A note on first-order reasoning for minimum models

David Rydeheard and Jesús Héctor Domínguez Sánchez

School of Computer Science, University of Manchester
Oxford Road, Manchester, M13 9PL, UK.
david.rydeheard@manchester.ac.uk

Abstract

In this note, we consider a notion of minimum model suitable for formulating a semantics of evolvable computing systems using a revision-based logic. We explore a range of proof systems for reasoning in the logic of minimum models and consider their application to the simulation of evolvable systems. Finally, we outline how these proof systems may be implemented in a logical framework.

1 Introduction

An approach to the formal modelling of *evolvable computational systems* has been developed by Howard Barringer and others [4]. By 'evolvable' we mean systems that may change their internal structure during execution in response to an external stimulus or to internal monitoring. The attempt to provide formal models followed discussions with Brian Warboys and his group who were working on 'evolvability' in computer systems for companies so that systems may adapt to changing business structure and business environments [16]. An architecture for such systems based on 'supervised evolution' was proposed [25, 3]. After exploring several logical approaches to formalising evolvability, Dov Gabbay suggested the use of so-called 'revision-based logic' [14, 22, 35, 15] to model such systems. This note explores the mathematical foundation of this work, first presenting a suitable notion of model and then reporting on the work of the second author developing proof systems and their implementations for such models.

In its simplest form, revision-based logic describes the state of a system in terms of a set of observed properties. Computational actions are expressed as revisions to this set – properties are removed and new ones are added as computation proceeds. This approach to the specification of evolvable systems has several advantages:

- Revision-based logics themselves may be considered to be evolvable systems: Revisions of the set of formulae describing the state are considered to be normal computational steps. Evolutionary steps are those that change the logic through changes in the signature and axioms,
- Revision-based logics have be used to model computational systems which have an evolutionary capability [4],
- Revision-based descriptions have a built-in persistency. There is a well-recognised problem arising when defining a computation as a series of small steps: We wish to specify these steps in terms of only what changes, whereas the usual notion of model requires us also to specify what remains unchanged as well. This additional specification can often be large and unwieldy, and difficult to manage as computation progresses. In revision-based logic however computational steps are, in their simplest form, additions and deletions from a set and persistency is the remainder of the set remaining unchanged,

- Descriptions in revision-based logic are not only specifications of systems but also can be executed. Revision steps can be implemented to provide a logical simulation of the system i.e. an animation of a specification (see [1, 2] for a description of the ESAT system which is based on this idea).

In employing revision-based logic as a specification mechanism, we need to address the question of what notion of model is appropriate and how this choice affects the reasoning systems we may employ. The key idea is to consider specifications as defining models which are *minimum* in an order. This order is determined by what we consider to be 'observable' about a system. This has several consequences:

- Minimum models do not always exist. Whether a specification specifies anything at all depends on the existence of minimum models. The properties required of specifications for minimum models to exist are investigated in Section 3. Our experience (see e.g. [4, 5, 6, 7]) is that, through suitable choices of observations, theories and specifications, we may describe a range of computational systems, including evolvable systems, using minimum model semantics,
- Minimum models are unique only relative to the observable properties and need not be isomorphic or logically equivalent.

There has been considerable development of minimum, and minimal, models, including a variety of notions of minimality and a range of applications, especially in Artificial Intelligence, focussing on attempts to capture the 'closed-world assumption' in reasoning using techniques such as 'circumscription' (amongst an extensive literature, see for example [21, 13, 26, 32, 19, 20, 15, 29, 27, 28, 18, 12, 33]). Other work develops reasoning methods and their complexities (for example [9, 10, 34]).

We explain how this paper fits into these developments. The work here arose from a practical source: the attempt to implement a semantics of computational systems based on revision logic [4]. Such an implementation takes a logical specification of a system and simulates system execution by implementing revision steps. To do so, we need to determine what are the relevant models and how we may implement reasoning to compute revision steps. We introduce a notion of 'relative minimality' for models, extending minimality relative to a set of predicates [19, 20], to minimality relative to formulae drawn from an arbitrary set of closed formulae. We consider these formulae as 'possible observations' of models. This approach corresponds to the application we have in mind: We describe computational systems as models of a logic. The systems are 'monitored' by supervisory systems. A supervisory system detects 'monitoring events' based on observations which are determined by the 'instrumentation' of the monitored system. Observations of models in this sense form the starting point for this account. Indeed, a more general approach would admit two logics, one for determining the space of models, the other a logic for describing monitoring events, which may be, for example, a temporal logic. We do not consider this additional generality here.

We thus define an order on models in terms of what observations hold. Once an order on models is determined, the mathematical account of the resultant logic of minimum models follows a standard pattern (see e.g. [12]) and is straightforward. We include this account here both for completeness of presentation and to provide the mathematical background for developing reasoning systems. Minimum, rather than minimal, models provide the appropriate semantics for the computational systems under consideration. However, because this minimum is relative and determined by the set of possible observations, a range of non-isomorphic (and non-logically equivalent) models may serve as semantic models. In cases where minimum models do not exist, some of the reasoning systems we develop later become inconsistent.

Turning now to reasoning: In the propositional case, logics of minimum models are decidable, so the interest is in what decision procedures are available and their complexity. See [9, 10] for a survey of results in this area. In the first-order case, however, it has long been recognised that logics of minimality in general are not only not decidable, they are not even semi-decidable. A reduction of the problem of refutation in first-order minimum model logic to the Turing halting problem is described in [30]. Thus the questions arise: what practical approaches are there to reasoning in first-order minimum model logics, to what extent do these provide an implementation of revision-based logics, and how applicable are these to the semantics of computational systems so as to provide a simulation of evolvable systems? This is the material of this paper, where, by analysing the interaction between logical theories and sets of closed formulae serving as observations, we propose several approaches to developing reasoning systems and consider their applicability.

Towards the end of the paper, we turn to another aspect of implementing logics of minimum models, namely how may these logics be expressed in a logical framework, such as Isabelle [24] or Edinburgh LF [17]. Logical frameworks consist of typed λ-calculi together with mechanisms for representing logics. As well as providing proof construction tools, some also have in-built automated (semi-)decision procedures or have the provision for adding new (semi-)decision procedures. There are several aspects of the logics of minimum models that deviate from standard λ-calculi accounts of proof systems – we show how to address these in a logical framework in Section 5.

2 Introducing minimum models

For those not familiar with observational descriptions and minimum models in the form which we consider, an example may be useful.

2.1 Example: Blocks World

We consider the widely used example of a 'Blocks World', as introduced in [35]. A Blocks World consists of a finite collection of blocks (which we think of as of similar sizes) which may be placed upon each other to form towers or may reside upon a table. There is thus a predicate $on(x, y)$ stating that block x is directly on block y. Other predicates may be present, for example, the blocks may be coloured. To simplify the account, we do not consider an explicit table (this would require a typed world of blocks and tables which behave differently, but we do not need this). We consider blocks which are not on anything to be on the table (this is equivalent, in the usual account of a Blocks World, to there being a single table of unlimited capacity).

Let us illustrate a Blocks World state with four blocks A, B, C and D. Consider Figure 1, in which there are two towers of two blocks each. We may describe this state as the following set of 'observations', recording which blocks are on others:

$$\Delta = \{on(A, B), on(C, D)\}.$$

Thus, for this example, we choose observations to be ground atomic formulae constructed by applying the predicate *on* to names of blocks. Of course, there are other models of the set Δ, for example that depicted in Figure 2. However, the state in Figure 1 is, in a sense to be defined later, a minimum model of Δ. The observation $on(B, C)$, which holds in Figure 2, fails to hold in the minimum model, Figure 1. Indeed, we have, in the minimum model,

$$\Delta \mid\!\approx \neg on(B, C)$$

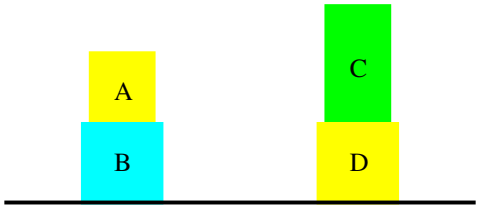

Figure 1: A Blocks World state.

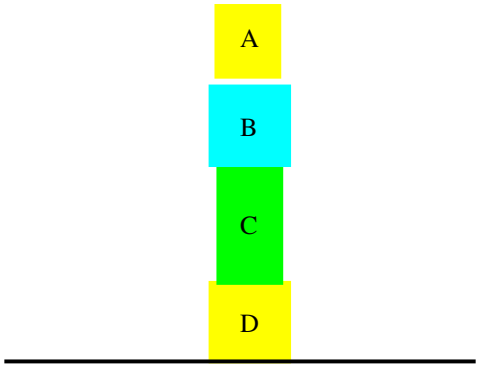

Figure 2: Another Blocks World state satisfying Δ.

where $\approx\!\!\models$ is the satisfaction relation for minimum models, which we define later. This holds in the minimum model because

$$\Delta \vdash_W on(B, C)$$

where $\vdash$ is standard first-order entailment under a theory W, and W is a suitable Blocks World theory. A simple Blocks World theory as a collection of axioms, which suffices for these examples, is given below. For an axiomatisation of a fuller account of Blocks World, see [11].

PREDICATES
$on : Block \times Block$
$above : Block \times Block$

AXIOMS
$$\begin{aligned}
&\forall x, x_1, x_2, x_3 : Block. \\
&\quad \neg on(x, x) \quad \wedge \\
&\quad on(x, x_1) \wedge on(x, x_2) \Rightarrow (x_1 = x_2) \quad \wedge \\
&\quad on(x_1, x) \wedge on(x_2, x) \Rightarrow (x_1 = x_2) \quad \wedge \\
&\quad on(x_1, x_2) \Rightarrow above(x_1, x_2) \quad \wedge \\
&\quad above(x_1, x_2) \wedge above(x_2, x_3) \Rightarrow above(x_1, x_3) \quad \wedge \\
&\quad above(x_1, x_2) \Rightarrow \neg above(x_2, x_1)
\end{aligned}$$

Several aspects of minimum model descriptions are illustrated in this example. Notice how the existence and form of minimum model depends critically on the choice of formulae that may serve as observations: the absence of observations from the set Δ may have implications

for the minimum model. Notice also how proof in minimum models depends on *both* first-order derivability and first-order non-derivability. This is a major obstacle in the way of developing proof systems for this notion of minimum models.

Consider now what happens when states change. We describe this by updating the set of observation formulae which describe the state, removing some observations and adding new ones. For example, suppose that, with the state in Figure 1, block A is moved to the table, to result in the state depicted in Figure 3. This transformation can be described by removing the

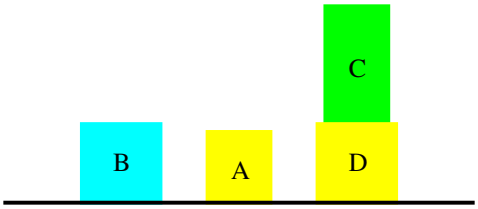

Figure 3: Another blocksworld state.

formula $on(A, B)$ from Δ to result in the set:

$$\Delta' = \{on(C, D)\}.$$

Then the state in Figure 3 is indeed a minimum model of Δ', with blocks A, B, C and D.

If, instead of using minimum models determined by observations, we try to describe the above models and transformation using sets of general formulae to capture properties that are intended to hold in a model and also those that are intended not to hold, the above description becomes more complex and revisions to sets of formulae need to take account both properties which hold and those that don't and their interaction through an axiomatic theory. For this reason and others, it has become standard in certain applications to consider interpretations via minimum models.

There is a simplicity to the above example which will not be the case in general. The formulae that may be observed are not only of a simple form but also, in this example, are independent of each other in the following sense: In the axiomatic theory of Blocks World, new observations of this form are not derivable from others. This is not an ingredient in formulating and developing a notion of minimum model, where any closed formulae may serve as observations, but is one of several useful properties for developing the restricted proof systems for logics of minimum models which we present in Section 4.

3 Minimum models

3.1 Models and satisfaction

We consider a typed, first-order (classical) logic $\mathcal{L}$ with formulae $\mathsf{Form}(\mathcal{L})$ built from a signature of function and predicate symbols. Let $\mathsf{Sen}(\mathcal{L})$ be the set of closed formulae (sentences) of $\mathcal{L}$.

Set-theoretic models are standard:

Definition 3.1. *Let $\mathcal{L}$ be a first-order typed logic. A* model *α of $\mathcal{L}$ is an allocation of a set $\alpha(T)$ to each type T of $\mathcal{L}$, a function $\alpha(f) : \alpha(T_1) \times \cdots \times \alpha(T_n) \to \alpha(T)$ to each function symbol $f : T_1 \times \cdots \times T_n \to T$ of $\mathcal{L}$, and a relation $\alpha(p) \subseteq \alpha(T_1) \times \cdots \times \alpha(T_n)$ to each predicate symbol $p : T_1 \times \cdots \times T_n$ of $\mathcal{L}$.*

The definition of the interpretation of a formulae $\psi \in \mathsf{Form}(\mathcal{L})$ in a model is standard. A closed first-order formula ψ is satisfied in a model α, written

$$\alpha \models \psi,$$

iff the interpretation of ψ in α is true. We extend this to sets of closed formulae Ψ: $\alpha \models \Psi$ iff for all $\psi \in \Psi$, $\alpha \models \psi$.

For a set of closed formulae Ψ and closed formula ψ, we write

$$\Psi \models \psi$$

iff for all $\mathcal{L}$-models α, $\alpha \models \Psi \implies \alpha \models \psi$.

A first-order *theory* is a set of axioms i.e. a set $W \subseteq \mathsf{Sen}(\mathcal{L})$. We say α is a model of W (or α is a W-model) when, for all $\psi \in W$, $\alpha \models \psi$. We write $\Psi \models_W \psi$ iff for all W-models α, if $\alpha \models \Psi$ then $\alpha \models \psi$.

Where the typed logic admits 'enumeration types', defining types in terms of finite set of elements, the axioms include, for each enumeration type, sentences which say the set of elements is exhaustive and the elements are distinct.

3.2 Observations and minimum models

Let $\mathcal{O}$ be a set of closed $\mathcal{L}$-formulae which we interpret as a collection of 'possible observations' about a model ($\mathcal{O}$ may be empty). In this section, the choice of observation formulae is arbitrary. In later sections, we consider how the set $\mathcal{O}$ and the theory W interact and consider restrictions on $\mathcal{O}$.

We define a model determined by a subset Δ of $\mathcal{O}$, as follows.

Definition 3.2. *For theory W of $\mathcal{L}$ and $\Delta \subseteq \mathcal{O}$, a W-model α satisfies Δ iff $\alpha \models \Delta$. Let $\mathsf{Mod}_W(\Delta)$ be the set of all W-models that satisfy Δ.*

Define a pre-order $\lesssim$ on $\mathsf{Mod}_W(\Delta)$ by

$$\alpha \lesssim \beta \text{ iff } \forall \varphi \in \mathcal{O}.\ \alpha \models \varphi \implies \beta \models \varphi.$$

We now consider minimum models *in $\mathsf{Mod}_W(\Delta)$ under this pre-order, i.e. models $\alpha \in \mathsf{Mod}_W(\Delta)$ such that for all models $\beta \in \mathsf{Mod}_W(\Delta)$, $\alpha \lesssim \beta$, that is:*

$$\forall \varphi \in \mathcal{O}.\ \alpha \models \varphi \implies \forall \beta \in \mathsf{Mod}_W(\Delta).\ \beta \models \varphi.$$

Minimum models are "observationally equivalent", i.e. for two minimum models α and α',

$$\forall \varphi \in \mathcal{O}.\ \alpha \models \varphi \text{ iff } \alpha' \models \varphi.$$

However, minimum models need not be isomorphic or logically equivalent.

Minimum models need not exist – it depends on the theory W, the choice of observations $\mathcal{O}$ and the set Δ.

Example 3.1. *Consider the Blocks World theory W with blocks A, B, C, D and observations $\mathcal{O} = \{on(x, y) \mid x, y \in Block\}$.*

For every minimum model α of $\Delta = \{on(A, B), on(C, D)\}$ (see Figure 1), we have

$$\alpha \models \neg on(B, C).$$

Proof. *If not, then $\alpha \models on(B, C)$. Now, because α is minimum, we have for all $\beta \in \mathsf{Mod}_W(\Delta)$, $\beta \models on(B, C)$ as $on(B, C) \in \mathcal{O}$. But because the interpretation of B is distinct from that of C, there is a model in $\mathsf{Mod}_W(\Delta)$ which does not satisfy $on(B, C)$.*

We introduce another satisfaction relation, $\models\!\!\approx$, defined using minimum models:

Definition 3.3. *For W an $\mathcal{L}$-theory, $\Delta \subseteq \mathcal{O}$, for a set of closed $\mathcal{L}$-formulae $\mathcal{O}$, and ψ a closed $\mathcal{L}$-formula, write*

$$\Delta \models\!\!\approx_W \psi$$

iff for all α minimum *in $\mathsf{Mod}_W(\Delta)$, $\alpha \models \psi$.*

Note that $\Delta \models_W \psi \implies \Delta \models\!\!\approx_W \psi$.

Example 3.2. *In the Blocks World theory W with $\Delta = \{on(A,B), on(C,D)\}$, we have $\Delta \models\!\!\approx_W \neg on(B,C)$.*

3.3 Characterising minimum models

We now consider the existence of minimum models, beginning with a characterisation theorem.

Theorem 3.1 (Characterising minimum models)**.** *Consider a theory W of $\mathcal{L}$ and $\Delta \subseteq \mathcal{O}$, for a set of closed $\mathcal{L}$-formulae $\mathcal{O}$. Define $\mathcal{T}(\Delta) \subseteq \mathsf{Form}(\mathcal{L})$, by*

$$\mathcal{T}(\Delta) = \Delta \cup \{\neg\varphi \mid \varphi \in \mathcal{O} \text{ and } \neg(\forall \beta \in \mathsf{Mod}_W(\Delta).\ \beta \models \varphi)\}.$$

Then $\alpha \in \mathsf{Mod}_W(\Delta)$ is minimum iff $\alpha \models \mathcal{T}(\Delta)$.
Proof. *If $\alpha \in \mathsf{Mod}_W(\Delta)$ is minimum, then for all $\varphi \in \mathcal{O}$,*

$$\alpha \models \varphi \Leftrightarrow \Delta \models_W \varphi$$

(by definition of minimum). Hence, for $\varphi \in \mathcal{O}$,

$$\Delta \not\models_W \varphi \Rightarrow \alpha \models \neg\varphi.$$

Thus $\alpha \models \mathcal{T}(\Delta)$.

Conversely, suppose $\alpha \models \mathcal{T}(\Delta)$. Then for all $\varphi \in \mathcal{O}$, if $\alpha \models \varphi$ then $\Delta \models_W \varphi$ (since, if not i.e. $\Delta \not\models_W \varphi$ then $\alpha \models \neg\varphi$ – contradiction). Hence, $\forall \beta \in \mathsf{Mod}_W(\Delta).\ \beta \models \varphi$. Thus α is minimum.

Corollary 3.1 (Existence of minimum models)**.** *Consider a theory W of $\mathcal{L}$ and $\Delta \subseteq \mathcal{O}$, for a set of closed $\mathcal{L}$-formulae $\mathcal{O}$. A minimum model exists in $\mathsf{Mod}_W(\Delta)$ iff $\mathcal{T}(\Delta)$ is W-consistent, i.e. there is a model $\gamma \in \mathsf{Mod}_W(\Delta)$ with $\gamma \models \mathcal{T}(\Delta)$.*

Example 3.3. *Revisiting the Blocks World example, consider an extension so that blocks have colours with, say, just two colours, red and green, with axiom*

$$\forall x : Block.\ red(x) \Leftrightarrow \neg green(x).$$

Let the colours be observable, i.e. for each block x, $red(x), green(x) \in \mathcal{O}$. Consider the case of two blocks A and B only. Let

$$\Delta = \{on(A,B), red(A)\}.$$

Then Δ has no minimum models since the colour of B is not specified. Hence, $\neg red(B) \in \mathcal{T}(\Delta)$ and $\neg green(B) \in \mathcal{T}(\Delta)$ and so $\mathcal{T}(\Delta)$ is inconsistent in the presence of the above axiom. However, if

$$\Delta = \{on(A,B), red(A), green(B)\},$$

then Δ has minimum models.

We now characterise the satisfaction relation $\mid\!\approx$ in terms of $\models$.

Corollary 3.2 (Characterising satisfaction for minimum models)**.** *Consider a theory W of $\mathcal{L}$ and $\Delta \subseteq \mathcal{O}$, for a set of closed $\mathcal{L}$-formulae $\mathcal{O}$. Define $\mathcal{T}(\Delta) \subseteq \mathsf{Form}(\mathcal{L})$ as above (Theorem 3.1). Then, for any closed $\psi \in \mathsf{Form}(\mathcal{L})$, we have*

$$\Delta \mid\!\approx_W \psi \;\; \textit{iff} \;\; \mathcal{T}(\Delta) \models_W \psi.$$

Proof. *Assume $\Delta \mid\!\approx_W \psi$, i.e. for all $\alpha \in \mathsf{Mod}_W(\Delta)$ minimum, $\alpha \models \psi$. But by Theorem 3.1, α is minimum iff $\alpha \models \mathcal{T}(\Delta)$. Thus, for all $\alpha \in \mathsf{Mod}_W(\Delta)$ with $\alpha \models \mathcal{T}(\Delta)$, $\alpha \models \psi$, i.e. $\mathcal{T}(\Delta) \models_W \psi$.*

Conversely, assume $\mathcal{T}(\Delta) \models_W \psi$, i.e. $\forall \alpha \in \mathsf{Mod}_W(\Delta).\ \alpha \models \mathcal{T}(\Delta) \Rightarrow \alpha \models \psi$. But $\alpha \models \mathcal{T}(\Delta)$ iff α is minimum. Hence $\Delta \mid\!\approx_W \psi$ as required.

4 Proof systems

We now turn to proof systems for minimum model reasoning. We begin with a negative result for reasoning in its full generality.

Proposition 4.1. *The minimum model satisfaction relation $\mid\!\approx_W$ is* not *semidecidable.*
Proof. *The non-halting problem for Turing machines is expressible in minimum model reasoning. See [30] for details.*

A direct consequence of Proposition 4.1 is that *there is no general proof system for minimum model reasoning that is sound, complete and effective* (where 'effective' means that the collection of proofs forms a decidable set).

Thus, to formulate a proof system we need either to introduce restrictions on the form of theories or of observations (or both), or to relax the completeness requirement for the proof system.

We begin by introducing a condition on the form of the theories and observations:

Definition 4.1. *Given $\mathcal{O} \subseteq Sen(\mathcal{L})$ and theory W, we say $\Delta \subseteq \mathcal{O}$ is* $\mathcal{O}$-closed with respect to *W if $\{\psi \mid \psi \in \mathcal{O}$ and $\Delta \vDash_W \psi\} \subseteq \Delta$.*

Definition 4.2. *Given $\mathcal{O} \subseteq Sen(\mathcal{L})$ and theory W, we say W is* $\mathcal{O}$-independent *iff for every $\Delta \subseteq \mathcal{O}$ which is W-consistent, Δ is $\mathcal{O}$-closed with respect to W.*

These are strong conditions on the form of observations and the accompanying theory, saying that the only observations derivable from a set of observations are those already in the set. However they enable us to give a sound and complete proof system which has applications, as we now show. Later we will consider weakening these conditions.

Let $W \subseteq \mathsf{Sen}(\mathcal{L})$ be a first-order theory. Let Υ be a sound and complete set of Natural Deduction style inference rules for classical first-order logic with equality and axioms W. Inference rules in Υ are of the form:

$$\frac{\Gamma_1 \vdash \psi_1 \quad \ldots \quad \Gamma_n \vdash \psi_n}{\Gamma \vdash \psi}$$

where the contexts $\Gamma, \Gamma_i \subseteq \mathsf{Form}(\mathcal{L})$ for $1 \leq i \leq n$ are sets and $\psi, \psi_i \in \mathsf{Form}(\mathcal{L})$ for $1 \leq i \leq n$. For each axiom φ in theory W, there is a rule

$$\Gamma \vdash \varphi.$$

The projection rule is, for $\varphi \in \Gamma$,

$$\Gamma \vdash \varphi.$$

We now modify Υ to express minimum model reasoning for the case when Δ is $\mathcal{O}$-closed w.r.t W. Let Υ_1 be the proof system obtained by changing the sequents in the rules of Υ as follows: sequents $\Gamma \vdash \varphi$ become

$$\mathcal{O}; \Delta; \Gamma \ \vdash_1 \varphi$$

for $\mathcal{O} \subseteq \mathsf{Sen}(\mathcal{L})$ and finite $\Delta \subseteq \mathcal{O}$. For the sequents we consider, Δ is finite, but the development below may be extended to infinite Δ.

The projection rules are, for $\varphi \in \Gamma$ or $\varphi \in \Delta$,

$$\mathcal{O}; \Delta; \Gamma \ \vdash_1 \varphi.$$

Finally, we add the *Minimum Model Rule*:

$$\frac{\varphi \in \mathcal{O} \quad \varphi \notin \Delta}{\mathcal{O}; \Delta; \Gamma \ \vdash_1 \neg\varphi}$$

This proof system is *sound and complete* for minimum model reasoning in the case that Δ is $\mathcal{O}$-closed w.r.t W, in the following sense:

Theorem 4.1 (Characterising the soundness of Υ_1)**.** *Let $\mathcal{O} \subseteq \mathsf{Sen}(\mathcal{L})$ and $W \subseteq \mathsf{Sen}(\mathcal{L})$ with $\Delta \subseteq \mathcal{O}$ finite, then for all $\varphi \in \mathsf{Sen}(\mathcal{L})$,*

$$\mathcal{O}; \Delta; W \ \vdash_1 \varphi \implies \Delta \mid\!\approx_W \varphi$$

iff there is no minimum model in $\mathsf{Mod}_W(\Delta)$ or Δ is $\mathcal{O}$-closed w.r.t. W.

Thus, for the proof system to be sound, every finite Δ needs to be $\mathcal{O}$-closed w.r.t. W, hence:

Corollary 4.1 (Soundness of Υ_1 for $\mathcal{O}$-independence)**.** *Let $\mathcal{O} \subseteq \mathsf{Sen}(\mathcal{L})$ and $W \subseteq \mathsf{Sen}(\mathcal{L})$ be $\mathcal{O}$-independent, then, for every finite $\Delta \subseteq \mathcal{O}$,*

$$\mathcal{O}; \Delta; W \ \vdash_1 \varphi \implies \Delta \mid\!\approx_W \varphi.$$

Theorem 4.2 (Completeness of Υ_1)**.** *Let $\mathcal{O} \subseteq \mathsf{Sen}(\mathcal{L})$ and $W \subseteq \mathsf{Sen}(\mathcal{L})$ then, for every finite $\Delta \subseteq \mathcal{O}$,*

$$\Delta \mid\!\approx_W \varphi \implies \mathcal{O}; \Delta; W \ \vdash_1 \varphi.$$

Proof. *Note that completeness does not require any restrictions on Δ or $\mathcal{O}$. The soundness results follow from the characterisation and existence of minimum models in the previous section (Theorem 3.1 and Corollaries 3.1 and 3.2). Completeness follows directly from the completeness of first-order logic.*

This proof system suffices for the Blocks World example of Section 2. Indeed, this proof system Υ_1 is embedded in ESAT (Evolvable Systems Animator Tool) [1, 2] and has been used for modelling a range of examples of evolvable computational systems.

However, the conditions of $\mathcal{O}$-closedness and $\mathcal{O}$-independence are strong, and it is natural to explore how these may be relaxed. This is the content of the remainder of this section. There are various directions we may explore – we consider just two in this note (1) using decidable subsets, and (2) relaxing the completeness requirement.

4.1 Using decidable fragments of first-order logic

The above proof system is based on reducing the condition $\Delta \nvdash \varphi$ to the test $\varphi \notin \Delta$ which, for finite Δ is decidable. Here we consider another approach to decidability, namely using *a decidable fragment of first-order logic*. We do not require that the theory itself is decidable, only that the interaction of observations with the theory is decidable in a sense that we make precise below.

We consider a general method of incorporating decidable fragments, illustrating it with examples of fragments that are sets of sentences in an untyped first-order logic in Prenex Normal Form. Many decidable fragments of first-order logic (see, for example, [8]) are expressed in this form with constraints on the quantification and the presence or absence of function symbols and of equality. This explains the form of the definitions below. Decidable fragments of first-order logic in other forms may be handled similarly.

Consider a class of formulae $\mathcal{I}$ which forms a decidable fragment of (untyped) first-order logic. Consider a theory $W \subseteq \mathsf{Sen}(\mathcal{L})$ which is finite (a finite number of axioms) and observations $\mathcal{O}$ which are quantifier-free sentences. Let W^U be an untyped form of W (i.e. for every $\psi \in \mathsf{Sen}(\mathcal{L})$, we have $W \vdash \psi \iff W^U \vdash \psi^U$ where ψ^U is the standard untyped conversion of ψ). We say a set $S \subseteq W$ is an $\mathcal{I}$-decidable subtheory of W if $PNF(\bigwedge S^U) \in \mathcal{I}$, where PNF is a Prenex Normal Form.

Now let $\mathcal{I}$ be a decidable fragment of first-order logic and $S \subseteq W$ an $\mathcal{I}$-decidable subtheory of theory W. We define the proof system $\Upsilon_2(\mathcal{I}, S)$ with entailment relation $\vdash_2^S$ to be the proof system Υ_1 replacing the Minimum Model Rule with:

$$\frac{\psi \in \mathcal{O} \quad \Delta \nvdash_S \psi}{\mathcal{O}; \Delta; \Gamma \ \vdash_2^S \ \neg\psi}$$

Here $\vdash_S$ is standard first-order entailment under theory S. The idea is that, if sentences in the theory W which express dependencies between observations are in the class $\mathcal{I}$, the proof rule above may be applied by using the decidability of $\Delta \nvdash_S \psi$ instead of its provability.

Under suitable conditions, this proof system is sound, complete and effective:

Theorem 4.3 (Soundness for $\mathcal{O}$-closedness in $\Upsilon_2(\mathcal{I}, S)$)**.** *Let $\mathcal{O} \subseteq \mathsf{Sen}(\mathcal{L})$ be a set of quantifier-free sentences and $W \subseteq \mathsf{Sen}(\mathcal{L})$ a finite theory of $\mathcal{L}$. Let $S \subseteq W$ be an $\mathcal{I}$-decidable subtheory of theory W. Then, for every $\psi \in \mathsf{Sen}(\mathcal{L})$ and every finite $\Delta \subseteq \mathcal{O}$, such that $\{\varphi \in \mathcal{O} \mid \Delta \models_S \varphi\}$ is $\mathcal{O}$-closed with respect to W,*

$$\mathcal{O}; \Delta; W \ \vdash_2^S \psi \Longrightarrow \Delta \mid\!\approx_W \psi.$$

Thus, we replace the strong requirement that Δ is $\mathcal{O}$-closed w.r.t W, with the weaker requirement that $\{\varphi \in \mathcal{O} \mid \Delta \models_S \varphi\}$ is $\mathcal{O}$-closed w.r.t W so that, if all dependencies amongst observations are in a decidable S, then we can decide whether an observation follows from Δ or not.

Theorem 4.4 (Completeness for $\Upsilon_2(\mathcal{I}, S)$)**.** *Let $\mathcal{O} \subseteq \mathsf{Sen}(\mathcal{L})$ be a set of quantifier-free sentences and $W \subseteq \mathsf{Sen}(\mathcal{L})$ a finite theory of $\mathcal{L}$. Let $S \subseteq W$ be an $\mathcal{I}$-decidable subtheory of theory W. Then, for every $\psi \in \mathsf{Sen}(\mathcal{L})$ and every finite $\Delta \subseteq \mathcal{O}$,*

$$\mathcal{O}; \Delta; W \ \vdash_2^S \psi \Longleftarrow \Delta \mid\!\approx_W \psi.$$

Theorem 4.5 (Effectiveness for $\Upsilon_2(\mathcal{I}, S)$)**.** *Let $\mathcal{L}$ be a typed, first-order language with a finite number of constant symbols. Let $\mathcal{O} \subseteq \mathsf{Sen}(\mathcal{L})$ be a recursive set of quantifier-free formulae,*

recursive $W \subseteq \mathsf{Sen}(\mathcal{L})$ *be a finite theory of* $\mathcal{L}$*,* $S \subseteq W$ *be an* $\mathcal{I}$*-decidable subtheory of theory* W*, and finite* $\Delta \subseteq \mathcal{O}$*. If the set* $\mathcal{I}$ *is recursive, then* $\Upsilon_2(\mathcal{I}, S)$ *is effective i.e. the collection of proofs in the proof system* $\Upsilon_2(\mathcal{I}, S)$*, for a given* $\mathcal{O}$*,* Δ *and* W*, forms a recursive set.*

For more details of these results and examples of decidable fragments, see [31].

The technique of this section provides a further approach to the design and simulation of evolvable systems using revision-based logic and monitored properties of a system based on observations. Some limited experimentation [31] suggests that this approach with quite simple decision procedures may be useful for modelling a range of computational systems. Clearly, more experimentation is required, and, for this purpose, simulation tools such as ESAT [1, 2] could be extended with this decidability technique.

4.2 Incomplete proof systems

The disadvantage of the proof systems above is that if the set Δ is not $\mathcal{O}$-closed, or the observational closure of Δ with respect to a part of the theory is not $\mathcal{O}$-closed with respect to the full theory, then the systems become unsound. The application of minimum models in the description of evolvable systems involves changing the theory dynamically [4]. As the theory changes, testing the $\mathcal{O}$-closedness condition becomes impractical. One possible solution is to determine how changes to theories affect $\mathcal{O}$-closedness. Here, we take another route and consider proof systems which are sound and, though in general incomplete, have a "degree of completeness".

Let $\Vdash$ be a relation of type $\mathcal{P}(\mathsf{Sen}(\mathcal{L})) \times \mathcal{P}(\mathsf{Sen}(\mathcal{L})) \times \mathsf{Sen}(\mathcal{L})$ where $\mathcal{P}$ is the powerset operator. For $\Delta \in \mathcal{P}(\mathsf{Sen}(\mathcal{L}))$, $W \in \mathcal{P}(\mathsf{Sen}(\mathcal{L}))$ and $\psi \in \mathsf{Sen}(\mathcal{L})$, we write $\Delta \Vdash_W \psi$. We call $\Vdash$ *admissible* iff

$$\Delta \Vdash_W \varphi \Longrightarrow \Delta \nvdash_W \varphi.$$

If, in addition, $\Vdash$ is also recursive we say that $\Vdash$ is a recursive admissible relation.

For admissible relation $\Vdash$, let $\Upsilon_3(\Vdash)$ be the proof system with entailment relation $\vdash_3$ obtained from Υ_1 by changing the Minimum Model Rule to:

$$\frac{\psi \in \mathcal{O} \quad \Delta \Vdash_W \psi}{\mathcal{O}; \Delta; \Gamma \ \vdash_3 \ \neg\psi}$$

This proof system is sound:

Theorem 4.6 (Soundness for $\Upsilon_3(\Vdash)$)**.** *Let* $\mathcal{O} \subseteq \mathsf{Sen}(\mathcal{L})$ *and* $W \subseteq \mathsf{Sen}(\mathcal{L})$*, then, for every* $\psi \in \mathsf{Sen}(\mathcal{L})$ *and every finite* $\Delta \subseteq \mathcal{O}$*:*

$$\mathcal{O}; \Delta; W \ \vdash_3 \psi \Longrightarrow \Delta \mid\!\approx_W \psi$$

Effectiveness is a direct consequence of the recursive admissibility of $\Vdash$:

Theorem 4.7 (Effectiveness for $\Upsilon_3(\Vdash)$)**.** *Given recursive* $\mathcal{O} \subseteq \mathsf{Sen}(\mathcal{L})$ *and recursive* $W \subseteq \mathsf{Sen}(\mathcal{L})$ *and finite* $\Delta \subseteq \mathcal{O}$*. If* $\Vdash$ *is a recursive admissible relation then* $\Upsilon_3(\Vdash)$ *is effective, i.e the collection of proofs in the proof system* $\Upsilon_3(\Vdash)$*, for a given* $\mathcal{O}$*,* Δ *and* W*, forms a recursive set.*

What recursive admissible relations $\Vdash$ are available? We describe briefly a few options.

Recursive relations $\Vdash$ may be constructed from decidable fragments of first-order logic (see the previous section). For a given decidable fragment $\mathcal{I}$, the 'degree of completeness' varies from standard first-order logic through to complete systems for the logic of minimum models,

depending on the extent to which the interaction of observables with the theory lies within the decidable fragment (i.e. the scope of *$\mathcal{I}$-decidable subtheories* defined in the previous section).

Another construction of suitable recursive relations $\Vdash$ is based on searching for finite models to falsify a sentence. For this to generate admissible relations we need an additional property which we call the *finite representation property*:

Definition 4.3. *Let $\mathcal{C} \subseteq \mathsf{Sen}(\mathcal{L})$ be a non-empty set of sentences. Let $\mathcal{O} \subseteq \mathcal{C}$ and $W \subseteq \mathsf{Sen}(\mathcal{L})$ be a finite theory of $\mathcal{L}$. Let $\Delta \subseteq \mathcal{O}$ be finite. Define the class $\mathsf{FMod}_W(\Delta)$ to be the class of W-models α that satisfy Δ and have a finite domain for each type. We say that the pair $\langle W, \Delta\rangle$ has the* $\mathcal{C}$-finite representation property *if for every $\psi \in \mathcal{C}$*

$$\Delta \models_W \psi \iff \text{for every } \alpha \in \mathsf{FMod}_W(\Delta),\ \alpha \models \psi.$$

For a finitely axiomatised theory W, a finite Δ, a recursive $\mathcal{C}$ and $\langle W, \Delta\rangle$ having the $\mathcal{C}$-finite representation property, the following terminating routine defines a recursive admissible relation $\Delta \Vdash_W \psi$:

if $\Delta \subseteq \mathcal{C}$ and $\psi \in \mathcal{C}$ **then**
  Test in parallel: $\Delta \vdash_W \psi$ and $\Delta \perp_W \psi$
  If first condition ends first: Output NO
  If second condition ends first: Output YES
else
  Output NO
end if

where $\Delta \perp_W \psi$ means that there is a finite model that satisfies $W \cup \Delta$ but does not satisfy ψ.

A logic which defines all types using a finite enumeration of constants, so forcing models to be finite, has the $\mathcal{C}$-finite representation property for any $\mathcal{C}$. More generally, logics with the finite model property have the $\mathcal{C}$-finite representation property for any $\mathcal{C}$. However, other logics are of interest. In [31], we explore conditions on the form of sentences of theories which enforce the $\mathcal{C}$-finite representation property – conditions such as limiting existential quantification to cases where existence always has a closed term witness in the logic, or logics where all function symbols in the theory have accompanying primitive recursive definition schemas. These conditions suggest various applications, including to 'database-like' theories.

5 Implementation in a logical framework

We now turn to implementing the above proof systems by expressing them in a logical framework.

Logical frameworks are formal systems which enable us to define logics. They typically consist of a definition mechanism and a methodology for expressing logics in the formal system. The definition mechanism allows us to name and define the elements of the logic, including connectives, quantifiers, modal operators, predicate and function names and proof rules. Meta-level results establish that, with suitable definitions, the structure of the logic, including the construction of terms and formulae and the notions of provability and the structure of proofs, are correctly encoded. Logical frameworks such as the Edinburgh Logical Framework [17] and Isabelle [24] are based on typed λ-calculi, using λ-terms to describe the structure of proofs, and also to describe functional constructs in the logic, such as proof transformations. Types allow us to specify the functional behaviour of the elements of the logic and to define parameterised collections of proofs.

The advantage of this approach to implementing logics is that features common to many logics, such as the construction of terms and formulae, the handling of variables with binding, scope and variable contexts, the notion of sequents and rules, and the concept of proofs and provability, are handled once in the logical framework and are then available for free for any logic that can be suitably defined. Moreover, the process of defining a logic in a logical framework gives insight into the underlying structure of a logic, for example forcing us to consider how inference rules are applied and how a logic may have more than one notion of provability. Not all logics may be embedded in a logical framework in a 'natural manner' – some logics have, for example, a non-standard treatment of variables or notion of proof, or require extra-logical features in their definition.

The proof systems for reasoning in minimum models described in the previous section have non-standard features and pose some challenges for a logical framework. For example, we need to distinguish the treatment of formulae as syntactic entities (as elements of a set) from formulae as logical entities (identifiable up to logical equivalence). We also need to incorporate decidable relations and decision procedures into proof systems. We show how these issues are addressed by encoding the proof systems in Isabelle [24], a typed logical framework which incorporates resolution based on higher-order unification. Isabelle turns out to have useful features in its approach to encoding logics, in the facility for defining proof tactics, and in its use of the programming language Standard ML [23].

In Isabelle, the theory `HOL` (classical higher-order logic) is defined in terms of a meta-logic called `Pure`. `Pure` is an intuitionistic higher-order logic with implication (`==>`) and universal quantifier (`!!`). `HOL` is built by defining the classical logical operators: universal quantifier (`ALL`), existential quantifier (`EX`), implication (`-->`), equality (`=`), and (`&`), or (`|`), negation (`~`), bi-conditional (`<->`). We use expressions in `Pure` to define the inference rules in `HOL`. For example, the Modus Ponens rule is expressed as

```
(P --> Q) ==> P ==> Q
```

which is usually written as `[| P --> Q; P |] ==> Q`. The symbols `P` and `Q` are propositional variables; `P --> Q` is a well-formed expression in `HOL`. The operator `==>` is the implication operator of `Pure`.

We show how we may implement in Isabelle the proof systems of the previous section.

In order to use the inference rules of classical logic already encoded in Isabelle, we modify the above proof systems so that, for sequents $\mathcal{O}; \Delta; W \vdash_1 \psi$, we treat $\mathcal{O}$ and as globally defined and add Δ and the axioms of W to the 'working set' of formulae used in proof construction.

We illustrate how the proof system Υ_1 is implemented using a form of the Blocks World of Section 2.

```
theory BlocksWorld
imports Main
  datatype Block = A | B | C | D
  datatype Formula = on Block Block | above Block Block
                           | And Formula Formula
consts
  Obs      :: "Formula set"
  Delta    :: "Formula set"
```

The set of blocks is defined using an enumeration type. A key idea is to introduce a type of formulae, `Formula`, to represent the syntax of formulae (we use a simple example here with the single connective of conjunction). Both `Obs` (for the set $\mathcal{O}$) and `Delta` (for Δ) are sets of

formulae which we define below. In order to use the internal logic of Isabelle, we need to treat formulae logically. To do so, we introduce an explicit conversion function, `convert`, with result of type `bool` in `HOL`, so that `convert(x)` is treated as a logical formula by Isabelle:

```
fun convert :: "Formula => bool"   where
"(convert (And x y))  = ((convert x) & (convert y))"
```

We define example sets of observations, using set comprehension in Isabelle:

```
defs
  Obs_def :
    "Obs == {t. EX x::Block. EX y::Block. (t = (on x,y))}"
  Delta_def :
    "Delta == {r. (r = (on A B) | r = (on C D) )}"
```

Now we add the axioms of the Blocks World theory, for example:

```
ax1: "ALL x::Block. ALL y::Block.
         (convert (on x y)) --> (convert (above x y))"
```

Finally, we define the Minimum Model Rule:

```
min_rule: "[| P : Obs; P ~: Delta |] ==> ~(convert P)"
```

This is a direct translation of the Minimum Model Rule of Υ_1 into `Pure`, distinguishing syntactic occurrences of formulae from logical occurrences.

This outlines the key implementation steps for the proof system Υ_1. We are now in a position to develop proofs in this system using Isabelle. For examples of such proofs, see [31].

We now turn to the proof system $\Upsilon_2(\mathcal{I}, S)$ using a decidable fragment $\mathcal{I}$ of first-order logic. We use the same `BlocksWorld` theory and the same treatment of the sets of formulae $\mathcal{O}$ and Δ.

In $\Upsilon_2(\mathcal{I}, S)$, we test the assumption $\Delta \nvdash_S \psi$ in the Minimum Model Rule via a decision procedure. There are two possible treatments of this in Isabelle: (1) introduce a decision procedure through the definition of a computable function, *together with a proof of the termination of the function definition*; or, (2) define a decision procedure externally in a programming language and then incorporate it into the proof construction tools of Isabelle. In the latter case, proofs will be correct only up to the correctness of the external definitions. However, in the former case, we need to supply a proof of termination, which in general requires the definition of a suitable well-founded order.

To illustrate how we may develop a form of the proof system $\Upsilon_2(\mathcal{I}, S)$ we choose option (2) and, as an example of a decision procedure, we use one of the automated search tactics built into Isabelle.

We first define the structures used to incorporate a decision procedure into Isabelle and then define the Minimum Model Rule of $\Upsilon_2(\mathcal{I}, S)$:

```
datatype Answer = yes | no
consts
  provable :: "Formula => Answer"
axioms
  min__rule: "[| P : Obs; (provable P) = no |] ==> ~(convert P)"
  pos_rule: "(provable P) = yes ==> convert P"
```

Here the declaration `provable` is what will be instantiated with a decision procedure. The `pos_rule` says that, if a formula can be established as provable, then it holds in `HOL`.

How do we apply the Minimum Model Rule for $\Upsilon_2(\mathcal{I}, S)$? The current goal in the proof becomes `(provable P) = no` where `P` is a formula that has been established to be in `Obs` and we are given a list of axioms representing the decidable fragment S. We now invoke an Isabelle tactic that we have encoded as a program in the language Standard ML [23]. The tactic augments the current proof state by extracting `P` from the expression `(provable P) = no`, adds a new proof state with the subgoal `convert(P)` and the working set containing the list of axioms provided (in S). We then execute a tactic that implements the decision procedure. If the tactic fails, discharge the subgoal. If the tactic succeeds, return failure. The decision procedure itself is supplied as an 'oracle' which allows us to build theorems in Isabelle without providing a proof. Details of this construction may be found in [31].

Similar techniques may be used to implement the incomplete proof systems of Section 4.2. In these implementations, as well as using the definitional mechanisms in the λ-calculus of Isabelle, we have made essential use of the 'open' nature of this logical framework which allows us to develop proof tactics and to incorporate ML code for decision procedures.

In terms of applications to the description of computational systems and animating revision logic specifications, the current ESAT (Evolvable Systems Animator Tool) [1, 2] uses a range of automated first-order theorem provers (15 in all in the current system), including resolution-based provers and SAT-solvers, all running in parallel ('in competition'), together with the minimum model rule of Υ_1 above (i.e. requiring $\mathcal{O}$-closedness). The implementation of proof systems in a logical framework described here opens up another avenue to system simulation in which we use the proof construction tools of Isabelle, which include some automated methods, techniques using proof tactics which we may build, and direct constructions of proofs in the underlying logic. The various proof systems of Section 4 may thus be incorporated into the animation tool, providing us with an experimental platform for exploring revision-based descriptions of computational systems using these proof systems.

6 Conclusions

We have defined a notion of minimum model under sets of observations and described its application to the revision-based description of computational systems. We have set out the landscape of reasoning systems for logics of minimum models, presenting the difficulty in the way of a general solution and exploring some of the restricted solutions. Other approaches may be worth investigating, e.g. circumscription methods [21, 13, 26, 32, 19, 20] whereby a second-order axiom (or a first-order axiom schema) may be instantiated to provide first-order reasoning.

Outside this search for reasoning techniques, there are several other topics worth investigating. We have focussed on modelling the state of computation in a revision-based logic. The other ingredient is the process of revision itself. We need to understand how to reason about revisions and their interaction with the minimum models introduced here. Moreover, to apply this to the formalisation of evolvable systems, we need also to consider how to accommodate changes in the theory into our methods of reasoning, so that we may reason incrementally about evolutionary computations.

References

[1] D. Afifi, D. Rydeheard, and H. Barringer. Automated reasoning in the simulation of evolvable systems. In *FLOC 2010 Workshop on Practical Aspects of Automated Reasoning (PAAR)*, 2010.

[2] D. Afifi, D. Rydeheard, and H. Barringer. ESAT: A tool for animating logic-based specifications of evolvable computing systems. In *Runtime Verification, RV 2010. LNCS 6418*, 2010.

[3] D. Balasubramaniam, R. Morrison, G. N. C. Kirby, K. Mickan, B. C. Warboys, I. Robertson, R. A. Snowdon, R. M. Greenwood, and W. Seet. A software architecture approach for structuring autonomic systems. In *Proceeding of ICSE 2005 Workshop on the Design and Evolution of Autonomic Application Software (DEAS 2005)*, St Louis, MO, USA, 2005. ACM Digital Library.

[4] H. Barringer, D. Gabbay, and D. Rydeheard. Modelling evolvable component systems: Part I - A logical framework. *Logic Journal of IGPL*, 17(6):631–696, 2009.

[5] H. Barringer, Dov Gabbay, and David Rydeheard. From runtime verification to evolvable systems. In *7th International Workshop, Runtime Verification 2007*, pages 97–110, Vancouver, Canada, 2007. Lecture Notes in Computer Science 4839 (2007) Springer.

[6] H. Barringer, D. Rydeheard, and D. Gabbay. A logical framework for monitoring and evolving software components. In *Proceeding of the First Joint IEEE/IFIP Symposium on Theoretical Aspects of Computer Science (TASE 2007)*, Shanghai, China, June 2007. IEEE Computer Society Press.

[7] H. Barringer, D. Rydeheard, B. C. Warboys, and D. M. Gabbay. A revision-based logical framework for evolvable software. In *Proceeding of IASTED International Multi-Conference: Software Engineering (SE07)*, pages 78–83, Innsbruck, Austria, February 2007.

[8] E. Börger, E. Grädel, and Y. Gurevich. *The Classical Decision Problem*. Perspectives in Mathematical Logic, Springer-Verlag, 1997.

[9] M. Cadoli and M. Lenzerini. The complexity of closed world reasoning and circumscription. In *AAAI-90 Proceedings*, pages 550–555, 1990.

[10] M Cadoli and M. Schaerf. A survey of complexity results for non-monotonic logics. *J. Logic Programming*, 17:127–160, 1993.

[11] S. A. Cook and Yongmei Liu. A complete axiomatization for blocks world. *J. Logic and Computation*, 13(4), 2003.

[12] M. Davis. The mathematics of non-monotonic reasoning. *Artificial Intelligence*, 13:73–80, 1980.

[13] D. W. Etherington, R. E. Mercer, and R. Reiter. On the adequacy of predicate circumscription for closed-world reasoning. *Computational Intelligence*, 1:11–15, 1985.

[14] R. E. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3–4):189–208, 1971.

[15] M. R. Geneseret and N. J. Nilson. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publishers, 1987.

[16] R. M. Greenwood, I. Robertson, B. C. Warboys, and B. S. Yeomans. An evolutionary approach to process system development. In *Proceedings of the International Process Technology Workshop*, Villard de Lans (Grenoble), 1999.

[17] Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. *Journal of the ACM*, 40(1):194–204, 1993.

[18] J. Hintikka. Model minimization - an alternative to circumscription. *Journal of Automated Reasoning*, 4:1–13, 1988.

[19] V. Lifschitz. Pointwise circumscription. In *Proceedings of the 5th National Conference on Artificial Intelligence (AAAI'86)*, pages 406–410, 1986.

[20] V. Lifschitz. Circumscription. In *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3*, pages 297–352. Oxford University Press, 1994.

[21] J. McCarthy. Circumscription—a form of non-monotonic reasoning. *Artificial Intelligence*, 13:27–

39, 1980.

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

[23] R. Milner, M. Tofte, R. Harper, and D. MacQueen. *The Definition of Standard ML (Revised)*. MIT Press, 1997.

[24] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer-Verlag, 2002.

[25] F. Oquendo, B. C. Warboys, R. Morrison, R. Dindeleux, F. Gallo, H. Garavel, and C. Occhipinti. Archware: Architecting evolvable software. In *EWSA*, pages 257–271, 2004.

[26] D. Perlis and J. Minker. Completeness results for circumscription. *Artificial Intelligence*, 28(1):29–42, 1986.

[27] R. Reiter. On closed world databases. In H. Gallaire and J. Minker, editors, *Logic and Data Bases*, New York, 1978. Plenum.

[28] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[29] P. Rondogiannis and W. W. Wadge. Minimum model semantics for logic programs with negation-as-failure. *ACM Trans. Comput. Logic*, 6(2):441–467, 2005.

[30] J. H. D. Sánchez. A proof of the not-semidecidability of minimum model entailment. 2010. Available at the following URL: http://www.cs.manchester.ac.uk/evolve.

[31] J. H. D. Sánchez. Minimum models: Reasoning and automation. Master's thesis, School of Mathmatics, University of Manchester, U.K., 2009. Available online at: http://www.cs.manchester.ac.uk/evolve.

[32] J. C. Shepherdson. A sound and complete semantics for a version of negation as failure. *Theor. Comput. Sci.*, 65(3):343–371, 1989.

[33] M. A. Suchenek. Forcing versus closed world assumption. In *Methodologies for Intelligent Systems*, pages 453–460. North-Holland, 1987.

[34] M. A. Suchenek. First-order syntactic characterizations of minimal entailment, domain-minimal entailment, and Herbrand entailment. *J. Automated Reasoning*, 10(2):237–263, 1993.

[35] T. Winograd. *Understanding Natural Language*. Academic Press, New York, 1972.