skolem functions map a time point to a later time point. Consider the following clauses in the usable and set-of-support list:

2 $\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}, t) \neq \mathit{nearly\text{-}exhausted} \vee$
$\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}, t) \neq \mathit{exhausted}$

14 $t \not> f(t) \vee \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}, t) = \mathit{exhausted} \vee t > t' \vee$
$\mathit{secretion}(\mathit{b\text{-}cells}, \mathit{insulin}, t') = \mathit{up}$

15 $\neg\mathrm{Drug}(\mathrm{SU}, f(t)) \vee \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}, t) = \mathit{exhausted} \vee t > t' \vee$
$\mathit{secretion}(\mathit{b\text{-}cells}, \mathit{insulin}, t') = \mathit{up}$

51 $0 > t \vee \mathrm{Drug}(\mathrm{SU}, t)$

53 $\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}, 0) = \mathit{nearly\text{-}exhausted}$

For example, assumption (53) models the capacity of the B cells, i.e., nearly exhausted at time 0 where the property as shown above should be refuted. Note that some of the clauses are introduced in the translation to propositional logic, for example assumption (2) is due to the fact that that values of the capacity are mutually exclusive. This is consistent with the original formalisation, as functions map to unique elements for element of the domain.

Early in the proof, OTTER deduced that if the capacity of insulin in B cells is nearly-exhausted, then it is not completely exhausted:

56 `[neg_hyper,53,2]` $\mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}, 0) \neq \mathit{exhausted}$

Now we skip a part of the proof, which results in information about the relation between the capacity of insulin and the secretion of insulin in B cells at a certain time point:

517 [neg_hyper,516,53] $0 \not> f'(0)$

765 [neg_hyper,761,50,675]
$capacity(b\text{-}cells, insulin, f'(0)) \neq nearly\text{-}exhausted \vee$
$secretion(b\text{-}cells, insulin, f'(0)) = down$

This information allows OTTER to quickly complete the proof, by combining it with the information about the effects of a sulfonylurea drug:

766 [neg_hyper,765,15,56,517]
$capacity(b\text{-}cells, insulin, f(0)) \neq nearly\text{-}exhausted \vee$
$\neg \mathrm{Drug}(\mathrm{SU}, f'(0))$

767 [neg_hyper,765,14,56,517]
$capacity(b\text{-}cells, insulin, f(0)) \neq nearly\text{-}exhausted \vee$
$0 \not> f(0)$

after which (53) can be used as a nucleus to yield:

768 [neg_hyper,767,53] $0 \not> f_1(0)$

and consequently by taking (51) as a nucleus, we find that at time point 0 the capacity of insulin is not nearly exhausted:

769 [neg_hyper,768,51,766]
$capacity(b\text{-}cells, insulin, 0) \neq nearly\text{-}exhausted$

This directly contradicts one of the assumptions and this results in an empty clause:

770 [binary,769.1,53.1] $\bot$

A.2 Proof of Lemma 4.1

Let Γ and Π denote lists of literals. An n-tuple $(x_1, \ldots, x_n) \in \{\mathsf{in}, \mathsf{out}, \mathsf{normal}\}^n$ is called a *mode specification* for an n-place relation symbol $R \in Rel$, denoted by α, β, γ. The set of *input variables* of the atom $R(t_1, \ldots, t_n)$ (where t_i is a term) given a mode specification is defined by:

$$in(R(t_1, \ldots, t_n), (x_1, \ldots, x_n)) = \bigcup \{vars(t_i) \mid 1 \leq i \leq n, x_i = \mathsf{in}\}.$$

Analogously, the set of *output variables* is given by

$$out(R(t_1, \ldots, t_n), (x_1, \ldots, x_n)) = \bigcup \{vars(t_i) \mid 1 \leq i \leq n, x_i = \mathsf{out}\}.$$

An input/output specification is a function S which assigns to every n-place relation symbol R a set $S^+(R/n) \subseteq \{\mathsf{in}, \mathsf{out}, \mathsf{normal}\}^n$ of positive mode specification and a set $S^-(R/n) \subseteq \{\mathsf{in}, \mathsf{normal}\}^n$ of negative mode specifications for R.

Definition A.1 (Definition 2.1 [Stärk, 1994]). *A clause* $\Pi \rightarrow A$ *is called* correct *with respect to an input/output specification* S *or* S-correct *iff*

(C1) *for all positive modes* $\alpha \in S^+(A)$ *there exists a permutation of the literals of the body* Π *of the form* $B_1, \ldots, B_m, \neg C_1, \ldots \neg C_n$ *and for all* $1 \leq i \leq m$ *a positive mode* $\beta_i \in S^+(B_i)$ *such that*

- *for all* $1 \leq i \leq m$, $in(B_i, \beta_i) \subseteq in(A, \alpha) \cup \bigcup_{1 \leq j \leq i} out(B_j, \beta_j)$,
- $out(A, \alpha) \subseteq in(A, \alpha) \cup \bigcup_{1 \leq j \leq m} out(B_j, \beta_j)$,
- *for all* $1 \leq i \leq n$, $S^-(C_i) \neq \varnothing$ *and* $vars(C_i) \subseteq in(A, \alpha) \cup \bigcup_{1 \leq j \leq m} out(B_j, \beta_j)$,

(C2) *for all negative modes* $\alpha \in S^-(A)$ *for all positive literals* B *of* Π *there exists a negative mode* $\beta \in S^-(B)$ *with* $in(B, \beta) \subseteq in(A, \alpha)$ *and for all negative literals* $\neg C$ *of* Π *there exists a positive mode* $\gamma \in S^+(C)$ *with* $in(C, \gamma) \subseteq in(A, \alpha)$.

A program P is called *correct* with respect to an input/output specification S iff all clauses of P are S-correct.

Definition A.2 (Definition 2.2 [Stärk, 1994]). *A goal* Γ *is called* correct *with respect to an input/output specification* S *or* S-correct *iff there exists a permutation* $B_1, \ldots, B_m, \neg C_1, \ldots \neg C_n$ *of the literals of* Γ *and for all* $1 \leq i \leq m$ *a positive mode* $\beta_i \in S^+(B_i)$ *such that*

(G1) *for all* $1 \leq i \leq m$, $in(B_i, \beta_i) \subseteq \bigcup_{1 \leq j \leq i} out(B_j, \beta_j)$,

(G2) *for all* $1 \leq i \leq m$, $S^-(C_i) \neq \varnothing$ *and* $vars(C_i) \subseteq \bigcup_{1 \leq j \leq m} out(B_j, \beta_j)$.

Theorem A.1 (reformulation of Theorem 5.4 [Stärk, 1994]). *Let* P *be a normal program which is correct with respect to the input/output specification* S *and let* $L_1, \ldots, L_r$ *be a goal.*

(a) *If* $\mathrm{COMP}(P) \models L_1 \wedge \ldots \wedge l_r$ *and* $L_1, \ldots, L_r$ *is correct with respect to* S *then there exists a substitution* θ *such that there is a successful SLDNF derivation for* $L_1, \ldots, L_r$ *with answer* θ. *(...)*

Define $S^+ = S^- = \{(\mathsf{normal}\}$ for every unary predicate and $\{(\mathsf{normal}, \mathsf{normal})\}$ for every binary predicate. Observe that Γ contains only definite clauses, so each condition in Definition 1 is trivially satisfied, thus Γ is S-correct. As the goal ψ is grounded, all clauses of Definition 2 are trivially satisfied, thus also S-correct. Hence, by Theorem 3, we find that there is a successful SLDNF derivation of ψ given Γ.

A.3 Proof of Theorem 7.1

A helpful semantic notion is faithfulness, which means that the failure propositions correspond exactly to the failure of the the action it has been introduced for.

Definition A.3. *σ is called* faithful *iff for all $a \in \mathcal{A}$ and all i s.t. $0 \leq i \leq |\sigma|$ holds $\alpha_i(a) = \mathsf{failed}$ iff $\pi_i(f_a) = \top$.*

In the following two lemmas, it is proven that the reduction is found with respect to those faithful models. In the first lemma, we show that Φ acts as $\mathsf{failmodel}$ on the syntactic level, which is then used to prove equivalence of formulas with its reduction.

Lemma A.1. *For all faithful σ and φ:*

$$\mathit{failmodel}(\sigma) \models \varphi \text{ iff } \sigma \models \Phi(\varphi)$$

Proof. By induction on the structure of φ. Suppose $\varphi = a$: ($\Rightarrow$) suppose $\mathsf{failmodel}(\sigma) \models a$ then $\alpha_0(a) \neq \mathsf{failed}$. By faithfulness $\pi_i(f_a) = \bot$, thus $\sigma \models \neg f_a$. All steps can be reversed. The rest of the cases follow almost immediately, taking into account that if the model is faithful, so is every interval within this model, and vice versa. □

Lemma A.2. *For all faithful models σ it holds that $\sigma \models \varphi \leftrightarrow \mathit{reduce}(\varphi)$.*

Proof. By induction on the structure of φ. In this case, the only interested case is for $\varphi = \mathbf{fail}(\psi)$: ($\Rightarrow$) $\sigma \models \mathbf{fail}(\psi)$ iff $\sigma \not\models \psi$ and $\mathsf{failmodel}(\sigma) \not\models \psi$. By I.H. on the first part, it follows that $\sigma \not\models \mathsf{reduce}(\varphi)$. As σ is faithful, it follows that $\mathsf{failmodel}(\sigma)$ is faithful. Therefore $\mathsf{failmodel}(\sigma) \not\models \mathsf{reduce}(\varphi)$. Using Lemma A.1, we get $\sigma \not\models \Phi(\mathsf{reduce}(\varphi))$. Therefore $\sigma \models \neg\mathsf{reduce}(\varphi) \wedge \neg\Phi(\mathsf{reduce}(\varphi))$. By definition, $\sigma \models \mathsf{reduce}(\mathbf{fail}(\varphi))$. All steps are valid in the other direction as well. □

These results do not hold for any model, e.g., it is not for all models the case that $f_a \rightarrow \neg a$. A weak form of faithfulness can be encoded as an ITL formula, bounded by the number of actions in some formula. The fact it is bounded by actions in a formula is relevant, because we may have an infinite number of actions in the language, while each formula has a finite length in standard temporal logic.

Using Definition 7.12, we can then proof the main lemma of this section, which characterises the relation between a formula and its reduction for any model.

Lemma A.3. *$\models \varphi$ iff $\models \mathcal{I}(\varphi) \rightarrow \mathit{reduce}(\varphi)$*

Proof. Without loss of generality, this property can be reformulated as

$$\models \neg\varphi \text{ iff } \mathcal{I}(\varphi) \models \mathsf{reduce}(\neg\varphi)$$

as every formula can be stated as a negation and $\mathcal{I}(\neg\varphi) = \mathcal{I}(\varphi)$. Using the definition of reduce, and taking negation on both sides, rewrite this to:

$$\exists_\sigma \sigma \models \varphi \text{ iff } \exists_\sigma \sigma \models \mathcal{I}(\varphi) \wedge \mathsf{reduce}(\varphi)$$

($\Rightarrow$) Suppose there is some σ such that $\sigma \models \varphi$. Construct a σ' such that $\pi'_i(f_a) = \top$ iff $\alpha_i(a) = \mathsf{failed}$, for all $0 \leq i \leq |\sigma|$, actions a, and all fresh variables f_a introduced in the reduction. Let σ' be the same as σ in every other respect. As φ does not contain any variables f_a, it is clear that then $\sigma' \models \varphi$. As σ' is faithful (by construction), it then follows by Lemma A.2 that $\sigma' \models \mathsf{reduce}(\varphi)$. Moreover, by construction, it follows that $\sigma' \models \mathcal{I}(\varphi)$.
($\Leftarrow$) Suppose for some σ, $\sigma \models \mathcal{I}(\varphi) \wedge \mathsf{reduce}(\varphi)$. Construct σ' such that for all i such that $0 \leq i \leq |\sigma|$, and all actions a:

- if $\pi_i(f_a) = \top$ then $\alpha'_i(a) = \mathsf{failed}$
- if $\pi_i(f_a) = \bot$ and $\alpha_i(a) = \mathsf{active}$ then $\alpha'_i(a) = \mathsf{active}$
- if $\pi_i(f_a) = \bot$ and $\alpha_i(a) \neq \mathsf{active}$ then $\alpha'_i(a) = \mathsf{inactive}$

In all other respects (length, valuation of atomic propositions), σ and σ' are the same. We then prove for all i and $a \in \mathit{actions}(\varphi)$:

$$\alpha_i(a) = \mathsf{active} \Leftrightarrow \alpha'_i(a) = \mathsf{active}$$

($\Rightarrow$) $\alpha_i(a) = \mathsf{active}$. Then, by the fact $\sigma \models \mathcal{I}(\varphi)$, we know that $\pi_i(f_a) = \bot$. Thus, by definition $\alpha'_i(a) = \mathsf{active}$. ($\Leftarrow$) Suppose $\alpha_i(a) \neq \mathsf{active}$. Then either $\alpha'_i(a) = \mathsf{failed}$ (if $\pi_i(f_a) = \top$) or $\alpha'_i(a) = \mathsf{inactive}$ (if $\pi_i(f_a) = \bot$). In any case, we conclude: $\alpha'_i(a) \neq \mathsf{active}$.
As $\mathsf{reduce}(\varphi)$ does not contain a **fail** operator, it cannot distinguish if an action is inactive or failed. Hence, it follows that $\sigma' \models \mathsf{reduce}(\varphi)$. It is easy to see that σ' is faithful, so by Lemma A.2 it follows that $\sigma' \models \varphi$. □

Theorem A.2.

$$\models \varphi \textit{ iff } \mathcal{I}(\varphi) \models_{\mathrm{ITL}} \mathit{reduct}(\varphi)$$

Proof. (sketch). By Lemma A.3, we know $\models \varphi$ iff $\models \mathcal{I}(\varphi) \rightarrow \mathsf{reduce}(\varphi)$. Observe that the right side does not contain the **fail** operator, hence it cannot distinguish between failures and inactivations. Therefore, $\models \mathcal{I}(\varphi) \rightarrow \mathsf{reduce}(\varphi)$ if all actions are interpreted as propositions. By doing this, $\mathcal{I}(\varphi) \rightarrow \mathsf{reduce}(\varphi)$ is also an ITL formula. Finally, note that the semantics of ITL and ITALF coincide for the language of ITL. □

A.4 Proof of Example 1 of Chapter 7

This appendix provides an outline of the proof performed in KIV. The first steps of the proof consists of simple manipulation of the formulas in order to

put them in a comfortable form for presenting the proof. First, recall that the translated failure axiom for diet is:

$$\mathbf{G}\,(\mathit{diet} \rightarrow \mathbf{X!}\,((\mathit{diet} \wedge \mathit{Condition}(\mathit{hyperglycaemia}) \leftrightarrow \mathbf{X!}\,\mathbf{fail}\ \mathit{diet}))$$

Reduction of this to an ITL formula yields:

$$\mathbf{G}\,(\mathit{diet} \rightarrow \mathbf{X!}\,((\mathit{diet} \wedge \mathit{Condition}(\mathit{hyperglycaemia}) \leftrightarrow \mathbf{X!}\,(\neg \mathit{diet} \wedge f_{\mathit{diet}})))$$

which, by the use of Γ, can be written as:

$$\mathbf{G}\,(\mathit{diet} \rightarrow (\mathbf{X!}\,(\mathit{diet} \wedge \mathit{Condition}(\mathit{hyperglycaemia})) \leftrightarrow \mathbf{X!}\,\mathbf{X!}\,f_{\mathit{diet}})) \quad \text{(A.1)}$$

Second, from the background knowledge, we know that:

$$\mathbf{G}\,(\mathit{diet} \wedge \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{normal} \rightarrow \mathbf{X!}\,\mathrm{Condition}(\mathit{normoglycaemia}))$$

which, together with the fact that $\mathbf{G}\ \mathit{capacity}(\mathit{b\text{-}cells}, \mathit{insulin}) = \mathit{normal}$, it can be automatically derived that:

$$\mathbf{G}\,(\mathit{diet} \rightarrow \mathbf{X!}\,\mathrm{Condition}(\mathit{normoglycaemia})) \quad \text{(A.2)}$$

Finally, note that the proof obligation can be presented as

$$\mathbf{X}\,\mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia}) \quad \text{(A.3)}$$

By weakening all the uninteresting parts for proving the property, we finally end up with the main proof obligation:

$$\begin{array}{ll}
\mathbf{G}\,(\mathit{diet} \rightarrow (\mathbf{X!}\,(\mathit{diet} \wedge \mathrm{Condition}(\mathit{hyperglycaemia})) \leftrightarrow \mathbf{X!}\,\mathbf{X!}\,f_{\mathit{diet}})), & \mathit{Eq}.(1) \\
\mathbf{G}\,(\mathit{diet} \rightarrow \mathbf{X!}\,\mathrm{Condition}(\mathit{normoglycaemia})), & \mathit{Eq}.(2) \\
(\mathbf{G}\ \mathit{treatmentdiet} \wedge \neg\,\mathbf{last});\ \mathbf{X!}\,f_{\mathit{diet}}, & \mathcal{M} \\
\mathbf{G}\,(\mathit{treatmentdiet} \rightarrow \mathit{diet}), & \\
\vdash\ \mathbf{X}\,\mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia}) & \mathit{Eq}.(3)
\end{array}$$

Symbolically executing this sequent requires only one possible situation that needs to be proven:

$$\begin{array}{l}
\mathbf{G}\,(\mathit{diet} \rightarrow (\mathbf{X!}\,(\mathit{diet} \wedge \mathrm{Condition}(\mathit{hyperglycaemia})) \leftrightarrow \mathbf{X!}\,\mathbf{X!}\,f_{\mathit{diet}})), \\
\mathbf{G}\,(\mathit{diet} \rightarrow \mathbf{X!}\,\mathrm{Condition}(\mathit{normoglycaemia})), \\
(\mathbf{G}\ \mathit{treatmentdiet});\ \mathbf{X!}\,f_{\mathit{diet}}, \\
\mathbf{G}\,(\mathit{treatmentdiet} \rightarrow \mathit{diet}), \\
\mathrm{Condition}(\mathit{normoglycaemia}), \neg\,\mathbf{X!}\,f_{\mathit{diet}} \\
\vdash\ \mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia})
\end{array}$$

This sequent represents the situation where diet has been applied in the first step. From this it was derived that then the condition is normoglycaemia.

Using this fact, the failure axiom is used to derive that $\neg\,\mathbf{X!}\,f_{diet}$, i.e., diet will not fail in the next step. The rest of the proof consists of the claim that this temporal situation will remain as it is. So we reason by induction that $\mathbf{G}\,\mathrm{Condition}(\mathit{normoglycaemia})$. Abbreviate the sequent above as $\Gamma \vdash \Delta$: then the sequent is rewritten to:

$$\begin{array}{l}\mathbf{G}\,(\mathit{diet} \rightarrow (\mathbf{X!}\,(\mathit{diet} \wedge \mathrm{Condition}(\mathit{hyperglycaemia})) \leftrightarrow \mathbf{X!}\,\mathbf{X!}\,f_{diet})),\\ \mathbf{G}\,(\mathit{diet} \rightarrow \mathbf{X!}\,\mathrm{Condition}(\mathit{normoglycaemia})),\\ (\mathbf{G}\,\mathit{treatmentdiet});\,\mathbf{X!}\,f_{diet},\\ \mathbf{G}\,(\mathit{treatmentdiet} \rightarrow \mathit{diet}),\\ \mathrm{Condition}(\mathit{normoglycaemia}), \neg\,\mathbf{X!}\,f_{diet},\\ t = N, N = N'' + 1\,\mathbf{until}\,\neg\,\mathrm{Condition}(\mathit{normoglycaemia}), \mathbf{IND\text{-}HYP} \vdash\end{array}$$

where $\mathbf{IND\text{-}HYP} \triangleq t < N \rightarrow (\bigwedge \Gamma \rightarrow \bigvee \Delta)$, N a fresh dynamic variable and t a static variable. The remaining steps consists of symbolically executing this sequent, which ends up in the same sequent with $t = N - 1$. Then, the induction hypothesis can be applied, which finishes the proof.

Appendix B

Specifications

B.1 Background Knowledge Diabetes Mellitus type 2

The following specification is in the TPTP (Thousands of Problems for Theorem Provers) [Sutcliffe and Suttner, 1998] syntax. Recent releases of this library also contain scripts to convert the problem to OTTER's input syntax.

```
fof(irreflexivity_gt,axiom,(
    ! [X] : ~ gt(X,X) )).

fof(transitivity_gt,axiom,(
    ! [X,Y,Z] :
      ( ( gt(X,Y)
        & gt(Y,Z) )
     => gt(X,Z) ) )).

fof(xorcapacity1,axiom,(
    ! [X0] :
      ( bcapacityne(X0)
      | bcapacityex(X0)
      | bcapacitysn(X0) ) )).

fof(xorcapacity2,axiom,(
    ! [X0] :
      ( ~ bcapacityne(X0)
      | ~ bcapacityex(X0) ) )).

fof(xorcapacity3,axiom,(
    ! [X0] :
      ( ~ bcapacityne(X0)
```

```
      | ~ bcapacitysn(X0) ) )).

fof(xorcapacity4,axiom,(
    ! [X0] :
      ( ~ bcapacityex(X0)
      | ~ bcapacitysn(X0) ) )).

fof(xorcondition1,axiom,(
    ! [X0] :
      ( conditionhyper(X0)
      | conditionhypo(X0)
      | conditionnormo(X0) ) )).

fof(xorcondition2,axiom,(
    ! [X0] :
      ( ~ conditionhyper(X0)
      | ~ conditionhypo(X0) ) )).

fof(xorcondition3,axiom,(
    ! [X0] :
      ( ~ conditionhyper(X0)
      | ~ conditionnormo(X0) ) )).

fof(xorcondition4,axiom,(
    ! [X0] :
      ( ~ conditionhypo(X0)
      | ~ conditionnormo(X0) ) )).

fof(insulin_effect,axiom,(
    ! [X0] :
      ( ! [X1] :
          ( ~ gt(X0,X1)
         => drugi(X1) )
     => ! [X1] :
          ( ~ gt(X0,X1)
         => ( uptakelg(X1)
            & uptakepg(X1) ) ) ) )).

fof(liver_glucose,axiom,(
    ! [X0,X1] :
      ( ~ gt(X0,X1)
     => ( uptakelg(X1)
       => ~ releaselg(X1) ) ) )).

fof(sulfonylurea_effect,axiom,(
```

```
    ! [X0] :
      ( ( ! [X1] :
            ( ~ gt(X0,X1)
           => drugsu(X1) )
        & ~ bcapacityex(X0) )
     => ! [X1] :
          ( ~ gt(X0,X1)
         => bsecretioni(X1) ) ) ).

fof(biguanide_effect,axiom,(
    ! [X0] :
      ( ! [X1] :
          ( ~ gt(X0,X1)
         => drugbg(X1) )
     => ! [X1] :
          ( ~ gt(X0,X1)
         => ~ releaselg(X1) ) ) ).

fof(sn_cure_1,axiom,(
    ! [X0] :
      ( ( ! [X1] :
            ( ~ gt(X0,X1)
           => bsecretioni(X1) )
        & bcapacitysn(X0)
        & qilt27(X0)
        & ! [X1] :
            ( gt(X0,X1)
           => conditionhyper(X1) ) )
     => ! [X1] :
          ( ~ gt(X0,X1)
         => conditionnormo(X1) ) ) ).

fof(sn_cure_2,axiom,(
    ! [X0] :
      ( ( ! [X1] :
            ( ~ gt(X0,X1)
           => ~ releaselg(X1) )
        & bcapacitysn(X0)
        & ~ qilt27(X0)
        & ! [X1] :
            ( gt(X0,X1)
           => conditionhyper(X1) ) )
     => ! [X1] :
          ( ~ gt(X0,X1)
         => conditionnormo(X1) ) ) ).
```

```
fof(ne_cure,axiom,(
    ! [X0] :
      ( ( ( ! [X1] :
              ( ~ gt(X0,X1)
             => ~ releaselg(X1) )
          | ! [X1] :
              ( ~ gt(X0,X1)
             => uptakepg(X1) ) )
        & bcapacityne(X0)
        & ! [X1] :
            ( ~ gt(X0,X1)
           => bsecretioni(X1) )
        & ! [X1] :
            ( gt(X0,X1)
           => conditionhyper(X1) ) )
     => ! [X1] :
          ( ~ gt(X0,X1)
         => conditionnormo(X1) ) ) )).

fof(ex_cure,axiom,(
    ! [X0] :
      ( ( ! [X1] :
            ( ~ gt(X0,X1)
           => uptakelg(X1) )
        & ! [X1] :
            ( ~ gt(X0,X1)
           => uptakepg(X1) )
        & bcapacityex(X0)
        & ! [X1] :
            ( gt(X0,X1)
           => conditionhyper(X1) ) )
     => ! [X1] :
          ( ~ gt(X0,X1)
         => ( conditionnormo(X1)
            | conditionhypo(X1) ) ) ) )).
```

Some sample conjectures follow. The first is a theorem given the axioms above:

```
% Whether or not patients with subnormal production
% of glucose in the B-cells and a low QI index are
% cured with sulfonylurea.

% Status   : Theorem

fof(treatmentsn2,conjecture,
```

```
    ( ( ! [X0] :
          ( ~ gt(n0,X0)
         => drugsu(X0) )
      & ! [X0] :
          ( gt(n0,X0)
         => conditionhyper(X0) )
      & bcapacitysn(n0)
      & qilt27(n0) )
   => ! [X0] :
        ( ~ gt(n0,X0)
       => conditionnormo(X0) ) )).
```

The second example illustrates a non-theorem:

```
% There is not a suitable therapy for patients
% with exhausted B-cells available.
% Status   : CounterSatisfiable

fof(treatmentex_sub,conjecture,
    ( ( ! [X0] :
          ( ~ gt(n0,X0)
         => ( drugi(X0)
            & drugsu(X0)
            & drugbg(X0) ) )
      & ! [X0] :
          ( gt(n0,X0)
         => conditionhyper(X0) )
      & bcapacityex(n0) )
   => ! [X0] :
        ( ~ gt(n0,X0)
       => conditionnormo(X0) ) )).
```

All other properties can be found in the latest TPTP release or at `http://www.tptp.org`.

B.2 KIV-Asbru Plans

The following is an example of an Asbru plan as used in Chapter 6:

```
asbru('BCT') = mk-asbru-def
 (mk-aasbruc(mk-acond(\lambda pdh, vh, ash, as, ac.
 \not (pdh[ac]['multi-centricity'] .val
  \or
 pdh[ac]['diffuse-maligant-microcalcifications'].val
  \or
 pdh[ac]['previous-breast-radiotherapy'].val)),
```

```
   false,false),
 setup_condition,
 suspend_condition,
 resume_condition,
 abort_condition,
 complete_condition,
 sequential,
 'BCTsurg' + 'MRM' ',
 wait-for-n(1, 'BCTsurg' + 'MRM' '),
false); used for : s ,ls;
```

The filter condition ensures that properties of the patient are fulfilled before the plan is considered. If so, it sequentially executes plans 'BCTsurg' and 'MRM' and completes when one of them has succesfully completed. A more comprehensive explanation of the constructs can be found in [Schmitt et al., 2005].

B.3 SMV Translation of Asbru Plan

The following SMV illustrates the SMV code as generated by the tools and method described in [Bäumler et al., 2006]. Each of the parameters of the plan is defined and for each plan an SMV module implements the state chart semantics of Asbru. For the plan in Appendix B.2, the following is generated:

```
...

BCTp_is_terminated
  :=   (BCTp_state = rejected) | (BCTp_state = aborted)
     | (BCTp_state = completed);
BCTp_is_aborted
  := (BCTp_state = rejected) | (BCTp_state = aborted);
BCTp_consider_condition
  :=   (treatmentp_control_consider_signal = BCT)
     | (treatmentp_control_consider_signal = all);
BCTp_filter_condition
  := ~(multi_centricity=1
   | diffuse_maligant_microcalcifications=1
   | previous_breast_radiotherapy=1);
BCTp_setup_condition := 1;
BCTp_activate_condition
  := ((treatmentp_control_activate_signal = BCT)
      | (treatmentp_control_activate_signal = all));
BCTp_parentterm_condition
  := ( treatmentp_is_terminated
     | treatmentp_state=inactive);
BCTp_reject_condition
```

```
  :=   BCTp_parentterm_condition
     | (BCTp_state = considered
       & !BCTp_filter_condition);
BCTp_abort_multi_condition
  := BCTp_abort_condition
   | BCTp_parentterm_condition
   | BCTp_child_is_aborted;
BCTp_abort_condition := 0;
BCTp_complete_multi_condition
  := (BCTp_complete_condition & BCTp_waitfor_condition);
BCTp_complete_condition := 1;
BCTp_waitfor_condition
  := (BCTsurgp_state=completed) | (MRMp_state=completed);
BCTp_is_in_selection_phase
  := ( BCTp_state = considered) | (BCTp_state = possible)
     | (BCTp_state = ready);
BCTp_retry_condition := 0;
BCTp_child_is_aborted
  := (BCTsurgp_is_aborted) & (MRMp_is_aborted);
BCTp_ifthen_condition := 1;

...

module BCT_plan(BCTp_state,BCTp_parentterm_condition,
BCTp_consider_condition,BCTp_filter_condition,
BCTp_is_terminated,BCTp_is_in_selection_phase,
BCTp_setup_condition,BCTp_reject_condition,
BCTp_activate_condition,BCTp_abort_multi_condition,
BCTp_complete_multi_condition,
BCTp_retry_condition,tick)
{
 init(BCTp_state) := inactive;
 case
 {
   BCTp_parentterm_condition :
       next(BCTp_state) := inactive;
   BCTp_is_terminated
     & BCTp_retry_condition :
       next(BCTp_state) := considered;
   BCTp_state = inactive
     & BCTp_consider_condition :
       next(BCTp_state) := considered;
   BCTp_is_in_selection_phase
     & BCTp_reject_condition :
       next(BCTp_state) := rejected;
```

```
   BCTp_state = considered
     & BCTp_filter_condition :
       next(BCTp_state) := possible;
   BCTp_state = possible
     & BCTp_setup_condition :
       next(BCTp_state) := ready;
   BCTp_state = ready
     & BCTp_activate_condition :
       next(BCTp_state) := activated;
   BCTp_state = activated
     & BCTp_complete_multi_condition :
       next(BCTp_state) := completed;
   BCTp_state = activated
     & BCTp_abort_multi_condition :
       next(BCTp_state) := aborted;
   1 : next(BCTp_state) := BCTp_state;
 }
}
```

Bibliography

[Aaby and Narayana, 1988] Aaby, A. A. and Narayana, K. T. (1988). Propositional temporal interval logic is PSPACE complete. In *9th International Conference on Automated Deduction*, number 310 in LNCS, pages 218–237. Springer.

[Advani et al., 1998] Advani, A., Lo, K., and Shahar, Y. (1998). Intention-based critiquing of guideline-oriented medical care. In *Proceedings of AMIA Annual Symposium*, pages 483–487.

[Ågotnes et al., 2007] Ågotnes, T., van der Hoek, W., Rodríguez-Aguilar, J., Sierra, C., and Wooldridge, M. (2007). On the logic of normative systems. In Veloso, M. M., editor, *Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1175–1180. AAAI Press.

[Allen, 1983] Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

[Alur et al., 1998] Alur, R., Henzinger, T. A., and Kupferman, O. (1998). Alternating-time temporal logic. *LNCS*, 1536:23–60.

[Alves-Foss and Lam, 1999] Alves-Foss, J. and Lam, F. S. (1999). Dynamic denotational semantics of Java. In *Formal Syntax and Semantics of Java*, volume 1523 of *LNCS*. Springer.

[American Cancer Society, 2006] American Cancer Society (2006). Do we know what causes breast cancer? `http://www.cancer.org/docroot/CRI/content/CRI_2_4_2X_Do_we_know_what_causes_breast_cancer_5.asp`. Accessed: August 15, 2007.

[Anderson, 1999] Anderson, R. (1999). The formal verification of a payment system. In Hinchey, M. and Bowen, J., editors, *Industrial-Strength Formal Methods in Practice (Formal Approaches to Computing and Information Technology (FACIT))*, pages 43–52.

[Arden Syntax Technical Committee of HL7, 1999] Arden Syntax Technical Committee of HL7 (1999). *Arden Syntax for Medical Logic Systems*.

[Areces et al., 2000] Areces, C., Gennari, R., Heguiabehere, J., and de Rijke, M. (2000). Tree-based heuristics in modal theorem proving. In *Proceedings of the ECAI'2000*, Berlin, Germany.

[Ashcroft, 1975] Ashcroft, E. A. (1975). Proving assertations about parallel programs. *Journal of Computer System Science*, 10(1):110–135.

[Baeten, 2004] Baeten, J. C. M. (2004). A brief history of process algebra. Technical Report CSR-04-02, Technical University Eindhoven.

[Balser, 2005] Balser, M. (2005). *Verifying Concurrent Systems with Symbolic Execution – Temporal Reasoning is Symbolic Execution with a Little Induction*. PhD thesis, University of Augsburg, Augsburg, Germany.

[Balser et al., 2002a] Balser, M., Duelli, C., and Reif, W. (2002a). Formal semantics of Asbru - an overview. In *Proceedings of the International Conference on Integrated Design and Process Technology*, Passadena. Society for Design and Process Science.

[Balser et al., 2002b] Balser, M., Duelli, C., Reif, W., and Schellhorn, G. (2002b). Verifying concurrent systems with symbolic execution. *Journal of Logic and Computation*, 12(4):549–560.

[Balser et al., 2006] Balser, M., Duelli, C., Reif, W., and Schmitt, J. (2006). Formal semantics of asbru – v2.12. Technical report, University of Augsburg.

[Balser et al., 2000] Balser, M., Reif, W., Schellhorn, G., Stenzel, K., and Thums, A. (2000). Formal system development with KIV. In Maibaum, T., editor, *Fundamental Approaches to Software Engineering*, number 1783 in LNCS. Springer-Verlag.

[Bäumler et al., 2006] Bäumler, S., Balser, M., Dunets, A., Reif, W., and Schmitt, J. (2006). Verification of medical guidelines by model checking – a case study. In Valmari, A., editor, *Proceedings of 13th International SPIN Workshop on Model Checking of Software*, volume 3925 of *LNCS*, pages 219–233. Springer-Verlag.

[Benerecetti et al., 1998] Benerecetti, M., Giunchiglia, F., and Serafini, L. (1998). Model checking multiagent systems. *Journal of Logic and Computation*, 8(3):401–423.

[Bergstra and Klop, 1987] Bergstra, J. A. and Klop, J. W. (1987). A universal axiom system for process specification. In *CWI Quarterly 15*, pages 3–23. CWI.

[Biere et al., 1999] Biere, A., Cimatti, A., Clarke, E. M., Fujita, M., and Zhu, Y. (1999). Symbolic model checking using SAT procedures instead of BDDs. In *Design Automation Conference (DAC'99)*.

[Biere et al., 2003] Biere, A., Cimatti, A., Clarke, E. M., Strichman, O., and Zhu, Y. (2003). *Bounded Model Checking*, volume 58 of *Advances in Computers*. Academic Press.

[Bowen and Hinchey, 1997] Bowen, J. P. and Hinchey, M. G. (1997). The use of industrial-strength formal methods. In *21st International Computer Software & Application Conference (COMPSAC'97)*, pages 332–337. IEEE Computer Society Press.

[Boyd et al., 2005] Boyd, C. M., Darer, J., Boult, C., Fried, L. P. Boult, L., and W., W. A. (2005). Clinical practice guidelines and quality of care for older patients with multiple comorbid diseases. *The Journal of the American Medical Association*, 294(6):716–724.

[Boyer et al., 1995] Boyer, R. S., Kaufmann, M., and Moore, J. S. (1995). The Boyer-Moore theorem prover and its interactive enhancement. *Computers and Mathematics with Applications*, 29(2):27–62.

[Boyer and Moore, 1975] Boyer, R. S. and Moore, J. S. (1975). Proving theorems about LISP functions. *Journal of the Association for Computing Machinery*, 22(1):129–144.

[Bruijne et al., 2007] Bruijne, M. C. d., Zegers, M., Hoonhout, L. H. F., and Wagner, C. (2007). *Onbedoelde schade in Nederlandse ziekenhuizen: dossieronderzoek van ziekenhuisopnames in 2004.* Instituut voor Extramuraal Geneeskundig Onderzoek (NIVEL).

[Bryant, 1986] Bryant, R. E. (1986). Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, 35(8):677–691.

[Bundy, 1988] Bundy, A. (1988). Meta-level inference two applications. *Journal of Automated Reasoning*, 4(1):15–27.

[Burch et al., 1990] Burch, J. R., Clarke, E. M., McMillan, K., L., D. D., and Hwang, L. J. (1990). Symbolic model checking: 10^{20} states and beyond. *Information and Computation*, 98(2):142–170.

[Burrows et al., 1989] Burrows, M., Abadi, M., and Needham, R. (1989). A logic of authentication. In *Proceedings of the Royal Society of London, Series A*, volume 426, pages 233–271.

[Cau and B., 1996] Cau, A. and B., M. (1996). Using PVS for interval temporal logic proofs. part 1: The syntactic and semantic encoding. Technical report, SERCentre, De Montfort University, Leicester.

[CBO, 2002] CBO (2002). *Richtlijn Behandeling van het mammacarcinoom*. van Zuiden.

[CBO, 2005] CBO (2005). *Evidence-based Richtlijnontwikkeling – Handleiding voor werkgroepleden*. CBO. `http://www.cbo.nl/product/richtlijnen/handleiding_ebro/handl_totaal.pdf?`

[Chabrier and Fages, 2003] Chabrier, N. and Fages, F. (2003). Symbolic model checking of biochemical networks. In Priami, C., editor, *Proceedings CMSB*, number 2602 in LNCS, pages 149–162.

[Chaochen, 1993] Chaochen, Z. (1993). Duration calculi: an overview. In Bjørner, D., Broy, M., and Pottosin, I., editors, *Formal Methods in Programming and their Applications*, volume 735 of *LNCS*, pages 256–266, Berlin. Springer.

[Chaudron et al., 1999] Chaudron, M. R. V., Tretmans, J., and Wijbrans, K. (1999). Lessons from the application of formal methods to the design of a storm surge barrier control system. In Wing, J., Woodcock, J., Davies, J., Wing, J. M., Woodcock, J., and Davies, J., editors, *Proceedings World Congress on Formal Methods in the Development of Computing Systems*, volume 1709 of *LNCS*, pages 1511–1526. Springer-Verlag.

[Church, 1936] Church, A. (1936). An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58:345–363.

[Clark, 1978] Clark, K. L. (1978). Negation as failure. In Gaillaire, H. and Minker, J., editors, *Logic and Data Bases*, pages 293–322. Plenum Press, New York.

[Clarke et al., 1986] Clarke, E. M., Emerson, E. A., and Sistla, A. P. (1986). Automatic verification of finite state concurrent systems using temporal logic. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263.

[Clarke et al., 1994] Clarke, E. M., Grumberg, O., and Long, D. E. (1994). Model checking and abstraction. *ACM Transactions on Programming Languages and Systems*, 16(5):1512–1542.

[Clarke et al., 2001] Clarke, E. M., Grumberg, O., and Peled, A. D. (2001). *Model Checking*. The MIT Press, Cambridge, Massachusetts, London, England.

[Clarke et al., 1989] Clarke, E. M., Long, D. E., and McMillan, K. L. (1989). A language for compositional specification and verification of finite state hardware controllers. In Darringer, J. A. and Rammig, F. J., editors, *Proceedings of the 9th International Symposium on Computer Hardware Description Languages and Their Applications*, pages 281–295. North-Holland.

[Clayton and Hripsak, 1995] Clayton, P. and Hripsak, G. (1995). Decision support in healthcare. *International Journal of Biomedical Computing*, 39:59–66.

[Cluzeau et al., 2003] Cluzeau, F. A., Burgers, J. S., M., B., Grol, R., M., M., Littlejohns, P., Grimshaw, J., and Hunt, C. (2003). Development and validation of an international appraisal instrument for assessing the quality of clinical practice guidelines: the AGREE project. *Quality and Safety in Health Care*, 12(1):18–23.

[Cocchiarella, 2002] Cocchiarella, N. (2002). Philosophical perspectives on quantification in tense and modal logic. In Gabbay, D. and Guenther, F., editors, *Handbook of Philosophical Logic*, volume 7, pages 235–275. Kuwer Academic Publishers.

[Cohen and Levesque, 1990] Cohen, P. R. and Levesque, H. J. (1990). Intention is choice with commitment. *Artificial Intelligence*, 42:213–261.

[Console et al., 1991] Console, L., Dupre, D. T., and Torasso, P. (1991). On the relationship between abduction and deduction. *Journal of Logic and Computation*, 1(5):661–690.

[Console and Torasso, 1991] Console, L. and Torasso, P. (1991). A spectrum of logical definitions of model-based diagnosis. *Computational Intelligence*, pages 133–141.

[Dams et al., 1993] Dams, D., Grumberg, O., and Gerth, R. (1993). Generation of reduced models for checking fragments of ctl. In *CAV '93: Proceedings of the 5th International Conference on Computer Aided Verification*, pages 479–490, London, UK. Springer-Verlag.

[Davis et al., 1962] Davis, M., Logemann, G., and Loveland, D. (1962). A machine program for theorem proving. *Communications of the ACM*, 5(7):394–397.

[Davis and Putman, 1969] Davis, M. and Putman, H. (1969). A computing procedure for quantification theory. *Journal of the ACM*, 7:201–215.

[Dijkstra, 1975] Dijkstra, E. W. (1975). Guarded commands, nondeterminacy and formal derivation of programs. *Communications of the ACM*, 18(8):453–457.

[Dixon et al., 1998] Dixon, C., Fisher, M., and Wooldridge, M. (1998). Resolution for temporal logics of knowledge. *Journal of Logic and Computation*, 8(3):345–372.

[Duftschmid and Miksch, 1999] Duftschmid, G. and Miksch, S. (1999). Knowledge-based verification of clinical guidelines by detection of anomalies. *OEGAI Journal*, pages 37–39.

[Duftschmid et al., 2002] Duftschmid, G., Miksch, S., and Gall, W. (2002). Verification of temporal scheduling constraints in clinical practice guidelines. *Artificial Intelligence in Medicine*, 25:93–121.

[Duftschmid et al., 1998] Duftschmid, G., Miksch, S., Shahar, Y., and Johnson, P. (1998). Multi-level verification of clinical protocols. In *Proceedings of the Workshop on Validation and Verification of Knowledge-Based Systems*, Trento, Italy.

[Dung, 1995] Dung, P. M. (1995). An argumentation theoretic foundation of logic programming. *Journal of Logic Programming*, 22:151–177.

[Eker et al., 2002] Eker, S., Knapp, M., Laderoute, K., Lincoln, P., Meseger, J., and Sönmez, K. M. (2002). Pathway logic: Symbolic analysis of biological signaling. In *Proceedings of the seventh Pacific Symposium on Biocomputing*, pages 400–412.

[Emerson and Halpern, 1982] Emerson, E. A. and Halpern, J. Y. (1982). Decision procedures and expressiveness in the temporal logic of branching time. In *STOC '82: Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 169–180, New York, NY, USA. ACM Press.

[Errampalli et al., 2004] Errampalli, D. D., Priami, C., and Quaglia, P. (2004). A formal language for computational systems biology. *OMICS A Journal Of Integrative Biology*, 8(4):371–380.

[Fabrega et al., 1998] Fabrega, F. J. T., Herzog, J. C., and D., G. J. (1998). Why is a security protocol correct? In *IEEE Symposium on Security and Privacy*.

[Fagin et al., 1995] Fagin, R., Halpern, J. Y., Moses, Y., and Vardi, M. Y. (1995). *Reasoning about Knowledge*. MIT Press, Cambridge.

[Fensel et al., 1996] Fensel, D., Schonegge, A., Groenboom, R., and B., W. (1996). Specification and verification of knowledge-based systems. In Gaines, B. R. and Musen, M. A., editors, *Proceedings of the 10th Banff knowledge acquisition for knowledge-based systems workshop*, pages 1–20, Department of Computer Science, University of Calgary.

[Fidge et al., 1998] Fidge, C. J., Hayes, I. J., Martin, A. P., and Wabenhorst, A. K. (1998). A set-theoretic model for real-time specification and reasoning. In Jeuring, J., editor, *Mathematics of Program Construction (MPC'98)*, volume 1422 of *LNCS*, pages 188–206, Berlin. Springer.

[Fisher et al., 2001] Fisher, M., Dixon, C., and Peim, M. (2001). Clausal temporal resolution. *ACM Transactions on Computational Logic*, 2(1):12–56.

[Floyd, 1967] Floyd, R. W. (1967). Assigning meaning to programs. In *Proceedings of Symposia in Applied Mathematics*, pages 19–32, Providence, RI. American Mathematical Society.

[Fox and Das, 2000] Fox, J. and Das, S. (2000). *Safe and Sound: Artificial Intelligence in Hazardous Applications*. AAAI Press.

[Fox et al., 2008] Fox, J., Dunlop, R., Black, E., Chronakis, I., Patkar, V., South, M., and Thomson, R. (2008). *Computer-based Medical Guidelines and Protocols: a Primer and Current Trends*, chapter From Guidelines to Careflow: analysis and discussion. IO Press.

[Fox et al., 1996] Fox, J., Johns, N., Rahmanzadeh, A., and Thomson, R. (1996). PRO*forma*: A method and language for specifying clinincal guidelines and protocols. In Brender, J., Christensen, J., Scherrer, J. R., and McNair, P., editors, *Medical Informatics Europe*, pages 516–520. IOS Press.

[Fox et al., 1997] Fox, J., Johns, N., Rahmanzadeh, A., and Thomson, R. (1997). PRO*forma*: a general technology for clinical decision support systems. *Computer Methods and Programs in Biomedicine*, 54:59–67.

[Gabbay, 1987] Gabbay, D. M. (1987). The declarative past and imperative future: executable temporal logic for interactive systems. In Barringer, H., editor, *Temporal Logic in Specification*, volume 398 of *LNCS*, pages 409–448, London, UK. Springer-Verlag.

[Gabbay, 1989] Gabbay, D. M. (1989). The declarative past and imperative future: Executable temporal logic for interactive systems. In Barringer, H., editor, *Temporal Logic in Specification*, volume 398 of *LNCS*, pages 409–448. Springer-Verlag, Berlin.

[Gabbay et al., 1994] Gabbay, D. M., Hodkinson, I., and Reynolds, M. (1994). *Temporal Logic: Mathematical Foundations and Computational Aspects*. Oxford Logic Guides. Oxford University Press.

[Gabbay and Woods, 2004] Gabbay, D. M. and Woods, J. (2004). *The Rise of Modern Logic: from Leibniz to Frege*. Handbook of the History of Logic. Elsevier.

[Ganong, 2005] Ganong, W. F. (2005). *Review of Medical Physiology*. McGraw-Hill, 22nd edition.

[Gertner, 1995] Gertner, A. S. (1995). *Critiquing: effective decision support in time-critical domains*. PhD thesis, Deptartment of Computer & Information Science, University of Pennsylvania.

[Gill, 1962] Gill, A. (1962). *Introduction to the Theory of Finite-State Machines*. McGraw-Hill.

[Giunchiglia et al., 1994] Giunchiglia, F., Spalazzi, L., and Traverso, P. (1994). Planning with failure. In *Artificial Intelligence Planning Systems*, pages 74–79.

[Goldblatt, 1987] Goldblatt, R. (1987). Logics of time and computation. CSLI lecture notes, Center for the Study of Langugage and Information.

[Goldstine and von Neumann, 1947] Goldstine, H. H. and von Neumann, J. (1947). Planning and coding of problems for an electronic computing instrument. In Taub, A., editor, *Collected Works of J. von Neumann*, volume 5, pages 80–151. Pergamon.

[Good, 1970] Good, D. I. (1970). *Toward a Man-Machine System for Proving Program Correctness*. PhD thesis, University of Wisconsin, Madison, WI.

[Graham et al., 2003] Graham, I., Harrison, M., and Brouwers, M. (2003). Evaluating and adapting practice guidelines for local use: a conceptual framework. In Pickering, S. and Tomlinson, J., editors, *Clinical Governance and Best Value: Meeting the Modernisation Agenda*, pages 213–229. Churchill Livingstone, London.

[Graham and Harrison, 2005] Graham, I. D. and Harrison, M. B. (2005). Evaluation and adaptation of clinical practice guidelines. *Evidence-based nursing*, 8:68–72.

[Graham et al., 2002] Graham, I. D., Harrison, M. B., Brouwers, M., Davies, B. L., and Dunn, S. (2002). Facilitating the use of evidence in practice: evaluating and adapting clinical practice guidelines for local use by health care organizations. *Journal of obstetric, gynecologic, and neonatal nursing*, 31:599–611.

[Graham et al., 2005] Graham, I. D., Harrison, M. B., Lorimer, K., Piercianowski, T., Friedberg, E., Buchanan, M., and Harris, C. (2005). Adapting national and international leg ulcer practice guidelines for local use: the ontario leg ulcer community care protocol. *Advances in skin & wound care*, 18:307–318.

[Green et al., 2002] Green, F. L., Page, D. L., Fleming, I. D., Fritz, A., Balch, M. C., Haller, D. G., and Morrow, M. (2002). *AJCC Cancer Staging Manual*. Springer-Verlag, New York.

[Groot et al., 2008] Groot, P. C., Hommersom, A. J., and Lucas, P. J. F. (2008). Adaptation and refinement: From guidelines to protocols. In *Computerized Guidelines and Protocols*. IOS Press. To appear.

[Groot et al., 2007] Groot, P. C., Hommersom, A. J., Lucas, P. J. F., Serban, R., ten Teije, A., and van Harmelen, F. (2007). The role of model checking in critiquing based on clinical guidelines. In Bellazzi, R., Abu-Hanna, A., and

Hunter, J., editors, *11th Conference on Artificial Intelligence in Medicine*, number 4595 in LNAI, pages 411–420. Springer-Verlag Berlin Heidelberg.

[Grumberg and Long, 1994] Grumberg, O. and Long, D. E. (1994). Model checking and modular verification. *ACM Transactions on Programming Languages and Systems*, 16(3):843–871.

[Guyton and Hall, 2000] Guyton, A. C. and Hall, J. E. (2000). *Textbook of Medical Physiology*. W.B. Saunders Company.

[Hilbert and Ackermann, 1928] Hilbert, D. and Ackermann, W. (1928). *Grundzüge der Theoretischen Logik*. Verlag von Julius Springer, Berlin.

[Hindriks et al., 1998] Hindriks, K., Boer, F. d., van der Hoek, W., and Meyer, J.-J. C. (1998). Failure, monitoring and recovery in the agent language 3APL. In Giacomo, G. D., editor, *AAAI 1998 Fall Symposium on Cognitive Robotics*, pages 68–75.

[Hoare, 1969] Hoare, C. A. R. (1969). An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580.

[Hoey and Todkill, 2001] Hoey, J. and Todkill, A. M. (2001). No mere theory: Olli Miettinen's "The modern scientific physician". *Canadian Medical Association Journal*, 165(4):439–440.

[Hofestädt and Thelen, 1998] Hofestädt, R. and Thelen, S. (1998). Quantitative modeling of biochemical networks. *In Silico Biology*, 1(1):39–53.

[Hommersom et al., 2006] Hommersom, A. J., Groot, P. C., Lucas, P., Marcos, M., and Martinez-Salvador, B. (2006). A constraint-based approach to medical guidelines and protocols. In *ECAI 2006 WS – AI techniques in healthcare: evidence based guidelines and protocols*.

[Hommersom et al., 2007] Hommersom, A. J., Groot, P. C., Lucas, P. J. F., Balser, M., and Schmitt, J. (2007). Verification of medical guidelines using background knowledge in task networks. *IEEE Transactions on Knowledge and Data Engineering*, 19(6):832–846.

[Hommersom and Lucas, 2007] Hommersom, A. J. and Lucas, P. J. F. (2007). Actions with failures in interval temporal logic. In *Proceedings of the Eight Workshop on Computational Logic in Multi-Agent Systems (CLIMA VIII)*.

[Hommersom et al., 2004a] Hommersom, A. J., Lucas, P. J. F., and Balser, M. (2004a). Meta-level verification of the quality of medical guidelines using interactive theorem proving. In *Logics in Artificial Intelligence: 9th European Conference*, volume 3229 of *LNCS*, pages 654–666, Lisbon, Portugal. Springer-Verlag.

[Hommersom et al., 2005a] Hommersom, A. J., Lucas, P. J. F., and van Bommel, P. (2005a). Automated theorem proving for quality-checking medical guidelines. In *Empirically Successful Classical Automated Reasoning (ESCAR)*. CADE.

[Hommersom et al., 2005b] Hommersom, A. J., Lucas, P. J. F., van Bommel, P., and van der Weide, T. P. (2005b). A history-based algebra for quality-checking medical guidelines. In Miksch, S., Hunter, J., and Karavnou, E., editors, *Artificial Intelligence in Medicine*, volume 3581 of *LNAI*, pages 163–168. Springer-Verlag.

[Hommersom et al., 2004b] Hommersom, A. J., Meyer, J. J., and de Vink, E. P. (2004b). Update semantics of security protocols. *Synthese*, 142:229–267. Knowledge, Rationality and Action subseries.

[Hripcsak et al., 1994] Hripcsak, G., Ludemann, P., Pryor, T. A., Wigertz, O. B., and Clayton, P. D. (1994). Rationale for the Arden Syntax. *Comput Biomed Res*, 7:291–324.

[Jech, 1995] Jech, T. (1995). OTTER experiments in a system of combinatory logic. *Journal of Automated Reasoning*, 14(3):413–426.

[Kamp, 1968] Kamp, J. A. W. (1968). *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles.

[Keeney and Raiffa, 1976] Keeney, R. L. and Raiffa, H. (1976). *Decisions with Multipe Objectives: Preferences and Value Tradeoffs*. John Wiley & Sons.

[Kersting and De Raedt, 2000] Kersting, K. and De Raedt, L. (2000). Bayesian logic programs. In Cussens, J. and Frisch, A., editors, *Proceedings of the Work-in-Progress Track at the 10th International Conference on Inductive Logic Programming*, pages 138–155.

[King, 1969] King, J. C. (1969). *A Program Verifier*. PhD thesis, Carnegie-Mellon University, Pittburgh, PA.

[Knapp et al., 2002] Knapp, A., Merz, S., and Rauh, C. (2002). Model checking timed uml state machines and collaborations. In *7th International Symposium on Formal Techniques in Real-Time and Fault Tolerant Systems*, volume 2469 of *LNCS*, pages 395–414. Springer-Verlag.

[Knuth and Bendix, 1970] Knuth, D. E. and Bendix, P. B. (1970). Simple word problems in universal algebras. In Leech, J., editor, *Computational Algebra*, pages 263–297. Pergamon Press.

[Konikowska, 1998] Konikowska, B. (1998). A three-valued linear temporal logic for reasoning about concurrency. Technical report, ICS PAS, Warsaw.

[Kosara and Miksch, 2001] Kosara, R. and Miksch, S. (2001). Metaphors of movement: a visualization and user interface for time-oriented, skeletal plans. *Artificial Intelligence in Medicine*, 22(2):111–131.

[Kraus and Lehmann, 1988] Kraus, S. and Lehmann, D. (1988). Knowledge, belief and time. *Theoretical Compututer Science*, 58(1–3):155–174.

[Kuhn and Portner, 2002] Kuhn, S. T. and Portner, P. (2002). Tense and time. In Gabbay, D. M. and Guenthner, F., editors, *Handbook of philosophical logic*, volume 7, pages 277–346. Kluwer, second edition.

[Kupferman and Vardi, 1998] Kupferman, O. and Vardi, M. Y. (1998). Modular model checking. *LNCS*, 1536:381–401.

[Kuttler, 2006] Kuttler, C. (2006). Simulating bacterial transcription and translтion in a stochastic π-calculus. In Priami, C. and Plotkin, G., editors, *Transactions on Computational Systems Biology VI*, number 4220 in LNBI, pages 113–149.

[Laroussinie and Schnoebelen, 2000] Laroussinie, F. and Schnoebelen, Ph. (2000). Specification in CTL+past for verification in CTL. *Information and Computation*, 156(1-2):236–263.

[Lehmann, 1998] Lehmann, E. (1998). Compartmental models for glycaemic prediction and decision-support in clinical diabetes care: promise and reality. *Computer Methods and Programs in Biomedicine*, 56(2):193–204.

[Lucas, 1993] Lucas, P. J. F. (1993). The representation of medical reasoning models in resolution-based theorem provers. *Artificial Intelligence in Medicine*, 5:395–419.

[Lucas, 1995] Lucas, P. J. F. (1995). Logic engineering in medicine. *The Knowledge Engineering Review*, 10(2):153–179.

[Lucas, 1997] Lucas, P. J. F. (1997). Symbolic diagnosis and its formalisation. *The Knowledge Engineering Review*, 12(2):109–146.

[Lucas, 2003] Lucas, P. J. F. (2003). Quality checking of medical guidelines through logical abduction. In Coenen, F., Preece, A., and Mackintosh, A. L., editors, *Proceedings of AI-2003, the 23rd SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, volume XX, pages 309–321, London. Springer.

[Lucas et al., 1998] Lucas, P. J. F., Boot, H., and Taal, B. G. (1998). Decision-theoretic network approach to treatment management and prognosis. *Knowledge-based Systems*, 11:321–330.

[Lucas et al., 2005] Lucas, P. J. F., Hommersom, A. J., Galan, J. C., Marcos, M., Coltell, O., Rosenbrand, K., Wittenberg, J., and van Croonenborg, J. (2005). Protocure deliverable 1.3: New model of guideline development process. `http://www.keg.uji.es/deliverables/D13-new-model.pdf.gz`, Accessed: October 1, 2007.

[Lucas and van der Gaag, 1991] Lucas, P. J. F. and van der Gaag, L. C. (1991). *Principles of Expert Systems*. Addison-Wesley, Wokingham.

[Lucas and van der Gaag, 2005] Lucas, P. J. F. and van der Gaag, L. C. (2005). Principles of intelligent systems. `http://www.cs.ru.nl/~peterl/teaching/IS/boek.html`.

[Magni et al., 2000] Magni, P., Bellazzi, R., Sparacino, G., and Cobelli, C. (2000). Bayesian identification of a population compartmental model of c-peptide kinetics. *Annals of Biomedical Engineering*, 28:812–823.

[Mani and Aliferis, 2007] Mani, S. and Aliferis, C. (2007). A causal modeling framework for generating clinical practice guidelines from data. In Bellazzi, R., Abu-Hanna, A., and Hunter, J., editors, *Artificial Intelligence in Medicine*, LNAI.

[Manna and Pnueli, 1994] Manna, Z. and Pnueli, A. (1994). Temporal verification diagrams. In *Theoretical Aspects of Computer Software*, pages 726–765.

[Marcos et al., 2002] Marcos, M., Balser, M., ten Teije, A., and van Harmelen, F. (2002). From informal knowledge to formal logic: a realistic case study in medical protocols. In *Proceedings of the 12th EKAW-2002*.

[Marcos et al., 2001] Marcos, M., Berger, G., van Harmelen, F., ten Teije, A., Roomans, H., and Miksch, S. (2001). Using critiquing for improving medical protocols: Harder than it seems. In *8th European Conference on Artificial Intelligence in Medicine*, pages 431–441.

[Marcos et al., 2006] Marcos, M., Martínez-Salvador, B., Hommersom, A. J., Groot, P. C., Lucas, P. J. F., Jovell, A., and Blancafort, S. (2006). Case-study in transformations for protocol development from guidelines. Technical Report D5.1, Protocure II.

[Markey, 2003] Markey, N. (2003). Temporal logic with past is exponentially more succinct. *EATCS Bulletin*, 79:122–128.

[McCarthy, 1962] McCarthy, J. (1962). Towards a mathematical science of computation. In Popplewell, C. M., editor, *Proceedings of IFIP Congress*, pages 21–28. North-Holland.

[McCarthy, 1963] McCarthy, J. (1963). A basis for a mathematical theory of computation. In Braffort, P. and Hirschberg, D., editors, *Computer Programming and Formal Systems*, pages 33–70. North-Holland.

[McCune, 2007] McCune, N. (2007). Prover9. http://www.cs.unm.edu/~mccune/prover9. Accessed: August 15, 2007.

[McCune, 1997] McCune, W. (1997). Solution of the Robbins problem. *Journal of Automated Reasoning*, 19(3):263–276.

[McCune, 2001] McCune, W. (2001). MACE 2.0 Reference Manual and Guide. Technical Memo ANL/MCS-TM-249, Argonne National Laboratory, Argonne, IL.

[McCune, 2003] McCune, W. (2003). Otter 3.3 Reference Manual. Technical Memo ANL/MCS-TM-263, Argonne National Laboratory, Argonne, IL.

[McMillan, 1993] McMillan, K. L. (1993). *Symbolic Model Checking*. Kluwer Academic Publishers.

[Miksch, 1999] Miksch, S. (1999). Plan management in the medical domain. *AI Communications*, 12(4):209–235.

[Miller et al., 1999] Miller, D. W., Frawley, S. J., and Miller, P. L. (1999). Using semantic constraints to help verify the completeness of a computer-based clinical guideline for childhood immunization. *Computer Methods and Programs in Biomedicine*, 58(3):267–280.

[Miller, 1984] Miller, P. (1984). *A Critiquing Approach to Expert Computer Advice: ATTENDING*. Pittman Press, London.

[Milner, 1980] Milner, R. (1980). *A Calculus of Communicating Systems*. Springer-Verlag.

[Milner, 1999] Milner, R. (1999). *Communicating and Mobile Systems: the Pi-Calculus*. Cambridge University Press.

[Minsky, 1975] Minsky, M. (1975). A framework for representing knowledge. In Winston, P. H., editor, *The Psychology of Computer Vision*. McGraw-Hill, New York (U.S.A.).

[Moore, 1979] Moore, R. C. (1979). *Reasoning about Knowledge and Action*. PhD thesis, MIT.

[Morris and Jones, 1984] Morris, F. L. and Jones, C. B. (1984). An early program proof by Alan Turing. *Annals of the History of Computing*, 6(2):139–143.

[Moszkowski, 1996] Moszkowski, B. (1996). The programming language Tempura. *Journal of Symbolic Computation*, 22(5/6):730–733.

[Mulyar et al., 2007] Mulyar, N., van der Aalst, W., and Peleg, M. (2007). A pattern-based analysis of clinical computer-interpretable guideline modeling languages. *Journal of the American Medical Informatics Association*, 14:781–787.

[NHG, 2003] NHG (2003). http://web.archive.org/web/20031018093525/http://nhg.artsennet.nl/standaarden/M01/start.htm. Accessed: August 31, 2007.

[Oheimb and Nipkow, 1999] Oheimb, D. v. and Nipkow, T. (1999). Machine-checking the Java specification: Proving type-safety. In *Formal Syntax and Semantics of Java*, volume 1523 of *LNCS*. Springer.

[Ohlbach, 1988] Ohlbach, H. J. (1988). A Resolution Calculus for Modal Logics. In Lusk, E. and Overbeek, R., editors, *Proceedings CADE-88: International Conference on Auomated Deduction*, volume 310 of *LNCS*, pages 500–516. Springer-Verlag.

[Ollenschläger et al., 2004] Ollenschläger, G., Marshall, C., Qureshi, S., Rosenbrand, K., Burgers, J., Mäkelä, M., and Slutsky, J. (2004). Improving the quality of health care: using international collaboration to inform guideline programmes by founding the Guidelines International Network (G-I-N)*. *Quality and Safety in Health Care*, 13:455–460.

[Owicki and Gries, 1976] Owicki, S. S. and Gries, D. (1976). An axiomatic proof technique for prallel programs. *Acta Informatica*, 6:319–340.

[Owre et al., 1992] Owre, S., Rushby, J. M., , and Shankar, N. (1992). PVS: A prototype verification system. In Kapur, D., editor, *11th International Conference on Automated Deduction (CADE)*, volume 607 of *LNAI*, pages 748–752, Saratoga, NY. Springer-Verlag.

[Patil, 1981] Patil, R. S. (1981). Causal representation of patient illness for ELECTROLYTE and ACID-BASE diagnosis. Technical Report MIT/LCS/TR-267, MIT.

[Paulson, 1989] Paulson, L. C. (1989). The foundation of a generic theorem prover. *Journal of Automated Reasoning*, 5(3):363–397.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

[Peleg et al., 2000] Peleg, M., Boxwala, A., Ogunyemi, O., Zeng, P., Tu, S., Lacson, R., Begnstam, E., and Ash, N. (2000). GLIF3: The evolution of a guideline representation format. In *Proceedings of American Medical Informatics Association Symposium*, pages 645–649.

[Peleg et al., 2003] Peleg, M., Tu, S., Bury, J., Ciccarese, P., and Fox, J. (2003). Comparing computer-interpretable guideline models: a case-study approach. *Journal of the American Medical Informatics Association*, 10(1):52–68.

[Pelletier et al., 2002] Pelletier, F. J., Sutcliffe, G., and Suttner, C. B. (2002). The development of CASC. *AI Communications*, 15(2-3):79–90.

[Phillips and Vojtěchovskiý, 2005] Phillips, J. D. and Vojtěchovskiý, P. (2005). Linear groupoids and the associated wreath products. *Journal of Symbolic Computation*, 40(3):1106–1125.

[Pnueli, 1977] Pnueli, A. (1977). The temporal logic of programs. In *Proceedings of the 18th IEEE Symposium on Foundation of Computer Science*, pages 46–57.

[Pnuelli, 1981] Pnuelli, A. (1981). A temporal logic of concurrent programs. *Theoretical Computer Science*, 13:45–60.

[Poole, 1990] Poole, D. (1990). A Methodology for using a Default and Abductive Reasoning System. *International Journal of Intelligent System*, 5(5):521–548.

[Poole, 1993] Poole, D. (1993). Probabilistic Horn abduction and Bayesian networks. *Artificial Intelligence*, 64(1):81–129.

[Prakken and Vreeswijk, 2002] Prakken, H. and Vreeswijk, G. (2002). *Logics for defeasible argumentation*, volume 4. Kluwer Academic Publishers, Dordrecht.

[Pratt, 1976] Pratt, V. R. (1976). Semantical considerations on Floyd-Hoare logic. In *Proceedings of the 17th Annual IEEE Symposium on Foundations of Computer Science*, pages 109–121.

[Prior, 1957] Prior, A. N. (1957). *Time and Modality*. Clarendon Press.

[Pronk et al., 2007] Pronk, T. E., de Vink, E. P., Bosnacki, D., and Breit, T. M. (2007). Stochastic modeling of codon bias with PRISM. In Linden, I. and Talcott, C., editors, *Proceedings of MTCoord 2007*. Computer Science Department, University of Cyprus, Nicosia.

[Queille and Sifakis, 1982] Queille, J. and Sifakis, J. (1982). Specification and verification of concurrent systems is CESAR. In *International Symposium on Programming, LNCS 137*, pages 337–351. Springer Verlag.

[Rao and Georgeff, 1991] Rao, A. S. and Georgeff, M. P. (1991). Modeling rational agents within a BDI-architecture. In Allen, J., Fikes, R., and Sandewall, E., editors, *Proceedings of KR'91*, pages 473–484. Morgan Kaufmann.

[Reif, 1995] Reif, W. (1995). The KIV Approach to Software Verification. In Broy, M. and Jähnichen, S., editors, *KORSO: Methods, Languages, and Tools for the Construction of Correct Software*, volume 1009 of *LNCS*. Springer-Verlag, Berlin.

[Reynolds, 2005] Reynolds, M. (2005). An axiomatization of pctl. *Information and Computation*, 201(1):72–119.

[Riazanov and Voronkov, 2002] Riazanov, A. and Voronkov, A. (2002). The design and implementation of VAMPIRE. *AI Communications*, 15(2):91–110.

[Richardson and Domingos, 2006] Richardson, M. and Domingos, P. (2006). Markov logic networks. *Machine Learning*, 62(1-2):107–136.

[Robinson, 1965a] Robinson, J. A. (1965a). Automated deduction with hyper-resolution. *International Journal of Computatational Mathematics*, 1:23–41.

[Robinson, 1965b] Robinson, J. A. (1965b). A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41.

[Roses, 2005] Roses, D. F. (2005). *Breast Cancer*. Elsevier, Philadelphia, PA, 2nd edition.

[Schellhorn and Ahrendt, 1997] Schellhorn, G. and Ahrendt, W. (1997). Reasoning about abstract state machines: The wam case study. *Journal of Universal Computer Science*, 3(4):377–413. available at http://hyperg.iicm.tugraz.ac.at/jucs/.

[Schellhorn et al., 2006] Schellhorn, G., Grandy, H., Haneberg, D., and Reif, W. (2006). The mondex challenge: Machine checked proofs for an electronic purse. In Misra, J., Nipkow, T., and Sekerinski, E., editors, *Formal Methods 2006, Proceedings*, volume 4085 of *LNCS*, pages 16–31. Springer.

[Schmidt and Hustadt, 2003] Schmidt, R. A. and Hustadt, U. (2003). Mechanised reasoning and model generation for extended modal logics. In de Swart, H., Orlowska, E., Schmidt, G., and Roubens, M., editors, *Theory and Applications of Relational Structures as Knowledge Instrument*, volume 2929 of *LNCS*, pages 38–67. Springer.

[Schmitt et al., 2005] Schmitt, J., Balser, M., and Reif, W. (2005). Complementary material to deliverable d4.2b: Improved verification system (prototype). In *Protocure II - Integrating formal methods in the development process of medical guidelines and protocols*. Protocure II.

[Schmitt et al., 2006a] Schmitt, J., Balser, M., and Reif, W. (2006a). Support for interactive verification of asbru in kiv. Technical Report 2006-16, Universität Augsburg, Institut für Informatik.

[Schmitt et al., 2006b] Schmitt, J., Reif, W., Seyfang, A., and Miksch, S. (2006b). Temporal dimension of medical guidelines: The semantics of asbru time annotations. In *ECAI 2006 WS – AI techniques in healthcare: evidence-baded guidelines and protocols*.

[Schulz, 2002] Schulz, S. (2002). E - a brainiac theorem prover. *AI Communications*, 15(2/3):111–126.

[Serban et al., 2006] Serban, R., ten Teije, A., van Harmelen, F., Hommersom, A., Lucas, P. J. F., Jovell, A., Blancafort, S., Wittenberg, J., Rosenbrand, K., and van Croonenborg, J. (2006). How to use model-checking for critiquing using medical guidelines. Technical report, Protocure. http://www.keg.uji.es/deliverables/D52-how-to-use-model-checking.pdf.

[Serban et al., 2004] Serban, R., ten Teije, A., van Harmelen, F., Marcos, M., Polo, C., Galan, J. C., Hommersom, A. J., Lucas, P. J. F., Rosenbrand, K., Wittenberg, J., and van Croonenborg, J. (2004). Library of design patterns for guidelines. http://www.keg.uji.es/deliverables/D25-library-of-design-patterns.pdf.

[Seyfang et al., 2002] Seyfang, A., Kosara, R., and Miksch, S. (2002). Asbru's reference manual, asbru version 7.3. Technical Report Asgaard-TR-20002-1, Vienna University of Technology, Institute of Software Technology.

[Seyfang et al., 2006] Seyfang, A., Miksch, S., Marcos, M., Wittenberg, J., Polo-Conde, C., and Rosenbrand, K. (2006). Bridging the grap between informal and formal guideline representations. In Brewka, K., Coradeschi, S., Perini, A., and Traverso, P., editors, *17th European Conference on Artificial Intelligence*, volume 141, pages 447–451. IOS Press.

[Shahar, 1997] Shahar, Y. (1997). A framework for knowledge-based temporal abstraction. *Artificial Intelligence*, 90(1-2):79–133.

[Shahar and Cheng, 2000] Shahar, Y. and Cheng, C. (2000). Model-based visualization of temporal abstractions. *Computational Intelligence*, 16(2):279–306.

[Shahar et al., 1997] Shahar, Y., Miksch, S., and Johnson, P. (1997). A task-specific ontology for the application and critiquing of time oriented clinical guidelines. In *Proceedings of the sixth Conference on Artificial Intelligence in Medicine in Europe*, pages 51–61.

[Shahar et al., 1998] Shahar, Y., Miksch, S., and Johnson, P. (1998). The Asgaard project: A task-specific framework for the application and critiquing of time-orientied clinical guidelines. *Artificial Intelligence in Medicine*, 14:29–51.

[Shepherdson, 1987] Shepherdson, J. C. (1987). Negation in logic programming. In Minker, J., editor, *Deductive Databases and Logic Programming*, pages 19–88. Morgan Kaufmann Publishers.

[Shiffman, 1997] Shiffman, R. N. (1997). Representation of clinical practice guidelines in conventional and augmented decision tables. *Journal of the American Medical Informatics Association*, 4:382–393.

[Shiffman and Greenes, 1994] Shiffman, R. N. and Greenes, R. A. (1994). Improving clinical guidelines with logic and decision-table techniques: application in hepatitis immunization recommendations. *Medical Decision Making*, 14:245–254.

[Shortliffe, 1976] Shortliffe, E. (1976). *Computer-based Medical Consultations*. Elsevier.

[Shortliffe, 1974] Shortliffe, E. H. (1974). *MYCIN: a rule-based computer program for advising physicians regarding antimicrobial therapy selection.* PhD thesis, Stanford University.

[SIGN, 1998] SIGN (1998). *Breast Cancer in Women*. SIGN.

[Silverman, 1992] Silverman, B. G. (1992). Survey of expert critiquing systems: Practical and theoretical frontiers. *Communications of the ACM*, 35(4):106–127.

[Sips et al., 2006] Sips, R., Braun, L., and Roos, N. (2006). Applying formal medical guidelines for critiquing. In *ECAI 2006 WS – AI techniques in healthcare: evidence based guidelines and protocols*.

[Sistla and Clarke, 1985] Sistla, A. P. and Clarke, E. M. (1985). The complexity of propositional linear temporal logics. *Journal of the Association for Computing Machinery*, 32(3):733–749.

[Slind, 1991] Slind, K. (1991). An implementation of higher order logic. Technical Report 91–419–03, Computer Science Department, University of Calgary.

[Somenzi and Bloem, 2000] Somenzi, F. and Bloem, R. (2000). Efficient büchi automata from ltl formulae. In Emerson, E. A. and Sistla, A. P., editors, *Twelfth Conference on Computer Aided Verification (CAV'00)*, pages 248–263. Springer-Verlag. LNCS 1855.

[Stanford Encyclopedia, 2002] Stanford Encyclopedia (2002). Leibniz's philosophy of mind. `http://plato.stanford.edu/entries/leibniz-mind/`. Accessed August 15, 2007.

[Stärk, 1994] Stärk, R. F. (1994). Input/output dependencies of normal logic programs. *Journal of Logic and Computation*, 4(3):249–262.

[Sutcliffe and Suttner, 1998] Sutcliffe, G. and Suttner, C. B. (1998). The TPTP Problem Library: CNF Release v1.2.1. *Journal of Automated Reasoning*, 21(2):177–203.

[Sutton and Fox, 2003] Sutton, D. and Fox, J. (2003). The syntax and semantics of the pro*forma* guideline modeling language. *Jounal of the American Medical Informatics Association*, 10(5):433–443.

[Tarski, 1944] Tarski, A. (1944). The semantic conception of truth and the foundations of semantics. *Philosophy and Phenomenological Research*, 4(3):341–376.

[ten Teije et al., 2006] ten Teije, A., Marcos, M., Balser, M., van Croonenborg, J., Duelli, C., van Harmelen, F., Lucas, P., Miksch, S., Reif, W., Rosenbrand, K., and S., S. (2006). Improving medical protocols by formal methods. *Artificial Intelligence in Medicine*, 36(3):193–209.

[Terenziani, 2000] Terenziani, P. (2000). Integrating temporal reasoning with periodic events. *Computational Intelligence*, 16(2):210–256.

[Tretmans et al., 2001] Tretmans, J., Wijbrans, K., and Chaudron, M. (2001). Software engineering with formal methods: The development of a storm surge barrier control system revisiting seven myths of formal methods. *Formal Methods in System Design*, 19(2):195–215.

[Tu and Musen, 1999] Tu, S. and Musen, M. (1999). A flexible approach to guideline modeling. In *Proceedings of American Medical Informatics Association Symposium (AMIA 1999)*, pages 420–424.

[Tucker and Noonan, 2007] Tucker, A. B. and Noonan, R. E. (2007). *Programming Languages – Principles and Paradigms*. McGraw-Hill, second edition.

[Turing, 1936] Turing, A. M. (1936). On computable numbers, with an application to the Entscheidungsproblem. In *Proceedings of the London Mathematical Society*, volume 42, pages 230–265.

[Turing, 1949] Turing, A. M. (1949). Checking a large routine. In *Report of a Conference on High Speed Automatic Calculating Machines*, pages 67–69. Cambridge University.

[Turing, 1950] Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 50:433–460.

[Vaandrager, 2006] Vaandrager, F. W. (2006). Does it pay off? model-based verification and validation of embedded systems! In *PROGRESS White Papers 2006*, pages 43–66.

[van Bemmel and Musen, 2002] van Bemmel, J. and Musen, M., editors (2002). *Handbook of Medical Informatics*. Springer-Verlag, Heidelberg.

[van der Meyden, 1994] van der Meyden, R. (1994). Axioms for knowledge and time in distributed systems with perfect recall. In *Symposium on Logic in Computer Science*, pages 448–457.

[van Everdingen et al., 2004] van Everdingen, J. J. E., Burgers, J. S., Assendelft, W. J. J., Swinkels, J. A., van Barneveld, T. A., and van de Klundert, J. L. M., editors (2004). *Evidence-based Guideline Development*, chapter Formulating Recommendations, page 171. Bohn, Houten.

[van Gerven, 2007] van Gerven, M. (2007). *Bayesian Networks for Clinical Decision Support*. PhD thesis, University of Nijmegen.

[Vardi, 2001] Vardi, M. Y. (2001). Branching vs. linear time: Final showdown. *LNCS*, 2031:1–19.

[Vardi and Wolper, 1994] Vardi, M. Y. and Wolper, P. (1994). Reasoning about infinite computations. *Information and Computation*, 115(1):1–37.

[Venema, 2001] Venema, Y. (2001). Temporal logic. In Goble, L., editor, *The Blackwell Guide to Philosophical Logic*, pages 203–223. Blackwell Publishers, Malden, USA.

[Wild et al., 2004] Wild, S., Roglic, G., Green, A., Sicree, R., and King, H. (2004). Global prevalence of diabetes: Estimates for the year 2000 and projections for 2030. *Diabetes Care*, 27:1047–1053.

[Wolper, 1986] Wolper, P. (1986). Expressing interesting properties of programs in propositional temporal logic. In *POPL '86: Proceedings of the 13th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 184–193, New York, NY, USA. ACM Press.

[Woolf et al., 1999] Woolf, S., Grol, R., Hutchinson, A., Eccles, M., and Grimshaw, J. (1999). Potential benefits, limitations, and harms of clinical guidelines. *British Medical Journal*, 318:527–530.

[Wos et al., 1984] Wos, L., Overbeek, R., Lusk, E., and Boyle, J. (1984). *Automated Reasoning: Introduction and Applications*. Prentice-Hall, Englewood Cliffs, NJ.

[Wos et al., 1965] Wos, L., Robinson, G., and Carson, D. (1965). Efficiency and completeness of the set of support strategy in theorem proving. *ACM Journal*, 12:536–541.

[Wyatt, 2002] Wyatt, J. C. (2002). `http://www.apsr78.dsl.pipex.com/jwconsultancy/medinf.html`. Accessed August 1, 2007.

[Zanardo and Carmo, 1993] Zanardo, A. and Carmo, J. (1993). Ockhamist computational logic: Past-sensitive necessitation in CTL. *Journal of Logic and Computation*, 3(3):249–268.

SIKS Dissertatiereeks

1998

1998-1 Johan van den Akker (CWI)
DEGAS - An Active, Temporal Database of Autonomous Objects

1998-2 Floris Wiesman (UM)
Information Retrieval by Graphically Browsing Meta-Information

1998-3 Ans Steuten (TUD)
A Contribution to the Linguistic Analysis of Business Conversations within the Language/Action Perspective

1998-4 Dennis Breuker (UM)
Memory versus Search in Games

1998-5 Eduard Oskamp (RUL)
Computerondersteuning bij Straftoemeting

1999

1999-1 Mark Sloof (VU)
Physiology of Quality Change Modelling; Automated modelling of Quality Change of Agricultural Products

1999-2 Rob Potharst (EUR)
Classification using decision trees and neural nets

1999-3 Don Beal (UM)
The Nature of Minimax Search

1999-4 Jacques Penders (UM)
The practical Art of Moving Physical Objects

1999-5 Aldo de Moor (KUB)
Empowering Communities: A Method for the Legitimate User-Driven Specification of Network Information Systems

1999-6 Niek Wijngaards (VU)
Re-design of compositional systems

1999-7 David Spelt (UT)
Verification support for object database design

1999-8 Jacques Lenting (UM)
Informed Gambling: Conception and Analysis of a Multi-Agent Mechanism for Discrete Reallocation

2000

2000-1 Frank Niessink (VU)
Perspectives on Improving Software Maintenance

2000-2 Koen Holtman (TUE)
Prototyping of CMS Storage Management

2000-3 Carolien Metselaar (UvA)
Sociaal-organisatorische gevolgen van kennistechnologie; een procesbenadering en actorperspectie

2000-4 Geert de Haan (VU)
ETAG, A Formal Model of Competence Knowledge for User Interface Design

2000-5 Ruud van der Pol (UM)
Knowledge-based Query Formulation in Information Retrieval

2000-6 Rogier van Eijk (UU)
Programming Languages for Agent Communication

2000-7 Niels Peek (UU)
Decision-theoretic Planning of Clinical Patient Management

2000-8 Veerle Coupé (EUR)
Sensitivity Analyis of Decision-Theoretic Networks

2000-9 Florian Waas (CWI)
Principles of Probabilistic Query Optimization

2000-10 Niels Nes (CWI)
Image Database Management System Design Considerations, Algorithms and Architecture

2000-11 Jonas Karlsson (CWI)
Scalable Distributed Data Structures for Database Management

2001

2001-1 Silja Renooij (UU)
Qualitative Approaches to Quantifying Probabilistic Networks

2001-2 Koen Hindriks (UU)
Agent Programming Languages: Programming with Mental Models

2001-3 Maarten van Someren (UvA)
Learning as problem solving

2001-4 Evgueni Smirnov (UM)
Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets

2001-5 Jacco van Ossenbruggen (VU)
Processing Structured Hypermedia: A Matter of Style

2001-6 Martijn van Welie (VU)
Task-based User Interface Design

2001-7 Bastiaan Schonhage (VU)
Diva: Architectural Perspectives on Information Visualization

2001-8 Pascal van Eck (VU)
A Compositional Semantic Structure for Multi-Agent Systems Dynamics

2001-9 Pieter Jan 't Hoen (RUL)
Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes

2001-10 Maarten Sierhuis (UvA)
Modeling and Simulating Work Practice BRAHMS: a multiagent modeling and simulation language for work practice analysis and design

2001-11 Tom van Engers (VU)
Knowledge Management: The Role of Mental Models in Business Systems Design

2002

2002-01 Nico Lassing (VU)
Architecture-Level Modifiability Analysis

2002-02 Roelof van Zwol (UT)
Modelling and searching web-based document collections

2002-03 Henk Ernst Blok (UT)
Database Optimization Aspects for Information Retrieval

2002-04 Juan Roberto Castelo Valdueza (UU)
The Discrete Acyclic Digraph Markov Model in Data Mining

2002-05 Radu Serban (VU)
The Private Cyberspace Modeling Electronic Environments inhabited by Privacy-concerned Agents

2002-06 Laurens Mommers (UL)
Applied legal epistemology; Building a knowledge-based ontology of the legal domain

2002-07 Peter Boncz (CWI)
Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications

2002-08 Jaap Gordijn (VU)
Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas

2002-09 Willem-Jan van den Heuvel (KUB)

Integrating Modern Business Applications with Objectified Legacy Systems

2002-10 Brian Sheppard (UM)
Towards Perfect Play of Scrabble

2002-11 Wouter Wijngaards (VU)
Agent Based Modelling of Dynamics: Biological and Organisational Applications

2002-12 Albrecht Schmidt (UvA)
Processing XML in Database Systems

2002-13 Hongjing Wu (TUE)
A Reference Architecture for Adaptive Hypermedia Applications

2002-14 Wieke de Vries (UU)
Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems

2002-15 Rik Eshuis (UT)
Semantics and Verification of UML Activity Diagrams for Workflow Modelling

2002-16 Pieter van Langen (VU)
The Anatomy of Design: Foundations, Models and Applications

2002-17 Stefan Manegold (UvA)
Understanding, Modeling, and Improving Main-Memory Database Performance

2003

2003-01 Heiner Stuckenschmidt (VU)
Ontology-Based Information Sharing in Weakly Structured Environments

2003-02 Jan Broersen (VU)
Modal Action Logics for Reasoning About Reactive Systems

2003-03 Martijn Schuemie (TUD)
Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy

2003-04 Milan Petkovic (UT)
Content-Based Video Retrieval Supported by Database Technology

2003-05 Jos Lehmann (UvA)
Causation in Artificial Intelligence and Law - A modelling approach

2003-06 Boris van Schooten (UT)
Development and specification of virtual environments

2003-07 Machiel Jansen (UvA)
Formal Explorations of Knowledge Intensive Tasks

2003-08 Yongping Ran (UM)
Repair Based Scheduling

2003-09 Rens Kortmann (UM)
The resolution of visually guided behaviour

2003-10 Andreas Lincke (UvT)
Electronic Business Negotiation: Some experimental studies on the interaction between medium, innovation context and culture

2003-11 Simon Keizer (UT)
Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks

2003-12 Roeland Ordelman (UT)
Dutch speech recognition in multimedia information retrieval

2003-13 Jeroen Donkers (UM)
Nosce Hostem - Searching with Opponent Models

2003-14 Stijn Hoppenbrouwers (KUN)
Freezing Language: Conceptualisation Processes across ICT-Supported Organisations

2003-15 Mathijs de Weerdt (TUD)
Plan Merging in Multi-Agent Systems

2003-16 Menzo Windhouwer (CWI)
Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouses

2003-17 David Jansen (UT)
Extensions of Statecharts with Probability, Time, and Stochastic Timing

2003-18 Levente Kocsis (UM)
Learning Search Decisions

2004

2004-01 Virginia Dignum (UU)
A Model for Organizational Interaction: Based on Agents, Founded in Logic

2004-02 Lai Xu (UvT)
Monitoring Multi-party Contracts for E-business

2004-03 Perry Groot (VU)
A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving

2004-04 Chris van Aart (UVA)
Organizational Principles for Multi-Agent Architectures

2004-05 Viara Popova (EUR)
Knowledge discovery and monotonicity

2004-06 Bart-Jan Hommes (TUD)
The Evaluation of Business Process Modeling Techniques

2004-07 Elise Boltjes (UM)
Voorbeeldig onderwijs; voorbeeldgestuurd onderwijs, een opstap naar abstract denken, vooral voor meisjes

2004-08 Joop Verbeek(UM)
Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale politi ele gegevensuitwisseling en digitale expertise

2004-09 Martin Caminada (VU)
For the Sake of the Argument; explorations into argument-based reasoning

2004-10 Suzanne Kabel (UVA)
Knowledge-rich indexing of learning-objects

2004-11 Michel Klein (VU)
Change Management for Distributed Ontologies

2004-12 The Duy Bui (UT)
Creating emotions and facial expressions for embodied agents

2004-13 Wojciech Jamroga (UT)
Using Multiple Models of Reality: On Agents who Know how to Play

2004-14 Paul Harrenstein (UU)
Logic in Conflict. Logical Explorations in Strategic Equilibrium

2004-15 Arno Knobbe (UU)
Multi-Relational Data Mining

2004-16 Federico Divina (VU)
Hybrid Genetic Relational Search for Inductive Learning

2004-17 Mark Winands (UM)
Informed Search in Complex Games

2004-18 Vania Bessa Machado (UvA)
Supporting the Construction of Qualitative Knowledge Models

2004-19 Thijs Westerveld (UT)
Using generative probabilistic models for multimedia retrieval

2004-20 Madelon Evers (Nyenrode)
Learning from Design: facilitating multidisciplinary design teams

2005

2005-01 Floor Verdenius (UVA)
Methodological Aspects of Designing Induction-Based Applications

2005-02 Erik van der Werf (UM))
AI techniques for the game of Go

2005-03 Franc Grootjen (RUN)
A Pragmatic Approach to the Conceptualisation of Language

2005-04 Nirvana Meratnia (UT)
Towards Database Support for Moving Object data

2005-05 Gabriel Infante-Lopez (UVA)
Two-Level Probabilistic Grammars for Natural Language Parsing

2005-06 Pieter Spronck (UM)
Adaptive Game AI

2005-07 Flavius Frasincar (TUE)
Hypermedia Presentation Generation for Semantic Web Information Systems

2005-08 Richard Vdovjak (TUE)
A Model-driven Approach for Building Distributed Ontology-based Web Applications

2005-09 Jeen Broekstra (VU)
Storage, Querying and Inferencing for Semantic Web Languages

2005-10 Anders Bouwer (UVA)
Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments

2005-11 Elth Ogston (VU)
Agent Based Matchmaking and Clustering - A Decentralized Approach to Search

2005-12 Csaba Boer (EUR)
Distributed Simulation in Industry

2005-13 Fred Hamburg (UL)
Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen

2005-14 Borys Omelayenko (VU)
Web-Service configuration on the Semantic Web; Exploring how semantics meets pragmatics

2005-15 Tibor Bosse (VU)
Analysis of the Dynamics of Cognitive Processes

2005-16 Joris Graaumans (UU)
Usability of XML Query Languages

2005-17 Boris Shishkov (TUD)
Software Specification Based on Re-usable Business Components

2005-18 Danielle Sent (UU)
Test-selection strategies for probabilistic networks

2005-19 Michel van Dartel (UM)
Situated Representation

2005-20 Cristina Coteanu (UL)
Cyber Consumer Law, State of the Art and Perspectives

2005-21 Wijnand Derks (UT)
Improving Concurrency and Recovery in Database Systems by Exploiting Application Semantics

2006

2006-01 Samuil Angelov (TUE)
Foundations of B2B Electronic Contracting

2006-02 Cristina Chisalita (VU)
Contextual issues in the design and use of information technology in organizations

2006-03 Noor Christoph (UVA)
The role of metacognitive skills in learning to solve problems

2006-04 Marta Sabou (VU)
Building Web Service Ontologies

2006-05 Cees Pierik (UU)
Validation Techniques for Object-Oriented Proof Outlines

2006-06 Ziv Baida (VU)
Software-aided Service Bundling - Intelligent Methods & Tools for Graphical Service Modeling

2006-07 Marko Smiljanic (UT)
XML schema matching – balancing efficiency and effectiveness by means of clustering

2006-08 Eelco Herder (UT)
Forward, Back and Home Again - Analyzing User Behavior on the Web

2006-09 Mohamed Wahdan (UM)
Automatic Formulation of the Auditor's Opinion

2006-10 Ronny Siebes (VU)
Semantic Routing in Peer-to-Peer Systems

2006-11 Joeri van Ruth (UT)
Flattening Queries over Nested Data Types

2006-12 Bert Bongers (VU)
Interactivation - Towards an ecology of people, our technological environment, and the arts

2006-13 Henk-Jan Lebbink (UU)
Dialogue and Decision Games for Information Exchanging Agents

2006-14 Johan Hoorn (VU)
Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change

2006-15 Rainer Malik (UU)
CONAN: Text Mining in the Biomedical Domain

2006-16 Carsten Riggelsen (UU)
Approximation Methods for Efficient Learning of Bayesian Networks

2006-17 Stacey Nagata (UU)
User Assistance for Multitasking with Interruptions on a Mobile Device

2006-18 Valentin Zhizhkun (UVA)
Graph transformation for Natural Language Processing

2006-19 Birna van Riemsdijk (UU)
Cognitive Agent Programming: A Semantic Approach

2006-20 Marina Velikova (UvT)
Monotone models for prediction in data mining

2006-21 Bas van Gils (RUN)
Aptness on the Web

2006-22 Paul de Vrieze (RUN)
Fundaments of Adaptive Personalisation

2006-23 Ion Juvina (UU)
Development of Cognitive Model for Navigating on the Web

2006-24 Laura Hollink (VU)
Semantic Annotation for Retrieval of Visual Resources

2006-25 Madalina Drugan (UU)
Conditional log-likelihood MDL and Evolutionary MCMC

2006-26 Vojkan Mihajlovic (UT)
Score Region Algebra: A Flexible Framework for Structured Information Retrieval

2006-27 Stefano Bocconi (CWI)
Vox Populi: generating video documentaries from semantically annotated media repositories

2006-28 Borkur Sigurbjornsson (UVA)
Focused Information Access using XML Element Retrieval

2007

2007-01 Kees Leune (UvT)
Access Control and Service-Oriented Architectures

2007-02 Wouter Teepe (RUG)
Reconciling Information Exchange and Confidentiality: A Formal Approach

2007-03 Peter Mika (VU)
Social Networks and the Semantic Web

2007-04 Jurriaan van Diggelen (UU)
Achieving Semantic Interoperability in Multi-agent Systems: a dialogue-based approach

2007-05 Bart Schermer (UL)
Software Agents, Surveillance, and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance

2007-06 Gilad Mishne (UVA)
Applied Text Analytics for Blogs

2007-07 Natasa Jovanovic (UT)
To Whom It May Concern - Addressee Identification in Face-to-Face Meetings

2007-08 Mark Hoogendoorn (VU)
Modeling of Change in Multi-Agent Organizations

2007-09 David Mobach (VU)
Agent-Based Mediated Service Negotiation

2007-10 Huib Aldewereld (UU)
Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols

2007-11 Natalia Stash (TUE)
Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System

2007-12 Marcel van Gerven (RUN)
Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty

2007-13 Rutger Rienks (UT)
Meetings in Smart Environments; Implications of Progressing Technology

2007-14 Niek Bergboer (UM)
Context-Based Image Analysis

2007-15 Joyca Lacroix (UM)
NIM: a Situated Computational Memory Model

2007-16 Davide Grossi (UU)
Designing Invisible Handcuffs. Formal investigations in Institutions and Organizations for Multi-agent Systems

2007-17 Theodore Charitos (UU)
Reasoning with Dynamic Networks in Practice

2007-18 Bart Orriens (UvT)
On the development an management of adaptive business collaborations

2007-19 David Levy (UM)
Intimate relationships with artificial partners

2007-20 Slinger Jansen (UU)
Customer Configuration Updating in a Software Supply Network

2007-21 Karianne Vermaas (UU)
Fast diffusion and broadening use: A research on residential adoption and usage of broadband internet in the Netherlands between 2001 and 2005

2007-22 Zlatko Zlatev (UT)
Goal-oriented design of value and process models from patterns

2007-23 Peter Barna (TUE)
Specification of Application Logic in Web Information Systems

2007-24 Georgina Ramírez Camps (CWI)
Structural Features in XML Retrieval

2007-25 Joost Schalken (VU)
Empirical Investigations in Software Process Improvement

2008

2008-01 Katalin Boer-Sorbán (EUR)
Agent-Based Simulation of Financial Markets: A modular, continuous-time approach

2008-02 Alexei Sharpanskykh (VU)
On Computer-Aided Methods for Modeling and Analysis of Organizations

2008-03 Vera Hollink (UVA)
Optimizing hierarchical menus: a usage-based approach

2008-04 Ander de Keijzer (UT)
Management of Uncertain Data – towards unattended integration

2008-05 Bela Mutschler (UT)
Modeling and simulating causal dependencies on process-aware information systems from a cost perspective

2008-06 Arjen Hommersom (RUN)
On the Application of Formal Methods to Clinical Guidelines: an Artificial Intelligence Perspective

Samenvatting

In de geneeskunde kunnen fouten grote gevolgen hebben voor de gezondheid en levensverwachting van patiënten. Goede kwaliteit van zorg is daarom zeer belangrijk. In toenemende mate wordt in de gezondheidszorg gedefinieerd hoe voor een bepaald ziektebeeld kwaliteit van zorg moet worden opgevat. Diverse nationale en internationale instanties zien erop toe dat de opgestelde richtlijnen voor het handhaven van de kwaliteit van zorg worden nageleefd.

In de informatica zijn in de afgelopen decennia methoden en technieken ontwikkeld waarmee eigenschappen van allerlei soorten systemen kunnen worden onderzocht. Hierbij kan men denken aan het simuleren van het gedrag van een systeem aan de hand van een model van het systeem. Als men eenmaal een model van een systeem heeft, is het ook mogelijk bepaalde eigenschappen ervan te bewijzen of de toestanden van het systeem te doorzoeken. De wiskundige methoden en technieken die hiertoe gebruikt worden, staan in de informatica bekend als formele methoden.

Een medische of klinische richtlijn is een document met aanbevelingen, adviezen en handelingsinstructies ter ondersteuning van de besluitvorming van professionals in de zorg en patiënten, berustend op de resultaten van wetenschappelijk onderzoek met daarop gebaseerde discussie en aansluitende meningsvorming gericht op het expliciteren van doeltreffend en doelmatig professioneel handelen[1]. Vanuit het gebied uit de informatica dat zich bezig houdt met kunstmatige intelligentie is sinds enige tijd interesse om deze richtlijnen te gebruiken als basis voor elektronische beslissingsondersteuning van artsen of ter aanvulling van een elektronische patiëntdossier. Hierdoor is het begrip 'elektronische richtlijn' ontstaan. Vanuit het oogpunt van formele methoden roept dit verschillende vragen op. De eerste vraag is of de elektronische richtlijn inderdaad een juiste afspiegeling is van de papieren richtlijn. De tweede vraag, die meer medische georiënteerd is, is of de aanbevelingen, adviezen en handelingsinstructies wel juist zijn. Het proces om correctheid van de richtlijn te bewijzen wordt ook wel *verificatie* van richtlijnen genoemd.

De hoofdstukken 1, 2 en 3 van dit proefschrift zijn inleidende hoofdstukken waarin de relevante technische en medische concepten worden geïntroduceerd. In hoofdstuk 4 wordt de de kwaliteit van de richtlijn voor de behandeling van

[1] Van Everdingen e.a., Evidence-based richtlijnontwikkeling, 2004, Houten, Bohn Stafleu Van Loghum, p.4

diabetes mellitus type 2 in de huisartsenpraktijk onderzocht. Door middel van hypothetisch te redeneren over verschillende mogelijke behandelingen, gebruikmakende van medische achtergrondkennis, kunnen er uitspraken worden gedaan over wat de juiste behandeling is. In het bijzonder gaat het in dit hoofdstuk over de vraag hoe dit kan worden uitgevoerd in een automatische stellingenbewijzer.

In de afgelopen tijd zijn er speciale talen ontwikkeld om de verschillende interventies en de temporele relatie tussen deze interventies die in richtlijnen staan te modelleren. In hoofdstuk 5 wordt deze taal gebruikt om de diabetes richtlijn uit hoofdstuk 4 te beschrijven en wordt een methode ontwikkeld om deze kennis met de medische achtergrondkennis te combineren. De kwaliteit wordt onderzocht met behulp van een interactieve stellingenbewijzer.

Een andere manier om formele methoden te gebruiken in de context van richtlijnen is om de richtlijn als gouden standaard te nemen voor verificatie. Protocollen zijn documenten afgeleid van richtlijnen die zijn toegespitst op lokale situaties (instellingsniveau). Dit roept de vraag op in hoeverre deze protocollen consistent zijn met de (nationale) richtlijn. Dezelfde vraag kan worden gesteld voor de handelingen van artsen in de praktijk: in hoeverre zijn deze handelingen in overeenstemming met de aanbevelingen uit de richtlijn? Gebruikmakende van een techniek genaamd model-checking worden deze vragen in hoofdstuk 6 onderzocht.

In hoofdstuk 7 gaat het over de vraag hoe richtlijnen kunnen worden gespecificeerd in een logische taal. Ondanks dat dit geen aanpak is die lijkt op huidige richtlijnmodelleertalen, is het een onderzoeksrichting die kan worden gemotiveerd uit enerzijds de structuur van moderne richtlijn en anderzijds de speciale wensen die bij verificatie aan een formeel model kunnen worden gesteld, zoals de complexiteit van redeneren. Inzichten voorkomende uit dit hoofdstuk zouden ondersteuning kunnen bieden bij het ontwerpen van een specifieke richtlijnmodelleertaal.

In hoofdstuk 8 worden de verschillende hoofdstukken in perspectief gebracht. We gaan in op de vragen die de basis vormden voor dit onderzoek en welke vragen verder zouden kunnen worden onderzocht.

Curriculum Vitae

Arjen Hommersom was born on February 25, 1980 in Velp, the Netherlands. In 1998, he finished secondary school at 't Rhedens in Rozendaal. As he had a keen interest in both medicine and technology, he decided to study medical technical computer science at the University of Utrecht. During the first years, he became interested in the area of artificial intelligence, so he focused his studies on courses related to this area. At the end of 2003, he graduated in the field of intelligent systems by writing a thesis which deals with reasoning about security protocols using logic.

Since the beginning of 2004, he has been a junior researcher at the institute for computing and information sciences. He never lost his interest in medicine, which is the reason he sought a working environment where he could combine his interest in artificial intelligence with medicine. He became a member of the Protocure project, which was a European project aimed at improving the quality of medical guidelines using information technology.

Currently, Arjen is working as a postdoctoral researcher at the same research institute.