Nonmonotonic Inference Systems for Modelling Dynamic Processes

Craig MacNish

CUED/F-INFENG/TR.102 (1992)

Department of Engineering
University of Cambridge
Trumpington Street
Cambridge CB2 1PZ, UK

Nonmonotonic Inference Systems for Modelling Dynamic Processes

Craig MacNish

Trinity College
Cambridge[1]

This report is a copy of a dissertation by the same name submitted in February 1992 for the degree of Doctor of Philosophy at the University of Cambridge

[1] Current address: Department of Computer Science, University of York, York YO1 5DD, UK

Declaration

This dissertation describes my own work and includes nothing which is the outcome of work done in collaboration. Any work which is derived from other sources is explicitly cited in the text.

Craig MacNish
February 1992

Abstract

The ability to model dynamic processes mathematically is an important aspect of automatic control. While traditional control theory has proven successful in characterising continuous dynamic processes, it appears that different mathematical tools are required for modelling processes characterised by discrete events. This thesis investigates the use of nonmonotonic logics as a basis for modelling such processes.

We develop a nonmonotonic inference system which has a number of desirable features for process modelling. The system is suitable for expressing causal relationships and supports defeasible inferences from incomplete knowledge bases. The relationship between the input and output of the system is precisely described. The system is guaranteed to generate a single consistent output for any allowable input, and the output can be generated using algorithms based on classical deduction.

The system is based on two well-known nonmonotonic reasoning formalisms—Shoham's logic of *chronological ignorance* (CI) and Reiter's *default logic*. A number of improvements to these formalisms are suggested.

We argue that the use of modal logic to represent knowledge of assertions in CI is unwarranted and provide an alternative truth-functional language called *asserted logic*. A simplified version of CI for causal theories is developed using asserted logic and shown to be equivalent to Shoham's original formalism.

We propose solutions to two problems associated with default logic, namely *incoherence* and the *multiple extension problem*. The former makes use of asserted logic to normalise defaults. The latter is based on an extension of default logic, called *hierarchical default logic* (HDL), which incorporates an ordered default structure. In order to calculate the belief set for HDL theories we redefine default logic in a way which "factors out" deductive closure. This leads to a proof procedure for default logic which extends to the hierarchical framework.

We provide a proof theory for chronological ignorance based on HDL, using asserted logic as the underlying language. We then suggest improvements to the proof-theoretic framework and, in particular, an alternative framework which does not require an epistemic logic.

The use of the resulting system, called *chronological augmentation*, is demonstrated by providing models for two simple assembly processes.

Keywords

Automated reasoning, nonmonotonic logics, knowledg-based modelling, discrete-event dynamic systems, artificial intelligence, planning.

Acknowledgements

This work has been funded by Trinity College, Cambridge, under the Prince of Wales Studentship. Without the generous support of the College it would not have been possible. Additional financial support for academic travel has been provided by Cambridge University Engineering Department.

I would like to thank Frank Fallside, my supervisor, for valuable advice during the course of this work. I would also like to thank Rajeev Goré and Ian Green for many stimulating and enjoyable discussions. Raj provided a number of useful theorem prover implementations, and Ian provided valuable comments on an earlier draft of this thesis. My work has also benefited from discussions with André Trudel, John Gooday and Mary-Anne Williams, and from help with computing facilities from members of the CUED Speech, Vision, Robotics and AI Group.

I would like to give a special thanks to my family in Australia for their understanding and encouragement over the years, and to my friends and colleagues both here and in Australia for keeping life enjoyable and morale high. Finally I would like to give a special thanks to Jo for her patience and support.

Contents

List of Figures

List of Tables

Notation and Abbreviations

List of Symbols

Sets and Relations

$\Re$, $\mathbb{N}$, $\mathbb{Z}$	real numbers, natural numbers and integers
$\cup$, $\cap$, $\setminus$	union, intersection and difference of sets
$\subseteq$, $\subset$, $\in$	subset, proper subset and element
$\emptyset$	empty set
$A_1 \times A_2$	cartesian product of A_1 and A_2
A^2	$A \times A$
$\wp(A)$, 2^A	power set of A
$\mathcal{D}(R)$, $\mathcal{R}(R)$	domain and range of relation R
$R_1 \circ R_2$	composition of relations R_1 and R_2

Logic

t, f, u	truth values "true", "false" and "unknown"
$\neg$, $\wedge$, $\vee$	negation, conjunction and disjunction connectives
$\rightarrow$, $\leftrightarrow$	implication and equivalence connectives
$-$	strong negation connective
$\forall$, $\exists$	universal and existential quantifiers
$\models$	satisfaction, logical and tautological consequence
$\vdash$	first-order and propositional deduction
$\bigwedge_{i=1}^{m} \alpha_i$	$\alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_m$
L, $\mathrm{L_S}$, $\mathrm{L_P}$	classical first-order, sentential and propositional logics
Th	deductive closure ("theorems")
A_D, E_D	augmentation and extension over default set D
$\boldsymbol{A_D}$, $\boldsymbol{E_D}$	recursive augmentation and extension of hierarchical default set $\boldsymbol{D}$
$\Box$, $\Diamond$	the modal operator and its dual
$\mathbf{T}$, $\mathbf{F}$, $\mathbf{P}$, $\mathbf{U}$	the assertion operator and related "operators"
TRUE	reification relation in CI
KI	Kripke interpretation

Notational Conventions

This work refers to various sources adopting a wide range of notational conventions. Where possible we have attempted to standardise the notation according to the following conventions.

Type style	**Uses (and examples)**
lowercase Greek letters	variables ranging over wffs (α, β, γ, ϕ, ψ, ...)
	truth valuations and valuations (σ, τ)
uppercase Greek letters	variables ranging over sets of wffs (Φ, Ψ, Δ, ...)
typewriter style characters	well-formed formulas
—lowercase	predicate, function and constant symbols
—uppercase	variables, metalevel relations
lowercase (maths) italic letters	variables ranging over predicate, function and constant symbols (p,f,t, ...)
	variables ranging over base formulas (x, y, z, ...)
uppercase (maths) italic letters	sets other than wffs (parameter sets, graphs, ...)
	generic names (and wffs) of logics (L, L_1, L_2, ...)
uppercase and lowercase italics	names of relations (R,Th,CWA,...)
uppercase bold (maths) italics	hierarchical parameter sets ($\boldsymbol{D}$, $\boldsymbol{F}$, $\boldsymbol{U}$, ...)
	recursive relations ($\boldsymbol{A_D}$, $\boldsymbol{E_D}$, ...)
uppercase roman letters	the names (and wffs) of specific logics (L, AL, ...)

Abbreviations

AI	artificial intelligence
AL	asserted logic
AEL	autoepistemic logic
BTK	the temporal logic of Bacchus *et al*
CI	chronological ignorance
CWA	closed world assumption
DEDS	discrete-event dynamic system
GCH	the temporal logic of Trudel
HAEL	hierarchic autoepistemic logic
HDL	hierarchical default logic
KB	knowledge base
LCT	continuous-time logic
MEP	multiple extension problem
TAL	temporal asserted logic
TK	logic of temporal knowledge
cmi	chronologically maximally ignorant
ltp	latest time point
wff(s)	well-formed formula(s)

Chapter 1

Introduction

A model means, in the context of a study of the dynamics and control of plant, a representation of the plant behaviour in terms of mathematical statements... The 'representation [...] in terms of mathematical statements' is open to various interpretations.

R. J. Richards, *An Introduction to Dynamics and Control* (1979)

The need for reliable theories of automatic control has been recognised by engineers for more than a century. Central to this requirement is the ability to characterise mathematically or *model* the systems or processes under consideration. While traditional control theory has proven successful for modelling and controlling continuous dynamic systems, it is not applicable to many domains which are naturally characterised by discrete events. It appears that different mathematical tools will be required for modelling these types of systems.

The last 40 years have seen the emergence of systems which seek to mimic certain aspects of "human reasoning" by automating the inference mechanisms of mathematical logic. These systems have found applications in areas such as automatic planning and "intelligent" knowledge-based systems and appear to provide a promising approach for modelling dynamic systems.

Applications such as these have highlighted problems, however, which are not easily handled by traditional or *classical* logics. In order to overcome the restrictions of traditional systems a number of extensions to classical logics have been proposed. Important examples include *nonmonotonic* logics, which incorporate default information, and *temporal* or *state-based* logics which can be used to reason about causality and the effects of actions over time.

This thesis investigates and develops nonmonotonic and state-based reasoning systems which are appropriate for modelling discrete-event dynamic systems.

1.1 Towards Logic-Based Control

The basic structure of a feedback control system is shown in Figure 1.1. The system consists of the process (or a model of the process) to be controlled, a component for comparing

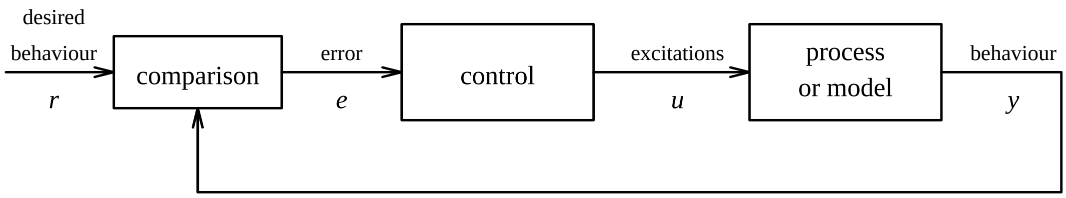

Figure 1.1: The structure of a typical feedback control system.

the behaviour of the process with the desired behaviour, and a controller which generates *excitations* or inputs to the process based upon the *error* between the desired and actual behaviour. (For an introduction to control concepts see [Ric79, Dor86].) The variables r, e, u and y represent information which is passed around the system and the comparison, control and modelling components perform some operation on this information.

In traditional or *quantitative* control theory the information passed around the system is typically described in terms of continuous real-valued functions [Lei87b] and the operations are described by *transfer functions* based on differential or difference equations (or their frequency domain counterparts). This type of representation has proven highly successful in domains in which the quantities under consideration change continuously over time. We call these *continuous dynamic systems*. Examples include electrical, mechanical and thermodynamic systems in which the physical laws governing the processes can be described by differential equations [Dor86].

There are many systems, however, which stand to benefit from the application of feedback control concepts, but for which quantitative information alone is not adequate. Typical examples include

- high-level control systems for robots, in which decisions must be made about where the robots should go and what actions they should take,
- production scheduling systems, in which the delivery and assembly of components must be co-ordinated,
- safety shutdown systems, in which alarm signals must be interpreted and acted upon, and
- traffic control systems, in which mobile objects must be scheduled to avoid collisions and optimise travel times.

These systems require descriptions of the operating environment, the relationships between components, and the goals or tasks to be performed. They are often called *discrete-event dynamic systems* (DEDSs) since events which change the state of the system occur over distinct time intervals.

1.1.1 Declarative Knowledge and Mechanised Logic

Much of the information required for describing DEDSs appears to be expressed most conveniently by declarations or statements about the world; for example "part A is connected to part B", "robot 1 is at the conveyor", "the evacuation alarm is on" and so on. Information of this type is often called *declarative knowledge* (see [GN87, Chap 2] for a detailed definition). We use the term *declarative control* to refer to control systems that deal with declarative knowledge.[1]

Since the 1950's a great deal of effort has been devoted, primarily in artificial intelligence (AI) and theoretical computer science, to finding useful ways of representing and reasoning about declarative information. Some of the more influential approaches include *mechanised logics* (for example [Rob65]), *production rules* [NS72], *semantic networks* [Sim73], *frames* [Min75], *truth maintenance systems* [Doy79] and *fuzzy logic* [Zad75].

Approaches based on the mechanisation of mathematical logic form an attractive basis for declarative control for a number of reasons. The principles of mathematical logic have been extensively studied and documented. These principles provide tools for investigating the behaviour of logic-based systems, just as the principles of differential calculus provide a basis for analysing traditional control systems. Furthermore, the techniques involved, like those of differential calculus, are independent of the domain of application. This allows us to develop a *deductive analytic theory* [Lei87b]; that is, a class of system with very general properties which can be specialised to the particular system under study. In addition the implementation of logic-based systems has been studied extensively, and standard "theorem-proving" systems (for example [Fit88]) and logic-based programming languages ([CM84]) are readily available.

Logic-based systems have a number of disadvantages, however, both from a representational and from a computational point of view. The attractive properties of logic systems are a consequence of adhering to formal rules which limit the scope of the knowledge that can be represented. In addition first-order logic is not decidable [BJ80] (that is, there is no effective procedure for deciding whether one formula is a logical consequence of others) and even its decidable subclasses, such as propositional logic, suffer from computational explosion. The satisfiability problem for propositional logic, for example, is known to be NP-complete [GJ79]. Problems such as these have prompted many researchers interested in applications of declarative knowledge to choose alternative approaches. Rule-based systems, including production rules and fuzzy logic, are particularly common for applications in modelling and control [Lei87a].

1.1.2 Nonclassical Logics

An alternative to abandoning logical systems is to investigate variations on classical logics with the aim of improving their representational and computational properties. These *nonclassical* logics generally fall into two categories. *Extensions* to classical logic add new symbols and inference procedures while preserving the existing framework. *Restrictions* to

[1] In fact a lot can be said about continuous dynamic systems using declarative knowledge. This is achieved by discretising the domain. For example, motion of an object might be catagorized according to whether the relevant derivatives are positive (+), negative (−) or zero (0). Systems based on this approach are often called *qualitative reasoning systems* [Bob84]. This approach represents perhaps the most extensive application of artificial intelligence to modelling and control (see for example [Lei87a]) to the extent that the term knowledge-based control is often used synonymously with qualitative control. In order to avoid this association we use the term *declarative control*.

classical logic share the same symbols but do not support the same inferences [Haa78]. The former category includes *modal logics* [HC68] which form the basis of logics of knowledge and belief [Hin62, McD82, Moo85a]. Examples of the latter category include *intuitionistic logic* [Hey66] and multiple-valued logics [Res69]. Overviews of these systems can be found in [GN87, Haa78, Tur84].

In this thesis we examine variations to classical logic which appear to be appropriate for modelling discrete-event dynamic systems. These variations are often called nonmonotonic and temporal or state-based reasoning systems and have grown largely in response to problems encountered in automatic planning and intelligent knowledge-based systems. The motivation for these logics is discussed in Section 1.3.

1.2 Logical Prerequisites

We assume in this thesis that the reader has a working knowledge of first-order logic and proof theory. General introductions to logic and automated theorem proving can be found in [Bun83, CL73], while a good introduction to logics in the context of artificial intelligence is contained in [GN87]. The first three chapters of [BM77] provide a rigorous account of the syntax and semantics of propositional and first-order logic, and the conventions used in this thesis are adopted largely from this source. For a thorough introduction to computational aspects of logic the reader is referred to [BJ80], and for an accessible introduction to philosophical issues [Haa78] is recommended.

The logics developed in this thesis are based on a classical first-order logic L, the precise definition of which is given in Appendix A. For brevity we use the name of the logics (such as L) to denote the set of all well-formed formulas (wffs) belonging to the logics.

We often make use of two subclasses of L: a *sentential language* denoted L_S and a *propositional language* denoted L_P. The sentential (or *closed*) language consists of the sentences (or closed wffs) of L. First-order consequence ($\models$) and deduction ($\vdash$) are restricted to sentences when used in this context.

The propositional language, L_P, consists of the sentences of L_S which can be formed using only individual constants, extralogical predicate symbols and the logical connectives. Thus we do not allow variables, quantifiers, equality, or function symbols with arity greater than zero. We call the resulting formulas *propositional sentences*. In this context first-order consequence and deduction are restricted to propositional sentences.

An advantage of dealing with L_P is that first-order (or logical) consequence and propositional (or tautological) consequence are equivalent (see Appendix A). The same is true of first-order and propositional deduction. We can therefore use propositional proof procedures to determine the validity of L_P formulas. We adopt this approach, rather than defining a separate propositional language, so that we can use the same syntax for our propositional and first-order languages.

We call a set of well-formed formulas a *theory* or *knowledge base* (KB). The latter term often refers more specifically to a store of declarative knowledge.

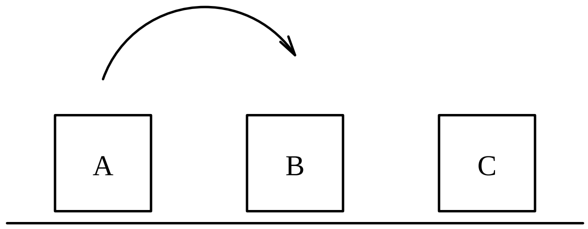

Figure 1.2: A "blocks world" system with one possible event indicated.

1.3 A Declarative Model

The task of a model is to transform a set of excitations (or *input description*) into an *output description* which predicts the system response. In the control loop illustrated in Figure 1.1 the excitations are instructions from the controller. Alternatively they may be a set of initial conditions for testing a specific scenario.

The model itself must contain enough information to characterise the dynamic system under consideration. The information we use to model DEDSs can be roughly divided into the following three types:

A Priori Facts describe assertions about the system which are not conditional upon other factors. They might include information which does not change over time, such as "object R1 is a robot", or initial conditions which hold each time the system is tested on a new scenario.

Causal Relationships describe conditional assertions and changes (or lack of change) in the state of the system due to events or actions. In the "blocks world" system shown in Figure 1.2, for example, the event "block A is placed on block B" changes the position of block A and the fact that the top of block B is clear, while preserving the state of block C.

We will generally consider systems in which events (or chains of events) are initiated by the excitations to the model.

Defaults describe what *assumptions* should be made about the system when the information required is not available. They are necessary because the declarative knowledge that we have about a system is usually incomplete. Consider, for example, a mobile robot which is provided with a map (or set of facts) describing some terrain which it must cross. The map may be inaccurate for a number of reasons: the detail of the map may be insufficient due to storage limitations, the terrain may have changed since the map was constructed, and the map cannot contain the positions of other mobile objects. The robot must proceed on the assumption that the map is correct and revise its knowledge as its sensors pick up contradictory information.

Inferences that are based on assumptions and which can later be withdrawn are called *defeasible inferences*.

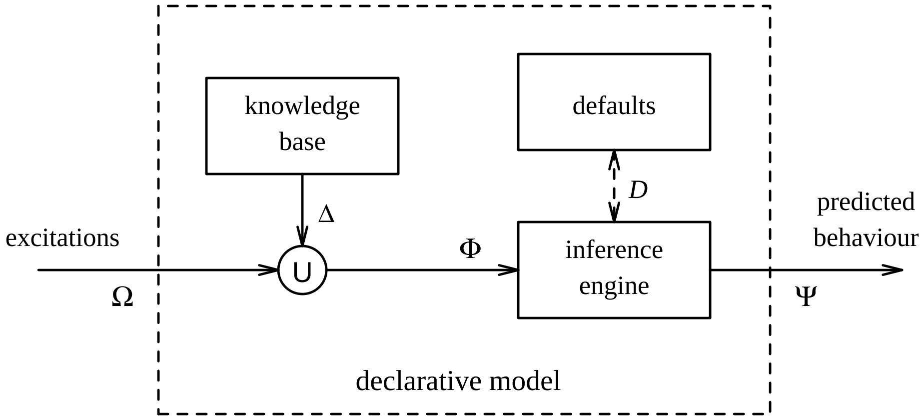

Figure 1.3: The structure of a logic-based declarative model.

Logic systems which are designed to represent and reason about causal relationships in dynamic systems are often called *causal*, *temporal* or *state-based* reasoning systems. The latter terms reflect the need to attach some notion of state or time to assertions. This is necessary for modelling discrete-event dynamic systems since we are primarily interested in the way information changes over time. We need to be able to state when events occur in relation to one another, for how long they occur, what must be true at the time they occur and so on. The problem of adequately incorporating information about time in logic-based systems has received much attention and a number of instructive contributions are described in the following chapter.

Logic systems that have the ability to incorporate default information are often called *nonmonotonic* logics. (The term "nonmonotonic" is defined in Chapter 3.) There are many possible ways of augmenting incomplete knowledge using default information and the behaviour of a nonmonotonic reasoning system is highly dependent on what strategy is chosen. The characteristics of nonmonotonic systems are discussed, together with a number of examples, in Chapter 3.

Our approach to modelling, taking these different types of information into account, is illustrated in Figure 1.3. The model includes a knowledge-base, or set of sentences characterising the dynamic process, and a procedure for generating the predicted behaviour, often referred to as an *inference engine*. The knowledge-base consists of *a priori* facts and causal relationships. The defaults are incorporated as part of the inference engine since their applicability depends on what other information is present or can be inferred. The knowledge-base Δ, the input Ω and the output Ψ are all logical theories.

The knowledge-base and the inference engine together determine the relationship between the input Ω and the output Ψ in an analogous way to the transfer function of a model in a quantitative control system. If we are to use the model as a component in a larger system then we need to know something about the properties of this relationship. In particular, we want to make sure that the model produces a meaningful output for any reasonable set of excitations. We also need an effective procedure for calculating the output. Our approach is aimed at attaining these properties in a nonmonotonic inference system.

1.4 Outline of the Thesis

We begin in the following chapter by surveying some of the influential contributions to nonmonotonic state-based reasoning and providing an example of our early work which motivated the current approach.

In Chapter 3 we suggest a mathematical representation, called a *transfer relation*, which can be used to describe a wide range of reasoning formalisms and compare their characteristic properties. We identify the desirable properties for declarative modelling and describe a number of nonmonotonic formalisms in this context. This serves both to illustrate the properties described and to introduce formalisms referred to in later chapters. Two formalisms in particular, *chronological ignorance* (CI) and *default logic*, form the basis of subsequent work.

Of the formalisms considered, causal theories in CI appear to provide the most promising approach for declarative modelling. Closer examination shows, however, that the CI formalism can be improved in a number of ways. We begin by showing that the use of modal logic in CI is unwarranted. In Chapter 4 we present a truth-functional alternative to modal logic called *asserted logic* (AL) along with a sound and complete proof theory. In Chapter 5 we provide a "temporal" version of AL and use this to define a simplified version of CI for causal reasoning. We then verify that this new framework produces equivalent results.

The following three chapters are chiefly concerned with developing a proof theory for CI based on default logic. In Chapter 6 we provide a new definition of default logic which "factors out" deductive closure. This leads to a proof procedure which can be implemented using a classical theorem prover. We then propose a method for overcoming the problem of *incoherence* in default logic by using asserted logic as the underlying language.

In Chapter 7 we describe a new formalism, called *hierarchical default logic* (HDL), which provides greater control over the application of default information. This approach can be used to overcome the *multiple extension problem* in default logic. We show that the proof procedure for default logic extends to HDL and that this procedure can be simplified significantly for particular types of defaults.

The proof theory for causal CI, which is based on HDL, is presented in Chapter 8. We show that this theory is sound and complete with respect to the semantics of CI. We then discuss the intuitive meaning of causal sentences in CI and argue that the inference mechanism in CI is misleading. We provide four alternative approaches in HDL which, we argue, provide more natural representations. The last approach has the additional advantage that it requires only classical logic rather than modal or asserted logic.

Chapter 9 describes the implementation of the final formalism and provides two examples of assembly tasks which illustrate its use in declarative modelling. Finally in Chapter 10 we summarise the results of the thesis and suggest some promising areas for further research.

Chapter 2

Background Literature and Early Work

In order to model dynamic systems we need to use a logic which enables us to reason about changes over time as well as incorporate default information. That is, we require a nonmonotonic temporal (or state-based) logic. In this chapter we review some of the contributions which led to the development of nonmonotonic temporal reasoning systems. We then outline an early attempt to extend such a system to deal with more difficult problems and discuss how this motivated the approach in this thesis.

2.1 The Situation Calculus

The idea of using a state-based logical framework for reasoning about the dynamic world was proposed by McCarthy in 1963 [McC63]. McCarthy suggests that "human intelligence depends essentially on the fact that we can represent in language facts about our situation, our goals, and the effects of the various actions we can perform". He defines a *situation* to be the complete state of affairs at some instant of time and assumes that the "laws of motion" will determine all future situations from a given situation. Facts about situations were to be stated in an "extended predicate calculus" although the precise form of this calculus is not provided. Instead McCarthy gives examples of the types of information which the calculus should be able to represent. For instance, the fact

$$\mathtt{at(john, home)(s)}$$

(or $\mathtt{at(john, home, s)}$) is intended to mean that $\mathtt{john}$ is at $\mathtt{home}$ in situation $\mathtt{s}$. Similarly the action

$$\mathtt{moves(person, object, location)(s)}$$

is intended to mean that $\mathtt{person}$ moves $\mathtt{object}$ to $\mathtt{location}$ in the situation $\mathtt{s}$. A predicate or function which has a situation as an argument is called a *fluent*.

In order to express causal relationships, McCarthy proposes a second-order predicate (or modal operator) $\mathtt{cause}$ which ranges over fluents. For example,

$$\forall \mathtt{S, P, O, L}\ \ (\mathtt{moves(P, O, L)(S)} \rightarrow \mathtt{cause}(\lambda\, \mathtt{S1}\, \mathtt{at(O, L)(S1)})\mathtt{(S)})$$

is intended to describe the effect of a person P moving an object O to location L.

The ideas proposed by McCarthy were developed into a more complete framework by McCarthy and Hayes [MH69]. This framework, known as the *situation calculus*, was widely adopted and variations on it are still in use today (see for example [GN87]).

The situation calculus uses fluents of the form

$$\mathtt{result}(\mathtt{p},\mathtt{a},\mathtt{s})$$

to express the effects of actions. The `result` function evaluates to the situation which occurs when person `p` has carried out action `a` starting from situation `s`. For example, the formula

$$\mathtt{has}(\mathtt{p},\mathtt{k},\mathtt{s}) \wedge \mathtt{fits}(\mathtt{k},\mathtt{sf}) \wedge \mathtt{at}(\mathtt{p},\mathtt{sf},\mathtt{s}) \rightarrow \mathtt{open}(\mathtt{sf},\mathtt{result}(\mathtt{p},\mathtt{opens}(\mathtt{sf},\mathtt{k}),\mathtt{s}))$$

asserts that if in a situation `s` a person `p` has a key `k` which fits the safe `sf`, then in the situation resulting from `p` performing the action `opens(sf,k)` the safe is open.

This approach has the advantage that the term representing the state after a sequence of actions contains a list of those actions. This list can be used as a plan for execution by an intelligent agent. One of the disadvantages of this type of approach is that the sequence of situations is determined by a sequence of individual actions. Therefore there is no way of expressing actions which overlap or are performed in parallel.

2.2 Question-Answering Systems

The first working implementation of state-based theorem proving is believed to be Green's question-answering system QA3 [Gre69a, Gre69b]. The system uses the resolution principle [Rob65] with an answer literal used to keep track of instantiations.

Green's first approach [Gre69b] was to use axioms of the form

$$\mathtt{p}(\mathtt{s}) \rightarrow \mathtt{q}(\mathtt{f}(\mathtt{s}))$$

where `p` is a predicate describing the initial state, `f` is a function (or action) mapping the initial state to a new state, and `q` is a predicate describing the new state. For example, a problem in which a robot is at a start node `a` and has paths available from nodes `a` to `b` and nodes `b` to `c` could be axiomatised as follows:

$$\begin{array}{lcl} \forall \mathtt{S}\ (\mathtt{at}(\mathtt{a},\mathtt{S}) & \rightarrow & \mathtt{at}(\mathtt{b},\mathtt{move}(\mathtt{a},\mathtt{b},\mathtt{S}))) \\ \forall \mathtt{S}\ (\mathtt{at}(\mathtt{b},\mathtt{S}) & \rightarrow & \mathtt{at}(\mathtt{c},\mathtt{move}(\mathtt{b},\mathtt{c},\mathtt{S}))) \\ & & \mathtt{at}(\mathtt{a},\mathtt{s0}) \end{array}$$

If the robot's goal is to move to node `c` then the question

$$\exists \mathtt{S}\ \mathtt{at}(\mathtt{c},\mathtt{S})$$

is posed to the theorem prover, which returns the answer

$$\mathtt{yes},\quad \mathtt{S} = \mathtt{move}(\mathtt{b},\mathtt{c},\mathtt{move}(\mathtt{a},\mathtt{b},\mathtt{s0}))$$

indicating that the goal can be achieved by moving from `a` to `b` and then from `b` to `c`.

Green later refines this approach [Gre69a] to use axioms of the form

$$\mathtt{p(s)}\rightarrow\mathtt{q(f(a,s))}$$

where `a` represents some action. The function `f`, which is called a *state transformation function*, is a variant on McCarthy and Hayes' `result` function. This revised approach has the advantage that actions appear as arguments and can therefore be quantified over.

2.3 The Frame Problem and STRIPS

In general most facts about the world do not change when an action is performed. One of the problems of a logical system such as QA3 or the situation calculus is that it must include axioms describing all the facts which do not change from state to state as well as those which do. The need to include a large number of axioms just to preserve knowledge between states is referred to as the *frame problem* (see for example [Bro87, GN87, Gre69a, MH69, Nil82]).

The frame problem caused many authors to abandon purely logical approaches in favour of logic-like or hybrid approaches. A well-known example is the SRI problem solver STRIPS [FN71]. In STRIPS each state or *world model* is represented by a set of well-formed formulas. However theorem proving methods are used only to answer questions within a particular world. The effects of actions are encoded by *operators* (or *production rules*[Nil82]). An operator consists of a *precondition*, an *add list* and a *delete list*. It is applicable in some world if its precondition can be deduced from that world. When an operator is applied a new world model is formed by adding and deleting facts according to the add and delete lists. In this way, any facts not mentioned in the add and delete lists are automatically propagated into the new world. To find a suitable sequence of operators STRIPS uses a modification of a technique called *means-ends analysis* developed for the problem solver GPS [EN69].

Due to the difficulties associated with the frame problem, many of the planning and reasoning systems developed in the early 1970's were based on the STRIPS paradigm. A number of influential approaches are discussed by Waldinger [Wal77]. Waldinger also points out a number of inadequacies of the STRIPS approach. For example, all of the indirect side-effects of an action in STRIPS must be stated explicitly in the add and delete lists. Thus if a complex subassembly is moved from one position to another, the add and delete lists must change the position of every component in the subassembly. This difficulty can be avoided in a proof-theoretic system such as Green's since the indirect consequences of an action will follow automatically from a suitable axiomatisation of the problem.

Kowalski [Kow79] argues that a satisfactory treatment of the frame problem can be obtained by using terms rather than atoms to express (nontemporal) statements about the world. These statements are associated with states using a binary relation `Holds`. For example, McCarthy's statement

$$\mathtt{at(john,home,s)}$$

would be expressed in Kowalski's formalism as

$$\texttt{Holds(at(john,home),s)}.$$

The treatment of statements as individuals in this way is often called *reification*.

The advantage of this approach is that we are able to quantify over non-temporal statements. We can therefore express the frame axioms concisely using a statement of the form

$$\forall\, \texttt{A,X,S}\ \ (\texttt{Holds(X,S)} \land \texttt{Preserves(A,X)} \rightarrow \texttt{Holds(X,result(A,S))})$$

where `Preserves(A,X)` expresses the fact that the action `A` preserves the truth of the statement `X`.

This approach requires some extensions to first-order logic since relations such as `Holds` and `Preserves` are meta-level concepts. Also, while the frame axiom is concisely expressed, the frame problem still exists since the `Preserves` clauses must be generated for every appropriate action `A` and statement `X`. Kowalski achieves this using a technique called *macro-processing*.

2.4 Nonmonotonic Reasoning

Nonmonotonic logics emerged in the 1970's in response to the need to be able to make inferences from incomplete knowledge bases. The general idea of nonmonotonic reasoning is to support beliefs which, while not being logically deducible from the knowledge base, are justified in some weaker sense. This often involves some method for specifying "reasonable" candidates and some method for determining whether those candidates can be consistently believed. The term "nonmonotonic" comes from the fact that, unlike systems based purely on deduction, increasing the size of the knowledge base can decrease the number of inferences.

Numerous nonmonotonic systems have been proposed with various objectives in mind. An introduction to some of the better known systems is provided by Genesereth and Nilsson [GN87, Chap 6]. In Chapter 3 we provide a detailed description and comparison of four influential systems—Reiter's *closed world assumption* [Rei78] and *default logic* [Rei80] formalisms, Moore's *autoepistemic logic* [Moo85b] and Shoham's logic of *chronological ignorance* [Sho88a].

The frame problem provided an impetus for incorporating nonmonotonic reasoning in state-based reasoning formalisms. The idea was to include one or more inference rules stating that facts which were true before some action will by default be true after the action unless they are explicitly changed by the action. Returning to Kowalski's frame axiom, for example, rather than have to explicitly generate all the `Preserves` statements, we would be able to say that all action/statement pairs are preserved by default and only explicitly state the exceptions.

This idea was formalised in default logic by Reiter [Rei80] based on a proposal by Sandewall [San72]. Reiter suggests a default schema of the form

$$\frac{r(x,s) : r(x,f(x,s))}{r(x,f(x,s))}$$

which, roughly speaking, says that if any relation r of x is true in state s and it is consistent to believe it is true in the state that results from state transition (or action) f, then it can be concluded that r of x is also true in the resulting state. Note that a schema rather than an axiom must be used to express the frame rule since the first-order framework does not permit quantifying over relations.

A similar approach is advocated by McCarthy [McC86] in which events (or actions) are considered to be *abnormal* with respect to some fact about an individual if they change that fact. A nonmonotonic formalism called *circumscription* [McC80] is used to minimise abnormality and thereby preserve most facts. Hanks and McDermott [HM86] demonstrate that these approaches on their own are not adequate since they are liable to produce unintuitive conclusions.

2.5 Chronological Ignorance

The frame problem has been redefined by Shoham and McDermott in terms of the *qualification problem* and the *extended prediction problem* [SM88]. The qualification problem (previously called the *initiation problem* [Sho86]) is the problem of having to take into account a large (possibly infinite) number of facts about the past in order to make a sound prediction about the future. The extended prediction problem is to do with the length of time into the future about which sound predictions can be made. As a solution to these problems and the problems highlighted by Hanks and McDermott, Shoham proposed a combination of nonmonotonic and temporal reasoning called the *logic of chronological ignorance*(CI)[Sho86, Sho88a, Sho88b].

CI is based on a temporal logic in which primitive propositions are associated with pairs of time points denoted by the integers. The pairs are intended to represent temporal intervals. For example, the formula

$$\textsc{true}(1, 5, \texttt{colour}(\texttt{house17}, \texttt{red}))$$

is intended to mean that the assertion `colour(house17,red)` holds over the interval from time `1` to time `5`. An assertion about a single time point (or state) is expressed using an interval of zero duration.

The temporal logic is extended to an *epistemic* logic (or logic of *knowledge*) by introducing the modal operator $\Box$, interpreted as "known to be true", and its dual $\Diamond$, interpreted as "possibly true". For example, the formula

$$\Box\textsc{true}(2, 2, \texttt{fire\_gun}) \wedge \Diamond\textsc{true}(2, 2, \texttt{firingpin}) \rightarrow \Box\textsc{true}(3, 3, \texttt{noise}) \qquad (2.1)$$

is intended to mean that if it is known that a gun is fired at time `2` and it is possible that there is a firing pin (that is, it is not known that there is not a firing pin) at time `2` then it is known that there is a noise at time `3`. The resulting logic is called a logic of *temporal knowledge* (TK).

Finally, the logic is rendered nonmonotonic by imposing a *preference ordering* on interpretations. The order is chosen so as to minimise what is known, starting at the earliest time point and proceeding in the direction of increasing time (hence the name "chronological ignorance").

The preference ordering can be used to give a sense of unidirectionality to sentences, which is useful for representing causal relationships. In sentence (2.1), for example, the ordering ensures that the truth values of the antecedents will be decided upon before the truth value of the consequent. Thus what is known to be true at later times is determined by what was known to be true (or not known to be false) at earlier times. Shoham defines a class of theories called *causal theories* based on this idea.

Shoham concentrates on the prediction (or modelling) task rather than the plan formation problem. This fact, along with his adherence to well-defined logical semantics, allows Shoham to guarantee certain properties of his system. In particular, it is shown that by restricting the logical descriptions to causal theories, the system is guaranteed to produce a single, consistent prediction. Furthermore, the important information in this prediction can be computed efficiently. This guarantee of predictable behaviour makes the logic an attractive basis for declarative modelling, and much of the work in this thesis is based on Shoham's approach.

Since this work began a number of enhancements and criticisms of Shoham's approach have emerged. Rayner [Ray89] argues that Shoham and McDermott's account of the extended prediction problem [SM88] is unfounded, particularly with regard to continuous dynamic systems. Rayner also points out that CI in its current form is incapable of solving typical problems in these systems.

Bell [Bel91] suggests that propositional causal theories are too limiting for many applications and proposes a first-order extension to the CI framework for causal theories.

Galton [Gal91b] questions the interpretation of Shoham's operator $\Box$ as a *knowledge* operator and suggests that a more accurate reading would be "there is reason to believe...". He argues further that since formulas which include this operator have a different status in the logic to those which do not, the operator does not truly resemble a modal operator.

Bacchus *et al* [BTK89] argue that Shoham's reified temporal logic is overly restrictive and sacrifices classical proof theory unnecessarily. They propose an alternative two-sorted temporal logic, called BTK, which is shown to subsume Shoham's temporal logic. Galton [Gal91a] also argues against reified temporal logics, and in particular the temporal logics of Allen [All84] and Shoham [Sho88b].

2.6 An Example: Multiple Agent Collision Avoidance

In this section we describe some of our early work aimed at extending the chronological ignorance formalism [Sho88a] and applying it to a control problem [MF90c]. This acts both as an introductory example and to illustrate our motivation for developing a simpler CI formalism based on classical proof theory.

The aim in this example is to use a CI based formalism to predict collisions between robotic agents operating in a common workspace, and to modify their plans so that the collisions are avoided. This is an important problem in the implementation of flexible manufacturing systems [RSB85]. Traditional path planning systems (for example [Bro83, LP83, RW87]) are chiefly concerned with single robotic agents in static environments, and attempts to extend these methods to environments containing more than one mobile object are often limited to gross movements in sparse environments [RSB85, RB87] or simplified shapes with few degrees of freedom [FS89].

Our approach is to separate the temporal and spatial components of the problem. We then concentrate on the temporal aspect and develop a reasoning system which predicts possible collisions and uses a local optimisation strategy for avoiding them.

2.6.1 The Collision Avoidance Problem

We assume that a number of robotic agents are operating in a common environment along routes determined by a static path planner (for example [RW87]). We also assume that an intersection detection program (for example [Cam89]) is available which can identify areas in the paths where collisions are possible. The role of the temporal reasoning system is to use information from such programs to predict collisions and alter the agents' plans accordingly.

Static path planners produce *road maps* [RW87] in which nodes connected by directed arcs represent successive agent positions. The arcs may also be labelled with the cost (generally in terms of time) of travel between the nodes. Figure 2.1 shows an example of two road maps in a shared workspace.

As each agent executes its prescribed motion it sweeps a volume or *envelope* in three dimensions. To isolate the possible collision areas we insert new nodes at the positions where these envelopes converge and diverge as illustrated in Figure 2.2. This is called *merging* the road maps. The new arcs bounded by these nodes are called *intersection segments*, and the segments corresponding to an intersection are marked as intersection pairs. In Figure 2.1, for example, the intersection segments are illustrated by bold lines. The original nodes are marked v_1^1 to v_3^1 and v_1^2 to v_6^2. Nodes v_4^1 to v_7^1 and v_7^2 to v_{10}^2 have been added to bound the intersection segments. Note that the intersection of envelopes isolates the possible collision regions without knowledge of the agents' temporal behaviour.

Graphical Representation

The information required by the reasoning system can be conveniently represented using graphs. Given a collection of agents $a_1, a_2, \ldots, a_{n_r}$ each associated with a merged road map, we make the following definitions.

Definition 2.6.1 The *route graph* for an agent $a_r, 1 \leq r \leq n_r$ is a weighted graph $G_r = (V_r, E_r)$ in which each vertex $v_i^r \in V_r,\ \ i = 1, \ldots, n$ represents one of the n nodes in the merged road map for a_r, and edges $(v_i^r, v_j^r) \in E_r$ represent the accessibility of node v_j^r from node v_i^r. Assigned to each edge (v_i^r, v_j^r) is a weighting w_{ij}^r equal to the time taken to travel from v_i^r to v_j^r.

Definition 2.6.2 The *intersection graph* for route graphs $G_1, \ldots, G_{n_r}$ is a graph $G_I = (V_I, E_I)$ in which $V_I = E_1 \cup E_2 \cup \cdots \cup E_{n_r}$ and edges $(v_k^I, v_l^I) \in E_I$ link vertices $v_k^I, v_l^I \in V_I$ which correspond to intersection pairs in the merged road maps.

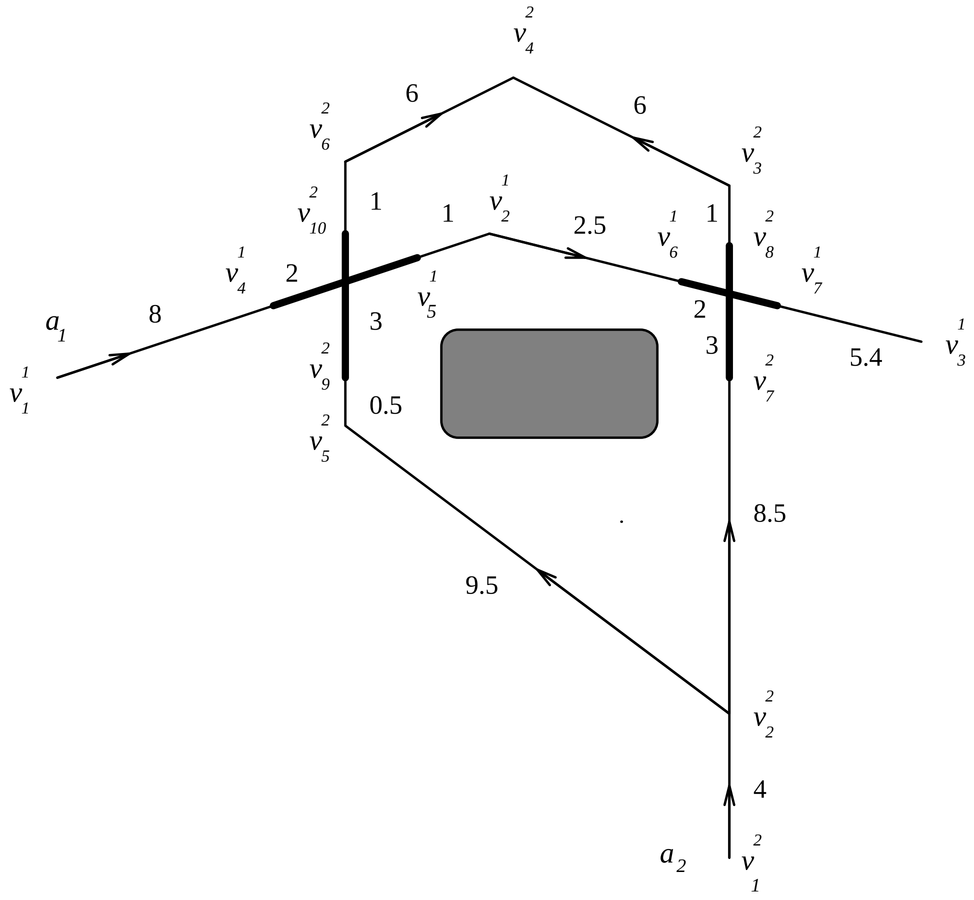

Figure 2.1: Merged road maps for two agents in a common workspace.

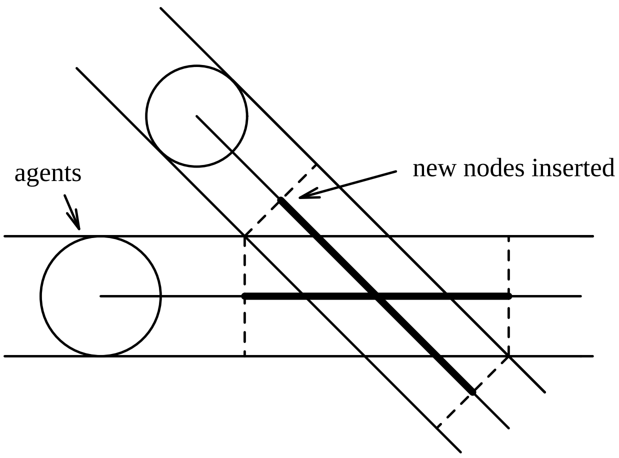

Figure 2.2: New nodes inserted to bound intersection segments.

The intersection graph therefore indicates segments of the road maps which should not be occupied simultaneously.

Route and intersection graphs are simple graphs and can be represented by adjacency matrices.

Definition 2.6.3 Let G_r be a route graph. The adjacency matrix $M_r = [m^r_{ij}]$ of G_r is the $n \times n$ matrix defined by

$$m^r_{ij} = \begin{cases} w^r_{ij}, & \text{if } (v^r_i, v^r_j) \in E_r \\ 0, & \text{otherwise.} \end{cases}$$

Definition 2.6.4 Let $G_I = (V_I, E_I)$ be an intersection graph. If $V_I = \{v^I_1, v^I_2, \ldots, v^I_n\}$, then the adjacency matrix $M_I = [m^I_{kl}]$ of G_I is the $n \times n$ matrix defined by

$$m^I_{kl} = \begin{cases} 1, & \text{if } (v^I_k, v^I_l) \in E_I \\ 0, & \text{otherwise.} \end{cases}$$

This matrix will be symmetric since G_I is an undirected graph.

We show later how these graphs can be used to generate sentences for temporal reasoning. We also provide solutions for the example shown in Figure 2.1. First we introduce our temporal logic and temporal reasoning system.

2.6.2 LCT: A Continuous Temporal Logic

In order to use chronological ignorance to reason about the collision avoidance problem we need to extend the temporal language to overcome a number of restrictions. First, since CI views time as the set of integers it does not permit intervals of arbitrary duration. Secondly, Shoham's assertions over intervals are not homogeneous; that is, if an assertion is true on an interval it does not follow that it is true over subintervals of that interval. For example, if the assertion $\Box\text{TRUE}(1, 5, \texttt{at(robot, position\_1)})$ is true in CI it does not follow that $\Box\text{TRUE}(3, 4, \texttt{at(robot, position\_1)})$ is true. Finally, CI only allows a form of weak negation over temporal intervals. If we assert $\Box\neg\text{TRUE}(1, 5, \texttt{at(robot, position\_1)})$ this means that it is *not* true that, for the complete (nonhomogeneous) interval from time `1` to time `5`, `robot` is at `position_1`. Thus `robot` may be at `position_1` for some subinterval. There is no way in CI of saying that from time `1` to time `5`, `robot` is *not* at `position_1`.

The continuous-time logic LCT is based on Shoham's modal logic of temporal knowledge TK, modified to support continuous time, homogeneous assertions and strong negation. To achieve continuous time we take temporal constants from the set of reals $\Re$ rather than the integers. In order to preserve continuity we only allow intervals which are closed at the left and open at the right. We also allow points only in the antecedents of implication formulas. A sentence $\Box\text{TRUE}(t_1, t_2, p)$ associates proposition p with the interval $[t_1, t_2)$ if $t_1 < t_2$ or the point t_1 if $t_1 = t_2$. We introduce *arithmetic terms*, such as $\texttt{T} - \texttt{8}$, which can be used in place of temporal constants and are evaluated (once the variable `T` is instantiated) according to the standard arithmetic rules. Also, where Shoham allowed primitive propositions p, we allow p or $-p$ where '$-$' is a strong negation connective. As in TK the modal operator $\Box$ is used as a knowledge operator and sentences in the modal language are interpreted using Kripke interpretations. The syntax and semantics of LCT are given in Appendix C.

2.6.3 Projection Theories

Shoham's causal theories allow knowledge of the future to be inferred from knowledge of the past. We extend this idea to define a class of theories in LCT which also allow inferences that change information in the past. We call these theories *projection theories* and construct them as follows.

A *fact* is an atomic formula $\textsc{true}(t_1, t_2, [-]p)$ where $t_1, t_2 \in \Re$, $p \in P$ (a set of primitive propositions) and $t_1 < t_2$. A *known fact* is a sentence of the form $\Box\alpha$ where α is a fact. Known facts are used to store *a priori* knowledge about the system and to describe its state. Knowledge is therefore associated with temporal intervals rather than points, preserving continuity.

A *test literal* is a wff $\Box(\mathtt{T}{-}d_1, \mathtt{T}{-}d_2, [-]p)$ or $\Diamond(\mathtt{T}{-}d_1, \mathtt{T}{-}d_2, [-]p)$ where $\mathtt{T} \in V$ (a set of temporal variables), $p \in P$ and $d_1, d_2 \in \Re$ are constants such that $d_1 \geq d_2 > 0$. d_1 and d_2 are called the maximum and minimum delays respectively. A test literal is further defined as a *point test* if $d_1 = d_2$.

An *assertion literal* is a wff $\Box(\mathtt{T}{+}e_1, \mathtt{T}{+}e_2, [-]p)$ where $\mathtt{T} \in V$, $p \in P$ and $e_1, e_2 \in \Re$ are constants such that $e_1 < e_2$. e_1 and e_2 are called the minimum and maximum extensions respectively. An assertion literal is further defined as an *immediate assertion* if $e_1 = 0$.

A *projection rule* is a sentence of the form

$$\forall \mathtt{T} \quad (\alpha \rightarrow \beta)$$

where $\mathtt{T} \in V$, α is a (non-empty) conjunction of test literals (with no free variables other than $\mathtt{T}$) and β is a (non-empty) conjunction of assertion literals (with no free variables other than $\mathtt{T}$). As all projection rules are universally quantified, we often omit the quantifier.

Projection rules derive new knowledge from what is already known. They are used to project the consequences of earlier knowledge forward, and to update past knowledge based on later results.

A *projection theory* is a collection of known facts and projection rules. As an example of a simple projection theory, consider the following sentences which describe an agent `a` travelling from position `x1` to position `x2` between times `0` and `100`.

$$\Box(\mathtt{20}, \mathtt{100}, \mathtt{reached}(\mathtt{a}, \mathtt{x1}))$$
$$\forall \mathtt{T} \ \ (\Box(\mathtt{T}{-}\mathtt{8}, \mathtt{reached}(\mathtt{a}, \mathtt{x1})) \wedge \Diamond(\mathtt{T}{-}\mathtt{8}, \mathtt{T}, \mathtt{clear}(\mathtt{x1}, \mathtt{x2})) \rightarrow \Box(\mathtt{T}, \mathtt{100}, \mathtt{reached}(\mathtt{a}, \mathtt{x2})))$$

The first sentence is a known fact representing the initial boundary condition that agent `a` has reached position `x1` by time `20`. The second sentence is a projection rule which requires that the agent reaches `x2` eight time units after reaching `x1` if the path is clear, and contains an assumption that the path is clear unless there is information to the contrary.

Figure 2.3 illustrates three of the (infinite number of) interpretations which satisfy this theory. The first two interpretations make assumptions which do not follow from the theory. For example, the first interpretation supports the knowledge that agent `a`

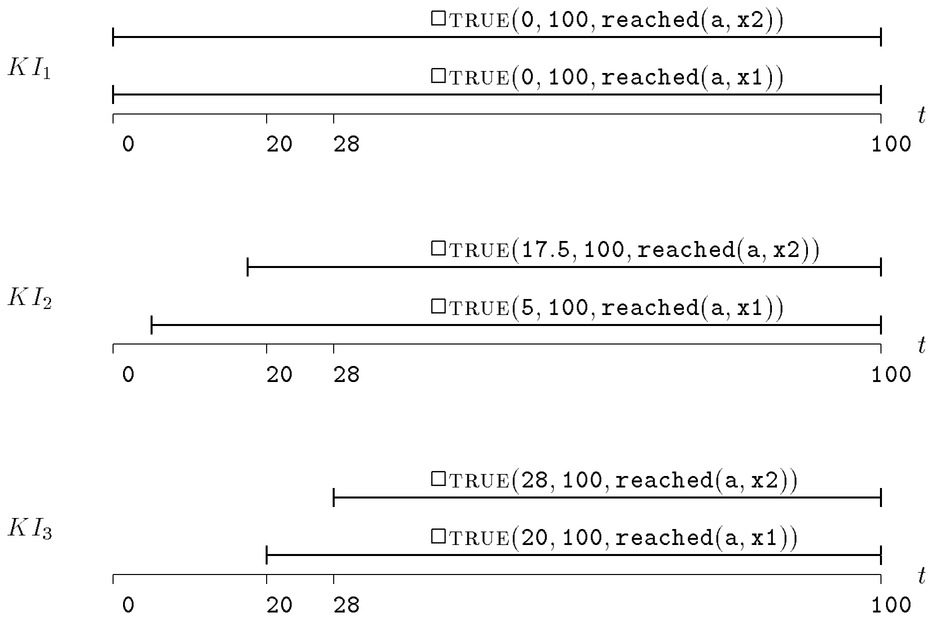

Figure 2.3: Possible interpretations for a projection theory.

had reached `x1` at time `0`. The third interpretation, on the other hand, supports only the knowledge which *must* be true in order to satisfy the theory. This is the *preferred* interpretation; it contains the information we wish to be able to generate.

2.6.4 Chronological Minimisation

Notice that in the third interpretation in Figure 2.3 the known information is asserted for as little time as possible. If, for example, the known fact $\Box$TRUE(28, 100, reached(a, x2)) in KI_3 was "shortened" to $\Box$TRUE(30, 100, reached(a, x2)) the resulting interpretation would no longer satisfy the projection theory (consider for example a variable assignment T = 28.5). This temporal minimisation of knowledge is captured by the CI minimisation strategy.

We impose a partial order on interpretations which is similar to that advocated by Shoham [Sho88a], modified to allow for continuous time. The preferred (or minimal) interpretations under the partial order are those in which known information occurs as late as possible and for the least amount of time.

Definition 2.6.5 A Kripke interpretation KI_2 is *chronologically more ignorant* than a Kripke interpretation KI_1 (written $KI_1 \sqsubset_{ci} KI_2$) in LCT if there exists a time t_0 such that

1. for any (possibly negated) proposition x and time $t < t_0$, if $KI_2 \models \Box(t, x)$ then $KI_1 \models \Box(t, x)$, and

2. there exists some (possibly negated) proposition x such that $KI_1 \models \Box(t_0, x)$ but $KI_2 \not\models \Box(t_0, x)$.

Definition 2.6.6 KI is a *chronologically maximally ignorant* (cmi) interpretation for a projection theory Φ if KI satisfies Φ and there is no interpretation KI' such that KI' satisfies Φ and $KI \sqsubset_{ci} KI'$.

The collection of known facts common to all cmi interpretations describes the predicted behaviour.

Automatic Generation of Known Facts

The set of known facts common to cmi interpretations is built up, from a knowledge base containing the initial conditions, by stepping through contiguous time intervals and adding the consequences of current knowledge. Intuitively, projection rules can be regarded as templates which slide forward along the time axis adding their consequents to the knowledge base when their antecedents are satisfied. With non-causal rules (rules with negative extensions) the knowledge base must be regenerated from the point where the new knowledge was added, since the new knowledge may invalidate rules which have been used previously.

An algorithm for generating the known facts in the cmi models is given in Appendix C. The algorithm takes as arguments a list F of known facts and S of projection rules (from a projection theory Φ) and the times t_0 and t_∞ at which model generation will start and stop respectively, and returns a list of the known facts in the cmi models. The algorithm has been implemented in Lisp.

If the theory Φ includes non-causal rules then we must ensure that the algorithm will terminate (reach t_∞) and not keep looping back into the past. To achieve this we only include non-causal projection rules which 'invalidate themselves'. These rules are of the form $\Diamond(a_1, a_2, x) \wedge \cdots \rightarrow \Box(a_3, a_4, -x) \wedge \cdots$ where $[a_1, a_2) \cap [a_3, a_4) \neq \emptyset$.

2.6.5 Solving the Collision Avoidance Problem

The collision avoidance system must be able to predict when and where collisions will occur and employ some strategy for modifying the agents' plans to avoid them. There are many possible strategies for the latter task. For example, some agents may have automatic priority over others, the priorities may depend on the tasks being performed, or they may vary depending on external factors like the availability of components. Alternatively the strategy may aim for local or global time optimisation or to minimise the completion times of construction tasks. The choice of strategy depends on the application, and may vary in a particular application due to external factors.

An advantage of a declarative system such as the one we have described is that different strategies can be employed simply by changing the data presented to the prediction algorithm. The algorithm itself is independent of the application.

The strategy we employ performs local optimisation (the solutions generated minimise the time lost by two agents arriving randomly at a collision area) and subject to this, For each route graph edge $v_k^I = (v_i^r, v_j^r) \in V_I$:

if there exists no $v_l^I \in V_I$ such that $m_{kl}^I = 1$ **then** form the sentence

$\Box(\mathtt{T}{-}m_{ij}^r, \mathtt{reached}(\mathtt{a\_}r, \mathtt{v\_}r\mathtt{\_}i)) \rightarrow \Box(\mathtt{T}, t_{\max}, \mathtt{reached}(\mathtt{a\_}r, \mathtt{v\_}r\mathtt{\_}j))$

else for each edge $v_l^I = (v_i^{r'}, v_j^{r'}) \in V_I$ such that $m_{kl}^I = 1$ form the sentences

$\Box(\mathtt{T}{-}m_{ij}^r, \mathtt{reached}(\mathtt{a\_}r, \mathtt{v\_}r\mathtt{\_}i)) \wedge \Diamond(\mathtt{T}{-}m_{ij}^r, -\mathtt{reached}(\mathtt{a\_}r, \mathtt{v\_}r\mathtt{\_}j))$

$\wedge\Diamond(\mathtt{T}{-}m_{ij}^r, \mathtt{T}{-}m', \mathtt{clear\_int}(k', l'))$

$\rightarrow\Box(\mathtt{T}{-}m_{ij}^r, \mathtt{T}, -\mathtt{clear\_int}(k', l')) \wedge \Box(\mathtt{T}{-}m_{ij}^r, \mathtt{T}, \mathtt{alloc\_int}(\mathtt{a\_}r, k', l'))$

$\Box(\mathtt{T}{-}m_{ij}^r, \mathtt{T}{-}m', \mathtt{alloc\_int}(\mathtt{a\_}r, k', l')) \rightarrow \Box(\mathtt{T}, t_{\max}, \mathtt{reached}(\mathtt{a\_}r, \mathtt{v\_}r\mathtt{\_}j))$

$$\text{where } m' = \begin{cases} m_{ij}^r, & m_{ij}^r \leq m_{ij}^{r'} \\ \frac{1}{2}(m_{ij}^r + m_{ij}^{r'}), & m_{ij}^r > m_{ij}^{r'} \end{cases}$$

and $k' = \min(k, l), \;\; l' = \max(k, l)$.

Figure 2.4: Automatic generation of projection rules.

models routes which minimise each agent's total travel time. Explanation of the local optimisation strategy and the form of the appropriate projection rules is given in [MF90c].

The required projection rules can be generated directly from the adjacency matrices described earlier. Let $G_1 \ldots, G_{n_r}$ and G_I be route and intersection graphs with associated adjacency matrices $M_1, \ldots, M_{n_r}$ and M_I, and $t_{\max}$ be the maximum time to which model generation is required. The required projection rules are constructed by substituting for the variables (in italics) shown in Figure 2.4.

The Two Agent Example

We now return to the two agent problem in Figure 2.1 and show some examples of the collision free models generated. The projection theory consists of projection rules described above and initial conditions giving the times when the agents begin the routes. Table 2.1, which should be examined in conjunction with Figure 2.1, shows part of the model generated for various initial conditions and explains what this means in terms of the agents' plans. In each case efficient routes are taken while avoiding possible collisions.

2.6.6 Lessons from the Example

While the system described generates successful solutions for this particular example, it is difficult to verify that this will be the case in general. Unless we can prove that our algorithm for generating the consequences of projection theories (Algorithm C.3.1) is sound and (to a lesser extent) complete with respect to our semantics (Figure C.1 and Definitions 2.6.5 and 2.6.6) then the semantics are of little use. They are simply used as a guide for writing the algorithm. In this case, particularly as our algorithm (like Shoham's [Sho88a, Thm 4.8]) makes no use of logical proof theory, it is arguable whether we are using a logic-based system at all.

Proving the soundness and completeness of our algorithm is more difficult than in Shoham's case for a number of reasons. First, we have allowed projection rules in which the consequents have earlier time points than the antecedents. This is necessary because we are using the sentences not only to predict collisions but also to adjust the predictions according to the control strategy. In the system described in this thesis we avoid this problem by adopting the feedback framework shown in Figure 1.1 which explicitly separates the modelling and control functions.

The second problem is that by permitting more complex sentences we have strayed even further from classical logic. Our response to this problem is to return to the original chronological ignorance formalism and investigate ways in which the formalism and the language upon which it is based can be simplified.

Thirdly our algorithm, like Shoham's, makes no use of classical proof theory. This means that the algorithm must do all the deductive work which might otherwise be performed by classical theorem provers. This has motivated us to investigate various proof-theoretic approaches to nonmonotonic reasoning. We use these to develop a proof theory for chronological ignorance based on classical deduction.

Table 2.1: Results generated for the collision avoidance example.

Initial Conditions	**Knowledge Generated**
$\Box(3, 100, \mathtt{reached(a\_2, v\_2\_1)})$	$\Box(18.5, 100, \mathtt{reached(a\_2, v\_2\_8)})$
$\Box(6, 100, \mathtt{reached(a\_1, v\_1\_1)})$	$\Box(19.5, 100, \mathtt{reached(a\_1, v\_1\_6)})$
	$\Box(25.5, 100, \mathtt{reached(a\_2, v\_2\_4)})$
	$\Box(26.9, 100, \mathtt{reached(a\_1, v\_1\_3)})$

The default: no collisions are predicted and the agents reach their goals in the minimum possible time.

$\Box(2.5, 100, \mathtt{reached(a\_2, v\_2\_1)})$	$\Box(15.0, 100, \mathtt{reached(a\_2, v\_2\_7)})$
$\Box(3, 100, \mathtt{reached(a\_1, v\_1\_1)})$	$\Box(16.5, 100, \mathtt{reached(a\_1, v\_1\_6)})$
	$\Box(18.0, 100, \mathtt{reached(a\_2, v\_2\_8)})$
	$\Box(20.0, 100, \mathtt{reached(a\_1, v\_1\_7)})$
	$\Box(25.0, 100, \mathtt{reached(a\_2, v\_2\_4)})$
	$\Box(25.4, 100, \mathtt{reached(a\_1, v\_1\_3)})$

A collision may have occurred since both agents would be crossing the second intersection area at the same time. The local minimisation strategy allocates priority to a_2 and a_1 is delayed by 1.5 time units.

$\Box(3, 100, \mathtt{reached(a\_2, v\_2\_1)})$	$\Box(15.5, 100, \mathtt{reached(a\_2, v\_2\_7)})$
$\Box(2, 100, \mathtt{reached(a\_1, v\_1\_1)})$	$\Box(15.5, 100, \mathtt{reached(a\_1, v\_1\_6)})$
	$\Box(17.5, 100, \mathtt{reached(a\_1, v\_1\_7)})$
	$\Box(20.5, 100, \mathtt{reached(a\_2, v\_2\_8)})$
	$\Box(21, 100, \mathtt{reached(a\_2, v\_2\_6)})$
	$\Box(22.9, 100, \mathtt{reached(a\_1, v\_1\_3)})$
	$\Box(27, 100, \mathtt{reached(a\_2, v\_2\_4)})$

If the path via v_7^2 were the only choice for a_2 both agents would arrive at the second intersection at the same time, a_1 would be given priority, and a_2 would be delayed by two time units. However the longer route via v_6^2 allows a_2 to save 0.5 time units, so the alternative route is used.

Chapter 3

Reasoning Formalisms and Transfer Relations

A large number of nonmonotonic reasoning formalisms have been proposed during the past fifteen years and it is now apparent that there are many similarities between these formalisms. For this reason much of the recent work in nonmonotonic reasoning has been directed towards unifying the various formalisms. One of the obstacles to this process has been the wide range of terminology used and the lack of a common framework for characterising the formalisms.

In this chapter we provide a general framework for describing and comparing reasoning formalisms (both monotonic and nonmonotonic). The framework allows the common characteristics of different formalisms to be categorised in terms of logical and mathematical properties—that is, properties which are not specific to any particular formalism. We define a number of these properties and identify those which are appropriate for declarative modelling. A number of well-known formalisms are then reviewed in this context.

3.1 Transfer Relations

In order to discuss a wide variety of reasoning formalisms an abstract representation is required. We achieve this by viewing all reasoning formalisms as a set of transformations from input descriptions to the inferences or beliefs which they sanction, or from *object descriptions* to *image descriptions*. Each transformation is described by a *pair* consisting of an object description and its image. Since there may be more than one way to augment an incomplete description, there may be more than one transformation for each object description. The set of all such transformation pairs is therefore a binary relation from the object space to the image space. We call this relation a *transfer relation*.

If our object theories are subsets of some language L_1 and the image theories are subsets of some language L_2, then a transfer relation R is a relation from the power set $\wp(L_1)$ to $\wp(L_2)$. If a common language L is used for object and image theories then R is said to be a relation *on* $\wp(L)$. Such a relation is illustrated in Figure 3.1. The domain of a transfer relation R, denoted $\mathcal{D}(R)$, consists of the sets from the object space which appear in the first position of a pair in R. Similarly the range, denoted $\mathcal{R}(R)$, consists of

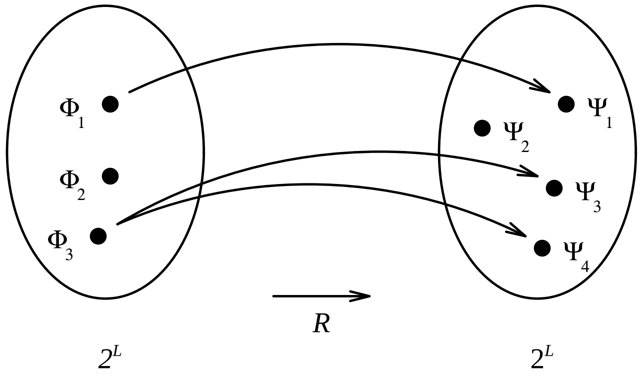

Figure 3.1: A transfer relation $R = \{\langle\Phi_1, \Psi_1\rangle, \langle\Phi_3, \Psi_3\rangle, \langle\Phi_3, \Psi_4\rangle\}$ on $\wp(L)$.

the sets from the image space which appear in the second position of a pair in R. Our conventions for describing relations are given in more detail in Appendix A.

A transfer relation is defined by declaring the object and image spaces and describing which of the object-image pairs are elements of the relation. As an example, consider a standard theorem prover for the propositional language $\mathrm{L_P}$. The beliefs sanctioned by an object theory are simply the theorems, or deductive closure, of the theory. The formalism can therefore be described by a relation Th (read *theorems of*) on $\wp(\mathrm{L_P})$ such that

$$\langle\Phi, \Psi\rangle \in Th \text{ iff } \Psi = \{\psi \mid \Phi \vdash \psi\}.$$

In this example $\mathcal{D}(Th) = \wp(\mathrm{L_P})$ and $\mathcal{R}(Th)$ consists of the deductively closed sets in $\wp(\mathrm{L_P})$.

It is often convenient to distinguish a class of reasoning formalisms and hence transfer relations which differ only in some set of parameters. We use the notation R_P to denote a relation R taking parameter set P.

3.2 Properties of Transfer Relations

We can identify a number of properties of transfer relations which are not specific to any particular reasoning formalism. These properties can be divided into three groups according to their level of abstraction.

3.2.1 Properties of Relations

At the highest level of abstraction we have properties of the relations (or sets of pairs) themselves—we are not concerned what the elements of the pairs consist of.

Partial and Total Relations

We call a relation R a *total relation* if every element of the object space has at least one image. Otherwise it is called a *partial relation*. Thus a relation R from $\wp(L_1)$ to $\wp(L_2)$ is total if and only if $\mathcal{D}(R) = \wp(L_1)$.

Many nonmonotonic formalisms fail to sanction a set of beliefs for some object theories, and are therefore described by partial transfer relations. Total relations are desirable for modelling since we do not want the inference procedure to fail to generate an output description for some inputs.

Transfer Functions and Branching

A relation R is said to be *branching* if some object theories have more than one image and *nonbranching* otherwise. Thus a relation R is nonbranching if and only if $\langle\Phi, \Psi_1\rangle \in R$ and $\langle\Phi, \Psi_2\rangle \in R$ implies $\Psi_1 = \Psi_2$. The *branching factor* of a relation is the greatest number of images designated for any object theory. A nonbranching relation is a *function*. In this case we use the notation $R(\Phi)$ as shorthand for $\{\Psi \mid \langle\Phi, \Psi\rangle \in R\}$.

Branching is a common characteristic of reasoning formalisms which deal with incomplete knowledge since there are usually various ways of filling in the missing knowledge. While some attempts have been made to deal with multiple images, branching is generally seen as a disadvantage of nonmonotonic formalisms. In the case of modelling we wish to avoid branching since we require a single output description.

3.2.2 Properties of Sets

At the next level of abstraction we have properties of sets. In this case we are concerned with the sets which make up the pairs.

Expansion and Contraction

A transfer relation is an *expanding* if each object theory is a subset of its images and *contracting* if each object theory is a superset of its images. Thus R is an expanding relation if $\langle\Phi, \Psi\rangle \in R$ implies $\Phi \subseteq \Psi$ and contracting if $\langle\Phi, \Psi\rangle \in R$ implies $\Psi \subseteq \Phi$.

Most reasoning formalisms are expanding since inferences which are made explicit by the proof procedure are added to the original knowledge. However some formalisms have been proposed which allow contraction as part of belief revision [AGM85]. Performing revision operations directly on the image can be regarded as an alternative to expanding a modified object theory [WF90].

Monotonicity

A function R is *monotonic* if the image of each object theory contains the images of all subsets of that theory; that is, if $\Phi \subseteq \Phi'$ implies $R(\Phi) \subseteq R(\Phi')$. Otherwise it is called *nonmonotonic*. The use of the term "monotonic", adopted from [MD80], is analogous to its use for numerical functions, in which f is monotonically increasing if $x \leq x'$ implies $f(x) \leq f(x')$. Note that $\subseteq$, unlike $\leq$, is a partial order.

3.2.3 Logical Properties

At the lowest level of abstraction we are concerned with the logical properties of the object and image theories.

Consistency and Determinism

A transfer relation is *consistent* (or more accurately *preserves consistency*) if no consistent theory has an inconsistent image; that is, if $\langle \Phi, \Psi \rangle \in R$ and Φ is consistent implies Ψ is consistent.

Some authors argue that commonsense reasoning should be contradiction-tolerant and formalisms have been examined which tolerate inconsistency [Lin87]. In general, however, inconsistency is considered to be a disadvantage since it provides contradictory information.

A transfer relation which is total, nonbranching and consistent provides a single, unambiguous image for all consistent object theories. We call such relations *deterministic*.

Deductive Closure

A transfer relation is *deductively closing* if every image theory specified by the relation is deductively closed. Otherwise it is said to be *nonclosing*. A closing relation can be formed from any transfer relation R on L by the composite relation $Th \circ R$.

From a practical point of view it is often preferable to deal with nonclosing relations rather than closing relations, since the latter expand finite theories into infinite theories.

Completion

We define a theory to be *complete* if it entails every formula or its negation (or possibly both). A transfer relation is *completing* if every image theory is *complete*.

Completion is generally undesirable from a practical point of view due to the large number of formulas required.

We say that a class of relations is total if every member of the class is total, or in other words, if each allowable set of parameters determines a total relation. Similarly for nonbranching, expanding, nonmonotonic, consistent, closing and completing relations.

In the following sections we introduce four well-known nonmonotonic formalisms and discuss their properties.

3.3 The Closed World Assumption

The *closed world assumption* (CWA) was proposed by Reiter [Rei78] as a method for completing knowledge bases. It is one of the earliest nonmonotonic reasoning formalisms used in AI. The idea is simply that if any atomic sentence (or proposition) cannot be deduced from the knowledge base then its negation is added to the knowledge base. The default information used by the CWA is therefore implicit and consists of the set of all negated atomic sentences.

The closed world assumption can be defined as a transfer relation (or function) CWA on $\wp(\mathrm{L_S})$ where

$$CWA(\Phi) = \Phi \cup \{\neg\alpha \mid \alpha \text{ is an atomic sentence and } \Phi \nvdash \alpha\}. \tag{3.1}$$

The relation *CWA* is nonmonotonic since adding any new information to Φ may reduce the number of negated sentences added under the CWA. The relation is also total and nonbranching. The latter property is achieved, however, at the expense of consistency. For example, $CWA(\{\mathtt{at(r,a)} \vee \mathtt{at(r,b)}\})$ includes $\neg\mathtt{at(r,a)}$ and $\neg\mathtt{at(r,b)}$ which along with the object theory is inconsistent. The *CWA* relation is completing since the image theories entail every atomic sentence or its negation and therefore entail every sentence or its negation. It is also expanding and nonclosing. A summary of the properties of the CWA along with other formalisms defined in this chapter is given in Table 3.1.

The inconsistency problem of the CWA can be avoided by restricting the object space. We define *nondisjunctive* theories as follows.

Definition 3.3.1 A theory Φ is *disjunctive* if there exists positive literals $\alpha_1, \ldots, \alpha_n$ such that $\Phi \models \alpha_1 \vee \cdots \vee \alpha_n$ but $\Phi \not\models \alpha_i$, $0 \leq i \leq n$. Otherwise it is *nondisjunctive.*

Consistent nondisjunctive theories are exactly those which have a consistent augmentation under the closed world assumption [She84]. The reason for this is that a nondisjunctive theory augmented by negative literals does not entail any new positive literals, and since each negative literal added under the CWA is consistent with the original theory, no contradictions can be introduced. Nondisjunctive theories are considered further in Section 6.4.

If we let $cnd(\wp(\mathrm{L_S}))$ be the set of consistent nondisjunctive theories in $\wp(\mathrm{L_S})$ then the relation *CWA* defined from $cnd(\wp(\mathrm{L_S}))$ to $\wp(\mathrm{L_S})$ according to (3.1) preserves consistency.

The CWA is not a very versatile form of nonmonotonic reasoning because of the limited control over default information. Only negative information can be added and it cannot be added selectively.

3.4 Default Logic

Default logic is a nonmonotonic formalism proposed by Reiter [Rei80] in which default information is incorporated using additional rules of inference. A *default* is an inference rule of the form

$$\frac{\alpha : \beta_1, \ldots, \beta_m}{\gamma}$$

(also written $\alpha : \beta_1, \ldots, \beta_m \,/\, \gamma$) where the formulas α, $\beta_1 \ldots \beta_m$ and γ are known respectively as the prerequisite, justifications and consequent of the default. If D is a set of defaults then $\textsc{consequents}(D)$ is the set $\{\gamma \mid (\alpha : \beta_1, \ldots, \beta_m \,/\, \gamma) \in D\}$.

A default is *closed* if it contains only closed formulas and *open* otherwise. We restrict our attention to closed defaults. An open default can be treated as a schema for the set of closed defaults which are its substitution instances [Rei80, Sec 7]. The results for closed defaults can be extended to open defaults providing a countably infinite set of defaults is allowed [Kon88b].

We also restrict our attention to defaults with single justifications since this simplifies the discussion and it is believed that single justifications are sufficient for practical reasoning tasks (see for example [Eth87]). The results can be extended to include multiple justifications in a straightforward way.

Table 3.1: A summary of properties for common reasoning formalisms.

	Nmon	**Tot**	**Nbra**	**Con**	**Nclo**	**Exp**	**NComp**
Deductive Closure							
Th		√	√	√		√	√
Closed World Ass.							
CWA	√	√	√		√	√	
Default Logic							
E_D	√			√		√	√
E_D (normal defaults)	√	√		√		√	√
AE Logic							
E_{wg}, E_{mg}, E_{sg}	√			√		√	√
CI							
K_{TK}	√	√	√	√	√		√

Key			
Nmon	Nonmonotonic	**Nclo**	Nonclosing
Tot	Total	**Exp**	Expanding
Nbra	Nonbranching	**NComp**	Noncompleting
Con	Consistent		

Defaults in which the justification entails the consequent are called *seminormal defaults* and are normally written in the form $\alpha : \beta \wedge \gamma \,/\, \gamma$. Defaults in which the consequent also entails the justification are called *normal defaults* and are generally written in the form $\alpha\!: \gamma \,/\, \gamma$. Normal defaults in which the prerequisite is empty are called *free defaults*.

The literature on default logic focuses on pairs $\langle D, \Phi\rangle$, called *default theories*, where D is a set of defaults and Φ is a set of sentences called the *underlying theory*. The beliefs sanctioned by such a theory are called an *extension* of the default theory.

We take a slightly different view and consider defaults as parameters to a transfer relation. Extensions are then images under the relation. From this viewpoint the *extension relation* can be defined in terms of Reiter's original characterisation [Rei80, Def 1] as follows.

Definition 3.4.1 Let D be a set of (closed) defaults and $\mathrm{L_S}$ be a closed first-order language. The *extension relation* E_D is a relation on $\wp(\mathrm{L_S})$ such that $\langle \Phi, \Psi\rangle \in E_D$ if and only if

$$\Psi = \Gamma \tag{3.2a}$$

where Γ is a minimal set satisfying

$$\Phi \subseteq \Gamma \tag{3.2b}$$

$$Th(\Gamma) = \Gamma \tag{3.2c}$$

$$\text{if } (\alpha : \beta \,/\, \gamma) \in D,\ \alpha \in \Gamma \text{ and } \neg\beta \notin \Psi \text{ then } \gamma \in \Gamma \tag{3.2d}$$

Like Reiter's characterisation Definition 3.4.1 is *self-referential* since the elements of Γ and hence the extensions Ψ are defined in terms of Γ and Ψ. In other words the applicability of defaults depends on which defaults are applied. Sets Ψ which satisfy conditions (3.2a)–(3.2d) for some theory Φ and default set D are called *fixed points*.

For some object theories and default sets there are no fixed points satisfying (3.2a)–(3.2d) while for others there are many. Thus the extension relation is in general partial and branching. The former problem is often referred to as *incoherence* [Eth87] and the latter referred to as the *multiple extension problem* (MEP).

Total relations can be ensured by restricting the parameter set to normal defaults [Rei80, Thm 3.1], or by restricting the theories and default sets to those which are ordered in the sense defined by Etherington [Eth87]. Both of these solutions restrict the representational scope of defaults, and we propose an alternative approach in Chapter 6. The branching problem, or MEP, is considered further in Chapter 7.

The extension relation preserves consistency [Rei80, Cor 2.2], is noncompleting, and is expanding due to (3.2b). It is also closing due to condition (3.2c). We provide a nonclosing alternative in Chapter 6.

3.5 Autoepistemic Logic

Autoepistemic logic (AEL) was proposed by Moore [Moo85b] as a solution to problems encountered with the nonmonotonic logics of McDermott and Doyle [McD82, MD80]. It is intended to be a logic in which an intelligent agent can reason about its own beliefs (hence the name "autoepistemic"). The formalism has been further investigated and clarified by Konolige [Kon88b]. Our definitions are adapted from [Kon88b].

The symbols of AEL are those of a standard first-order language, L_S, along with the modal *self-belief* operator which we denote $\Box$. The sentences of AEL are those of L_S plus those generated by the rule:

if ϕ is a sentence of AEL, then $\Box\phi$ is a wff of AEL.

Note that the argument of the modal operator never contains free variables and therefore there is no quantifying into a modal context.

Sentences of L_S are called *ordinary sentences*. If Φ is an AEL theory then we write Φ_0 to denote the ordinary sentences in Φ; that is $\Phi_0 = \Phi \cap L_S$. Sentences of the form $\Box\phi$ are called *modal atoms*.

A *valuation* on AEL is a pair $\langle\sigma, \Gamma\rangle$ consisting of a standard first-order valuation σ, and an AEL theory Γ. Ordinary atoms are evaluated with respect to σ in the usual way. Modal atoms are evaluated according to the rule:

$$(\Box\phi)^{\langle\sigma,\Gamma\rangle} = t \text{ if and only if } \phi \in \Gamma.$$

The notions of satisfaction and logical consequence are extended to incorporate AEL valuations in the obvious way.

An AEL theory Γ is said to be a *stable set* if it satisfies:

$$\Gamma \text{ is closed under logical consequence} \qquad (3.3a)$$

$$\text{if } \phi \in \Gamma \text{ then } \Box\phi \in \Gamma \tag{3.3b}$$

$$\text{if } \phi \notin \Gamma \text{ then } \neg\Box\phi \in \Gamma \tag{3.3c}$$

The symbol $\models_{\mathrm{ss}}$ is used to indicate logical consequence restricted to the valuations $\langle\sigma, \Gamma\rangle$ in which Γ is a stable set. Note that the definition of stable sets is self-referential; the elements of Γ are defined in terms of Γ itself. Equations (3.3a)–(3.3c) define fixed points for the set Γ.

A sentence is said to be in *normal form* if it is in the form

$$\neg\Box\alpha \vee \Box\beta_1 \vee \ldots \vee \Box\beta_n \vee \omega \tag{3.4}$$

where α, β_i and ω are ordinary sentences. Any of the disjuncts other than ω may be absent. Konolige [Kon88b, Prop 3.9] shows that any AEL theory is equivalent to a theory in normal form.

Finally, for any theory Φ we let $\Box\Phi$ denote the set of formulas $\{\Box\phi \mid \phi \in \Phi\}$ and $\neg\Box\overline{\Phi}$ denote the set of formulas $\{\neg\Box\phi \mid \phi \notin \Phi\}$.

Konolige's *weakly grounded extensions* (E_{wg}), *moderately grounded extensions* (E_{mg}) and *strongly grounded extensions* (E_{sg}) can now be defined as transfer relations on $\wp(\mathrm{AEL})$ as follows:

Definition 3.5.1 *For all* AEL *theories* Φ *and* Ψ*:*

$$\langle\Phi, \Psi\rangle \in E_{\mathrm{wg}} \quad \textit{if and only if} \quad \Psi = \{\alpha \mid \Phi \cup \Box\Psi_0 \cup \neg\Box\overline{\Psi}_0 \models_{\mathrm{ss}} \alpha\} \tag{3.5}$$

$$\langle\Phi, \Psi\rangle \in E_{\mathrm{mg}} \quad \textit{if and only if} \quad \Psi = \{\alpha \mid \Phi \cup \Box\Phi \cup \neg\Box\overline{\Psi}_0 \models_{\mathrm{ss}} \alpha\} \tag{3.6}$$

Definition 3.5.2 *For all* AEL *theories* Φ *and* Ψ*, where* Φ *is in normal form,*

$$\langle\Phi, \Psi\rangle \in E_{\mathrm{sg}} \textit{ if and only if } \langle\Phi, \Psi\rangle \in E_{\mathrm{wg}} \textit{ and } \Psi = \{\alpha \mid \Phi' \cup \Box\Phi' \cup \neg\Box\overline{\Psi}_0 \models_{\mathrm{ss}} \alpha\} \tag{3.7}$$

where Φ' *is the set of sentences* $\neg\Box\alpha \vee \Box\beta_1 \vee \cdots \vee \Box\beta_n \vee \omega$ *of* Φ *such that none of* $\beta_1, \ldots, \beta_n$ *is contained in* Ψ*.*

All moderately grounded extensions are weakly grounded [Kon88b, Prop 2.8] and all strongly grounded extensions are moderately grounded [Kon88b, Prop 3.10]. Thus $E_{\mathrm{sg}} \subseteq E_{\mathrm{mg}} \subseteq E_{\mathrm{wg}}$.

Weakly grounded extensions (and hence moderately and strongly grounded extensions) are stable sets [Moo85b]. The extension relations are therefore expanding, closing and consistent. In general the relations are partial and branching since there may be zero, one or many fixed points for equations (3.3a)–(3.3c).

Hierarchic autoepistemic logic (HAEL) [Kon88a] is a variation on AEL in which the object theories are replaced by well-founded linearly ordered sets of subtheories. The resulting formalism, like the CWA, avoids branching at the expense of consistency. HAEL, along with Konolige's equivalence result for default logic and AEL, is considered further in Section 7.5.

3.6 Chronological Ignorance

The chronological ignorance formalism [Sho88a] was introduced in Section 2.5. The formalism differs from many nonmonotonic systems in that it is defined from a model-theoretic rather than proof-theoretic standpoint. Instead of specifying image theories directly from the (syntax of) the object theory, CI specifies preferred models (Kripke interpretation/world pairs) for the object theory. The image theories can be considered to be the formulas satisfied by the preferred models. Formalisms defined in this way are often referred to as *model preference logics*.

As outlined in Section 2.5, CI is defined in the modal logic TK in which primitive propositions are reified over pairs of integers representing time intervals. Although TK allows temporal variables and temporal relations these are not allowed in causal theories. The theories are constructed from a propositional part of TK whose well-formed formulas can be defined as follows:

1. If α is an atomic formula of $\mathrm{L_P}$ and $t_1, t_2 \in \mathbb{Z}$ then $\textsc{true}(t_1, t_2, \alpha)$ is a wff.

2. If ϕ_1 and ϕ_2 are wffs, then so are $\neg\phi_1$, $\phi_1 \rightarrow \phi_2$ and $\Box\phi_1$.

The connectives $\wedge$, $\vee$, $\leftrightarrow$ and the operator $\Diamond$ are defined in the usual way. A *base sentence* is a sentence which does not contain the modal operator. The *latest time point* (*ltp*) of a $\mathrm{TK_P}$ sentence is the largest of the (temporal) integers appearing in the sentence.

The nonmonotonicity of CI comes from imposing a partial order on models for TK theories.

Definition 3.6.1 [Sho88a, Def 3.4] A model M_2 is *chronologically more ignorant* than a model M_1 (written $M_1 \sqsubset_{\mathrm{ci}} M_2$) if there exists a time point t_0 such that
(1) for any base sentence ϕ whose *ltp* $\leq t_0$, if $M_2 \models \Box\phi$ then $M_1 \models \Box\phi$ and
(2) there exists some base sentence ϕ whose *ltp* is t_0 such that $M_1 \models \Box\phi$ but $M_2 \not\models \Box\phi$.

Definition 3.6.2 [Sho88a, Def 3.5] M is said to be a *chronologically maximally ignorant* (or *cmi*) model of ϕ if $M \models \phi$ and there is no other M' such that $M' \models \phi$ and $M \sqsubset_{\mathrm{ci}} M'$.

We will often refer to this strategy for specifying preferred models as *chronological minimisation* since it minimises assignments of t to sentences (in this case known base sentences) over time.

No method is provided for generating the formulas which are satisfied by the cmi models of arbitrary object theories. Instead Shoham defines a class of theories whose cmi models have desirable properties. These theories are called causal theories and are defined as follows.

A *causal statement* is a sentence of the form

$$\Box\alpha_1 \wedge \ldots \wedge \Box\alpha_m \wedge \Diamond\beta_1 \wedge \ldots \wedge \Diamond\beta_n \rightarrow \Box\gamma$$

where $\alpha_1, \ldots, \alpha_m, \beta_1, \ldots, \beta_n$ and γ are (possibly negated) atomic base sentences and the *ltp* of the consequent is greater than the *ltp*'s of the antecedents. If $\alpha_1, \ldots, \alpha_m$ are missing (that is, identically true) then the sentence is called a *boundary condition*, otherwise it is called a *causal rule*.

Definition 3.6.3 A *causal theory* Φ is a set of causal statements such that:

(1) There is a time point t_0 such that if $\phi \rightarrow \Box\gamma$ is a boundary condition in Φ and the *ltp* of γ is t_1, then $t_0 < t_1$.

(2) There do not exist two sentences in Φ such that one includes $\Diamond\textsc{true}(t_1, t_2, p)$ on its left-hand side and the other includes $\Diamond\neg\textsc{true}(t_1, t_2, p)$ on its left-hand side for any p, t_1 and t_2.

(3) If $\phi_1 \rightarrow \Box\textsc{true}(t_1, t_2, p)$ and $\phi_2 \rightarrow \Box\neg\textsc{true}(t_1, t_2, p)$ are two sentences in Φ then $\{\phi_1\} \cup \{\phi_2\}$ is inconsistent.

We let *causal*($\wp$(TK)) denote the causal theories of TK and define the images under CI as follows.

Definition 3.6.4 K_{TK} is a relation from *causal*($\wp$(TK)) to $\wp$(TK) such that $\langle \Phi, \Psi \rangle \in K_{\mathrm{TK}}$ if and only if

$$\Psi = \{\Box\alpha \mid \alpha \text{ is an atomic base literal and } M \models \Box\alpha \text{ for all cmi models } M \text{ of } \Phi\}.$$

Shoham [Sho88a, Thm 4.4] shows that there is a unique consistent set of known base literals (and their tautological consequences) which is common to the cmi models of a causal theory. The relation K_{TK} is therefore total, nonbranching and consistent. It is also nonclosing, and for finite object theories the set of known base literals is finite. An algorithm for generating these literals is discussed in Chapter 5. The tautological consequences of the known base literals are given by the closing relation $Th \circ K_{\mathrm{TK}}$.

Since the relation K_{TK} is deterministic and nonclosing, it makes an attractive basis for a nonmonotonic modelling system. However, as we suggested in Chapter 2, the complex semantics and lack of a proof theory make it difficult to modify or extend the formalism. The following chapters are concerned with simplifying the semantics of the CI formalism and developing an appropriate proof theory.

Chapter 4

Asserted Logic

The logic of chronological ignorance is a nonmonotonic version of Shoham's modal logic of *temporal knowledge* (TK) [Sho88a]. The "knowledge" part comes from an interpretation of the modal operator $\Box$ as asserting that its argument is *known* to be true and its dual $\Diamond$ as asserting that its argument is *possibly* true. In this chapter we argue that the utility of this logic can be achieved without resorting to modal logic. We provide an alternative logic, called *asserted logic* [MF90a], which provides the benefits of Shoham's modal logic without sacrificing truth functionality and classical proof-theory. The equivalence of the logics with regard to chronological ignorance is verified in the following chapter.

4.1 Modal Logics of Knowledge

The use of modal logic to represent knowledge and possibility in nonmonotonic systems follows attempts to incorporate concepts such as consistency into an object language [McD82, MD80]. The idea is to sanction weak inferences of the form $\Diamond\alpha$ (read "α is possible") when an assertion α is consistent with the knowledge base—that is when $\neg\alpha$ cannot be proven. This general principle is followed in CI by giving preference to interpretations in which the assertions which are *known* are minimised. Thus if p is a proposition and $\Box\neg p$ cannot be proven, precedence is given to interpretations in which $\neg\Box\neg p$ (or $\Diamond p$) is true.

Modal logic has been adopted for this task with little supporting motivation, and it is worth considering whether it is the appropriate choice of language. The shift from classical to modal logic sacrifices truth functionality [Qui76]—for example, $\Box\alpha$ and $\Box\beta$ may have different truth values even though α and β have the same truth value—and complicates the semantics, axiomatisation and proof theory of the logic. Interpretations for modal logic require a family of structures or *possible worlds* related by an accessibility relation [HC68, Kri63]). Different accessibility relations correspond to different modal systems and axiomatisations, and classical proof theory is no longer adequate (see for example [Wal90]).

In return for these sacrifices modal logic allows a great deal of freedom in the construction of formulas. The scope of the modal operator can include atomic formulas, compound formulas, and formulas which themselves include the modal operator. One test of the ap-

Table 4.1: The confidence spectrum for modal logic.

Informal Interpretation	Formal Interpretation		
	α	$\Box\alpha$	$\Box\neg\alpha$
known to be true	true in all worlds	t	f
possibly true	true in some world	–	f
nothing known	true in some worlds, false in others	f	f
possibly false	false in some world	f	–
known to be false	false in all worlds	f	t

propriateness of modal logic is whether these compound and nested forms have practical applications. For example, while it seems natural to interpret $\Box p \vee \Box q$ as asserting that p is known or q is known, it is less clear why we would wish to be able to use $\Box(p \vee q)$ (in each possible world p or q is true, but neither p nor q need be known) or $\Diamond\Diamond p$ (p is possibly possibly true) in this context.

This issue is taken further by Galton [Gal91a] who argues that from a representational point of view Shoham's use of $\Box$ does not resemble a modal operator, since the formulas $\Box\alpha$ and α have different statuses in the logic. Galton suggests that a more accurate reading of Shoham's operator would be "There is reason to believe that…".

4.2 The "Confidence" Spectrum

The one respect in which Shoham's operator does resemble a modal operator, according to Galton, is that it does not commute with negation. That is, $\Box\neg\alpha$ is in general not equivalent to $\neg\Box\alpha$. Another way of saying this is that $\Box\alpha \vee \Box\neg\alpha$ is not tautologically true. That is, the logic can be considered to escape the law of excluded middle at what might be called the *epistemic* level—it is not necessary for a formula to be either known to be true, or known to be false. The result is that the modal operator and negation connective can be used to obtain the spectrum of interpretations shown in Table 4.1, which we call a *confidence spectrum*. With this spectrum available, a minimisation strategy is no longer forced to commit assertions to one extreme or the other, but can make use of the "middle ground". This is exactly what is done by the CI minimisation strategy: if an assertion is not forced to take the values "known to be true" or "known to be false" then it is assigned the value "nothing known". In fact this is the *only* benefit of modality in chronological ignorance.

If we only require knowledge of atomic formulas, as is the case in CI, then this spectrum can be achieved using classical logic. To illustrate this point consider a classical propositional language. A simple language of knowledge, which we will call $\mathrm{K_P}$, can be constructed by replacing each proposition p by the two propositions formed by prefixing p with `known_` and `known_not_` (we will use *known_p* and *known_not_p* as meta-variables representing the resulting propositions). An assignment of t to *known_p* (respectively *known_not_p*) is intended to mean that the original proposition p is known to be true

Table 4.2: The confidence spectrum for K_P.

Informal Interpretation	Formal Interpretation	
	known_p	*known_not_p*
known to be true	t	f
possibly true	–	f
nothing known	f	f
possibly false	f	–
known to be false	f	t

(respectively false). To complete the formalism we add the formula

$$\neg(known\_p \wedge known\_not\_p) \tag{4.1}$$

for each pair of propositions to ensure that p cannot be known to be true and false simultaneously. The resulting confidence spectrum, which is shown in Table 4.2, is identical to the spectrum for modal logic. In this case, however, the formal interpretation is simply a classical truth valuation.

Chronological ignorance can be redefined using a temporal version of the classical logic K_P, although the representation is somewhat unwieldy. The intended predicate symbols must be altered and formulas in the form of (4.1) must be explicitly added for all atomic formulas. What we would like instead is a language which has the convenient representation found in Shoham's TK without sacrificing the truth functional semantics and proof theory of K_P. Our solution is a language called asserted logic (AL). Asserted logic replaces the '`known_`' part of K_P formulas by a truth functional operator and the '`not_`' part by a strong negation connective. The collection of formulas of the form (4.1) can then be replaced by an axiom schema which is incorporated in the logic.

4.3 Definition of Asserted Logic

We begin by defining the syntax and semantics of asserted logic and then discuss its interpretation. Only the propositional semantics is provided since this is sufficient for a discussion of Shoham's causal theories. A first-order semantics for AL is suggested in Appendix B.

4.3.1 Syntax of AL

Let L be a classical first order language in which the connectives $\neg$ and $\rightarrow$ and the quantifier $\forall$ are taken as primitive (see Appendix A). The symbols of AL are the symbols of L, a (3-valued negation) connective '$-$' and the (assertion) operator $\mathbf{T}$.

A new construction, called a *base formula*, is defined as follows:

1. If p is an atomic formula of L then p is a base formula of AL.

2. If x is a base formula of AL, then $-x$ is a base formula of AL.

If p is an atomic formula of L, then p is called an *atomic base formula* of AL and p and $-p$ are called *base literals*. We will refer to base formulas with no free variables as *base sentences*, and to variable-free atomic base formulas as *propositions*.

The well-formed formulas (wffs) of AL are defined as follows:

1. If x is a base formula then $\mathbf{T}x$ is a wff.

2. If α is a wff then $\neg\alpha$ is a wff.

3. If α and β are wffs then $\alpha \rightarrow \beta$ is a wff.

4. If α is a wff and v is a variable then $\forall v\alpha$ is a wff.

The connectives $\wedge$, $\vee$ and $\leftrightarrow$ and the existential quantifier $\exists$ are defined in terms of $\neg$, $\rightarrow$ and $\forall$ in the usual way, and parentheses are used for clarity where necessary (see Appendix A). Formulas constructed according to rules (1), (2), (3) and (4) are called *atomic*, *negation*, *implication* and *universal* formulas respectively. A *literal* is an atomic formula or a negated atomic formula.

It is important to note that base formulas are not well-formed—they only appear in a theory within the scope of the assertion operator (hence the name *asserted logic*). Thus $\mathbf{T}p$ and $\mathbf{T}{-p} \rightarrow \neg\mathbf{T}q$ (where p and q are base formulas) are examples of wffs while p, $\neg p \wedge \mathbf{T}p$, $\mathbf{T}(p \rightarrow q)$ and $\mathbf{TT}p$ are not wffs.

Connectives which are outside the scope of the assertion operator (the standard connectives) are called *external connectives*. Those within its scope (in this case only 3-valued negation) are called *internal connectives*.

We say that a formula α is a *(propositional) combination* of formulas $\beta_1, \ldots, \beta_k$ if it can be compounded from them using $\neg$ and $\rightarrow$ according to rules (2) and (3). The *degree of complexity* of a formula α (written deg α) is the sum obtained by adding 2 for each occurrence of $\rightarrow$ and 1 for each occurrence of $\neg$ in α.

As with the classical language L, we may sometimes wish to restrict our attention to the propositional part of AL. We let $\mathrm{AL_P}$ denote the wffs of AL that can be formed using only individual constants, extralogical predicate symbols, the (internal and external) logical connectives and the assertion operator.

4.3.2 Propositional Semantics of AL

AL differs from the classical language L in that statements involving a single predicate, such as `at(john,home,s)`, can be assigned one of three truth values; t (true), f (false) or u (unknown). A 3-valued negation connective '$-$' is also provided. The connective is defined by the following truth table:

y	$-y$
t	f
u	u
f	t

The need for 3-valued connectives other than negation is avoided, however, by encasing 3-valued statements (or base formulas) within the scope of Bochvar's *assertion operator*

T [Haa78]. The assertion operator maps the three truth values t, f and u onto the two traditional values t and f according to the following truth table:

y	$\mathbf{T}y$
t	t
u	f
f	f

Note that the connective $-$ and the operator **T** are truth functional. If base formulas x and y have the same truth value, then $-x$ and $-y$ have the same value as do $\mathbf{T}x$ and $\mathbf{T}y$. Therefore, as in classical logic, truth tables can be used to check the validity of AL formulas.

Whereas classical truth-valuations make assignments to well-formed formulas subject only to the constraints imposed by the logical connectives (see Appendix A), truth-valuations on AL must assign values to both wffs and base formulas subject to the added constraints imposed by the operator **T** and the connective $-$.

Definition 4.3.1 A *truth valuation* on AL is a mapping σ assigning to each base formula x a value x^σ from the set $\{t, f, u\}$ and to each wff α a value α^σ from the set $\{t, f\}$, such that for all base formulas y and wffs β and γ

1. $(-y)^\sigma = t$ iff $y^\sigma = f$, and $(-y)^\sigma = f$ iff $y^\sigma = t$,
2. $(\mathbf{T}y)^\sigma = t$ iff $y^\sigma = t$,
3. $(\neg\beta)^\sigma = t$ iff $\beta^\sigma = f$,
4. $(\beta{\rightarrow}\gamma)^\sigma = t$ iff $\beta^\sigma = f$ or $\gamma^\sigma = t$.

Conditions (1) and (2) define the properties of the assertion operator and negation connective according to the above truth tables. Conditions (3) and (4) are the standard constraints on truth valuations for the connectives $\neg$ and $\rightarrow$.

We call a literal $\mathbf{T}x$ or $\mathbf{T}{-x}$ a *strong assertion* since satisfying the literal requires that x take a particular truth value. A literal $\neg\mathbf{T}x$ or $\neg\mathbf{T}{-x}$ is called a *weak assertion* since satisfying it constrains the allowable truth values, but does not force one particular value. Similarly we call the connectives $-$ and $\neg$ strong and weak negation respectively.

If α^σ is fixed arbitrarily for all atomic and universal formulas α, then (as with classical logic) conditions (3) and (4) define β^σ for *all* wffs β. Similarly, if x^σ is fixed arbitrarily for all atomic base formulas x, then conditions (1) and (2) define α^σ for all atomic formulas α. Thus, a mapping of the atomic base formulas onto $\{t, f, u\}$ and the universal formulas onto $\{t, f\}$ can be extended in a unique way by conditions (1)–(4) into a truth valuation.

4.3.3 Satisfaction and Tautological Consequence

As in classical logic a truth valuation σ on AL *satisfies* a set Φ of wffs (written $\sigma \models \Phi$) if $\phi^\sigma = t$ for every formula $\phi \in \Phi$.

We say ϕ is a *tautology* if $\sigma \models \phi$ for every truth valuation σ. It is always possible to check whether a formula ϕ is a tautology in a finite number of steps by constructing a truth table for ϕ in terms of its universal and atomic base formulas.

Theorem 4.3.2 *Let α be a tautology of* L. *Then a wff α' obtained by replacing each atomic subformula of α with* any *atomic formula of* AL *is a tautology of* AL.

Proof. Clearly any truth assignment to the universal and atomic formulas of α' satisfies α' since the connectives of L and AL have the same meaning. Any assignment of truth values to the universal and atomic base formulas of α' must induce one of these truth assignments. $\square$

Thus the tautological schemata of classical logic also represent tautologies in AL. The converse, however, is false. For example,

$$\neg(\mathbf{T}\mathtt{p} \wedge \mathbf{T}{-}\mathtt{p})$$

is a tautology in AL which is not a tautology when the atomic formulas are replaced by those from classical logic.

Tautological consequence and equivalence are defined as for classical logic (see Appendix A).

Lemma 4.3.3 *A wff α is a tautological consequence of a set of wffs $\{\phi_1, \ldots, \phi_k\}$ iff*

$$\phi_1 \rightarrow \phi_2 \rightarrow \ldots \rightarrow \phi_k \rightarrow \alpha$$

is a tautology.

Proof. Let $\{\phi_1, \ldots, \phi_k\} \models \alpha$. For each truth valuation σ such that $\sigma \models \{\phi_1, \ldots, \phi_k\}$ clearly $(\phi_1 \rightarrow \phi_2 \rightarrow \ldots \rightarrow \phi_k \rightarrow \alpha)^\sigma = t$ since $\phi_1^\sigma = \ldots = \phi_k^\sigma = \alpha^\sigma = t$. For each truth valuation σ such that $\sigma \not\models \{\phi_1, \ldots, \phi_k\}$ there exists a minimum j, $1 \leq j \leq k$ such that $\phi_j^\sigma = f$. Then $(\phi_j \rightarrow \ldots \rightarrow \phi_k \rightarrow \alpha)^\sigma = t$ and hence $(\phi_1 \rightarrow \ldots \rightarrow (\phi_j \rightarrow \ldots \alpha))^\sigma = t$.

Conversely, assume $\phi_1 \rightarrow \phi_2 \rightarrow \ldots \rightarrow \phi_k \rightarrow \alpha$ is a tautology. For any truth valuation σ such that $\sigma \models \{\phi_1, \ldots, \phi_k\}$, $\phi_1^\sigma = \phi_2^\sigma = \ldots = \phi_k^\sigma = t$ and as $(\phi_1 \rightarrow \phi_2 \rightarrow \ldots \rightarrow \phi_k \rightarrow \alpha)^\sigma = t$, it must be the case that $\alpha^\sigma = t$. $\square$

4.3.4 Epistemological Interpretation

The assertion operator $\mathbf{T}$, like the modal operator $\Box$, can be interpreted as asserting that its argument is *known* to be true or, using Galton's interpretation [Gal91a], that *there is reason to believe* that its argument is true. This leads to the confidence spectrum shown in Table 4.3. The logic escapes the law of excluded middle at the "epistemic" level since $\mathbf{T}x \vee \mathbf{T}{-}x$ is not tautologically true.

For notational convenience we define $\mathbf{P}$ (possibly), $\mathbf{F}$ (known to be false) and $\mathbf{U}$ (unknown) as follows.

$$\begin{aligned} \mathbf{P}x &=_{\text{def}} \neg\mathbf{T}{-}x \\ \mathbf{F}x &=_{\text{def}} \mathbf{T}{-}x \\ \mathbf{U}x &=_{\text{def}} \neg(\mathbf{T}x \vee \mathbf{T}{-}x) \end{aligned}$$

The five levels of the confidence spectrum therefore correspond to assignments of t to $\mathbf{T}x$, $\mathbf{P}x$, $\mathbf{U}x$, $\mathbf{P}{-}x$ and $\mathbf{F}x$ respectively. Note that $\mathbf{P}$, $\mathbf{F}$ and $\mathbf{U}$ are not additional operators

Table 4.3: The confidence spectrum for AL.

Informal Interpretation	Formal Interpretation x	$\mathbf{T}x$	$\mathbf{T}{-}x$
reason to believe	t	t	f
no reason to disbelieve	t or u	–	f
no reason to believe or disbelieve	u	f	f
no reason to believe	u or f	f	–
reason to disbelieve	f	f	t

(although it may sometimes be convenient to refer to them as such) but meta-language symbols with the substitutions defined above.

While $\mathbf{T}$ exhibits epistemological characteristics often associated with the modal operator $\Box$ it is important to remember that $\mathbf{T}$ is simply a truth functional operator. In fact, *any* propositional AL theory can be mapped directly onto an equivalent classical theory [MF90b]. Classical "off the shelf" theorem provers can therefore be used to test the validity of propositional AL theorems.

4.4 Propositional Calculus in AL

The propositional calculus provides a sound and complete method for generating the tautological consequences of any set of wffs. It is sound in the sense that a formula α can be deduced from a set of formulas Φ only if it is a tautological consequence of Φ. It is complete in the sense that if a formula α is a tautological consequence of a set of formulas Φ, then there is a deduction of α from Φ.

In this section we show that a simple modification to the standard propositional calculus is needed to transport it to AL, and outline a sequence of results proving soundness and completeness. Many of the proofs are based on those for the standard propositional calculus, for which further details can be found in [BM77]. An alternative proof of soundness and completeness based on a mapping from AL theories to classical theories is given in [MF90b].

4.4.1 Deductions in AL

A *deduction* is an application of *propositional axioms* and *rules of inference* to a set of formulas in order to derive a tautological consequence of those formulas. The axioms described by the following schemata, where α, β and γ represent wffs, are sufficient for defining deductions in the classical propositional calculus:

Ax. I $\quad \alpha{\to}\beta{\to}\alpha$

Ax. II $\quad (\alpha{\to}\beta{\to}\gamma){\to}(\alpha{\to}\beta){\to}\alpha{\to}\gamma$

Ax. III $\quad (\neg\alpha{\to}\beta){\to}(\neg\alpha{\to}\neg\beta){\to}\alpha$

To modify the calculus for AL we add the axiom schemata

Ax. IV $\neg(\mathbf{T}x \wedge \mathbf{T}{-}x)$
Ax. V $\mathbf{T}x \leftrightarrow \mathbf{T}{-}{-}x$

where x represents a base formula.[1] Axiom IV ensures that $\mathbf{T}x$ and $\mathbf{T}{-}x$ cannot both be deduced from a consistent theory, thus capturing the intuitive fact that a proposition cannot be both known to be true and known to be false. Axiom V permits the nesting of strong negation connectives.

It is important to note that since base formulas do not appear outside the scope of the assertion operator we do not require rules such as necessitation ($\vdash x \,/\, \vdash \Box x$) and axioms such as reflexivity ($\Box x \rightarrow x$) which are associated with modal logics.

Following [BM77] we use one rule of inference, *modus ponens*, which allows a formula β (the conclusion) to be deduced from formulas $\alpha \rightarrow \beta$ (the major premiss) and α (the minor premiss). The following lemma shows that modus ponens is *(semantically) sound* in AL.

Lemma 4.4.1 *For any wffs α and β in* AL*,*

$$\{\alpha, \alpha \rightarrow \beta\} \models \beta.$$

Proof. The formula $\alpha \rightarrow (\alpha \rightarrow \beta) \rightarrow \beta$ is a tautology in L and hence AL from Theorem 4.3.2. Therefore $\{\alpha, \alpha \rightarrow \beta\} \models \beta$ from Lemma 4.3.3. $\Box$

Deductions can now be defined more rigorously as follows.

Definition 4.4.2 A *(propositional) deduction* from a set Φ of wffs is a non-empty sequence of wffs $\psi_1, \ldots, \psi_n$ such that for each k, $1 \leq k \leq n$, ψ_k is a propositional axiom, $\psi_k \in \Phi$, or ψ_k can be obtained by modus ponens from earlier formulas in the sequence (i.e. there are $i, j < k$ such that $\psi_j = \psi_i \rightarrow \psi_k$).

In this regard Φ is often called a set of *hypotheses*. A formula α is *deducible* from Φ (written $\Phi \vdash \alpha$) if there is a deduction from Φ whose last formula is α.

Clearly if $\Phi \subseteq \Psi$ then any deduction from Φ is also a deduction from Ψ. Also, if $\phi_1, \ldots, \phi_n$ is a deduction from Φ and $1 \leq k \leq n$, then $\phi_1, \ldots, \phi_k$ is a deduction from Φ.

A *(propositional) proof* is a deduction from the empty set of hypotheses. A formula α is *provable* (written $\vdash \alpha$) if there is a proof whose last formula is α.

Theorem 4.4.3 (Deduction Theorem) *Given a deduction of β from $\Phi \cup \{\alpha\}$ we can construct a deduction of $\alpha \rightarrow \beta$ from Φ. (Thus, if $\Phi \cup \{\alpha\} \vdash \beta$, then $\Phi \vdash \alpha \rightarrow \beta$.)*

Proof. (Sketch) Let $\psi_1, \ldots, \psi_n$ be a deduction of β from $\Phi \cup \{\alpha\}$. It can be shown [BM77, Thm 1.10.4] by induction on $k = 1, \ldots, n$ that a deduction of $\alpha \rightarrow \psi_k$ from Φ can be constructed by considering each of the possible cases: ψ_k is an axiom, $\psi_k \in \Phi$, $\psi_k = \alpha$, or ψ_k is obtained by modus ponens from two earlier formulas ψ_i and ψ_j, $i, j < k$. $\Box$

[1] Note that adding $\mathbf{T}x \vee \mathbf{T}{-}x$ as an axiom schema would revert the logic to a classical 2-valued system.

4.4.2 Soundness, Consistency and Completeness

Theorem 4.4.4 (soundness) [BM77, Thm 1.10.2] *If* $\Phi \vdash \alpha$*, then* $\Phi \models \alpha$*. In particular, if* $\vdash \alpha$*, then* $\models \alpha$*.*

Proof. Let $\psi_1, \ldots, \psi_n$ be a deduction of α from Φ (thus $\psi_n = \alpha$). We show by induction on $k = 1, \ldots, n$ that $\Phi \models \psi_k$.

If ψ_k is a propositional axiom it is tautologically true and therefore satisfied by any truth valuation. If $\psi_k \in \Phi$ then clearly $\Phi \models \psi_k$. Finally, if for some $i, j < k$ we have $\psi_j = \psi_i \rightarrow \psi_k$, then $\{\psi_i, \psi_j\} \models \psi_k$ by Lemma 4.4.1, but by the induction hypothesis $\Phi \models \psi_i$ and $\Phi \models \psi_j$, hence clearly $\Phi \models \psi_k$. □

Definition 4.4.5 A set Φ of wffs is *(propositionally) inconsistent* if for some β both $\Phi \vdash \beta$ and $\Phi \vdash \neg\beta$. Otherwise Φ is *(propositionally) consistent.*

The set $\{\mathbf{T}x, \mathbf{F}x\}$ is inconsistent, for example, since $\{\mathbf{T}x, \mathbf{F}x\} \vdash \mathbf{F}x$ and $\{\mathbf{T}x, \mathbf{F}x\} \vdash \neg\mathbf{F}x$ using Axiom IV and modus ponens.

It can be shown [BM77, Thm 1.10.5(c)] that $\{\beta, \neg\beta\} \vdash \alpha$ for any formulas β and α. It follows that if Φ is inconsistent, then $\Phi \vdash \alpha$ for any formula α.

The soundness of the propositional calculus leads to the following result.

Theorem 4.4.6 [BM77, Thm 1.10.6] *No truth valuation satisfies an inconsistent set of wffs.*

Proof. Let $\Phi \vdash \beta$ and $\Phi \vdash \neg\beta$. If $\sigma \models \Phi$ then by Theorem 4.4.4, $\sigma \models \beta$ and $\sigma \models \neg\beta$. Contradiction. □

Theorem 4.4.6 shows that the empty set of formulas is consistent as it is satisfied (vacuously) by every truth valuation.

Theorem 4.4.7 [BM77, Thm 1.10.8(a)] *For any* Φ *and* α*,* $\Phi \cup \{\neg\alpha\}$ *is inconsistent iff* $\Phi \vdash \alpha$*.*

Proof. If $\Phi \cup \{\neg\alpha\}$ is inconsistent, then for some β we have $\Phi \cup \{\neg\alpha\} \vdash \beta$ and $\Phi \cup \{\neg\alpha\} \vdash \neg\beta$. By the Deduction Theorem, $\Phi \vdash \neg\alpha \rightarrow \beta$ and $\Phi \vdash \neg\alpha \rightarrow \neg\beta$. Using Axiom III with two applications of modus ponens we get a deduction of α from $\{\neg\alpha \rightarrow \beta, \neg\alpha \rightarrow \neg\beta\}$. Thus $\Phi \vdash \alpha$.

Conversely, if $\Phi \vdash \alpha$ then $\Phi \cup \{\neg\alpha\}$ is inconsistent because $\Phi \cup \{\neg\alpha\} \vdash \alpha$ and $\Phi \cup \{\neg\alpha\} \vdash \neg\alpha$. □

Theorem 4.4.7 has practical importance because it shows that if we can test the consistency of a knowledge base then we can also show whether a formula is deducible from the knowledge base.

Theorem 4.4.8 *A set of formulas* Φ *is consistent iff every finite subset of* Φ *is consistent.*

Proof. Let Φ be consistent and assume there exists a set $\Theta \subseteq \Phi$ such that Θ is inconsistent, that is $\Theta \vdash \beta$ and $\Theta \vdash \neg\beta$ for some formula β. As $\Theta \subseteq \Phi$ any deduction from Θ is also a deduction from Φ. Therefore, $\Phi \vdash \beta$ and $\Phi \vdash \neg\beta$ so Φ is inconsistent. Contradiction.

Conversely, let all finite subsets of Φ be consistent and assume Φ is inconsistent, that is $\Phi \vdash \beta$ and $\Phi \vdash \neg\beta$ for some formula β. Construct the set Θ of all formulas $\phi \in \Phi$ used in the deductions of β and $\neg\beta$. Then Θ is a finite subset of Φ, but $\Theta \vdash \beta$ and $\Theta \vdash \neg\beta$ so Θ is inconsistent. Contradiction. $\square$

A set Φ of formulas is *maximal consistent in* AL if it is propositionally consistent but is not a proper subset of any consistent set of formulas. Note that a consistent set Φ is maximal consistent iff $\alpha \in \Phi$ or $\neg\alpha \in \Phi$ for every formula α.

The following proof makes use of Zorn's Lemma which states that for each element b of an inductive set $\langle A, \leq \rangle$ there is a maximal element $a \in A$ such that $b \leq a$.

Theorem 4.4.9 [BM77, Thm 1.13.3] *Let Φ be a consistent set of wffs. Then there is a maximal consistent set Ψ such that $\Phi \subseteq \Psi$.*

Proof. Consider the family of all consistent sets of formulas, partially ordered by inclusion $\subseteq$. If $\{\Phi_i : i \in I\}$ is an arbitrary totally ordered subfamily of that family, then by Theorem 4.4.8 the union $\bigcup\{\Phi_i : i \in I\}$ is also consistent, as every finite subset of the union is included in some Φ_i. The result follows from Zorn's Lemma. $\square$

We can now show the converse of Theorem 4.4.6.

Theorem 4.4.10 *If Φ is a consistent set of formulas then there is a truth valuation σ satisfying Φ.*

Proof. By Theorem 4.4.9 we can assume that $\Phi \subseteq \Psi$, where Ψ is maximal consistent in AL. Therefore, for each atomic base formula x, Ψ must include exactly one formula from each pair in the sequence $\langle \mathbf{T}x, \neg\mathbf{T}x \rangle, \langle \mathbf{T}{-}x, \neg\mathbf{T}{-}x \rangle, \langle \mathbf{T}{-}{-}x, \neg\mathbf{T}{-}{-}x \rangle, \ldots$. There are three possible options:

1. If $\mathbf{T}x \in \Psi$ then $\neg\mathbf{T}{-}x \in \Psi$ (since $\mathbf{T}x \vdash \neg\mathbf{T}{-}x$ by Axiom IV) and $\mathbf{T}{-}{-}x \in \Psi$ (since $\mathbf{T}x \vdash \mathbf{T}{-}{-}x$ by Axiom V). Similarly $\neg\mathbf{T}{-}{-}{-}x \in \Psi$, $\mathbf{T}{-}{-}{-}{-}x \in \Psi$ and so on. These are precisely the formulas satisfied by the assignment $x^\sigma = t$.

2. If $\mathbf{T}{-}x \in \Psi$ then Ψ includes the formulas $\mathbf{T}{-}{-}{-}x, \mathbf{T}{-}{-}{-}{-}{-}x, \ldots$ (due to Axiom V) and the formulas $\neg\mathbf{T}x, \neg\mathbf{T}{-}{-}x, \ldots$ (due to Axiom IV). These are precisely the formulas satisfied by the assignment $x^\sigma = f$.

3. If neither $\mathbf{T}x \in \Psi$ nor $\mathbf{T}{-}x \in \Psi$ then both $\neg\mathbf{T}x \in \Psi$ and $\neg\mathbf{T}{-}x \in \Psi$. By Axiom V, Ψ also includes $\neg\mathbf{T}{-}{-}x$, $\neg\mathbf{T}{-}{-}{-}x$ and so on. These are precisely the formulas satisfied by the assignment $x^\sigma = u$.

A truth valuation σ can therefore be determined by requiring that for every atomic base formula x,

$$x^\sigma = \begin{cases} t & \mathbf{T}x \in \Psi \\ f & \mathbf{T}{-}x \in \Psi \\ u & \text{otherwise} \end{cases}$$

and for every universal formula α,

$$\alpha^\sigma = \begin{cases} t & \alpha \in \Psi \\ f & \text{otherwise.} \end{cases}$$

The resulting truth valuation is such that for every atomic or universal formula ϕ, $\phi^\sigma = t$ iff $\phi \in \Psi$. By induction on deg ϕ this holds for every wff ϕ. Since $\Phi \subseteq \Psi$ it follows that $\sigma \models \Phi$. $\square$

Theorem 4.4.11 (strong completeness) [BM77, Thm 1.13.5] *If $\Phi \models \alpha$ then $\Phi \vdash \alpha$. In particular, if $\models \alpha$ then $\vdash \alpha$.*

Proof. If $\Phi \models \alpha$ then no truth valuation can satisfy $\Phi \cup \{\neg\alpha\}$ (if $\sigma \models \Phi$ then $\alpha^\sigma = t$ and $(\neg\alpha)^\sigma = f$, so $\sigma \not\models \neg\alpha$). Hence by Theorem 4.4.10, $\Phi \cup \{\neg\alpha\}$ is inconsistent, and by Theorem 4.4.7, $\Phi \vdash \alpha$. $\square$

4.5 Remarks

We have shown the propositional calculus defined in AL to be sound and complete. Soundness is important in a reasoning system because it ensures that only (semantically) correct results will be deduced from a set of hypotheses. Completeness guarantees that it is possible to deduce all such results.

In the derivation of the above results we have shown that a number of theorems from classical propositional calculus can be transported to asserted logic. Of course this is also the case for many others which have not been shown. Some of these results provide useful guarantees for practical reasoning systems. For example, the following lemma shows that deduced formulas can be added to a consistent knowledge base without fear of introducing inconsistencies.

Lemma 4.5.1 *If Φ is a propositionally consistent set of wffs and $\Phi \vdash \alpha$, then $\Phi \cup \{\alpha\}$ is propositionally consistent.*

Proof. If Φ is consistent, then by Theorem 4.4.10 there is a truth valuation σ satisfying Φ. Also, if $\Phi \vdash \alpha$ then from Theorem 4.4.4, $\Phi \models \alpha$. Hence $\sigma \models \alpha$, and from Theorem 4.4.6, $\Phi \cup \{\alpha\}$ is consistent. $\square$

Guarantees of this sort support the use of logic-based formalisms for manipulating declarative knowledge. The results outlined above are intended to illustrate that the advantages of logic-based systems are not lost when asserted logic is used in place of classical logic.

Finally, theorem provers for AL can be obtained by straightforward modifications to classical theorem provers. We have modified Fitting's propositional tableau theorem prover [Fit88] for use with AL by further reducing the nodes of the tableau containing atomic formulas to leaves containing base formulas.[2] The implementation is in Prolog.

[2] An earlier version of the theorem prover was implemented by Rajeev Goré.

Chapter 5

Simplified Chronological Ignorance

This chapter describes the first stage of the development of a calculus for causal chronological ignorance. We show that the use of modal logic in CI is unnecessary, and that the same results can be achieved using asserted logic. In fact we could use any language which gives access to the confidence spectrum described in Chapter 4.

The translation of CI to asserted logic simplifies the semantics of CI, and makes it easier to prove that CI is deterministic for causal theories. Since we will develop our calculus using asserted logic, it also gives us a semantics against which we can prove soundness and completeness.

5.1 Temporal AL

In order to minimise knowledge over time we need to attach temporal information to assertions. As we discussed earlier, Shoham [Sho88a] achieves this by reifying propositions with pairs of temporal arguments which are intended to denote intervals. Bacchus *et al* [BTK89] argue that this approach is overly restrictive and sacrifices classical proof theory unnecessarily. They show that a two-sorted logic called BTK is more expressive and subsumes Shoham's temporal logic. Galton [Gal91b] also argues against reified temporal logics.

Since causal theories are propositional our choice here is not particularly important. However we will follow the approach of Bacchus *et al* for two reasons. First we would like to take advantage of the less restrictive syntax of BTK and allow single temporal arguments. The reason for this is that we will want to use the logic for state-based reasoning. Shoham's solution of representing states or time points as intervals with zero duration is more cumbersome and, as argued by Bacchus *et al*, offers no advantages in return. The second reason is that we would like to allow for a later extension of our system to first-order logic. Since we are developing a proof-theoretic formalism we will require a logic with a well-established proof theory. Standard proof theories for many sorted logics are directly applicable to BTK. Moreover, Bacchus *et al* provide a mapping from BTK theories to classical first-order theories which permit the use of standard theorem provers.

5.1.1 The Temporal Logic BTK

BTK is a standard many-sorted logic with two disjoint sorts, for nontemporal and temporal objects. The syntax of BTK is that of a classical first-order language L with a few exceptions. The variables are divided into temporal and nontemporal sorts. Function symbols and the constants (which are taken to be 0-ary function symbols) are classed according to the sort which they return, and each predicate or function symbol has arity (m, n) where m and n are natural numbers—the first m arguments being nontemporal while the last n are temporal.[1] Well-formed formulas are constructed in the usual way with the restriction that arguments of the correct sort must be given to predicates and functions.

The propositional semantics of BTK is the same as that of classical logic. The first-order semantics is modified to include two non-empty universes, U and T. An interpretation function σ maps each (m, n)-ary nontemporal function symbol to an operation from $U^m \times T^n$ to U, each (m, n)-ary temporal function symbol to an operation from $U^m \times T^n$ to T, and each (m, n)-ary predicate symbol to a relation on $U^m \times T^n$. Finally, quantified variables are understood to range only over the appropriate universe.

5.1.2 TAL

Temporal asserted logic (TAL) is formed in the same way as AL, described in Chapter 4, except that the underlying classical language L is replaced by the two-sorted language BTK. For consistency with Shoham we take all temporal constant symbols from the set of integers $\mathbb{Z}$. We also use only predicate symbols with arity $(m, 1)$ or $(m, 2)$. The former can be interpreted as associating an assertion with a particular time point or a state, while the latter associates an assertion with an interval. Thus `loaded(gun, 1)` is intended to indicate that a gun is loaded at time 1, while `red(house17, 2, 12)` indicates that a particular house is red in the interval from 2 to 12.

Finally we require Shoham's notion of the *latest time point* (*ltp*) appearing in a propositional sentence.[2] This is simply the greater (with respect to the normal interpretation of the integers) of the temporal constant symbols appearing in the sentence. For simplicity we will follow the convention that the latest time point in an atomic base sentence is always written as the last argument. Thus the *ltp* of a base sentence $p(\ldots, t_1, t_2)$ is t_2.

5.2 Chronological Minimisation

TAL is made nonmonotonic by imposing a partial order on truth valuations and choosing a least element in the partial order which satisfies the object theory. The partial order gives preference to truth valuations in which the base formulas that are *known* occur as *late* as possible. Our partial order is simpler than Shoham's [Sho88a, Def. 3.4] since the truth functionality of TAL allows us to specify the truth values of base formulas directly.

[1] We have changed the order of temporal and non-temporal terms from [BTK89] to conform with the bulk of the literature on state-based reasoning (see Chapter 2).

[2] See [Sho88a, Sec. 3.3] for the definition of *ltp* for quantified sentences.

Definition 5.2.1 A truth valuation σ_2 is *chronologically more ignorant* than a truth valuation σ_1 (written $\sigma_2 \prec_c \sigma_1$) if there exists a time t_0 such that

1. for any base sentence x whose $ltp \leq t_0$, if $x^{\sigma_1} = u$ then $x^{\sigma_2} = u$, and
2. there exists some base sentence y whose ltp is t_0 such that $y^{\sigma_2} = u$ but $y^{\sigma_1} \neq u$.

Definition 5.2.2 σ is a *chronologically maximally ignorant* (cmi) truth valuation for a theory Φ if $\sigma \models \Phi$ and there is no $\sigma' \prec_c \sigma$ such that $\sigma' \models \Phi$.

According to Definition 5.2.2, preferred truth valuations assign the value u to base formulas wherever possible, choosing base formulas with smaller *ltp*'s where necessary. For example, if an object theory entails a disjunction $\mathbf{T}x_1 \vee \mathbf{T}x_2$ but does not entail either $\mathbf{T}x_1$ or $\mathbf{T}x_2$, then the preferred truth valuation assigns the value u to the base formula x_1 or x_2 with the smallest *ltp*. If there is a choice between base formulas with the same *ltp*'s then there will generally not be a unique cmi truth valuation. Causal theories, which are considered in the following section, are designed so that this problem does not arise.

5.3 Causal Theories

Causal theories are intended to capture the notion of physical causality—that what is true prior to some point in time determines what is true after that time. They can be defined in TAL as follows.

Definition 5.3.1 A *causal sentence* is a sentence of the form

$$\bigwedge_{i=1}^{m} \mathbf{T}x_i \ \wedge \ \bigwedge_{j=1}^{n} \mathbf{P}y_j \ \rightarrow \ \mathbf{T}z, \qquad m, n \in \mathbb{N} \tag{5.1}$$

where

1. if $m = 0$ or $n = 0$ the corresponding (empty) conjunction is identically true, and
2. x_i, y_j $(i, j > 0)$ and z are base literals such that
   (a) $ltp(x_i) < ltp(z)$ for $i = 1, \ldots, m$, and
   (b) $ltp(y_j) < ltp(z)$ for $j = 1, \ldots, n$.

A causal sentence is called a *boundary condition* if $m = 0$, and a *causal rule* otherwise.

Definition 5.3.2 A *causal theory* Φ is a set of causal sentences such that

1. there is a time point t_0 such that for all boundary conditions $\beta \in \Phi$, $t_0 < ltp(\beta)$,
2. there do not exist sentences $\alpha_1 \rightarrow \mathbf{T}z$ and $\alpha_2 \rightarrow \mathbf{T}{-z}$ in Φ such that $\{\alpha_1, \alpha_2\}$ is consistent, and

3. there is no x such that $\mathbf{P}x$ and $\mathbf{P}{-}x$ both appear on the left-hand side of sentences in Φ.[3]

Causal sentences are thus implications from knowledge prior to some point in time to knowledge at that time. The partial order on interpretations gives a sense of unidirectionality to the implication connective since earlier assertions, which appear to the left of the connective, are minimised preferentially. This unidirectionality appears to be useful for modelling causality [Sho88b]. More importantly from a technical point of view, however, the constraints on causal theories guarantee that the partial order on interpretations has a unique least element satisfying any given theory. We now prove this to be the case.

5.4 Unique CMI Truth Valuations

Shoham's causal theories are shown to be deterministic by the "unique model theorem" [Sho88a, Thm. 4.4] which says that the same base sentences are known in all cmi interpretations of a causal theory. A stronger result follows in our formalism; namely, that there is exactly *one* cmi truth valuation for any causal theory.

Theorem 5.4.1 (Unique cmi truth valuation) *Let Φ be a causal theory and Σ be the set of all truth valuations on* TAL. *Reduce Σ to Σ_∞ as follows:*

- *Let t_0 be a time point such that for all boundary conditions $\beta \in \Phi$, $t_0 < ltp(\beta)$. Let Σ_0 be the set of truth valuations $\sigma \in \Sigma$ in which $x^\sigma = u$ for all atomic base sentences x whose $ltp \leq t_0$.*
- *For all $i > 0$, $i \in \mathbb{N}$, let $t_i = t_{i-1} + 1$ and Σ_i be the set of truth valuations σ such that $\sigma \in \Sigma_{i-1}$ and*

$$x^\sigma = \begin{cases} t & \alpha \rightarrow \mathbf{T}x \in \Phi \text{ and } \sigma \models \alpha & (5.2a) \\ f & \alpha \rightarrow \mathbf{T}{-}x \in \Phi \text{ and } \sigma \models \alpha & (5.2b) \\ u & \text{otherwise} & (5.2c) \end{cases}$$

for all atomic base sentences x whose ltp is t_i. Let $\Sigma_\infty = \bigcap_{i=0}^{\infty} \Sigma_i$.

Then

1. *Σ_∞ contains a single truth valuation,*
2. *if $\sigma \in \Sigma_\infty$ then $\sigma \models \Phi$, and*
3. *σ is a cmi truth valuation for Φ if and only if $\sigma \in \Sigma_\infty$.*

[3] The third condition is not strictly necessary since we do not require Shoham's *soundness conditions* [Sho88a, Def. 5.1], and there is therefore a consistent interpretation of $\{\mathbf{P}x, \mathbf{P}{-}x\}$ in which x is assigned the value u. We include the condition here for compatibility with Shoham's formalism (see Definition 3.6.3). The soundness conditions are discussed further in Chapter 8.

Proof.

[1] The second condition of Definition 5.3.2 ensures that (5.2a) and (5.2b) cannot be satisfied simultaneously. The construction therefore defines a mapping from each atomic base formula onto a unique truth value in $\{t, f, u\}$. The mapping extends according to Definition 4.3.1 to a unique truth valuation.

[2] Let $\sigma \in \Sigma_\infty$. Any sentence $\phi \in \Phi$ whose $ltp \leq t_0$ must be a causal rule of the form $\mathbf{T}x_1 \wedge \ldots \rightarrow \mathbf{T}z$ where $ltp(x_1) < t_0$. Therefore $x_1^\sigma = u$, $(\mathbf{T}x_1)^\sigma = f$ and $\sigma \models \phi$. Any sentence whose $ltp > t_0$ must also be satisfied since it is of the form $\alpha \rightarrow \mathbf{T}z$ and from (5.2a)–(5.2c), if $\alpha^\sigma = t$ then $(\mathbf{T}z)^\sigma = t$.

[3] ($\Leftarrow$) Assume $\sigma \in \Sigma_\infty$ is not a cmi truth valuation for Φ. From [2] $\sigma \models \Phi$ so there must be some $\sigma' \prec_c \sigma$ such that $\sigma' \models \Phi$. Let t_e be the earliest time point at which σ and σ' differ. Then, since $\sigma' \prec_c \sigma$, there is some atomic base sentence x with ltp t_e such that $x^\sigma \neq u$ and $x^{\sigma'} \neq x^\sigma$. From the construction there must be some sentence $\alpha \rightarrow \mathbf{T}x$ or $\alpha \rightarrow \mathbf{T}{-}x$ such that $\alpha^\sigma = t$. But $ltp(\alpha) < t_e$ so $\alpha^{\sigma'} = \alpha^\sigma = t$. Clearly σ' does not satisfy this sentence and hence $\sigma' \not\models \Phi$ — contradiction.

($\Rightarrow$) Assume σ is a cmi truth valuation for Φ but $\sigma \notin \Sigma_\infty$. From [1] and [2] there exists $\sigma' \in \Sigma_\infty$ such that $\sigma' \models \Phi$. Since $\sigma \notin \Sigma_\infty$ either

1. $x^\sigma \neq u$ for some atomic base sentence x whose $ltp \leq t_0$, or
2. there exists i such that $\sigma \in \Sigma_{i-1}$ but $\sigma \notin \Sigma_i$, in which case either
   (a) $\alpha \rightarrow \mathbf{T}x \in \Phi$ for some atomic base sentence x whose $ltp = t_i$ and $\sigma \models \alpha$, but $x^\sigma \neq t$,
   (b) $\alpha \rightarrow \mathbf{T}{-}x \in \Phi$ for some atomic base sentence x whose $ltp = t_i$ and $\sigma \models \alpha$, but $x^\sigma \neq f$, or
   (c) there is no base sentence x whose $ltp = t_i$ such that $\alpha \rightarrow \mathbf{T}x \in \Phi$ and $\sigma \models \alpha$, but $x^\sigma \neq u$.

In the case of 1 or 2(c), $\sigma' \prec_c \sigma$, and in the case of 2(a) or 2(b), $\sigma \not\models \Phi$. Therefore σ is not a cmi truth valuation for Φ — contradiction. □

The following are direct consequences of Theorem 5.4.1:

- All causal theories are consistent.
- All causal theories have a unique cmi truth valuation.

We let *causal*($\wp$(TAL)) denote the causal theories of TAL and define the images under CI as follows.

Definition 5.4.2 K_{TAL} is a relation from *causal*($\wp$(TAL)) to $\wp$(TAL) such that $\langle \Phi, \Psi \rangle \in K_{\mathrm{TAL}}$ if and only if

$$\Psi = \{\mathbf{T}x \mid x \text{ is a base literal and } \sigma \models \mathbf{T}x \text{ where } \sigma \text{ is the cmi model for } \Phi\}.$$

Theorem 5.4.1 ensures that the relation K_{TAL} is deterministic.

5.4.1 Finite Causal Theories

The construction in Theorem 5.4.1 shows that if Φ is a finite causal theory then the number of atomic base sentences which are assigned a value other than u by the cmi truth valuation for Φ is finite and no greater than the number of sentences in Φ. To see this note that atomic base sentences can only be assigned a value of t or f according to conditions (5.2a) and (5.2b) respectively, and each formula in Φ can satisfy at most one of these conditions.

The atomic base sentences assigned values other than u are found by the following algorithm which simply assigns the truth values determined by (5.2a)–(5.2b) .

Algorithm 5.4.3 Let Φ be a finite causal theory with cmi truth valuation σ.

Step 1 Let T be a list of all the sentences in Φ and S be a list of all the atomic base sentences appearing in T.

Step 2 Sort the sentences in T and S by *ltp*.

Step 3 If T is empty then halt. The atomic base sentences assigned t by σ are labelled t in S, and the atomic base sentences assigned f by σ are labelled f in S.

Step 4 Remove the first element $\bigwedge_{i=1}^{m} \mathbf{T}x_i \;\wedge\; \bigwedge_{j=1}^{n} \mathbf{P}y_j \;\rightarrow\; \mathbf{T}z$ from T. If one of the following conditions holds:

1. for some i, x_i is a positive base literal and x_i is not labelled t in S,
2. for some i, x_i is a negative base literal $-x_i'$ and x_i' is not labelled f in S,
3. for some j, y_j is a positive base literal and y_j is labelled f in S,
4. for some j, y_j is a negative base literal $-y_j'$ and y_j' is labelled t in S,

then go to Step 3. Otherwise label the atomic part of z either t or f in S depending on whether z is a positive or negative base literal, and go to Step 3.

Thus

$$K_{\mathrm{TAL}}(\Phi) = \{\mathbf{T}x \mid x \text{ is labelled } t \text{ in } S\} \cup \{\mathbf{T}{-y} \mid y \text{ is labelled } f \text{ in } S\}.$$

5.5 Equivalence of the Formalisms

Other than syntactic variations, Algorithm 5.4.3 is precisely the algorithm provided by Shoham [Sho88a, pp. 300-301]. This correspondence is formalised by the following corollary. We make use of a syntactic mapping *TK* from TAL base sentences to TK base sentences defined by:

$$\begin{aligned} TK(p(\ldots,t)) &=_{\mathrm{def}} \ \mathrm{TRUE}(t,t,p'(\ldots)); \quad p \text{ has arity } (m,1), \ \ p' \text{ has arity } m \\ TK(p(\ldots,t_1,t_2)) &=_{\mathrm{def}} \ \mathrm{TRUE}(t_1,t_2,p'(\ldots)); \quad p \text{ has arity } (m,2), \ \ p' \text{ has arity } m \\ TK(-x) &=_{\mathrm{def}} \ \neg\, TK(x) \end{aligned}$$

Corollary 5.5.1 *Let Φ be a finite causal theory with cmi truth valuation σ. Let*

$$\Phi' = \left\{ \bigwedge_{i=1}^{m} \Box TK(x_i) \;\wedge\; \bigwedge_{j=1}^{n} \Diamond TK(y_j) \rightarrow \Box TK(z) \;\middle|\; \bigwedge_{i=1}^{m} \mathbf{T}x_i \;\wedge\; \bigwedge_{j=1}^{n} \mathbf{P}y_j \rightarrow \mathbf{T}z \;\in \Phi \right\}.$$

Then for any base literal w,

$$\mathbf{T}w \in K_{\mathrm{TAL}}(\Phi) \text{ if and only if } \Box TK(w) \in K_{\mathrm{TK}}(\Phi').$$

That is, the base literals assigned t by the cmi model of a TAL theory correspond to the base sentences which are *known* in all cmi models of the corresponding TK theory.

5.6 Remarks

We have shown that the chronological minimisation strategy which forms the basis of CI does not require the use of modal logic. In fact it can be stated more succinctly using asserted logic, which has a simpler, truth functional semantics. By providing an equivalent nonmodal account of CI for causal theories we have simplified the semantics of CI and the proof of the "unique model theorem".

The most important advantage of using asserted logic rather than modal logic, however, is that we do not sacrifice classical proof theory. As we described in Chapter 4, a sound and complete version of the propositional calculus can be transported to AL with the addition of two axiom schemata. Alternatively, we can map propositional AL theories onto the classical language $\mathrm{K_P}$ and thereby use standard theorem provers to verify AL theorems. In Chapter 8 we will show how this can be used to provide a calculus for causal CI.

On a broader level this result calls into question the ready acceptance of modal logic for representing epistemic concepts such as knowledge and possibility. Since the inclusion of modality complicates the semantics and proof theory of a logic, its use should be justified—for example by the need to nest modal operators or apply them to compound formulas.

Chapter 6

A New Look at Default Logic

In Chapter 3 we introduced default logic and outlined a number of properties of the extension relation. While default rules provide a versatile mechanism for incorporating assumptions, the extension relation suffers from a number of undesirable properties. It is deductively closing, partial for seminormal defaults, and branching for all types of defaults. In addition default logic lacks a proof theory for arbitrary defaults and a local proof theory with respect to seminormal defaults.

We now proceed to address each of these problems. In this chapter we show that deductive closure can be factored out of the extension relation providing a nonclosing definition of default logic. This in turn suggests a procedure for testing extension membership for nonnormal defaults based around a classical theorem prover. We then propose an approach for converting seminormal defaults to normal defaults thus overcoming the problem of partial relations (or incoherence).

6.1 Default Logic Without Deductive Closure

Returning to the definition of the extension relation (Def. 3.4.1) in Chapter 3, recall that condition (3.2c) ensures that extensions are deductively closed. This means that finite theories have infinite images under the extension relation, and even for finite sets of defaults the images cannot be generated. Reiter [Rei80, Thm 2.5] shows that any extension can be written as the deductive closure of a set consisting of the object theory and the consequents of some of the defaults. The result can be expressed as follows.

Lemma 6.1.1 *Let E_D be defined according to Definition 3.4.1. If $\langle \Phi, \Psi \rangle \in E_D$ then $\Psi = Th(\Theta)$ where*

$$\Theta = \Phi \cup \{\gamma \mid (\alpha : \beta \,/\, \gamma) \in D, \;\; \alpha \in \Psi \text{ and } \neg\beta \notin \Psi\}. \tag{6.1}$$

Lemma 6.1.1 is important because it shows that for finite sets of defaults, the extensions of finite theories are finitely axiomatisable. Furthermore, the proper axioms are a subset of $\Phi \cup \text{CONSEQUENTS}(D)$.

The converse of Lemma 6.1.1 does not hold. That is, if $\Psi = Th(\Theta)$ where Θ satisfies (6.1), it may not be the case that $\langle \Phi, \Psi \rangle \in E_D$. Consider for example a default set

$$D = \left\{ \frac{\mathtt{p} :}{\mathtt{p}} \right\}$$

and an object theory $\Phi = \{\,\}$. The set $\Theta = \{\mathtt{p}\}$ satisfies (6.1) but $\langle \{\,\}, \mathit{Th}(\{\mathtt{p}\}) \rangle \notin E_D$. The correct extension is $\mathit{Th}(\{\,\})$ as is intuitively expected since the default is sanctioned only by its own conclusion.

A more manageable definition of default logic can be provided by considering minimal sets of proper axioms whose deductive closure is an extension; that is, those for which a bidirectional result similar to Lemma 6.1.1 holds. At first sight it may appear that the solution is to demand that Θ be a minimal (with respect to set inclusion) theory satisfying the right-hand side of equation (6.1). This restriction is insufficient, however, as the following example shows.[1] Consider a default set

$$D = \left\{ \frac{\mathtt{p} :}{\mathtt{p}}, \frac{: \neg\mathtt{p}}{\mathtt{q}} \right\}$$

and an object theory $\Phi = \{\,\}$. The set $\Theta = \{\mathtt{p}\}$ is a minimal set satisfying (6.1) but $\langle \{\,\}, \mathit{Th}(\{\mathtt{p}\}) \rangle \notin E_D$. In this case the correct extension is $\mathit{Th}(\{\mathtt{q}\})$.

The required sets, which we call *augmentations* of the underlying theory, are defined as follows.

Definition 6.1.2 (Augmentations) Let D be a set of (closed) defaults and $\mathrm{L_S}$ be a closed first-order language. The *augmentation relation* A_D is a relation on $\wp(\mathrm{L_S})$ such that $\langle \Phi, \Theta \rangle \in A_D$ if and only if

$$\Theta = \Phi \cup \{\gamma \mid (\alpha : \beta \,/\, \gamma) \in D,\ \ \Theta \vdash \alpha \ \text{ and } \Theta \not\vdash \neg\beta\} \tag{6.2a}$$

and there is no set $\Theta' \subset \Theta$ such that

$$\Theta' = \Phi \cup \{\gamma \mid (\alpha : \beta \,/\, \gamma) \in D,\ \ \Theta' \vdash \alpha \ \text{ and } \Theta \not\vdash \neg\beta\}. \tag{6.2b}$$

The extra constraint on the set Θ allows us to replace Lemma 6.1.1 with a stronger bidirectional result. The following theorem shows that a set of formulas is an extension if and only it is the deductive closure of an augmentation.

Theorem 6.1.3 *Let E_D be defined according to Definition 3.4.1 and A_D be defined according to Definition 6.1.2. Then $\langle \Phi, \Psi \rangle \in E_D$ if and only if there exists some Θ such that $\langle \Phi, \Theta \rangle \in A_D$ and $\Psi = \mathit{Th}(\Theta)$. That is,*

$$E_D = \mathit{Th} \circ A_D. \tag{6.3}$$

Proof. ($\Leftarrow$) Let $\langle \Phi, \Theta \rangle \in A_D$. From Definition 3.4.1, $\langle \Phi, \mathit{Th}(\Phi) \rangle \in E_D$ if and only if $\Gamma = \mathit{Th}(\Theta)$ where Γ is a minimal set satisfying:

$$\Phi \subseteq \Gamma \tag{6.4a}$$

$$\mathit{Th}(\Gamma) = \Gamma \tag{6.4b}$$

$$\text{if } (\alpha : \beta \,/\, \gamma) \in D,\ \alpha \in \Gamma \text{ and } \neg\beta \notin \mathit{Th}(\Theta) \text{ then } \gamma \in \Gamma \tag{6.4c}$$

[1] If this restriction were sufficient then there would be no need to distinguish between Ψ and Γ in Definition 3.4.1.

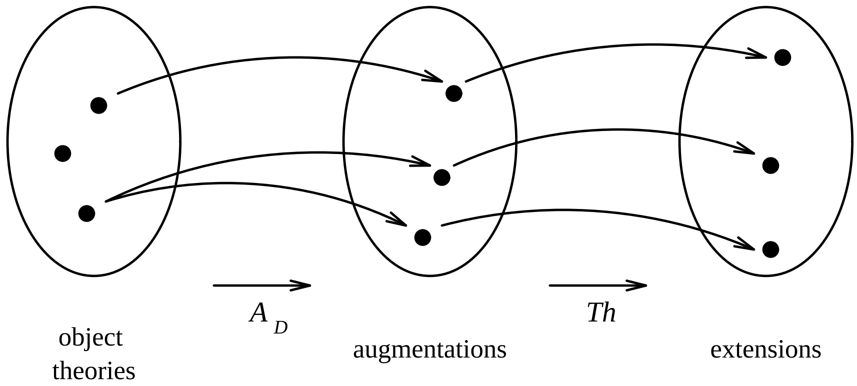

Figure 6.1: Extension as a composite relation $E_D = Th \circ A_D$.

The set $Th(\Theta)$ satisfies (6.4a) for Γ since from (6.2a), $\Phi \subseteq \Theta$, and satisfies (6.4b) since $Th(Th(\Theta)) = Th(\Theta)$. It also satisfies (6.4c) since from (6.2a), if $\alpha \in Th(\Theta)$ and $\neg\beta \notin Th(\Theta)$ then $\gamma \in \Theta$. Therefore by the minimality of Γ we have $\Gamma \subseteq Th(\Theta)$.

Let

$$\Theta' = \Phi \cup \{\, \gamma \mid (\alpha : \beta \,/\, \gamma) \in D,\ \alpha \in \Gamma \text{ and } \neg\beta \notin Th(\Theta)\,\}. \tag{6.5}$$

From (6.5) and (6.2a), $\Theta' \subseteq \Theta$ since $\Gamma \subseteq Th(\Theta)$. Also $\Theta' \subseteq \Gamma$ since Θ' is a minimal set satisfying (6.4a) and (6.4c). Therefore $Th(\Theta') \subseteq Th(\Gamma)$, and from (6.4b), $Th(\Theta') \subseteq \Gamma$. But $Th(\Theta')$ satisfies (6.4a)–(6.4c) so by the minimality of Γ we also have $\Gamma \subseteq Th(\Theta')$ and hence $\Gamma = Th(\Theta')$. Substituting for Γ, (6.5) reduces to (6.2b), so it cannot be the case that $\Theta' \subset \Theta$ since then from Definition 6.1.2, $\langle \Phi, \Theta \rangle \notin A_D$. Therefore $\Theta' = \Theta$ and $\Gamma = Th(\Theta)$. Hence $\langle \Phi, Th(\Theta) \rangle \in E_D$.

($\Rightarrow$) Let $\langle \Phi, \Psi \rangle \in E_D$. Then from Lemma 6.1.1, $\Psi = Th(\Theta)$ where Θ satisfies (6.2a). Also, from Definition 3.4.1, $\Gamma = Th(\Theta)$.

Assume $\langle \Phi, \Theta \rangle \notin A_D$. Then there exists a set $\Theta' \subset \Theta$ which satisfies (6.2b). The set $Th(\Theta')$ satisfies condition (6.4a) for Γ since from (6.2b), $\Phi \subseteq \Theta'$, and satisfies (6.4b) since $Th(Th(\Theta')) = Th(\Theta')$. It also satisfies (6.4c) since from (6.2b), if $\alpha \in Th(\Theta')$ and $\neg\beta \notin Th(\Theta)$ then $\gamma \in \Theta'$, so by the minimality of Γ we have $\Gamma \subseteq Th(\Theta')$ and hence $Th(\Theta) \subseteq Th(\Theta')$. But $\Theta' \subset \Theta$ so it must be the case that $Th(\Theta) = Th(\Theta')$, in which case the r.h.s. of (6.2b) is equivalent to the r.h.s. of (6.2a) and $\Theta' = \Theta$ contradicting the choice of Θ'. Hence $\langle \Phi, \Theta \rangle \in A_D$. □

Extensions can therefore be defined as composite relations as illustrated in Figure 6.1. The existence of an augmentation is a necessary and sufficient condition for the existence of an extension.

6.2 A Proof Procedure for Nonnormal Defaults

In this section we describe a proof procedure for default logic. In the following we call a default set *instance-finite* if it contains a finite number of closed defaults, or if it contains a finite number of defaults and the underlying language contains a finite number of variables, constant symbols, predicate symbols and no function symbols with arity greater than zero. The latter restriction makes the Herbrand Universe finite, ensuring a finite number of closed instances of open defaults [Eth87].

6.2.1 Previous Approaches

The proof theory for default logic provided by Reiter [Rei80] is applicable only to normal defaults. The theory has the advantage that it is *local* with respect to defaults; that is, it is possible to prove that a formula is a member of some extension of a given theory without taking all defaults into account. This result is a consequence of the *semi-monotonicity* property of closed normal defaults [Rei80, Thm 3.2]. The procedure is decidable for decidable first-order subclasses. Reiter argues that the extension membership problem for (closed normal) default theories in first-order logic in general is not semi-decidable [Rei80, Thm 4.9].

Etherington [Eth87, Sec 5] describes a proof procedure which is applicable to arbitrary instance-finite default sets. The procedure generates sets of formulas whose deductive closure are extensions. It is not immediately clear whether these sets obey the minimality requirements of Definition 6.1.2 and are therefore augmentations, however they must be logically equivalent to augmentations due to Theorem 6.1.3.

A problem with Etherington's procedure is that it may cycle between candidates for extensions indefinitely. The procedure is therefore not guaranteed to halt even when the language is restricted to a decidable subclass of first-order logic. This problem is shown to be overcome only for a severely restricted class of default theories called *network theories* [Eth87, Thm 2].

6.2.2 Exhaustive Search for Augmentations

Definition 6.1.2 provides a procedure for determining augmentations which requires only a standard theorem prover and simple set manipulation. The procedure can be described as follows.

Algorithm 6.2.1 Let D be a set of (closed) defaults and $P = \wp(\textsc{consequents}(D))$. Let $\Phi \subseteq \mathrm{L_S}$ and $A = \{\}$.

Step 1 If P is empty then halt. A contains the augmentations of Φ. Otherwise remove the first element Ω from P.

Step 2 Let $\Theta = \Phi \cup \Omega$. If Θ satisfies (6.2a) then let $P' = \wp(\Omega) \setminus \{\Omega\}$ and go to Step 3, otherwise go to Step 1.

Step 3 If P' is empty then replace A by $A \cup \{\Theta\}$ and go to Step 1. Otherwise, remove the first element Ω' from P'.

Step 4 Let $\Theta' = \Phi \cup \Omega'$. If Θ' satisfies (6.2b) then go to Step 1, otherwise go to Step 3.

The tests (6.2a) and (6.2b) are not decidable for countably infinite default sets since each test appeals to all defaults; that is there is no locality with respect to defaults. The procedure is therefore not semidecidable for infinite sets of defaults. Also, like the proof procedures of Reiter and Etherington, the procedure relies on a provability test and is not semidecidable for the full first-order logic. This is to be expected given Reiter's decidability result [Rei80, Thm 4.9].

For instance-finite default sets in a decidable first-order subclass, however, the tests are decidable. Equation (6.2a) shows that the candidates Θ are limited to the union of the object theory Φ with subsets of $\textsc{consequents}(D)$. If D is instance-finite then there are finitely many candidates and also finitely many subsets of the candidates which must be tested against subcondition (6.2b). Every augmentation can be found by exhaustively searching the set of candidates for those which satisfy conditions (6.2a) and (6.2b). Algorithm 6.2.1 therefore provides an effective procedure for generating the augmentations of a given theory whether or not the defaults are normal. Extension membership can be tested for any formula ϕ by constructing the augmentations and checking the consistency of $\neg\phi$ with each augmentation.

In practice the restriction to instance-finite default sets will not concern us since it will follow from our use of finite knowledge bases and decidable languages. However, while the procedure is guaranteed to halt with the correct answer under these conditions, it is computationally explosive and impractical for large numbers of defaults. In the following sections and in Chapter 7 we investigate, along with the problems of incoherence and multiple extensions, some techniques which permit improved proof procedures.

6.3 Avoiding Incoherence Using Asserted Logic

Although default logic was proposed as an enhancement to classical logic, it is equally applicable to other logic systems. In this section we show how a switch to asserted logic as the underlying language can be used to normalise defaults, thereby avoiding incoherence and restoring semi-monotonicity. The approach was first reported in [Mac91b].

6.3.1 Weakened Justifications

One of the advantages of using AL for default reasoning is that it allows the weakening of default justifications while maintaining the intended function of the defaults. To illustrate this consider the default

$$\mathbf{T}x : \mathbf{P}y \,/\, \mathbf{T}z$$

which can be interpreted "if x is known and it is consistent to believe that y is possible, then infer z". Given $\mathbf{T}x$ and no information about y, the default will support the inference of $\mathbf{T}z$. If the information $\mathbf{T}y$ ('y is known to be true') is added the inference will remain. However, if the information $\mathbf{T}{-}y$ ('y is known to be false') is added the default will no longer be applicable and the inference withdrawn. Thus the weak justification controls the applicability of the default in the usual way.[2]

Extending this idea gives the following general form for seminormal defaults.

$$\frac{\bigwedge_i \mathbf{T}x_i : \bigwedge_j \mathbf{P}y_j \;\wedge\; \bigwedge_k \mathbf{T}z_k}{\bigwedge_k \mathbf{T}z_k} \tag{6.6}$$

Note that we could equivalently have written $\neg\mathbf{T}y'_j$ where $y'_j = -y_j$ in place of $\mathbf{P}y_j$ in the justification.

Example 1 In a robotics environment where it is assumed that the only obstacles are the ones that the agent knows about, the default

$$D = \left\{ \frac{\mathbf{T}\,\mathtt{at(r,1,1)} : \neg\mathbf{T}\,\mathtt{obstacle(1,1,2,1)} \wedge \mathbf{T}\,\mathtt{move(r,1,1,2,1)}}{\mathbf{T}\,\mathtt{move(r,1,1,2,1)}} \right\}$$

can be interpreted "if robotic agent `r` is at position (1,1) and no obstacles are known to be blocking the route from (1,1) to (2,1), then `r` can move to position (2,1)". As expected, applying this default to the theory $\{\mathbf{T}\,\mathtt{at(r,1,1)}\}$ leads to the inference $\{\mathbf{T}\,\mathtt{move(r,1,1,2,1)}\}$, while applying the default to the theory $\{\mathbf{T}\,\mathtt{at(r,1,1)},\ \mathbf{T}\,\mathtt{obstacle(1,1,2,1)}\}$ allows no additional inferences.

6.3.2 Normalising Defaults

Seminormal defaults permit incoherence because applying a seminormal default does not update the knowledge base to reflect one of the reasons for its application, namely the consistency of its justification. In order to overcome this problem we need to be able to update the knowledge base to reflect the outcome of a test of the form "is y possible" without asserting that "y is true". In classical logic we can ascertain that y is possible if we cannot deduce $\neg y$, but the only way to update the KB to reflect this is to make the stronger assertion that y is true. In asserted logic, however, it can be achieved using the normal default $(\,:\mathbf{P}y\,/\,\mathbf{P}y)$. Notice that the consequent of this default will not add any *known* information to the KB, but it will prevent the application of any seminormal default which contradicts $\mathbf{P}y$.

This technique can be used to convert the seminormal default (6.6) into the normal default

$$\frac{\bigwedge_i \mathbf{T}x_i : \bigwedge_j \mathbf{P}y_j \;\wedge\; \bigwedge_k \mathbf{T}z_k}{\bigwedge_j \mathbf{P}y_j \;\wedge\; \bigwedge_k \mathbf{T}z_k} \tag{6.7}$$

We call this process *normalising* the default.

When a normalised default is invoked, the weak assertions $\bigwedge_j \mathbf{P}y_j$ added to the KB reflect the fact that the addition of the strong assertions $\bigwedge_k \mathbf{T}z_k$ depends on the consistency of $\bigwedge_j \mathbf{P}y_j$. The amount of known information in the KB does not increase as a direct result

[2]It is important here not to confuse the source of nonmonotonicity, which comes from the default formalism and not from the **P** 'operator' (recall that AL itself is monotonic).

of normalising the default. In fact, the amount of known information will only increase due to normalising defaults if the KB entails weak to strong implications (which we consider further in the following section). Otherwise the weak assertions serve only to inhibit any contradicting defaults, and so prevent incoherence.

Example 2 Etherington [Eth87] proposes default logic representations for inheritance networks in which default links with exceptions are represented by seminormal defaults. To illustrate this he encodes the following information.

Molluscs are normally Shell-bearers.
Cephalopods must be molluscs but normally are not Shell-bearers.
Nautili must be Cephalopods and Shell-bearers.

The corresponding default theory is

$$\begin{aligned} D &= \left\{ \frac{\mathtt{m(x)} : \mathtt{sb(x)} \wedge \neg\mathtt{c(x)}}{\mathtt{sb(x)}}, \frac{\mathtt{c(x)} : \neg\mathtt{sb(x)} \wedge \neg\mathtt{n(x)}}{\neg\mathtt{sb(x)}} \right\} \\ \Phi &= \{ \mathtt{c(x)} \rightarrow \mathtt{m(x)},\ \mathtt{n(x)} \rightarrow \mathtt{c(x)},\ \mathtt{n(x)} \rightarrow \mathtt{sb(x)} \}. \end{aligned}$$

Adding $\mathtt{n(x)}$ to the theory Φ gives an extension containing $\mathtt{c(x)}$, $\mathtt{m(x)}$ and $\mathtt{sb(x)}$; that is, a Nautilus is known to be a Cephalopod, a Mollusc and a Shell-bearer. On the other hand, adding $\mathtt{c(x)}$ gives an extension containing $\mathtt{m(x)}$ and $\neg\mathtt{sb(x)}$; that is, a Cephalopod (not known to be a Nautilus) is inferred to be a Mollusc with no shell.

We begin by verifying that the AL theory corresponding to Etherington's seminormal default theory produces the same result. The theory is given by

$$\begin{aligned} D &= \left\{ \frac{\mathbf{T}\,\mathtt{m(x)} : \mathbf{T}\,\mathtt{sb(x)} \wedge \neg\mathbf{T}\,\mathtt{c(x)}}{\mathbf{T}\,\mathtt{sb(x)}}, \frac{\mathbf{T}\,\mathtt{c(x)} : \mathbf{T} - \mathtt{sb(x)} \wedge \neg\mathbf{T}\,\mathtt{n(x)}}{\mathbf{T} - \mathtt{sb(x)}} \right\} \\ \Phi &= \{ \mathbf{T}\,\mathtt{c(x)} \rightarrow \mathbf{T}\,\mathtt{m(x)},\ \mathbf{T}\,\mathtt{n(x)} \rightarrow \mathbf{T}\,\mathtt{c(x)},\ \mathbf{T}\,\mathtt{n(x)} \rightarrow \mathbf{T}\,\mathtt{sb(x)} \}. \end{aligned}$$

Adding $\mathbf{T}\,\mathtt{n(x)}$ to Φ gives an extension containing $\mathbf{T}\,\mathtt{c(x)}$, $\mathbf{T}\,\mathtt{m(x)}$ and $\mathbf{T}\,\mathtt{sb(x)}$, while adding $\mathbf{T}\,\mathtt{c(x)}$ to Φ gives an extension containing $\mathbf{T}\,\mathtt{m(x)}$ and $\mathbf{T} - \mathtt{sb(x)}$. These extensions correspond to those above.

Normalising the above defaults gives type (6.7) defaults

$$D = \left\{ \frac{\mathbf{T}\,\mathtt{m(x)} : \mathbf{T}\,\mathtt{sb(x)} \wedge \neg\mathbf{T}\,\mathtt{c(x)}}{\mathbf{T}\,\mathtt{sb(x)} \wedge \neg\mathbf{T}\,\mathtt{c(x)}}, \frac{\mathbf{T}\,\mathtt{c(x)} : \mathbf{T} - \mathtt{sb(x)} \wedge \neg\mathbf{T}\,\mathtt{n(x)}}{\mathbf{T} - \mathtt{sb(x)} \wedge \neg\mathbf{T}\,\mathtt{n(x)}} \right\}.$$

The extension for $\mathbf{T}\,\mathtt{n(x)}$ is the same. The extension for $\mathbf{T}\,\mathtt{c(x)}$, however, contains the additional weak assertion $\neg\mathbf{T}\,\mathtt{n(x)}$. This assertion records the fact that the inference regarding the Cephalopod not being a Shell-bearer depends on it not being a Nautilus. Notice that no other strong assertions are entailed by this weak assertion, so the known information in the extension remains the same. Finally, it should be noted that this theory responds correctly to new information. For example, if $\mathbf{T}\,\mathtt{n(x)}$ is subsequently added to $\Phi \cup \{\mathbf{T}\,\mathtt{c(x)}\}$ then both $\mathbf{T} - \mathtt{sb(x)}$ and $\neg\mathbf{T}\,\mathtt{n(x)}$ disappear from the extension.

Table 6.1 shows the normalised AL representation for all of Etherington's (propositional) link types [Eth87, p56].

Table 6.1: A normalised representation of Etherington's link types for inheritance networks with exceptions.

Name	Symbol	Normalised AL representation
Strict IS-A	$A\cdot \Longrightarrow \cdot B$	$\mathbf{T}A{\rightarrow}\mathbf{T}B$
Strict ISN'T-A	$A\cdot \not\Longrightarrow \cdot B$	$\mathbf{T}A{\rightarrow}\mathbf{F}B$
Default IS-A	$A\cdot \longrightarrow \cdot B$	$\mathbf{T}A : \mathbf{T}B \,/\, \mathbf{T}B$
Default ISN'T-A	$A\cdot \not\longrightarrow \cdot B$	$\mathbf{T}A : \mathbf{F}B \,/\, \mathbf{F}B$
Exceptions	$C_i \cdot - - >$	$\dfrac{\mathbf{T}A : \mathbf{T}B \wedge \neg\mathbf{T}C_1 \wedge \ldots \wedge \neg\mathbf{T}C_n}{\mathbf{T}B \wedge \neg\mathbf{T}C_1 \wedge \ldots \wedge \neg\mathbf{T}C_n}$
		$\dfrac{\mathbf{T}A : \mathbf{F}B \wedge \neg\mathbf{T}C_1 \wedge \ldots \wedge \neg\mathbf{T}C_n}{\mathbf{F}B \wedge \neg\mathbf{T}C_1 \wedge \ldots \wedge \neg\mathbf{T}C_n}$

Example 3 The seminormal default theory

$$\Phi = \{\}, \quad D = \left\{ \frac{: \neg\mathbf{T}y \wedge \mathbf{T}x}{\mathbf{T}x}, \frac{: \neg\mathbf{T}z \wedge \mathbf{T}y}{\mathbf{T}y}, \frac{: \neg\mathbf{T}x \wedge \mathbf{T}z}{\mathbf{T}z} \right\}$$

is incoherent. Normalising gives a coherent theory

$$\Phi = \{\}, \quad D = \left\{ \frac{: \neg\mathbf{T}y \wedge \mathbf{T}x}{\neg\mathbf{T}y \wedge \mathbf{T}x}, \frac{: \neg\mathbf{T}z \wedge \mathbf{T}y}{\neg\mathbf{T}z \wedge \mathbf{T}y}, \frac{: \neg\mathbf{T}x \wedge \mathbf{T}z}{\neg\mathbf{T}x \wedge \mathbf{T}z} \right\}$$

which has three extensions containing $\{\mathbf{T}x, \neg\mathbf{T}y\}$, $\{\mathbf{T}y, \neg\mathbf{T}z\}$ and $\{\mathbf{T}z, \neg\mathbf{T}x\}$ respectively.

As this example shows, incoherence in a seminormal theory will often manifest itself as multiple extensions in a normalised theory. This is not surprising since incoherence, like multiple extensions, is caused by conflicts among defaults. In this example no pair of assertions from $\{\mathbf{T}x, \mathbf{T}y, \mathbf{T}z\}$ can be true at the same time, so it seems reasonable that only one appears in each extension.

There is some doubt as to whether theories like that in Example 3 have practical applications. In most examples from the literature the effect is similar to that shown in Example 2. The extensions are equivalent for seminormal and normalised theories except for the presence of weak assertions. However, from a proof-theoretic point view there is a significant gain in using normalised defaults.

6.4 Nondisjunctive Theories

We now examine the class of theories which is free from unwanted side-effects due to weak assertions. Intuitively these theories must be free of weak to strong implications that can play a part in the deduction of new strong assertions. Formally, it is the class of theories which entails a disjunction of atoms only if it also entails an atom which subsumes it. In other words, it is the class of nondisjunctive theories defined in Section 3.3.

As we explained in Section 3.3, augmenting a nondisjunctive theory by negative literals does not add to the positive literals entailed. This can be formalised by the following theorem. Note that since we are dealing at the level of wffs, the subsequent discussion applies to both classical logic and AL.

Theorem 6.4.1 *If Φ is a nondisjunctive theory then for any positive literal α such that $\Phi \not\vdash \alpha$ there are no positive literals $\beta_1, \ldots, \beta_n$ such that $\Phi \cup \{\neg\beta_1, \ldots, \neg\beta_n\}$ is consistent and $\Phi \cup \{\neg\beta_1, \ldots, \neg\beta_n\} \vdash \alpha$.*

Proof. Assume such an α and $\beta_1, \ldots, \beta_n$ exist. Then $\Phi \cup \{\neg\beta_1, \ldots, \neg\beta_n\} \vdash \alpha$ so from the Deduction Theorem $\Phi \vdash (\neg\beta_1 \rightarrow \ldots \rightarrow (\neg\beta_{n-1} \rightarrow (\neg\beta_n \rightarrow \alpha)) \ldots)$ or equivalently $\Phi \vdash \beta_1 \vee \ldots \vee \beta_{n-1} \vee \beta_n \vee \alpha$. Since Φ is nondisjunctive and $\Phi \not\vdash \alpha$ it follows that $\Phi \vdash \beta_i$ for some i, $1 \leq i \leq n$. But then $\Phi \cup \{\neg\beta_1, \ldots, \neg\beta_n\}$ is inconsistent which contradicts the choice of $\beta_1, \ldots, \beta_n$. □

6.4.1 Horn Theories

In general the conditions for nondisjunctiveness are difficult to ensure. There are, however, classes for which nondisjunctiveness is guaranteed. One important class is Horn theories. (A Horn theory is one which consists solely of Horn clauses; that is, disjunctions of literals $\alpha_1 \vee \ldots \vee \alpha_n$ at most one of which is positive.)

Lemma 6.4.2 *All Horn theories are nondisjunctive.*

Proof. Let Φ be a Horn theory. If Φ is inconsistent then it entails all literals and is clearly nondisjunctive. If Φ is consistent then CWA(Φ) is consistent [Rei78, Thm 6] and therefore Φ is nondisjunctive [She84, Thm 6]. □

The following weak converse to Lemma 6.4.2 also holds.

Lemma 6.4.3 *Any nondisjunctive theory is equivalent to some Horn theory.*

Proof. Let Φ be a nondisjunctive theory. Convert Φ to conjunctive normal form and rewrite as a set of clauses. Replace each clause which contains a disjunction of positive literals by a literal which entails it. In this way all non-Horn clauses can be removed and the resulting theory is Horn. □

Since Horn theories are easier to deal with than general nondisjunctive theories and are equivalent in the above sense, we will extend Theorem 6.4.1 to the desired result for Horn theories.

We call a set D of defaults Horn if for all defaults $(\alpha : \beta \,/\, \gamma) \in D$, γ is a conjunction of Horn clauses. We also make use of the following notation. If D is a set of defaults and $\langle \Phi, \Psi \rangle \in E_D$ then

$$\textsc{gd}(\Psi, D) = \{(\alpha : \beta \,/\, \gamma) \in D \mid \alpha \in \Psi \text{ and } \neg\beta \notin \Psi\}.$$

Furthermore, if Φ is a Horn theory, then $\textsc{heads}(\Phi)$ is the set of positive literals appearing in Φ.

Theorem 6.4.4 *Let D be a set of Horn defaults, Φ be a consistent Horn theory and $\langle \Phi, \Psi \rangle \in E_D$. Then for any positive literal ϕ, if $\Phi \cup \textsc{heads}(\textsc{consequents}(\textsc{gd}(\Psi, D))) \not\vdash \phi$ then $\phi \notin \Psi$.*

Proof. Assume $\Phi \cup$ HEADS(CONSEQUENTS(GD(Ψ, D))) $\nvdash \phi$ and $\phi \in \Psi$. From [Rei80, Thm 2.5], $\Psi = Th(\Phi \cup$CONSEQUENTS(GD(Ψ, D))) and therefore $\Phi \cup$ CONSEQUENTS(GD(Ψ, D)) $\vdash \phi$. Since D is Horn, CONSEQUENTS(GD(Ψ, D)) can be expressed as the union of a set of headed Horn clauses $\Omega = \{\omega_1, \ldots, \omega_m\}$ and a set of headless Horn clauses $\Upsilon = \{\upsilon_1, \ldots, \upsilon_n\}$, where $\Phi \cup \Omega \cup \Upsilon \vdash \phi$. Also, since Φ is consistent Ψ is consistent [Rei80, Cor 2.2], and therefore $\Phi \cup \Omega \cup \Upsilon$ and its subsets are consistent.

For $i = 1, \ldots, n$ each clause $\upsilon_i \in \Upsilon$ is a disjunction of negated literals $\upsilon_i = \neg\upsilon_{i1} \vee \ldots \vee \neg\upsilon_{ik_i}$ for some $k_i > 0$. But $\Phi \cup \Omega \cup \{\upsilon_1, \ldots, \upsilon_{n-1}, \neg\upsilon_{n1} \vee \ldots \vee \neg\upsilon_{nk_n}\}$ is consistent so for some j, $\Phi \cup \Omega \cup \{\upsilon_1, \ldots, \upsilon_{n-1}, \neg\upsilon_{nj}\}$ is consistent. But $\neg\upsilon_{nj} \vdash \upsilon_n$, so if $\Phi \cup \Omega \cup \{\upsilon_1, \ldots, \upsilon_{n-1}, \upsilon_n\} \vdash \phi$ then $\Phi \cup \Omega \cup \{\upsilon_1, \ldots, \upsilon_{n-1}, \neg\upsilon_{nj}\} \vdash \phi$. But $\Phi \cup \Omega \cup \{\upsilon_1, \ldots, \upsilon_{n-1}, \neg\upsilon_{nj}\}$ is consistent and $\Phi \cup \Omega \cup \{\upsilon_1, \ldots, \upsilon_{n-1}\}$ is Horn and therefore nondisjunctive, so from Theorem 6.4.1, $\Phi \cup \Omega \cup \{\upsilon_1, \ldots, \upsilon_{n-1}\} \vdash \phi$. Repeating this argument n times shows that if $\Phi \cup \Omega \cup \Upsilon \vdash \phi$ then $\Phi \cup \Omega \vdash \phi$.

For $i = 1, \ldots, m$ each clause $\omega_i \in \Omega$ can be written $\omega_i = \omega_{i1} \vee \neg\omega_{i2} \vee \ldots \vee \neg\omega_{ik}$ where each ω_{ij}, $0 \leq j \leq k$, is a positive literal and $k \geq 1$. But for each i, $\omega_{i1} \vdash \omega_i$. Thus if $\Phi \cup \Omega \vdash \phi$ then $\Phi \cup \{\omega_{11}, \ldots, \omega_{m1}\} \vdash \phi$. That is, $\Phi \cup$HEADS(CONSEQUENTS(GD(Ψ, D))) $\vdash \phi$; contradiction. $\square$

Since GD(Ψ, D) $\subseteq D$, Theorem 6.4.4 ensures that only the positive literals appearing in the consequents of D (along with Φ) figure in the deduction of positive literals in extensions. For Horn theories and defaults in AL this means that only strong assertions in the consequents (along with Φ) can lead to strong assertions in an extension.

Horn theories and defaults are sufficiently descriptive for many practical applications. Any defaults in the form of (6.6) and (6.7) are Horn, as are the formulas and defaults corresponding to Etherington's link types (Table 6.1) for inheritance networks and the other examples appearing in this chapter.

Chapter 7

Hierarchical Default Logic

In this chapter we address the remaining obstacle to determinism in default logic—the branching or multiple extension problem. Our approach to the problem, called *hierarchical default logic* (HDL) [Mac91a], is based on linearly ordering sets of defaults and subsumes Brewka's *prioritized default logic* [Bre89].

Many authors have proposed modified nonmonotonic formalisms which overcome the branching problem by ordering formulas or interpretations. Examples include *prioritized circumscription* [Lif85], *graded default logic* [FG90], *hierarchic autoepistemic logic* [Kon88a] and *chronological ignorance* [Sho88a]. HDL is attractive because of its compact representation, and more importantly because it inherits (and improves upon) the proof procedures for default logic described in the previous chapter. We show that these procedures are substantially simplified if the defaults are restricted to free defaults.

7.1 Recursive Transfer Relations

We approach the branching problem by decomposing transfer relations into sequences of subrelations with different parameter sets. The subrelations are then recursively applied to the object theory.

Definition 7.1.1 Let R be a relation on $\wp(L)$ for some language L and $P_1, P_2, \ldots$ be sets of parameters for R. A *hierarchical parameter set* is a well-founded linearly ordered set $\boldsymbol{P} = \langle\{P_1, P_2, \ldots\}, \preceq\rangle$. Assume without loss of generality that the parameter sets are labelled so that $P_i \preceq P_j$ if and only if $i \leq j$. Then the *recursion of R over $\boldsymbol{P}$*, denoted $\boldsymbol{R_P}$, is defined by the composite relation

$$\boldsymbol{R_P} = \ldots \circ R_{P_3} \circ R_{P_2} \circ R_{P_1}. \tag{7.1}$$

In practice we will deal with finite hierarchical parameter sets $\boldsymbol{P} = \langle\{P_1, P_2, \ldots, P_n\}, \preceq\rangle$ in which case the recursion relation is defined by $\boldsymbol{R_P} = R_{P_n} \circ \ldots \circ R_{P_1}$.

The decomposition of transfer relations reduces the problem of guaranteeing desirable properties for the overall relation to that of ensuring the properties for each of its components. The following lemma can be verified from the properties of relations and simple set theory.

Lemma 7.1.2 *Let* $\boldsymbol{P} = \langle\{P_1, P_2, \ldots\}, \preceq\rangle$ *be a hierarchical parameter set for* R*. Then*

1. *if each* R_{P_i} *is consistent then* $\boldsymbol{R_P}$ *is consistent,*
2. *if each* R_{P_i} *is total then* $\boldsymbol{R_P}$ *is total,*
3. *if each* R_{P_i} *is nonbranching then* $\boldsymbol{R_P}$ *is nonbranching,*
4. *if each* R_{P_i} *is deterministic and monotonic then* $\boldsymbol{R_P}$ *is monotonic.*
5. *if each* R_{P_i} *is expanding then* $\boldsymbol{R_P}$ *is expanding,*
6. *if each* R_{P_i} *is closing then* $\boldsymbol{R_P}$ *is closing, and*
7. *if each* R_{P_i} *is completing then* $\boldsymbol{R_P}$ *is completing.*

The hierarchy obtained by separating and ordering parameter sets can be used to arbitrate between competing parameters without separating those which are compatible. This hierarchical separation appears to be appropriate for many applications. An example is given in Section 7.3.

Note that we have not specified any particular way in which the hierarchy should be chosen, thus allowing the use of previously proposed strategies such as shortest paths or inferential distance [THT87], specificity [Poo85], causal ordering [Sho88a] or any other method which is appropriate to the language and application.

7.2 Hierarchical Augmentations and Extensions

Hierarchical default logic (HDL) is the recursive version of default logic, in which the object theory is expanded by recursion of the augmentation or extension relations over a linearly ordered set of default sets.

Definition 7.2.1 Let $D_1, D_2, \ldots$ be default sets. A *hierarchical default set* is a well-founded linearly ordered set $\boldsymbol{D} = \langle\{D_1, D_2 \ldots\}, \preceq\rangle$. The *recursive augmentation relation* $\boldsymbol{A_D}$ is the recursion of A over $\boldsymbol{D}$. The *recursive extension relation* $\boldsymbol{E_D}$ is the recursion of E over $\boldsymbol{D}$.

For finite hierarchical default sets the recursive extension relation $\boldsymbol{E_D}$ corresponds to Brewka's prioritized default logic extensions [Bre89].

Algorithm 6.2.1, which specifies how augmentations can be constructed using a classical theorem prover, extends to hierarchical augmentations in a straightforward way. The image theories of each augmentation relation become the object theories of its successor. If the original object theory Φ is finite and the hierarchical default set $\boldsymbol{D}$ is instance finite (that is $\boldsymbol{D}$ is finite and all default sets in $\boldsymbol{D}$ are instance-finite) then all the images will be finite and effectively computable for decidable first-order subclasses.

We now show that Theorem 6.1.3, which relates augmentations and extensions, extends to recursive relations. That is, recursive extension is equivalent to the deductive closure of recursive augmentation. The proof makes use of three intermediate results which adopt the notation of Definition 3.4.1.

Lemma 7.2.2 *A set* Γ *satisfies* (3.2b)–(3.2d) *if and only if it satisfies* (3.2c), (3.2d) *and*

$$Th(\Phi) \subseteq \Gamma. \tag{7.2}$$

Proof. ($\Rightarrow$) Assume Γ satisfies (3.2b)–(3.2d). Then from (3.2b), $\Phi \subseteq \Gamma$ and hence $Th(\Phi) \subseteq Th(\Gamma)$. But from (3.2c), $Th(\Gamma) = \Gamma$ and therefore $Th(\Phi) \subseteq \Gamma$.

($\Leftarrow$) Assume Γ satisfies (7.2), (3.2c) and (3.2d). Clearly Γ also satisfies (3.2b) since $\Phi \subseteq Th(\Phi)$. □

Corollary 7.2.3 Γ *is a minimal set satisfying* (3.2b)–(3.2d) *if and only if* Γ *is a minimal set satisfying* (7.2), (3.2c) *and* (3.2d).

Theorem 7.2.4 *Let* E_D *be defined according to Definition 3.4.1. Then* $\langle \Phi, \Psi \rangle \in E_D$ *if and only if there exists* Θ *such that* $\Theta = Th(\Phi)$ *and* $\langle \Theta, \Psi \rangle \in E_D$. *That is,*

$$E_D = E_D \circ Th. \tag{7.3}$$

Proof. ($\Rightarrow$) Assume $\langle \Phi, \Psi \rangle \in E_D$ and let $\Theta = Th(\Phi)$. Then $\Psi = \Gamma$ where Γ is a minimal set satisfying (3.2b)–(3.2d). From Corollary 7.2.3, Γ is a minimal set satisfying (7.2), (3.2c) and (3.2d) and therefore $\langle Th(\Phi), \Psi \rangle \in E_D$.

($\Leftarrow$) Assume $\Theta = Th(\Phi)$ and $\langle \Theta, \Psi \rangle \in E_D$. Then $\Psi = \Gamma$ where Γ is a minimal set satisfying (7.2), (3.2c) and (3.2d). From Corollary 7.2.3, Γ is a minimal set satisfying (3.2b)–(3.2d) and therefore $\langle \Phi, \Psi \rangle \in E_D$. □

The desired result follows from repeated application of Theorem 7.2.4.

Theorem 7.2.5 *Let* $\boldsymbol{D} = \langle \{D_1, D_2, \ldots\}, \preceq \rangle$ *be a hierarchical default set. Then*

$$\boldsymbol{E_D} = Th \circ \boldsymbol{A_D}. \tag{7.4}$$

Proof. By induction.

1. $E_{D_1} = Th \circ A_{D_1}$ from Theorem 6.1.3.
2. For $i > 1$ assume $E_{D_{i-1}} \circ \ldots \circ E_{D_1} = Th \circ A_{D_{i-1}} \circ \ldots \circ A_{D_1}$. Then

$$\begin{aligned} E_{D_i} \circ E_{D_{i-1}} \circ \ldots \circ E_{D_1} &= E_{D_i} \circ Th \circ A_{D_{i-1}} \circ \ldots \circ A_{D_1} \\ &= E_{D_i} \circ A_{D_{i-1}} \circ \ldots \circ A_{D_1} && \text{[subs. (7.3)]} \\ &= Th \circ A_{D_i} \circ A_{D_{i-1}} \circ \ldots \circ A_{D_1} && \text{[subs. (6.3)]} \end{aligned}$$

□

This result shows that the procedure for generating recursive augmentations provides an extension membership test for recursive extensions. Once again we can generate the augmentation, in this case recursive, and use a classical theorem prover to test extension membership. Note that proof procedures which test extension membership directly from a finite object theory cannot be applied recursively in the same way since the images of each extension relation are infinite. Thus the extension membership test relies on the use of augmentations.

The following lemmas extend two of Reiter's results to recursive augmentations and extensions.

Table 7.1: A summary of properties for default logic and HDL.

	Nmon	**Tot**	**Nbra**	**Con**	**Nclo**	**Exp**	**NComp**
Default Logic, HDL							
E_D, $\boldsymbol{E_D}$	$\surd$			$\surd$		$\surd$	$\surd$
E_D, $\boldsymbol{E_D}$ (normal defaults)	$\surd$	$\surd$		$\surd$		$\surd$	$\surd$
A_D, $\boldsymbol{A_D}$	$\surd$			$\surd$	$\surd$	$\surd$	$\surd$
A_D, $\boldsymbol{A_D}$ (normal defaults)	$\surd$	$\surd$		$\surd$	$\surd$	$\surd$	$\surd$
$\boldsymbol{A_D}$ (normal, singletons)	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$

Key			
Nmon	Nonmonotonic	**Nclo**	Nonclosing
Tot	Total	**Exp**	Expanding
Nbra	Nonbranching	**NComp**	Noncompleting
Con	Consistent		

Lemma 7.2.6 *All recursive augmentations and extensions preserve consistency.*

Proof. Follows from [Rei80, Cor 2.2], Theorem 6.1.3 and Lemma 7.1.2. □

Lemma 7.2.7 *If a hierarchical default set $\boldsymbol{D}$ is normal (that is each D in $\boldsymbol{D}$ contains only normal defaults) then $\boldsymbol{A_D}$ and $\boldsymbol{E_D}$ are total relations on* $\mathrm{L_S}$.

Proof. Follows from [Rei80, Thm 3.1], Theorem 6.1.3 and Lemma 7.1.2. □

Other properties of augmentations and extensions extend to their recursive counterparts in a similar manner according to Lemma 7.1.2. A summary of the various properties is shown in Table 7.1.

The final requirement for determinism, that the relations are nonbranching, can be trivially achieved by restricting each set of defaults in $\boldsymbol{D}$ to a single default; in effect imposing a total order on the defaults. In many applications however a hierarchical structure is more appropriate and other methods of preventing branching at each step in the recursion need to be sought.

7.3 Example: The Yale Shooting Problem

As a solution to the frame problem in the situation calculus (see Chapter 2) McCarthy [McC86] proposed including a general frame axiom schema which can be written in the form

$$\mathtt{t}(f,s) \wedge \neg\mathtt{ab}(f,e,s) \rightarrow \mathtt{t}(f,\mathtt{result}(e,s))$$

and circumscribing over $\neg$`ab`, thus minimising the changes to facts. This schema conveys the information that any fact f persists over successive states unless the event e terminates it (the "abnormal" case).

Hanks and McDermott [HM86] show by means of an example, commonly called the *Yale shooting problem*, that this proposal can lead to the occurrence of unintuitive (or *anomalous*) inferences. The problem is not confined to circumscription but applies generally to branching transfer relations. We illustrate the problem using default logic.

The idea behind the shooting problem is that an infallible sharpshooter lies in wait for a victim. Initially the sharpshooter's gun is loaded and the victim is alive. After some delay the sharpshooter fires the gun. The question is whether or not the victim survives. The task of the frame axioms is to preserve the facts stating that the gun is loaded and the victim is alive until the gun is fired.

Hanks and McDermott's argument can be illustrated by the following theory:

$$\begin{aligned}\Phi \; = \; \{&\mathtt{t}(\mathtt{loaded},\mathtt{s1}),\\ &\mathtt{t}(\mathtt{alive},\mathtt{s2}),\\ &\mathtt{t}(\mathtt{loaded},\mathtt{s1}) \wedge \neg\mathtt{ab}(\mathtt{loaded},\mathtt{wait},\mathtt{s1}) \rightarrow \mathtt{t}(\mathtt{loaded},\mathtt{s2}),\\ &\mathtt{t}(\mathtt{alive},\mathtt{s2}) \wedge \neg\mathtt{ab}(\mathtt{alive},\mathtt{shoot},\mathtt{s2}) \rightarrow \mathtt{t}(\mathtt{alive},\mathtt{s3}),\\ &\mathtt{t}(\mathtt{loaded},\mathtt{s2}) \rightarrow \mathtt{ab}(\mathtt{alive},\mathtt{shoot},\mathtt{s2}),\\ &\mathtt{t}(\mathtt{loaded},\mathtt{s2}) \rightarrow \mathtt{t}(\mathtt{dead},\mathtt{s3})\}\end{aligned}$$

In this theory `s2` represents the state `result(wait,s1)` and `s3` represents the state `result(shoot,s2)`. The third and fourth sentences of Φ are instances of the frame axiom. The fifth sentence states that a loaded gun causes an exception to the second frame axiom—that is, the victim may not remain alive if the loaded gun is fired.

In order to minimise abnormality the following defaults are used:

$$\begin{aligned}D \; = \; \{ &: \neg\mathtt{ab}(\mathtt{loaded},\mathtt{wait},\mathtt{s1}) \,/\, \neg\mathtt{ab}(\mathtt{loaded},\mathtt{wait},\mathtt{s1}),\\ &: \neg\mathtt{ab}(\mathtt{alive},\mathtt{shoot},\mathtt{s2}) \,/\, \neg\mathtt{ab}(\mathtt{alive},\mathtt{shoot},\mathtt{s2})\}.\end{aligned}$$

The theory has two augmentations. The intuitively correct augmentation includes the sentence $\neg\mathtt{ab}(\mathtt{loaded},\mathtt{wait},\mathtt{s1})$ and entails $\mathtt{t}(\mathtt{dead},\mathtt{s3})$; that is, the victim dies as expected. The anomalous augmentation includes $\neg\mathtt{ab}(\mathtt{alive},\mathtt{shoot},\mathtt{s2})$ and entails $\mathtt{t}(\mathtt{alive},\mathtt{s3})$; that is, the gun becomes mysteriously unloaded and the victim survives.

This problem can be avoided using HDL by choosing the default hierarchy so that normality assumptions are made in chronological (or causal) order. In other words, we make the sequence of default sets correspond to the sequence of states. In this case the appropriate hierarchy is given by $\boldsymbol{D} = \langle\{D_1, D_2\}, \preceq\rangle,\; D_1 \preceq D_2$ where

$$\begin{aligned}D_1 \; &= \{ : \neg\mathtt{ab}(\mathtt{loaded},\mathtt{wait},\mathtt{s1}) \,/\, \neg\mathtt{ab}(\mathtt{loaded},\mathtt{wait},\mathtt{s1})\}, \text{ and}\\ D_2 \; &= \{ : \neg\mathtt{ab}(\mathtt{alive},\mathtt{shoot},\mathtt{s2}) \,/\, \neg\mathtt{ab}(\mathtt{alive},\mathtt{shoot},\mathtt{s2})\}.\end{aligned}$$

It is easily verified that Φ has a single augmentation under $\boldsymbol{A_D}$ which provides the intuitively correct solution.

Of course, this scheme is not on its own sufficient to guarantee a unique solution when there is more than one normality assumption for each state. However it reduces the problem to that of ensuring a unique augmentation based on the assumptions in each state. One way to achieve this is to impose further restrictions on allowable sentences in the style of Shoham's causal theories [Sho88a]. We adopt this approach in the following chapter to give a proof theory for Shoham's chronological ignorance framework.

7.4 Free Default Theories

Reiter argues that most realistic problems can be tackled using only normal defaults [Rei80]. In problems which require seminormal defaults (see for example [Eth87, RC81]) the technique described in the previous chapter can be used to normalise the defaults. In fact many practical reasoning tasks can be handled using only prerequisite free normal defaults, or *free* defaults. This is evidenced by the fact that many nonmonotonic reasoning formalisms, such as the closed world assumption [Rei78] and chronological ignorance [Sho88a], have no mechanism for incorporating prerequisites. The normality assumptions corresponding to the frame axioms in the previous section are typical examples. The application of the assumptions depends only on their consistency and not on any other preconditions. A transformation from normal to free defaults is proposed in [BQQ83].

Since free defaults are a subclass of normal defaults they adopt the desirable properties of normal defaults, such as totality and semimonotonicity. However free defaults permit a more economical proof theory. In the following discussion we abbreviate free defaults by writing their consequents only. Thus a free default γ is shorthand for $:\gamma \,/\, \gamma$.

The definition of augmentations (Def. 6.1.2) simplifies for free defaults as follows.

Lemma 7.4.1 *Let F be a set of (closed) free defaults. Then $\langle \Phi, \Theta \rangle \in A_F$ if and only if*

$$\Theta \quad = \quad \Phi \cup \{\gamma \mid \gamma \in F \ \text{ and } \Theta \nvdash \neg\gamma\}\,. \tag{7.5}$$

Proof. For free defaults the r.h.s. of conditions (6.2a) and (6.2b) are identical. Therefore any set satisfying (6.2a) cannot have a strict subset satisfying (6.2b) and must be an augmentation. Condition (6.2a) reduces to (7.5). □

Algorithm 6.2 for generating defaults simplifies to two steps accordingly.

Notice that the right hand side of (7.5) is simply a consistency requirement; that is, for all $\gamma \in F$ either $\gamma \in \Theta$ or $\Theta \cup \gamma$ is inconsistent.

Corollary 7.4.2 *Let F be a set of free defaults and Φ be consistent. Then Θ is an augmentation of Φ if and only if Θ is a maximal (with respect to set inclusion) consistent subset of $\Phi \cup F$ containing Φ.*

In particular, if F is a singleton then the augmentation relation is described by the function

$$A_F(\Phi) = \begin{cases} \Phi \cup \{\gamma\} & \text{if } \Phi \cup \{\gamma\} \text{ is consistent,} \\ \Phi & \text{otherwise.} \end{cases}$$

Similarly, the augmentation relation for hierarchical default sets containing only singletons reduces to the following.

Corollary 7.4.3 *Let $\boldsymbol{F} = \langle \{\{\gamma_1\}, \{\gamma_2\}, \ldots\}, \preceq \rangle$ be a (countably infinite) hierarchical free default set where $\{\gamma_1\} \preceq \{\gamma_2\} \preceq \cdots$. Then $\boldsymbol{A_F}(\Phi) = \Theta_\infty$ where*

$$\Theta_0 = \Phi$$

and for $i = 1, 2, \ldots$

$$\Theta_i = \begin{cases} \Theta_{i-1} \cup \{\gamma_i\} & \text{if } \Theta_{i-1} \cup \{\gamma_i\} \text{ is consistent,} \\ \Theta_{i-1} & \text{otherwise.} \end{cases}$$

Corollary 7.4.3 describes a simple construction for augmentations in which formulas are sequentially either added to the knowledge base or rejected based on a consistency test. The cost of the algorithm (neglecting the consistency test) grows linearly with the number of defaults. The overall complexity is therefore dominated by the consistency test rather than the default reasoning aspect. The following theorem shows that this construction can be used to generate the augmentations for finite free default sets.

Theorem 7.4.4 *Let $F = \{\gamma_1, \ldots, \gamma_n\}$ be a finite free default set. Then $\langle \Phi, \Theta \rangle \in A_F$ if and only if $\Theta = \boldsymbol{A_F}(\Phi)$ where $\boldsymbol{F} = \langle \{\{\gamma_1\}, \ldots, \{\gamma_n\}\}, \preceq \rangle$ for some linear order $\preceq$.*

Proof. If Φ is inconsistent the result is trivial. Assume Φ is consistent.

($\Rightarrow$) Choose $\preceq$ so that for any $\gamma_i, \gamma_j \in F$, if $\gamma_i \in A_F(\Phi)$ and $\gamma_j \notin A_F(\Phi)$ then $\{\gamma_i\} \preceq \{\gamma_j\}$. The construction of $\boldsymbol{A_F}(\Phi)$ given in Corollary 7.4.3 will add only those defaults which appear in $A_F(\Phi)$ since $A_F(\Phi)$ is maximally consistent in the sense of Corollary 7.4.2.

($\Leftarrow$) Let $\Theta = \boldsymbol{A_F}(\Phi)$ for some $\preceq$. For each default $\gamma \in F$,

1. if $\gamma \in \Theta$ then $\Theta \not\vdash \neg\gamma$ since Θ is consistent by Lemma 7.2.6, and

2. if $\gamma \notin \Theta$ then from Corollary 7.4.3, $\Theta_i \vdash \neg\gamma$ for some i, $0 \leq i \leq n-1$, and since $\Theta_i \subseteq \Theta$, $\Theta \vdash \neg\gamma$.

Thus $\Theta = \Phi \cup \{\gamma \mid \gamma \in F \text{ and } \Theta \not\vdash \neg\gamma\}$ and hence $\langle \Phi, \Theta \rangle \in A_F$ by Lemma 7.4.1. □

In the worst case we must try each ordering $\preceq$ in order to generate all the images of Φ under A_F. However if we require only a single augmentation then a single ordering can be chosen. In particular, if it can be guaranteed that A_F is nonbranching then any ordering can be used in the construction.

Corollary 7.4.5 *Let F and $\boldsymbol{F}$ be defined according to Theorem 7.4.4. If A_F is nonbranching then $A_F(\Phi) = \boldsymbol{A_F}(\Phi)$ for any $\preceq$.*

Proof. From Theorem 7.4.4, $\langle \Phi, \boldsymbol{A_F}(\Phi) \rangle \in A_F$ for any order $\preceq$. Since A_F is nonbranching, $A_F(\Phi) = \boldsymbol{A_F}(\Phi)$. □

The construction in Corollary 7.4.3 can therefore be used for hierarchical default sets containing nonsingletons providing each application of defaults leads to a single augmentation. We simply split each default set up into singletons, imposing an arbitrary ordering between defaults from the same set. In the shooting problem, for example, if it can be guaranteed that the normality assumptions for each state are compatible, then Corollary 7.4.3 can be used to compute the augmentations.

7.4.1 Further Computational Short-cuts

Corollary 7.4.3 provides a procedure for generating the augmentations of hierarchical free default theories with unique augmentations and hence determining whether any formula is a member of the corresponding extensions. However it is not always necessary to compute the entire augmentation in order to prove extension membership. The following lemma shows that if a formula can be deduced from a subset of the augmentation then it is included in the extension. The results adopt the notation of Corollary 7.4.3.

Lemma 7.4.6 *For any wff α, if $\Theta_i \vdash \alpha$ for some i then $\alpha \in \boldsymbol{E_F}(\Phi)$.*

Lemma 7.4.6 shows that Corollary 7.4.3 provides a semidecidable extension membership test for countably infinite default sets in decidable languages. If a formula is contained in the extension then eventually the construction will generate a set from which it can be deduced.

As with all nonmonotonic reasoning systems, if new facts are added to the theory (due to say, further observations) the extension may change. In this case it is undesirable to have to recompute the entire augmentation. The following lemma allows a backtracking scheme.

Lemma 7.4.7 *For any set of wffs Φ', if $\Theta_k \cup \Phi'$ is consistent for some k, $0 \leq k \leq n$, then $\Theta'_n = \boldsymbol{A_F}(\Phi \cup \Phi')$ can be constructed as follows.*

For $0 \leq i \leq k$

$$\Theta'_i = \Theta_i \cup \Phi'$$

and for $k+1 \leq i \leq n$

$$\Theta'_i = \begin{cases} \Theta'_{i-1} \cup \{\gamma_i\} & \text{if } \Theta'_{i-1} \cup \{\gamma_i\} \text{ is consistent,} \\ \Theta'_{i-1} & \text{otherwise.} \end{cases}$$

Thus, when we add new information to the object theory, we only need to recalculate the augmentation from the point where the inconsistency manifests itself. Of course finding this point may in itself be expensive, but the expense can be substantially reduced using simple heuristics. For example, it may be the case that most new observations do not contradict the current extension. In this case we have $k = n$ and the following corollary applies.

Corollary 7.4.8 *If $\alpha \in \boldsymbol{E_F}(\Phi)$ and Φ' is a set of wffs such that $\boldsymbol{A_F}(\Phi) \cup \Phi'$ is consistent then $\alpha \in \boldsymbol{E_F}(\Phi \cup \Phi')$.*

This result shows that the formalism is monotonic with regard to the addition of consistent formulas. For example if we wish to add an observation β where $\boldsymbol{A_F}(\Phi) \nvdash \neg\beta$ then the augmentation becomes $\boldsymbol{A_F}(\Phi) \cup \{\beta\}$ and all formulas previously in the extension remain.

7.5 Comparison with AEL and HAEL

Konolige [Kon88b] has established an equivalence between default theory extensions and strongly grounded autoepistemic logic (AEL) extensions. The equivalence result can be expressed in terms of transfer relations as follows.

Definition 7.5.1 Let D be a set of defaults. Then the *autoepistemic transformation* on $\wp(\mathrm{L_S})$, denoted AE_D, is defined by

$$AE_D : \Phi \mapsto \Phi \cup \{(\Box\alpha \wedge \neg\Box\neg\beta) \rightarrow \gamma \mid (\alpha : \beta \,/\, \gamma) \in D\}. \tag{7.6}$$

The *kernel* of an AEL theory Φ, denoted $KRN(\Phi)$, is the set of ordinary sentences appearing in Φ.

Theorem 7.5.2 [Kon88b, Thm 5.5] *Let E_{sg} denote the strongly grounded extension relation (see Section 3.5). Then*

$$E_D = KRN \circ E_{\mathrm{sg}} \circ AE_D \tag{7.7}$$

The AEL relation corresponding to the recursive extension relation is therefore defined as follows.

Corollary 7.5.3 *Let $\boldsymbol{D}$ be a hierarchical default set. Then*

$$\boldsymbol{E_D} = \ldots \circ KRN \circ E_{\mathrm{sg}} \circ AE_{D_2} \circ KRN \circ E_{\mathrm{sg}} \circ AE_{D_1} \tag{7.8}$$

The resulting extensions differ markedly from the extensions of Konolige's *hierarchic autoepistemic theories* [Kon88a]. In the latter case the problems of partial and branching relations associated with autoepistemic *stable sets* (see Section 3.5) are avoided by applying belief operators only to subtheories lower in the hierarchy. This alters the nature of the self-belief operator and sacrifices the consistency preservation property.

Chapter 8

Chronological Augmentation

In this chapter we provide a proof theory for chronological ignorance based on hierarchical default logic. We verify that, as expected, the HDL formalism is deterministic when restricted to causal theories and is sound and complete with respect to causal CI. We then consider the assumptions which are implicit in causal CI and argue that more intuitive representations can be provided using explicit assumptions in HDL. In particular we show that adopting an explicit representation can remove the need for an epistemic logic. Finally we discuss the qualification problem in relation to chronological ignorance.

8.1 A Proof Theory for Chronological Ignorance

We consider a proof theory for chronological ignorance to be a syntactic method (based on classical proof theory) for transforming object theories Φ into image theories $K_{\mathrm{TAL}}(\Phi)$ (or $Th \circ K_{\mathrm{TAL}}(\Phi)$) as defined semantically in Chapter 5. The proof theory is sound and complete if it specifies all such image theories for Φ and no others.

Intuitively the required image theories can be found by stepping forward through time looking for base sentences whose truth value is not uniquely constrained, and adding sentences which constrain their truth values to u. These base sentences can be found (using consistency tests) without reference to the semantics. Eventually this process will constrain the possible truth valuations to a single one (not necessarily unique for noncausal theories since there may be a choice of truth values between base sentences with the same *ltp*). We conjecture that this will be a cmi truth valuation for arbitrary object theories, although we only prove it to be the case for causal theories.

This method for generating the formulas entailed by cmi truth valuations can be formalised using hierarchical default logic. In order to force the values of unconstrained base sentences to the value u we create a hierarchical free default set which contains the assumption $\mathbf{U}x$ for every atomic base sentence x. These assumptions are applied in order of increasing time by organising the assumptions into sets with the same *ltp*, and arranging the sets in chronological order. We call this approach *chronological augmentation* (CA).

The hierarchical default set must be well-founded and therefore must have a least element. We allocate to this set all the assumptions with *ltp* prior to some time point. Since the assumptions in this set will not be applied in chronological order, this time point must be chosen so that the set does not contain any conflicting defaults. This is achieved by choosing a sufficiently early time point. For arbitrary finite theories we can choose any time point preceding the earliest *ltp* appearing in the theory. The truth-values for all base formulas with *ltp* prior to that point will be unconstrained and therefore all of the assumptions can be consistently applied. For causal theories we can choose any time point which precedes the *ltp* of all boundary conditions in the theory. Such a time point is guaranteed to exist due to the first condition of Definition 5.3.2.

The formal definition of the required hierarchical default set for causal theories follows. Recall that $\mathbf{U}x$ is an abbreviation for $\neg\mathbf{T}x \wedge \neg\mathbf{T}{-}x$.

Definition 8.1.1 Let Φ be a causal theory and t_0 be a time point such that for all boundary conditions $\beta \in \Phi$, $t_0 < \mathit{ltp}(\beta)$. A set of *u-defaults* for Φ is a hierarchical free default set $\boldsymbol{U} = \langle\{U_0, U_1, U_2, \ldots\}, \preceq\rangle$ where

$$U_0 = \{\mathbf{U}x \mid x \text{ is an atomic base sentence and } \mathit{ltp}(x) \leq t_0\}$$

and for all $i > 0$, $i \in \mathbb{N}$,

$$U_i = \{\mathbf{U}x \mid x \text{ is an atomic base sentence and } \mathit{ltp}(x) = t_0 + i\},$$

and $U_j \preceq U_k$ if and only if $j \leq k$.

The following theorem verifies that the augmentation of a causal theory over its u-defaults is nonbranching and precisely matches its cmi truth valuation.

Theorem 8.1.2 *Let Φ be a causal theory with cmi truth valuation σ_{cmi} and $\boldsymbol{U}$ be a set of u-defaults for Φ. Let $\boldsymbol{A_U}$ be the recursive augmentation over $\boldsymbol{U}$ on $\wp(\text{TAL})$. Then*

1. *$\boldsymbol{A_U}$ is deterministic,*
2. *$\sigma_{\text{cmi}} \models \boldsymbol{A_U}(\Phi)$, and*
3. *σ_{cmi} is the only truth valuation satisfying $\boldsymbol{A_U}(\Phi)$.*

Proof. Lemma 7.2.6 shows that $\boldsymbol{A_U}$ preserves consistency, and Lemma 7.2.7 shows that $\boldsymbol{A_U}$ is a total relation since $\boldsymbol{U}$ contains only normal defaults.

To prove items 1 and 2 it remains to show that $\boldsymbol{A_U}$ is nonbranching and $\sigma_{\text{cmi}} \models \boldsymbol{A_U}(\Phi)$. The proof is by induction on increasing time (indicated by the subscript i) and follows the construction and notation of Theorem 5.4.1. Note that in (5.2a) and (5.2b), if $\sigma \models \alpha$ for some $\sigma \in \Sigma_{i-1}$ then $\sigma \models \alpha$ for all $\sigma \in \Sigma_{i-1}$ since $\mathit{ltp}(\alpha) < t_i$ and all $\sigma \in \Sigma_{i-1}$ agree on truth assignments to formulas whose $\mathit{ltp} < t_i$.

1. *(boundary step)* For all x such that $\mathbf{U}x \in U_0$, $x^{\sigma_{\text{cmi}}} = u$ and therefore $\sigma_{\text{cmi}} \models \mathbf{U}x$. From Theorem 5.4.1, $\sigma_{\text{cmi}} \models \Phi$ and hence $\sigma_{\text{cmi}} \models \Theta_0$ where $\Theta_0 = \Phi \cup U_0$. Θ_0 is therefore a maximally consistent subset of $\Phi \cup U_0$ and from Corollary 7.4.2, A_{U_0} is nonbranching and $A_{U_0}(\Phi) = \Theta_0$.

   Finally, since $\mathbf{U}x \in \Theta_0$ for each atomic base sentence x whose $\mathit{ltp} \leq t_0$, the truth valuations which satisfy Θ_0 assign u to these sentences and are therefore a subset of Σ_0.

2. *(induction step)* For $i \geq 1$ assume $A_{U_0}, \ldots, A_{U_{i-1}}$ are nonbranching, $\sigma_{\text{cmi}} \models \Theta_{i-1}$ and $\{\sigma \mid \sigma \models \Theta_{i-1}\} \subseteq \Sigma_{i-1}$ where $\Theta_{i-1} = A_{U_{i-1}} \circ \ldots \circ A_{U_0}(\Phi)$.

   Consider all atomic base sentences x such that $\mathbf{U}x \in U_i$.

   (a) If $x^{\sigma_{\text{cmi}}} = t$ then from (5.2a), $\alpha \rightarrow \mathbf{T}x \in \Phi$ and for all $\sigma \in \Sigma_{i-1}$, $\sigma \models \alpha$. But by assumption $\sigma \models \Theta_{i-1}$ only if $\sigma \in \Sigma_{i-1}$ and hence $\Theta_{i-1} \models \alpha$. Thus $\Theta_{i-1} \vdash \alpha$, $\Theta_{i-1} \vdash \mathbf{T}x$, and $\Theta_{i-1} \cup \{\mathbf{U}x\}$ is inconsistent.

   (b) If $x^{\sigma_{\text{cmi}}} = f$ then from (5.2b), $\alpha \rightarrow \mathbf{T}{-}x \in \Phi$ and for all $\sigma \in \Sigma_{i-1}$, $\sigma \models \alpha$. But by assumption $\sigma \models \Theta_{i-1}$ only if $\sigma \in \Sigma_{i-1}$ and hence $\Theta_{i-1} \models \alpha$. Thus $\Theta_{i-1} \vdash \alpha$, $\Theta_{i-1} \vdash \mathbf{T}{-}x$, and $\Theta_{i-1} \cup \{\mathbf{U}x\}$ is inconsistent.

   Let $\{x_1, x_2, \ldots\}$ be the remaining atomic base sentences not satisfying (a) or (b), that is those assigned u by σ_{cmi}. Then $\sigma_{\text{cmi}} \models \{\mathbf{U}x_1, \mathbf{U}x_2, \ldots\}$ and since $\sigma_{\text{cmi}} \models \Theta_{i-1}$, $\sigma_{\text{cmi}} \models \Theta_i$ where $\Theta_i = \Theta_{i-1} \cup \{\mathbf{U}x_1, \mathbf{U}x_2, \ldots\}$. Θ_i is therefore consistent. From (a) and (b), Θ_i is a unique maximally consistent subset of $\Theta_{i-1} \cup U_i$ containing Θ_{i-1} and hence from Corollary 7.4.2, A_{U_i} is nonbranching and $A_{U_i}(\Theta_{i-1}) = A_{U_i} \circ \ldots \circ A_{U_0}(\Phi) = \Theta_i$.

   Finally, we have $\sigma \models \Theta_{i-1}$ only if $\sigma \in \Sigma_{i-1}$. But for each atomic base sentence x whose $ltp = t_i$, if $x^{\sigma_{\text{cmi}}} = t$ then $\Theta_i \vdash \mathbf{T}x$, if $x^{\sigma_{\text{cmi}}} = f$ then $\Theta_i \vdash \mathbf{T}{-}x$ and if $x^{\sigma_{\text{cmi}}} = u$ then $\mathbf{U}x \in \Theta_i$. Therefore the truth valuations which satisfy Θ_i are a subset of those satisfying Θ_{i-1} which also satisfy (5.2a)–(5.2c), and are thus a subset of Σ_i.

To prove item 3 assume $\sigma \models \boldsymbol{A_U}(\Phi)$ but $\sigma \neq \sigma_{\text{cmi}}$. For any atomic base sentence x exactly one of the following holds; $x^{\sigma}_{\text{cmi}} = t$ in which case $\boldsymbol{A_U}(\Phi) \vdash \mathbf{T}x$ from 2(a) and $x^{\sigma} = t$; $x^{\sigma}_{\text{cmi}} = f$ in which case $\boldsymbol{A_U}(\Phi) \vdash \mathbf{T}{-}x$ from 2(b) and $x^{\sigma} = f$; or $x^{\sigma}_{\text{cmi}} = u$ in which case $\mathbf{U}x \in \boldsymbol{A_U}(\Phi)$ and $x^{\sigma} = u$. Since σ and σ_{cmi} agree on assignments to all atomic base sentences they must agree on assignments to all sentences according to Definition 4.3.1—contradiction. □

Corollary 8.1.3 *Let Φ be a causal theory with u-defaults $\boldsymbol{U}$. Then for any formula α, $\alpha \in \boldsymbol{E_U}(\Phi)$ if and only if $\alpha \in Th(K_{\text{TAL}}(\Phi))$. That is,*

$$\boldsymbol{E_U} = Th \circ K_{\text{TAL}} \tag{8.1}$$

where $\boldsymbol{E_U}$ is restricted to causal theories.[1]

Proof. For any formula α, if $\sigma_{\text{cmi}} \models \alpha$ then from Theorem 8.1.2, $\boldsymbol{A_U}(\Phi) \models \alpha$ and hence $\boldsymbol{A_U}(\Phi) \vdash \alpha$. Similarly, if $\boldsymbol{A_U}(\Phi) \vdash \alpha$ then $\boldsymbol{A_U}(\Phi) \models \alpha$ and from Theorem 8.1.2, $\sigma_{\text{cmi}} \models \alpha$. Thus $\sigma_{\text{cmi}} \models \alpha$ iff $\alpha \in Th(\boldsymbol{A_U}(\Phi))$. From Definition 5.4.2, $\mathbf{T}x \in K_{\text{TAL}}(\Phi)$ iff x is a base literal and $\sigma_{\text{cmi}} \models \mathbf{T}x$. Therefore, for any formula α, $\sigma_{\text{cmi}} \models \alpha$ iff $K_{\text{TAL}}(\Phi) \vdash \alpha$, or $\alpha \in Th(K_{\text{TAL}}(\Phi))$. □

Note that $K_{\text{TAL}}(\Phi)$ is simply the set of formulas $\mathbf{T}x$ in $\boldsymbol{E_U}(\Phi)$ where x is a base literal.

---

[1]Strictly speaking we should restrict the domain by using the relation $E_U \cap causal(\wp(\text{TAL})) \times \wp(\text{TAL})$.

Our (syntactic) HDL formalism with the appropriate defaults is therefore sound and complete with respect to the (semantic) CI formalism described in Chapter 5.

An immediate advantage of chronological augmentation over CI is that the restriction to causal theories can be relaxed somewhat. Since augmentation is concerned only with deductions from an object theory (and not the syntactic form of the theory) we require only that the object theory is *tautologically equivalent* to a causal theory. We can therefore build up various alternative sentence constructions. For example, if

$$\begin{aligned} \alpha &\rightarrow \beta_1 \\ \alpha &\rightarrow \beta_2 \\ &\vdots \end{aligned}$$

are causal sentences then the sentence

$$\alpha \rightarrow \beta_1 \wedge \beta_2 \wedge \cdots$$

can also be used in causal theories. Similarly, if

$$\begin{aligned} \alpha_1 &\rightarrow \beta \\ \alpha_2 &\rightarrow \beta \\ &\vdots \end{aligned}$$

are causal sentences then the sentence

$$\alpha_1 \vee \alpha_2 \vee \cdots \rightarrow \beta$$

can be used in causal theories. While our subsequent discussion continues to refer to causal theories in the strict sense it should be remembered that in practice we can substitute any equivalent theory.

8.2 Finite Causal Theories

If we restrict our attention to a part of the language TAL containing only a finite number of atomic base sentences then the set of u-defaults is finite and the augmentation $\boldsymbol{A_U}$ can be effectively computed. This can be achieved by considering only that part of TAL which consists of sentences constructed from the atomic base sentences appearing in the (finite) object theory. Since the truth values of all the atomic base sentences which do not appear in the object theory are unconstrained, these base sentences are all assigned the value u by the cmi truth valuation for that theory. The following theorem verifies that with this restriction we only need to consider the u-defaults containing the atomic base sentences in the object theory.

Theorem 8.2.1 *Let Φ be a causal theory with u-defaults $\boldsymbol{U}$. Let $\boldsymbol{U}_a$ be the hierarchical default set obtained from $\boldsymbol{U}$ by removing all defaults $\mathbf{U}x$ such that x does not appear in Φ. Then for any base sentence z appearing in Φ, $\mathbf{T}z \in \boldsymbol{E_U}(\Phi)$ (respectively $\mathbf{F}z \in \boldsymbol{E_U}(\Phi)$, $\mathbf{U}z \in \boldsymbol{E_U}(\Phi)$) if and only if $\mathbf{T}z \in \boldsymbol{E}_{\boldsymbol{U}_a}(\Phi)$ (respectively $\mathbf{F}z \in \boldsymbol{E}_{\boldsymbol{U}_a}(\Phi)$, $\mathbf{U}z \in \boldsymbol{E}_{\boldsymbol{U}_a}(\Phi)$).*

Proof. The proof is by induction on increasing time (indicated by the subscript i) following the notation of Theorem 5.4.1. We show that for any atomic base sentence x appearing in Φ we can deduce $\mathbf{T}x$, $\mathbf{F}x$ or $\mathbf{U}x$ from $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi)$ according to whether σ_{cmi} assigns x the value t, f or u respectively. Thus $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi)$ agrees with $\boldsymbol{A}_{\boldsymbol{U}}(\Phi)$ on all base sentences appearing in Φ and their tautological consequences.

1. *(boundary step)* $\boldsymbol{A}_{U_0} = \Phi \cup U_0$ is consistent and $U_{a0} \subseteq U_0$. Therefore $\Phi \cup U_{a0}$ is consistent and $\boldsymbol{A}_{U_{a0}} = \Phi \cup U_{a0}$. Thus for all atomic base sentences x appearing in Φ whose *ltp* $\leq t_0$, $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \mathbf{U}x$. Also from Theorem 5.4.1, $x^{\sigma_{\text{cmi}}} = u$.

2. *(induction step)* For $i > 1$, assume for all atomic base sentences x appearing in Φ whose *ltp* $< t_i$,

$$\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \begin{cases} \mathbf{T}x & x^{\sigma_{\text{cmi}}} = t, \\ \mathbf{F}x & x^{\sigma_{\text{cmi}}} = f, \\ \mathbf{U}x & x^{\sigma_{\text{cmi}}} = u. \end{cases}$$

   For any formula $\alpha \rightarrow \mathbf{T}z$ whose *ltp* $= t_i$, α is of the form

$$\bigwedge_{k=1}^{m} \mathbf{T}w_k \;\wedge\; \bigwedge_{j=1}^{n} \mathbf{P}y_j \qquad m, n \in \mathbb{N}$$

   where w_k, y_j $(k, j > 0)$ are base literals such that

   (a) *ltp*$(w_k) < t_i$ for $k = 1, \ldots, m$, and

   (b) *ltp*$(y_j) < t_i$ for $j = 1, \ldots, n$.

   Therefore $\alpha^{\sigma_{\text{cmi}}} = t$ if and only if both $x_k^{\sigma_{\text{cmi}}} = t$ for $k = 1, \ldots, m$ and $y_j^{\sigma_{\text{cmi}}} \neq f$ for $j = 1, \ldots, n$. But by assumption $x_k^{\sigma_{\text{cmi}}} = t$ if and only if $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \mathbf{T}x_k$. Also $y_j^{\sigma_{\text{cmi}}} = t$ (respectively $y_j^{\sigma_{\text{cmi}}} = u$) if and only if $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \mathbf{T}y_j$ (respectively $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \mathbf{U}y_j$) and hence $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \mathbf{P}y_j$. Thus $\alpha^{\sigma_{\text{cmi}}} = t$ if and only if $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \alpha$.

   Now for any atomic base sentence x appearing in Φ whose *ltp* $= t_i$:

   (a) If $x^{\sigma_{\text{cmi}}} = t$ then from Theorem 5.4.1 there exists $\alpha \rightarrow \mathbf{T}x \in \Phi$ such that *ltp*$(\alpha) < t_i$ and $\alpha^{\sigma_{\text{cmi}}} = t$. Therefore $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \alpha$ and $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \mathbf{T}x$.

   (b) If $x^{\sigma_{\text{cmi}}} = f$ then from Theorem 5.4.1 there exists $\alpha \rightarrow \mathbf{F}x \in \Phi$ such that *ltp*$(\alpha) < t_i$ and $\alpha^{\sigma_{\text{cmi}}} = t$. Therefore $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \alpha$ and $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \mathbf{F}x$.

   (c) If $x^{\sigma_{\text{cmi}}} = u$ then either there is no sentence with consequent $\mathbf{T}x$ or $\mathbf{F}x$ in Φ, or there is a sentence $\alpha \rightarrow \mathbf{T}x$ or $\alpha \rightarrow \mathbf{F}x$ in Φ but $\alpha^{\sigma_{\text{cmi}}} \neq t$ and hence $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \nvdash \alpha$. In either case $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \nvdash \mathbf{T}x$ and $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \nvdash \mathbf{F}x$ hence $\{\mathbf{U}x\} \cup \boldsymbol{A}_{\boldsymbol{U}_a}(\Phi)$ is consistent. Since $\mathbf{U}x \in U_i'$, $\mathbf{U}x \in \boldsymbol{A}_{\boldsymbol{U}_a}(\Phi)$ and hence $\boldsymbol{A}_{\boldsymbol{U}_a}(\Phi) \vdash \mathbf{U}x$. $\square$

Any base sentences not assigned the values t or f by augmentation over $\boldsymbol{U}_a$ are assigned the value u by augmentation over $\boldsymbol{U}$.

Hierarchical augmentation over $\boldsymbol{U}_a$ performs essentially the same function as Algorithm 5.4.3. In the case of $\boldsymbol{A}_{\boldsymbol{U}_a}$, however, adherence to the constraints imposed by the logical connectives is handled automatically by the deductive machinery rather than done "by hand" using labelling rules. While this is less efficient, it is more appealing from a theoretical point of view since it is based on traditional deductive methods and can be implemented using standard theorem provers. Furthermore, as we show later in this chapter, the HDL formalism is more flexible since it allows us to use different types of defaults without changing the formalism.

The idea behind Theorem 8.2.1 can be taken a step further. If we are only concerned with finding the base sentences which are known in the augmented theory then we only need to include the u-defaults corresponding to the weak assertions in the causal theory. This can be formalised as follows.

Corollary 8.2.2 *Let* Φ *be a causal theory with u-defaults* $\boldsymbol{U}$*. Let* $\boldsymbol{U}_p$ *be the hierarchical default set obtained from* $\boldsymbol{U}$ *by removing all defaults* $\mathbf{U}x$ *such that neither* $\mathbf{P}x$ *nor* $\mathbf{P}{-}x$ *appears in the antecedent of a sentence in* Φ*. Then for any base sentence* z*,* $\mathbf{T}z \in \boldsymbol{E}_{\boldsymbol{U}}(\Phi)$ *if and only if* $\mathbf{T}z \in \boldsymbol{E}_{\boldsymbol{U}_p}(\Phi)$*.*

Proof. Any strong assertion in $\boldsymbol{E}_{\boldsymbol{U}}$ must be the consequent of a causal sentence whose antecedent can be deduced from the theory and the u-defaults (see the proof of Theorem 8.1.2). Of the conjuncts in the antecedent, however, only the weak assertions can be deduced from the u-defaults. Therefore the only u-defaults which are needed for the deduction of strong assertions are those which entail the weak assertions in the antecedents of the causal sentences. □

Again any base sentences not assigned the values t or f by augmentation over $\boldsymbol{U}_p$ must be assigned the value u by augmentation over $\boldsymbol{U}$.

Theorem 8.2.1 and Corollary 8.2.2 have important practical consequences since they allow significant reductions in the number of defaults which must be applied in the augmentations. In the sequel we will assume that we are dealing with finite sets of u-defaults, and therefore the augmentations over the u-defaults are effectively computable.

8.3 Implicit Assumptions in Chronological Ignorance

Causal theories have a number of technically desirable properties. They are always consistent and have a unique cmi truth valuation (or in the case of Shoham's original formulation, there is a unique set of known sentences entailed by all cmi models). We now examine more closely the motivation for causal theories and the knowledge they are intended to represent.

8.3.1 Causal Rules and the Ostrich Principle

Shoham [Sho88a, Sec 5.1] reasons that causal sentences obey the *ostrich principle*, or *what-you-don't-know-won't-hurt-you*. This is illustrated by the following rule.

$$\begin{aligned}&\Box\text{TRUE}(\mathtt{t},\mathtt{t},\mathtt{loaded}) \wedge \Box\text{TRUE}(\mathtt{t},\mathtt{t},\mathtt{fire})\\ &\quad \wedge\Diamond\text{TRUE}(\mathtt{t},\mathtt{t},\mathtt{air})\\ &\quad \wedge\Diamond\text{TRUE}(\mathtt{t},\mathtt{t},\mathtt{firingpin})\end{aligned}$$

$$
\begin{aligned}
&\wedge\Diamond\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{no\_marshmallow\_bullets}) \\
&\wedge \ldots \wedge \Diamond\text{other mundane conditions} \\
&\rightarrow\Box\textsc{true}(\mathtt{t}+\mathtt{1},\mathtt{t}+\mathtt{1},\mathtt{noise}) \quad \text{for all } \mathtt{t}
\end{aligned} \tag{8.2}
$$

Shoham [Sho88a, p. 302] states that

> ... we need not say anything about there being air [that is $\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$] to be able to infer that there will be a noise after the firing... On the other hand, if there is *no* air we had better state that fact explicitly [that is $\Box\neg\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$] in the initial conditions, otherwise we will erroneously conclude that there will be a loud noise.

In minimising knowledge, however, we *are* saying something about air. In fact it could be argued that we are saying something which contradicts the assumption we really wish to make. If the causal theory itself entails nothing about air, then the minimal models entail the sentence

$$\neg\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air}) \wedge \neg\Box\neg\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air}).$$

Thus we are saying that it is not known that there is air and not known that there is no air. On the other hand, invoking rule (8.2) makes the implicit assumption that it *is* known that there is air, since it allows us to deduce that it is known that there is a noise. Note that the implicit assumption here is $\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$ and not $\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$. The latter assertion is not strong enough to support an inference such as $\Box\textsc{true}(\mathtt{t}+\mathtt{1},\mathtt{t}+\mathtt{1},\mathtt{noise})$.

The discrepancy between what is actually being stated and what is implied can be seen more clearly by noting that the implication

$$\Diamond\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})\rightarrow\Box\textsc{true}(\mathtt{t}+\mathtt{1},\mathtt{t}+\mathtt{1},\mathtt{noise}) \tag{8.3}$$

is equivalent to the conjunction of the following sentences:

$$\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air}) \rightarrow \Box\textsc{true}(\mathtt{t}+\mathtt{1},\mathtt{t}+\mathtt{1},\mathtt{noise}) \tag{8.4}$$

$$\neg\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air}) \wedge \neg\Box\neg\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air}) \rightarrow \Box\textsc{true}(\mathtt{t}+\mathtt{1},\mathtt{t}+\mathtt{1},\mathtt{noise}) \tag{8.5}$$

In the absence of the information that either $\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$ is true or $\Box\neg\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$ is true, chronological minimisation forces both $\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$ and $\Box\neg\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$ to be false and the second sentence is invoked. The second sentence, however, has neutral antecedents and clearly does not represent causality. It makes an implicit assumption that the causality represented by the first sentence is applicable. We are forcing $\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$ to be false and acting as though it is true.

8.3.2 Causal Rules and "Real" Physics

The second explanation provided by Shoham [Sho88a, Sec. 5.2] is that causal rules relate to rules of lawful change. Shoham [Sho88a, p. 303] states that

Rules of lawful change are believed to be universally true, and therefore one could expect them to hold in every possible world. Correspondingly, the causal rule under discussion might have been expected to be:

$$\begin{aligned}\Box(&\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{loaded}) \wedge \textsc{true}(\mathtt{t},\mathtt{t},\mathtt{fire}) \wedge \\ &\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air}) \wedge \\ &\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{firingpin}) \wedge \\ &\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{no\_marshmallow\_bullets}) \wedge \\ &\ldots \text{other mundane conditions} \\ &\rightarrow \textsc{true}(\mathtt{t}+\mathtt{1},\mathtt{t}+\mathtt{1},\mathtt{noise})) \qquad \text{for all } \mathtt{t}\end{aligned} \tag{8.6}$$

This rule does not suit the CI minimisation strategy since it requires conditions such as $\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$ to be stated explicitly. Shoham makes no objections to this rule, however, but rather justifies his own alternative by stating that '... the trick is to make sure that a world which violates true physics is not the "real world."' This is to be achieved by introducing the following "soundness conditions".

Definition 8.3.1 [Sho88a, Def 5.1] The *soundness conditions* of a causal theory Ψ are the set of sentences $\Diamond\textsc{true}(t_1,t_2,\mathtt{p})\rightarrow\textsc{true}(t_1,t_2,\mathtt{p})$ such that $\Diamond\textsc{true}(t_1,t_2,p)$ appears on the left-hand side of some sentence in Ψ.

It is not clear, however, what the utility of these sentences is since the consequents are never used. The "unique model theorem" [Sho88a, Thm 4.4] and corresponding algorithm [Sho88a, Thm 4.8] only identify *known* base sentences (those specified by the relation K_{TK}). The truth values of other base sentences are, in general, left undefined by chronological minimisation. In addition, as we mentioned above, the base sentences are not strong enough to invoke other causal rules. For example, the consequent of the soundness condition

$$\Diamond\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})\rightarrow\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})$$

is not sufficient to invoke the causality in sentence (8.4). Note that we cannot make the assumption explicit by using stronger soundness conditions of the form $\Diamond\alpha\rightarrow\Box\alpha$ since this is equivalent to $\Box\neg\alpha \vee \Box\alpha$ (thus reintroducing the law of excluded middle at the "epistemic" level) and chronological minimisation has no means of arbitrating between these alternatives.

One effect that the soundness conditions do have is to force an extra restriction on causal theories, namely that $\Diamond\alpha$ and $\Diamond\neg\alpha$ do not both appear on the left-hand side of causal sentences. This also reflects the intuitive notion that we cannot assume that an atomic sentence is known to be true and known to be false.

8.4 Alternative Approaches

The discrepancy between the unknown antecedents and their known consequents is necessary in CI because knowledge is always minimised. All unconstrained base formulas are forced to take the "unknown" value and therefore implications made by default must be based on unknown information.

Hierarchical default logic provides a more flexible way of making assumptions and avoids this bias towards knowledge minimisation. The explicit assumptions in HDL can be either weak or strong assertions, and can be used to minimise or maximise selective knowledge. To illustrate this we describe four alternative approaches to causal reasoning using chronological augmentation.

8.4.1 Method I — Weak Assumptions

The asserted logic form of implication (8.3) is given by the sentence

$$\mathbf{P}\,\mathtt{air}(t) \quad \rightarrow \quad \mathbf{T}\,\mathtt{noise}(t+1) \tag{8.7}$$

which is equivalent to the conjunction of the sentences:

$$\mathbf{T}\,\mathtt{air}(t) \quad \rightarrow \quad \mathbf{T}\,\mathtt{noise}(t+1) \tag{8.8a}$$

$$\mathbf{U}\,\mathtt{air}(t) \quad \rightarrow \quad \mathbf{T}\,\mathtt{noise}(t+1) \tag{8.8b}$$

In writing (8.7) it seems natural to expect that this sentence, rather than sentence (8.8b), will be invoked by default. This requires constraining the truth value of $\mathtt{air}(t)$ to t *or* u rather than forcing it to u, and therefore cannot be achieved in CI. While it may be possible to modify the preference criteria for CI to achieve this, we would also have to modify the associated algorithms for generating the known sentences.

In HDL, on the other hand, the intended assumption can be made simply by replacing the default $\mathbf{U}\,\mathtt{air}(t)$ by the default $\mathbf{P}\,\mathtt{air}(t)$. This default no longer forces a unique cmi truth valuation since it allows $\mathtt{air}(t)$ to be assigned the values t or u. However it still leads to a unique augmentation: since $\mathbf{P}\,\mathtt{air}(t)$ is consistent with and less constraining than $\mathbf{U}\,\mathtt{air}(t)$ it cannot introduce a contradiction.

Along with representing the intended information more accurately, this scheme responds to new information in the way that we would expect. If in CI we learn that an assertion $\mathbf{T}x$ is true after applying the assumption $\mathbf{U}x$, the extension must be recalculated to take account of the contradiction between $\mathbf{U}x$ and $\mathbf{T}x$. In the proposed system, however, $\mathbf{P}x$ and its consequences will still hold when $\mathbf{T}x$ is added to the theory. This is due to the consistency of $\{\mathbf{P}x, \mathbf{T}x\}$ and the monotonicity with respect to consistent information of augmentation over free defaults (see Corollary 7.4.8). While this change will not alter what is known in the image set, it may have computational advantages in avoiding backtracking (see Section 7.4.1). In terms of truth valuations, adding $\mathbf{T}x$ to a theory containing only $\mathbf{P}x$ excludes half of the applicable truth valuations (those in which x is assigned the value u), whereas adding $\mathbf{T}x$ to a theory containing $\mathbf{U}x$ forces a change to a new truth valuation.

For convenience we define the *p-defaults* of a causal theory as follows.

Definition 8.4.1 Let Φ be a causal theory and t_0 be a time point such that for all boundary conditions $\beta \in \Phi$, $t_0 < ltp(\beta)$. A set of *p-defaults* for Φ is a hierarchical free default set $\boldsymbol{P} = \langle \{P_0, P_1, P_2, \ldots\}, \preceq \rangle$ where

$$P_0 = \{\mathbf{U}x \mid x \text{ is an atomic base sentence and } ltp(x) \leq t_0\},$$

for all $i > 0$, $i \in \mathbb{N}$,

$$\begin{aligned} P_i &= \{\mathbf{P}x \mid x \text{ is a base literal, } ltp(x) = t_0 + i \text{ and } \mathbf{P}x \text{ appears in } \Phi\} \cup \\ &\quad \{\mathbf{U}x \mid x \text{ is an atomic base sentence, } ltp(x) = t_0 + i \text{ and} \\ &\quad\quad \text{neither } \mathbf{P}x \text{ nor } \mathbf{P}{-}x \text{ appears in } \Phi\}, \end{aligned}$$

and $P_j \preceq P_k$ if and only if $j \leq k$.

If $\boldsymbol{P}$ is a set of p-defaults for Φ then we denote by $\boldsymbol{P}_a$ the hierarchical default set obtained by removing from $\boldsymbol{P}$ all defaults $\mathbf{U}x$ such that x does not appear in Φ. Similarly, we denote by $\boldsymbol{P}_p$ the set obtained by removing from $\boldsymbol{P}$ all defaults $\mathbf{U}x$ such that neither $\mathbf{P}x$ nor $\mathbf{P}{-}x$ appears in Φ. Thus $\boldsymbol{P}_p$ contains only atomic sentences $\mathbf{P}y$ appearing in Φ.

Theorem 8.2.1 and Corollary 8.2.2 can be modified in a straightforward way to show that the same known information is entailed by $\boldsymbol{A_P}$, $\boldsymbol{A_{P_a}}$ and $\boldsymbol{A_{P_p}}$. Clearly this is also the known information entailed by $\boldsymbol{A_U}$, $\boldsymbol{A_{U_a}}$ and $\boldsymbol{A_{U_p}}$.

8.4.2 Method II — Strong Assumptions

We have shown that defaults can be used to minimise or constrain knowledge. We now examine their use in maximising selective knowledge. This is achieved simply by replacing defaults of the form $\mathbf{P}x$ by defaults of the form $\mathbf{T}x$. The set obtained by making this substitution, for all defaults corresponding to weak antecedents, is called the *t-defaults* of a causal theory.

Definition 8.4.2 *Let Φ be a causal theory with p-defaults $\boldsymbol{P}$. A set of t-defaults for Φ is a hierarchical free default set $\boldsymbol{T} = \langle\{T_0, T_1, T_2, \ldots\}, \preceq\rangle$ obtained from $\boldsymbol{P}$ by replacing each default of the form $\mathbf{P}x$ by the default $\mathbf{T}x$. The subsets $\boldsymbol{T}_a$ and $\boldsymbol{T}_p$ are obtained similarly.*

Note that in order to ensure that the augmentation over t-defaults is nonbranching, the third condition of Definition 5.3.2 (that $\mathbf{P}x$ and $\mathbf{P}{-}x$ do not both appear on the left-hand side of sentences in Φ) is necessary. Without this restriction, contradictory assumptions $\mathbf{T}x$ and $\mathbf{T}{-}x$ might both appear in a default set.

In the following subsections we consider what can be gained by using strong assumptions. Before doing this, however, we consider what (if anything) is lost.

The changes in the behaviour of the system due to using t-defaults can be viewed in terms of the applicability of the defaults and the effects of the defaults. We look first at the issue of applicability. Since $\mathbf{T}x \models \mathbf{P}x$, the assumption $\mathbf{P}x$ is applicable wherever $\mathbf{T}x$ is. The converse is in general not true since, for example, $\mathbf{P}x$ is consistent with $\mathbf{P}{-}x$ while $\mathbf{T}x$ is not. This problem will not occur in causal theories, however, because $\mathbf{P}{-}x$ cannot be entailed by a causal theory and cannot be added by a set of t-defaults. Thus for causal theories a t-default will be applicable if and only if the corresponding p-default is.

The effect on the image of using t-defaults is, on the other hand, different from the effect of u-defaults or p-defaults. If Φ is a causal theory, then $\boldsymbol{A_T}(\Phi)$ includes all the known information in $\boldsymbol{A_P}(\Phi)$ (and hence $\boldsymbol{A_U}(\Phi)$) since each default $\mathbf{T}x$ in $\boldsymbol{T}$ corresponds to a default $\mathbf{P}x$ in $\boldsymbol{P}$ and $\mathbf{T}x \models \mathbf{P}x$. However, $\boldsymbol{A_T}(\Phi)$ may contain additional known information—namely the applicable defaults from $\boldsymbol{T}$. The extension $\boldsymbol{E_T}(\Phi)$ will also contain their additional tautological consequences.

The need to avoid this extra known information might therefore provide a justification for choosing p-defaults over t-defaults. It may be possible to envisage situations where this information is detrimental. For example, in addition to Sentence (8.7) our theory might contain a sentence like

$$\mathbf{T}\,\mathtt{air}(t) \quad\rightarrow\quad \mathbf{T}\,\mathtt{can\_breathe}(t+1).$$

While it might be safe to assume that there is a noise, it may be far more dangerous to assume that we can breathe, and in this case the p-defaults must be used. We are not currently aware of any practical applications where this distinction is warranted, however, and in general it seems natural to make the assumptions we have applied explicit by using strong defaults.

8.4.3 Method III — Horn Causal Theories

Consider once again sentence (8.7) and its equivalent replacement by sentences (8.8a)–(8.8b). The strong assumption $\mathbf{T}\,\mathtt{air}(t)$ can be considered to invoke Sentence (8.8a). This sentence clearly represents the 'rule of lawful change' embodied in (8.7) and might be considered a more intuitive description of the causal relationship than (8.7). This begs the question as to why we need (8.7) rather than simply using the universal rule given by (8.8a).

Shoham's arguments for the form of causal theories, examined in Section 8.3, provide no reason for us to employ (8.7) over (8.8a). The choice of the former was necessary only for the CI minimisation technique which we have now escaped.

If we use t-defaults to maximise knowledge selectively, the part of implication (8.7) embodied by (8.8b) will never be invoked. We can therefore remove that part of the implication altogether by replacing the causal sentence (8.7) by the 'rule of lawful change' (8.8a).

Notice that making this replacement leaves only Horn sentences. We call a causal theory containing only Horn sentences a *Horn causal theory*, and define a transformation *Horn* from causal theories to Horn causal theories as follows.

Definition 8.4.3 *Horn* is a function on causal theories such that

$$\mathit{Horn}(\Phi) = \{\bigwedge_{i=1}^{m} \mathbf{T}x_i \;\wedge\; \bigwedge_{j=1}^{n} \mathbf{T}y_j \rightarrow \mathbf{T}z \quad\mid\quad \bigwedge_{i=1}^{m} \mathbf{T}x_i \;\wedge\; \bigwedge_{j=1}^{n} \mathbf{P}y_j \rightarrow \mathbf{T}z \in \Phi\}.$$

The fact that the "unknown" parts of the causal sentences play no part in the augmentation by t-defaults is formalised by the following lemma.

Lemma 8.4.4 *Let Φ be a causal theory and $\boldsymbol{T}$ be a set of t-defaults for Φ. Then*

$$\mathit{Horn}(\boldsymbol{A_T}(\Phi)) = \boldsymbol{A_T}(\mathit{Horn}(\Phi)).$$

Similarly for the sets $\boldsymbol{T}_a$ and $\boldsymbol{T}_p$.

Since the transformation *Horn* does not change the known information in a causal theory, it follows that the known information in $\boldsymbol{A_T}(\mathit{Horn}(\Phi))$ is the same as that in $\boldsymbol{A_T}(\Phi)$.

If we use the transformation *Horn*, then the weak antecedents in the causal theory act simply as syntactic markers for generating the subset $\boldsymbol{T}_p$ of t-defaults. We can therefore bypass this transformation and write our causal theories directly as Horn theories, providing we also write the set $\boldsymbol{T}_p$ directly. Alternatively we can "tag" the defaults in some other way so that they can be collected automatically. We will do this using the "prime" symbol; thus the sentence

$$\mathbf{T}x_1 \wedge \mathbf{T}x_2' \rightarrow \mathbf{T}z$$

indicates that $\mathbf{T}x_2$ is to appear as a default. This symbol has no logical meaning, that is it can be considered to be "invisible" from a logical point of view.

There are several potential advantages to using Horn default theories. For example, we are able to use the technique described in Chapter 6 for normalising defaults with the assurance that it will not introduce any new known information (see Theorem 6.4.4). If we wish to use qualified defaults of the form

$$\frac{: \mathbf{P}y \wedge \mathbf{T}z}{\mathbf{T}z}$$

for example, we can normalise these to give free defaults of the form

$$\frac{: \mathbf{P}y \wedge \mathbf{T}z}{\mathbf{P}y \wedge \mathbf{T}z}.$$

In using this scheme, however, it will be necessary to ensure that the weak assertions do not reintroduce branching into the formalism.

In addition it seems likely that we will be able to make use of efficient resolution procedures for Horn clauses, although we have yet to examine what adjustments must be made to such procedures to accommodate asserted logic.

8.4.4 Method IV — Classical Causal Theories

In Subsection 8.4.2 we argued that there is generally little advantage in using p-defaults over t-defaults. In the previous subsection we showed that if we choose to use t-defaults then the advantage of using causal theories with weak antecedents is purely syntactic; the weak antecedents indicate what information must be maximised by default. We could alternatively write Horn causal theories directly, and either specify the defaults directly or tag the appropriate antecedents.

Since $\boldsymbol{T}$ and $\boldsymbol{T}_a$ differ from $\boldsymbol{T}_p$ only in that they contain additional defaults of the form $\mathbf{U}x$, the assertions which are *known* due to augmenting a Horn theory with $\boldsymbol{T}$, $\boldsymbol{T}_a$ or $\boldsymbol{T}_p$, are exactly the same. If we are only concerned with known information, therefore, the set of defaults $\boldsymbol{T}_p$ is sufficient. The truth values of sentences constructed from base literals not appearing in the causal theory will simply be left undefined, in the same way that they are left undefined in a classical deductive system.

If we adopt this scheme then *no weak assertions* are used anywhere in the system and we never make use of the "unknown" truth value. The system can therefore be collapsed into an equivalent system in classical logic. To illustrate this point, consider any positive base literal x appearing in the object theory. Since x is never assigned the value u, the value of the sentence $\neg\mathbf{T}x$ is always the same as that of $\mathbf{T}{-x}$. Therefore, as we are dealing with a truth-functional language, we can replace each occurrence of $\mathbf{T}{-x}$ with

$\neg\mathbf{T}x$ without changing the truth value of any sentence. If this is done for all positive base literals in the theory (and defaults) then the strong negation symbol '$-$' will not appear. All atomic sentences will then be of the form $\mathbf{T}x$ and will take the same truth value as x. Clearly we can remove the assertion operator altogether and simply deal with a classical 2-valued system. Therefore, unless we wish to make use of weak antecedents for the reasons discussed in Subsection 8.4.2, or Horn theories for reasons such as those suggested in Subsection 8.4.3, there is no need to use an "epistemic" logic at all.

With this in mind we can redefine causal theories using the temporal logic BTK defined in Section 5.1. We adopt the same conventions that are used with TAL: we take temporal constant symbols from the integers, use only predicate symbols with arity $(m, 1)$ or $(m, 2)$, and always write the latest time point as the last argument in a base sentence. As usual we only use propositional sentences in causal theories.

Definition 8.4.5 A *causal sentence* is a BTK sentence of the form

$$\bigwedge_{i=1}^{m} \alpha_i \rightarrow \beta, \qquad m \in \mathbb{N} \tag{8.9}$$

where some of the antecedents may be tagged as defaults and

1. if $m = 0$ the corresponding (empty) conjunction is identically true, and
2. α_i, $i = 1, \ldots, m$ and β are literals such that $ltp(\alpha_i) < ltp(\beta)$.

A causal sentence is called a *boundary condition* if $m = 0$, and a *causal rule* otherwise.

Definition 8.4.6 A *causal theory* Φ is a set of causal sentences such that

1. there is a time point t_0 such that for all boundary conditions $\phi \in \Phi$, $t_0 < ltp(\phi)$,
2. there do not exist sentences $\alpha_1 \rightarrow \beta$ and $\alpha_2 \rightarrow \neg\beta$ in Φ such that $\{\alpha_1, \alpha_2\}$ is consistent, and
3. if defaults are tagged then there is no atomic sentence α such that α and $\neg\alpha$ are both tagged.

If the defaults are specified directly then we require that there is no atomic sentence α such that α and $\neg\alpha$ both appear in the hierarchical default set.

Examples of chronological augmentation over classical causal theories are given in the following chapter. Note that, as we discussed in Section 8.1, any theory which is tautologically equivalent to a causal theory can also be used as an object theory.

8.5 The Qualification Problem

One of the principle motivations for the development of chronological ignorance was to overcome the qualification and extended prediction problems associated with temporal reasoning. In this section we discuss the extent to which CI succeeds in the first aim, and look at what the nonmonotonicity actually provides with regard to causal theories.

The qualification problem is described by Shoham and McDermott [SM88, pp. 58–59] (see also [Sho88b]) as follows:

> Any rules of change (or physics) must support inferences of the form "if *this* is true at this time then *that* is true at that time." ...the first problem [qualification] is that the "if" part might get too large for practical use. For example, if we wish to be careful about our predictions, in order to infer that a ball rolling in a certain direction will continue doing so we must verify that there are no strong winds, that no-one is about to pick up the ball, that the ball does not consist of condensed explosives...and so on.
>
> The alternative is to be less conservative and base the predictions on only very partial information, hoping that those factors which have been ignored will not get in the way. This means that from time to time we must be prepared to make mistakes in our predictions...and be able to recover from them when we do.

The introduction of nonmonotonicity is an attempt to follow the second alternative—to base predictions on partial evidence and make the predictions defeasible so that they can be withdrawn in the light of new information. The problem is that, as we have shown in the previous sections, CI does not allow us to *ignore* any relevant factors. Not only must they be explicitly included in the preconditions of causal rules, they must also be assigned (explicitly or implicitly) a default value. To illustrate this, consider once again Shoham's causal rule [Sho88a, Sec 5.1]

$$\begin{aligned}&\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{loaded})\wedge\Box\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{fire})\\&\quad\wedge\Diamond\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{air})\\&\quad\wedge\Diamond\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{firingpin})\\&\quad\wedge\Diamond\textsc{true}(\mathtt{t},\mathtt{t},\mathtt{no\_marshmallow\_bullets})\\&\quad\wedge\ldots\wedge\Diamond\text{other mundane conditions}\\&\quad\rightarrow\Box\textsc{true}(\mathtt{t}+\mathtt{1},\mathtt{t}+\mathtt{1},\mathtt{noise})\qquad\text{for all }\mathtt{t}.\end{aligned}\tag{8.10}$$

The line

$$\wedge\ldots\wedge\Diamond\text{other mundane conditions}$$

which appears in one form or another in each practical example of a causal (or inertial) theory provided in [Sho86, Sho88a, Sho88b], distracts from the fact that each one of these conditions must be explicitly stated. If one is to consider mundane conditions (such as marshmallow bullets) the preconditions, or "if" part of the rule, will indeed be too large for practical use. If a practical number of preconditions are included then those which are omitted are ignored in the same way that they are ignored in classical logic—the predictions are not defeasible with respect to these conditions. The formalism therefore clearly fails to solve the qualification problem as stated above.

What the nonmonotonicity (chronological minimisation or defaults) does allow us to ignore when we construct an object theory, is what value these preconditions will *actually* take. That is, we don't need to ascertain in advance whether the preconditions are true or false and add facts to the theory accordingly. Instead the facts are added automatically by the nonmonotonic inference system according to the defaults we have provided. The inference system also adjusts these values automatically for each new or revised object theory.

While nonmonotonic reasoning does not solve the qualification problem, it can provide some help in dealing with the frame problem. Defaults can be used to reduce the amount of work done by the frame axioms since only information which is not in its default state must be propagated forward. This is demonstrated by examples in the following chapter.

Chapter 9

Examples of Declarative Modelling

In this chapter we provide two examples which illustrate the use of chronological augmentation for declarative modelling. The examples themselves are not intended to produce any particular results but rather to give a practical feel for some of the theoretical results which have been discussed.

9.1 Implementation

The structure of our declarative model, which was first shown in Figure 1.3, is illustrated once more in Figure 9.1. An experimental system following this structure is briefly described in this section. The system has been implemented in Prolog.

As the inference engine for the system we use chronological augmentation according to Method IV with the hierarchical default set $\boldsymbol{D}$ stated explicitly. The underlying language is therefore (propositional) BTK, and consistency checks are made with a classical propositional theorem prover. The theorem prover used is Fitting's propositional tableau [Fit88]. The object theory Φ consists of the union of the input Ω and the knowledge base Δ. Both Ω and Δ must be logically equivalent to causal theories described by Definition 8.4.6.

Implementation of the inference engine is straightforward. The recursive augmentation $\boldsymbol{A_D}$ is calculated according to the construction in Corollary 7.4.3. This involves reducing the hierarchical default set to singletons, by arbitrarily ordering defaults with the same *ltp*. Corollary 7.4.5, along with the fact that chronological augmentation is nonbranching, ensures that the order chosen does not affect the result. The defaults are then sequentially added to the object theory or discarded, depending upon the result of a consistency test.

In the examples which follow we specify the knowledge base and the defaults using schemata. Each schema can be regarded as a template for the actual sentences, which are generated automatically by replacing the variables (written in italics) with appropriate individuals from the domain. For example a schema

$$\neg\texttt{connected}(p_1, p_2)$$

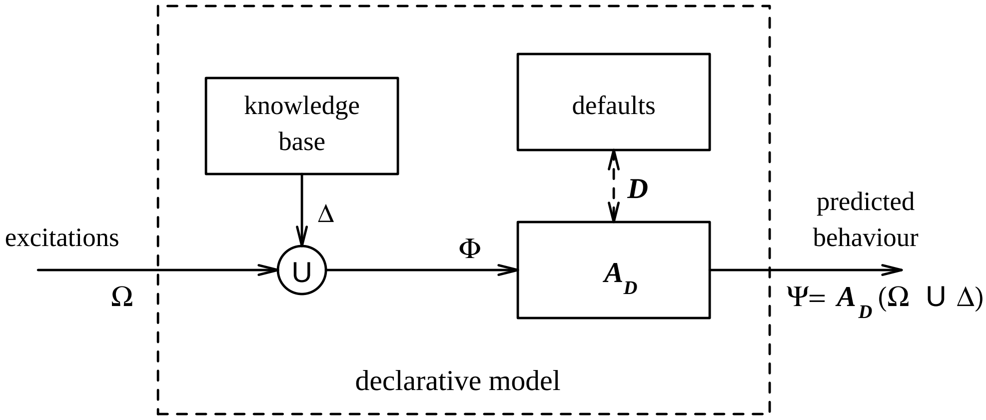

Figure 9.1: The structure of a logic-based declarative model.

where p_1 and p_2 represent assembly components, would be instantiated by searching a knowledge base containing sentences such as

$\texttt{component}(\texttt{c1}), \texttt{component}(\texttt{c2}), \texttt{component}(\texttt{c3}), \ldots$

This would generate sentences of the form

$\neg\texttt{connected}(\texttt{c1}, \texttt{c2}), \neg\texttt{connected}(\texttt{c1}, \texttt{c3}), \ldots$

9.2 Assembly of a Ball-Point Pen

A simple example commonly used to illustrate the generation of mechanical assembly sequences is the assembly of a ball-point pen [HdMS91]. The pen consists of a cap, head, body, tube, ink and button as shown in Figure 9.2. The problem is to assemble the parts according to the connection graph shown in Figure 9.3. It is assumed in [HdMS91] that the parts can be assembled in any order subject to the following physical constraints:

1. the head cannot be connected to the body once the cap and the body are assembled,
2. the head cannot be connected to the tube once the head, the body and the button are assembled, and
3. the ink cannot be inserted in the tube unless the head and tube are assembled and the head, body and button are not assembled.

The task of our model is to describe the outcome of any particular group of operations or *assembly plan*. The knowledge in the system is described below.

9.2.1 Excitations

The input in this example is the assembly plan. This consists of a set of sentences of the form

$\texttt{join}(p1, p2, t)$

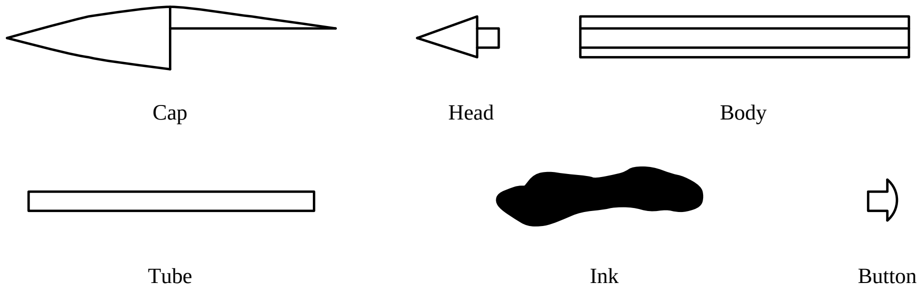

Figure 9.2: Parts of a ball-point pen.

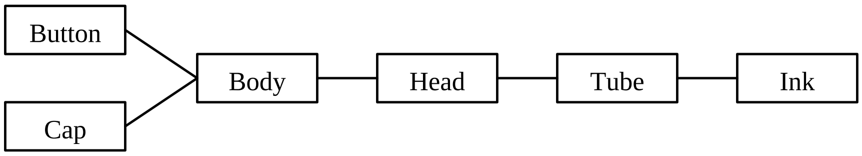

Figure 9.3: Connection graph for the ball-point pen.

instructing the workcell to assemble components p_1 and p_2 at time (or state) t. Examples of assembly plans are given later in this section.

9.2.2 Defaults

The state of the assembly at any particular time is described by a set of facts (or negated facts) of the form

$$\mathtt{conn}(p1, p2, t).$$

This sentence indicates that part p_1 is connected to part p_2 at time t.

By default we assume that no two parts are connected at any time unless there is information to the contrary. Our defaults therefore consist of all instantiations of the schema

$$\frac{: \neg\mathtt{conn}(p_1, p_2, t)}{\neg\mathtt{conn}(p_1, p_2, t)}$$

sorted into a hierarchical default set according to *ltp*. All other facts are assumed to be undefined.

The hierarchical default set, starting at time 0 and expressed as free defaults, is therefore a sequence of the form

$$\boldsymbol{D} = \{\{\neg\mathtt{conn}(\mathtt{cap}, \mathtt{head}, 0), \neg\mathtt{conn}(\mathtt{body}, \mathtt{head}, 0), \ldots\}, \{\neg\mathtt{conn}(\mathtt{cap}, \mathtt{head}, 1), \ldots\}, \ldots\}.$$

Table 9.1: Sentences describing the effects of assembly instructions.

Preconditions		Instructions		Postconditions
		$\texttt{join}(\texttt{body}, \texttt{button}, t)$	$\rightarrow$	$\texttt{conn}(\texttt{body}, \texttt{button}, t+1)$
		$\texttt{join}(\texttt{body}, \texttt{cap}, t)$	$\rightarrow$	$\texttt{conn}(\texttt{body}, \texttt{cap}, t+1)$
$\neg\texttt{conn}(\texttt{body}, \texttt{cap}, t)$	$\wedge$	$\texttt{join}(\texttt{body}, \texttt{head}, t)$	$\rightarrow$	$\texttt{conn}(\texttt{body}, \texttt{head}, t+1)$
$\neg[\texttt{conn}(\texttt{body}, \texttt{head}, t) \wedge \texttt{conn}(\texttt{body}, \texttt{button}, t)]$				
	$\wedge$	$\texttt{join}(\texttt{head}, \texttt{tube}, t)$	$\rightarrow$	$\texttt{conn}(\texttt{head}, \texttt{tube}, t+1)$
$\neg[\texttt{conn}(\texttt{body}, \texttt{head}, t) \wedge \texttt{conn}(\texttt{body}, \texttt{button}, t)]$				
$\wedge\ \texttt{conn}(\texttt{head}, \texttt{tube}, t)$	$\wedge$	$\texttt{join}(\texttt{ink}, \texttt{tube}, t)$	$\rightarrow$	$\texttt{conn}(\texttt{ink}, \texttt{tube}, t+1)$

9.2.3 The Knowledge Base

The knowledge base would normally consist of initial conditions and causal rules. No initial conditions are necessary in this case, however, since initially no parts are connected and this is handled automatically by the defaults.

The causal rules can be divided into two categories—those stating what does change from state to state (the effects of instructions) and the frame axioms stating what does not change from state to state.

Frame Axioms

Since there are no operations in this example for disassembling components, the frame axioms are straightforward. They simply say that any components which are connected in a particular state remain connected in the following state. This is expressed by instantiations of the schema

$$\texttt{conn}(p_1, p_2, t) \rightarrow \texttt{conn}(p_1, p_2, t+1).$$

The Effects of Instructions

The effect of a `join` instruction, providing the instruction does not contravene any of the physical constraints listed earlier, is that the parts are connected in the subsequent state. These effects are modelled by sentences of the general form

$$\bigwedge \mathit{preconditions} \wedge \mathit{instruction} \rightarrow \mathit{postconditions}.$$

The schemata for all valid instructions are given in Table 9.1. The last three sentences in the table have preconditions corresponding to the physical constraints. For example, the third sentence states that if an instruction is issued to assemble the head and body, and the cap and body are not currently assembled, then the head and body will be connected in the subsequent state. Note that the fourth and fifth sentences are not shown in causal form according to Definition 8.4.5. However each is logically equivalent to a conjunction of two causal sentences.

Table 9.2: Sentences characterising the modified assembly.

Preconditions		Instructions		Postconditions
		$\mathtt{join(body, button,} t)$	$\rightarrow$	$\mathtt{conn(body, button,} t+1)$
		$\mathtt{join(body, cap,} t)$	$\rightarrow$	$\mathtt{conn(body, cap,} t+1)$
$\neg\mathtt{conn(body, cap,} t)$	$\wedge$	$\mathtt{join(body, head,} t)$	$\rightarrow$	$\mathtt{conn(body, head,} t+1)$
$\neg\mathtt{conn(body, head,} t)$	$\wedge$	$\mathtt{join(head, tube,} t)$	$\rightarrow$	$\mathtt{conn(head, tube,} t+1)$
$\neg\mathtt{conn(body, head,} t)$ $\wedge\ \mathtt{conn(head, tube,} t)$	$\wedge$	$\mathtt{join(ink, tube,} t)$	$\rightarrow$	$\mathtt{conn(ink, tube,} t+1)$

9.2.4 Examples of Assembly Plans

We now have a complete model which can be used to test various assembly plans. We simply add the instruction set Ω to the knowledge base Δ and calculate the output $\boldsymbol{A_D}(\Omega \cup \Delta)$. We can then test whether any assertion α is entailed by the output description using a standard proof procedure.

As an example, consider the following two plans:

$$\begin{aligned}
\Omega_1 &= \{\mathtt{join(head, tube, 0), join(body, button, 0),} \\
&\quad \mathtt{join(ink, tube, 1), join(body, head, 2),} \\
&\quad \mathtt{join(body, cap, 3)}\} \\
\Omega_2 &= \{\mathtt{join(body, head, 0), join(body, cap, 1),} \\
&\quad \mathtt{join(head, tube, 2), join(ink, tube, 3),} \\
&\quad \mathtt{join(body, button, 4)}\}
\end{aligned}$$

If we wish to know whether the plans result in a complete assembly at time 5 we set

$$\begin{aligned}
\alpha &= \mathtt{conn(body, cap, 5)} \wedge \mathtt{conn(body, head, 5)} \wedge \mathtt{conn(head, tube, 5)} \\
&\quad \wedge \mathtt{conn(ink, tube, 5)} \wedge \mathtt{conn(body, button, 5)}
\end{aligned}$$

and obtain $\boldsymbol{A_D}(\Omega_1 \cup \Delta) \vdash \alpha$ and $\boldsymbol{A_D}(\Omega_2 \cup \Delta) \vdash \alpha$. That is, both plans are successful.

Note that the first plan Ω_1 contains an example of actions performed simultaneously and results in a complete assembly at time 4.

9.2.5 Changes to the Process

One of the advantages of being able to model dynamic processes is that it allows us to assess the effects of exceptions in the process, such as mechanical failures or loss of components. Similarly, if some intentional change is made to the process, we can assess whether the routines used previously are still appropriate or whether they must be replanned.

To illustrate this idea, assume the ball-point pen company adopts a new model in which it is no longer possible to assemble the head and tube or the tube and ink once the head is connected to the body. This requires a modification to the last two sentences shown in Table 9.1. The schemata for the new set of rules are shown in Table 9.2. We

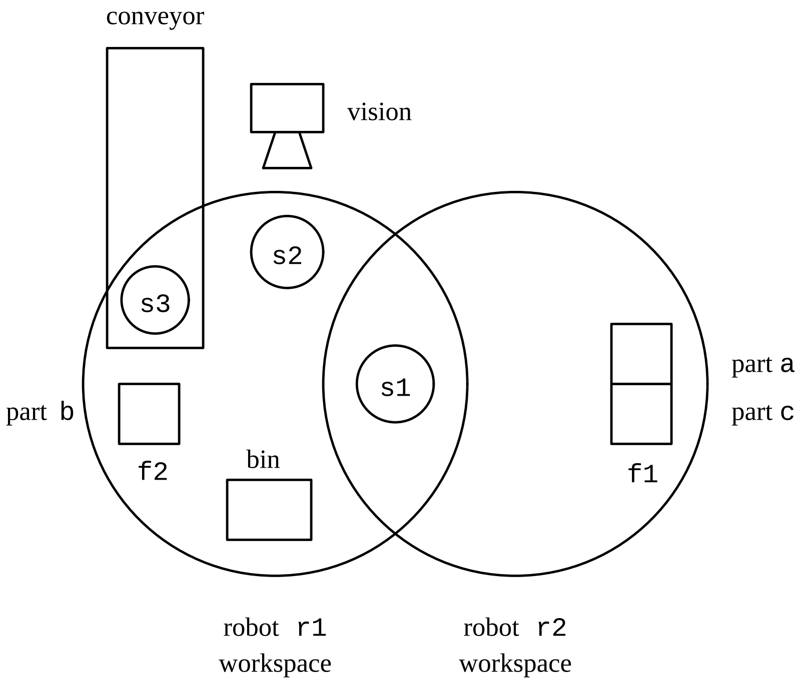

Figure 9.4: Configuration of the LAAS multirobot workcell [Fre91].

call the resulting knowledge base Δ'. If we now input the instruction sets Ω_1 and Ω_2 once more, we find that $\boldsymbol{A_D}(\Omega_1 \cup \Delta') \vdash \alpha$ and $\boldsymbol{A_D}(\Omega_2 \cup \Delta') \not\vdash \alpha$. That is, the first instruction set is still successful whereas the second set can no longer be used.

9.3 A Multirobot Workcell

As a more involved example of declarative modelling we consider a multirobot workcell described by Freedman [Fre91]. The example is originally adapted from work at LAAS, France, on a workcell testbed consisting of two six-degree-of-freedom robots, a fast monochrome vision system for component inspection, a revolving work table, a conveyor with infrared sensors for detecting components and associated jigs and feeders.

The configuration considered is shown in Figure 9.4. The robots `r1` and `r2` are to prepare the assembly `abc` (`c`-on-`b`-on-`a`) at site `s1`. `r2` is responsible for parts `a` and `c` which arrive together at the dual part feeder `f1`. `r1` is responsible for part `b` which is fed from `f2`. The complete assembly is then moved by `r1` to site `s2` for inspection. If the assembly passes the inspection then it is moved by `r1` to site `s3` on the conveyor, which removes it from the workcell, otherwise it is placed in the scrap bin.

Our conceptualisation of the system is described below. Once again our model takes the form of Figure 9.1 using Method IV and stating defaults explicitly.

Table 9.3: Actions performed by workcell elements.

Workcell element	Task	Description	Duration
robot r1	mate b to a at s1	$\texttt{mate}(\texttt{r1}, \texttt{b}, \texttt{a}, \texttt{s1}, t)$	2
	move abc from s1 to s2	$\texttt{move}(\texttt{r1}, \texttt{abc}, \texttt{s1}, \texttt{s2}, t)$	3
	move abc from s2 to s3	$\texttt{move}(\texttt{r1}, \texttt{abc}, \texttt{s2}, \texttt{s3}, t)$	3
	move abc from s2 to bin	$\texttt{move}(\texttt{r1}, \texttt{abc}, \texttt{s2}, \texttt{bin}, t)$	2
robot r2	place a at s1	$\texttt{place}(\texttt{r2}, \texttt{a}, \texttt{s1}, t)$	2
	mate c to ab at s1	$\texttt{mate}(\texttt{r2}, \texttt{c}, \texttt{ab}, \texttt{s1}, t)$	3
vision system	inspect assembly	$\texttt{inspect}(\texttt{vis}, \texttt{abc}, \texttt{s2}, t)$	4
conveyor	exit abc from workcell	$\texttt{exit}(\texttt{con}, \texttt{s3}, t)$	3
feeder for a and c	dispense parts	$\texttt{dispense}(\texttt{f1}, t)$	1
feeder for b	dispense part	$\texttt{dispense}(\texttt{f2}, t)$	1

9.3.1 Excitations

The excitations to the model are commands which invoke a particular action from some element in the workcell. Table 9.3 shows the possible actions and their durations for each workcell element. The commands may consist of a single action such as

$$\texttt{move}(\texttt{r1}, \texttt{abc}, \texttt{s1}, \texttt{s2}, 5)$$

which instructs robot r1 to move assembly abc from worksite s1 to s2 at time 5. However we also allow conditional instructions. For example, the excitation

$$\neg\texttt{pass}(\texttt{vis}, \texttt{abc}, 15) \rightarrow \texttt{move}(\texttt{r1}, \texttt{abc}, \texttt{s2}, \texttt{bin}, 16)$$

invokes the instruction to move the assembly to the bin if the assembly does not pass the visual inspection.

We assume that an input is also available from the vision system. The system returns an excitation $\texttt{pass}(\texttt{abc},t)$ or $\neg\texttt{pass}(\texttt{abc},t)$ four time units after the inspect command is issued, according to whether or not the assembly passes inspection.

9.3.2 Defaults

The assertions which are used to describe the state of the system at any particular time are shown in Table 9.4 along with their default values. The defaults are chosen to reflect the state of the system in the absence of any excitations, and assemblies are assumed to pass visual inspection.

We also require defaults for the actions listed in Table 9.3 since they are required by the frame axioms. The defaults reflect the fact that any particular action is usually not being performed at any particular time. For each assertion α listed in Table 9.3 the default is the negated assertion $\neg\alpha$.

Table 9.4: Conditions describing the workcell state and their default values.

Condition	Description	Default
part a ready at feeder	$\mathtt{ready}(\mathtt{a}, \mathtt{f1}, t)$	$\neg\mathtt{ready}(\mathtt{a}, \mathtt{f1}, t)$
part c ready at feeder	$\mathtt{ready}(\mathtt{c}, \mathtt{f1}, t)$	$\neg\mathtt{ready}(\mathtt{c}, \mathtt{f1}, t)$
worksite s1 free	$\mathtt{free}(\mathtt{s1}, t)$	$\mathtt{free}(\mathtt{s1}, t)$
part a at worksite s1	$\mathtt{at}(\mathtt{a}, \mathtt{s1}, t)$	$\neg\mathtt{at}(\mathtt{a}, \mathtt{s1}, t)$
part b ready at feeder	$\mathtt{ready}(\mathtt{b}, \mathtt{f2}, t)$	$\neg\mathtt{ready}(\mathtt{b}, \mathtt{f2}, t)$
assembly ab at worksite s1	$\mathtt{at}(\mathtt{ab}, \mathtt{s1}, t)$	$\neg\mathtt{at}(\mathtt{ab}, \mathtt{s1}, t)$
assembly abc at worksite s1	$\mathtt{at}(\mathtt{abc}, \mathtt{s1}, t)$	$\neg\mathtt{at}(\mathtt{abc}, \mathtt{s1}, t)$
worksite s2 free	$\mathtt{free}(\mathtt{s2}, t)$	$\mathtt{free}(\mathtt{s2}, t)$
assembly abc at worksite s2	$\mathtt{at}(\mathtt{abc}, \mathtt{s2}, t)$	$\neg\mathtt{at}(\mathtt{abc}, \mathtt{s2}, t)$
assembly abc pass inspection	$\mathtt{pass}(\mathtt{abc}, t)$	$\mathtt{pass}(\mathtt{abc}, t)$
worksite s3 free	$\mathtt{free}(\mathtt{s3}, t)$	$\mathtt{free}(\mathtt{s3}, t)$
assembly abc at worksite s3	$\mathtt{at}(\mathtt{abc}, \mathtt{s3}, t)$	$\neg\mathtt{at}(\mathtt{abc}, \mathtt{s3}, t)$

9.3.3 The Knowledge Base

Once again the knowledge base does not require any initial conditions since the initial state is determined by the defaults. The knowledge base therefore consists of frame axioms and sentences describing the effects of the various actions.

Frame Axioms

The frame axioms are more complicated than in the previous example because the truth values of conditions describing the process may alternate many times, particularly if the workcell is to carry out repetitive operations. Because all the conditions have default values, however, the frame axioms only have to preserve the value of facts which are *not* in their default state. Those in their default state are preserved automatically.

As an example, the fact $\mathtt{ready}(\mathtt{a}, \mathtt{f1}, t_1)$ is normally false but becomes true following the action $\mathtt{dispense}(\mathtt{f1}, t_2)$. It should then remain true until component a is moved by the action $\mathtt{place}(\mathtt{r2}, \mathtt{a}, \mathtt{s1}, t_3)$. This is achieved by the frame axiom

$$\mathtt{ready}(\mathtt{a}, \mathtt{f1}, t) \wedge \neg\mathtt{place}(\mathtt{r2}, \mathtt{a}, \mathtt{s1}, t) \rightarrow \mathtt{ready}(\mathtt{a}, \mathtt{f1}, t+1).$$

Similar frame axioms for all the conditions listed in Table 9.4 are provided in Table 9.5. Note that the only actions required in the frame axioms are those which terminate non-default conditions. All other actions can be ignored.

Effects of Actions

As in the previous example the sentences describing the effects of actions take the form

$$\bigwedge \textit{preconditions} \wedge \textit{action} \rightarrow \textit{postconditions}.$$

The schemata for the sentences characterising the operation of the workcell are shown in Table 9.6. Schemata are given for all of the actions listed in Table 9.3 except for

Table 9.5: Frame axioms for the workcell.

Conditions		Terminating Actions		Conditions
$\texttt{ready(a, f1},t)$	$\wedge$	$\neg\texttt{place(r2, a, s1},t)$	$\rightarrow$	$\texttt{ready(a, f1},t+1)$
$\texttt{ready(c, f1},t)$	$\wedge$	$\neg\texttt{mate(r2, c, ab, s1},t)$	$\rightarrow$	$\texttt{ready(c, f1},t+1)$
$\texttt{ready(b, f2},t)$	$\wedge$	$\neg\texttt{mate(r1, b, a, s1},t)$	$\rightarrow$	$\texttt{ready(b, f2},t+1)$
$\neg\texttt{free(s1},t)$	$\wedge$	$\neg\texttt{move(r1, abc, s1, s2},t)$	$\rightarrow$	$\neg\texttt{free(s1},t+1)$
$\texttt{at(a, s1},t)$	$\wedge$	$\neg\texttt{mate(r1, b, a, s1},t)$	$\rightarrow$	$\texttt{at(a, s1},t+1)$
$\texttt{at(ab, s1},t)$	$\wedge$	$\neg\texttt{mate(r2, c, ab, s1},t)$	$\rightarrow$	$\texttt{at(ab, s1},t+1)$
$\texttt{at(abc, s1},t)$	$\wedge$	$\neg\texttt{move(r1, abc, s1, s2},t)$	$\rightarrow$	$\texttt{at(abc, s1},t+1)$
$\neg\texttt{free(s2},t)$	$\wedge$	$\neg\texttt{move(r1, abc, s2, s3},t)$		
	$\wedge$	$\neg\texttt{move(r1, abc, s2, bin},t)$	$\rightarrow$	$\neg\texttt{free(s2},t+1)$
$\texttt{at(abc, s2},t)$	$\wedge$	$\neg\texttt{move(r1, abc, s2, s3},t)$		
	$\wedge$	$\neg\texttt{move(r1, abc, s2, bin},t)$	$\rightarrow$	$\texttt{at(abc, s2},t+1)$
$\neg\texttt{pass(abc},t)$	$\wedge$	$\neg\texttt{move(r1, abc, s2, s3},t)$		
	$\wedge$	$\neg\texttt{move(r1, abc, s2, bin},t)$	$\rightarrow$	$\neg\texttt{pass(abc},t+1)$
$\texttt{at(abc, s3},t)$	$\wedge$	$\neg\texttt{exit(con, s3},t)$	$\rightarrow$	$\texttt{at(abc, s3},t+1)$
$\neg\texttt{free(s3},t)$	$\wedge$	$\neg\texttt{exit(con, s3},t)$	$\rightarrow$	$\neg\texttt{free(s3},t+1)$

the actions $\texttt{inspect(vis, abc, s2},t)$, $\texttt{exit(con, s3},t)$ and $\texttt{move(r1, abc, s2, bin},t)$. These actions are omitted because they do not require postconditions. The first instruction invokes the vision system and the postcondition is effectively the result $\texttt{pass(abc},t+4)$ or $\neg\texttt{pass(abc},t+4)$ returned after the inspection. The effect of the second action is to remove the assembly abc from the workcell. The knowledge that the assembly has gone, and worksite s3 is free, is handled by the last two frame axioms in Table 9.5 and the last two defaults in Table 9.4. The exit assertion cancels the frame axioms and the defaults take over. Since we are not concerned with where the assembly goes there are no other postconditions. The same is true of the third action, $\texttt{move(r1, abc, s2, bin},t)$, which moves a faulty assembly to the bin.

Note that some of the sentences in Table 9.6 have multiple postconditions. Sentences of this form are permitted since they are logically equivalent to a conjunction of causal sentences with single consequents.

9.3.4 Example of an Assembly Sequence

As in the previous example we can generate the response of the workcell to a set of excitations Ω by adding them to the knowledge base Δ and calculating the output $\boldsymbol{A_D}(\Omega \cup \Delta)$.

An example of an assembly sequence is

$$
\begin{aligned}
\Omega_1 = \{ & \texttt{dispense(f1, 0), dispense(f2, 0), place(r2, a, s1, 1)}, \\
& \texttt{mate(r1, b, a, s1, 3), mate(r2, c, ab, s1, 5), move(r1, abc, s1, s2, 8)}, \\
& \texttt{inspect(vis, abc, s2, 11)}, \\
& \texttt{pass(vis, abc, 15)} \rightarrow \texttt{move(r1, abc, s2, s3, 16)},
\end{aligned}
$$

Table 9.6: Sentences describing the effects of actions in the workcell.

Preconditions		Actions		Postconditions
$\neg$`ready`(`a`, `f1`,t) $\wedge$ $\neg$`ready`(`c`, `f1`,t)	$\wedge$	`dispense`(`f1`,t)	$\rightarrow$	`ready`(`a`, `f1`,$t+1$)
				$\wedge$ `ready`(`c`, `f1`,$t+1$)
$\neg$`ready`(`b`, `f2`,t)	$\wedge$	`dispense`(`f2`,t)	$\rightarrow$	`ready`(`b`, `f2`,$t+1$)
`ready`(`a`, `f1`,t) $\wedge$ `free`(`s1`,t)	$\wedge$	`place`(`r2`, `a`, `s1`,t)	$\rightarrow$	`at`(`a`, `s1`,$t+2$)
				$\wedge$ $\neg$`free`(`s1`,$t+2$)
`ready`(`b`, `f2`,t) $\wedge$ `at`(`a`, `s1`,t)	$\wedge$	`mate`(`r1`, `b`, `a`, `s1`,t)	$\rightarrow$	`at`(`ab`, `s1`,$t+2$)
`ready`(`c`, `f1`,t) $\wedge$ `at`(`ab`, `s1`,t)	$\wedge$	`mate`(`r2`, `c`, `ab`, `s1`,t)	$\rightarrow$	`at`(`abc`, `s1`,$t+3$)
`at`(`abc`, `s1`,t) $\wedge$ `free`(`s2`,t)	$\wedge$	`move`(`r1`, `abc`, `s1`, `s2`,t)	$\rightarrow$	`at`(`abc`, `s2`,$t+3$)
				$\wedge$ $\neg$`free`(`s2`,$t+3$)
`at`(`abc`, `s2`,t) $\wedge$ `free`(`s3`,t)	$\wedge$	`move`(`r1`, `abc`, `s2`, `s3`,t)	$\rightarrow$	`at`(`abc`, `s3`,$t+3$)
				$\wedge$ $\neg$`free`(`s3`,$t+3$)

$$\neg\texttt{pass}(\texttt{vis}, \texttt{abc}, 15)\rightarrow\texttt{move}(\texttt{r1}, \texttt{abc}, \texttt{s2}, \texttt{bin}, 16),$$
$$\texttt{at}(\texttt{abc}, \texttt{s3}, 19)\rightarrow\texttt{exit}(\texttt{con}, \texttt{s3}, 20)\}.$$

The success of the sequence can be ascertained by testing whether

$$\boldsymbol{A_D}(\Omega_1 \cup \Delta) \vdash \texttt{move}(\texttt{r1}, \texttt{abc}, \texttt{s2}, \texttt{bin}, 16) \vee \texttt{exit}(\texttt{con}, \texttt{s3}, 20).$$

If the workcell is to be used for repetitive assembly then efficiency can be improved by performing operations in parallel. For example, the next `dispense` operation from `f2` can take place at time `4` as soon as the component from the first operation has been moved to `s1`. Similarly, `r2` can place a new component `a` at site `s1` at time `9` as soon as the previous assembly has been moved. This can be verified by adding appropriate instructions to the excitations. Thus we would have a new input

$$\Omega_2 = \Omega_1 \cup \{\texttt{dispense}(\texttt{f2}, 4), \texttt{dispense}(\texttt{f1}, 6), \texttt{place}(\texttt{r2}, \texttt{a}, \texttt{s1}, 9), \ldots\}.$$

Note that this model is incomplete, however, since it does not include conditions describing which robots are busy at what times. The model will therefore allow simultaneous tasks to be assigned to a single robot. In order to overcome this problem we could include additional conditions `busy`($\texttt{r1}, t$) and `busy`($\texttt{r2}, t$) which are toggled at the appropriate times.

Chapter 10

Conclusions and Further Work

This chapter summarises the contributions of the thesis. We then discuss some of the problems which still remain and suggest where the solutions might be sought. Finally we outline some broader topics for further research.

10.1 Summary of Results

The central result of this work is the development of a nonmonotonic inference system for modelling dynamic processes which:

- is guaranteed to be deterministic; that is, for every consistent input description it will produce a single consistent output description. The relationship between the input and output is precisely described by a declarative transfer function.
- can be implemented in a clear and straightforward way using a classical theorem prover.

The main disadvantage of this approach is that it inherits the computational complexity of classical (propositional) logic.

In the process of developing this system we have produced a number of intermediate results which represent significant contributions to the nonmonotonic reasoning literature. These include:

1. A suggested framework for comparing the characteristics of different nonmonotonic inference systems.
2. A simplified semantics for chronological ignorance in which Shoham's modal logic of "knowledge" is replaced by a new logic called asserted logic. Asserted logic is truth functional and can be mapped onto classical logic.
3. Proposed solutions to two of the pervading problems associated with default logic, namely incoherence and the multiple extension problem. The former makes use of asserted logic to normalise defaults. The latter uses the logic HDL which incorporates a hierarchical default structure. In order to calculate the extensions for HDL we redefine default logic in such a way that deductive closure is "factored" out. This

leads to a proof procedure for default logic which is effective for nonnormal as well as normal defaults, and extends to the hierarchical framework.

4. A proof theory for causal chronological ignorance. The proof theory is based on HDL, using asserted logic as the underlying language.

5. Improvements to the proof-theoretic CI formalism and, in particular, an alternative framework which does not require an epistemic logic. In the latter case the proof procedure for HDL makes use of a theorem prover for classical propositional logic.

10.2 Improvements to the System

There are many areas where further enhancements to the system might be made, particularly with regard to the knowledge which can be represented. In this section we highlight a number of these areas and suggest some ideas for further research.

10.2.1 Non-causal Theories

It is worth noting that while we have used causal theories to guarantee determinism, the use of causal theories is not built in to our system. Rather we have demonstrated one particular use of HDL. We have already shown that using a proof-theoretic formalism relaxes the restriction on object theories from causal theories to theories which are logically equivalent to causal theories. Experience with the system on practical examples indicates that other types of non-causal sentences would be useful in the object theories.

In the robotic workcell problem described in Section 9.3 we used conditional instructions such as

$$\neg\texttt{pass}(\texttt{vis},\texttt{abc},15)\rightarrow\texttt{move}(\texttt{r1},\texttt{abc},\texttt{s2},\texttt{bin},16). \tag{10.1}$$

In fact this is a poor representation of the conditional instruction. Unlike the causal rules in the knowledge base, the delay (of one time unit) in the expression does not correspond to any physical delay. It is put there simply to conform with the restrictions on causal rules. The preferred instruction would have been

$$\neg\texttt{pass}(\texttt{vis},\texttt{abc},15)\rightarrow\texttt{move}(\texttt{r1},\texttt{abc},\texttt{s2},\texttt{bin},15). \tag{10.2}$$

Unfortunately, however, causal theories do not allow sentences in which the cause and effect occur simultaneously.

Shoham [Sho88a, Sec 9.2] discusses simultaneous cause-effect relationships and suggests incorporating them, without losing the 'unique model property' by imposing a partial order on propositions. The same approach, extended to include defaults, could be used to prevent branching in our formalism. To allow sentence (10.2), for example, we would place $\neg\texttt{pass}(\texttt{vis},\texttt{abc},15)$ before $\texttt{move}(\texttt{r1},\texttt{abc},\texttt{s2},\texttt{bin},15)$ in the partial order. This would prevent any implication (or chain of implications) in which $\texttt{move}(\texttt{r1},\texttt{abc},\texttt{s2},\texttt{bin},15)$ appeared on the left and $\neg\texttt{pass}(\texttt{vis},\texttt{abc},15)$ appeared on the right. In addition the default for $\texttt{move}(\texttt{r1},\texttt{abc},\texttt{s2},\texttt{bin},15)$ would appear after the default for $\texttt{pass}(\texttt{vis},\texttt{abc},15)$ in the ordered default set. While this approach has not been formalised it does not appear to pose any prohibitive problems. Similar orderings on theories in the context of default logic and the incoherence problem are discussed by Etherington [Eth87].

Another type of information which we would like to be able to express in our knowledge base is nontemporal assertions. Recall that in the generation of sentences from schemata (Section 9.1) we made use of an external knowledge base with assertions like

$$\texttt{component(c1)}, \texttt{component(c2)}, \ldots$$

expressing the fact that `c1, c2` and so on are assembly components. We might also like to use sentences like

$$\texttt{tube(c1)} \rightarrow \texttt{component(c1)}.$$

An advantage of our proof-theoretic approach is that non-temporal sentences such as these can be added directly to causal theories. They will simply be treated in the same way that they would be treated in a classical deductive system. In order to make full use of the non-temporal information, however, we would like to be able to include nontemporal assertions in the antecedents of causal rules. For example, we might like to use a frame axiom such as

$$\texttt{component(c1)} \wedge \texttt{component(c2)} \wedge \texttt{conn(c1, c2,}t\texttt{)} \rightarrow \texttt{conn(c1, c2,}t+1\texttt{)}.$$

Again these sentences can be used providing we do not permit non-temporal assertions in defaults and that we adhere to the appropriate conditions on causal theories. If nontemporal assertions are required in the consequents of causal sentences or in the default set, however, then some new means of ensuring determinism must be sought.

The HDL formalism will support other variations on object theories and defaults. More work needs to be done on modelling practical systems to discover what types of knowledge representation are most appropriate.

10.2.2 A First-Order System

Until now we have restricted our attention to propositional causal sentences and defaults, which are generated from schemata. In the pen assembly problem of Section 9.2, for example, frame axioms are generated from the schema

$$\texttt{conn}(p_1, p_2, t) \rightarrow \texttt{conn}(p_1, p_2, t+1) \tag{10.3}$$

and defaults are generated from the schema

$$\frac{: \neg\texttt{conn}(p_1, p_2, t)}{\neg\texttt{conn}(p_1, p_2, t)}. \tag{10.4}$$

Clearly this approach can lead to large theories and default sets. If there are m applicable components and n time points then (without additional restrictions) schema (10.3) will generate $m \times m \times n$ propositional sentences. This information could be expressed far more concisely if we allowed quantification over non-temporal and temporal terms.

Extending our formalism to allow quantification over non-temporal terms in causal sentences is straightforward. The results (other than decidability) for the proof theory of

HDL in Chapters 6 and 7 were given for knowledge bases consisting of first-order sentences. If we do not allow function symbols into nontemporal terms (that is we add only quantifiers and nontemporal variables to our propositional language) then the undecidability of first-order logic does not cause problems. The resulting causal theories are what Etherington [Eth87] calls *finite theories*—that is, they have only a finite number of predicate symbols, constants and variables making the Herbrand Universe finite.

All that we need to do to extend our system, therefore, is replace the propositional theorem prover with a first-order theorem prover. The propositional sentences represented by schema (10.3) could then be replaced by a first-order schema

$$\forall\, \mathtt{P1}, \mathtt{P2}\ \ (\mathtt{conn}(\mathtt{P1}, \mathtt{P2}, t) \rightarrow \mathtt{conn}(\mathtt{P1}, \mathtt{P2}, t+1)). \tag{10.5}$$

Quantification of this sort was allowed in Green's question-answering system QA3 [Gre69a] described in Section 2.2.

For temporal terms in causal sentences the problem is more difficult because we require an arithmetic addition function. Technically (if we assume a finite set of time points) we could define a function `plus` by including all instantiations of the schema

$$\mathtt{plus}(t_1, t_2) = t_1 + t_2 \tag{10.6}$$

and replace schema (10.5) with the sentence

$$\forall\, \mathtt{P1}, \mathtt{P2}, \mathtt{T}\ \ (\mathtt{conn}(\mathtt{P1}, \mathtt{P2}, \mathtt{T}) \rightarrow \mathtt{conn}(\mathtt{P1}, \mathtt{P2}, \mathtt{plus}(\mathtt{T}, 1))).$$

This approach merely shifts the problem to schema (10.6), however, and may still lead to a large number of instantiations. It also requires a change to first-order BTK [BTK89], and introduces undecidability problems. What we would like to be able to do is evaluate arithmetic functions either within or alongside the proof procedure. This is part of a more general problem which is discussed below.

As far as the defaults are concerned there appears to be little to gain from switching to a first-order system. The reason is that we do not require first-order defaults such as

$$\frac{:\ \forall\, \mathtt{P1}, \mathtt{P2}, \mathtt{T}\ \ \neg\mathtt{conn}(\mathtt{P1}, \mathtt{P2}, \mathtt{T})}{:\ \forall\, \mathtt{P1}, \mathtt{P2}, \mathtt{T}\ \ \neg\mathtt{conn}(\mathtt{P1}, \mathtt{P2}, \mathtt{T})}$$

(which is only applicable if *all* components are not connected at *all* times) but rather open defaults such as

$$\frac{:\neg\mathtt{conn}(\mathtt{P1}, \mathtt{P2}, \mathtt{T})}{:\neg\mathtt{conn}(\mathtt{P1}, \mathtt{P2}, \mathtt{T})}. \tag{10.7}$$

However, an open default is simply treated as a schema for the set of closed defaults which are its substitution instances [Rei80, Sec 7].

10.2.3 Quantitative Information

The need to incorporate arithmetic terms in temporal statements is part of a more general problem of incorporating quantitative information within a logic-based reasoning framework. Many reasoning problems, particularly in process modelling and control, have a quantitative aspect to them. Examples range from component inventories in production scheduling to descriptions of the positions and motion of robotic agents. We would therefore like to be able to perform arithmetical (and possibly algebraic) operations and comparisons on both temporal and nontemporal terms.

One approach to this problem may be to investigate the possibility of incorporating in the temporal logic a restricted form of arithmetic (such as Presburger arithmetic) which can be described within a decidable first-order logic system (see for example [Coo72]). An alternative approach would be to represent and manipulate quantitative information alongside (rather than within) logical formulas using a system such as labelled deduction [Gab90]. The idea here is to free the deductive mechanism from performing mathematical operations which may be done more easily within another framework.

10.2.4 Representation of Time

The representation of time that we have adopted is appropriate for constructing causal theories and hierarchical default sets which ensure determinism. Also, unlike systems based on the situation calculus in which states are the result of action sequences (see Chapter 2), time is absolute. This has the advantage that actions can overlap or occur in parallel.

The representation is inadequate, however, for representing a variety of concepts. Examples include continuous time, homogeneous assertions and strong negation, which are required for the collision avoidance problem described in Section 2.6. These concepts fall within the scope of the *interval representation problem* discussed by Trudel [Tru91].

Trudel proposes a continuous-time framework in which the information which is true over an interval is completely determined by what is true at each point in the interval. Statements about intervals are made by "integrating" over their internal points. For example, the sentence

$$\mathtt{integral}(0, 10, \mathtt{moving}(\mathtt{conveyor}), 6.5)$$

indicates that a conveyor is operating for 6.5 time units between time 0 and time 10. These ideas are formalised in a temporal logic GCH, together with a sound (although not complete) axiomatisation. An interesting area for further research would be to examine whether a more expressive temporal representation such as that found in GCH could be employed in our system.

10.3 Adaptive Planning

The attention given to AI systems in control engineering is in part due to their promise in handling exceptions and unpredictable behaviour. One approach to dealing with exceptions is *adaptive planning*. Rather than completely reschedule a process when an exception is detected, the aim of adaptive planning is to make minor modifications to the current plan in order to keep the system running as close to the original schedule as possible. The ability to modify production plans efficiently in automated manufacturing systems, for example, can lead to significant reductions in manufacturing costs [NG91].

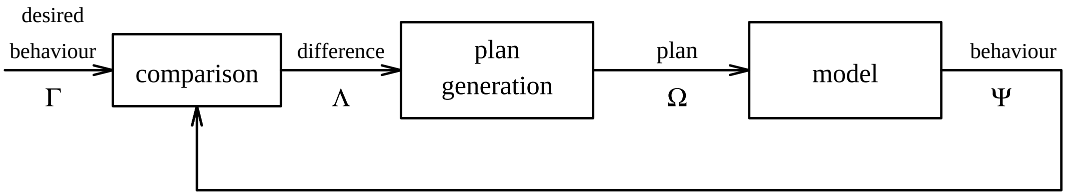

Figure 10.1: The structure of a declarative feedback control system.

A related use of adaptive planning is for plan generation and optimisation. In a complex system it is often not possible to generate production plans by search methods due to the large search space. A more promising approach may be to take a suboptimal plan which is known to work and use some knowledge of the system to improve it progressively.

In Chapter 1 we introduced, by analogy with quantitative control, a paradigm for closed-loop declarative control. The general structure of the system, updated to show the flow of declarative information, is illustrated in Figure 10.1. The declarative model that we have developed in this thesis generates a flow of information from input to output which suits this paradigm, and we have shown how the relationship between the input and output can be described accurately by a declarative transfer function.

Our longer term aim is to use the declarative model, in the feedback configuration shown, as a tool for adaptive planning. The operation of the envisioned system is similar for both exception handling and plan optimisation. Under normal conditions, with a successful plan, the output of the model and the desired behaviour coincide, resulting in a stable system. In the case of an exception, the model is revised to reflect the change in the system and the predicted behaviour is then regenerated. For plan optimisation, the desired behaviour is adjusted to demand more efficient operation. In either case the disturbance may result in a discrepancy between the desired and predicted behaviour. The controller uses this information, together with knowledge of the system or predetermined strategies, to make an incremental modification to the plan or schedule. The modified plan is fed back into the model and the predicted behaviour is regenerated and compared once more with the desired behaviour. This process continues until (ideally) the predicted and desired behaviours converge, resulting in a stable system.

As an example of plan modification consider the collision avoidance problem described in Section 2.6. The default plan, illustrated in the first scenario in Table 2.1, is for the agents to travel to their destinations by the most direct route. However, if the timing is less fortuitous, as in the second or third scenarios, the predicted behaviour shows a potential collision. This contradicts the desired behaviour and the controller must adjust the plans to avoid a collision. In this case the plans can be adjusted by inserting delays or using alternative routes. The strategy for modifying the plans may also attempt to optimise the travel times.

The problem of modifying plans based on the information in logical descriptions may require meta-level reasoning. One approach to the problem would be to attempt to implement, in a meta-level reasoning system, the *means-ends analysis* technique developed for the General Problem Solver (GPS) [EN69]. Means-ends analysis uses look-up tables to find operators which reduce the "differences" between the current and desired states. The technique is analysed in detail in [Ern69] and forms the basis of the state-space searching strategy used in STRIPS [FN71]. However, neither GPS nor STRIPS represent operators in a logical framework.

More generally the problem of finding assertions which produce a desired change in the output description appears to be one of *abduction*. An approach advocated by Gabbay [Gab91] for incorporating abduction in labelled deductive systems fits in well with our earlier proposal for using labelled deduction to represent temporal and other quantitative information. Further research in this direction appears to be promising.

10.4 Relationships with Non-AI Approaches

We have focused our attention in this thesis on techniques for knowledge representation and reasoning from the field of artificial intelligence. A valuable area for further research would be to investigate integrating our approach with other mathematical frameworks for modelling and analysing discrete-event dynamic systems.

Homem de Mello and Sanderson [HdMS91], for example, compare five alternative representations for assembly sequences: directed graphs, AND/OR graphs, the *set of establishment conditions* and two types of *sets of precedence relationships*. They discuss automatic generation of assembly sequences, mappings from one representation to another and correctness and completeness of the representations.

Freedman [Fre91], from which the robotic workcell problem in Chapter 9 was adapted, discusses the use of Petri nets for modelling and evaluating flexible manufacturing systems. Freedman looks in particular at temporal extensions to Petri net theory and their use in calculating cycle times for manufacturing processes.

It would be of interest to formalise the relationships between these systems and the logic-based approach. We expect, for example, that it would be possible to transform state descriptions and assembly plans from Homem de Mello and Sanderson's representations into logical descriptions, thus providing a declarative representation which is suitable for mechanised reasoning.

Appendix A

Mathematical Prerequisites

This appendix is intended to clarify the mathematical conventions adopted in the thesis. Texts providing a thorough introduction to discrete mathematics are widely available. Examples include [CB84, SM77]. A number of texts on mathematical logic are cited in Section 1.2.

A.1 Relations and Functions

The *cartesian product* of a finite sequence of sets A_1 to A_n, denoted by

$$A_1 \times A_2 \times \ldots \times A_n,$$

is the set of n-tuples $\langle a_1, a_2, \ldots, a_n \rangle$ such that $a_1 \in A_1, a_2 \in A_2, \ldots, a_n \in A_n$. If $A_i = A$ for $i = 1, \ldots, n$ then $A_1 \times \ldots \times A_n$ is denoted by A^n.

An *n-ary relation* R on A_1 to A_n is a subset of $A_1 \times A_2 \times \ldots \times A_n$. If $n = 2$ then R is called a *binary relation* from A_1 to A_2, and the *domain* of R and *range* of R are defined respectively by

$$\mathcal{D}(R) = \{a_1 \mid \langle a_1, a_2 \rangle \in R\}$$

and

$$\mathcal{R}(R) = \{a_2 \mid \langle a_1, a_2 \rangle \in R\}.$$

If $R \subseteq A^n$ then R is called an *n-ary relation on A*.

If R_1 is a relation from A to B and R_2 is a relation from B to C then the *composition* of R_1 and R_2, denoted $R_2 \circ R_1$, is the relation from A to C defined by

$$R_2 \circ R_1 = \{\langle a, c \rangle \mid \langle a, b \rangle \in R_1 \text{ and } \langle b, c \rangle \in R_2 \text{ for some } b \in B\}.$$

We assume associativity to the right, thus $R_3 \circ R_2 \circ R_1$ is equivalent to $R_3 \circ (R_2 \circ R_1)$.

A *function* (*mapping* or *transformation*) f from A to B, denoted $f : A \rightarrow B$, is a binary relation from A to B in which $\langle a, b_1 \rangle \in R$ and $\langle a, b_2 \rangle \in R$ implies $b_1 = b_2$. If f is a function and $\langle a, b \rangle \in f$ then we write $f(a) = b$ and call b the *image* of a under f.

A function $f : A \rightarrow B$ is called a *total function* if $\mathcal{D}(f) = A$.

An *n-ary operation* f on A is a function from A^n to A. For brevity we write $f(a_1, \ldots, a_n)$ for $f(\langle a_1, \ldots, a_n \rangle)$. A 0-ary operation on A is simply a member of A.

A.2 First-order Logic

This section defines the classical first-order language L which is used in this thesis. The development roughly follows [BM77], which should be consulted for a more detailed treatment.

A.2.1 Syntax of L

The symbols of L are as follows:

- An infinite sequence of (individual) variables.
- For each natural number n, a set of n-ary function symbols.
- For each positive natural number n, a set of n-ary predicate symbols, at least one of which is nonempty.
- The connectives $\neg$ (negation) and $\rightarrow$ (implication).
- The universal quantifier $\forall$.
- The punctuation marks , ( ) [ ] [1]

The 0-ary function symbols (if any) are called *individual constants*. The variables, connectives, universal quantifier and equality predicate (if present) are called *logical* symbols. The function symbols and predicate symbols (other than equality) are called *extralogical* symbols.[2]

The terms of L are as follows:

1. A single variable is a term.
2. If f is an n-ary function symbol and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.

Note that the 0-ary function symbols are terms.

The well-formed formulas[3] (wffs) of L are as follows:

1. If p is an n-ary predicate symbol and $t_1, \ldots, t_n$ are terms then $p(t_1, \ldots, t_n)$ is a wff.
2. If α is a wff then $\neg\alpha$ is a wff.
3. If α and β are wffs then $\alpha \rightarrow \beta$ is a wff.
4. If α is a wff and x is a variable then $\forall x\, \alpha$ is a wff.

---

[1] The punctuation marks are not strictly necessary [BM77, Chap. 1] however we include them for clarity.

[2] More accurately L defines a class of first-order languages since we allow the extralogical symbols to be arbitrarily chosen.

[3] By "formula" we mean "well-formed formula" unless otherwise indicated.

Formulas formed according to (1), (2), (3) and (4) are called *atomic*, *negation*, *implication* and *universal* formulas respectively.

The connectives $\wedge$ (conjunction), $\vee$ (disjunction), $\leftrightarrow$ (equivalence) and the existential quantifier $\exists$ are defined (as metalinguistic substitutions) in terms of $\neg$, $\rightarrow$ and $\forall$ as follows:

1. $\alpha \wedge \beta \ =_{\text{def}} \ \neg(\alpha \rightarrow \neg\beta)$
2. $\alpha \vee \beta \ =_{\text{def}} \ (\neg\alpha) \rightarrow \beta$
3. $\alpha \leftrightarrow \beta \ =_{\text{def}} \ (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
4. $\exists x \ \alpha \ =_{\text{def}} \ \neg\forall x \ \neg\alpha$

where α and β are wffs and x is a variable.

We will often omit parentheses with the convention that $\leftrightarrow$, $\rightarrow$, $\vee$, $\wedge$ and $\neg$ should be taken in that order of priority and connectives of equal priority should be read with association to the right. The scope of the quantifiers $\forall$ and $\exists$ are understood to be as short as possible, and we allow abbreviations such as $\forall\, \mathtt{XY}$ in place of $\forall\, \mathtt{X}\, \forall\, \mathtt{Y}$.

A *theory* (or *knowledge base* (KB)) is a set of well-formed formulas. For brevity we use the name of the language, L, to denote the set of well-formed formulas of L. Thus a theory is a set Φ such that $\Phi \subseteq \mathrm{L}$ or equivalently $\Phi \in \wp(\mathrm{L})$.

A variable which is not bound by a quantifier is called a *free* variable. A wff with no free variables is called a *closed* wff or a *sentence*. A wff with no variables is often called a *ground* sentence.

A.2.2 Propositional Semantics of L

Definition A.2.1 A *truth valuation* on L is a mapping σ assigning to each wff α a value α^σ from the set $\{t, f\}$, such that for all wffs β and γ

1. $(\neg\beta)^\sigma = t$ iff $\beta^\sigma = f$,
2. $(\beta \rightarrow \gamma)^\sigma = t$ iff $\beta^\sigma = f$ or $\gamma^\sigma = t$.

If α^σ is fixed arbitrarily for all universal and atomic formulas of L, then conditions (1) and (2) define β^σ for *all* wffs β. Thus, a mapping of the universal and atomic formulas into $\{t, f\}$ can be extended in a unique way by conditions (1) and (2) into a truth valuation.

A truth valuation σ on L *satisfies* a set Φ of wffs (written $\sigma \models \Phi$) iff $\phi^\sigma = t$ for every formula $\phi \in \Phi$. If Φ consists of just one formula ϕ we write $\sigma \models \phi$ for $\sigma \models \{\phi\}$.

We say ϕ is a *tautology* if $\sigma \models \phi$ for every truth valuation σ. It is always possible to check whether a formula ϕ is a tautology in a finite number of steps by constructing a truth table for ϕ in terms of its universal and atomic components.

A formula α is a *tautological consequence* of a set Φ of formulas (written $\Phi \models \alpha$) if $\sigma \models \alpha$ for every truth valuation σ such that $\sigma \models \Phi$. If Φ is empty we write $\models \alpha$. The empty set of formulas is satisfied by any truth valuation, hence $\models \alpha$ iff α is a tautology. Two formulas α and β are *tautologically equivalent* iff $\{\alpha\} \models \beta$ and $\{\beta\} \models \alpha$.

A.2.3 First-Order Semantics of L

Definition A.2.2 A first-order *interpretation* is a structure I consisting of a non-empty universe of discourse U, a mapping from each n-ary function symbol f to an n-ary operation f^I on U, and a mapping from each n-ary predicate symbol p to an n-ary relation p^I on U.

A *valuation* σ is an interpretation I together with an assignment of a value $x^\sigma \in U$ to each variable x. We call I the *underlying* structure of σ and define f^σ and p^σ to be the operation f^I and relation p^I respectively. Also $\sigma(x/u)$ is a valuation which agrees with σ everywhere except on x, and $x^{\sigma(x/u)} = u$.

A valuation induces mappings from the terms of L to the individuals of U and from the formulas of L to the set $\{t, f\}$. These mappings are called the Basic Semantic Definition (BSD).

Definition A.2.3 (Basic Semantic Definition)

(T1) If x is a variable, then x^σ is already defined.

(T2) If f is an n-ary function symbol and $t_1, \ldots, t_n$ are terms, then

$$[f(t_1, \ldots, t_n)]^\sigma = f^\sigma(t_1^\sigma, \ldots, t_n^\sigma).$$

(F1) If p is an n-ary extralogical predicate symbol and $t_1, \ldots, t_n$ are terms, then

$$[p(t_1, \ldots, t_n)]^\sigma = \begin{cases} t & \text{if } \langle t_1^\sigma, \ldots, t_n^\sigma \rangle \in p^\sigma, \\ f & \text{otherwise.} \end{cases}$$

(F1$^=$) If s and t are terms and L is a language with equality then

$$(s = t)^\sigma = \begin{cases} t & \text{if } s^\sigma = t^\sigma, \\ f & \text{otherwise.} \end{cases}$$

(F2) For every formula β,

$$(\neg\beta)^\sigma = \begin{cases} t & \text{if } \beta^\sigma = f, \\ f & \text{otherwise.} \end{cases}$$

(F3) For all formulas β and γ,

$$(\beta \rightarrow \gamma)^\sigma = \begin{cases} t & \text{if } \beta^\sigma = f \text{ or } \gamma^\sigma = t, \\ f & \text{otherwise.} \end{cases}$$

(F4) For every formula β and variable x,

$$(\forall x \beta)^\sigma = \begin{cases} t & \text{if } \beta^{\sigma(x/u)} = t \text{ for every } u \in U, \\ f & \text{otherwise,} \end{cases}$$

where U is the universe of σ.

Clauses (F2) and (F3) ensure that every valuation induces a truth valuation (see Definition A.2.1). We can therefore define satisfaction and logical consequence in terms of the induced truth valuation in the normal way. A valuation σ on L *satisfies* a set Φ of wffs ($\sigma \models \Phi$)[4] if $\phi^\sigma = t$ for every formula $\phi \in \Phi$. A wff is *valid* (or *logically true*) if

[4] We use the same symbol for propositional and first-order satisfaction. The meaning will be clear from the context.

it is satisfied by every valuation. A formula α is a *logical consequence* of a set Φ of wffs (written $\Phi \models \alpha$) if $\alpha^\sigma = t$ for every valuation σ such that $\sigma \models \Phi$.

Note that while every valuation induces a truth valuation, the converse does not hold. In other words, every tautology of L is valid, whereas not every valid formula of L is a tautology. For example, $\forall x\,(\alpha \rightarrow \alpha)$ is valid in L but not a tautology.

A.2.4 Subclasses of First-order Logic

For convenience we distinguish two subclasses of the first-order language L. The first consists of the sentences (or closed formulas) of L. We call this a *sentential language* and denote it $\mathrm{L_S}$. The second consists of the sentences of $\mathrm{L_S}$ which can be formed using only individual constants, extralogical predicate symbols and the logical connectives. Thus we do not allow variables, quantifiers, equality, or function symbols with arity greater than zero. We call this a *propositional language* and denote it $\mathrm{L_P}$.

In practice our knowledge bases do not contain free variables and therefore use languages based on $\mathrm{L_S}$ or $\mathrm{L_P}$. When the first-order consequence ($\models$) or deduction ($\vdash$) relations are used in the context of one of these languages it is assumed that both the theory (on the left-hand side) and the formula (on the right-hand side) must come from that language.

An advantage of dealing with $\mathrm{L_P}$ is that, unlike L or $\mathrm{L_S}$, every truth valuation is induced by some valuation. Any set of truth values can be assigned to the atomic formulas of $\mathrm{L_P}$ by choosing an interpretation which maps the predicate symbols to appropriate relations. This assignment can then be extended to a truth assignment to all formulas according to the rules governing logical connectives which are common to both valuations (rules (F2) and (F3) in Definition A.2.3) and truth valuations (Definition A.2.1). Thus the valid formulas of $\mathrm{L_P}$ are exactly the tautologies of $\mathrm{L_P}$, and the provable formulas of $\mathrm{L_P}$ are exactly the propositionally provable formulas of $\mathrm{L_P}$. We can therefore use propositional proof procedures to determine the validity of $\mathrm{L_P}$ formulas.

Appendix B

First-order Semantics for Asserted Logic

In this appendix we propose a first-order semantics for asserted logic. This would be needed, for example, in order to extend our arguments regarding causal CI theories to the first-order theories recently proposed by Bell [Bel91]. The discussion uses the syntax for AL defined Section 4.3.

B.1 Semantics

A first-order interpretation normally maps each predicate symbol to a single relation. This leads naturally to a 2-valued semantics since inclusion in the relation is a boolean function. Thus an atomic formula can be assigned a value t if and only if the n-tuple of individuals corresponding to its arguments is included in the relation.

Clearly this is not sufficient for a 3-valued semantics. We overcome this by instead mapping each predicate symbol to a pair of mutually exclusive relations.

Definition B.1.1 An *interpretation* in AL is a structure I consisting of a non-empty universe of discourse U, a mapping from each n-ary function symbol f to an n-ary operation f^I on U, and a mapping from each n-ary predicate symbol p to an ordered pair $p^I = \langle p^I_t, p^I_f \rangle$ of n-ary relations on U such that $p^I_t \cap p^I_f = \emptyset$.

A *valuation* σ is an interpretation I together with an assignment of a value $x^\sigma \in U$ to each variable x. We call I the underlying structure of σ and define f^σ and p^σ to be the operation f^I and pair of relations p^I respectively.

The Basic Semantic Definition for AL can now be defined as follows.

Definition B.1.2 (Basic Semantic Definition for AL)

(T1) If x is a variable, then x^σ is already defined.

(T2) If f is an n-ary function symbol and $t_1, \ldots, t_n$ are terms, then

$$[f(t_1, \ldots, t_n)]^\sigma \quad = \quad f^\sigma(t_1^\sigma, \ldots, t_n^\sigma).$$

(B1) If p is an n-ary extralogical predicate symbol and $t_1, \ldots, t_n$ are terms, then

$$[p(t_1, \ldots, t_n)]^\sigma \quad = \quad \begin{cases} t & \text{if } \langle t_1^\sigma, \ldots, t_n^\sigma \rangle \in p_t^\sigma, \\ f & \text{if } \langle t_1^\sigma, \ldots, t_n^\sigma \rangle \in p_f^\sigma, \\ u & \text{otherwise} \end{cases}$$

where $p^\sigma = \langle p_t^\sigma, p_f^\sigma \rangle$.

(B1$^=$) If s and t are terms and AL includes the equality predicate then

$$(s = t)^\sigma \quad = \quad \begin{cases} t & \text{if } s^\sigma = t^\sigma, \\ f & \text{otherwise.} \end{cases}$$

(B2) If x is a base formula then

$$(-x)^\sigma \quad = \quad \begin{cases} t & \text{if } x^\sigma = f, \\ f & \text{if } x^\sigma = t, \\ u & \text{otherwise.} \end{cases}$$

(F1) For every base formula x,

$$(\mathbf{T}x)^\sigma \quad = \quad \begin{cases} t & \text{if } x^\sigma = t, \\ f & \text{otherwise.} \end{cases}$$

(F2) For every formula β,

$$(\neg\beta)^\sigma \quad = \quad \begin{cases} t & \text{if } \beta^\sigma = f, \\ f & \text{otherwise.} \end{cases}$$

(F3) For all formulas β and γ,

$$(\beta \rightarrow \gamma)^\sigma \quad = \quad \begin{cases} t & \text{if } \beta^\sigma = f \text{ or } \gamma^\sigma = t, \\ f & \text{otherwise.} \end{cases}$$

(F4) For every formula β and variable x,

$$(\forall x \beta)^\sigma \quad = \quad \begin{cases} t & \text{if } \beta^{\sigma(x/u)} = t \text{ for every } u \in U, \\ f & \text{otherwise,} \end{cases}$$

where U is the universe of σ.

Clauses (B2) and (F1)–(F3) ensure that truth values are assigned to wffs according to the conditions for a truth valuation (see Definition 4.3.1). Satisfaction and logical consequence can therefore be defined in terms of the truth valuation in the normal way.

Appendix C

Details of the Logic LCT

This appendix provides the syntax and semantics of the logic LCT used in the multiple agent collision avoidance example described in Chapter 2. LCT is adapted from the logic TK which is described in [Sho88a, Sho88b]. We also provide an algorithm for generating the *known facts* in the cmi models of projection theories.

C.1 Syntax of LCT

Let P be a set of primitive propositions, V be a set of (temporal) variables, and $\Re$ the real numbers. We define an *arithmetic term* as follows:

1. If $a \in V \cup \Re$ then a is an arithmetic term.

2. If a_1 and a_2 are arithmetic terms then $a_1{+}a_2$ and $a_1{-}a_2$ are arithmetic terms.

The well-formed formulas (wffs) of LCT can now be defined as follows:

1. If a_1 and a_2 are arithmetic terms then $a_1 = a_2$ and $a_1 < a_2$ are wffs.

2. If a_1 and a_2 are arithmetic terms and $p \in P$ then $\text{TRUE}(a_1, a_2, p)$ and $\text{TRUE}(a_1, a_2, -p)$ are wffs.

3. If φ_1 and φ_2 are wffs then so are $\varphi_1 \rightarrow \varphi_2$, $\neg\varphi_1$ and $\Box\varphi$.

4. If φ is a wff and $v \in V$, then $\forall v \varphi$ is a wff.

The connectives $\wedge$, $\vee$ and $\leftrightarrow$ and the existential quantifier $\exists$ are defined in terms of $\neg$, $\rightarrow$ and $\forall$ in the usual way, and parentheses are used for clarity. $\Diamond$ is defined by $\Diamond\varphi \equiv \neg\Box\neg\varphi$. For convenience we abbreviate $\text{TRUE}(a, a, p)$ to $\text{TRUE}(a, p)$ and $\text{TRUE}(a, a, -p)$ to $\text{TRUE}(a, -p)$. We also abbreviate $\Box\text{TRUE}(a_1, a_2, p)$ to $\Box(a_1, a_2, p)$ with similar abbreviations for $\neg\Box\text{TRUE}(a, p)$, $\Diamond\text{TRUE}(a_1, a_2, p)$ and so on. We write $\text{TRUE}(a_1, a_2, [-]p)$ to mean either a formula $\text{TRUE}(a_1, a_2, p)$ or $\text{TRUE}(a_1, a_2, -p)$.

$$
\begin{array}{lcl}
KI, w & \models & a_1 = a_2[VA] \textit{ iff } MR(a_1) = MR(a_2). \\
KI, w & \models & a_1 < a_2[VA] \textit{ iff } MR(a_1) \prec MR(a_2). \\
KI, w & \models & \textsc{true}(a_1, a_2, p)[VA] \textit{ iff} \\
& & \quad a_1 = a_2[VA] \text{ and } \langle w, MR(a_1)\rangle \in M_2(p) \\
& & \quad \text{or } a_1 < a_2[VA] \text{ and for all } t \text{ st } MR(a_1) \preceq t \prec MR(a_2),\ \langle w, t\rangle \in M_2(p) \\
& & \quad \text{or } a_2 < a_1[VA] \text{ and for all } t \text{ st } MR(a_2) \preceq t \prec MR(a_1),\ \langle w, t\rangle \in M_2(p). \\
KI, w & \models & \textsc{true}(a_1, a_2, -p)[VA] \textit{ iff} \\
& & \quad a_1 = a_2[VA] \text{ and } \langle w, MR(a_1)\rangle \notin M_2(p) \\
& & \quad \text{or } a_1 < a_2[VA] \text{ and for all } t \text{ st } MR(a_1) \preceq t \prec MR(a_2),\ \langle w, t\rangle \notin M_2(p) \\
& & \quad \text{or } a_2 < a_1[VA] \text{ and for all } t \text{ st } MR(a_2) \preceq t \prec MR(a_1),\ \langle w, t\rangle \notin M_2(p). \\
KI, w & \models & \Box\varphi[VA] \textit{ iff } KI, w' \models \varphi[VA] \text{ for all } w' \in W. \\
KI, w & \models & (\varphi_1 \rightarrow \varphi_2)[VA] \textit{ iff } KI, w \models \varphi_2[VA] \text{ or } KI, w \not\models \varphi_1[VA]. \\
KI, w & \models & (\neg\varphi)[VA] \textit{ iff } KI, w \not\models \varphi[VA]. \\
KI, w & \models & (\forall v \varphi)[VA] \textit{ iff } KI, w \models \varphi[VA'] \text{ for all } VA' \text{ that agree} \\
& & \text{with } VA \text{ everywhere except possibly on } v.
\end{array}
$$

Figure C.1: Conditions under which LCT formulas are satisfied.

C.2 Semantics of LCT

Let T be an infinite linear series of time points which is dense and continuous (see [MF90c]). A *Kripke interpretation* is a pair $\langle W, M\rangle$ where W is a (nonempty) universe of possible worlds, and $M = \langle M_1, M_2\rangle$ is a meaning function $M_1 : \Re \rightarrow T$ and $M_2 : p \rightarrow 2^{W\times T}$.

A variable assignment is a function $VA : V \rightarrow \Re$. If a is an arithmetic expression, we define $R(a)[VA]$ to be the result of substituting the variables in a according to VA and evaluating the expression according to standard arithmetic rules. We also define $MR(a) \equiv M_1 \circ R(a)[VA]$.

Figure C.1 lists the conditions under which a Kripke interpretation KI and a world $w \in W$ satisfy a wff φ under a variable assignment VA (written $KI, w \models \varphi[VA]$).

Models are defined as for the logic TK. A Kripke interpretation KI and a world w are a *model* for a wff φ (written $KI, w \models \varphi$) if $KI, w \models \varphi[VA]$ for all variable assignments VA. If φ is a sentence and $KI, w \models \varphi[VA]$ for some VA, then $KI, w \models \varphi$. φ_2 is a logical consequence of φ_1 (written $\varphi_1 \models \varphi_2$) if and only if φ_2 is satisfied by all models of φ_1. A Kripke interpretation KI and a world w are a model for a theory Ψ if for each sentence $\varphi \in \Psi$, $KI, w \models \varphi$.

Note that, unlike TK, temporal assertions are homogeneous in the sense that $\textsc{true}(t_1, t_4, p) \models \textsc{true}(t_2, t_3, p)$ for any $t_1, t_2, t_3, t_4 \in \Re$ such that $t_1 \leq t_2 \leq t_3 \leq t_4$. Also our semantics allow both strong negation $\textsc{true}(t_1, t_2, -p)$ and weak negation $\neg\textsc{true}(t_1, t_2, p)$.

C.3 An Inference Algorithm for Projection Theories

The following algorithm generates the known facts in the cmi models of a projection theory. The algorithm takes as arguments a list F of known facts and S of projection rules and the times t_0 and t_∞ at which model generation will start and stop respectively. Note that for any initial condition $\Box(t', t'', [-]p)$ in F we require $t_0 \leq t'$. x refers to a proposition p or negated proposition $-p$, and $-(-p) \equiv p$. The algorithm has been implemented in Lisp.

Algorithm C.3.1

procedure generate(F, S, t_0, t_∞);
begin
$F_0 := F;\ j := 0;\ t_{-1} := t_0 - 1;$
while $t_j \neq t_\infty$ **do**
  begin
  if $t_j \neq t_{j-1}$ **then** $t_{j+1} := t_\infty$ **else** $t_{j+1} := t_j$;
  $t'_{j+1} := t_\infty;\ S' := S;\ L :=$ empty list of lists;
  while S' is not empty **do**
    begin
    Remove the first sentence $\Sigma \supset \Pi$ from S', make a list A of the conjuncts
    in the antecedent Σ and a list C of the conjuncts in the consequent Π;
    while A is not empty **do**
      begin
      Remove the first conjunct a from A;
      if a is of the form $\Box(t\text{–}d_1, t\text{–}d_2, x)$ **then**
        begin
        if F_j contains a fact $\Box(t', t'', x)$ such that $t' \leq t_j - d_1 < t_j - d_2 < t''$ **then**
          $t_{j+1} = \min(t_{j+1}, t_j + d_2, t'' + d_2);\ b := true$
        else if F_j contains a fact $\Box(t', t'', x)$ such that
              $t' \leq t_j - d_1 \leq t_j - d_2 = t''$ and $t_j \neq t_{j-1}$ **then**
          $t_{j+1} = t_j;\ b := true$
        else if F_j contains a fact $\Box(t', t'', x)$ such that
              $d_1 - d_2 \leq t'' - t'$ and $t_j - d_1 < t'$ **then**
          $t_{j+1} := \min(t_{j+1}, t_j + d_2, t' + d_1);\ b := false$
        else $t_{j+1} := \min(t_{j+1}, t_j + d_2);\ b := false$;
        end;
      if a is of the form $\Diamond(t\text{–}d_1, t\text{–}d_2, x)$ **then**
        begin
        if F_j contains a fact $\Box(t', t'', -x)$ such that $t' < t_j - d_2$ and
              $t_j - d_1 < t''$, or $t - d_1 = t - d_2 = t'$ **then**
          $t_{j+1} = \min(t_{j+1}, t_j + d_2, t'' + d_1);\ b := false$
        else if F_j contains a fact $\Box(t', t'', -x)$ such that
              $t_j - d_2 = t'$ and $t_j = t_{j-1}$ **then**
          $t_{j+1} = \min(t_{j+1}, t_j + d_2, t'' + d_1);\ b := false$
        else if F_j contains a fact $\Box(t', t'', -x)$ such that $t_j - d_2 \leq t'$ **then**

```
            t_{j+1} := min(t_{j+1}, t_j + d_2, t' + d_2); b := true
          else t_{j+1} := min(t_{j+1}, t_j + d_2); b := true;
          end;
        end;
      if b = true then add C to L;
      end;
    while L is not empty do
      begin
      Remove the first list of conjuncts C from L;
      while C is not empty do
        begin
        Remove the first conjunct □(t+e_1, t+e_2, x) from C;
        Append □(t_j+e_1, min(t_{j+1}+e_2, t_∞), x) to F_j;
        t'_{j+1} := min(t'_{j+1}, t_j+e_1);
        if F_j contains □(t_1, t_2, x) and □(t_3, t_4, x) such that t_1 ≤ t_3 ≤ t_2
          then remove these and replace with □(t_1, t_5, x) where t_5 = max(t_2, t_4);
        if F_j contains □(t_1, t_2, x) and □(t_3, t_4, −x) such that t_1 ≤ t_3 < t_2
          then report "error: no model for F,S" and stop;
        end;
      end;
    t_{j+1} := min(t_{j+1}, t'_{j+1}); F_{j+1} := F_j; j := j + 1;
    end;
Report F_j contains the known facts in the cmi models;
end;
```

Bibliography

[AGM85] Alchourron, C., Gärdenfors, P. and Mackinson, D. Partial Meet Contraction and Revision Functions. *J. Symbolic Logic*, **50** pp. 510–530, 1985.

[All84] Allen, J. F. Towards a General Theory of Action and Time. *Artificial Intelligence*, **23** pp. 123–154, 1984.

[Bel91] Bell, J. Extended Causal Theories. *Artificial Intelligence*, **48** pp. 211–224, 1991.

[BJ80] Boolos, G. S. and Jeffrey, R. C. *Computability and Logic*. Cambridge University Press, Cambridge, UK, second edition, 1980.

[BM77] Bell, J. and Machover, M. *A Course in Mathematical Logic*. North Holland, Amsterdam, The Netherlands, 1977.

[Bob84] Bobrow, D., editor. *Qualitative Reasoning about Physical Systems*. North Holland, Amsterdam, The Netherlands, 1984. Also *Artificial Intelligence: Special Volume on Qualitative Reasoning about Physical Systems*, **24**, 1984.

[BQQ83] Besnard, P., Quiniou, R. and Quinton, P. A Theorem Prover for a Decidable Subset of Default Logic. In *Proc. AAAI-83*, pp. 27–30, Washington, USA, 1983.

[Bre89] Brewka, G. Preferred Subtheories: An Extended Logical Framework for Default Reasoning. In *Proc. IJCAI-89*, pp. 1043–1048, Detroit, USA, 1989.

[Bro83] Brooks, R. Solving the Find–path Problem by Good Representation of Free Space. *IEEE Transactions on Systems, Man, and Cybernetics*, **SMC-13** pp. 190–197, 1983.

[Bro87] Brown, F. M., editor. *The Frame Problem in Artificial Intelligence: Proceedings of the 1987 Workshop*. Morgan Kaufmann, Los Altos, USA, 1987.

[BTK89] Bacchus, F., Tenenberg, J. and Koomen, J. A. A Non-Reified Temporal Logic. In *KR'89: Proc. 1st International Conference on Principles of Knowledge Representation and Reasoning*, pp. 2–10, Toronto, Canada, 1989. Morgan Kaufmann, San Mateo, USA.

[Bun83] Bundy, A. *The Computer Modelling of Mathematical Reasoning*. Academic Press, London, UK, 1983.

[Cam89] Cameron, S. Efficient Intersection Tests for Objects Defined Constructively. *International Journal of Robotics Research*, **8** pp. 3–25, 1989.

[CB84] Cooke, D. and Bez, H. *Computer Mathematics*. Cambridge University Press, Cambridge, UK, 1984.

[CL73] Chang, C. L. and Lee, R. C. T. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, New York, USA, 1973.

[CM84] Clocksin, W. F. and Mellish, C. S. *Programming in Prolog*. Springer-Verlag, Berlin, Germany, second edition, 1984.

[Coo72] Cooper, D. Theorem Proving in Arithmetic without Multiplication. In Meltzer, B. and Michie, D., editors, *Machine Intelligence 7*, pp. 91–99. Edinburgh University Press, Edinburgh, UK, 1972.

[Dor86] Dorf, R. C. *Modern Control Systems*. Addison-Wesley, London, UK, fourth edition, 1986.

[Doy79] Doyle, J. A Truth Maintenance System. *Artificial Intelligence*, **12** pp. 231–272, 1979.

[EN69] Ernst, G. W. and Newell, A. *GPS: A Case Study in Generality and Problem Solving*. Academic Press, London, UK, 1969.

[Ern69] Ernst, G. W. Sufficient Conditions for the Success of GPS. *J. ACM*, **16** pp. 517–533, 1969.

[Eth87] Etherington, D. W. Formalizing Nonmonotonic Reasoning Systems. *Artificial Intelligence*, **31** pp. 41–85, 1987. Errata: *Artificial Intelligence*, **34** p. 395, 1988.

[FG90] Froidevaus, C. and Grossetête, C. Graded Default Theories for Uncertainty. In *ECAI90: Proc. 9th European Conference on Artificial Intelligence*, pp. 283–288, Stockholm, Sweden, 1990. Pitman, London, UK.

[Fit88] Fitting, M. First Order Modal Tableaux. *J. Automated Reasoning*, **4** pp. 191–213, 1988.

[FN71] Fikes, R. and Nilsson, N. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Artificial Intelligence*, **2** pp. 189–208, 1971.

[Fre91] Freedman, P. Time, Petri Nets, and Robotics. *IEEE Transactions on Robotics and Automation*, **7** pp. 417–433, 1991.

[FS89] Fujimura, K. and Samet, H. A Hierarchical Strategy for Path Planning Among Moving Obstacles. *IEEE Transactions on Robotics and Automation*, **5** pp. 61–69, 1989.

[Gab90] Gabbay, D. Labelled Deductive Systems, Part I. Technical Report CIS-Bericht-90-22, Centrum für Informations- und Sprachverarbeitung, Universität München, Munich, Germany, 1990.

[Gab91] Gabbay, D. Abduction in Labelled Deductive Systems — A Conceptual Abstract. In Kruse, R. and Siegel, P., editors, *Symbolic and Quantitative Approaches to Uncertainty*, pp. 3–11. Springer-Verlag, Berlin, Germany, 1991. LNCS–548.

[Gal91a] Galton, A. A Critique of Yoav Shoham's Theory of Causal Reasoning. In *Proc. AAAI-91*, Anaheim, USA, 1991.

[Gal91b] Galton, A. Reified Temporal Theories and How To Unreify Them. In *Proc. IJCAI-91*, Sydney, Australia, 1991.

[GJ79] Garey, M. and Johnson, D. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, New York, USA, 1979.

[GN87] Genesereth, M. R. and Nilsson, N. J. *Logical Foundations of Artificial Intelligence.* Morgan Kaufmann, Los Altos, CA, 1987.

[Gre69a] Green, C. Application of Theorem Proving to Problem Solving. In *Proc. IJCAI-69*, pp. 219–239, Washington, USA, 1969. (Also in Webber, B. L. and Nilsson, N. J., editors, *Readings in Artificial Intelligence*, pp. 202–222. Tioga, Palo Alto, USA, 1981.).

[Gre69b] Green, C. Theorem-Proving by Resolution as a Basis for Question-Answering Systems. In Meltzer, B. and Michie, D., editors, *Machine Intelligence 4*, pp. 183–205. Edinburgh University Press, Edinburgh, UK, 1969.

[Haa78] Haack, S. *Philosophy of Logics.* Cambridge University Press, Cambridge, UK, 1978.

[HC68] Hughes, G. and Cresswell, M. *An Introduction to Modal Logic.* Methuen, London, UK, 1968.

[HdMS91] Homem de Mello, L. S. and Sanderson, A. C. Representations of Mechanical Assembly Sequences. *IEEE Transactions on Robotics and Automation*, **7** pp. 211–227, 1991.

[Hey66] Heyting, A. *Intuitionism.* Springer-Verlag, Berlin, Germany, 1966.

[Hin62] Hintikka, J. *Knowledge and Belief: An Introduction to the Logic of the Two Notions.* Cornell Universty Press, Ithaca, USA, 1962.

[HM86] Hanks, S. and McDermott, D. Default Reasoning, Nonmonotonic Logics, and the Frame Problem. In *Proc. AAAI-86*, pp. 328–333, Philadelphia, USA, 1986.

[Kon88a] Konolige, K. Hierarchic Autoepistemic Theories for Nonmonotonic Reasoning. In *Proc. AAAI-88*, pp. 439–443, Saint Paul, USA, 1988.

[Kon88b] Konolige, K. On the Relation Between Default Theories and Autoepistemic Logic. *Artificial Intelligence*, **35** pp. 343–382, 1988. Errata: *Artificial Intelligence*, **41** p. 115, 1989.

[Kow79] Kowalski, R. *Logic for Problem Solving.* Elsevier North Holland, New York, USA, 1979.

[Kri63] Kripke, S. Semantical Considerations on Modal Logic. *Acta Philosophica Fennica*, **16**, 1963.

[Lei87a] Leitch, R. R. editor. *IEE Proceedings-D: Control Theory and Applications. Special Issue on Artificial Intelligence*, **134** pp. 217–300, 1987.

[Lei87b] Leitch, R. R. Modelling of Complex Dynamic Systems. *IEE Proceedings-D: Control Theory and Applications*, **134** pp. 245–250, 1987.

[Lif85] Lifschitz, V. Computing Circumscription. In *Proc. IJCAI-85*, pp. 121–127, Los Angeles, USA, 1985.

[Lin87] Lin, F. Reasoning in the Presence of Inconsistency. In *Proc. AAAI-87*, pp. 139–143, Seattle, USA, 1987.

[LP83] Lozano-Perez, T. Spatial Planning: A Configuration Space Approach. *IEEE Transactions on Computers*, **C-32** pp. 108–120, 1983.

[Mac91a] MacNish, C. Hierarchical Default Logic. In Kruse, R. and Siegel, P., editors, *Symbolic and Quantitative Approaches to Uncertainty*, pp. 246–253. Springer-Verlag, Berlin, Germany, 1991. LNCS–548.

[Mac91b] MacNish, C. Well-behaved Reasoning with Seminormal Defaults. In *Proc. 4th International Symposium on AI*, pp. 201–207, Cancún, Mexico, 1991.

[McC63] McCarthy, J. Situations, Actions and Causal Laws. Memo 2, Artificial Intelligence Project, Stanford University, Stanford, USA, 1963. (Reprinted in Minsky, M., editor. *Semantic Information Processing*, pp. 410–418. MIT Press, Cambridge, USA, 1968).

[McC80] McCarthy, J. Circumscription—A Form of Non-Monotonic Reasoning. *Artificial Intelligence*, **13** pp. 27–39, 1980.

[McC86] McCarthy, J. Applications of Circumscription to Formalising Common-Sense Knowledge. *Artificial Intelligence*, **28** pp. 89–116, 1986.

[McD82] McDermott, D. Non-Monotonic Logic II: Non-Monotonic Modal Theories. *J. ACM*, **29** pp. 33–57, 1982.

[MD80] McDermott, D. and Doyle, J. Non-Monotonic Logic I. *Artificial Intelligence*, **13** pp. 41–72, 1980.

[MF90a] MacNish, C. and Fallside, F. Asserted 3-valued Logic for Default Reasoning. In *AI'90: Proc. 4th Australian Joint Conference on Artificial Intelligence*, pp. 65–80, Perth, Australia, 1990. World Scientific Publishing, Singapore.

[MF90b] MacNish, C. and Fallside, F. Asserted 3-valued Logic for Default Reasoning (Extended Version). Technical Report CUED/F-INFENG/TR.40, Cambridge University Engineering Department, Cambridge, UK, 1990.

[MF90c] MacNish, C. and Fallside, F. Temporal Reasoning: A Solution for Multiple Agent Collision Avoidance. In *Proc. 1990 IEEE International Conference on Robotics and Automation*, pp. 494–499, Cincinnati, USA, 1990. (See also Technical Report CUED/F-INFENG/TR.37, Cambridge University Engineering Department, Cambridge, UK, 1989.).

[MH69] McCarthy, J. and Hayes, P. Some Philosophical Problems from the Standpoint of Artificial Intelligence. In Meltzer, B. and Michie, D., editors, *Machine Intelligence 4*, pp. 463–502. Edinburgh University Press, Edinburgh, UK, 1969. (Also in Webber, B. L. and Nilsson, N. J., editors, *Readings in Artificial Intelligence*, pp. 431–450. Tioga, Palo Alto, USA, 1981).

[Min75] Minsky, M. A Framework for Representing Knowledge. In Winston, P. H., editor, *The Psychology of Computer Vision*, pp. 211–277. McGraw-Hill, New York, USA, 1975.

[Moo85a] Moore, R. C. A Formal Theory of Knowledge and Action. In Hobbs, J. and Moore, R., editors, *Formal Theories of the Commonsense World*, pp. 319–358. Ablex, Norwood, USA, 1985.

[Moo85b] Moore, R. C. Semantical Considerations on Nonmonotonic Logic. *Artificial Intelligence*, **25** pp. 75–94, 1985.

[NG91] Nof, S. Y. and Grant, F. H. Adaptive/Predictive Scheduling: Review and a General Framework. *International Journal of Production Planning and Control*, **2** pp. 298–312, 1991.

[Nil82] Nilsson, N. J. *Principles of Artificial Intelligence*. Springer-Verlag, Berlin, Germany, 1982.

[NS72] Newell, A. and Simon, H. A. *Human Problem Solving*. Prentice-Hall, Englewood Cliffs, USA, 1972.

[Poo85] Poole, D. L. On the Comparison of Theories: Preferring the Most Specific Explanation. In *Proc. IJCAI-85*, pp. 144–147, Los Angeles, USA, 1985.

[Qui76] Quine, W. V. Three Grades of Modal Involvement. In *The Ways of Paradox*, pp. 158–176. Harvard University Press, USA, 1976.

[Ray89] Rayner, M. Did Newton Solve the 'Extended Prediction Problem'? In *KR'89: Proc. 1st International Conference on Principles of Knowledge Representation* *and Reasoning*, pp. 381–385, Toronto, Canada, 1989. Morgan Kaufmann, San Mateo, USA.

[RB87] Roach, J. W. and Boaz, M. N. Coordinating the Motions of Robot Arms in a Common Workspace. *IEEE Journal of Robotics and Automation*, **RA-3** pp. 437–444, 1987.

[RC81] Reiter, R. and Criscuolo, G. On Interacting Defaults. In *Proc. IJCAI-81*, pp. 270–276, 1981.

[Rei78] Reiter, R. On Closed World Data Bases. In Gallaire, H. and Minker, J., editors, *Logic and Data Bases*, pp. 55–76. Plenum Press, New York, USA, 1978.

[Rei80] Reiter, R. A Logic for Default Reasoning. *Artificial Intelligence*, **13** pp. 81–132, 1980.

[Res69] Rescher, N. *Many-Valued Logic*. McGraw-Hill, New York, USA, 1969.

[Ric79] Richards, R. J. *An Introduction to Dynamics and Control*. Longman, London, UK, 1979.

[Rob65] Robinson, J. A. A Machine-Oriented Logic Based on the Resolution Principle. *J. ACM*, **12** pp. 23–41, 1965.

[RSB85] Rhodes, D., Stenson, E. H. and Blanchfield, P. Collision Avoidance Between Robots Operating in the Same Cell. In Billingsley, J., editor, *Robots and Automated Manufacture*, pp. 99–110. Peter Peregrinus, 1985.

[RW87] Rueb, K. D. and Wong, A. K. C. Structuring Free Space as a Hypergraph for Roving Robot Path Planning and Navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-9** pp. 263–273, 1987.

[San72] Sandewall, E. An Approach to the Frame Problem and its Implementation. In Meltzer, B. and Michie, D., editors, *Machine Intelligence 7*, pp. 195–204. Edinburgh University Press, Edinburgh, UK, 1972.

[She84] Shepherdson, J. C. Negation as Failure: A Comparison of Clark's Completed Data Base and Reiter's Closed World Assumption. *J. Logic Programming*, **1** pp. 51–79, 1984.

[Sho86] Shoham, Y. Chronological Ignorance: Time, Nonmonotonicity, Necessity and Causal Theories. In *Proc. AAAI-86*, pp. 389–393, 1986.

[Sho88a] Shoham, Y. Chronological Ignorance: Experiments in Nonmonotonic Temporal Reasoning. *Artificial Intelligence*, **36** pp. 279–331, 1988.

[Sho88b] Shoham, Y. *Reasoning About Change: Time and Causation from the Standpoint of Artificial Intelligence*. MIT Press, London, UK, 1988.

[Sim73] Simmons, R. F. Semantic Networks: Their Computation and Use for Understanding English Sentences. In Schank, R. and Colby, K., editors, *Computer Models of Thought and Language*, pp. 63–113. Freeman, San Francisco, USA, 1973.

[SM77] Stanat, D. F. and McAllister, D. F. *Discrete Mathematics in Computer Science*. Prentice-Hall, Englewood Cliffs, USA, 1977.

[SM88] Shoham, Y. and McDermott, D. Problems in Formal Temporal Reasoning. *Artificial Intelligence*, **36** pp. 49–61, 1988.

[THT87] Touretzky, D. S., Horty, J. F. and Thomason, R. H. A Clash of Intuitions: The Current State of Nonmonotonic Multiple Inheritance Systems. In *Proc. IJCAI-87*, pp. 476–482, Milan, Italy, 1987.

[Tru91] Trudel, A. The Interval Representation Problem. *International Journal of Intelligent Systems*, **6** pp. 509–547, 1991.

[Tur84] Turner, R. *Logics for Artificial Intelligence*. Ellis Horwood, Chichester, UK, 1984.

[Wal77] Waldinger, R. Achieving Several Goals Simultaneously. In Elcock, E. W. and Michie, D., editors, *Machine Intelligence 8*, pp. 94–136. Edinburgh University Press, Edinburgh, UK, 1977.

[Wal90] Wallen, L. A. *Automated Proof Search in Non-classical Logics: Efficient Matrix Proof Methods for Modal and Intuitionistic Logics*. MIT Press, Boston, USA, 1990.

[WF90] Williams, M. A. and Foo, N. Y. Default Theory Extensions as Justified Belief Sets. In *AI'90: Proc. 4th Australian Joint Conference on Artificial Intelligence*, pp. 94–102, Perth, Australia, 1990. World Scientific Publishing, Singapore.

[Zad75] Zadeh, L. A. Fuzzy Logic and Approximate Reasoning. *Synthese*, **30** pp. 407–428, 1975.