Applications of Abduction #3: "Black-Box" to "Gray-Box" Models

Tim Menzies * Simon Goss †

November 6, 1995

Abstract

Military Operations Research makes extensive use of large complex simulation models. These simulation models are often *black-boxes*; i.e. they are opaque and incomprehensible. A considerable effort is involved in commissioning a new model into service. We characterise this activity as the construction of *gray-box* approximate causal models of the functional behaviour of *black-box* simulation software.

While the imported black-box models are typically numeric, determinate and precise, their associated gray-box models are under-specified, indeterminant and vague. Here we explore the use of the `HT4` *abductive inference* engine to support the process of using ad-hoc experience with black-box models to construct and maintain partially-specified gray-box models.

1 Introduction

Military Operations Research (OR) makes extensive use of large complex simulation models. These can be the result of many person-years of development and incorporate modules obtained from external sources. It is common that such a system may be obtained from a third party. These simulation models are often *black-boxes*; i.e. they are opaque and incomprehensible. A considerable effort is involved in commissioning a new model into service. In the process, familiarity and expertise is gained in the model. The black-box nature of these simulation systems complicates their verification and validation for local conditions. Customisation is also difficult for the same reason.

We characterise this activity as the construction of *gray-box* approximate causal models of the functional behaviour of the black-box simulation software. Such gray-box models have several advantages. They can serve to *document* and *preserve* the expertise gained in the pain of commissioning the system. Further, such gray-box models can *inspected*, *verified* and *validated*, thus increasing our confidence in the results obtained using the software. Finally, such gray-box models can simplify *customisation* of legacy systems.

While the imported black-box models are typically numeric, determinate and precise, their associated gray-box models are under-specified, indeterminant and vague. Yet these gray-box models represent our best understanding of the innerworkings of complex black-box models. Here we explore the use of the `HT4` [22, 24] *abductive inference* engine to support the process of using ad-hoc experience with black-box models to construct and maintain partially-specified gray-box models.

This paper is structured as follows. Section 2 of this document reviews the problem of commissioning a model obtained from a remote site to produce a list of requirements on a methodology for validating black-box models. Section 3 describes our preferred abductive framework. Section 4 argues that the requirements in section 2 can be meet by the `HT4` abductive inference engine. Section 5 discusses related work in the qualitative reasoning, abductive, and truth maintenance literature. The conclusion discusses further work.

Note that portions of this work have appeared previously (see [24]).

2 Using Remote Models

The remote model commissioning problem is summarised in Figure 1. Some group called TEAM-1 derives some model $\mathcal{M}_1$ representing their *initial understanding* of a problem (e.g. modeling the

*Dept. of Software Development, Monash University, Caulfield East, Melbourne, VIC, Australia, 3145; +61-3-903-1033; +61-3-903-1077(fax);
Email: `timm@insect.sd.monash.edu.au`;
URL: `http://www.sd.monash.edu.au/~timm`

†Defence Science and Technology Organisation- Air Operations Division, PO Box 4331 Melbourne, VIC, Australia, 3001. *Email*: `simon.goss@dsto.defence.gov.au`

performance of a fighter aircraft). This model is *operationalised* in some third generation language to become $\mathcal{M}_2$. Perhaps an attempt is made to document $\mathcal{M}_1$ in a *manual* $\mathcal{M}_3$. $\mathcal{M}_1$ is commonly a research prototype comprising thousands or hundreds of thousands of lines of code. Hence $\mathcal{M}_3$ is typically incomplete. $\mathcal{M}_2$ and $\mathcal{M}_3$ are then shipped to another site where a second team (TEAM-2) tries to understand them. Conceptually, TEAM-2 builds $\mathcal{M}_4$, a model representing the *local understanding* of $\mathcal{M}_2$ and the incomplete $\mathcal{M}_3$. Current practice is for $\mathcal{M}_4$ to be documented in an incomplete manner (e.g. some procedural manual advising parametric sensitivity and constants relating to the local physical and operational environment).

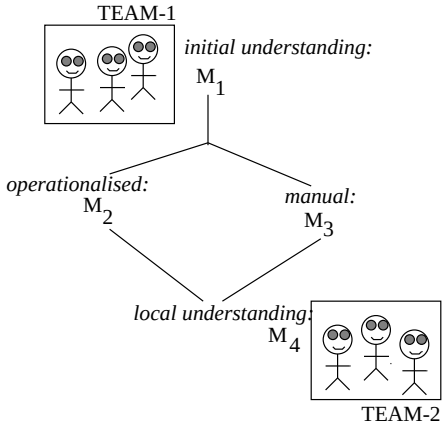

Figure 1: Commissioning Remote Models

The effort required to use $\mathcal{M}_2$ with confidence can be non-trivial. In our experience in the Air Operations Division, the effective operational use of large OR simulation codes can be time consuming. This can be even more time consuming when object code is supplied without source code or access to the author.

In essence $\mathcal{M}_2$ is a *black-box model* that Team 2 must convert (with some support from $\mathcal{M}_3$) into a *gray-box model* $\mathcal{M}_4$. Once validated, $\mathcal{M}_4$ would be used for planning, prediction, and optimisation studies. Note the emphasis on *validation*. Local conditions may invalidate $\mathcal{M}_2$. The Australian Defence Forces (ADF) use aircraft in configurations that are different to how they are used overseas. Certain decision parameters for scenario outcomes are stored in compiled numerical matrices and are inaccessible to TEAM-2 . For example, these parameters may (i) be based on experimental data from tests in other climates or (ii) contain certain tacit assumptions about aircraft operation. Prior to relying on $\mathcal{M}_2$, TEAM-2 would like to validate this model under local conditions.

Therefore, our desired solution supports:

Requirement 1 *(i) Validation of models; (ii) planning and prediction using the validated model; (iii) generating multiple options from the validated models, from which we can chose the optimum approach.*

The validation module would be particularly important. Each translation from $\mathcal{M}_{\rm i}$ to $\mathcal{M}_{\rm j}$ can introduce errors. Also, even though we imply that the members of TEAM-1 and TEAM-2 have the same model, this may not be the case. Individuals within a team may incorrectly believe they share the same view of a problem. Such a validation engine would allow individuals to check their own model as well as settling disputes between competing models; e.g. the best models have fewer problems.

While we refer to the construction of $\mathcal{M}_1$ and $\mathcal{M}_4$, these models may never be formally recorded. For example, $\mathcal{M}_4$ may only ever be tacit since it is built during the second team's informal conversations about $\mathcal{M}_2$ and $\mathcal{M}_3$. This is a major problem since if staff are transferred, they take their hard-won understanding of $\mathcal{M}_2$ with them. We need to somehow structure the development of $\mathcal{M}_4$ such that the experience gained in this process is not lost. Therefore:

Requirement 2 *An ideal model comprehension tool would be a workbench within which $\mathcal{M}_4$ can be documented.*

Note that TEAM-2 may not be able to communicate openly with TEAM-1 . The company that employs TEAM-1 may have only sold $\mathcal{M}_2$ and $\mathcal{M}_3$ as stand-alone products without any consultancy support. Nor may TEAM-2 have full access to the source code of $\mathcal{M}_2$. For example, legal or contractual obligations of TEAM-1 may prevent disclosure of portions of $\mathcal{M}_2$ to (say) non-US citizens. Such portions may only be available in binary format. Hence, $\mathcal{M}_4$ will be an under-specified "back of the envelope" sort of model containing guesses about the internal structure of $\mathcal{M}_2$. Therefore:

Requirement 3 *The representation system of $\mathcal{M}_4$ must be able to handle under-specified models.*

Such under-specified models are indeterminant at runtime. When competing influences act on the same entity, but the magnitude of these influences is under-specified, then the modeling system

must be able to create one world for each possible outcome. Note that the ability to create multiple worlds also supports the processing of "what-if" scenarios. This is a useful function for models built for exploratory purposes such as $\mathcal{M}_4$.

Assumption management will also be useful when we try to execute the guess that is $\mathcal{M}_4$. Inference over an uncertain model will generate assumptions whenever we traverse some unmeasured portion of the model. Mutually exclusive assumptions must be managed in separate worlds. Therefore:

Requirement 4 *A model comprehension tool should include assumption management and multiple world reasoning.*

3 Abduction

In this section, we discuss an inference procedure called abduction. In the next section we will argue that this procedure can satisfy the requirements of commissioning remote models.

3.1 An Introduction to Abduction

Informally, abduction is inference to the best explanation [30]. Given α , β, and the rule $R_1 : \alpha \vdash \beta$, then *deduction* is using the rule and its preconditions to make a conclusion $(\alpha \wedge R_1 \Rightarrow \beta)$; *induction* is learning R_1 after seeing numerous examples of β and α; and *abduction* is using the postcondition and the rule to assume that the precondition could be true $(\beta \wedge R_1 \Rightarrow \alpha)$ [19].

More formally, abduction is the search for assumptions $\mathcal{A}$ which, when combined with some theory $\mathcal{T}$ achieves some goal $\mathcal{G}$ without causing some contradiction [4]. That is:

EQ_1: $\mathcal{T} \cup \mathcal{A} \vdash \mathcal{G}$

EQ_2: $\mathcal{T} \cup \mathcal{A} \nvdash \perp$

While abduction can be used to generate explanation engines, we believe that EQ_1 and EQ_2 are more than just a description of "inference to the best explanation". EQ_1 and EQ_2 can be summarised as follows: make what inferences you can that are relevant to some goal, without causing any contradictions. Our basic argument is that that the proof trees used to solve EQ_1 and EQ_2 contain many of the inferences we want to make.

3.2 The HT4 Abductive Inference Engine

In order to understand abduction in more detail, we describe our `HT4` abductive inference engine [22, 24]. To execute `HT4`, the user must supply a theory $\mathcal{T}$ comprising a set of uniquely labeled statements $\mathcal{S}_x$. For example, from Figure 2, we could say that:

```
s[1] = plus_plus(a,b).
s[2] = minus_minus(b,c).
etc.
```

Figure 2 is an under-specified qualitative model [14]. In that figure:

- $X \stackrel{++}{\rightarrow} Y$ denotes that Y being *UP* or *DOWN* could be explained by X being *UP* or *DOWN* respectively;
- $X \stackrel{--}{\rightarrow} Y$ denotes that Y being *UP* or *DOWN* could be explained by X being *DOWN* or *UP* respectively.

Note that the results of this model may be uncertain; i.e. it is indeterminate. In the case of both A and B going *UP*, then we have two competing influences of C and it is indeterminate whether C goes *UP*, *DOWN*, or remains *STEADY*.

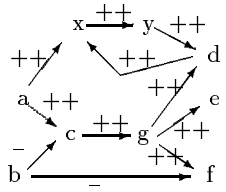

Figure 2: $\mathcal{T}_1$: An indeterminate qualitative model.

The dependency graph $\mathcal{D}$ connecting literals in $\mathcal{T}$ is an and-or graph comprising $<< \mathcal{V}^{and}, \mathcal{V}^{or} >, \mathcal{E}, \mathcal{I} >$; i.e. a set of directed edges $\mathcal{E}$ connecting vertices $\mathcal{V}$ containing invariants $\mathcal{I}$. $\mathcal{I}$ is defined in the negative; i.e. $\neg\mathcal{I}$ means that no invariant violation has occurred (e.g. if $\mathcal{I}(p, \neg p)$, then we block the simultaneous belief in a proposition *and* its negation). Each edge $\mathcal{E}_x$ and vertex $\mathcal{V}_y$ is labeled with the $\mathcal{S}_z$ that generated it.

For example, returning to the theory $\mathcal{T}$ of Figure 2, let us assume that (i) each node of that figure can take the value *UP*, *DOWN*, or *STEADY*; (ii) the conjunction of an *UP* and a *DOWN* can explain a *STEADY*; and (iii) no change can be explained in terms of a *STEADY* (i.e. a *STEADY* vertex has no children). With these assumptions,

we can expand Figure 2 into Figure 3. In that figure, $\mathcal{V}^{and}$ vertices are denoted (e.g.) *&002* while all other vertices are $\mathcal{V}^{or}$ vertices. Note that in practice, the assumptions used to convert $\mathcal{T}$ into $\mathcal{D}$ are contained in a domain-specific *model-compiler*.

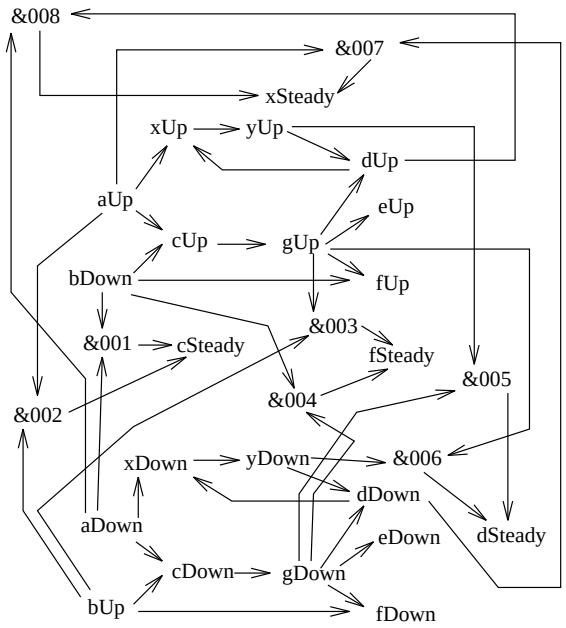

Figure 3: $\mathcal{D}$ calculated from the $\mathcal{T}$ of Figure 2

Not shown in Figure 3 are the invariants. For a qualitative domain, where nodes can have one of a finite number of mutually exclusive values, the invariants are merely all pairs of mutually exclusive assignments; e.g.:

```
i(aUp, aSteady). i(aSteady, aUp).
i(aUp, aDown).   i(aDown,   aUp).
i(bUp, bSteady). i(bSteady, bUp).
i(bUp, bDown).   i(bDown,   bUp).
etc.
```

3.2.1 Using HT4

HT4 extracts subsets of $\mathcal{E}$ which are relevant to some user-supplied $\mathcal{TASK}$. Each $\mathcal{TASK}_x$ is a triple $< \mathcal{IN}, \mathcal{OUT}, \mathcal{BEST} >$. Each task comprises some $\mathcal{OUT}$puts to be reached, given some $\mathcal{IN}$put ($\mathcal{OUT} \subseteq \mathcal{V}$ and $\mathcal{IN} \subseteq \mathcal{V}$). For the rest of this paper we will explore the example where:

```
IN  = {aUp, bUp}
OUT = {dUp, eUp, fDown}
```

$\mathcal{IN}$ can be either be a member of the known $\mathcal{FACTS}$ or a $\mathcal{DEFAULT}$ belief which we can assume if it proves convenient to do so. Typically, $\mathcal{FACTS} = \mathcal{IN} \cup \mathcal{OUT}$. If there is more than one way to achieve the $\mathcal{TASK}$, then the $\mathcal{BEST}$ operator selects the preferred way(s).

To reach a particular output $\mathcal{OUT}_z \in \mathcal{OUT}$, we must find a proof tree $\mathcal{P}_x$ using vertices $\mathcal{P}_x^{used}$ whose single leaf is $\mathcal{OUT}_z$ and whose roots are from $\mathcal{IN}$ (denoted $\mathcal{P}_x^{roots} \subseteq \mathcal{IN}$). All immediate parent vertices of all $\mathcal{V}_y^{and} \in \mathcal{P}_x^{used}$ must also appear in $\mathcal{P}_x^{used}$. One parent of all $\mathcal{V}_y^{or} \in \mathcal{P}_x^{used}$ must also appear in $\mathcal{P}_x^{used}$ *unless* $\mathcal{V}_y^{or} \in \mathcal{IN}$ (i.e. is an acceptable root of a proof). No subset of $\mathcal{P}_x^{used}$ may contradict the $\mathcal{FACTS}$; e.g. for invariants of arity 2:

$$\neg(\mathcal{V}_y \in \mathcal{P}_x^{used} \wedge \mathcal{V}_z \in \mathcal{FACTS} \wedge \mathcal{I}(\mathcal{V}_y, \mathcal{V}_z))$$

For our example, the proofs are:

```
p(1) = {aUp, xUp,   yUp,   dUp}
p(2) = {aUp, cUp,   gUp,   dUp}
p(3) = {aUp, cUp,   gUp,   eUp}
p(4) = {bUp, cDown, gDown, fDown}
p(5) = {bUp, fDown}
```

3.2.2 Assumptions

The union of the vertices used in all proofs that are not from the $\mathcal{FACTS}$ is the HT4 assumption set $\mathcal{A}_{all}$; i.e.

$$\mathcal{A}_{all} = \left(\bigcup_{\mathcal{V}_y} \left\{ \mathcal{V}_y \in \mathcal{P}_x^{used} \right\} \right) - \mathcal{FACTS}$$

The proofs in our example makes the assumptions:

```
a = {xUp, yUp, cUp, gUp, cDown, gDown}
```

The union of the subsets of $\mathcal{A}_{all}$ which violate $\mathcal{I}$ are the *controversial assumptions* $\mathcal{A}_C$:

$$\mathcal{A}_C = \bigcup_{\mathcal{V}_x} \{ \mathcal{V}_x \in \mathcal{A}_{all} \wedge \mathcal{V}_y \in \mathcal{A}_{all} \wedge \mathcal{I}(\mathcal{V}_x, \mathcal{V}_y) \}$$

The controversial assumptions of our example are:

```
ac = {cUp, gUp, cDown, gDown}
```

Within a proof $\mathcal{P}_y$ the *preconditions* for $\mathcal{V}_y \in \mathcal{P}_x^{used}$ are the transitive closure of all the parents of $\mathcal{V}_y$ in that proof. The *base controversial assumptions* ($\mathcal{A}_B$) are the controversial assumptions which have no controversial assumptions in their preconditions (i.e. are not downstream of any other controversial assumptions). The base controversial assumptions of our example are:

```
ab = {cUp, cDown}
```

3.2.3 Worlds

Maximal consistent subsets of $\mathcal{P}$ (i.e. maximal with respect to size, consistent with respect to $\mathcal{I}$) are grouped together into worlds $\mathcal{W}$ ($\mathcal{W}_i \subseteq \mathcal{E}$). Each world $\mathcal{W}_i$ contains a consistent set of beliefs that are relevant to the $\mathcal{TASK}$. The union of the vertices used in the proofs of $\mathcal{W}_i$ is denoted $\mathcal{W}_i^{used}$. In terms of separating the proofs into worlds, $\mathcal{A}_B$ are the crucial assumptions. We call the maximal consistent subsets of $\mathcal{A}_B$ the *environments* $\mathcal{ENV}$ ($\mathcal{ENV}_i \subset \mathcal{A}_B \subseteq \mathcal{A}_C \subseteq \mathcal{A}_{all} \subseteq \mathcal{V}$). The environments of our example are:

```
env(1) = {cUp}
env(2) = {cDown}
```

The union of the proofs that do not contradict $\mathcal{ENV}_i$ is the world $\mathcal{W}_i$. In order to check for non-contradiction, we compute the exclusions set $\mathcal{X}$. $\mathcal{X}_i$ are the base controversial assumptions that are inconsistent with $\mathcal{ENV}_i$. The exclusions of our example are:

```
x(1) = {cDown}
x(2) = {cUp}
```

A proof $\mathcal{P}_j$ belongs in world $\mathcal{W}_i$ if it does not use any member of $\mathcal{X}_i$ (the excluded assumptions of that world); i.e.

$$\mathcal{W}_i = \bigcup_{\mathcal{P}_j} \left\{ \mathcal{P}_j^{used} \cap \mathcal{X}_i = \emptyset \right\}$$

Note that each proof can exist in multiple worlds. The worlds of our example are:

```
w(1) = {p(1), p(2), p(3), p(5)}
w(2) = {p(1), p(4), p(5)}
```

$\mathcal{W}_1$ is shown in Figure 4 and $\mathcal{W}_2$ is shown in Figure 5.

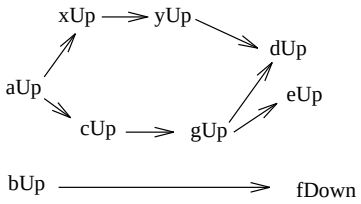

Figure 4: $\mathcal{W}_1$

For any world $\mathcal{W}_i$, $\mathcal{W}_i^{causes}$ are the members of $\mathcal{IN}$ found in $\mathcal{W}_i$ ($\mathcal{W}_i^{causes} = \mathcal{W}_i^{used} \cap \mathcal{IN}$). The achievable or *covered* goals $\mathcal{G}$ in $\mathcal{W}_i$ are

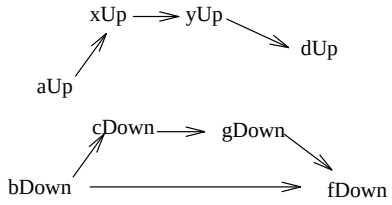

Figure 5: $\mathcal{W}_2$

the members of $\mathcal{OUT}$ found in that world ($\mathcal{W}_i^{covered} = \mathcal{W}_i^{used} \cap \mathcal{OUT}$). Continuing our example:

```
causes(w(1)) = {aUp, bUp}
causes(w(2)) = {aUp, bUp}

cover(w(1))  = {dUp, eUp, fDown}
cover(w(2))  = {dUp, fDown}
```

3.2.4 The $\mathcal{BEST}$ of all Possible Worlds

Note that, in our example, we have generated more than one world and we must now decide which world(s) we prefer. This is done using the $\mathcal{BEST}$ criteria. Numerous $\mathcal{BEST}$s can be found in the literature; e.g. the $\mathcal{BEST}$ worlds are the one which contain:

1. the most specific proofs (i.e. largest size) [8];
2. the fewest *causes* [35];
3. the greatest *cover* [22];
4. the most number of specific concepts [32];
5. the largest subset of $\mathcal{E}$ [29];
6. the largest number of *covered* outputs [28];
7. the most number of edges that model processes which are familiar to the user [31];
8. the most number of edges that have been used in prior acceptable solutions [18];

Our view is that $\mathcal{BEST}$ is domain specific; i.e. we believe that there is no universally best $\mathcal{BEST}$.

4 Abduction and Remote Model Commissioning

In this section we argue that our abductive model can be used to satisfy the requirements of remove

model comprehension; i.e. it can support validation, planning, prediction, optimisation, inference over under-specified models using assumption management and multiple-worlds reasoning.

4.1 Inference Over Under-Specified Models

`HT4` can execute over indeterminate/ under-specified models. Further, if this execution generates assumptions, then these assumptions are managed in mutually exclusive worlds ($\mathcal{W}$).

4.2 Validation

Validation tests a model's validity against external semantic criteria. Given a library of known behaviours (i.e. a set of pairs $< \mathcal{IN}, \mathcal{OUT} >$), abductive validation uses a $\mathcal{BEST}$ that favours the worlds with largest number of covered outputs (i.e. maximise $\mathcal{IN} \cap \mathcal{W}_x$) [28].

Note that this definition of *validation* corresponds to answering the following question: "can a model of X explain known behaviour of X?". We have argued elsewhere that this is the definitive test for a model [22]. Note that this is a non-naive implementation of validation since it handles certain interesting cases. In the situation where no current model explains all known behaviour, competing theories can be assessed by the extent to which they cover known behaviour. $\mathcal{M}_X$ is definitely better than $\mathcal{M}_Y$ if $\mathcal{M}_X$ explains far more behaviour than theory $\mathcal{M}_Y$.

As an example of validation-as-abduction, recall that $\mathcal{W}_1$ (see Figure 4) was generated from $\mathcal{T}_1$ when $\mathcal{IN} = \{$`aUp, bUp`$\}$ and $\mathcal{IN} = \{$`dUp, eUp,fDown`$\}$. Note that $\mathcal{W}_1^{covered}$ is all of $\mathcal{OUT}$. $\mathcal{T}_1$ is hence not invalidated since there exists a set of assumptions under which the known behaviour can be explained.

See the related work section for a discussion of other validation approaches.

4.3 Planning

Planning is the search for a set of operators that convert some current state into a goal state. We can represent planning in our abductive approach as follows:

- Represent operators as rules that convert some state to some other state;
- Augment each operator rule with:
  - a unique label $\mathcal{S}_1$, $\mathcal{S}_2$, etc. When $\mathcal{D}$ is generated, each edge will now include the name(s) of the operator(s) that generated it.
  - A cost figure representing the effort required to apply this operator rule.
- Set $\mathcal{IN}$ to the current state, $\mathcal{OUT}$ to the goal state, and $\mathcal{FACTS} = \mathcal{IN} \cup \mathcal{OUT}$.
- Set $\mathcal{BEST}_{PLANNING}$ to favour the world(s) with the least cost. The cost of a world is the maximum of the "proof cost" of each member of $\mathcal{OUT}$. The "proof cost" of $\mathcal{OUT}_i$ is the minimum cost of the proofs that cover $\mathcal{OUT}_i$.
- Run HT4. Collect and cache the generated worlds.
- For each $\mathcal{BEST}$ world, collect all the names of the operators used in the edges of that world. These operators will be in a tree structure that reflects the structure of the $\mathcal{BEST}$ worlds. Report these trees as the output plans.

A related task to planning is *monitoring*; i.e. the process of checking that the current plan(s) are still possible. The worlds generated by the above planner will contain some assumptions. As new information comes to light, some of these assumptions will prove to be invalid. Delete those worlds from the set of possible plans. The remaining plans represent the space of possible ways to achieve the desired goals in the current situation. If all plans are rejected, then run HT4 again with all the available data.

4.4 Optimisation

We view optimisation as planning with a $\mathcal{BEST}$ operator that favours the lower cost world(s).

4.5 Prediction

Prediction is the process of seeing what will follow from some events $\mathcal{IN}$. This can be implemented in HT4 by making $\mathcal{OUT} \subseteq \mathcal{V} - \mathcal{IN}$; i.e. find all the non-input vertices we can reach from the inputs. For prediction, $\mathcal{FACTS}$ should not be $\mathcal{IN} \cup \mathcal{OUT}$ since this will be the entire dependency graph. If the $\mathcal{IN}$ is certain, then $\mathcal{FACTS} = \mathcal{IN}$ (i.e. only the inputs cannot be contradicted). This is a non-naive implementation of prediction since mutually exclusive predictions (the *covered* elements of $\mathcal{OUT}$) will be found in different worlds.

Note that in the special case where:

- $\mathcal{IN}$ are all root vertices in $\mathcal{D}$.
- $\mathcal{FACTS} = \emptyset$
- $\mathcal{OUT} = \mathcal{V} - \mathcal{IN}$

then our abductive system will compute ATMS-style [2] *total envisionments*; i.e. all possible consistent worlds that are extractable from the theory. A more efficient case is that $\mathcal{IN}$ is smaller than all the roots of the graph and some *interesting subset* of the vertices have been identified as possible reportable outputs (i.e. $\mathcal{OUT} \subset \mathcal{V} - \mathcal{IN}$).

5 Related Work

5.1 General Abductive Reasoning

Note that this work is part of Menzies' *abductive reasoning project*. Menzies argues that abduction provides a comprehensive picture of declarative knowledge-based systems (KBS) inference such as prediction, classification, explanation, quantitative reasoning, planning, monitoring, set-covering diagnosis, consistency-based diagnosis, validation, and verification [24]. Menzies also believes that abduction is a useful framework for intelligent decision support systems [23], diagrammatic reasoning [27], single-user knowledge acquisition, and multiple-expert knowledge acquisition [25]. Further, abduction could model certain interesting features of human cognition including the situated nature of cognition [26]. Others argue elsewhere that abduction is also a framework for natural-language processing [29], design [33], visual pattern recognition [34], analogical reasoning [5], financial reasoning [11], machine learning [12] and case-based reasoning [18].

5.2 Qualitative Reasoning

We are not the first researchers to argue that intuitions about models can be represented in an indeterminant, under-specified modeling framework. The qualitative reasoning (QR) community focuses on the processing of systems called qualitative differential equations (QDE) which are:

- Piece-wise well-approximated by low-order linear equations or by first-order non-linear differential equations;
- Whose numeric values are replaced by one of three qualitative states: up, down, or steady [14].

Since QDEs are under-specified, they can be written faster than their fully-specified quantitative counterparts. Hence, they have been proposed as a tool for recording intuitions. However, we do not suggest using QR for building $\mathcal{M}_4$. A QDE is still a mathematical equation and mathematics is a poor model for causality. Ohms's Law ($R = \frac{V}{I}$) relates resistance R to current I and voltage V. Note that changes in voltage and current do not cause changes in resistance, even though the mathematical formulae suggests this is possible. Resistors cannot be manufactured to a certain specification merely by attaching wire to some rig and altering the voltage and current over the rig. Ignoring the effects of temperature and high-voltage breakdown, resistance is an invariant built into the physics of a wire. Hidden within Ohm's Law are rules regarding the direction of causality between voltage, current, and resistance. Such rules are invisible to a mathematical formulation.

Causality was a central concern in QR till the mid-1980s [1] and it is a construct we wish to support in $\mathcal{M}_4$.

> ... It is clear that causality plays an essential role in our understanding of the world ... to understand a situation means to have a causal explanation of the situation [13].

Initially two qualitative ontologies were proposed: DeKleer & Brown's 1984 `CONFLUENCES` system [3] and Forbus's 1984 qualitative process theory (`QPT`) [6]. Later work in 1986 recognised that both these systems processed QDEs and a special theorem prover, `QSIM`, was written by Kuipers especially for QDEs [16]. Compilers were written to covert QPT models into `QSIM`. Note that the evolution of QR worked down from complex representations (`QPT` to `QSIM` to simpler graph-theoretic approach). Kuipers himself now believes that underlying `QSIM` was a more basic inference process: Mackworth's arc consistency algorithm [17, 21] which is based around a simple graph-theoretic framework (though Mackworth's work can be expressed in a logic framework [20]). Note the evolution of the QR work from complex representations (e.g. `QPT`) to simpler graph-theoretic approaches.

After an inclusive public debate between public debate in 1986 between the `CONFLUENCES` approach and a rival theory [15], the term "causality" was avoided by many QR researchers. Forbus's 1992 retrospective on causality and the 1980s QR research is primarily negative:

> ... In terms of violating human intuitions, each system of qualitative physics fails in some way to handle causality properly. Like (QPT) theory, deKleer and Brown's CONFLUENCES theory... fails to distinguish between equations representing causal versus non-causal laws. Kuipers QSIM contains no account of causality at all [7].

In summary, the 1980s experiment with using QDEs to model causal intuitions failed. We prefer our directed-graph approach since this at least gives us a strong sense of inference direction.

5.3 Truth-Maintenance

Here we have explored a graph-theoretic framework for mon-monotonic logic. An alternative approach is the logic-based approach pioneered by DeKleer's assumption-based truth maintenance system [2]). In his ATMS framework, an inference engine passes justifications to a database which, as a side-effect, would incrementally modify sets of consistent literals storing the root assumptions of different worlds. Forbus & DeKleer proposed this as a general inference procedure for knowledge-based *problem solvers* [7]. We have a similar intuition. However, unlike the ATMS, Menzies does not divide the inference process between an inference engine and an ATMS database. Rather, Menzies argues that a thorough declarative reading of common KBS can be mapped into the world-generation process described in section 3.

In later work, DeKleer linked his approach with Reiter's default logic [36]. An *extension* E of a default theory is a set of literals from the theory which do not violate a set of invariants (called the *justifications*). All formulae whose preconditions (called *prerequisites*) are satisfied by E and whose invariants are consistent with E are also in E. An HT4 world differs from a default logic extension in that the latter is closed under deduction and contains all literals that are consistent with E. HT4's worlds only contain *relevant literals*; i.e. only the literals that are on proofs leading to known *outputs*. HT4 regards full extension generation as wasted computation.

At its core, the ATMS builds the dependency network between literals in a knowledge base and explores this network. Invariant knowledge is maintained such that mutually incompatible subsets of this dependency network are avoided. Such a representation can be used for validation. Thus dependency network can be used to determine inputs that will exercise all branches of the knowledge base. This is the core of the validation systems by Ginsberg [9] and Zlatereva [37]. However, note that once an input suite is inferred, an expert still has to decide what are the appropriate outputs for those inputs. In the case of vague models (where there is no definitive oracle), the correct outputs are unknown. The remove model comprehension problem is a model construction activity and the constructed model is less a picture of a domain than a device for exploring that domain. Asking a member of TEAM-2 for the correct output across an uncertain knowledge base that is being built to explore an area of uncertainty seems, in our view, inappropriate.

We note that HT4 has much in common with the Ginsberg/Zlatereva approaches. All these systems are based on a TMS variant. More precisely, all these systems use some style of non-monotonic logic. We prefer our approach since we believe that our graph-theoretic approach is a more minimal framework than the logic-based style of Ginsberg and Zlatereva. Initially, we found that logic-based approaches to TMS were very complicated. After mapping the TMS process down to a graph-theoretic process, we found the TMS process more approachable and simpler to understand. HT4 could be used in a Ginsberg/Zlatereva style. If we use HT4 to generate all possible worlds, then the roots of those worlds will be test suite inputs that will exercise all branches of the KB. We hesitate to suggest this as standard practice, however, since the generation of all worlds is even slower than HT4 usual practice of generating worlds for the relevant literals (see the discussion of complexity in [22]).

6 Conclusion

There is a pressing need for some methodology to structure the creation and recording of the understanding of remote models; i.e. the generation of $\mathcal{M}_4$. In terms of the computational requirements of $\mathcal{M}_4$, an appropriate modeling language must support:

- Validation;
- Planning;
- Prediction;
- Optimisation;
- Inference over under-specified models;
- Assumption management and multiple-worlds reasoning.

In this paper, we have argued that abduction is a promising approach since it satisfies these criteria.

We have also noted similarities with of the remote model commissioning problem to the QR and TMS literature. While both the QR and TMS literature supply us with insights into our problem, we find the TMS literature more relevant than QR.

Potentially fruitful avenues to explore include:

- A *proof-of-concept study* in which a gray-box model is built using our abductive framework from a readily-available dynamic simulation black-box computer game. The advantage of using such a game is that, unlike OR models, it is small enough to explain in a paper. Furthermore, the game would be available to other researchers.
- *Situation awareness*: When faced with a novel domain, people learn models. There are many styles of learning. We conjecture that people learn models to the depth required for some particular purpose. The resulting models are hence approximate. One way to characterise our current proposal is the construction of approximate models gained through incomplete experience of the entity being modeled. We speculate that this represents a form of situation awareness [10].

References

[1] E. Coiera. The Qualitative Representation of Physical Systems. *The Knowledge Engineering Review*, 7:1–23, 1 1992.

[2] J. DeKleer. An Assumption-Based TMS. *Artificial Intelligence*, 28:163–196, 1986.

[3] J. DeKleer and J.S. Brown. A Qualitative Physics Based on Confluences. *Artificial Intelligence*, 25:7–83, 1984.

[4] K. Eshghi. A Tractable Class of Abductive Problems. In *IJCAI '93*, volume 1, pages 3–8, 1993.

[5] B Falkenhainer. Abduction as Similarity-Driven Explanation. In P. O'Rourke, editor, *Working Notes of the 1990 Spring Symposium on Automated Abduction*, pages 135–139, 1990.

[6] K. Forbus. Qualitative Process Theory. *Artificial Intelligence*, 24:85–168, 1984.

[7] K. Forbus. Pushing the Edge of the (QP) Envelope. In B. Faltings and P. Struss, editors, *Recent Advances in Qualitative Physics*, pages 245–261. The MIT Press, 1992.

[8] C.L. Forgy. RETE: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. *Artificial Intelligence*, pages 17–37, 19 1982.

[9] A. Ginsberg. A new Approach to Checking Knowledge Bases for Inconsistentcy and Redundancy. In *Proc. 3rd Annual Expert Systems in Government Conference*, pages 102–111, 1987.

[10] J. Gledhill and S. Goss. Modeling Situation Awareness. In preparation.

[11] W. Hamscher. Explaining Unexpected Financial Results. In P. O'Rourke, editor, *AAAI Spring Symposium on Automated Abduction*, pages 96–100, 1990.

[12] K. Hirata. A Classification of Abduction: Abduction for Logic Programming. In *Proceedings of the Fourteenth International Machine Learning Workshop, ML-14*, page 16, 1994. Also in *Machine Intelligence* 14 (to appear).

[13] Y. Iwasaki. Causal Ordering in a Mixed Structure. In *Proceedings of AAAI '88*, pages 313–318, 1988.

[14] Y. Iwasaki. Qualitative Physics. In P.R. Cohen A. Barr and E.A. Feigenbaum, editors, *The Handbook of Artificial Intelligence*, volume 4, pages 323–413. Addison Wesley, 1989.

[15] Y. Iwasaki and H.A. Simon. Causality in Device Behaviour. *Artificial Intelligence*, 29:3–31, 1986.

[16] B. Kuipers. Qualitative Simulation. *Artificial Intelligence*, 29:229–338, 1986.

[17] B.J. Kuipers. Reasoning with Qualitative Models. *Artificial Intelligence*, 59:125–132, 1993.

[18] D.B. Leake. Focusing Construction and Selection of Abductive Hypotheses. In *IJCAI '93*, pages 24–29, 1993.

[19] H. Levesque. A Knowledge-Level Account of Abduction (Preliminary Version). In *IJCAI '89*, volume 2, pages 1061–1067, 1989.

[20] A. Mackworth. The Logic of Constraint Satisfaction. *Artificial Intelligence*, 58:3–20, 1992.

[21] A.K. Mackworth. Consistency in Networks of Relations. *Artificial Intelligence*, 8:99–118, 1977.

[22] T. J. Menzies and P. Compton. The (Extensive) Implications of Evaluation on the Development of Knowledge-Based Systems. In *Proceedings of the 9th AAAI-Sponsored Banff Knowledge Acquisition for Knowledge Based Systems*, 1995.

[23] T.J. Menzies. Applications of Abduction #1: Intelligent Decision Support Systems. Technical Report TR95-16, Department of Software Development, Monash University, 1995.

[24] T.J. Menzies. Applications of Abduction #2: Knowledge Level Modeling. Technical Report TR95-23, Department of Software Development, Monash University, 1995.

[25] T.J. Menzies. *Principles for Generalised Testing of Knowledge Bases*. PhD thesis, University of New South Wales, 1995.

[26] T.J. Menzies. Situated Semantics is a Side-Effect of the Computational Complexity of Abduction. In *Australian Cognitive Science Society, 3rd Conference*, 1995.

[27] T.J. Menzies and P. Compton. *A Precise Semantics for Vague Diagrams*, pages 149–156. World Scientific, 1994.

[28] T.J. Menzies and W. Gambetta. Exhaustive Abduction: A Practical Model Validation Tool. In *ECAI '94 Workshop on Validation of Knowledge-Based Systems*, 1994.

[29] H.T. Ng and R.J. Mooney. The Role of Coherence in Constructing and Evaluating Abductive Explanations. In *Working Notes of the 1990 Spring Symposium on Automated Abduction*, volume TR 90-32, pages 13–17, 1990.

[30] P. O'Rourke. Working Notes of the 1990 Spring Symposium on Automated Abduction. Technical Report 90-32, University of California, Irvine, CA., 1990. September 27, 1990.

[31] C.L. Paris. The Use of Explicit User Models in a Generation System for Tailoring Answers to the User's Level of Expertise. In A. Kobsa and W. Wahlster, editors, *User Models in Dialog Systems*, pages 200–232. Springer-Verlag, 1989.

[32] D. Poole. On the Comparison of Theories: Preferring the Most Specific Explanation. In *IJCAI '85*, pages 144–147, 1985.

[33] D. Poole. Hypo-Deductive Reasoning for Abduction, Default Reasoning, and Design. In P. O'Rourke, editor, *Working Notes of the 1990 Spring Symposium on Automated Abduction.*, volume TR 90-32, pages 106–110, 1990.

[34] D. Poole. A Methodology for Using a Default and Abductive Reasoning System. *International Journal of Intelligent Systems*, 5:521–548, 1990.

[35] J. Reggia, D.S. Nau, and P.Y Wang. Diagnostic Expert Systems Based on a Set Covering Model. *Int. J. of Man-Machine Studies*, 19(5):437–460, 1983.

[36] R. Reiter. A Logic for Default Reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[37] N. Zlatereva. Truth Mainteance Systems and Their Application for Verifying Expert System Knowledge Bases. *Artificial Intelligence Review*, 6, 1992.