View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Elsevier - Publisher Connector

Artificial Intelligence 175 (2011) 49–78

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

$\mathcal{M}$odular-$\mathcal{E}$ and the role of elaboration tolerance in solving the qualification problem

Antonis Kakas[a], Loizos Michael[b], Rob Miller[c,*]

[a] *University of Cyprus, P.O. Box 20537, CY-1678, Cyprus*
[b] *Harvard University, Cambridge, MA 02138, USA*
[c] *University College London, London WC1E 6BT, UK*

ARTICLE INFO

Article history:
Available online 3 April 2010

Keywords:
Reasoning about actions
Elaboration tolerance
Qualification problem
Default reasoning

ABSTRACT

We describe $\mathcal{M}$odular-$\mathcal{E}$ ($\mathcal{ME}$), a specialized, model-theoretic logic for reasoning about actions. $\mathcal{ME}$ is able to represent non-deterministic domains involving concurrency, static laws (constraints), indirect effects (ramifications), and narrative information in the form of action occurrences and observations along a time line. We give formal results which characterize $\mathcal{ME}$'s high degree of modularity and elaboration tolerance, and show how these properties help to separate out, and provide principled solutions to, different aspects of the qualification problem. In particular, we identify the *endogenous* qualification problem as the problem of properly accounting for highly distributed, and potentially conflicting, causal knowledge when reasoning about the effects of actions. We show how a comprehensive solution to the endogenous qualification problem helps simplify the *exogenous* qualification problem — the problem of reconciling conflicts between predictions about what should be true at particular times and actual observations. More precisely, we describe how $\mathcal{ME}$ is able to use straightforward default reasoning techniques to solve the exogenous qualification problem largely because its robust treatments of the frame, ramification and endogenous qualification problems combine into a particular characteristic of elaboration tolerance that we formally encapsulate as a notion of "free will".

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

For half a century the seminal work of John McCarthy has been instrumental in identifying the major areas, issues and directions for progress in Knowledge Representation and logic-based A.I. In particular, research in reasoning about action and change (RAC), originally initiated by McCarthy and collaborators, continues to be a central topic in this field. "Classical" RAC formalisms such as the Situation, Event and Fluent Calculi [26,28,32,36], as well as their more semantically specialized counterparts such as TAL, Dynamic Logic and the Languages $\mathcal{A}$, $\mathcal{E}$ and $\mathcal{C}+$ [5,8,10,12,15], are increasingly proving their worth not only in traditionally related areas such as planning and cognitive robotics, but also in fields as diverse as computational biology, the semantic web and software engineering [27,33,38].

Irrespective of the particular logic or formalism being developed, certain issues have for a long time been considered fundamental in RAC. Foremost among these is of course the *frame problem* — how to succinctly and flexibly represent and reason about the non-effects of actions, avoiding such pitfalls as the infamous Hanks and McDermott problem [11]. More recently there has been much focus on the *ramification problem* — how to accommodate knowledge and inferences about the

* Corresponding author.
E-mail addresses: antonis@ucy.ac.cy (A. Kakas), loizos@eecs.harvard.edu (L. Michael), rsm@ucl.ac.uk (R. Miller).
URLs: http://www2.cs.ucy.ac.cy/~antonis/ (A. Kakas), http://www.eecs.harvard.edu/~loizos/ (L. Michael), http://www.ucl.ac.uk/slais/rob-miller/ (R. Miller).

0004-3702/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.artint.2010.04.008

indirect and knock-on effects of actions. Largely as a result of this, there has been a growing consensus that RAC formalisms need to explicitly incorporate some notion of *causality*.

However, a third fundamental issue identified by McCarthy, although generally perceived as important, has received relatively little attention to date. This is the *qualification problem* [22,37], which, as we will see, relates to various difficulties in adequately qualifying statements either about the executability or about the effects (both direct and indirect) of actions. One can in fact naturally cast the qualification problem in terms of McCarthy's "Appearance and Reality" phenomenon [25]. The reality in a domain, captured in the context of RAC formalisms by effect laws, is not directly observable, and these laws are of necessity incomplete. This incompleteness leads in some cases to inconsistency, rendering the reasoning process incapable of drawing any conclusion. The qualification problem can be viewed as the problem of how to let appearances in a domain (captured by observations, etc.) qualify our beliefs about the reality, so that the inconsistency is lifted, and meaningful conclusions may be drawn.

The qualification problem is the main focus of this paper. But, just as the ramification problem has proved to be intimately related to the frame problem, the qualification problem is inextricably linked to both the frame and ramification problems, so that development of a solution to one inevitably leads to a re-examination of solutions to the other two. Hence, as a necessary preliminary to our main aim, we also present an exceptionally robust and complete solution to the frame and ramification problems combined. We also illustrate how McCarthy's notion of *elaboration tolerance* [24], long considered desirable in RAC frameworks, plays a key role in the combined solution to all three problems.

1.1. Some terminology

In order to describe the various aspects of the qualification problem, and our approach to solving them, it is convenient to first summarize some of the terminology commonly used in this research area. Time-varying properties of the world, i.e., those that can potentially be affected by actions, are usually referred to as *fluents*. The terms *action* and *event* are often used synonymously. A *state* is an assignment of a truth value to each fluent. *Narrative-based* formalisms (such as TAL, the Event Calculus and the Language $\mathcal{E}$) employ a structure of *time-points* (or time *intervals*) that is defined independently from actions and fluents. One way or another, explicitly or implicitly, these frameworks associate a unique state with each time-point in each model. In such formalisms, a fluent *holds* at a time-point if it is true in the associated state. On the other hand, non-narrative-based formalisms (such as the Situation and Fluent Calculi and the Language $\mathcal{A}$) avoid the use of an independent time structure by reasoning directly about sequences of actions. Such formalisms, one way or another, explicitly or implicitly, associate a unique state with each "allowed" action sequence in each model. The term *situation* is sometimes used to refer to a sequence of actions and sometimes used to refer to the state associated with a sequence of actions.

There is also a common terminology as regards different types of statements or sentences within RAC formalisms. The terms *effect law*, *effect axiom*, and *causal law* are sometimes used to characterize a sentence describing a specific causal effect. For example, to describe the effect of turning the ignition key of a car we may write (in the syntax of the Situation Calculus, the Event Calculus and the Language $\mathcal{C}+$ respectively):

$$Holds(EngineRunning, Do(TurnKey, s)) \leftarrow Holds(BatteryCharged, s)$$

$$Initiates(TurnKey, EngineRunning, t) \leftarrow HoldsAt(BatteryCharged, t)$$

$$TurnKey\ \textbf{causes}\ EngineRunning\ \textbf{if}\ BatteryCharged$$

The qualification *BatteryCharged* in this effect law is an example of a *fluent precondition* — although it is not necessary for *BatteryCharged* to hold in order for the *TurnKey* action to be successfully executed, *BatteryCharged* is necessary for the *TurnKey* action to have this particular effect on *EngineRunning*. In contrast to effect laws, *executability laws* specify conditions under which an action can or cannot be successfully executed, e.g. (Situation Calculus, Event Calculus and Language $\mathcal{C}+$ syntax respectively):

$$Poss(TurnKey, s) \equiv Holds(HasKey, s)$$

$$Impossible(TurnKey, t) \leftarrow \neg HoldsAt(HasKey, t)$$

$$\textbf{nonexecutable}\ TurnKey\ \textbf{if}\ \neg HasKey$$

In this statement the qualification *HasKey* is an *executability* or *action precondition*, and indeed such sentences are also sometimes referred to as *action precondition axioms*.

Most formalisms also allow various action-independent relationships between fluents or sets of fluents to be described. *Domain constraints* or *state constraints* are simply constraints on the set of allowed states that have to be taken into account one way or another when constructing state transition rules from the domain description as a whole. For example, to describe that broken cars' engines cannot run we may write (in the Situation Calculus):

$$\neg[Holds(EngineRunning, s) \wedge Holds(EngineBroken, s)]$$

Exactly how a formalism should take such statements into account is of course debatable, a central issue being about whether they should result in more or in less change. (In this particular example one might argue that an action that causes *EngineBroken* should therefore also cause $\neg$*EngineRunning*. But one is unlikely to argue that an action that causes *EngineRunning* should therefore also cause $\neg$*EngineBroken*, but rather that $\neg$*EngineBroken* should act as an extra fluent precondition for effect laws that initiate *EngineRunning*.) In many formalisms the ambiguity of such statements can be avoided by re-expressing them as "unidirectional" *ramification laws* or *indirect effect laws*. For example, in the Language $\mathcal{E}$ we may write:

$$\neg \mathit{EngineRunning}\ \texttt{whenever}\ \{\mathit{EngineBroken}\}$$

to express that causing *EngineBroken* has the knock-on effect of causing $\neg$*EngineRunning*.

Observation statements (or simply *observations*) are assertions that particular fluents hold or do not hold at particular time-points or in particular states. For example, in the narrative-based Event Calculus we may assert that a car's engine was observed to be running at time 3 by $\mathit{HoldsAt}(\mathit{EngineRunning}, 3)$. The nearest we can get to such assertions in non-narrative formalisms is via sentences such as $\mathit{Holds}(\mathit{EngineRunning}, \mathit{Do}(\mathit{TurnKey}, S0))$ (Situation/Fluent Calculus syntax). If regarded as an "observation", the implication of this statement is (roughly) that the world was at some point in the past in state $S0$, that in that state a *TurnKey* action was performed, and that immediately afterwards *EngineRunning* was seen to be true.

The last type of statement it is useful to identify here is an (*action* or *event*) *occurrence statement*. Occurrence statements assert that a particular action was performed, or at least attempted, at a specific time-point (or over a specific time-interval), as e.g. in the Language $\mathcal{E}$ proposition *TurnKey* `happens-at` 2. Obviously such assertions are possible only in narrative-based formalisms. In what follows, we refer to the collection of observations and occurrence statements within a theory as its *narrative*.

1.2. Aspects of the qualification problem

McCarthy [23] describes the qualification problem as follows: "In order to fully represent the conditions for the successful performance of an action, an impractical and implausible number of qualifications would have to be included in the sentences expressing them". In this section we disambiguate and provide some terminology for various aspects of this problem. The analysis offered here is inspired by, and borrows some vocabulary from, that of Thielscher [37].

We first note that McCarthy's assertion above applies equally to fluent and action preconditions. One consequence of the need to describe a large number of conditions for a successful performance or effect of an action is the need to distribute such information across several sentences. For example, we have already illustrated in the previous section how fluent preconditions may be contained both explicitly in effect laws and implicitly in domain constraints (turning the ignition key causes a car's engine to run if the battery is charged, but broken engines will not run). Therefore when computing or reasoning about the effects or success of a particular action attempt, this distributed information about relevant qualifications will have to be identified and combined in a principled way. We refer to this as the *endogenous qualification problem* — "endogenous" since it concerns the interpretation of information already contained within the given representation of the domain. The endogenous qualification problem is closely related to the issue of *elaboration tolerance* [21,24], in that a good solution to it is more likely to allow new information about qualifications to be added to a theory as new sentences, without the need to otherwise alter the existing theory. In turn, elaboration tolerance is strongly linked with the need to have a *modular* semantics for RAC frameworks that properly separates different aspects of the domain knowledge, as argued e.g. in [13].[1]

But McCarthy's description of the qualification problem principally relates to a more fundamental issue about modelling the real world. Any model or representation (of any kind — logical, mathematical, analogical, etc.) is necessarily incomplete or approximate in some respects, arguably by the very definition of a model or representation. In this sense it is impossible to "fully represent" anything. In particular, if a logic-based representation is used to make predictions about the real world, those predictions will occasionally be contradicted by direct experience. More particularly still, if a logic-based agent makes predictions about the effects of an (attempted) action, subsequent observations will occasionally reveal that the attempt failed to produce some or all of the expected effects, because of factors (qualifications) outside the scope of the representation. No amount of refining the representation will stop this from happening, albeit only very occasionally. We refer to the *exogenous qualification problem* as the problem of engineering a representation flexible enough to be able to recover from occasional mismatches between predictions about actions' effects and actual observations, yet robust enough to be useful as a predictive tool. "Exogenous" since it partially concerns factors outside the scope of, or even the language of, the representation. Note that, since the exogenous qualification problem involves the notions of observation and action occurrence, its solution will require a narrative-based formalism, or at least a formalism enhanced with some narrative reasoning capabilities.

McCarthy and others (notably Thielscher) have illustrated that default and non-monotonic reasoning has a significant role to play in dealing with aspects of the qualification problem [23,37]. Indeed, a "weak" version of the exogenous qualification

[1] In this paper, Herzig and Varzinczak show how, in "non-modularized" theories, pairs of action laws can sometimes give rise to unintended domain constraints, and how in turn this can prevent the domain from being elaborated with conflicting domain constraints.

problem is the problem of representing that some fluents (or their negations) always hold under "normal" circumstances, so that they should be assumed by default when inferring the effects of actions. However, two points should be noted. First, a solution to this problem will not help when abnormal factors unrepresentable by any combination of the fluents in the language interfere with an (attempted) action. Second, if a fluent has a default truth value, this should manifest itself everywhere in the reasoning process, and not just when the fluent appears as a precondition in an effect or executability law. For example, if we are told only that car engines are not normally broken, we should be able to infer by default that a given car engine is not broken at a given time, irrespective of whether that car has had its ignition key turned. So the problem of dealing with default fluent values is separate from, although strongly related to, the exogenous qualification problem.

How do the endogenous and exogenous qualification problems interact? In this paper, we argue that "endogenous" explanations for particular observations are preferable to "exogenous" ones, and that a "complete" solution to the endogenous qualification problem is necessary for a "clean" solution to the exogenous problem. For example, suppose a simple representation includes only the effect law "*TurnKey* `causes` *EngineRunning*" and the domain constraint $\neg(\mathit{Broken} \wedge \mathit{EngineRunning})$, but that the semantics fails to recognize $\neg\mathit{Broken}$ as an implicit precondition for the effect law. A narrative in which a *TurnKey* action is followed by an observation of $\neg\mathit{EngineRunning}$ then introduces a potential inconsistency. The formalism may then be tempted to resolve this apparent conflict of information by prematurely appealing to some general solution to the exogenous qualification problem — offering a mysterious, unrepresentable explanation for the "failure" of *TurnKey* instead of the obvious possibility of *Broken*.

A full solution to the exogenous qualification problem necessitates a view of occurrence statements as identifying only *attempts* to execute actions (even though we may choose to build into a formalism a default principle that such attempts can be assumed to result in actual action executions unless there is evidence to the contrary). If we do not view occurrence statements in this way, we overly limit the options as regards recovery from unexpected observations. For example, suppose we have two effect laws stating that *PushButton* causes *IndicatorLightOn* and *PushButton* causes *LifeSupportOn*. Suppose also that we have a narrative which records an occurrence of *PushButton* but also a subsequent observation $\neg\mathit{IndicatorLightOn}$. Clearly we should not only consider the possibility that some unrepresented fluent precondition prevented the triggering of the *IndicatorLightOn* effect law, and hence confidently maintain the belief that *LifeSupportOn*. We also need to entertain the possibility that an unrepresented action precondition stopped *PushButton* from executing correctly (i.e., that this instance of *PushButton* represents a "failed" execution attempt), and therefore view both its effects with suspicion.

The treatment of occurrence statements as execution attempts rather than certainties has another important benefit, concerning the *modularity* of action theories. Recent commentary about the need for modularity in RAC formalisms has been concerned with non-narrative aspects of theories — for example, the need to avoid generation of implicit domain constraints from collections of effect laws (see e.g. [13]). In this paper we illustrate the importance of adhering to a complimentary, temporally directed, principle of modularity specifically concerning narratives — action occurrences in the future should not implicitly force the past to necessarily have a certain property. As we will see, a model-theoretic analysis of this requirement shows it to be equivalent to the principle that an agent can attempt to execute an action in any circumstance (albeit entirely ineffectively if the action's executability conditions are not satisfied). We therefore refer to this principle as the *free will property*. This "free will" is not only crucial to the decoupling of the exogenous and endogenous qualification problems, but also helps avoid other problems such as anomalous plans in abductive planning systems. Conceptually, it helps to underline the ontological difference between actions and fluents. It makes good engineering sense as well — in cognitive robotics terms, for example, it mirrors a proper differentiation between effectors, sensors and environment (in which actions are internal signals from a robot's reasoning engine to its effectors, whereas observations are signals from the environment to the reasoning engine via sensors).

1.3. Structure of the paper

The remainder of this paper is organized as follows. In Section 2 we discuss some shortcomings of existing RAC formalisms with respect to the above analysis of the qualification problem and related issues. In Section 3 we introduce the action description language $\mathcal{M}$odular-$\mathcal{E}$ ($\mathcal{ME}$), engineered to overcome the limitations we have identified. Section 3.1 describes the syntax of the language and gives an informal insight into its model-theoretic semantics via a series of examples, and Sections 3.2 and 3.3 give a formal definition of the semantics. In Section 4 we prove that $\mathcal{ME}$ does indeed have the necessary characteristics (such as the "free will" property) that we have identified as important. In Section 5 we show how $\mathcal{ME}$ can be extended to deal with default fluent values, and in Section 6 we show how this extension, together with the results from Section 4, lays the groundwork for a principled solution to the exogenous qualification problem. We conclude in Section 7 with a summary of our results, some implications for RAC formalisms in general, and some directions for future study.

2. Some shortcomings of existing approaches

2.1. Incomplete solutions to the frame problem

To our knowledge, the most complete recent discussion of the qualification problem in the sense of Section 1.2 is given by Thielscher [37]. He first demonstrates how a failure to take causality into proper account when engineering a non-monotonic solution to the qualification problem can give rise to anomalous models. He then proposes a framework based on monotonic Fluent Calculus embedded in a straightforward default theory to overcome these problems. The resulting formalism is able to cope with a broad range of phenomena relating to the qualification problem, for example, covering both "accidents" (one-off failures of actions) and persistent failures, and offering the possibility of prioritizing between alternative explanations of action failure.

However, Thielscher's approach has a limitation that renders it inappropriate as a candidate solution to the narrative-centered exogenous qualification problem as described in Section 1.2. This is that the solution to the frame problem that it employs does not cover the case of failed actions (or action "attempts", as we have argued for above). This can be illustrated with a variation of the "life support" domain mentioned in the previous section. In the Fluent Calculus, the basic knowledge about the effect that the *PushButton* action has on the fluents *IndicatorLightOn* and *LifeSupportOn* can be described by the state update axiom (FC1) below. We will suppose that *PushButton* is unconditionally executable (FC2), and to motivate the need to push the button, we will also assert (FC3) that the patient is very ill in the initial situation $S0$:

$$\begin{aligned}&\mathit{Poss}(\mathit{PushButton}, s) \rightarrow\\&\quad [\mathit{State}(\mathit{Do}(\mathit{PushButton}, s)) =\\&\qquad \mathit{State}(s) \circ \mathit{IndicatorLightOn} \circ \mathit{LifeSupportOn}]\end{aligned} \tag{FC1}$$

$$\mathit{Poss}(\mathit{PushButton}, s) \tag{FC2}$$

$$\mathit{Holds}(\mathit{PatientVeryIll}, S0) \tag{FC3}$$

Axioms (FC1)–(FC3), together with the standard fluent calculus domain independent theory, correctly entail the following three facts about the situation after a *PushButton* action:

$$\mathit{Holds}(\mathit{PatientVeryIll}, \mathit{Do}(\mathit{PushButton}, S0)) \tag{Conclusion 1}$$

$$\mathit{Holds}(\mathit{IndicatorLightOn}, \mathit{Do}(\mathit{PushButton}, S0)) \tag{Conclusion 2}$$

$$\mathit{Holds}(\mathit{LifeSupportOn}, \mathit{Do}(\mathit{PushButton}, S0)) \tag{Conclusion 3}$$

As discussed above, the exogenous qualification problem is the problem of how to modify theories such as this so that they can accommodate unexpected "observations" such as $\neg \mathit{Holds}(\mathit{IndicatorLightOn}, \mathit{Do}(\mathit{PushButton}, S0))$. Thielscher accomplishes this by augmenting the fluent calculus with abnormality fluents such as $\mathit{Ab}(\mathit{IndicatorLightable}, \mathit{BulbBroken})$, where the second argument is a "cause" for the abnormality indicated by the first argument. For convenience he also defines the macro $\mathsf{Ab}(x, s)$ as $(\exists y)\mathit{Holds}(\mathit{Ab}(x, y), \mathit{State}(s))$. For each abnormality fluent $\mathit{Ab}(X, Y)$, a default rule establishes $\neg \mathit{Holds}(\mathit{Ab}(X, Y), S0)$. The domain description above is transformed into something like

$$\begin{aligned}&\mathit{Poss}(\mathit{PushButton}, s) \rightarrow\\&\quad ([\neg \mathsf{Ab}(\mathit{IndicatorLightable}, s) \rightarrow\\&\qquad \mathit{State}(\mathit{Do}(\mathit{PushButton}, s)) =\\&\qquad\quad \mathit{State}(s) \circ \mathit{IndicatorLightOn} \circ \mathit{LifeSupportOn}]\\&\quad \wedge\\&\quad [\mathsf{Ab}(\mathit{IndicatorLightable}, s) \rightarrow\\&\qquad \mathit{State}(\mathit{Do}(\mathit{PushButton}, s)) = \mathit{State}(s) \circ \mathit{LifeSupportOn}])\end{aligned} \tag{FC1$'$}$$

$$\mathit{Poss}(\mathit{PushButton}, s) \leftrightarrow \neg \mathsf{Ab}(\mathit{ButtonPushable}, s) \tag{FC2$'$}$$

$$\mathit{Holds}(\mathit{PatientVeryIll}, S0) \tag{FC3$'$}$$

Because of the associated default rules, (FC1′)–(FC3′) still give (Conclusion 1)–(Conclusion 3). But if we now add to the axiomatization the "observation" $\neg \mathit{Holds}(\mathit{IndicatorLightOn}, \mathit{Do}(\mathit{PushButton}, S0))$ the underlying default theory gives two classes of extensions, with $\mathsf{Ab}(\mathit{IndicatorLightable}, S0)$ and $\mathsf{Ab}(\mathit{ButtonPushable}, S0)$ respectively. Although this gives us the desired uncertainty about *LifeSupportOn* in $\mathit{Do}(\mathit{PushButton}, S0)$, it also allows extensions with the negation of (Conclusion 1). In other words it admits the possibility that the failed *PushButton* action has cured the patient! The essence of the problem here is that the solution to the frame problem, encapsulated as it is by state update axioms of the general form $\mathit{Poss}(A(\vec{x}), s) \rightarrow \Gamma[\mathit{State}(\mathit{Do}(A(\vec{x}), s)), \mathit{State}(s)]$, covers only successful actions and not failed action attempts. The same is true of Reiter's Situation Calculus [32]. Agents employing such formalisms are therefore required to detect action failures at the instant of (attempted) execution.

2.2. Action preconditions as narrative constraints

In some formalisms able to represent action occurrences either at a syntactic level (e.g., the version of the Event Calculus in [30]) or at a semantic level (e.g., Language $\mathcal{C}+$), action preconditions are expressed as constraints on the conditions under which actions can occur. For example, in the framework of [30] and in Language $\mathcal{C}+$ the fact that $\neg \mathit{ButtonStuck}$ is an action precondition for *PushButton* would be written respectively as:

$$\mathit{Happens}(\mathit{PushButton}, t) \rightarrow \neg \mathit{HoldsAt}(\mathit{ButtonStuck}, t)$$

$$\textbf{nonexecutable}\ \mathit{PushButton}\ \textbf{if}\ \mathit{ButtonStuck}$$

where the second of these expressions translates to the collection of "causal rules" $\bot \Leftarrow t : (\mathit{PushButton} \wedge \mathit{ButtonStuck})$ for each time-point t. In both cases the semantics ensures that there can be no models in which a *PushButton* occurs at a time-point where *ButtonStuck* is true. Clearly this style of representing action preconditions renders these formalisms inappropriate as a foundation for a solution to the exogenous qualification problem, because it excludes the possibility of explaining unexpected observations after the occurrence of such actions in terms of unusual values for the actions' preconditions (e.g., explanations such as "the indicator failed to light because the button was stuck when an attempt was made to push it"). Other limitations of this type of approach to action preconditions, such as the difficulties it poses for a straightforward logical characterization of planning, are discussed in [28]. But the general point is that it militates against modularization of domain descriptions into narrative and non-narrative components, because the action preconditions constrain the classes of narratives that can be considered.

3. $\mathcal{M}$odular-$\mathcal{E}$

We have opted to investigate the qualification problem by development of an action description language $\mathcal{M}$odular-$\mathcal{E}$ ($\mathcal{ME}$), thus following the methodology first proposed in [8]. $\mathcal{ME}$ is intended as a more expressive successor to the Language $\mathcal{E}$, which in turn was developed from and is closely related to the Event Calculus (see [28] for an overview of the Event Calculus and its relationship with Language $\mathcal{E}$). There are of course pros and cons to using action description languages (ADLs) as opposed to general purpose logic. In this case we have found the ADL methodology useful because the level of abstraction it provides has helped expose key characteristics of a problem on which there has been little previous work. The resulting constraints on the expressivity of $\mathcal{ME}$ (e.g., lack of quantification and nested sentence structures) are, at this stage of our investigation, a small price to pay for the insights gained. Furthermore we shall see that $\mathcal{ME}$ employs a standard model-theoretic notion of entailment, so that the approach may be in principle be adapted for classical logic-based frameworks, such as the Event Calculus, at a later date.[2]

In the next section we give $\mathcal{ME}$'s syntax and sketch its important characteristics via a series of examples. We then give a model-theoretic semantics. $\mathcal{ME}$'s models, like Language $\mathcal{E}$'s, are designations of truth values for each fluent at each point along a time-line that meet certain criteria, including an Event Calculus-like notion of default persistence (to solve the frame problem). As in the Language $\mathcal{E}$, $\mathcal{ME}$'s fluents can only change truth value at time-points at which events, action laws and ramification statements justify the change, but $\mathcal{ME}$ uses a notion of quasi-instantaneous "processes" to compute ramifications, thus allowing it to identify all possible chains of quasi-instantaneous causation even in complex cases where ramifications loop, race and/or compete against each other.

3.1. $\mathcal{M}$odular-$\mathcal{E}$ syntax and examples

Definition 1 *(Domain language).* An $\mathcal{ME}$ **domain language** is a tuple $\langle \Pi, \preccurlyeq, \Delta, \Phi \rangle$, where $\preccurlyeq$ is a total ordering defined over the non-empty set Π of *time-points*, Δ is a non-empty set of *action constants*, and Φ is a non-empty set of *fluent constants*.

Definition 2 *(Formula, literal and conjunction).* A **fluent formula** is a propositional formula containing only fluent constants (used as extra-logical symbols), the standard connectives $\neg$, $\rightarrow$, $\leftarrow$, $\leftrightarrow$, $\vee$ and $\wedge$, and the truth value constants $\top$ and $\bot$. A **fluent literal** is either a fluent constant or its negation. An **action literal** is either an action constant or its negation. A **fluent conjunction** is a conjunction of fluent literals.

Definition 3 *(Converse).* Let E be an action or fluent constant. The **converse** of E, written $\overline{E}$, is $\neg E$, and the converse of $\neg E$, written $\overline{\neg E}$, is E.

Definition 4 *(Domain description or theory).* A **domain description or theory** in $\mathcal{ME}$ is a collection of the following types of statements, where ϕ is a fluent formula, T is a time-point, A is an action constant, C is a (possibly empty) finite set of fluent and action literals, L is a fluent literal, and E is a non-empty finite set of action constants and fluent literals:

[2] An alternative methodology would have been to use the Event Calculus directly as a substrate to this investigation, and attempt to express the definitions in Sections 3.2 and 3.3 as classical logic axioms. However, a little experimentation along these lines shows that this would lead to an axiomatization that was rather dense and hard to follow.

– **h-propositions** of the form: ϕ $\texttt{holds-at}$ T
– **o-propositions** of the form: A $\texttt{occurs-at}$ T
– **c-propositions** of the form: C $\texttt{causes}$ L
– **p-propositions** of the form: ϕ $\texttt{prevents}$ E
– **a-propositions** of the form: $\texttt{always}$ ϕ

A domain description is **finite** if it contains only a finite number of propositions.

Singleton sets of fluent or action literals in c-propositions of the form $\{P\}$ will sometimes be written without enclosing braces, i.e., as P. The intended meaning of h-propositions is straightforward — they can be used to record "observations" about the domain along the time line. The o-proposition "A $\texttt{occurs-at}$ T" means that an attempt to execute A occurs at T. Together, the h- and o-propositions describe the "narrative" component of a domain description. "C $\texttt{causes}$ L" means that, at any time-point, the combination of actions, inactions and preconditions described via C will *provisionally* cause L to hold immediately afterwards. As we shall see, the provisos automatically accompanying this causal rule are crucial — in any model the potential effect L competes with other potential effects, and may be overridden, for example, because it would otherwise result in a more-than-instantaneous violation of a domain constraint described with an a-proposition. The rule "C $\texttt{causes}$ L" is thus qualified both locally (via C) and globally via the total set of c-, p- and a-propositions. "ϕ $\texttt{prevents}$ E" means that the circumstances described by ϕ prevent the simultaneous causation/execution of the effects/actions listed in E. "$\texttt{always}$ ϕ" means that $\neg\phi$ can never hold, other than in temporary, instantaneous "transition states" which form part of an instantaneous chain of indirect effects. In other words, "$\texttt{always}$ ϕ" describes a domain constraint or static law at the granularity of observable time.

In the remainder of this section we give a flavor of $\mathcal{ME}$'s semantics by discussing the models that a series of examples gives rise to. As will be seen in Section 3.3, a *model* in $\mathcal{ME}$ is simply an assignment of a truth value to each fluent (and action) at each time-point that meets certain criteria, and a domain description is *consistent* if it has a model. These examples (as opposed to the standard examples in the literature) have been chosen to illustrate the role of each type of proposition in Definition 4 and to demonstrate how $\mathcal{ME}$ deals with such phenomena as non-determinism, conflicting effects of simultaneously executed actions, looping chains of ramifications, and race conditions between competing series of indirect effects. For all the examples in the remainder of the paper, we assume (unless otherwise stated) that Π in Definition 1 is the set of integers or real numbers, and that Δ and Φ are exactly the action and fluent constants mentioned in the example's propositions.

Example 1 *(Lift door).* A lift door can be opened and closed by pressing the "open" and "close" buttons respectively. The door is initially open, and both buttons are pressed simultaneously. This scenario can be described with a single fluent *DoorOpen* and two actions *PressOpen* and *PressClose*:

$$\{PressOpen\}\ \texttt{causes}\ DoorOpen \tag{LD1}$$

$$\{PressClose\}\ \texttt{causes}\ \neg DoorOpen \tag{LD2}$$

$$DoorOpen\ \texttt{holds-at}\ 1 \tag{LD3}$$

$$PressOpen\ \texttt{occurs-at}\ 2 \tag{LD4}$$

$$PressClose\ \texttt{occurs-at}\ 2 \tag{LD5}$$

Example 1 results in two models — one in which the door is open at times after 2 and one in which the door is closed. Note that, even though the conflicting actions are not prevented from executing together (i.e., there is no p-proposition "$\top$ $\texttt{prevents}$ $\{PressOpen, PressClose\}$"), they do not give rise to inconsistency. This is a manifestation of $\mathcal{ME}$'s "free will" property (formally defined in Section 4) — from any consistent initial state, and for any given collection of c- and p-propositions, any series of actions may be attempted without giving rise to inconsistency (see Corollary 2, Section 4). Put another way, any finite collection of o-, c- and p-propositions is consistent with any internally consistent collection of a-propositions. Consequently, the only way to engineer an inconsistent $\mathcal{ME}$ domain description (other than by inclusion of inconsistent a-propositions) is to include "observations" (h-propositions) along the time line which contradict the predictions that would otherwise be given by $\mathcal{ME}$'s semantics. In Section 6 we show how this remaining type of inconsistency can sometimes be overcome by attributing it to unknown exogenous reasons and applying a simple minimization to these.

Example 2 *(Alternative lift door).* The lift is re-designed so that it is physically impossible to press both buttons at once. But a passenger in the lift still attempts to press them simultaneously at time 2:

$$\{PressOpen\}\ \texttt{causes}\ DoorOpen \tag{ALD1}$$

$$\{PressClose\}\ \texttt{causes}\ \neg DoorOpen \tag{ALD2}$$

$\top$ `prevents` {*PressOpen*, *PressClose*} (ALD3)

DoorOpen `holds-at` 1 (ALD4)

PressOpen `occurs-at` 2 (ALD5)

PressClose `occurs-at` 2 (ALD6)

Again, Example 2 results in two models — one in which the door is open at times after 2 and one in which the door is closed. (ALD3) ensures that exactly one of the two actions fails to execute successfully, and hence fails in all of its effects.

The following series of "broken car" examples is to illustrate the modularity and elaboration tolerance of $\mathcal{ME}$, and how this is linked to the way a- and c-propositions interact.

Example 3 *(Broken car A).* Turning the key of a car causes its engine to start running. The key is turned at time 1:

{*TurnKey*} `causes` *Running* (BC1)

TurnKey `occurs-at` 1 (BC2)

In all models of this domain the car engine is running at all times after 1. (A more complete description would typically include some local qualifications for (BC1), e.g., "{*TurnKey*, *BatteryOK*} `causes` *Running*" — turning the key starts the engine only when the battery is OK, in which case models would also arise where, e.g., $\neg$*BatteryOK* and $\neg$*Running* at all time-points.)

Example 4 *(Broken car B).* We elaborate the previous description by stating in (BC3) that broken cars' engines cannot run:

{*TurnKey*} `causes` *Running* (BC1)

TurnKey `occurs-at` 1 (BC2)

`always` $\neg(\mathit{Broken} \wedge \mathit{Running})$ (BC3)

There are two classes of models for the elaborated domain (BC1)–(BC3) — one in which the car is broken and not running at times after 1, and one in which the car is not broken and running. The occurrence of *TurnKey* at 1 does not eliminate the model in which the car is broken because the semantics of $\mathcal{ME}$ allows (BC3) to act as a global qualification, in particular for (BC1). The *TurnKey* action does not force $\neg$*Broken* at earlier times, and thus if in addition the car is known to be broken the theory remains consistent after this elaboration. Without this characteristic, we would have to alter (BC1) to "{*TurnKey*, $\neg$*Broken*} `causes` *Running*" to accommodate (BC3), in other words explicitly encode as a local qualification the global qualification effect of (BC3) on (BC1). In $\mathcal{ME}$ this local qualification is redundant thus illustrating its modular nature; the a-proposition (BC3) has been simply added without further ado.

Example 5 *(Broken car C).* We elaborate Example 4 with two more causal rules and an extra action occurrence:

{*TurnKey*} `causes` *Running* (BC1)

TurnKey `occurs-at` 1 (BC2)

`always` $\neg(\mathit{Broken} \wedge \mathit{Running})$ (BC3)

{*Break*} `causes` *Broken* (BC4)

{*Broken*} `causes` $\neg$*Running* (BC5)

Break `occurs-at` 1 (BC6)

In all models of the domain (BC1)–(BC6), the car is broken and not running at times after 1. (BC5) describes an "indirect effect" or "ramification". It introduces an asymmetry between the *Running* and *Broken* fluents and their relationship with (BC3), preventing (BC3) from acting as a qualification for (BC4) in the same way as it does for (BC1). Translating global to local/explicit qualifications is therefore complex, as it requires consideration of the interactions between a- and c-propositions. $\mathcal{ME}$ deals with indirect effects by considering chains of instantaneous, temporary transition states ("nodes"). Within these causal chains, "processes" are introduced to describe the initiation and termination of fluents. These processes may "stretch" across several links of a given chain before they are complete, thus allowing all possible micro-orderings of effects to be considered. Because of the potential coarseness of the domain description with respect to the granularity of time, this is important for a proper treatment of collections of instantaneous effects which compete or "race" against each other. Furthermore, since the granularity of time in which these chains operate is finer than that of observable time, intermediate states within them may (temporarily) violate the static laws described by a-propositions. In Example 5, one of the chains allowed by the semantics completes the process initiating *Running* and then the process initiating *Broken*. At this point there is a state in which (BC3) is violated, but (BC5) then generates a new process terminating *Running* whose completion results in a consistent state further along the chain.

We next elaborate the previous two examples with the observation (BC-obs1) that the car is running at time 2:

Example 6 *(Broken car D).*

$$\{\mathit{TurnKey}\}\ \texttt{causes}\ \mathit{Running} \tag{BC1}$$

$$\mathit{TurnKey}\ \texttt{occurs-at}\ 1 \tag{BC2}$$

$$\texttt{always}\ \neg(\mathit{Broken} \wedge \mathit{Running}) \tag{BC3}$$

$$\mathit{Running}\ \texttt{holds-at}\ 2 \tag{BC-obs1}$$

Example 7 *(Broken car E).*

$$\{\mathit{TurnKey}\}\ \texttt{causes}\ \mathit{Running} \tag{BC1}$$

$$\mathit{TurnKey}\ \texttt{occurs-at}\ 1 \tag{BC2}$$

$$\texttt{always}\ \neg(\mathit{Broken} \wedge \mathit{Running}) \tag{BC3}$$

$$\{\mathit{Break}\}\ \texttt{causes}\ \mathit{Broken} \tag{BC4}$$

$$\{\mathit{Broken}\}\ \texttt{causes}\ \neg\mathit{Running} \tag{BC5}$$

$$\mathit{Break}\ \texttt{occurs-at}\ 1 \tag{BC6}$$

$$\mathit{Running}\ \texttt{holds-at}\ 2 \tag{BC-obs1}$$

Adding (BC-obs1) in Example 6 does not result in inconsistency, but allows us to infer that the car is not broken (in particular at earlier times). Note that $\mathcal{ME}$ would facilitate the opposite conclusion (*Broken*) in exactly the same way had the observation been "$\neg$*Running* holds-at 2". This is because it accords exactly the same status to globally derived qualifications (in this case from (BC3)) as to qualifications localized to particular c-propositions. However, adding (BC-obs1) in Example 7 does give rise to inconsistency at the level of the $\mathcal{ME}$'s "base semantics", because since there are no (local or globally derived) qualifications to (BC4) and (BC5), the theory would otherwise entail $\neg$*Running*. An intuitive explanation for (BC-obs1) in this context is that one or both of the effects of (BC4) and (BC5) "failed" due to exogenous circumstances (i.e., factors not included in the representation) implicitly qualifying these causal rules. This type of reasoning is captured within $\mathcal{ME}$ by the use of simple default minimization of such exogenous qualifications (see Section 6). The minimization policy is straightforward and robust because the base semantics fully accounts for all endogenous qualifications (i.e., those expressed in the domain) by virtue of its modularity and its encapsulation of global as well as local qualifications, as described above.

Example 8 *(Broken car F).* We elaborate Example 5 with the knowledge that the car was parked at time 0 in anti-theft mode (ATM), so that causing the engine to run (even for an instant) will trigger the alarm:

$$\{\mathit{TurnKey}\}\ \texttt{causes}\ \mathit{Running} \tag{BC1}$$

$$\mathit{TurnKey}\ \texttt{occurs-at}\ 1 \tag{BC2}$$

$$\texttt{always}\ \neg(\mathit{Broken} \wedge \mathit{Running}) \tag{BC3}$$

$$\{\mathit{Break}\}\ \texttt{causes}\ \mathit{Broken} \tag{BC4}$$

$$\{\mathit{Broken}\}\ \texttt{causes}\ \neg\mathit{Running} \tag{BC5}$$

$$\mathit{Break}\ \texttt{occurs-at}\ 1 \tag{BC6}$$

$$(\neg\mathit{Broken} \wedge \neg\mathit{Running} \wedge \neg\mathit{Alarm} \wedge \mathit{ATM})\ \texttt{holds-at}\ 0 \tag{BC7}$$

$$\{\mathit{Running}, \mathit{ATM}\}\ \texttt{causes}\ \mathit{Alarm} \tag{BC8}$$

Intuitively, even though at times after 1 the car will be broken and not running, the alarm may or may not be triggered in this narrative, depending on whether the (indirect) effect of the *Break* action takes effect just before or just after the effect of the *TurnKey* action. This is an example of a "race" condition between competing instantaneous effects. $\mathcal{ME}$ is able to deal correctly with such representations via its process-based semantics. It gives two models of this domain — in both models ($\mathit{Broken} \wedge \neg\mathit{Running}$) is true at times after 1, but in one model *Alarm* is true and in the other it is false. The example illustrates how $\mathcal{ME}$'s processes operate at a finer level of temporal granularity than "observable time" in order to deal with "instantaneous" indirect effects.[3]

Example 9 *(Oscillator).*

$$\{On\}\ \texttt{causes}\ \neg On \qquad \text{(OSC1)}$$

$$\{\neg On\}\ \texttt{causes}\ On \qquad \text{(OSC2)}$$

This example (which might e.g. represent the internal mechanism of an electric buzzer) has an infinite number of models in which the truth value of *On* is arbitrarily assigned at each time-point. It illustrates that $\mathcal{ME}$ is able to deal with "loops" of indirect effects without over-constraining models. It is important, for example, not to restrict the set of models to those in which the truth value of *On* alternates at each successive time-point. This is because the change within the domain is happening "instantaneously" — i.e., at an altogether finer granularity of time than "observable" time. Therefore the observable time-points are best considered as arbitrarily spaced "snapshots" of the finer-grained time continuum. A full treatment of such loops along these lines (as well as a full treatment of concurrency and non-determinism) is necessary for $\mathcal{ME}$ to exhibit the "free will" property and resulting modularity and elaboration tolerance described above.

3.2. $\mathcal{ME}$ base semantics; states, processes and causal change

In this and the next section we give a formal account of $\mathcal{ME}$'s semantics, starting with some straightforward definitions concerning states and processes.

Definition 5 *(States and satisfaction).* A **state** is a set S of fluent literals such that for each fluent constant F, either $F \in S$ or $\neg F \in S$ but not both. A formula ϕ **is satisfied in a state** S iff the interpretation corresponding to S is a classical model of ϕ.

Definition 6 *(A-consistency).* Let D be a domain description and S a state. S is **a-consistent with respect to** D iff for every a-proposition "$\texttt{always}\ \phi$" in D, ϕ is satisfied in S. D is **a-consistent** iff there exists a state which is a-consistent with respect to D. Let D_a denote the set of all a-propositions in D. Then given a fluent formula ψ, $D_a \models_a \psi$ iff ψ is entailed classically by the theory $T = \{\phi \mid \texttt{always}\ \phi \in D\}$.

Definition 7 *(Process).* A **process** is an expression of the form $\uparrow F$ or $\downarrow F$, where F is a fluent constant of the language. $\uparrow F$ is called **the initiating process of** F and $\downarrow F$ is called **the terminating process of** F. The **associated processes** of the c-propositions "$C\ \texttt{causes}\ F$" and "$C\ \texttt{causes}\ \neg F$" are respectively $\uparrow F$ and $\downarrow F$. $\uparrow F$ and $\downarrow F$ will also sometimes be written as $\text{proc}(F)$ and $\text{proc}(\neg F)$ respectively. An **active process log** is a set of processes.

Definitions 8–15 concern the identification of fluent changes following instantaneously from a given state and set of actions. A *causal chain* represents a possible instantaneous series of knock-on effects (i.e. ramifications) implied by the causal laws. There is a repeated two-phase mechanism for constructing the "nodes" of causal chains — a triggering phase in which new processes are generated from c-propositions applicable at that point, immediately followed by a resolution phase in which some of the already-active processes complete, resulting in an update of the corresponding fluents' truth values. The process triggering is appropriately limited by the p-propositions. The triggering and completion of a particular process may be separated by several steps in the chain, so that consideration of all such chains gives an adequate treatment of "race" conditions between competing instantaneous effects. Chains terminate either because they reach a state from which no change is possible (a *static node*) or because they loop back on themselves. We have made the working (but retractable) assumption that actions trigger processes only at the beginning of such chains, at which point the actions are "consumed".

Definition 8 *(Causal node).* A **causal node** (or **node**) is a tuple $\langle S, B, P\rangle$, where S is a state, B is a set of action constants and P is an active process log. $\langle S, B, P\rangle$ is **fully resolved** iff $P = \emptyset$, and is **a-consistent w.r.t. a domain description** D iff S is a-consistent w.r.t. D.

For example, the causal node intuitively associated with time 2 in Examples 1 and 2 is $\langle\{DoorOpen\}, \{PressOpen, PressClose\}, \emptyset\rangle$.

[3] An interesting (and more contentious) variation of Example 8 is to delete (BC4) and (BC6), and replace (BC7) with "$(Broken \wedge \neg Running \wedge \neg Alarm \wedge ATM)$ $\texttt{holds-at}$ 0" (so that the car is already broken at 1). $\mathcal{ME}$'s semantics still gives the two models with *Alarm* true in one and false in the other. This is because it treats (BC3) only as a "stability" constraint at the temporal granularity of "observable" time, and not as a "definitional" constraint that would transcend all levels of temporal granularity. Note, however, that we could eliminate the model in which *Alarm* was true by adding the p-proposition "*Broken* $\texttt{prevents}$ *Running*", meaning that *Broken* prevents *Running* from being caused (even instantaneously).

Definition 9 *(Triggering).* Let D be a domain description, $N = \langle S, B, P\rangle$ a node, L_t a set of fluent literals, $P_t = \{\mathrm{proc}(L) \mid L \in L_t\}$, and B_t a set of action constants. The set $(B_t \cup P_t)$ is **triggered at** N **with respect to** D iff

(1) $B_t \subseteq B$.
(2) For each p-proposition "ϕ `prevents` E" in D, either ϕ is not satisfied in S or $E \not\subseteq (B_t \cup L_t)$.
(3) For each $L \in L_t$ there is a c-proposition "C `causes` L" in D such that
  (i) for each action constant $A \in C$, $A \in B_t$,
  (ii) for each action literal $\neg A \in C$, $A \notin B_t$, and
  (iii) for each fluent literal $L' \in C$, $L' \in S$.

$(B_t \cup P_t)$ is **maximally triggered at** N **with respect to** D iff there is no other set $(B'_t \cup P'_t)$ also triggered at N with respect to D such that $(B_t \cup P_t)$ is a strict subset of $(B'_t \cup P'_t)$ or B_t is a strict subset of B'_t.

For example, if N is $\langle\{\mathit{DoorOpen}\}, \{\mathit{PressOpen}, \mathit{PressClose}\}, \emptyset\rangle$ and D is the domain of Example 2, then $\{\mathit{PressClose}\} \cup \{\downarrow\mathit{DoorOpen}\}$ is maximally triggered at N w.r.t. D (with L_t in the above definition as $\{\neg\mathit{DoorOpen}\}$). For Example 1, $\{\mathit{PressOpen}, \mathit{PressClose}\} \cup \{\uparrow\mathit{DoorOpen}, \downarrow\mathit{DoorOpen}\}$ is the only maximally triggered set at N w.r.t. D.

Definition 10 *(Process successor).* Let D be a domain description and $N = \langle S, B, P\rangle$ a node. A **process successor of** N **w.r.t.** D is a node of the form $\langle S, B_t, (P \cup P_t)\rangle$, where $(B_t \cup P_t)$ is maximally triggered at N with respect to D.

The node $\langle\{\mathit{DoorOpen}\}, \{\mathit{PressOpen}, \mathit{PressClose}\}, \emptyset\rangle$, has the unique process successor $\langle\{\mathit{DoorOpen}\}, \{\mathit{PressOpen}, \mathit{PressClose}\}, \{\uparrow\mathit{DoorOpen}, \downarrow\mathit{DoorOpen}\}\rangle$ in the context of Example 1.

Definition 11 *(Resolvant).* Let $N = \langle S, B, P\rangle$ and $N' = \langle S', \emptyset, P'\rangle$ be causal nodes. N' is a **resolvant of** N iff either $S' = S$ and $P = P' = \emptyset$, or there exists a non-empty subset R of P such that the following conditions hold.

(1) $P' = P \setminus R$.
(2) For each fluent constant F such that both $\uparrow F$ and $\downarrow F$ are in P, either both or neither $\uparrow F$ and $\downarrow F$ are in R.
(3) For each fluent constant F
  (i) if $\uparrow F \in R$ and $\downarrow F \notin R$ then $F \in S'$,
  (ii) if $\downarrow F \in R$ and $\uparrow F \notin R$ then $\neg F \in S'$,
  (iii) if $\downarrow F \notin R$ and $\uparrow F \notin R$ then $F \in S'$ iff $F \in S$.

N' is a **full resolvant of** N iff $P' = \emptyset$.

The node $\langle\{\mathit{DoorOpen}\}, \{\mathit{PressOpen}, \mathit{PressClose}\}, \{\uparrow\mathit{DoorOpen}, \downarrow\mathit{DoorOpen}\}\rangle$ has two full resolvants in the context of Example 1 — $\langle\{\mathit{DoorOpen}\}, \emptyset, \emptyset\rangle$ and $\langle\{\neg\mathit{DoorOpen}\}, \emptyset, \emptyset\rangle$. This is because none of the constraints on S' in Condition (3) of the above definition are applicable.

Definition 12 *(Stationary/static nodes).* Let D be a domain description and $N = \langle S, B, P\rangle$ a causal node. N is **stationary** iff for each resolvant $\langle S', \emptyset, P'\rangle$ of N, $S' = S$. N is **static w.r.t.** D iff every process successor of N w.r.t. D is stationary.

The central definition of causal chains now follows. Each non-terminating resolvant node in a chain represents an instantaneous intermediary state in the middle of a sequence of ramifications, together with the unresolved processes that may contribute towards the continuation of the sequence. The definition is slightly complicated by the need to deal with loops — Conditions (2), (3) and (4) below ensure that all chains will end when the first static or repeated node is encountered.

Definition 13 *(Causal chain).* Let D be a domain description and let N_0 be a node. A **causal chain rooted at** N_0 **with respect to** D is a (finite) sequence $N_0, N_1, \ldots, N_{2n}$ of nodes such that for each k, $0 \leqslant k \leqslant n-1$, N_{2k+1} is a process successor of N_{2k} w.r.t. D and N_{2k+2} is a resolvant of N_{2k+1}, and such that the following conditions hold:

(1) N_{2n} is fully resolved.
(2) N_{2n} is static, or there exists $k < n$ s.t. $N_{2n} = N_{2k}$.
(3) If there exists $j < k \leqslant n$ s.t. $N_{2j} = N_{2k}$ then $k = n$.
(4) There does not exist a $k < n$ s.t. N_{2k} is static.

In the context of Example 1, Fig. 1 shows the tree of all possible causal chains with the starting node $\langle\{\mathit{DoorOpen}\}, \{\mathit{PressClose}, \mathit{PressOpen}\}, \emptyset\rangle$. N_1 is the unique process successor of N_0, and the nodes N_2 and N'_2 (which are both static) are the only resolvants of N_1. Fig. 2 shows the corresponding tree for Example 2 — now N_0 has two process successors, one where *PressClose* succeeds and $\downarrow\mathit{DoorOpen}$ is triggered and one where *PressOpen* succeeds and $\uparrow\mathit{DoorOpen}$ is triggered.

$$\begin{tikzcd}
& N_0\colon \langle\{\mathit{DoorOpen}\}, \{\mathit{PressClose}, \mathit{PressOpen}\}, \emptyset\rangle \arrow[d] & \\
& N_1\colon \langle\{\mathit{DoorOpen}\}, \{\mathit{PressClose}, \mathit{PressOpen}\}, \{\uparrow\mathit{DoorOpen}, \downarrow\mathit{DoorOpen}\}\rangle \arrow[dl] \arrow[dr] & \\
N_2\colon \langle\{\mathit{DoorOpen}\}, \emptyset, \emptyset\rangle & & N_2'\colon \langle\{\neg\mathit{DoorOpen}\}, \emptyset, \emptyset\rangle
\end{tikzcd}$$

Fig. 1. A tree of causal chains in Example 1.

$$\begin{tikzcd}
& N_0\colon \langle\{\mathit{DoorOpen}\}, \{\mathit{PressClose}, \mathit{PressOpen}\}, \emptyset\rangle \arrow[dl] \arrow[dr] & \\
N_1\colon \langle\{\mathit{DoorOpen}\}, \{\mathit{PressOpen}\}, \{\uparrow\mathit{DoorOpen}\}\rangle \arrow[d] & & N_1'\colon \langle\{\mathit{DoorOpen}\}, \{\mathit{PressClose}\}, \{\downarrow\mathit{DoorOpen}\}\rangle \arrow[d] \\
N_2\colon \langle\{\mathit{DoorOpen}\}, \emptyset, \emptyset\rangle & & N_2'\colon \langle\{\neg\mathit{DoorOpen}\}, \emptyset, \emptyset\rangle
\end{tikzcd}$$

Fig. 2. A tree of causal chains in Example 2.

$$\begin{tikzcd}
N_0\colon \langle\{\mathit{On}\}, \emptyset, \emptyset\rangle \arrow[d] & N_0'\colon \langle\{\neg\mathit{On}\}, \emptyset, \emptyset\rangle \arrow[d] \\
N_1\colon \langle\{\mathit{On}\}, \emptyset, \{\downarrow\mathit{On}\}\rangle \arrow[d] & N_1'\colon \langle\{\neg\mathit{On}\}, \emptyset, \{\uparrow\mathit{On}\}\rangle \arrow[d] \\
N_2\colon \langle\{\neg\mathit{On}\}, \emptyset, \emptyset\rangle \arrow[d] & N_2'\colon \langle\{\mathit{On}\}, \emptyset, \emptyset\rangle \arrow[d] \\
N_3\colon \langle\{\neg\mathit{On}\}, \emptyset, \{\uparrow\mathit{On}\}\rangle \arrow[d] & N_3'\colon \langle\{\mathit{On}\}, \emptyset, \{\downarrow\mathit{On}\}\rangle \arrow[d] \\
N_4\colon \langle\{\mathit{On}\}, \emptyset, \emptyset\rangle & N_4'\colon \langle\{\neg\mathit{On}\}, \emptyset, \emptyset\rangle
\end{tikzcd}$$

Fig. 3. Causal chains in Example 9.

Example 9 shows why it is necessary to take loops into consideration when defining causal chains. Fig. 3 shows the only two possible causal chains for this domain. In both chains, Condition (2) of Definition 13 is satisfied with $k = 0$ and $n = 2$.

As regards Example 8, we may form several causal chains starting from the node corresponding to time 1. Here is a chain terminating with a state in which *Alarm* holds ($\mathit{Br} = \mathit{Broken}$, $\mathit{Ru} = \mathit{Running}$, $\mathit{Al} = \mathit{Alarm}$):

$$N_0\colon \langle\{\neg\mathit{Br}, \neg\mathit{Ru}, \neg\mathit{Al}, \mathit{ATM}\}, \{\mathit{Break}, \mathit{TurnKey}\}, \emptyset\rangle$$
$$N_1\colon \langle\{\neg\mathit{Br}, \neg\mathit{Ru}, \neg\mathit{Al}, \mathit{ATM}\}, \{\mathit{Break}, \mathit{TurnKey}\}, \{\uparrow\mathit{Br}, \uparrow\mathit{Ru}\}\rangle$$
$$N_2\colon \langle\{\mathit{Br}, \mathit{Ru}, \neg\mathit{Al}, \mathit{ATM}\}, \emptyset, \emptyset\rangle$$
$$N_3\colon \langle\{\mathit{Br}, \mathit{Ru}, \neg\mathit{Al}, \mathit{ATM}\}, \emptyset, \{\downarrow\mathit{Ru}, \uparrow\mathit{Al}\}\rangle$$
$$N_4\colon \langle\{\mathit{Br}, \neg\mathit{Ru}, \mathit{Al}, \mathit{ATM}\}, \emptyset, \emptyset\rangle$$

Here is another chain terminating with a state in which $\neg$*Alarm* holds:

$$N_0\colon \langle\{\neg\mathit{Br}, \neg\mathit{Ru}, \neg\mathit{Al}, \mathit{ATM}\}, \{\mathit{Break}, \mathit{TurnKey}\}, \emptyset\rangle$$
$$N_1\colon \langle\{\neg\mathit{Br}, \neg\mathit{Ru}, \neg\mathit{Al}, \mathit{ATM}\}, \{\mathit{Break}, \mathit{TurnKey}\}, \{\uparrow\mathit{Br}, \uparrow\mathit{Ru}\}\rangle$$
$$N_2'\colon \langle\{\mathit{Br}, \neg\mathit{Ru}, \neg\mathit{Al}, \mathit{ATM}\}, \emptyset, \{\uparrow\mathit{Ru}\}\rangle$$
$$N_3'\colon \langle\{\mathit{Br}, \neg\mathit{Ru}, \neg\mathit{Al}, \mathit{ATM}\}, \emptyset, \{\downarrow\mathit{Ru}, \uparrow\mathit{Ru}\}\rangle$$
$$N_4'\colon \langle\{\mathit{Br}, \neg\mathit{Ru}, \neg\mathit{Al}, \mathit{ATM}\}, \emptyset, \emptyset\rangle$$

Nodes, and in particular nodes that terminate causal chains, do not necessarily contain a-consistent states. But causal chains that do not terminate a-consistently are not discarded when computing direct and indirect instantaneous effects. Rather, the semantics identifies *proper causal descendants* within a tree of all possible causal chains starting from a given root node.[4] These are a-consistent nodes which are either within the terminating loop of a chain (Condition (1) in Definition 14), or are such that there are no other a-consistent nodes further from the root of the tree (Condition (2) in Definition 14). (For example, in Fig. 1, N_2 and N_2' are proper causal descendants of N_0 by Condition (1) below, with $j = k = n = 1$. In Fig. 3, N_0, N_2 and N_4 are proper causal descendants of N_0 with $j = 0$ and $n = 2$.)

Definition 14 *(Proper causal descendant).* Let D be a domain description and let N_0 and N be nodes. N is a **proper causal descendant of** N_0 **w.r.t.** D iff N is a-consistent w.r.t. D, and there exists a causal chain $N_0, N_1, \ldots, N_{2n}$ w.r.t. D such that $N = N_{2k}$ for some $0 \leqslant k \leqslant n$ and at least one of the following two conditions holds:

(1) There exists $j \leqslant k$ such that $N_{2j} = N_{2n}$.

[4] Note that in contrast to traditional tree terminology where a root is an "improper descendant", here the root itself can potentially be a "proper causal descendant".

$$\begin{tikzcd}
 & \langle\{\neg LO, \neg BS, OC\}, \{Promote\}, \emptyset\rangle \arrow[d] & \\
 & \langle\{\neg LO, \neg BS, OC\}, \{Promote\}, \{\uparrow LO, \uparrow BS\}\rangle \arrow[dl] \arrow[d] \arrow[dr] & \\
\langle\{LO, \neg BS, OC\}, \emptyset, \{\uparrow BS\}\rangle \arrow[d] & \langle\{LO, BS, OC\}, \emptyset, \emptyset\rangle & \underline{\langle\{\neg LO, BS, OC\}, \emptyset, \{\uparrow LO\}\rangle} \arrow[d] \\
\langle\{LO, \neg BS, OC\}, \emptyset, \{\uparrow BS\}\rangle \arrow[d] & & \langle\{\neg LO, BS, OC\}, \emptyset, \{\uparrow LO\}\rangle \arrow[d] \\
\langle\{LO, BS, OC\}, \emptyset, \emptyset\rangle & & \langle\{LO, BS, OC\}, \emptyset, \emptyset\rangle
\end{tikzcd}$$

Fig. 4. A tree of causal chains in Example 10.

(2) There does not exist a causal chain $N_0, N_1, \ldots, N_{2k}, N'_{2k+1}, \ldots, N'_{2m}$ w.r.t. D and a j such that $k < j \leqslant m$ and N'_{2j} is a-consistent w.r.t. D.

It is also useful to define a *stable state* as a state that does not always immediately cause its own termination (note that stable states can be in loops, but must be a-consistent):

Definition 15 *(Stable state).* Let D be a domain description and let S be a state. S is **stable w.r.t.** D if there exists a node $\langle S, \emptyset, P\rangle$ which is a proper causal descendant of $\langle S, \emptyset, \emptyset\rangle$.

Example 10 *(Promotion).* An employee gets promoted at time 1. Promotion results in a large office (LO) and big salary (BS). But nobody gets a large office when the building is overcrowded (OC), which it is at time 1:

$$\texttt{always}\ \neg(OC \wedge LO) \tag{PR1}$$

$$Promote\ \texttt{causes}\ LO \tag{PR2}$$

$$Promote\ \texttt{causes}\ BS \tag{PR3}$$

$$Promote\ \texttt{occurs-at}\ 1 \tag{PR4}$$

$$(\neg LO \wedge \neg BS \wedge OC)\ \texttt{holds-at}\ 1 \tag{PR5}$$

The tree of possible causal chains that arise at time 1 in this example, with the single proper causal descendant of the root node underlined, is given in Fig. 4.

This example illustrates the role of a-propositions in defining possible change. In contrast with p-propositions, they do not a-priori block processes from starting (being triggered). Rather, in the micro-cosmos of causal change at a single time-point, they even allow instantaneous change to occur that would violate them, provided that terminating nodes are consistent with them. If the terminating nodes are not consistent, as in this example, then we take as the possible change the deepest nodes that are consistent in the sense that for each such node there is no other node in a branch emanating from it that is also a-consistent. Hence a-propositions, as properties on the macro-cosmos of the observed states, constrain which triggered process are allowed to complete.

3.3. $\mathcal{ME}$ base semantics; time and temporal change

If a causal node corresponds to a particular time-point in the narrative of a given domain description (e.g., in Fig. 1, N_0 corresponds to time 2), then Definitions 16–22 below ensure that the states within its proper causal descendants indicate possible choices as to which fluents will change values in the time period immediately afterwards. These definitions are largely modifications of those in [16], but with the notion of a *change set* replacing that of initiation/termination points.

Definition 16 *(Interpretation/event matching).* An **interpretation** of $\mathcal{ME}$ is a mapping $H : (\Phi \cup \Delta) \times \Pi \to \{\text{true}, \text{false}\}$. H **event matches** the domain description D iff

$$\{\langle A, T\rangle \mid H(A, T) = \text{true}\} = \{\langle A, T\rangle \mid \text{“}A\ \texttt{occurs-at}\ T\text{”} \in D\}.$$

Definition 17 *(Time-point satisfaction).* Given a fluent formula ϕ of $\mathcal{ME}$ and a time-point T, an interpretation H **satisfies** ϕ **at** T iff the mapping M_T defined by $\forall F, M_T(F) = H(F, T)$ is a model of ϕ. Given a set Z of fluent formulae, H **satisfies** Z **at** T iff H satisfies ϕ at T for each $\phi \in Z$.

Definition 18 *(State/event base at a time-point).* Let D be a domain description, H an interpretation, and T a time-point. The **state at** T **w.r.t.** H, denoted $S(H, T)$, is the state $\{F \mid H(F, T) = \text{true}\} \cup \{\neg F \mid H(F, T) = \text{false}\}$. The **event base at** T **w.r.t.** H, denoted $B(H, T)$, is the set $\{A \mid H(A, T) = \text{true}\}$.

The definitions of a causal frontier and a change set now follow. A causal frontier at a particular time-point represents the state in a terminating node of a causal chain triggered at that time-point. The associated change set represents the difference (in terms of fluent literals) between this state and the state at the start of the causal chain.

Definition 19 *(Causal frontier).* Let D be a domain description, T a time-point, H an interpretation and S' a state. S' is a **causal frontier of** H **at** T **w.r.t.** D iff there exists a node $N = \langle S', B, P\rangle$ such that N is a proper causal descendant of $\langle S(H,T), B(H,T), \emptyset\rangle$ w.r.t. D.

Definition 20 *(Change set).* Let D be a domain description, H an interpretation, T a time-point and $\mathcal{C}$ a set of fluent literals. $\mathcal{C}$ is a **change set at** T **w.r.t.** H iff there exists a causal frontier S of H at T w.r.t. D such that $\mathcal{C} = S \setminus S(H,T)$.

Definition 21 *(Change mapping).* Let Φ^* be the set of all fluent literals (i.e. all fluent constants and their negations) in the language. A **change mapping** is a mapping $c : \Pi \rightarrow 2^{\Phi^*}$.

Definition 22 *(Model).* Let D be a domain description and let H be an interpretation that event matches D. Then H is a **model** of D iff there exists a change mapping c such that for all T, $c(T)$ is a change set at T w.r.t. H, and the following three conditions hold for every fluent literal L and time-points $T_1 \prec T_3$:

(1) If H satisfies L at T_1, and there is no time-point T_2 s.t. $T_1 \preccurlyeq T_2 \prec T_3$ and $\bar{L} \in c(T_2)$, then H satisfies L at T_3.
(2) If $L \in c(T_1)$, and there is no time-point T_2 s.t. $T_1 \prec T_2 \prec T_3$ and $\bar{L} \in c(T_2)$, then H satisfies L at T_3.
(3) H satisfies the following constraints:
 – For all h-propositions "ϕ `holds-at` T" in D, H satisfies ϕ at T.
 – For all time-points T, $S(H,T)$ is a stable state.

Intuitively, Condition (1) above states that fluents change their truth values only via successful effects of c-propositions, and (2) states that successfully initiating a literal establishes its truth value as true. Note also that Condition (3)'s requirement of stability ensures that $S(H,T)$ is a-consistent.

Definition 23 *(Consistency and entailment).* A domain description D is **consistent** if it has a model. D **entails** the h-proposition "ϕ `holds-at` T", written $D \models \phi$ `holds-at` T, iff for every model M of D, M satisfies ϕ at T.

Example 11 *(Faulty electric circuit).* The current in a faulty circuit is switched on causing a broken fuse, which in turn terminates the current:

$$\{\mathit{SwitchOn}\}\ \texttt{causes}\ \mathit{ElectricCurrent} \tag{EC1}$$

$$\{\mathit{ElectricCurrent}\}\ \texttt{causes}\ \mathit{BrokenFuse} \tag{EC2}$$

$$\{\mathit{BrokenFuse}\}\ \texttt{causes}\ \neg\mathit{ElectricCurrent} \tag{EC3}$$

$$\texttt{always}\ \neg(\mathit{ElectricCurrent} \wedge \mathit{BrokenFuse}) \tag{EC4}$$

$$\mathit{SwitchOn}\ \texttt{occurs-at}\ 1 \tag{EC5}$$

One causal chain that could be triggered at time 1 (with non-a-consistent nodes N_4 and N_5) is:

N_0: $\langle\{\neg\mathit{ElectricCurrent}, \neg\mathit{BrokenFuse}\}, \{\mathit{SwitchOn}\}, \emptyset\rangle$

N_1: $\langle\{\neg\mathit{ElectricCurrent}, \neg\mathit{BrokenFuse}\}, \{\mathit{SwitchOn}\}, \{\uparrow\mathit{ElectricCurrent}\}\rangle$

N_2: $\langle\{\mathit{ElectricCurrent}, \neg\mathit{BrokenFuse}\}, \emptyset, \emptyset\rangle$

N_3: $\langle\{\mathit{ElectricCurrent}, \neg\mathit{BrokenFuse}\}, \emptyset, \{\uparrow\mathit{BrokenFuse}\}\rangle$

N_4: $\langle\{\mathit{ElectricCurrent}, \mathit{BrokenFuse}\}, \emptyset, \emptyset\rangle$

N_5: $\langle\{\mathit{ElectricCurrent}, \mathit{BrokenFuse}\}, \emptyset, \{\downarrow\mathit{ElectricCurrent}\}\rangle$

N_6: $\langle\{\neg\mathit{ElectricCurrent}, \mathit{BrokenFuse}\}, \emptyset, \emptyset\rangle$

This chain is well-formed because N_6 is the first static resolvant node and is fully resolved (Definition 13). N_6 is a-consistent and therefore is a proper causal descendant of N_0 (Definition 14). So $\{\neg\mathit{ElectricCurrent}, \mathit{BrokenFuse}\}$ is a causal frontier at 1 of any interpretation that satisfies $(\neg\mathit{ElectricCurrent} \wedge \neg\mathit{BrokenFuse})$ at 1 (Definition 19), thus providing the change set $\{\mathit{BrokenFuse}\}$ (Definition 20). Note that at the temporal granularity level of the representation of this example (i.e. the granularity of models and of h-propositions), *ElectricCurrent*, the cause of *BrokenFuse*, is never true! *ElectricCurrent* is true only at a finer granularity (the granularity inside causal chains).

4. Properties and characteristics of $\mathcal{ME}$

4.1. Results concerning elaboration tolerance

As we have seen, $\mathcal{ME}$ provides principled, general mechanisms for causal laws to be qualified both by each other and by static laws, thus integrating all endogenous qualifications within one base-level semantic framework. We have also hinted that $\mathcal{ME}$ provides a high degree of modularity by its separation of information about causality (c-, p- and a-propositions), narrative information about attempted actions (o-propositions), and narrative information in the form of observations about fluents (h-propositions), and that these qualities make $\mathcal{ME}$ domain descriptions particularly elaboration tolerant. We now substantiate these claims with some formal results.

Theorem 1 and its corollaries below effectively show that an action occurrence (o-proposition) cannot be used as a kind of "observation" about what holds at the associated time-point. We label these results as "free will" properties because they effectively state that an agent may attempt any action in any circumstance. This is an important principle of modularity because it helps separate narrative information about actions from narrative information about fluents, and an important principle of elaboration tolerance because it allows us to add arbitrary o-propositions to the theory that refer to current and future time-points, irrespective of observations (h-propositions) referring to past time-points. Theorem 2 focuses on the elaboration tolerance of causal knowledge, showing in particular that we can add any arbitrary c- and p-propositions to a previously consistent collection of causal statements. This is because of the "completeness" of our solution to the endogenous qualification problem — $\mathcal{ME}$ is guaranteed to automatically qualify the new and existing causal knowledge so that it is mutually compatible.

Definition 24 *(Pre- and post-observation/action points).* Given a domain description D, a **post-observation point** of D is a time-point T_p such that, for every h-proposition of the form "ϕ `holds-at` T" in D, $T \preccurlyeq T_p$. A **pre-action point** (respectively **post-action point**) is a time-point T_a such that, for every o-proposition "A `occurs-at` T" in D, $T_a \preccurlyeq T$ (respectively $T_a \succ T$).

Definition 25 *(Projection domain description).* The domain description D is a **projection domain description** if there exists a time-point which is both a post-observation point and a pre-action point of D.

Lemma 1 *(Existence of proper causal descendant). Let D be a finite domain description and let $N_0 = \langle S_0, B_0, \emptyset\rangle$ be a node such that S_0 is a-consistent w.r.t. D. Then there exists a node $N = \langle S, B, P\rangle$ such that N is a proper causal descendant of N_0. Furthermore, if S_0 is stable w.r.t. D then so is S.*

Proof. It is sufficient to show that there exist a non-zero, finite number of causal chains rooted at N_0 w.r.t. D. The lemma then follows by consideration of the (finite) tree of all causal chains rooted at N_0, by noting that by Definition 14, if there were no proper causal descendants of N_0 then for any a-consistent node in the tree there would be another descendant node that was also a-consistent. Given that N_0 is a-consistent and the tree is finite this would lead to a contradiction.

To show that there exists a causal chain rooted at N_0 w.r.t. D, we argue as follows. First, notice that for any node there always exists a (possibly empty) maximally triggered set w.r.t. D, and therefore there always exists at least one process successor. By definition there also always exists a full resolvant of any node. Therefore the set of infinite sequences of the form $N_0, N_1, \ldots, N_{2m}, \ldots$ such that for each k, N_{2k+1} is a process successor of N_{2k} w.r.t. D and N_{2k+2} is a full resolvant of N_{2k+1} is non-empty. Consider the set of all finite sub-sequences of this set which start at N_0 and have as their last node the first N_{2k} which either appears earlier in the sub-sequence or is static w.r.t. D. Such a last node exists for each sequence because of the finiteness of D. By construction, each of these sub-sequences satisfy Conditions (1) to (4) of Definition 13 and therefore are causal chains rooted at N_0 with respect to D.

For a finite domain D, there can only be a finite number of causal chains because of the repetition restriction of Condition (3) of Definition 13. □

Lemma 2 *(Existence of change set). Let D be a finite domain description, and let H be an interpretation of D and T be a time-point such that $S(H, T)$ is a-consistent w.r.t. D. Then there exists a change set at T w.r.t. H.*

Proof. This follows directly from Lemma 1. □

Lemma 3 *(Existence of empty change set). Let D be a domain description and T be a post-observation and post-action point of D. Let S be a state which is stable w.r.t. D, and let H be an interpretation of D such that $S = S(H, T')$ for some $T' \succ T$. Then $\emptyset$ is a change set at T' w.r.t. H.*

Proof. This follows directly from Definition 15. □

Theorem 1 *(Free will theorem). Let M be a model of a finite domain description D, let O be a finite set of o-propositions, and let T_n be a time-point which is both a post-observation point for D and a pre-action point for O. Then there is a model M_O of $D \cup O$ such that for any fluent F and time-point $T \preccurlyeq T_n$, $M_O(F,T) = M(F,T)$.*

Proof. By induction on the number, k, of o-propositions in O. The base case of $k = 0$ is trivial. Suppose that the result holds whenever O contains k o-propositions and consider a set O with $k + 1$ o-propositions. Choose an o-proposition, A $\texttt{occurs} - \texttt{at}$ T_g, from O such that there exists no other o-proposition, A' $\texttt{occurs} - \texttt{at}$ T' in O with $T' \succ T_g$ (i.e. one of the o-propositions with the greatest time-point). Let O' denote the set O minus the element A $\texttt{occurs} - \texttt{at}$ T_g. By the inductive hypothesis on O' there exists a model $M_{O'}$ of $D \cup O'$ such that for any fluent F and time-point $T \preccurlyeq T_n$, $M_{O'}(F,T) = M(F,T)$.

We build the required model M_O of $D \cup O$ as follows. For any fluent F and time-point $T' \preccurlyeq T_g$, $M_O(F,T') = M_{O'}(F,T')$. Choose a change mapping c such that for any time $T' \prec T_g$ this is equal to the change mapping of the model $M_{O'}$. For the time-point T_g, consider the state, $S(M_{O'}, T_g)$, at T_g w.r.t. $M_{O'}$. By Lemma 2 there exists a change set, $\mathcal{C}$, at T_g w.r.t. $M_{O'}$. We then let $c(T_g) = \mathcal{C}$.

We complete the model M_O and the change mapping c for time-points after T_g as follows. For any fluent F if $F, \neg F \notin \mathcal{C}$ then for any $T'' \succ T_g$, $M_O(F,T'') = M_{O'}(F,T_g)$. Otherwise, if $F \in \mathcal{C}$ then for any $T'' \succ T_g$, $M_O(F,T'') = \text{true}$ and if $\neg F \in \mathcal{C}$ then for any $T'' \succ T_g$, $M_O(F,T'') = \text{false}$. For any time-point $T'' \succ T_g$, $c(T'') = \emptyset$.

By construction and Lemma 3, c is a change mapping w.r.t. M_O. The interpretation M_O satisfies the three axioms of a model. Conditions (1) and (2) follow directly from its construction. Condition (3.i) is trivial. For $T' \preccurlyeq T_g$, Condition (3.ii) follows from the stability of $S(M_{O'}, T')$. For $T'' \succ T_g$, the model M_O is constructed directly from the change set $\mathcal{C}$, and so the stability Condition (3.ii) follows from the observation that, by Definitions 14 and 15, the state component of a node which is a proper causal descendant must necessarily be stable. □

Corollary 1 *(Free will corollary). Let D and D' be domain descriptions and let T_n be a post-observation point for both D and D'. Let D and D' differ only by o-propositions referring to time-points greater than or equal to T_n and let M be a model of D. Then there is a model M' of D' such that $M(F,T) = M'(F,T)$ for all fluent constants F and all time-points T such that $T \preccurlyeq T_n$.*

Proof. This follows directly from Lemma 3 and Theorem 1, observing that D' can be constructed from D by first removing all the o-propositions in D which refer to time-points greater than or equal to T_n, and then adding the required o-propositions for D'. □

Corollary 2 *(Action elaboration tolerance corollary). Let D be a consistent domain description and let O be a finite set of o-propositions. If there exists a time-point T_n which is both a post-observation point for D and a pre-action point for O, then $D \cup O$ is consistent.*

Proof. This follows directly from Theorem 1. □

Theorem 2 demonstrates the robustness and elaboration tolerance of $\mathcal{ME}$ theories by showing that their consistency is contingent only on the internal consistency of the static laws and on whether observations match with predicted effects.

Theorem 2 *(Theorem of causal elaboration tolerance). Let D_a be a consistent domain description consisting only of a-propositions and let E be a finite set of o-, c- and p-propositions. Then $D_a \cup E$ is also a consistent domain description.*

Proof. Consider the domain D obtained from $D_a \cup E$ by deleting all its o-propositions. Let $val : \Phi \to \{\text{true}, \text{false}\}$ be a truth assignment on the set of fluents, Φ, of $D_a \cup E$ which is a model of the a-propositions of D_a (and hence also of D). Let H be the interpretation of D given by $H(F,T) = val(F)$ for all time-points T and any fluent F. By Lemma 3 the empty set is a change set at any time-point T w.r.t. H. It is easy to then see that H is a model of D with these change sets. By Theorem 1 the domain $D_a \cup E$, which can be obtained from D by adding back to it the o-propositions in E, has a model (that agrees with H for all time-points smaller than or equal to the earliest time-point referred to in the (finite) o-propositions of E). □

4.2. The qualifying role of a-propositions

As we have seen, $\mathcal{ME}$'s semantics allows the a-propositions in a domain description to *implicitly* qualify its causal effect laws. It is natural to ask whether, in principle, we could do without the qualifying role of a-propositions (we would still of course need to employ some mechanism for constraining, for example, the initial state). In other words, could the causal knowledge contained within an (arbitrary) domain description be re-written without a-propositions by including instead extra *explicit* qualifications within (possibly additional) c- and p-propositions? And even if the formal correctness of such a general transformation method could be ascertained, would it result in reasonably-sized, elaboration tolerant domain descriptions, and could it in the general case be computed within a reasonable time-frame?

It is outside the main focus of this paper to study these questions in detail and to answer them definitively — domains such as Example 8 with "side effects" (see in particular the remarks in the associated footnote) give an indication that a

$$\begin{tikzcd}
 & \langle\{\neg \mathit{AliceIn}, \neg \mathit{BobIn}\}, \{\mathit{AliceEnters}, \mathit{BobEnters}\}, \emptyset\rangle \arrow[d] & \\
 & \langle\{\neg \mathit{AliceIn}, \neg \mathit{BobIn}\}, \{\mathit{AliceEnters}, \mathit{BobEnters}\}, \{\uparrow \mathit{AliceIn}, \uparrow \mathit{BobIn}\}\rangle \arrow[dl] \arrow[d] \arrow[dr] & \\
\underline{\langle\{\mathit{AliceIn}, \neg \mathit{BobIn}\}, \emptyset, \{\uparrow \mathit{BobIn}\}\rangle} \arrow[d] & \langle\{\mathit{AliceIn}, \mathit{BobIn}\}, \emptyset, \emptyset\rangle & \underline{\langle\{\neg \mathit{AliceIn}, \mathit{BobIn}\}, \emptyset, \{\uparrow \mathit{AliceIn}\}\rangle} \arrow[d] \\
\langle\{\mathit{AliceIn}, \neg \mathit{BobIn}\}, \emptyset, \{\uparrow \mathit{BobIn}\}\rangle \arrow[d] & & \langle\{\neg \mathit{AliceIn}, \mathit{BobIn}\}, \emptyset, \{\uparrow \mathit{AliceIn}\}\rangle \arrow[d] \\
\langle\{\mathit{AliceIn}, \mathit{BobIn}\}, \emptyset, \emptyset\rangle & & \langle\{\mathit{AliceIn}, \mathit{BobIn}\}, \emptyset, \emptyset\rangle
\end{tikzcd}$$

Fig. 5. Causal chains in Example 12.

proof one way or the other[5] for such a transformation would be technically involved. However, it is perhaps worth touching the surface of some of the issues involved in this question by considering an example.

Example 12 *(Full lift).* A certain lift cannot hold more than one person. Two people attempt to enter the lift at the same time:

$$\{\mathit{AliceEnters}\}\ \texttt{causes}\ \mathit{AliceIn} \tag{FL1}$$

$$\{\mathit{BobEnters}\}\ \texttt{causes}\ \mathit{BobIn} \tag{FL2}$$

$$\texttt{always}\ \neg(\mathit{AliceIn} \wedge \mathit{BobIn}) \tag{FL3}$$

$$\neg \mathit{AliceIn} \wedge \neg \mathit{BobIn}\ \texttt{holds-at}\ 0 \tag{FL4}$$

$$\mathit{AliceEnters}\ \texttt{occurs-at}\ 1 \tag{FL5}$$

$$\mathit{BobEnters}\ \texttt{occurs-at}\ 1 \tag{FL6}$$

The tree of causal chains for this domain at time 1 is shown in Fig. 5. The two proper causal descendants (underlined) show that under $\mathcal{ME}$'s semantics the two causal laws (FL1) and (FL2) are non-deterministically qualified by the a-proposition of the domain.

Can this qualifying effect of the a-proposition be obtained in some other way? Let us first consider the naive approach of simulating the implicit qualification effect of the a-proposition (FL3) only by adding extra explicit qualifying conditions to (FL1) and (FL2), to form (FL1′) and (FL2′) respectively. We consider the tree of causal chains for this new domain at time 1 starting from a state where both *AliceIn* and *BobIn* are false as before. There are two cases to consider. First, suppose that the extra conditions of both (FL1′) and (FL2′) are satisfied by the initial node. In this case we end up with the same tree of causal chains as in the original domain, the end nodes of which would give rise to a model in which both *AliceIn* and *BobIn* were true. Second, suppose the extra conditions of at least one of (FL1′) and (FL2′) are not satisfied by the initial node. In this case the corresponding effect will not be triggered. This implies that the new domain will not have at least one of the two models that $\mathcal{ME}$ gives to the original domain.

An equivalent domain without extra c- or p-propositions is therefore impossible in this case. Intuitively, the role of the a-proposition (FL3) is best captured explicitly with three extra p-propositions:

$$\top\ \texttt{prevents}\ \{\mathit{AliceIn}, \mathit{BobIn}\}$$

$$\mathit{AliceIn}\ \texttt{prevents}\ \{\mathit{BobIn}\}$$

$$\mathit{BobIn}\ \texttt{prevents}\ \{\mathit{AliceIn}\}$$

It is easy to see that replacing (FL3) with these propositions does indeed give rise to the same models (even when only one of the fluent literals in (FL4) is negative, although these replacements do not of course independently constrain the initial state in the same way as (FL3)). But this is not to say that, from a practical point of view, a-propositions are unnecessary in examples such as this. For a similar example with a lift that cannot contain more than n people, the equivalent transformation to exclude a-propositions will give rise to a domain with size exponentially larger in n than the original.

So it is clear that a-propositions are a useful addition to the language. Moreover, Theorem 1 and its corollaries make a-propositions especially useful when thinking about $\mathcal{ME}$ in terms of state transition systems. A collection of c- and p-propositions effectively defines a state transition graph, which is *complete* in the sense that each arbitrary finite collection of actions gives rise to at least one directed edge from each vertex of the graph. The results of the previous subsection guarantee that the addition of an a-proposition to the theory filters out vertices of the graph without compromising this

[5] To prove, for example, that a-propositions are necessary, we would have to show that, for some particular collection T of c-, p- and a-propositions, there does not exist any collection T' of c- and p-propositions such that for any collection N of o- and h-propositions $T \cup N$ and $T' \cup N$ have the same set of models provided that the h-propositions in N ensure satisfaction of the a-propositions in T at an initial time-point.

completeness.[6] In turn this means that $\mathcal{ME}$ can guarantee static constraint properties that our problem specification may require by simply adding the associated a-propositions in the domain description, without affecting the completeness or consistency of the specification as a whole. We have studied the question of explicating the implicit qualifications of a-propositions in more detail in a technical annex[7] to this paper [14].

5. Fluents with default values

In some domains it is useful to be able to specify that some fluents are "normally true" or "normally false". For example, cars are normally not broken. In this section we extend the syntax of $\mathcal{M}$odular-$\mathcal{E}$ with *n-propositions* to facilitate the expression of such information. As we will see in Section 6, this extension also helps us address the exogenous qualification problem in a natural way.

Definition 26 *(N-theory or n-domain description).* An **n-theory or n-domain description** in $\mathcal{ME}$ is a domain description which may additionally contain **n-propositions** of the form

$$\texttt{normally}\, L$$

where L is a fluent literal. An interpretation is a **base model** or **b-model** of an n-theory iff it is a model of the n-theory without its n-propositions.

We have taken the view that n-propositions should play a relatively weak role in a domain description compared with explicit causal information (and specific observations). Hence the intended meaning of "`normally` L" is "if there is no evidence or cause for $\bar{L}$, assume L by default". To illustrate this in more detail, it is helpful to consider various combinations of the following n-, c- and h-propositions:

$$\texttt{normally}\, \neg Broken \tag{N1}$$

$$VeryColdWeather\ \texttt{causes}\ Broken \tag{N2}$$

$$\texttt{normally}\, VeryColdWeather \tag{N3}$$

$$\neg Broken\ \texttt{holds-at}\ 1 \tag{N4}$$

Let $D_n^1 = \{(\text{N1})\}$, $D_n^2 = \{(\text{N1}), (\text{N2})\}$, $D_n^3 = \{(\text{N1}), (\text{N2}), (\text{N3})\}$, and $D_n^4 = \{(\text{N1}), (\text{N2}), (\text{N3}), (\text{N4})\}$. Since n-propositions are "default statements", $\mathcal{ME}$ provides semantics for them by using them to define a preference relation between b-models (i.e., the models of the equivalent domain description without n-propositions). As we will see, we then define "n-models" as the most preferred b-models under this relation. We take the view that D_n^1 should have just one n-model in which $\neg Broken$ holds at all time-points (assuming *Broken* is the only fluent in the language). D_n^2 should have two n-models, one in which *VeryColdWeather* and *Broken* are both true and one in which they are both false. In the context of D_n^2, (N1) should eliminate the combination of $\neg VeryColdWeather$ and *Broken*, and (N2) should eliminate the combination of *VeryColdWeather* and $\neg Broken$. But note that (N1) should not eliminate the combination of *VeryColdWeather* and *Broken* as an n-model of D_n^2, because (N2) gives a cause for *Broken* in this context. As regards D_n^3, the addition of (N3) should also eliminate the combination $\neg VeryColdWeather$ and $\neg Broken$ (since there is no identifiable cause for $\neg VeryColdWeather$), leaving just one n-model with *VeryColdWeather* and *Broken*. As before, (N1) should not in this case eliminate this remaining model because (N2) gives *Broken* an identifiable cause. Finally, in the context of D_n^4, the addition of (N4) should override the default of (N3) to give just one n-model in which $\neg VeryColdWeather$ and $\neg Broken$ (the same model as this domain description would have without the n-propositions).

For domains with a least time-point (i.e. time 0 in the case of the naturals), it is only necessary to compare b-models at this initial time-point, since, in all $\mathcal{ME}$ models, all fluent truth values (default or otherwise) persist until their converse is caused. But, since $\mathcal{ME}$ can also accommodate time structures such as the reals or integers with no least point, for maximum generality we compare b-models at all time-points in some initial interval of the time-line. To identify which fluents have a "cause" at such time-points, we can build upon the notion of stability (Definition 15) used to provide a pointwise constraint on models in Definition 22. Stability ensures that, at any time-point, the processes which are triggered by the associated state alone (i.e. disregarding actions) can result (directly or indirectly) in a return to that same state. We can examine these

[6] State transition graphs are widely used in Computer Science to model and specify system behavior, and it is often desirable to prove certain properties of these graphs. For example, in a safety-critical event-driven system it may be necessary to demonstrate that no possible combination of inputs (events) will ever lead to an unsafe output state. Using $\mathcal{ME}$ as the requirements specification language, it would be possible to guarantee such a safety constraint by simply adding the associated a-proposition, without compromising the completeness or consistency of the specification as a whole.

[7] In this we argue that an attempt to rely solely on explicit qualifications when representing a domain is often impossible due to the different nature of a-propositions from c-propositions and p-propositions. Under some severe restrictions on the domain description a translation is possible but this might require an exponential growth of the domain size, or exponential-time reasoning in order to identify appropriate qualifications. Furthermore, such translated domains compromise elaboration tolerance, in the sense that the translation is not modular: adding new information to the domain can require the re-translation of the whole domain, not just the newly added statements.

same processes to identify "instantaneous causes" for some of the fluent literals in the state (Condition (2) in the definition below). Because causal chains terminate at static nodes, we also need to identify "one step" instantaneous causes separately (Condition (1) in the definition below).

Definition 27 *(Instantaneous cause).* Let D be an n-domain description, T be a time-point, L be a fluent literal, and M be a b-model of D. L has an **instantaneous cause at** T **w.r.t.** D **and** M iff either one of the following conditions holds:

(1) $\langle S(M,T), \emptyset, \emptyset\rangle$ is static and has a (stationary) process successor w.r.t. D that contains proc(L).
(2) There exists a k and n such that $0 \leqslant k \leqslant n$, a set P of processes and a causal chain $N_0, N_1, \ldots, N_{2k}, \ldots, N_{2n}$ such that
  (i) $N_0 = \langle S(M,T), \emptyset, \emptyset\rangle$,
  (ii) $N_{2k} = \langle S(M,T), \emptyset, P\rangle$,
  (iii) N_{2k} is a proper causal descendant of N_0, and
  (iv) proc(L) appears in $N_0, \ldots, N_{2k}$.

Continuing with our example, we can note that D_n^3 has three b-models:

M_1 satisfying (*VeryColdWeather* $\wedge$ *Broken*) at all time-points,

M_2 satisfying ($\neg$*VeryColdWeather* $\wedge$ *Broken*), and

M_3 satisfying ($\neg$*VeryColdWeather* $\wedge$ $\neg$*Broken*).

At each time-point, each of these b-models has just one, single-node causal chain,[8] so that we need refer only to Condition (1) in Definition 27 when searching for instantaneous causes. In fact at any given time-point T we can identify just one instantaneous cause in just one model — *Broken* has an instantaneous cause at T w.r.t. D_n^3 and M_1.

We can now define what it means for a fluent to have an "abnormal" truth value in an n-domain description:

Definition 28 *(Fluent abnormality).* Let D be an n-domain description, T be a time-point of D, L be a fluent literal, and M be a b-model of D. L is a **fluent abnormality at** T **w.r.t.** D **and** M iff

(1) M satisfies L at T,
(2) "`normally` $\bar{L}$" $\in D$, and
(3) L does not have an instantaneous cause at T w.r.t. D and M.

The set of all fluent abnormalities at T with respect to D and M is written $\mathit{FluentAbs}(D,M,T)$.

As regards the example n-domain description D_n^3, the fluent abnormalities at any time-point $T \in \Pi$ with respect to D_n^3 and each b-model are as follows:

$$\mathit{FluentAbs}(D_n^3, M_1, T) = \emptyset$$
$$\mathit{FluentAbs}(D_n^3, M_2, T) = \{\neg \mathit{VeryColdWeather}, \mathit{Broken}\}$$
$$\mathit{FluentAbs}(D_n^3, M_3, T) = \{\neg \mathit{VeryColdWeather}\}$$

Definition 29 *(N-entailment).* Let D be an n-domain description, and let M and M' be b-models of D. M is **n-preferable to** M' **w.r.t.** D, written $M \prec_D M'$, iff there exists a time-point T_0 such that for every time-point $T' \preccurlyeq T_0$, $\mathit{FluentAbs}(D,M,T') \subset \mathit{FluentAbs}(D,M',T')$. M is an **n-model of** D iff there is no other model M'' of D such that $M'' \prec_D M$. D **n-entails** the h-proposition "ϕ `holds-at` T", written $D \models_n \phi$ `holds-at` T, iff for every n-model M^n of D, M^n satisfies ϕ at T.

In terms of our example, Definition 29 gives $M_1 \prec_{D_n^3} M_3 \prec_{D_n^3} M_2$, so that M_1 is the only n-model, and, for example,

$D_n^3 \models_n$ (*VeryColdWeather* $\wedge$ *Broken*) `holds-at` 0.

We now identify one (wide) class of n-theories for which the semantics of n-propositions simplifies considerably, because it is not necessary to check for instantaneous causes.

Definition 30 *(N-simple domain description).* Let D be an n-domain description. D is **n-simple** iff for every pair of propositions "`normally` L" and "C `causes` L'" in D such that C does not contain a non-negated action constant, $L \neq L'$ and $L \neq \overline{L'}$.

[8] The process successors of the single nodes which comprise these causal chains are $\langle\{\mathit{VeryColdWeather}, \mathit{Broken}\}, \emptyset, \{\uparrow \mathit{Broken}\}\rangle$, $\langle\{\neg \mathit{VeryColdWeather}, \mathit{Broken}\}, \emptyset, \emptyset\rangle$ and $\langle\{\neg \mathit{VeryColdWeather}, \neg \mathit{Broken}\}, \emptyset, \emptyset\rangle$ respectively.

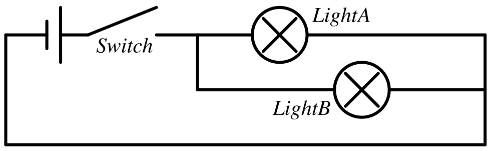

Fig. 6. Circuit with two light bulbs.

For n-simple theories, it follows directly from Definition 27 that it is necessary to check only Conditions (1) and (2) of Definition 28 when identifying fluent abnormalities, and necessary to compare sets of fluent abnormalities only at one arbitrary pre-action point when identifying n-models (see Definition 29).

As an illustration, the domain of Fig. 6 is represented in the following example (Example 13) as an n-simple theory.

Example 13 *(Light bulbs).* Pressing a switch causes two light bulbs connected in parallel to light up, as long as the bulbs are not faulty and the switch is not stuck:

$$\texttt{normally}\ \neg \mathit{SwitchStuck} \tag{LB1}$$

$$\texttt{normally}\ \neg \mathit{FaultyBulbA} \tag{LB2}$$

$$\texttt{normally}\ \neg \mathit{FaultyBulbB} \tag{LB3}$$

$$\mathit{PressDownSwitch}\ \texttt{causes}\ \mathit{ElectricCurrent} \tag{LB4}$$

$$\{\mathit{ElectricCurrent}, \neg \mathit{FaultyBulbA}\}\ \texttt{causes}\ \mathit{LightA} \tag{LB5}$$

$$\{\mathit{ElectricCurrent}, \neg \mathit{FaultyBulbB}\}\ \texttt{causes}\ \mathit{LightB} \tag{LB6}$$

$$\mathit{SwitchStuck}\ \texttt{prevents}\ \mathit{PressDownSwitch} \tag{LB7}$$

$$\texttt{always}\ (\neg \mathit{ElectricCurrent} \rightarrow (\neg \mathit{LightA} \wedge \neg \mathit{LightB})) \tag{LB8}$$

$$\neg \mathit{ElectricCurrent}\ \texttt{holds-at}\ 0 \tag{LB9}$$

$$\mathit{PressDownSwitch}\ \texttt{occurs-at}\ 0 \tag{LB10}$$

Let D_{lb} be the n-simple theory {(LB1), ..., (LB10)}. Assuming time to be represented as the naturals, D_{lb} has eight b-models corresponding to the eight possible combinations of truth values at time 0 for the fluents *SwitchStuck*, *FaultyBulbA* and *FaultyBulbB*. But only the b-model in which each of these is false has no fluent abnormalities, so that this b-model is the only n-model and, for example, $D_{lb} \models_n (\mathit{LightA} \wedge \mathit{LightB})$ `holds-at` 1.

Now consider the following (unexpected) observation:

$$\neg \mathit{LightA}\ \texttt{holds-at}\ 1 \tag{LB11}$$

Let $D^+_{lb} = D_{lb} \cup$ (LB11). D^+_{lb} has six b-models, since (LB11) eliminates interpretations in which both *SwitchStuck* and *FaultyBulbA* are false. Just two of these are also n-models — the b-model in which *SwitchStuck* is true and both *FaultyBulbA* and *FaultyBulbB* are false has a single fluent abnormality *SwitchStuck* at 0, and the b-model in which *FaultyBulbA* is true and both *SwitchStuck* and *FaultyBulbB* are false contains the single fluent abnormality *FaultyBulbA* at 0. Hence we can infer neither "*LightB* `holds-at` 1" nor "¬*LightB* `holds-at` 1" from D^+_{lb}. However, since the two n-models differ on the truth value of *ElectricCurrent* at times after 0, adding the observation "*ElectricCurrent* `holds-at` 1" or the converse "¬*ElectricCurrent* `holds-at` 1" would in each case eliminate one of the n-models, allowing us to infer respectively "*LightB* `holds-at` 1" or "¬*LightB* `holds-at` 1".

Note the similarity of Example 13 to the "life support" example in Section 2.1 (substitute lights A and B for the indicator light and life support machine respectively), which gives rise to anomalous models under the approach in [37]. It is clear that $\mathcal{ME}$ solves the problems identified in Section 2.1 whenever fluents with default values can be explicitly identified in the representation.

The extended semantics of the language $\mathcal{ME}$ as defined above, in order to incorporate default static knowledge in the formalism, is given through a simple minimization of abnormality on top of the base language semantics. This is linked to the free-will property of $\mathcal{ME}$ in the sense that lack of this property would interfere with the default minimization of the static knowledge, causing the simple minimization policy to give anomalous models. Consider the following example that illustrates this.

$$\texttt{normally}\ \mathit{VeryColdWeather} \tag{N1}$$

$$\mathit{Water}\ \texttt{causes}\ \mathit{NewShoots}^{9} \tag{N2}$$

$$\texttt{always}\ \neg(\mathit{VeryColdWeather} \wedge \mathit{NewShoots}) \tag{N3}$$

This domain has two classes of b-models, one where *VeryColdWeather* is true and one where it is false. The default knowledge given by the n-proposition gives the first class as the preferred default class with minimal abnormality. Consider now that an action of *Water* occurs at time 1. If the base framework did not have the free-will property, and in particular if it failed to regard (N3) as providing an extra qualification for (N2), then some of these b-models would be lost. With the addition of the action at time 1 the b-models with *VeryColdWeather* would become inconsistent. As a result we would be left only with the class of b-models where *VeryColdWeather* is false and hence the simple minimization of abnormality on these would not retain the conclusion that *VeryColdWeather* is true. These would be anomalous models as the attempted execution of an action cannot/should not change the default state of affairs of the past. Furthermore if we subsequently observed at time 2 that *NewShoots* was false then in frameworks where the models with the default state of affairs (i.e. models with *VeryColdWeather* true) are lost this observation could not be accounted for.

We end this section with an extension of Theorem 1 and Corollary 1 which covers n-models and n-entailment.

Theorem 3 *(Free will theorem for n-domains). Let D_1 and D_2 be n-domain descriptions and let T_n be a post-observation point for both D_1 and D_2. Let D_1 and D_2 differ only by o-propositions referring to time-points greater than or equal to T_n and let M_1 be an n-model of D_1. Then, there is an n-model M_2 of D_2 such that $M_1(F,T) = M_2(F,T)$ for all fluent constants F and all time-points T such that $T \preccurlyeq T_n$. In particular, for all fluent formulae ϕ and all time-points T such that $T \preccurlyeq T_n$, D_1 n-entails "ϕ `holds-at` T" iff D_2 n-entails "ϕ `holds-at` T".*

Proof. Consider the domains D'_1 and D'_2 obtained by removing all n-propositions from D_1 and D_2, respectively. Clearly, T_n is a post-observation time-point for both D'_1 and D'_2, and these domains differ only by o-propositions referring to time-points greater than or equal to T_n. Partition the models of each of D'_1 and D'_2 into classes, where for each pair of models M'_a, M'_b in the same class, for each fluent F, and for each time-point $T \preccurlyeq T_n$, it holds that $M'_a(F,T) = M'_b(F,T)$. Applying Corollary 1 twice, it follows that there is a bijection between classes of models under D'_1 and classes of models under D'_2.

Consider an n-model M_1 of D_1, and the class of models under D'_1 in which M_1 belongs. Consider the corresponding class of models under D'_2, according to the bijection discussed above, and let M_2 be a model of D'_2 from this class of models. Then, for any fluent F and time-point $T \preccurlyeq T_n$, $M_2(F,T) = M_1(F,T)$. This implies that for every $T' \preccurlyeq T_n$, $\mathit{FluentAbs}(D_2, M_2, T') = \mathit{FluentAbs}(D_1, M_1, T')$, since D_1 and D_2 differ only by o-propositions referring to time-points greater than or equal to T_n.

Assume by way of contradiction that M_2 is not an n-model of D_2. Then, there exists a model M'_2 of D'_2 that is n-preferable to M_2 w.r.t. D_2. Thus, there exists T_0 such that for every $T' \preccurlyeq T_0$, $\mathit{FluentAbs}(D_2, M'_2, T') \subset \mathit{FluentAbs}(D_2, M_2, T')$. Consider the class of models under D'_2 in which M'_2 belongs. Consider the corresponding class of models under D'_1, according to the bijection discussed above, and let M'_1 be a model of D'_1 from this class of models. Then, for any fluent F and time-point $T \preccurlyeq T_n$, $M'_1(F,T) = M'_2(F,T)$. This implies that for every $T' \preccurlyeq T_n$, $\mathit{FluentAbs}(D_1, M'_1, T') = \mathit{FluentAbs}(D_2, M'_2, T')$, since D_1 and D_2 differ only by o-propositions referring to time-points greater than or equal to T_n. Overall, there exists $T_{\min} = \min\{T_0, T_n\}$ such that for every $T' \preccurlyeq T_{\min}$, $\mathit{FluentAbs}(D_1, M'_1, T') \subset \mathit{FluentAbs}(D_1, M_1, T')$; a contradiction to the fact that M_1 is an n-model of D_1. This proves the first claim.

For the second claim consider a time-point $T \preccurlyeq T_n$. Note that D_1 n-entails "ϕ `holds-at` T" iff every n-model of D_1 satisfies ϕ at T. From the first claim, this holds iff every n-model of D_2 satisfies ϕ at T, which then holds iff D_2 n-entails "ϕ `holds-at` T", as required. □

6. Exogenous qualifications

As we have seen, $\mathcal{ME}$'s base semantics offers an elaboration tolerant solution to the endogenous qualification problem, where (general) properties of the domain can implicitly qualify the effect laws. It is, nonetheless, still possible that a domain can lose its meaning when an expected effect (i.e. an effect that belongs to every change set of every model at some time-point) is observed not to materialize. The causal laws have unexpectedly failed. To illustrate such a scenario let us elaborate Example 3 by adding the observation that the car is not running at time 2.

Example 14 *(Extended broken car A).* The key is turned at time 1 but we subsequently observe that the car is not running.

$\{\mathit{TurnKey}\}$ `causes` *Running* (BC1)

TurnKey `occurs-at` 1 (BC2)

$\neg$*Running* `holds-at` 2 (BC-obs2)

No known reason (explicit or implicit) can account for this unexpected observation, and the domain is in fact inconsistent. The failure of the effect law (BC1) needs to be attributed to an *exogenous* reason.

[9] "Shoots" in the gardening sense.

Similarly, an observation contrary to what we expect from a certain effect law may lead us to consider that other effects have failed to be produced, perhaps because an action has failed to be executed properly for some unknown reason. The "life support example" discussed in Section 2.1 illustrates this.

Example 15 *(Life support).* The life support button has been pushed at time 1 but we subsequently observe that the indicator light is not on.

$$\{\mathit{PushButton}\}\ \texttt{causes}\ \mathit{IndicatorLightOn} \tag{LS1}$$

$$\{\mathit{PushButton}\}\ \texttt{causes}\ \mathit{LifeSupportOn} \tag{LS2}$$

$$\mathit{PatientVeryIll}\ \texttt{holds-at}\ 0 \tag{LS3}$$

$$\mathit{PushButton}\ \texttt{occurs-at}\ 1 \tag{LS4}$$

$$\neg\mathit{IndicatorLightOn}\ \texttt{holds-at}\ 2 \tag{LS5}$$

The failure of the first causal law (LS1) could be an isolated failure, or it could be that the *PushButton* action execution (LS4) failed thus resulting in the absence of all the effects associated with this.

A way to reconcile such conflicts (and thus restore elaboration tolerance to the framework) is to assume that every effect law of a domain description is additionally qualified by a set of extra preconditions that encapsulate the normal conditions under which the law operates successfully [9]. These preconditions are outside the base logic's language or *exogenous* [37], and symbolically package together all the unknown conditions necessary for the effect law to successfully generate its effect. In what follows we will make the working (but retractable) assumption that these preconditions are unique and independent for each c-proposition in the domain. We will write them in the form *NormExo*(*Law_Id*) where *Law_Id* is the unique identifier of the c-proposition. Similarly, action executions can be exogenously qualified by assuming that for each action constant A the domain contains a p-proposition $\mathit{AbExo}(A)$ `prevents` A, where the unique new fluent $\mathit{AbExo}(A)$ encapsulates all of the unknown conditions (outside the base logic's language) that would prevent the proper execution of the action A.

The exogenous qualifications *NormExo*(*Law_Id*) are assumed to hold true by default unless the observations in a given narrative make the domain description inconsistent. Similarly, exogenous qualifications of the form $\mathit{AbExo}(A)$ are assumed by default to be false. To formalize this we can simply add the n-propositions

$$\texttt{normally}\ \mathit{NormExo}(\mathit{Law\_Id})$$

$$\texttt{normally}\ \neg\mathit{AbExo}(A)$$

for each *NormExo*(*Law_Id*) and $\mathit{AbExo}(A)$, using the n-entailment semantics as given in Definition 29 in the previous section. The examples above can then be understood via the following transformed n-domain descriptions.

Example 16 *(Extended broken car A with exogenous fluents).* A car key is turned at time 1 but we subsequently observe that the car is not running.

$$\{\mathit{TurnKey}, \mathit{NormExo}(\mathrm{BC1})\}\ \texttt{causes}\ \mathit{Running} \tag{BC1′}$$

$$\mathit{TurnKey}\ \texttt{occurs-at}\ 1 \tag{BC2}$$

$$\neg\mathit{Running}\ \texttt{holds-at}\ 2 \tag{BC-obs2}$$

$$\mathit{AbExo}(\mathit{TurnKey})\ \texttt{prevents}\ \mathit{TurnKey} \tag{BC-ab1}$$

$$\texttt{normally}\ \mathit{NormExo}(\mathrm{BC1}) \tag{BCN1}$$

$$\texttt{normally}\ \neg\mathit{AbExo}(\mathit{TurnKey}) \tag{BCN2}$$

This extended or transformed n-domain description now has models (n-models) according to the semantics given in Section 5, in which *Running* is false everywhere either because *NormExo*(BC1) is false or *AbExo*(*TurnKey*) is true. The exogenous qualifications allow us to recover from the apparent inconsistency introduced by the observation (BC-obs2).

Example 17 *(Life support with exogenous fluents).* The life support button has been pushed at time 1 but we subsequently observe that the indicator light is not on.

$$\{\mathit{PushButton}, \mathit{NormExo}(\mathrm{LS1})\}\ \texttt{causes}\ \mathit{IndicatorLightOn} \tag{LS1'}$$

$$\{\mathit{PushButton}, \mathit{NormExo}(\mathrm{LS2})\}\ \texttt{causes}\ \mathit{LifeSupportOn} \tag{LS2′}$$

$$\mathit{PatientVeryIll}\ \texttt{holds-at}\ 0 \tag{LS3}$$

PushButton `occurs-at` 1 (LS4)

¬*IndicatorLightOn* `holds-at` 2 (LS5)

AbExo(*PushButton*) `prevents` *PushButton* (LS-ab1)

`normally` *NormExo*(LS1) (LSN1)

`normally` *NormExo*(LS2) (LSN2)

`normally` ¬*AbExo*(*PushButton*) (LSN3)

The failure of *IndicatorLightOn* to become true after the action execution at time 1 can now be attributed either to the (abnormal) falsity of *NormExo*(LS1) or to the (abnormal) truth of *AbExo*(*PushButton*). In the latter case this will also mean that *LifeSupportOn* will also fail to become true. But note that the domain n-entails *PatientVeryIll* `holds-at` 2, illustrating that our approach avoids the anomalous model problem exemplified by this same domain in Section 2.1.

To illustrate how this problem of exogenous qualification can interact with the ramification problem and the constraints imposed by a-propositions, consider the following transformation of Example 7 where the domain now is extended with exogenous qualifications in the manner discussed above.

Example 18 *(Broken car E with exogenous fluents).*

{*TurnKey*, *NormExo*(BC1)} `causes` *Running* (BC1′)

TurnKey `occurs-at` 1 (BC2)

`always` ¬(*Broken* ∧ *Running*) (BC3)

{*Break*, *NormExo*(BC4)} `causes` *Broken* (BC4′)

{*Broken*, *NormExo*(BC5)} `causes` ¬*Running* (BC5′)

Break `occurs-at` 1 (BC6)

Running `holds-at` 2 (BC7)

AbExo(*TurnKey*) `prevents` *TurnKey* (BC-ab1)

AbExo(*Break*) `prevents` *Break* (BC-ab2)

`normally` *NormExo*(BC1) (BCN1)

`normally` ¬*AbExo*(*TurnKey*) (BCN2)

`normally` *NormExo*(BC4) (BCN3)

`normally` *NormExo*(BC5) (BCN4)

`normally` ¬*AbExo*(*Break*) (BCN5)

As explained in Section 3.1, the original Example 7 does not have any models. This extended n-domain does however have models (n-models). A preliminary inspection might suggest that we could now attribute the unexpected observation of *Running* at time 2 to abnormal truth values for any of the exogenous fluents *AbExo*(*Break*), *NormExo*(BC4) or *NormExo*(BC5). But in fact, the a-proposition (BC3) ensures that whenever *Running* is true *Broken* is false, which cuts out the possibility that it is *NormExo*(BC5) that has the abnormal truth value. Hence the extended n-domain has only n-models where the car is not broken at any time, and in which *NormExo*(BC5) is true, and either *NormExo*(BC4) is false or *AbExo*(*Break*) is true.

The examples above give an informal indication of part of our approach to exogenous qualification, illustrating how exogenous factors may be introduced via an automatic transformation of the domain into an extended language, hidden from the user. However, once we have introduced such exogenous factors we are faced with the problem of how to reason with them — in effect, how to reason about the unexpected failure of actions. Of course, we have no direct causal information about the exogenous factors, but one interesting question that we can ask is the extent to which they persist. Are the conditions *NormExo*(...) and *AbExo*(...) ordinary fluents (as in the examples above) which therefore persist over time, or is their existence limited to the instant at which they give rise to an unexpected non-effect? In the first case the persistence, for example, of the falsity of a *NormExo*(...) would mean that any subsequent application of the associated effect law would also not give rise to an effect, and thus in the terminology of [37] we have a "failure" of that law. But if the exogenous conditions do not persist we would have only an isolated unexpected non-effect, i.e. an "accident".

In some cases "accidents" will be the only possible explanation for the observations. For instance, if Example 14 is further extended with the o-proposition *TurnKey* `occurs-at` 3 and the h-proposition *Running* `holds-at` 4, we can account for this only by assuming that an exogenous factor prevented the expected application of (BC1) at time 1, but that this factor was no longer present at time 3.

We can capture such isolated exogenous conditions in our language by representing them as (abnormal) exogenous events (actions) that occur at the specific time where the expected effect would otherwise have been generated. Hence as well as introducing exogenous fluents of the form $\mathit{NormExo}(\ldots)$ and $\mathit{AbExo}(\ldots)$, we introduce exogenous (unknown) actions of the form $\mathit{ExoAct}(\ldots)$. We can then qualify our effect laws with the negation (absence) of such actions meaning that, under the normal assumption that no exogenous "blocking" event occurs at the time of interest, the effect law will be successful. For example, the exogenously qualified c-proposition

$$\{\mathit{TurnKey}, \neg\mathit{ExoAct}(\mathrm{BC1})\}\ \texttt{causes}\ \mathit{Running}$$

represents that the action *TurnKey* causes *Running* under the assumption that no exogenous blocking action occurs at the same time. We can exogenously qualify action executability via p-propositions in an analogous way:

$$\top\ \texttt{prevents}\ \{\mathit{PushButton}, \mathit{ExoAct}(\mathit{PushButton})\}$$

means that an execution of *PushButton* cannot be successful at the same time as a (successful) occurrence of its associated exogenous blocking action.

Finally, we may also wish to consider the possibility that the unexpected observation of the converse of a particular fluent literal L indicates that exogenous factors have prevented it from being caused in any way whatsoever. Once again we can indicate this either using persistent exogenous factors via extra p-propositions such as $\mathit{AbExo}(L)$ `prevents` L, or using non-persistent exogenous factors via extra p-propositions such as $\top$ `prevents` $\{L, \mathit{ExoAct}(L)\}$.

The choice of which types of exogenous qualification to consider will to some extent be domain dependent. For example, for some domains it may be appropriate to explain observations of unexpected non-effects only in terms of non-persisting "accidents". For other domains we may wish to regard some effect laws as exogenously qualified "defaults", but others as unbreakable rules. In the next section we formally define a general transformation of domain descriptions which encompasses all the types of exogenous qualification we have described above, for all effect laws, actions and fluents. But it will be obvious to the reader that the definitions could be straightforwardly simplified to restrict attention to certain types of exogenous qualification only.

6.1. Extended semantics for exogenous qualification

We begin by defining the syntactic transformation of a domain description D into its extended form.[10]

Definition 31 *(Default domain description).* Let D be an n-domain description. The **default domain description D_d associated with** D is the n-domain obtained by:

(i) replacing every c-proposition "C `causes` L" in D with the c-proposition "$C \cup \{\mathit{NormExo}(\mathrm{id}), \neg\mathit{ExoAct}(\mathrm{id})\}$ `causes` L", where id is a unique identifier for the c-proposition, and extending the underlying language with the action constant $\mathit{ExoAct}(\mathrm{id})$ and fluent constant $\mathit{NormExo}(\mathrm{id})$,
(ii) extending the underlying language with the action constant $\mathit{ExoAct}(A)$ and fluent constant $\mathit{AbExo}(A)$ for every action constant A, and adding the p-propositions "$\mathit{AbExo}(A)$ `prevents` A" and "$\top$ `prevents` $\{A, \mathit{ExoAct}(A)\}$",
(iii) extending the underlying language with the action constant $\mathit{ExoAct}(L)$ and fluent constant $\mathit{AbExo}(L)$ for every fluent literal L, and adding the p-propositions "$\mathit{AbExo}(L)$ `prevents` L" and "$\top$ `prevents` $\{L, \mathit{ExoAct}(L)\}$",
(iv) adding the n-propositions:
   - "`normally` $\mathit{NormExo}(\mathrm{id})$" for every fluent constant $\mathit{NormExo}(\mathrm{id})$ introduced in step (i),
   - "`normally` $\neg\mathit{AbExo}(A)$" for every fluent constant $\mathit{AbExo}(A)$ introduced in step (ii),
   - "`normally` $\neg\mathit{AbExo}(L)$" for every fluent constant $\mathit{AbExo}(L)$ introduced in step (iii).

We will refer to the extra fluents and actions introduced in steps (i)–(iv) of Definition 31 as the *exogenous fluents and actions* of D and D_d. As an example, the default domain description associated with the domain of Example 14 is as follows.

Example 19 *(Extended broken car A default domain description).*

$$\{\mathit{TurnKey}, \mathit{NormExo}(\mathrm{BC1}), \neg\mathit{ExoAct}(\mathrm{BC1})\}\ \texttt{causes}\ \mathit{Running} \qquad (\mathrm{BC1}_d)$$

$$\mathit{TurnKey}\ \texttt{occurs-at}\ 1 \qquad (\mathrm{BC2})$$

$$\neg\mathit{Running}\ \texttt{holds-at}\ 2 \qquad (\mathrm{BC\text{-}obs2})$$

$$\mathit{AbExo}(\mathit{TurnKey})\ \texttt{prevents}\ \mathit{TurnKey} \qquad (\mathrm{BC\text{-}p1})$$

[10] A further possibility for exogenous qualification is to introduce for every fluent literal L a new a-proposition of the form `always` $\neg(L \wedge \mathit{Abn}(L))$ with `normally` $\neg\mathit{Abn}(L)$. Then under the unknown exogenous condition $\mathit{Abn}(L)$ the fluent L cannot become true in any state but, unlike the case of p-propositions, it is allowed to be transiently true within the process of generating a change set. We will not discuss this further in this paper.

$\top$ `prevents` $\{\mathit{TurnKey}, \mathit{ExoAct}(\mathit{TurnKey})\}$ (BC-p2)

$\mathit{AbExo}(\mathit{Running})$ `prevents` $\mathit{Running}$ (BC-p3)

$\top$ `prevents` $\{\mathit{Running}, \mathit{ExoAct}(\mathit{Running})\}$ (BC-p4)

$\mathit{AbExo}(\neg \mathit{Running})$ `prevents` $\neg \mathit{Running}$ (BC-p5)

$\top$ `prevents` $\{\neg \mathit{Running}, \mathit{ExoAct}(\neg \mathit{Running})\}$ (BC-p6)

`normally` $\mathit{NormExo}(\mathrm{BC1})$ (BC-n1)

`normally` $\neg \mathit{AbExo}(\mathit{TurnKey})$ (BC-n2)

`normally` $\neg \mathit{AbExo}(\mathit{Running})$ (BC-n3)

`normally` $\neg \mathit{AbExo}(\neg \mathit{Running})$ (BC-n4)

The semantics of a domain description D can then be defined in terms of the models of the associated n-domain D_d, once we have also formalized the default interpretation of extra exogenous action occurrences. For this we can introduce the notion of an *e-model* as follows.

Definition 32 *(Exogenous action occurrence).* Let D be an n-domain description. An **exogenous action occurrence** of D is an o-proposition of the form A `occurs-at` T, where A is an exogenous action of D.

Definition 33 *(E-model).* Let D be an n-domain description, let D_d be the default domain associated with D and let H be an interpretation of D_d. Then H is an **e-model** of D iff there exists a set O of exogenous action occurrences of D such that H is an n-model of $D_d \cup O$.

Definition 34 *(Event abnormality).* Let D be an n-domain description, D_d be the default domain associated with D, T be a time-point, A be an exogenous action constant of D_d, and M be an e-model of D. $\langle A, T\rangle$ is an **event abnormality w.r.t.** D **and** M iff $M(A, T) = \text{true}$. The set of all event abnormalities w.r.t. D and M is written $\mathit{EventAbs}(D, M)$.

Definition 35 *(Exogenous abnormalities).* Let D and D_d be an n-domain and its associated default domain, let M be an e-model of D, let O be the (unique) set of exogenous action occurrences such that M is an n-model of $D_d \cup O$, and let T be a time-point. The set $\mathit{ExoFluAbs}(D, M, T)$ is defined as the set of exogenous fluent literals in $\mathit{FluentAbs}(D_d \cup O, M, T)$. The set $\mathit{ExogAbs}(D, M, T)$ is defined as $\mathit{ExoFluAbs}(D, M, T) \cup \mathit{EventAbs}(D, M)$.

The extended semantics of $\mathcal{ME}$ is given in terms of *default models*. A default model of a domain description D is given by first minimizing all fluent abnormalities (just as for n-entailment in Section 5), which is done implicitly in the definition of e-models in terms of n-models. From this set, further minimization selects those e-models of D with the fewest event and exogenous fluent abnormalities. In this way the semantics gives preference to explanations of unexpected observations which involve only abnormal truth values of known fluents, as opposed to explanations involving abnormal exogenous factors. The formal definition is as follows. Note that for n-consistent domains d-entailment is equivalent to n-entailment.

Definition 36 *(D-entailment).* Let D be an n-domain description and let M and M' be e-models of D. M is **d-preferable to** M' **w.r.t.** D, written $M \prec_D^d M'$, iff there exists a time-point T_0 such that for every time-point $T' \preccurlyeq T_0$, $\mathit{ExogAbs}(D, M, T') \subset \mathit{ExogAbs}(D, M', T')$. M is a **d-model of** D iff there is no other model M'' such that $M'' \prec_D^d M$. D **d-entails** the h-proposition "ϕ `holds-at` T", written $D \models_d \phi$ `holds-at` T, iff for every d-model M^d of D, M^d satisfies ϕ at T.

As regards Example 14 and its transformation into Example 19 we can see that Definition 36 gives rise to two types of d-models. In d-models of the first type, there are no exogenous action occurrences, and exactly one of the exogenous fluents $\mathit{NormExo}(\mathrm{BC1})$, $\mathit{AbExo}(\mathit{Running})$ or $\mathit{AbExo}(\mathit{TurnKey})$ has an abnormal truth value at time 1 (and therefore at all other times by persistence). For these d-models adding a later occurrence of *TurnKey*, e.g. "*TurnKey* `occurs-at` 3", will make no difference to the truth of *Running* at e.g. time 4, because the same persisting exogenous factor will re-prevent *Running* being initiated. In d-models of the second type, all exogenous fluents have their normal truth values, but exactly one of the exogenous actions $\mathit{ExoAct}(\mathrm{BC1})$, $\mathit{ExoAct}(\mathit{Running})$ or $\mathit{ExoAct}(\mathit{TurnKey})$ is true (i.e. occurs) at time 1. For these d-models adding "*TurnKey* `occurs-at` 3", will make *Running* true at times after 3. So in this way, i.e. by implicitly qualifying with both exogenous fluents and exogenous actions, we are able to represent domains in which, after observing an unexpected non-effect, we are unable to decide whether subsequent identical attempts to produce the effect will actually produce the effect.

Definition 31 of an associated default domain results in each fluent literal being independently exogenously qualified via a pair of p-propositions. This allows us to capture the possibility that an observed apparent failure of one effect law for the literal might be due to reasons not linked specifically to that law, and therefore that subsequent attempts to generate the literal via other causal laws will also not produce their normal effect. To illustrate this consider the following example.

Example 20 *(Jump start car).*

TurnKey `causes` *Running* (JS1)

JumpStart `causes` *Running* (JS2)

TurnKey `occurs-at` 1 (JS3)

¬*Running* `holds-at` 2 (JS4)

JumpStart `occurs-at` 3 (JS5)

The associated default domain of this example introduces exogenous fluents including *NormExo*(JS1), *AbExo*(*TurnKey*) and *AbExo*(*Running*) and exogenous actions including *ExoAct*(JS1), *ExoAct*(*TurnKey*) and *ExoAct*(*Running*). The unexpected observation (JS4) ensures that in each of the six d-models of the domain exactly one of these six exogenous factors has an abnormal truth value at time 1. In five of the d-models *Running* becomes true at times after 3 by virtue of the combination of (JS2) and (JS5). But in the d-model in which *AbExo*(*Running*) is true at time 1, the p-proposition "*AbExo*(*Running*) `prevents` *Running*" included in the associated default domain blocks the normal initiating effect of (JS2) at time 3, because *AbExo*(*Running*) persists from 1 to 3.

Default models allow us to make sense of a wider class of domain descriptions while maintaining elaboration tolerance and modularity. They do this by essentially weakening the effect laws and thus reducing the change that they would bring about. But what kind of domain descriptions do not have a default model? Apart from the obvious cases where observations at a single time-point are inconsistent with the a-propositions, inconsistent domain descriptions are those where an observed change in some fluent cannot be accounted for in terms of known action occurrences, as the following example illustrates.

Example 21 *(Engine noise).*

{*TurnKey*} `causes` *Running* (EN1)

{*Running*} `causes` *EngineNoise* (EN2)

¬*Running* `holds-at` 0 (EN3)

¬*EngineNoise* `holds-at` 0 (EN4)

EngineNoise `holds-at` 2 (EN5)

For any time-point in $[0, 2)$ (and any initial state at time 0) the change set for any interpretation that satisfies the observations at time 0 and the condition of persistence of ¬*Running* is empty. Hence the observed change in the truth value of *EngineNoise* cannot be accounted for. In fact, ¬*EngineNoise* must persist and this is incompatible with the observation that *EngineNoise* is true at time 2. Note that, as the change sets are empty, employing d-models (which as we have seen essentially reduce the change sets) cannot help us in this example.

Now suppose that we add *TurnKey* `occurs-at` 1 to Example 21. The extended domain has an n-model and hence also a d-model. But then suppose we also add ¬*Running* `holds-at` 2. This domain will again have no d-model. This is because any exogenous qualification introduced to explain the absence of an initiating cause for *Running* also blocks the initiation of *EngineNoise*. This elaborated example shows yet again the intricate link between the ramification and qualification problems — exogenous qualifications forced by unexpected observations sometimes break the causality chain associated with the effect laws.

Clearly, for such domains we need to allow models which do not event match the domain description (see Definition 16), but instead indicate additional event occurrences. The standard example for this type of domain is Kautz's "Stolen Car Problem" [18]. The problem of assuming or abducing new event occurrences has of course been extensively studied, especially in the context of planning (see e.g. [7,18,29,34]). In particular, a recent study [1] has investigated how to reason with domains in which such existentially quantified (partially ordered) events have been added.

We end this section with an extension of Theorem 1 and Corollary 1 which covers d-models and d-entailment.

Theorem 4 *(Free will theorem for D-entailment). Let D_1 and D_2 be n-domain descriptions over the same underlying language and let T_n be a post-observation point for both D_1 and D_2. Let D_1 and D_2 differ only by o-propositions referring to time-points greater than or equal to T_n and let M_1 be a d-model of D_1. Then, there is a d-model M_2 of D_2 such that $M_1(F, T) = M_2(F, T)$ for all fluent constants F and all time-points T such that $T \preccurlyeq T_n$. In particular, for all fluent formulae ϕ and all time-points T such that $T \preccurlyeq T_n$, D_1 d-entails "ϕ `holds-at` T" iff D_2 d-entails "ϕ `holds-at` T".*

Proof. Consider the default domain descriptions D_{1d} and D_{2d} associated with D_1 and D_2, respectively. Consider also any set O of exogenous action occurrences of D_1 and D_2; since the two domains are over the same underlying language, they have the same possible exogenous action occurrences. Clearly, T_n is a post-observation time-point for both $D_{1d} \cup O$ and $D_{2d} \cup O$, and these n-domains differ only by o-propositions referring to time-points greater than or equal to T_n. Partition the n-models of each of $D_{1d} \cup O$ and $D_{2d} \cup O$ into classes, where for each pair of n-models M_a, M_b in the same class, for each fluent F, and for each time-point $T \preccurlyeq T_n$, it holds that $M_a(F,T) = M_b(F,T)$. Applying Theorem 3 twice, it follows that there is a bijection between classes of n-models under $D_{1d} \cup O$ and classes of n-models under $D_{2d} \cup O$.

Consider a d-model M_1 of D_1, and the class of n-models under $D_{1d} \cup O$ (for the unique set O implied by M_1) in which M_1 belongs. Consider the corresponding class of n-models under $D_{2d} \cup O$, according to the bijection discussed above, and let M_2 be an n-model of $D_{2d} \cup O$ from this class of n-models. Then, for any fluent F and time-point $T \preccurlyeq T_n$, $M_2(F,T) = M_1(F,T)$. This implies that for every $T' \preccurlyeq T_n$, $\mathit{ExoFluAbs}(D_2, M_2, T') = \mathit{ExoFluAbs}(D_1, M_1, T')$, since D_1 and D_2 differ only by o-propositions referring to time-points greater than or equal to T_n. Since it also holds that $\mathit{EventAbs}(D_2, M_2) = \mathit{EventAbs}(D_1, M_1)$, it follows that for every $T' \preccurlyeq T_n$, $\mathit{ExogAbs}(D_2, M_2, T') = \mathit{ExogAbs}(D_1, M_1, T')$.

Assume by way of contradiction that M_2 is not a d-model of D_2. Then, there exists a set O' of exogenous action occurrences of D_1 and D_2, and an n-model M_2' of $D_{2d} \cup O'$ that is d-preferable to M_2 w.r.t. D_2. Thus, there exists T_0 such that for every $T' \preccurlyeq T_0$, $\mathit{ExogAbs}(D_2, M_2', T') \subset \mathit{ExogAbs}(D_2, M_2, T')$. Consider the class of models under $D_{2d} \cup O'$ in which M_2' belongs. Consider the corresponding class of models under $D_{1d} \cup O'$, according to the bijection discussed above, and let M_1' be a model of $D_{1d} \cup O'$ from this class of models. Then, for any fluent F and time-point $T \preccurlyeq T_n$, $M_1'(F,T) = M_2'(F,T)$. This implies that for every $T' \preccurlyeq T_n$, $\mathit{ExoFluAbs}(D_1, M_1', T') = \mathit{ExoFluAbs}(D_2, M_2', T')$, since D_1 and D_2 differ only by o-propositions referring to time-points greater than or equal to T_n. Since it also holds that $\mathit{EventAbs}(D_1, M_1') = \mathit{EventAbs}(D_2, M_2')$, it follows that for every $T' \preccurlyeq T_n$, $\mathit{ExogAbs}(D_1, M_1', T') = \mathit{ExogAbs}(D_2, M_2', T')$. Overall, there exists $T_{\min} = \min\{T_0, T_n\}$ such that for every $T' \preccurlyeq T_{\min}$, $\mathit{ExogAbs}(D_1, M_1', T') \subset \mathit{ExogAbs}(D_1, M_1, T')$; a contradiction to the fact that M_1 is a d-model of D_1. This proves the first claim.

For the second claim consider a time-point $T \preccurlyeq T_n$. Note that D_1 d-entails "ϕ `holds-at` T" iff every d-model of D_1 satisfies ϕ at T. From the first claim, this holds iff every d-model of D_2 satisfies ϕ at T, which then holds iff D_2 d-entails "ϕ `holds-at` T", as required. □

6.2. Failure associations

Reasoning about the failure of actions and effect laws can involve problems orthogonal to the issue of the persistence of failure. One such problem is that of *failure associations* — the possibility that failures of effects are linked to each other. We might for example know that if one effect law fails then another (associated) effect law will also fail, or more generally that all other effect laws sharing the same set of preconditions will also fail. We have also seen some examples of failure associations in the previous section. For instance, an association between the failure of all possible effects of an action A can be provided by a p-proposition such as $\mathit{AbExo}(A)$ `prevents` A.

But this knowledge of failure associations might be domain dependent and highly specific. In this section we briefly and informally explore how such information might be expressed in a domain description in a modular way. For brevity, and to minimize the introduction of new syntax, we assume here for the purposes of discussion that the user is able to refer to exogenous fluents directly.

We can capture domain specific failure associations by enriching our language for n-domains (and hence also for the associated default domains) to allow n-propositions that link together the exogenous qualification conditions of the effect laws. These new n-propositions are of the form "`normally` ϕ" where ϕ is a propositional formula on the fluent language extended with exogenous fluents.[11] In addition, ϕ may be a formula expressing that under certain conditions a fluent literal L_1 has priority over L_2 via the special binary predicate $L_1 > L_2$. The following example illustrates this enrichment of the language.

Example 22 *(Broken car with failure associations).*

$\mathit{TurnKey}$ `causes` $\mathit{Running}$ (FA1)

$\mathit{TurnKey}$ `causes` $\mathit{DashBoardLights}$ (FA2)

$\mathit{Running}$ `causes` $\mathit{EngineNoise}$ (FA3)

`normally` $\neg\mathit{NormExo}(\text{FA2}) \rightarrow \neg\mathit{NormExo}(\text{FA1})$ (FA4)

`normally` $\mathit{PetrolCar} \rightarrow (\neg\mathit{NormExo}(\text{FA1}) > \neg\mathit{NormExo}(\text{FA3}))$ (FA5)

`normally` $\mathit{PetrolCar}$ (FA6)

$\mathit{TurnKey}$ `occurs-at` 1 (FA7)

$\neg\mathit{Running}$ `holds-at` 0 (FA8)

[11] Here for simplicity we will assume that effect laws are exogenously qualified only by exogenous fluents.

Here (FA4) expresses the domain dependent knowledge that (normally) “failure of (FA2) also means failure of (FA1)”. Hence if we subsequently observe that *DashBoardLights* is false at some time after 1 then (FA4) would lead us to also conclude that *Running* is false. This is captured by a preference of models that satisfy this association rule over those which do not and hence, since *NormExo*(FA2) must be false in any b-model, only models with *NormExo*(FA1) also false can be default models. Similarly, (FA5) states that “for petrol cars it is more likely for (FA1) to fail than (FA3)”. If we observe ¬*EngineNoise* at 2 then, as cars in this domain are normally petrol cars (FA6), this priority will have the effect that only b-models that make *NormExo*(FA1) false can be default models, as these are preferred over b-models where *NormExo*(FA3) is false (thus capturing the preference of failure of (FA1) over (FA3)).

Failure associations are expressed using n- rather than a-propositions because we want to capture the default nature of these failure links — they are defeasible when the observations present evidence to the contrary of the associations. In our example above, if we observe both that *DashBoardLights* is false and that *Running* is true after time 1 then the only b-models possible have *NormExo*(FA2) false and *NormExo*(FA1) true, and hence in the default models of our theory the failure association rule (FA4) is not satisfied.

The semantics of n-propositions can be extended to capture this intended meaning by re-interpreting the default semantics for n-domains presented in the previous sections in terms of argumentation where, for example, an n-proposition `normally` L is understood as a stronger argument for L over its converse $\bar{L}$. We can employ known methods (see e.g. [2,17,31]) for formalizing preference reasoning through argumentation and the comparison of arguments to suitably extend the notion of a default model for domains with such n-propositions. The formal details are beyond the scope of this paper, but it is important to note that the semantics can be developed in a modular way, so that the default reasoning about persistence is separated from the domain dependent default reasoning about the n-propositions for failure association rules, and indeed separated from any other default rules known about the particular domain at hand.

7. Summary and related and future work

We have shown how $\mathcal{ME}$ can represent non-deterministic narrative domains involving concurrency, static laws and indirect effects. We have formally characterized $\mathcal{ME}$'s high degree of modularity and elaboration tolerance, enabled by an exceptionally full solution to the ramification problem able to deal with looping systems of indirect effects, and race conditions between competing causal laws. These properties help separate out, and provide principled solutions to, the endogenous and exogenous qualification problems. Endogenous qualifications may be either locally specified or globally derived within the base semantics, whereas exogenous qualifications are provided by the use of default minimization.

Our approach to the qualification problem and its links to ramifications partly follows that in [37]. But $\mathcal{ME}$'s fuller solution to the frame problem, which covers both successful and failed action attempts, enables it to use the same default reasoning mechanism to deal with not just the “weak” but also the “strong” qualification problem as described in [37]. Two other important aspects in which $\mathcal{ME}$ differs from [37] are (a) the more complete treatment of ramifications, e.g., for concurrent effects and (b) the notion of global qualification which gives $\mathcal{ME}$ a higher degree of modularity. These results are in line with the recent study of modularity in [13] which again highlights the link between modularity and free-will properties.

Kvarnström and Doherty provide a detailed discussion and solution in [20] to more-or-less what we have termed the endogenous qualification problem. Comparison with $\mathcal{ME}$ is somewhat difficult, however, as the authors specifically restrict their solution to “off-line planning and prediction” problems, and state that their formalism is not necessarily “able to conclude that an action was qualified because its successful execution would have contradicted a [later] observation”, which is of primary concern here. Nevertheless, Kvarnström and Doherty, like us, argue the need for a modular and elaboration tolerant approach, and their formalism is able to model a wide range of phenomena, including actions with duration and non-boolean valued fluents (for which $\mathcal{ME}$ currently lacks the necessary expressivity). The paper also includes an interesting collection of benchmark problems and a comprehensive analysis of related work.

Our solution to the qualification problem should be translatable to any action calculus expressive enough to represent a narrative of events and observations in “real time” and which exhibits (an equivalent of) the “Free Will” property described in Section 4. The (non-deterministic version of the) Event Calculus in [28] is one such example, although it is less expressive than $\mathcal{ME}$ in many respects (it has no equivalent of a-propositions, and cannot be used to describe indirect effects or ramifications). To be useful predictive tools, such action calculi also need to adhere to, or be augmented with, the principle that “failed” action attempts are ineffectual, with a solution to the frame problem that covers such scenarios, as discussed in Section 2.1.

The solution to the ramification problem that we have proposed is related to that in [3] in that the indirect effects of actions are defined constructively through causal laws. But $\mathcal{ME}$'s process-based semantics differs in that it (a) embraces non-determinism resulting from the possible orderings by which effects are realized, and (b) attributes meaning to domains (e.g., Example 11) that are deemed ill-formed in [3].

Irrespective of the qualification problem, the “free will” property of Theorem 1 is important to avoid *anomalous planning*, whereby unintended “plans” can be abduced or deduced for the converse of a precondition of an effect law by virtue of a lack of model for the successful application of that law. (See [28] for an example.) Although lack of space prevents an illustration here, anomalous plans are easy to construct in formalisms such as the Language $\mathcal{C}+$ [10] which express action non-executability in terms of inconsistency. But they also arise in any framework (such as [16]) unable to provide models for all combinations of causal laws.

We currently have a prototype implementation of $\mathcal{ME}$'s base semantics in Prolog. The declarative programming style should facilitate an easy proof of the soundness and completeness of the implementation w.r.t. to $\mathcal{ME}$'s semantics. On the other hand, different techniques might be needed to address the *computational qualification problem* [6] of avoiding considering the majority of qualifications during the computation. $\mathcal{ME}$ has a sound basis on which efficient reasoning techniques might be built, namely its free-will property. When seen in the context of a prediction task, the free-will property amounts to some sort of memory. An action execution does not affect the action's past, thus one need only consider the current state of affairs when computing the effects of an action. This allows a prediction engine to cache-in knowledge as it reasons forward in time after each action execution, obviating the need to recompute everything from scratch. Similar techniques might also prove useful when computing default models in explanation tasks, where one does not want to consider all possible ways causal laws might fail, but rather deduce which ones should fail. Things, however, are less clear in this setting, since the free-will property no longer holds as stated. An action attempt that is observed to fail may necessitate the reconsideration of assumptions on the values of certain default fluents, which in turn may affect the execution of past actions. Thus, when unexpected observations are encountered, more elaborate reasoning might be required. To the end of building an efficient computational model for $\mathcal{ME}$ we are currently considering the use of satisfiability methods or Answer Set Programming (along the lines of [4,19,35]), as well as argumentation (or abduction) based computational methods. We also aim to study subclasses (as in [4]) of $\mathcal{ME}$, where the computational complexity of reasoning decreases.

There are several aspects of $\mathcal{ME}$ that deserve further study. One is the extent to which static laws should be regarded as specific to the temporal granularity of the representation (how would we refine the role that a-propositions play in computing indirect effects?). Another, as discussed in Section 6.2, is how to extend the syntax and semantics to allow for succinct expression of domain dependent failure associations between causal laws. This requires us to study how to integrate in $\mathcal{ME}$ default static information that is more general than simply positive or negative facts, a problem that is of great interest on its own accord and which we are proposing to address with argumentation. A third issue is how to lift our approach to the first-order case while preserving properties such as the "Free Will Theorem" of Section 4. This would not be entirely straightforward because, for example, in the non-finite case it would not necessarily be the case that causal chains always terminate or loop back to an earlier state. However, the introduction of non-boolean fluents and a limited form of quantification over finite value sets in the manner of [10] should be unproblematic in $\mathcal{ME}$.

Acknowledgements

Many thanks to the anonymous reviewers for giving the originally submitted version of this paper such a careful and detailed reading, and for the many helpful and insightful suggestions that helped improve it. This work was partially supported by the IST program of the EC, FET under the IST-2001-32530 SOCS project, within the Global Computing proactive initiative.

References

[1] A. Bracciali, A.C. Kakas, Frame consistency: Computing with causal explanations, in: Proceedings of the 10th International Workshop on Non-Monotonic Reasoning (NMR'04), 2004, pp. 79–87.
[2] G. Brewka, T. Eiter, Preferred answer sets for extended logic programs, Artificial Intelligence 109 (1–2) (1999) 297–356.
[3] M. Denecker, D. Theseider-Dupré, K. Van Belleghem, An inductive definition approach to ramifications, Electronic Transactions on Artificial Intelligence 2 (1–2) (1998) 25–67.
[4] Y. Dimopoulos, A. Kakas, L. Michael, Reasoning about actions and change in answer set programming, in: Proceedings of the 7th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'04), 2004, pp. 61–73.
[5] P. Doherty, J. Gustafsson, L. Karlsson, J. Kvarnström, TAL: Temporal action logics language specification and tutorial, Electronic Transactions on Artificial Intelligence 2 (3–4) (1998) 273–306.
[6] C. Elkan, On solving the qualification problem, in: Working Notes of the AAAI Spring Symposium on Extending Theories of Actions: Formal Theory and Practical Applications, 1995, pp. 77–79.
[7] K. Eshghi, Abductive planning with event calculus, in: Proceedings of the 5th International Conference and Symposium on Logic Programming (ICLP/SLP'88), 1988, pp. 562–579.
[8] M. Gelfond, V. Lifschitz, Representing actions in extended logic programming, in: Proceedings of the Joint International Conference and Symposium on Logic Programming (JICSLP'92), 1992, pp. 559–573.
[9] M. Ginsberg, D. Smith, Reasoning about action II: The qualification problem, Artificial Intelligence 35 (3) (1988) 311–342.
[10] E. Giunchiglia, J. Lee, V. Lifschitz, N. McCain, H. Turner, Nonmonotonic causal theories, Artificial Intelligence 153 (1–2) (2004) 49–104.
[11] S. Hanks, D. McDermott, Nonmonotonic logic and temporal projection, Artificial Intelligence 33 (3) (1987) 379–412.
[12] D. Harel, Dynamic logic, in: D. Gabbay, F. Guenther (Eds.), Handbook of Philosophical Logic, vol. II. Extensions of Classical Logic, D. Reidel Publishing Company, Dordrecht, The Netherlands, 1984, pp. 497–604.
[13] A. Herzig, I. Varzinczak, Domain descriptions should be modular, in: Proceedings of the 16th European Conference on Artificial Intelligence (ECAI'04), 2004, pp. 348–352.
[14] A. Kakas, L. Michael, R. Miller, Modular-E and the role of elaboration tolerance in solving the qualification problem, Technical annex, http://www.ucl.ac.uk/slais/research/modular-e/, 2006.
[15] A. Kakas, R. Miller, A simple declarative language for describing narratives with actions, Journal of Logic Programming 31 (1–3) (1997) 157–200.
[16] A. Kakas, R. Miller, Reasoning about actions, narratives and ramification, Electronic Transactions on Artificial Intelligence 1 (4) (1998) 39–72.
[17] A.C. Kakas, P. Mancarella, P.M. Dung, The acceptability semantics for logic programs, in: Proceedings of the 11th International Conference on Logic Programming (ICLP'94), 1994, pp. 504–519.

[18] H. Kautz, The logic of persistence, in: Proceedings of the 5th National Conference on Artificial Intelligence (AAAI'86), 1986, pp. 401–405.
[19] H. Kautz, B. Selman, Pushing the envelope: Planning, propositional logic, and stochastic search, in: Proceedings of the 13th National Conference on Artificial Intelligence (AAAI'96), 1996, pp. 1194–1201.
[20] J. Kvarnström, P. Doherty, Tackling the qualification problem using fluent dependency constraints, Computational Intelligence 16 (2) (2000) 169–209.
[21] V. Lifschitz, Missionaries and cannibals in the Causal Calculator, in: Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR'00), 2000, pp. 85–96.
[22] J. McCarthy, Epistemological problems of Artificial Intelligence, in: Proceedings of the 5th International Joint Conference on Artificial Intelligence (IJCAI'77), 1977, pp. 1038–1044.
[23] J. McCarthy, Circumscription — A form of non-monotonic reasoning, Artificial Intelligence 13 (1–2) (1980) 27–39.
[24] J. McCarthy, Elaboration tolerance, http://www-formal.stanford.edu/jmc/elaboration/, 1999.
[25] J. McCarthy, Appearance and reality, http://www-formal.stanford.edu/jmc/appearance.html, 2006.
[26] J. McCarthy, P. Hayes, Some philosophical problems from the standpoint of Artificial Intelligence, Machine Intelligence 4 (1969) 463–502.
[27] S. McIlraith, T. Son, H. Zeng, Semantic web services, IEEE Intelligent Systems (Special Issue on the Semantic Web) 16 (2) (2001) 46–53.
[28] R. Miller, M. Shanahan, Some alternative formulations of the Event Calculus, in: Lecture Notes in Artificial Intelligence, vol. 2408, 2002, pp. 452–490.
[29] L. Missiaen, M. Bruynooghe, M. Denecker, CHICA, a planning system based on the Event Calculus, Journal of Logic and Computation 5 (5) (1995) 579–602.
[30] E. Mueller, Commonsense Reasoning, Morgan Kaufmann, 2006.
[31] H. Prakken, G. Sartor, A system for defeasible argumentation, with defeasible priorities, in: Proceedings of the International Conference on Formal and Applied Practical Reasoning (FAPR'96), 1996, pp. 510–524.
[32] R. Reiter, The frame problem in the Situation Calculus: A simple solution (sometimes) and a completeness result for goal regression, in: Vladimir Lifschitz (Ed.), Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy, Academic Press, San Diego, CA, 1991, pp. 359–380.
[33] A. Russo, R. Miller, B. Nuseibeh, J. Kramer, An abductive approach for analysing event-based requirements specifications, in: Proceedings of the 18th International Conference on Logic Programming (ICLP'02), 2002, pp. 22–37.
[34] M. Shanahan, An abductive Event Calculus planner, Journal of Logic Programming 44 (1–3) (2000) 207–240.
[35] M. Shanahan, M. Witkowski, Event Calculus planning through satisfiability, Journal of Logic and Computation 14 (5) (2004) 731–745.
[36] M. Thielscher, Introduction to the fluent calculus, Electronic Transactions on Artificial Intelligence 2 (3–4) (1998) 179–192.
[37] M. Thielscher, The qualification problem: A solution to the problem of anomalous models, Artificial Intelligence 131 (1–2) (2001) 1–37.
[38] N. Tran, C. Baral, C. Shankland, Issues in reasoning about interaction networks in cells: Necessity of event ordering knowledge, in: Proceedings of the 20th National Conference on Artificial Intelligence (AAAI'05), 2005, pp. 676–681.