Ramifications: An Extension and Correspondence Result for the Event Calculus

JEREMY FORTH, *Department of Computing, Imperial College London, 180 Queen's Gate, London SW7 2AZ, UK.*
E-mail: jforth@doc.ic.ac.uk

ROB MILLER, *S.L.A.I.S. University College London, Gower Street, London WC1E 6BT, UK.*
E-mail: rsm@ucl.ac.uk

Abstract

Classical logic Event Calculus, and the special purpose logical action language $\mathcal{E}$, are both well established formalisms for describing actions and change. However, there is yet to be an account of ramifications in Event Calculus sufficiently general to represent the classes of domains expressible in $\mathcal{E}$. Indeed, an adequately general ramification theory constructed in any general purpose logical language still awaits. Therefore, under the motivation of creating a flexible ramification theory in a universal language, suitable for integration into a rich action theory, a new enhanced version of classical logic Event Calculus named EC-R is proposed. EC-R supports representation and reasoning about domains containing ramifications for classes of domains more general than those possible under previous general purpose language formulations.

This article makes two main contributions. The first, EC-R, is a narrative-based action formalism able to represent concurrent events, non-deterministic actions and indirect causal effects by virtue of an integrated solution to the frame and ramification problems. The formalism can reason about significant subclasses of domains containing both mutually interacting effects and cyclic causal dependencies. The formalism is elaboration tolerant and may be integrated with the standard variants of the Event Calculus. The second contribution is the definition of a semantic mapping between EC-R and $\mathcal{E}$, and a proof of soundness and completeness of the EC-R theory with respect to $\mathcal{E}$'s model theoretic specification.

Keywords: Event calculus, ramifications, action language, narrative.

1 Introduction

This article develops a new enhanced version of classical Logic Event Calculus (EC-R) which is able to represent and reason about wide classes of domains involving indirect causal effects (ramifications). An equivalence is then shown between a particular instantiation of EC-R and the action language $\mathcal{E}$'s semantic specification. EC-R generalizes the ramifications formalism in [1] by accepting domains with concurrent and non-deterministic actions and cyclic fluent causal dependencies such as the Gearwheel domain (shown in an example later). We have used $\mathcal{E}$ as a high level semantic specification for EC-R, and we thereby build upon the work on action languages in [30] and [32] as directly as possible.

EC-R is a modular enhancement over previous versions of the Event Calculus, and may be instantiated with several existing axiom sets designed for particular domain classes. The soundness and completeness results for EC-R given subsequently do not limit EC-R's application to only those domains expressible in $\mathcal{E}$—EC-R is written in standard classical logic and this gives it an advantage in terms of ease of modification and extendibility.

Vol. 17 No. 4, © The Author, 2007. Published by Oxford University Press. All rights reserved.
For Permissions, please email: journals.permissions@oxfordjournals.org
Published online 4 June 2007 doi:10.1093/logcom/exm018

Event Calculus narrative-based principles for reasoning about action have been expressed in a number of different logical frameworks. The original implementation [33] is formalized in logic programming. A classical logic-based EC has been developed by [56, 57], while [31] developed an argumentation-based variant. All versions of the formalism are connected by the same ontology, in which events act upon fluents within an independently defined time structure. However, default maintenance of persistence is achieved using different styles of non-monotonic reasoning: negation as failure, circumscription and argumentation, respectively. The notion of an 'Event' is a general label for an occurrence of change, which may be due to an action performed, external stimulus or by chance.

Language $\mathcal{E}$, introduced in [29], and [30], is a specialized custom propositional logical language designed for the purpose of representing and reasoning about actions and change for a general class of causal ramification domain. It is based on an Event Calculus style narrative ontology of events (or actions), fluents and a time line. It gains its value through the succinct representation of knowledge about such domains, and a simple intuitively clear high-level semantics. While this 'action language' methodology does not hold well for all domain classes, and tends to break down for more difficult problems, it has proved useful for the classes of ramification domains discussed here. See [20, 38] for discussion about action language motivation and design.

Ramifications are regarded here in a similar way to that proposed originally by Finger in [16] as the implicit effects of actions. Finger characterized the ramification problem as being a potentially 'unbounded number of post-requisites', or an unbounded number of consequential effects of an action. The wish is to be able to reason about such a large number of implicit effects of an action without the requirement for either the domain axiomatization or the reasoning system to enumerate them exhaustively. The means of defining explicitly only the most immediate effects of an action and efficiently defining secondary effects through constraints may form the essence of a solution to the ramification problem, provided that the design places no requirement on the reasoning system to enumerate any more implicit effects than are needed for the immediate reasoning task.

There have been several different solutions developed in the literature for reasoning about ramifications using the Event Calculus (see [59] for a discussion). However, each variant has tackled only a relatively narrow class of problem. EC-R has been designed to correctly represent all the domains expressible in [59] as well as classes of domains which give rise to inconsistency and/or anomalous models in the previous general-purpose language-based action frameworks. This article demonstrates that by a structured extension of the principle of causality already embedded within EC, ramifications can be expressed by restricting changes to solely those propagated by causal effects.

The most straightforward of methods used to represent ramifications is to specify state constraints [16, 47] (sometimes referred to by the more general term of domain constraint) to exist between fluents. This has the effect of making one fluent's value dependent on the value of one or more other fluents. However, when a fluent is dependent on two or more other fluents, state constraints alone are insufficient to provide an unambiguous determination of the dependent fluent's value. State constraints may potentially determine that one of a given set of fluent state changes must be consequential, but not say which one. Augmenting to state constraints the requirement that any change be directly caused by some action eliminates the ambiguity inherent in state constraints. It may be noted however, that such an addition of the principle of causality does not render state constraints redundant. To see their necessity, it is sufficient to consider a domain description where no actions have yet taken place.

Causality has usually been seen as a primitive atomic concept describing an interconnection of domain variables in terms of the effect of one upon another. The existence of a causal effect relation is an assumption that derives its value from a power of prediction over future domain behaviour. Causality was introduced into action theories by [26, 36] and further developed by [40, 44, 62]. [26, 60] describe methods for inferring such causality relation assumptions from observations taken from a physical system. Causal change can exist in two main forms: event triggered causation and (implicit event) fluent-triggered causation. Causality may be represented formally as a predicate (as we choose in this article) or alternatively as a modal operator, see e.g. [44].

The language $\mathcal{E}\leftarrow$provides an account of ramifications based on a fixed point characterization. The conditions under which a least fixed point exists often reflect the circumstances under which a domain description makes intuitive sense from a physical causality point of view. In what follows, we make use of the well-known technical relationship between existence of fixed points and stratification. It may be useful to note that EC-R's method of stratification by causal predicate (as described below), rather than by other means (e.g. by fluent) allows it to be applied to wider classes of domains than other formalisms also based upon a general purpose logic such as [45].

This article is organized as follows. In Section 2 we recap the syntax and semantics of the language $\mathcal{E}$. Standard Event Calculus is recited in Section 3, rewritten in terms of causation points. The new EC-R theory is presented in Section 4, where different classes of ramification domains are also discussed. Section 5 gives two example axiomatizations of significant benchmark EC-R domains. Section 6 states the semantic definition of causation points and makes reference to a Least Fixed Point Logic definition in appendix, and uses this to prove a correspondence between EC-R and $\mathcal{E}$. Section 7 relates EC-R to specific recent works, Section 8 addresses details of EC-R in relation to wider challenges in the field and Section 9 summarizes and concludes the article.

In the following, we make frequent use of electronic logic circuits as canonical examples of domain classes. We have found this to be a rich and intuitive source of domains requiring commonsense reasoning. The circuits give a good graphical representation of causal relations, and it is often possible to draw a circuit isomorphic to any practical commonsense real-world domain.

2 Language $\mathcal{E}\leftarrow$

The basic definition of language $\mathcal{E}\leftarrow$is reproduced here, taken mainly from [30]. Some modifications have been made to the structure of definitions to be more suitable for translation later on.

2.1 Syntax of $\mathcal{E}\leftarrow$

$\mathcal{E}$ makes use of a set of fluent constants and a set of action constants along with the notion of the progression of time, represented using an ordering relation on a set of time points. Definitions 1–7 capture the syntax of the language.

Definition 1 (Domain Language)
A domain language for $\mathcal{E}$ is a tuple $\langle \Pi, \preceq, \Delta, \Phi \rangle$, where $\preceq$ is a partial (possibly total) ordering defined on 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 (Fluent literal)
A fluent literal L of $\mathcal{E}$ is an expression either of the form F or of the form $\neg F$, where $F \in \Phi$.

Definition 3 (h-propositions)
An h-proposition in $\mathcal{E}$ is an expression of the form

$$A \textbf{ happens-at } T$$

where $A \in \Delta$ and $T \in \Pi$.

Definition 4 (t-propositions)
A t-proposition in $\mathcal{E}$ is an expression of the form

$$L \textbf{ holds-at } T$$

where L is a fluent literal of $\mathcal{E}$, and $T \in \Pi$.

Definition 5 (c-propositions)
A c-proposition in $\mathcal{E}$ is an expression of the form

$$A \textbf{ initiates } F \textbf{ when } C$$

or alternatively of the form

$$A \textbf{ terminates } F \textbf{ when } C$$

where $F \in \Phi$, $A \in \Delta$ and C is a set of fluent literals of $\mathcal{E}$.

The definitions given so far are sufficient for defining simple domains without ramifications. Definition 6 is needed to extend the basic language to cover cases where causal dependency relations exist between fluents. A full domain description in terms of Definitions 1–6 is then possible in Definition 7.

Definition 6 (r-propositions)
An r-proposition in $\mathcal{E}$ is an expression of the form

$$L \textbf{ whenever } C$$

where L is a fluent literal, and C is a set of fluent literals of $\mathcal{E}$.

Definition 7 (Domain description)
A domain description in $\mathcal{E}$ is defined as the tuple $\langle \gamma, \eta, \tau, \rho \rangle$ where γ is a set of r-propositions, η is a set of c-propositions, τ is a set of h-propositions, ρ is a set of t-propositions.

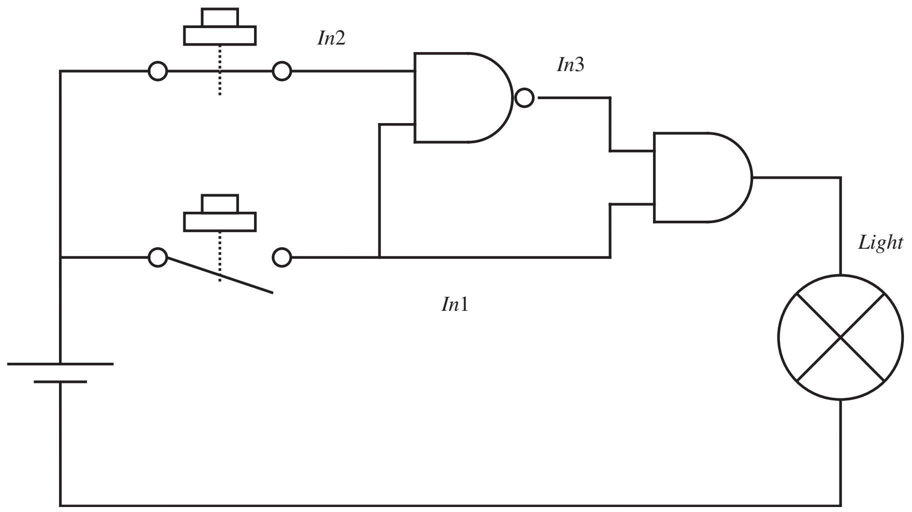

FIGURE 1. Ternary ramification domain D_1 at time 1

2.2 *Example $\mathcal{E}$ domain*

A ramification domain example of an electronic circuit is shown in Figure 1. It is constructed using a logical NAND and an AND gate.

The $\mathcal{E}$ domain description for this circuit is as follows.

$\neg In1$ **holds-at** 1	(Ex1.1)
$In2$ **holds-at** 1	(Ex1.2)
$Close1$ **happens-at** 3	(Ex1.3)
$Open2$ **happens-at** 5	(Ex1.4)
$Close1$ **initiates** $In1$	(Ex1.5)
$Close2$ **initiates** $In2$	(Ex1.6)
$Open1$ **terminates** $In1$	(Ex1.7)
$Open2$ **terminates** $In2$	(Ex1.8)
$\neg In3$ **whenever** $\{In1, In2\}\leftarrow$	(Ex1.9)
$In3$ **whenever** $\{\neg In1\}\leftarrow$	(Ex1.10)
$In3$ **whenever** $\{\neg In2\}\leftarrow$	(Ex1.11)
$Light$ **whenever** $\{In1, In3\}\leftarrow$	(Ex1.12)
$\neg Light$ **whenever** $\{\neg In1\}$	(Ex1.13)
$\neg Light$ **whenever** $\{\neg In3\}\leftarrow$	(Ex1.14)

Using **whenever** statements (Ex1.12–1.14) and appropriate fluent literals, both the 'if' aspect of logical AND gate modelling is captured as well as the 'only-if' behavior. (Ex1.13–1.14) specify what resemble 'contrapositive' conditions. The NAND gate is represented in a similar fashion to the AND gate (Ex1.9–1.11), with appropriate changes to causation polarity.

2.3 Semantics of language $\mathcal{E}$←

$\mathcal{E}$'s semantics are based on definitions of interpretations and models. Interpretations are defined as mappings of fluent, time-point pairs to *true* or *false*, representing the fluents holding or not holding, respectively at that time point (Definition 8). An interpretation satisfies a set of fluent literals at a time point if it assigns the relevant truth values to each of the corresponding fluent constants (Definition 9).

Definition 8 (Interpretation)
An interpretation of the language $\mathcal{E}$ is defined as a mapping

$$H : \Phi \times \Pi \mapsto \{true, false\} \leftarrow$$

Definition 9 (Point satisfaction)
Given a set of fluent literals C of $\mathcal{E}$ and a time point $T \in \Pi$, an interpretation H satisfies C at T iff $H(F, T) = true$ for each fluent constant $F \in C$, and $H(F', T) = false$ for each fluent literal $\neg F' \in C$.

The semantics identifies which interpretations are models via the notion of *initiation* and *termination points*, which we also refer to here together as *causation points*. Causation points are defined to include both the possibility of fluents being directly affected by events through c-propositions, and also indirectly affected by other fluents through r-propositions. Intuitively, in order to find time points at which a fluent literal L is triggered via the r-proposition L **whenever** C, we must look for time points at which one or more of the conditions in C becomes established, and the remaining conditions are already satisfied. This needs to be done recursively in order to capture effects via chains of r-propositions, and the semantics must ensure that conditions already satisfied are not about to change through further indirect effects (triggered at the same time point).

Definition 10 (below) uses a fixed point notion to capture these intuitions. The structure of Definition 10 has been modified from that in [30] due to our later formalization in Least Fixed Point Logic in appendix. Least Fixed Point Logic is useful in two respects. First, it provides an existing framework into which $\mathcal{E}$'s semantic conditions can be inserted, resulting in a rigorous definition of the least fixed point. Secondly, it represents a formal language with sufficient power to directly express the full classes of domains expressible in $\mathcal{E}$. Definition 10 is constructive, where the operator $\mathcal{F}$ is applied iteratively to collect effects successively deeper in the chain of ramifications until a fixed point is reached.

Definition 10 (Initiation/Termination point)
Let H be an interpretation of $\mathcal{E} = \langle \Pi, \preceq, \Delta, \Phi \rangle$, $D = (\gamma, \eta, \tau, \rho)$ be a domain description, and Ct be the set of symbols $\{init, term\}$. Let $Cp = (\Phi \times \Pi \times Ct)$ and $W = \mathcal{P}(Cp)$, and let the operator $\mathcal{F} : W \mapsto W$ be defined as follows. Given a (possibly empty) set $InTe \in W$, then for any $F \in \Phi$ and $T \in \Pi$, $(F, T, init) \in \mathcal{F}(InTe)$ (respectively $(F, T, term) \in \mathcal{F}(InTe)$) iff either of the following conditions are true:

(1) There exists an $A \in \Delta$ and a set C of fluent literals such that both of the following are true.

   (a) There exists both an h-proposition in η of the form 'A **happens-at** T' and a c-proposition in $\gamma\psi$of the form 'A **initiates** F **when** C' (respectively 'A **terminates** F **when** C').
   (b) H satisfies C at T.

(2) There exists an r-proposition in $\rho\psi$of the form 'F **whenever** C' (respectively '$\neg F$ **whenever** C') and a partition $\{C_1, C_2\}$ of C such that both the following apply.
   (a) C_1 is non-empty, and for all fluent constants where $F' \in C_1$, (F', T, init) $\in$ *InTe*, and for all fluent literals where $\neg F' \in C_1$, (F', T, term) $\in$ *InTe*.
   (b) There exists some $T_2 \in \Pi$ where $(T \prec T_2)$ such that for all T_1, $(T \preceq T_1 \preceq T_2)$, H satisfies C_2 at T_1.

Let $InTe^f$ be the least fixed point of the (monotonic) operator $\mathcal{F}$ starting from $\emptyset$. T is an initiation point (respectively termination point) for F in H relative to D iff $(F, T, init) \in InTe^f$ (respectively $(F, T, term) \in InTe^f$).

Definition 11 of a model specifies the persistence and causality properties we wish to see embodied in the temporal framework. It states that fluents' values do not spontaneously change without cause, but do change when required to do so by a causation point.

Definition 11 (Model)
For a domain description $D = (\gamma, \eta, \tau, \rho)$, an interpretation H of $\mathcal{E}$ is a model of D iff, for every $F \in \Phi$, and $T, T', T_1, T_3 \in \Pi$ such that $T_1 \prec T_3$, the following properties hold,

(1) Persistence. If there is no initiation point nor termination point T_2 for F in H relative to D such that $T_1 \preceq T_2 \prec T_3$ then $H(F, T_1) = H(F, T_3)$.
(2) Initiation. If T_1 is an initiation point for F in H relative to D, and there is no termination point T_2 for F in H relative to D such that $T_1 \prec T_2 \prec T_3$ then $H(F, T_3) = true$.
(3) Termination. If T_1 is a termination point for F in H relative to D, and there is no initiation point T_2 for F in H relative to D such that $T_1 \prec T_2 \prec T_3$ then $H(F, T_3) = false$.
(4) Initial conditions and associated static ramification constraints.
   (a) For all t-propositions in $\tau\psi$of the form 'F **holds-at** T', $H(F, T) = true$ and for all t-propositions in $\tau\psi$of the form '$\neg F$ **holds-at** T', $H(F, T) = false$.
   (b) For all r-propositions in $\rho\psi$of the form 'L **whenever** C', if H satisfies C at T then H satisfies $\{L\}$ at T.

Given the definition of a model, consistency and entailment can now be defined in a conventional way, in Definitions 12 and 13, respectively.

Definition 12 (Consistency)
A domain description is consistent iff it has an $\mathcal{E}$ model.

Definition 13 (Entailment)
A domain description D entails the t-proposition 'F **holds-at** T', *written* '$D \models_{\mathcal{E}} F$ **holds-at** T' iff for every $\mathcal{E}$ model H of D, $H(F, T) = true$. D entails the t-proposition '$\neg F$ **holds-at** T', iff for every $\mathcal{E}$ model H of D, $H(F, T) = false$.

Referring back to the example domain definition illustrated in Figure 1, it is possible to use the above Definitions 1–13 (paying particular attention to Definitions 10 and 11) to show the following entailments (Ex1.15–1.19).

$D_1 \models_{\mathcal{E}} \neg Light$ **holds-at** 2 (Ex1.15)
$D_1 \models_{\mathcal{E}} \neg Light$ **holds-at** 3 (Ex1.16)

$D_1 \models_{\mathcal{E}} \neg Light$ **holds-at** 4 (Ex1.17)
$D_1 \models_{\mathcal{E}} \neg Light$ **holds-at** 5 (Ex1.18)
$D_1 \models_{\mathcal{E}\leftarrow} Light$ **holds-at** 6 (Ex1.19)

3 Classical Logic Event Calculus

The particular Event Calculus domain independent axiomatization presented here is deterministic, in the sense that any attempt to initiate and terminate a fluent simultaneously will lead to inconsistency in the theory. This is not an inherent restriction of the Event Calculus (nor of EC-R), but rather has been specifically chosen here to provide correspondence with $\mathcal{E}$ for the purpose of showing that ramifications are handled correctly.

EC is written in a sorted predicate calculus with equality, with a sort $\mathcal{A}$ for *actions* (variables $a, a_1, a_2, \ldots$), a sort $\mathcal{F}$ for fluents (variables $f, f_1, f_2, \ldots$) and a sort $\mathcal{T}$ for timepoints (here either real numbers or integers, variables $t, t_1, t_2, \ldots$). Although the time structure is thus assumed to be linear in this work, again this is not an inherent restriction of the Event Calculus.

As described in [46, 57, 58], there are two parts to an Event Calculus theory. Domain Dependent Axioms (DDA) carry the causation relations along with event occurrence information specific to the domain, while Domain Independent Axioms (DIA) capture the general commonsense notions of persistence and causality.

3.1 Domain dependent EC axioms

An Event Calculus DDA theory takes the general form of (DD1), where Θ is a conjunction of clauses ('causal laws') specifying the positive instances of the predicates *Initiates*, *Terminates*, and Λ is a conjunction of positive instances (usually ground atomic formulae) of *Happens*.

$$CIRC[\Theta; Initiates, Terminates] \wedge CIRC[\Lambda; Happens] \wedge \Xi \qquad \text{(DD1)}\leftarrow$$

Ξ is a conjunction of time-independent formulae including uniqueness of names for actions and fluents, and $\Xi_h \subseteq \Xi$, a conjunction of *HoldsAt* formulae representing initial conditions. For example, if we wish to describe a simple 'light switch' domain in which the action *PressSwitch* results in the fluent *LightOn* as long as the fluent *Connected* holds, and in addition we know that at time 1 *Connected* does indeed hold and that at time 2 the switch is pressed, we express this as (Ex2.1–2.4).

$$Initiates(PressSwitch, LightOn, t) \leftarrow HoldsAt(Connected, t) \qquad \text{(Ex2.1)}\leftarrow$$

$$Happens(PressSwitch, 2) \qquad \text{(Ex2.2)}\leftarrow$$

$$HoldsAt(Connected, 1) \qquad \text{(Ex2.3)}\leftarrow$$

$$Connected \neq LightOn \qquad \text{(Ex2.4)}\leftarrow$$

If $\Theta_{Ex2} = (\text{Ex2.1})$, $\Lambda_{Ex2} = (\text{Ex2.2})$, and $\Xi_{Ex2} = (\text{Ex2.3}) \wedge (\text{Ex2.4})$, the expression $CIRC[\Theta_{Ex2}; Initiates, Terminates] \wedge CIRC[\Lambda_{Ex2}; Happens] \wedge \Xi_{Ex2}$ is equivalent to (Ex2.5).

$$
\begin{aligned}
&[Initiates(a,f,t) \;\equiv\; \leftarrow && \text{(Ex2.5)}\leftarrow\\
&\quad (a = PressSwitch \wedge f = LightOn \wedge HoldsAt(Connected, t))]\leftarrow\\
&\qquad \wedge\leftarrow\\
&[\neg Terminates(a,f,t)]\leftarrow\\
&\qquad \wedge\leftarrow\\
&[Happens(a,t) \;\equiv\; (a = PressSwitch \wedge t = 2)]\leftarrow\\
&\qquad \wedge\leftarrow\\
&[HoldsAt(Connected, 1) \;\wedge\; Connected \neq LightOn]\leftarrow
\end{aligned}
$$

3.2 Domain independent EC axioms

The Event Calculus DIA given in this section are a syntactic reformulation of the axioms given in [46] for 'deterministic Event Calculus'. The only difference is that, unlike [46], we have used two extra auxiliary predicates *InitiationPoint* and *TerminationPoint* to express the axioms a little more succinctly, because they will be useful when we come to adapt the EC axiomatization for ramifications in later sections. Both predicates are used to identify causation points; $InitiationPoint(F, T)$ means 'an action initiates fluent F at time T' while $TerminationPoint(F, T)$ means 'an action terminates fluent F at time T', defined by (CP1 and 2).

$$InitiationPoint(f,t) \;\equiv\; \exists a.(Happens(a,t) \;\wedge\; Initiates(a,f,t))\leftarrow \qquad \text{(CP1)}$$

$$TerminationPoint(f,t) \;\equiv\; \exists a.(Happens(a,t) \;\wedge\; Terminates(a,f,t))\leftarrow \qquad \text{(CP2)}$$

As in [46], we use four more auxiliary predicates. $Clipped(T_1, F, T_2)\leftarrow$(respectively $Declipped(T_1, F, T_2)$) means 'there is a termination point (respectively initiation point) for fluent F in the half-open time interval $[T_1, T_2]$'. $StoppedIn(T_1, F, T_2)$ (respectively $StartedIn(T_1, F, T_2)$) means 'there is a termination point (respectively initiation point) for fluent F in the open time interval (T_1, T_2)': (EC1–4).

$$Clipped(t_1,f,t_2) \overset{\text{def}}{\equiv} \exists t.[t_1 \le t < t_2 \;\wedge\; TerminationPoint(f,t)]\leftarrow \qquad \text{(EC1)}$$

$$Declipped(t_1,f,t_2) \overset{\text{def}}{\equiv} \exists t.[t_1 \le t < t_2 \;\wedge\; InitiationPoint(f,t)]\leftarrow \qquad \text{(EC2)}$$

$$StoppedIn(t_1,f,t_2) \overset{\text{def}}{\equiv} \exists t.[t_1 < t < t_2 \;\wedge\; TerminationPoint(f,t)]\leftarrow \qquad \text{(EC3)}$$

$$StartedIn(t_1,f,t_2) \overset{\text{def}}{\equiv} \exists t.[t_1 < t < t_2 \;\wedge\; InitiationPoint(f,t)]\leftarrow \qquad \text{(EC4)}$$

Change caused by events is captured by the following two axioms. (EC5) states that fluents initiated by an occurrence of an action continue to hold until an occurrence of an action which terminates them. Conversely, (EC6) states that fluents terminated by an occurrence of an action continue not to hold until an occurrence of an action which initiates them.

$$HoldsAt(f,t_2) \;\leftarrow\; [InitiationPoint(f,t_1) \;\wedge\; t_1 < t_2 \;\wedge\; \neg StoppedIn(t_1,f,t_2)]\leftarrow \qquad \text{(EC5)}$$

$$\neg HoldsAt(f,t_2) \leftarrow [TerminationPoint(f,t_1) \wedge t_1<t_2 \wedge \neg StartedIn(t_1,f,t_2)]\leftarrow \qquad \text{(EC6)}$$

Persistence is expressed in axioms (EC7, 8), stating that fluents change their truth values only via the occurrence of initiating and terminating actions.

$$HoldsAt(f,t_2) \leftarrow [HoldsAt(f,t_1) \wedge t_1<t_2 \wedge \neg Clipped(t_1,f,t_2)]\leftarrow \qquad \text{(EC7)}\leftarrow$$

$$\neg HoldsAt(f,t_2) \leftarrow [\neg HoldsAt(f,t_1) \wedge t_1<t_2 \wedge \neg Declipped(t_1,f,t_2)]\leftarrow \qquad \text{(EC8)}\leftarrow$$

Note that in a deterministic setting the use of both *Clipped/Declipped* and *StoppedIn/StartedIn* is necessary to capture both the principle that fluents cannot be simultaneously initiated and terminated, and the convention that causal effects manifest themselves only at times (immediately) after the corresponding causation points.

In terms of the light switch example (Ex2.1–2.4), it is straightforward to show that (CP1 and 2) and (EC1–6), together with (Ex2.5), entail (for example) *HoldsAt*(*LightOn*, 3).

The solution to the Frame Problem[1] comes from a forced (defined) separation of causal minimization from temporal projection. A causal domain closure assumption ensures that all effects are precipitated by an action, and do not occur otherwise. As shown in (DD1), causal minimization is performed by parallel circumscription on the two causal predicates *Initiates* and *Terminates* in the domain-dependent axiom set. For standard EC formulations such as (Ex2.1–2.4) this reduces to predicate completion, as illustrated in (Ex2.5). When ramifications are present, however, the situation changes.

4 Ramifications in classical logic EC

Where a domain has fluents with causal interdependencies, in which an event's effect propagation depends on the presence or absence of other effects, then these can often be represented as language $\mathcal{E}$ ramifications. This section develops a classical logic Event Calculus capable of reasoning about ramifications expressed through a '**whenever**' style construct from language $\mathcal{E}$ for classes of domains that encompasses those accepted by $\mathcal{E}$. [59] examined potential Event Calculus-based solutions to ramification domains in terms of the concepts state constraints and effect constraints. These distinct methods were originally conceived to represent static relationships and dynamic triggered changes for transition-driven interacting events, respectively.

A state constraint representing a static relationship that must always hold between fluents is represented in the Event Calculus by *HoldsAt* formulae, each using a single universally quantified time variable. The purpose of the formulae is to define one or more derived (non-frame, non-persisting) fluents. Such a state constraint is illustrated in (Ex3.1 and 3.2), whose purpose is to define the derived fluent $F1$ in terms of the frame fluent $F2$.

$$HoldsAt(F1,t)\leftarrow HoldsAt(F2,t) \qquad (Ex3.1)\leftarrow$$
$$\neg HoldsAt(F1,t)\leftarrow\neg HoldsAt(F2,t) \qquad (Ex3.2)\leftarrow$$

Effect constraints representing the dynamic propagation of event-triggered change are defined in the Event Calculus using *Initiates* and *Terminates* formulae, each using a single

[1]See section 8 for a wider discussion.

universally quantified action variable, as shown in (Ex3.3 and 3.4). (Ex3.5 and 3.6) are equivalent to formulae (Ex3.3 and 3.4) specifying the same effect constraint in terms of initiation points and termination points.

$$Initiates(a, F1, t) \leftarrow \quad Initiates(a, F2, t) \leftarrow \qquad \text{(Ex3.3)} \leftarrow$$

$$Terminates(a, F1, t) \leftarrow \quad Terminates(a, F2, t) \leftarrow \qquad \text{(Ex3.4)} \leftarrow$$

$$InitiationPoint(F1, t) \leftarrow \quad InitiationPoint(F2, t) \leftarrow \qquad \text{(Ex3.5)} \leftarrow$$

$$TerminationPoint(F1, t) \leftarrow \quad TerminationPoint(F2, t) \leftarrow \qquad \text{(Ex3.6)} \leftarrow$$

Here we will present a combination and enhancement of the two approaches, chiefly because such a combination aligns well with the semantic definition of ramifications in the language $\mathcal{E}$ using inductive fixed point definitions [2].[2]

[46] provides a translation of a version of $\mathcal{E}$ without ramifications into the language of classical logic Event Calculus. However, when a mapping of the full version of $\mathcal{E}\leftarrow$with ramifications described in [30] into classical logic Event Calculus is considered, the situation becomes considerably more complex.

When using the ramification construct '**whenever**' of language $\mathcal{E}$, there arises a natural question over the classes of domains that can be properly represented. There are two particular classes that merit further consideration, which we refer to below as *cycling domains* and *instantaneously propagated effect domains*, one special case of which is the *mutually coupled instantaneously propagated effect domain* exemplified in the Gearwheel example (in a later section).

4.1 Unstable ramification domains

As an example of cycling domains, consider a domain description having two 'complete' ramifications, defined below.

$$\begin{aligned} F_2 \ &\textbf{whenever} \ F_1 \\ \neg F_2 \ &\textbf{whenever} \ \neg F_1 \\ \neg F_1 \ &\textbf{whenever} \ F_2 \\ F_1 \ &\textbf{whenever} \ \neg F_2 \end{aligned}$$

Under $\mathcal{E}$'s semantics, a domain description comprising such a set of propositions is inconsistent. There are consequently no $\mathcal{E}$ models. Such a domain may arise in an electronic circuit containing two logic gates (a buffer and inverter), connected with each output connected to the other gate's input (Figure 2).

Clearly, if the logic gates are considered to be 'ideal' (with zero delay), then the circuit of Figure 2 is physically inconsistent. From an intuitive commonsense reasoning point of view, the circuit is not apparently seen as inconsistent. Instead, a small delay is assumed in the gates in order to make sense of the arrangement, and the circuit is found intuitively to cycle continuously. Therefore, even in a notional abstract sense, the logic circuit is assumed to have an arbitrarily small non-zero delay, because this is necessary to ascribe causality appropriately, and thereby reason about the behaviour in a

[2]See [61] for an alternative formulation of ramifications using inductive definitions.

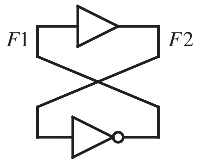

FIGURE 2. Unstable negative cycling domain

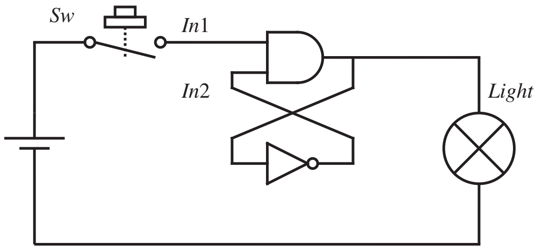

FIGURE 3. Conditionally stable negative cycling domain

meaningful way. In practice, a physical circuit of this type would oscillate continuously, and have no steady state.

Now consider a modification of Figure 2, shown in Figure 3, where instability is conditional.

$$\begin{array}{lll} Light & \textbf{whenever} & In1, In2 \\ \neg Light & \textbf{whenever} & \neg In1 \\ \neg Light & \textbf{whenever} & \neg In2 \\ \neg In2 & \textbf{whenever} & Light \\ In2 & \textbf{whenever} & \neg Light \end{array}$$

Figure 3 has one input state for which the circuit is stable (In1 false), and one where it is not (In1 true). Note that in all allowable $\mathcal{E}$ models, $In1$ is false.

While there are ways of handling this condition in an immediate logical sense (see [8]), it seems that examples like this are rare in the commonsense world (we had to resort to an esoteric electronic circuit). More usually, such conditionally stable domain descriptions raise questions over the correctness of the axiomatization.

It is possible to detect an unstable cycling condition in the axiomatization using a causal ramification consistency check. Causal stratification (c-stratification) used in conjunction with a state constraint consistency check are sufficient to realize this function. Should a domain axiomatization fail this test, there is an opportunity for an agent to take measures to update or correct its knowledge of the domain before attempting any reasoning.[3]

For this reason, and the fact that such examples go beyond $\mathcal{E}$'s specification, we will not address domains of this nature here. If required, other methods could be brought to bear on unstable domains including the use of stability as a semantic condition for a proposition in a language, or alternatively a three level logic where the third level represents an unstable state. Here we will focus on the expressivity contained in $\mathcal{E}$ because it facilitates results in later sections.

[3]See section 8 for more information.

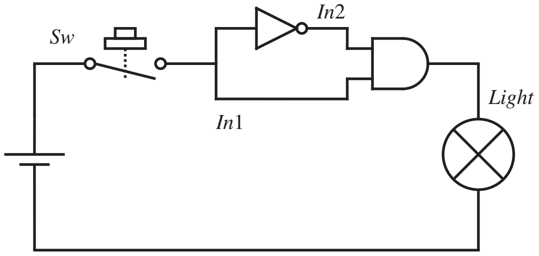

FIGURE 4. Mutually cancelling effect domain

Arguably, the most important class of ramification domain is that containing instantaneously propagated interacting effects, addressed next.

4.2 Instantaneously propagated effect domains

Consistent and stable domains may have instantaneously propagated effects. Figure 4 contains one example of this class. The light in Figure 4 should never illuminate, irrespective of the position of *Sw*. However, if reasoning does not handle instantaneously propagated effects correctly, this may not be indicated as so by the logical model.

The example is set up so that one action causes more than one intermediate effect, but where these effects combine to self-cancel at some point down the ramification chain. An ideal logic circuit in the form of Figure 4 would have a steady state output always of 0, and so this is reflected in our approach. However, a real circuit contains what designers call a 'hazard' due to the delay of the inverter. Upon transition 0–1 of *In*1 caused by action *Sw*, a real circuit will blip *Light* briefly at 1 until it settles again at 0.[4] A comparison may be made between this (unwanted) brief transition in the physical circuit and the stages of erroneous reasoning that may occur in faulty action theories concerning propagated effects.

To capture instantaneously propagated interacting effects, it may be tempting to use Event Calculus causation axioms of the form below, where for every dependency on a fluent truth value, there is an associated check that it is not imminently about to change. This check curtails the extent of causation.

$$\begin{aligned}&Initiates(a, Light, t) \leftarrow [Initiates(a, In1, t) \wedge HoldsAt(In2, t) \qquad \text{(Ex4.1)}\leftarrow\\ &\quad \wedge \neg\exists a1(Happens(a1, t) \wedge Terminates(a1, In2, t))]\leftarrow\end{aligned}$$

However, if not handled carefully, the inclusion of the capability to handle instantaneously propagated effects in this way can cause difficulty. Causation descriptions in the form of (Ex4.1) are inherently open to ambiguity when predicates *Initiates* and *Terminates* are minimized in parallel. This ambiguity becomes apparent in a non-unique causal minimization

[4]Although the physical circuit exhibits artefacts, it would not be desirable to attempt to duplicate the consequent behaviour in a reasoning system based on a domain description describing merely the abstract logic gate behaviour. If such artefacts are considered an important part of the domain, they should be modelled explicitly through more detailed causation statements.

with competing model tradeoffs in Yale Shooting Problem style ([25]).[5] The same problem can be seen in simpler domains providing the instantaneous propagation characteristic is retained. As an example, consider the following simple domain (Ex5.1–5.6) with one ramification, written in $\mathcal{E}$. Note that in $\mathcal{E}$, the domain has a unique model in which F holds at 2.

A **initiates** $F1$ (Ex5.1)
A **happens-at** 1 (Ex5.2)
$\neg F$ **holds-at** 1 (Ex5.3)
$\neg F1$ **holds-at** 1 (Ex5.4)
$F2$ **holds-at** 1 (Ex5.5)
F **whenever** $\{F1, F2\} \leftarrow$ (Ex5.6)

The first five propositions of the domain description (Ex5.1–5.5) may be expressed in EC as follows (Ex5.7–5.9).

$$Initiates(A, F1, t) \leftarrow \qquad \text{(Ex5.7)}$$
$$Happens(A, 1) \leftarrow \qquad \text{(Ex5.8)}$$
$$\neg HoldsAt(F, 1) \wedge \neg HoldsAt(F1, 1) \wedge HoldsAt(F2, 1) \leftarrow \qquad \text{(Ex5.9)}$$

In order to express (Ex5.7 and 5.8) in terms of causation points in a way suitable for minimization in this example, (CP1 and 2) are weakened to (CP3 and 4).

$$InitiationPoint(f, t) \ \leftarrow \ \exists a.(Happens(a, t) \ \wedge \ Initiates(a, f, t)) \leftarrow \qquad \text{(CP3)}$$

$$TerminationPoint(f, t) \ \leftarrow \ \exists a.(Happens(a, t) \ \wedge \ Terminates(a, f, t)) \leftarrow \qquad \text{(CP4)}$$

To translate the ramification constraint (Ex5.6) into EC, we specify the cases where F is triggered by changes in $F1$ and $F2$. Writing (Ex5.10) below in terms of *InitiationPoint* and *TerminationPoint*, the first disjunct in the condition says F is triggered if $F1$ is initiated and $F2$ is true, and $F2$ is not about to be terminated (as it might if there were mutually defeating ramification effects or concurrent actions).

$$\begin{aligned}
&InitiationPoint(F, t) \leftarrow \qquad \text{(Ex5.10)}\\
&\quad [InitiationPoint(F1, t) \ \wedge \ HoldsAt(F2, t) \ \wedge \neg TerminationPoint(F2, t)] \vee\\
&\quad [InitiationPoint(F2, t) \ \wedge \ HoldsAt(F1, t) \wedge \neg TerminationPoint(F1, t)] \vee\\
&\quad [InitiationPoint(F1, t) \ \wedge \ InitiationPoint(F2, t)] \leftarrow
\end{aligned}$$

Before permitting a change in F, (Ex5.10) checks $F2$ is not about to be terminated by a change in $F1$ caused by the original action.

The solution to the frame problem in EC relies upon theory partitioning (syntactically defining a separation) of the causal predicates *InitiationPoint* and *TerminationPoint* from other parts of the theory. These predicates are then minimized in parallel within their partition. (Ex5.11) below is the complete domain-dependent part of the theory, where (DD1)'s Θ is (Ex5.7), Λ is (Ex5.8) and Ξ includes (Ex5.9). It is produced by also minimizing direct effects of actions and events (in the usual way) in addition to indirect effects as expressed by the predicates *InitiationPoint* and *TerminationPoint*.

$$\text{CIRC}[\text{Ex5.10} \wedge \text{CP3} \wedge \text{CP4} \wedge \text{DD1}; InitiationPoint, TerminationPoint] \leftarrow \qquad \text{(Ex5.11)}$$

[5]In addition, (Ex4.1) would also preclude reasoning about domains with concurrent actions.

(Ex5.11) then allows the definitions of *InitiationPoint* and *TerminationPoint* to be reduced to the first order sentence (Ex5.12).

$$
\begin{aligned}
&[(\mathit{InitiationPoint}(f,t)\ \equiv ((f{=}F1 \wedge t{=}1)\ \vee\ (f{=}F \wedge t{=}1))) \wedge\leftarrow \\
&\quad \neg \mathit{TerminationPoint}(f,t)]\leftarrow \\
&\vee \\
&[(\mathit{InitiationPoint}(f,t)\ \equiv (f{=}F1 \wedge t{=}1)) \wedge\leftarrow \\
&\quad (\mathit{TerminationPoint}(f,t) \equiv (f{=}F2 \wedge t{=}1))]\leftarrow
\end{aligned}
\tag{Ex5.12}
$$

The disjunctive form of (Ex5.12) effectively gives rise to two models. The second of these is an anomalous model, where the initiation of F has been traded for the termination of $F2$. This tradeoff has obvious similarity to the Yale Shooting Problem. The example shows why we cannot just naively translate $\mathcal{E}$'s r-propositions such as (Ex5.6) into expressions such as (Ex5.10), in which positives are defined in terms of negatives as regards the predicates *InitiationPoint* and *TerminationPoint*.

4.3 Analysis

$\mathcal{E}\leftarrow$is constructed within a general framework of inductive definition that is capable of accepting domain descriptions that are stable (i.e. non-negative cycling). Inductive definitions specify a relation through a constructive process of iterating a function that defines a relation, given in terms of the presence or absence of specified tuples. Positive rule sets are straightforward, but cases where negation occurs within the body of rules (specifying absence) require special treatment. To properly account for the absence of rules (specified by negation), a closure assumption must be constructed separately for each individual constructive level, rather than over the entire theory. For ramification applications, this means causal effects are determined one layer at a time.

The classes of domains acceptable to $\mathcal{E}\leftarrow$thus include those in which the fluents can be ordered by layering in strata, one fluent per stratum. This results in one layer depending negatively only on effects in layers strictly lower.[6] A fluent influence graph (produced from the domain's **whenever** statements) would then show only acyclic dependency [4]. Fluent ordering of this type forms a syntactical analog to the causal nature of physical propagation.

However, not all intuitively sensible physical domains having a clear causality path have a valid stratification under this fluent ordering representation. Some domains may have a fluent dependency graph that is positive cyclic, but not unstable (such as the Gearwheel domain, described in an example later).

To rectify this anomaly, we will choose instead to apply stratification to the causal subtheory of EC-R. This gives conveniently good results, because owing to the difference in meaning of negation in $\mathcal{E}$'s fluent effect propositions and negation in EC-R causal theories, stratification is possible for positive cyclic domains in EC-R, while guaranteeing the absence of negative cycles as desired. Algorithms to create a stratification are well known, and may be incorporated as part of an automated knowledge base coherency check.

[6]The choice of any alternative correct stratification would have no effect on the semantics of the theory. Although stratification assignment is not unique, any correct choice of stratification is equivalent to any other correct choice from a semantic standpoint.

The stratification precondition for domain acceptability is a test not easily reproducible at the language $\mathcal{E}$ domain description stage. A stratification process pays special attention to the occurrence of negation. However, the type of negation important for the assessment of a domain's stability is invisible at the $\mathcal{E}$ domain specification level. It is not fluent negation but rather negation indicating the absence of a causation point that is important. This type of negation is only introduced through the translation into EC-R. Both the translation and stratification are syntactic functions, and it is possible to create a function composition to realize the precondition test on an $\mathcal{E}$ specification directly. However, in practice this method would be little simpler than applying both functions separately.

4.4 Restricting the domain class: stratification

In the previous sections we have seen that certain classes of domains can produce unstable fluent states, and that others while stable, can easily result in anomalous models. In $\mathcal{E}$, cycling domains such as those in Figures 2 and 3 fail to yield models (i) because the r-propositions cannot be satisfied as "static constraints" (see condition 4(b) in Definition 11 of a model), and (ii) because Definition 10 results in some time-points being both initiation and termination points for the same fluent. We wish to accept a class of domain for which a model is possible under $\mathcal{E}$'s definitions. Such a class explicitly excludes domains with negative cycles, but does include other instantaneously propagated mutually interacting effect domains.

To realize this domain class acceptance, it is necessary to formulate a classical logic theory by translation from $\mathcal{E}$, where each causation-point predicate is unique to a fluent. This is achieved using predicate indexing on the fluent name. The theory is then subject to a stratification process, which attempts to allocate new predicate indices so as to create a stratified theory. If this cannot be done, then the $\mathcal{E}$ domain specification from which the mapping resulted is not stable, and thus not acceptable to EC-R.

In addition to stratification of the causal subtheory of EC-R, the other precondition for a domain to be acceptable is consistency of the static component of the constraints (generated by translation from **whenever** statements in Definition 21 later on). Figure 2 is an example of a domain that has inconsistent static constraints. It may be helpful at this point to recall from the beginning of the section that ramifications are expressed by **whenever** statements translated into both classical causal effect and static constraint first-order formulae. This ensures that the effect and constraint formulae are always matched.

4.5 Translation to a classical ramification theory

Language $\mathcal{E}$'s propositions t and h can be mapped into classical logic straightforwardly. With reference to Θ, Λ and Ξ of the standard Event Calculus domain description (DD1), Definitions 14 and 15 define the mapping of these propositions.

Definition 14 (Initial conditions mapping)
The EC-R static constraint mapping from $\mathcal{E}$ is the theory Ξ_h such that the following conditions apply.

- ←for each t-proposition 'F **holds-at** T', Ξ_h contains the axiom $HoldsAt(F, t)$;
- ←for each t-proposition '$\neg F$ **holds-at** T', Ξ_h contains the axiom $\neg HoldsAt(F, t)$←
- ←Ξ_h contains no other formulae.

Definition 15 (Event occurrence mapping)
The *EC-R* static constraint mapping from $\mathcal{E}$ is the theory Λ such that the following conditions apply.

- for each h-proposition '*A* **happens-at** *T*', Λ contains the axiom $Happens(A, t)$;
- Λ contains no other formulae.

Language $\mathcal{E}$ has two types of causal constructs, one for direct actions (**initiates**, **terminates**) and another for expressing ramification relationships between fluents (**whenever**).

If the causal description of a domain is expressed in terms of members A of the set of actions, and members F of the set of fluents, then the direct action constructs A **initiates** F **when** C, and A **terminates** F **when** C, (where $C = \{L_1, ..., L_m\}$), may be mapped into classical logic by Definition 16.

Definition 16 (Direct causation)
The *EC-R* direct action causation description is the theory Θ such that

- for each c-proposition '*A* **initiates** *F* **when** *C*' in *D*, Θ contains the axiom
$Initiates(A, F, t) \leftarrow \bigwedge_{F_n \in C} HoldsAt(F_n, t) \wedge \bigwedge_{\neg F_n \in C} \Big(\neg HoldsAt(F_n, t)$;
- for each c-proposition '*A* **terminates** *F* **when** *C*' in *D*, Θ contains the axiom
$Terminates(A, F, t) \leftarrow \bigwedge_{F_n \in C} HoldsAt(F_n, t) \wedge \bigwedge_{\neg F_n \in C} \Big(\neg HoldsAt(F_n, t)$;
- Θ contains no other formulae.

In translating a given ramification domain description *D* into *EC-R*, all individual c-propositions and r-propositions are included in the *EC-R* causation description. For the purposes of theory construction, a subscript is applied to causal predicates to maintain a unique predicate for each fluent. The translation process begins with the formation of a separated causation description C_{Sd} for direct actions, as expressed by Definition 17.

Definition 17 (Separated causation$_d$)
The *EC-R* direct action causation description is the theory C_{Sd} such that

- for each c-proposition '*A* **initiates** *F* **when** *C*' in *D*, C_{Sd} contains the axiom

  $InitiationPoint_F(F, t) \leftarrow \exists a.(Happens(a, t) \wedge Initiates(a, F, t))$;

- for each c-proposition '*A* **terminates** *F* **when** *C*' in *D*, C_{Sd} contains the axiom

  $TerminationPoint_F(F, t) \leftarrow \exists a.(Happens(a, t) \wedge Terminates(a, F, t))$;
- C_{Sd} contains no other formulae.

Indirect triggering of fluents by actions is then expressed by looking at the conditions under which **whenever** is triggered. A 2-precondition example, *F* **whenever** $\{F1, F2\}$ is triggered in three cases: if both preconditions are initiated, or if the first fluent is initiated while the second holds, and is not terminated. It is also triggered if the first fluent holds, and is not terminated while the second is initiated. To help build a formula describing the general case, it is useful to specify these triggering subconditions separately.

Let $\Sigma_i(\varphi, \tau)$, $\Sigma_t(\varphi, \tau)$, $\Omega_{hp}(\varphi, \tau)$ and $\Omega_{hn}(\varphi, \tau)$ be the following subformulae expressed in terms of a fluent variable φψand time τ.

$$\Sigma_i(\varphi, \tau) \overset{\text{def}}{\equiv} InitiationPoint_{\varphi}(\varphi, t) \leftarrow \qquad \text{(C1)}$$

$$\Sigma_t(\varphi,\tau) \overset{\text{def}}{\equiv} \mathit{TerminationPoint}_{\varphi}(\varphi,t)\leftarrow \tag{C2}$$

$$\Omega_{hp}(\varphi,\tau) \overset{\text{def}}{\equiv} \mathit{HoldsAt}(\varphi,t) \wedge \neg\mathit{TerminationPoint}_{\varphi}(\varphi,t)\leftarrow \tag{C3}$$

$$\Omega_{hn}(\varphi,\tau) \overset{\text{def}}{\equiv} \neg\mathit{HoldsAt}(\varphi,t) \wedge \neg\mathit{InitiationPoint}_{\varphi}(\varphi,t)\leftarrow \tag{C4}$$

Both $\Sigma_i(\varphi,\tau)\leftarrow$and $\Sigma_t(\varphi,\tau)\leftarrow$represent causation point relations. $\Omega_{hp}(\varphi,\tau)\leftarrow$represents that the fluent denoted by $\varphi\psi$holds while not subject to imminent termination. $\Omega_{hn}(\varphi,\tau)$ represents that the fluent denoted by φ does not hold while not subject to imminent initiation.

Using the subconditions (C1–4), we may now enumerate the triggering conditions for an arbitrary **whenever** statement. To represent all conditions, a somewhat reduced powerset of the fluents listed in the **whenever** statement is formed, and then partitioned into two subsets, one of which is comprised of triggered fluents, the other of fluents whose state remains constant. The case in the powerset corresponding to none of the fluents being triggered is eliminated.

For a set C of conditional fluent literals such that $C \neq \emptyset$, let the following subset of a powerset partition be defined between the locally scoped variables.

$$P'(C) = \{\, (X,Y) : X \cup Y = C, X \cap Y = \emptyset, X \neq \emptyset \,\} \leftarrow$$

Propagated effects between fluents due to a **whenever** statement will then occur under the trigger conditions expressed compactly in (C5),

$$\Psi(C,\tau) \overset{\text{def}}{\equiv}\leftarrow \bigvee_{(C_1,C_2)\in P'(C)\leftarrow} \left(\bigwedge_{F\in C_1} \Sigma_i(F,\tau) \wedge \bigwedge_{F\in C_2} \Omega_{hp}(F,\tau) \right) \wedge\leftarrow \left(\bigwedge_{\neg F\in C_1} \Sigma_t(F,\tau) \wedge \bigwedge_{\neg F\in C_2} \Omega_{hn}(F,\tau) \right) \tag{C5}$$

In translating a given domain description D into *EC-R*, all individual c-propositions and r-propositions of the form L **whenever** C are included in the *EC-R* causation description.

Given the above specification of the triggering conditions for an arbitrary **whenever** statement, we must now just specify the effect to be propagated upon triggering. In the case of a statement L **whenever** C this would specify the effect to be literal L. Definition 18 then forms the separated causation description C_{Si} for indirect actions by utilizing (C5) to capture the conditions under which the **whenever** statements are triggered.

DEFINITION 18 (Separated causation$_i$)
The *EC-R* indirect action separated causation description is the theory C_{Si} such that for each c-proposition 'L **whenever** C' *in* D,

- if L is F then C_{Si} contains the axiom

  $\mathit{InitiationPoint}_F(F,t)\leftarrow \quad \Psi(C,\tau);$
- if L is $\neg F$ then C_{Si} contains the axiom

  $\mathit{TerminationPoint}_F(F,t)\leftarrow \quad \Psi(C,\tau);$
- C_{Si} contains no other formulae.

To form a complete separated causation description, let theory C_S represent $C_{Sd} \cup C_{Si}$, where C_S is modified to be re-expressed in clausal form. This involves merely eliminating disjunctions in (C5), see (Ex6.15) for an instantiated example.

Now that the translation of the language $\mathcal{E}$ statements into separated causation descriptions has been performed, stratification can be attempted. C-stratification (causal stratification) searches for an allocation of the unique causation point predicates to a stratum level. Definition 19 is adapted and specialized from the stratification conditions used in Answer Set logic programs presented in [38]. Function s from Definition 19 maps from fluents to stratum ordinals, defined such that the c-stratification conditions hold. (It may be helpful to consider that according to Definitions 17 and 18, C_S may only include clauses with heads either *InitiationPoint* or *TerminationPoint*.)

Definition 19 (C-stratification)
Given a non-empty set Φ of fluent constants, a theory C where each rule $r \in C$ is denoted by $head(r)$, and body literals by $pos(r)$ and $neg(r)$, then C is c-stratified iff there exists a mapping $s : \Phi \mapsto \mathcal{N}$ such that for all literals L_1, L_2, and all fluents F_1, F_2 the following apply:

(1) if $L_1(F_1, t) \in head(r)$ and $L_2(F_2, t) \in pos(r)$ then $s(F_1) \geq s(F_2)$
(2) if $L_1(F_1, t) \in head(r)$ and $L_2(F_2, t) \in neg(r)$ then $s(F_1) > s(F_2)$.

Let C_{ECR} be a c-stratified representation of C_S (already in clausal form) by substituting predicates $InitiationPoint_{s(F)\leftarrow}$and $TerminationPoint_{s(F)\leftarrow}$appropriately according to stratification function s.[7] If no stratification can be found, then the language $\mathcal{E}$ domain description is assumed to not meet the requirements for domain expression in EC-R.

C-stratification of theory C_S requires the replacement of each fluent-specific causation predicate with a stratum indexed causation predicate. Definition 20 defines the correct formulation of the causation theory subject to the stratification conditions.

Definition 20 (Causal theory)
Given a non-empty set Φ of fluent constants, a causal theory C_{ECR} is correctly formed iff there exists a stratification function $s : \Phi \mapsto \mathcal{N}$ such that,

- for every rule R_1 in C_S, there exists a rule R_2 in C_{ECR} where R_1 is equivalent to R_2 subject to the substitution of all literals L_F in R_1, by $L_{s(F)}$ in R_2 where L_x represents a literal name L subscripted with x.
- s is selected such that C_{ECR} is c-stratified according to Definition 19.
- s *produces strata numbered 0 to n.*

Having formulated the full causation description C_{ECR} in stratified form, causal domain closure must now be performed in stages. Each successive stage of ramification beginning with direct actions must be causally minimized separately before propagating to the next stage of effects. Prioritized parallel circumscription using the assigned stratification accomplishes this by expressing a preference for minimizing first the effects of direct actions, and then

[7]Technically, it is sufficient to stratify solely C_{Si} because C_{Sd} is already stratified (by definition). For reasons of presentational clarity however, we choose to stratify the combined theory C_S.

effects of successive levels of indirect actions. Circumscription policy to accomplish this prioritized minimization is described by (C6).

$$\begin{aligned}&\mathrm{CIRC}[C_{ECR} \wedge \ \mathrm{DD1}; \\ &\qquad \mathit{InitiationPoint}_0, \mathit{TerminationPoint}_0 \ > \cdots >\psi \\ &\qquad \mathit{InitiationPoint}_n, \mathit{TerminationPoint}_n]\end{aligned} \tag{C6}$$

Further intuitive clarity may be gained from considering the identical minimization policy stated in a hierarchical form, (C7). This description becomes a conjunction of causal closures, where each conjunct represents one stratum of causal effects. An equivalence between (C6) and (C7) is proved in [37].

$$\bigwedge_{i=0}^{n}\Bigg(\begin{aligned}&\mathrm{CIRC}[C_{ECR} \wedge \ \mathrm{DD1}; \\ &\qquad \mathrm{InitiationPoint}_i, \mathit{TerminationPoint}_i; \\ &\qquad \mathit{InitiationPoint}_{i+1}, \mathit{TerminationPoint}_{i+1} \ ;\cdots; \\ &\qquad \mathit{InitiationPoint}_n, \mathit{TerminationPoint}_n]\end{aligned} \tag{C7}$$

When considering Event Calculus causal theories, it is useful to use the property of circumscription in (Circ) below to freely factor *Happens* out of causal descriptions independently of the causal minimization process. For any sentence containing Υ, where Υ does not contain predicates P and Z, the property (Circ) holds. See [37] for more details on circumscription theory.

$$CIRC[\Gamma(P,Z) \wedge \Upsilon \, ; P \, ; R] \ \equiv \ CIRC[\Gamma(P,Z) \, ; P \, ; R] \wedge \Upsilon$$

What remains now is to combine each stratum's effect axiom into a single pair of minimized causal predicates containing the complete description of domain causality. Having created a set of predicates whose disjunction describes complete domain causality, it is possible to recombine into one pair of predicates as is usual in EC theories.[8]

$$\mathit{InitiationPoint}(f,t) \ \equiv \bigvee_{F \in \Phi} \mathit{InitiationPoint}_{s(F)}(F,t) \wedge (f = F) \tag{C8}$$

$$\mathit{TerminationPoint}(f,t) \ \equiv \bigvee_{F \in \Phi} \mathit{TerminationPoint}_{s(F)}(F,t) \wedge (f = F) \tag{C9}$$

With the triggering of change of fluent state now axiomatized, the static component of the **whenever** statement must be handled in terms of it imposing a constraint between fluents. Definition 21 defines the static constraint theory mapped from the $\mathcal{E}$ r-propositions.

DEFINITION 21 (Constraint mapping)
The *EC-R* static constraint mapping from $\mathcal{E}$ is the theory S_{ECR} such that the following conditions apply.

- for each r-proposition '*F* **whenever** *C*', S_{ECR} contains the axiom

$$HoldsAt(F,t) \leftarrow \bigwedge_{F_n \in C} HoldsAt(F_n,t) \wedge \bigwedge_{\neg F_n \in C}\Bigg(\neg HoldsAt(F_n,t);$$

[8]Here it is necessary to assume a finite language describing a finite set of fluents.

- for each r-proposition '$\neg F$ **whenever** C', S_{ECR} contains the axiom

$$\neg HoldsAt(F,t) \leftarrow \bigwedge_{F_n \in C} HoldsAt(F_n,t) \wedge \bigwedge_{\neg F_n \in C} \neg HoldsAt(F_n,t);$$

- S_{ECR} contains no other formulae.

Consistency of the static component of the constraints S_{ECR} is a pre-condition (in addition to stratification) for domain acceptance.

The complete domain-dependent causation theory is then defined to be T_{EC}. The full Event Calculus Ramification domain theory is further defined to be ECR.

$$\text{C7–9} \wedge S_{ECR} \qquad (T_{EC})$$

$$\text{EC1–8} \qquad (Det_{EC})$$

$$T_{EC} \wedge Det_{EC} \qquad \text{(ECR)}$$

The EC-R theory as presented is capable of representing ramifications in the classes of domains described using **whenever** statements, which are statically consistent and for which a c-stratification exists. In summary, direct effects (Θ) are represented in an equivalent way to those found in [46, 59]. Initial conditions and event occurrences along with causal minimization of direct actions are defined by (DD1). Indirect effects due to ramifications are translated to first order formulae (C_{Si}). Full causal minimization is then achieved using (C6). (C8–9) captures causation points for the entire domain. **Whenever** statements are also translated into S_{ECR} to capture the static constraint component of ramifications. T_{EC} forms a full domain dependent axiom set, while ECR adds the domain independent axiom set to represent the full theory. Domain acceptance is conditional upon successful stratification and the consistency of S_{ECR}.

EC-R represents somewhat of a departure from traditional [58] style Event Calculus theories in two ways. First, there is a separation of definition for causation points due to a direct action from those propagated indirectly from another causation point. Secondly, causal minimization does not reduce to simple predicate completion. Instead, a prioritized minimization policy is used, forming predicate completion one stratum at a time.

5 EC-R domain examples

To illustrate EC-R as formalized in the previous sections, we axiomatize two example domains to show different aspects of the theory. The first example continues with the domain depicted in Figure 1 (defined in section 2.2) to provide an instance of a mutually (and instantaneously propagated) interacting effect ramification domain, known to be difficult to model correctly. (Figure 4 is also a simple instance of the same class of domain drawn for illustrative purposes. If Figure 1 can be solved, then Figure 4 can also, but not vice-versa.)

The second example is chosen to illustrate how the Gearwheel example is supported in this formalism. The Gearwheel domain is positive cycling, and challenging for any formalism, but particularly so for those based on a general purpose language.

5.1 Mutually interacting effect domain

Proceeding with the axiomatization of domain dependent features, the unique name axioms for the actions and fluents of D_1 (Figure 1) are given in (Ex6.1 and 6.2).

$$\text{UNA[Open1, Open2, Close1, Close2]} \tag{Ex6.1}$$

$$\text{UNA[In1, In2, In3, Light]} \tag{Ex6.2}$$

The domain's two initial conditions, (Ex1.1 and 1.2), are expressed in the conventional way for Event Calculus (Ex6.3 and 6.4).

$$\neg HoldsAt(In1, 1)\leftarrow \tag{Ex6.3}$$

$$HoldsAt(In2, 1)\leftarrow \tag{Ex6.4}$$

Causal relations describing direct actions for the initiation and termination of fluents defined in (Ex1.5–1.8) are written in Event Calculus as (Ex6.5–6.8).

$$Initiates(Open1, In1, t)\leftarrow \tag{Ex6.5}$$

$$Initiates(Open2, In2, t)\leftarrow \tag{Ex6.6}$$

$$Terminates(Close1, In1, t)\leftarrow \tag{Ex6.7}$$

$$Terminates(Close2, In2, t)\leftarrow \tag{Ex6.8}$$

For the purposes of this example, we will choose event occurrences that illustrate some interesting transitions within the system. Axioms (Ex6.9 and 6.10) are added.

$$Happens(Close1, 3)\leftarrow \tag{Ex6.9}$$

$$Happens(Open2, 5)\leftarrow \tag{Ex6.10}$$

Domain closure for direct-effect causation and event occurrence are defined by (DD1). In this example, (DD1) is instantiated to become (Ex6.11).

$$\text{CIRC}[\text{Ex6.5–8}; Initiates, Terminates] \wedge \text{CIRC}[\text{Ex6.9–11}; Happens] \wedge \Xi \tag{Ex6.11}$$

Ξ is a conjunction of (Ex6.1–6.4). This minimization is equivalent to formulae (Ex6.12–6.14) conjoined with Ξ.

$$Initiates(a,f,t) \equiv ((a{=}Open1 \wedge f{=}In1) \vee (a{=}Open2 \wedge f{=}In2))\leftarrow \tag{Ex6.12}$$

$$Terminates(a,f,t) \equiv ((a{=}Close1 \wedge f{=}In1) \vee (a{=}Close2 \wedge f{=}In2))\leftarrow \tag{Ex6.13}$$

$$Happens(a,t) \equiv ((a{=}Open1 \wedge t{=}3) \vee (a{=}Close2 \wedge t{=}5))\leftarrow \tag{Ex6.14}$$

Direct action separated causation description theory C_{Sd} is produced by Definition 17, while the indirect action separated causation point theory C_{Si} is produced by Definition 18. (Ex6.15) is one example axiom from the C_{Si} set, written down for illustrative purposes.

$$\begin{aligned} &TerminationPoint_{In3}(In3, t)\leftarrow \\ &\quad [InitiationPoint_{In1}(In1, t) \wedge HoldsAt(In2, t) \wedge \neg TerminationPoint_{In2}(In2, t)] \vee \\ &\quad [InitiationPoint_{In2}(In2, t) \wedge HoldsAt(In1, t) \wedge \neg TerminationPoint_{In1}(In1, t)] \vee \\ &\quad [InitiationPoint_{In1}(In1, t) \wedge InitiationPoint_{In2}(In2, t)]\leftarrow \end{aligned} \tag{Ex6.15}$$

Representing (Ex6.15) in clausal form yields the following three rules (Ex6.16).

$$TerminationPoint_{In3}(In3, t) \leftarrow [InitiationPoint_{In1}(In1, t) \wedge HoldsAt(In2, t) \leftarrow \wedge \neg TerminationPoint_{In2}(In2, t)] \leftarrow$$
$$TerminationPoint_{In3}(In3, t) \leftarrow [InitiationPoint_{In2}(In2, t) \wedge HoldsAt(In1, t) \leftarrow \wedge \neg TerminationPoint_{In1}(In1, t)] \leftarrow$$
$$TerminationPoint_{In3}(In3, t) \leftarrow [InitiationPoint_{In1}(In1, t) \wedge InitiationPoint_{In2}(In2, t)] \leftarrow \tag{Ex6.16}$$

The ramification constraint for the AND gate is expressed using a pair of axioms comprising a state constraint and an effect constraint. First, to demonstrate that this class of domain can be represented, a stratification assignment can be chosen pursuant to Definition 20, expressed by the function $s : \Phi \mapsto \mathcal{N}$. An example choice of function s would be the following.

$$s = \{(In1, 0), (In2, 0), (In3, 1), (Light, 2)\} \leftarrow$$

C_{ECR} is then defined by Definition 20 (causal theory) to be $C_{Sd} \cup C_{Si}$ modified by function s, resulting in (Ex6.17–6.23). Beginning with causation relations describing transitions occurring due to ramification constraints, (Ex1.9) is a ternary ramification constraint that maps to the (Ex6.18) causation formula. (Ex6.17) is mapped from direct actions (Ex1.3–1.8) and follows directly from C_{Sd}.

$$InitiationPoint_0(In1, 3) \wedge TerminationPoint_0(In2, 5) \leftarrow \tag{Ex6.17}$$

$$\begin{aligned} &TerminationPoint_1(In3, t) \leftarrow \\ &\quad [InitiationPoint_0(In1, t) \wedge HoldsAt(In2, t) \wedge \neg TerminationPoint_0(In2, t)] \vee \\ &\quad [InitiationPoint_0(In2, t) \wedge HoldsAt(In1, t) \wedge \neg TerminationPoint_0(In1, t)] \vee \\ &\quad [InitiationPoint_0(In1, t) \wedge InitiationPoint_0(In2, t)] \leftarrow \end{aligned} \tag{Ex6.18}$$

(Ex1.10) and (Ex1.11) are simple ramification constraints, and map to (Ex6.19 and 6.20).

$$InitiationPoint_1(In3, t) \leftarrow \quad InitiationPoint_0(In1, t) \leftarrow \tag{Ex6.19}$$

$$InitiationPoint_1(In3, t) \leftarrow \quad TerminationPoint_0(In2, t) \leftarrow \tag{Ex6.20}$$

(Ex1.12–1.14) are similar in mapping structure to (Ex1.9–1.11), and produce similar formulae to those above. (Ex1.12–1.14) map to (Ex6.21–6.23) respectively.

$$\begin{aligned} &InitiationPoint_2(Light, t) \leftarrow \\ &\quad [InitiationPoint_0(In1, t) \wedge HoldsAt(In3, t) \wedge \neg TerminationPoint_1(In3, t)] \vee \\ &\quad [InitiationPoint_1(In3, t) \wedge HoldsAt(In1, t) \wedge \neg TerminationPoint_0(In1, t)] \vee \\ &\quad [InitiationPoint_0(In1, t) \wedge InitiationPoint_1(In3, t)] \leftarrow \end{aligned} \tag{Ex6.21}$$

$$TerminationPoint_2(Light, t) \leftarrow \quad TerminationPoint_0(In1, t) \leftarrow \tag{Ex6.22}$$

$$TerminationPoint_2(Light, t) \leftarrow \quad TerminationPoint_1(In3, t) \leftarrow \tag{Ex6.23}$$

Domain causal minimization is achieved using the prioritized Circumscription of (C6), instantiating to (Ex6.24) in this example. (Ex6.25) represents the same causal minimization expressed as hierarchical circumscription, generated from (C7).

$$\begin{aligned}\text{CIRC}[\text{Ex6.17–23}\ ;\ & \\ & \mathit{InitiationPoint}_0,\ \mathit{TerminationPoint}_0\ >\psi \\ & \mathit{InitiationPoint}_1,\ \mathit{TerminationPoint}_1\ >\psi \\ & \mathit{InitiationPoint}_2,\ \mathit{TerminationPoint}_2\ ]\end{aligned} \tag{Ex6.24}$$

$$\begin{aligned}\text{CIRC}[\text{Ex6.17–23}\ ;\ & \\ & \mathit{InitiationPoint}_0,\ \mathit{TerminationPoint}_0; \\ & \mathit{InitiationPoint}_1,\ \mathit{TerminationPoint}_1 \\ & \mathit{InitiationPoint}_2,\ \mathit{TerminationPoint}_2] \\ \wedge\ \text{CIRC}[\text{Ex6.17–23}\ ;\ & \\ & \mathit{InitiationPoint}_1,\ \mathit{TerminationPoint}_1; \\ & \mathit{InitiationPoint}_2,\ \mathit{TerminationPoint}_2] \\ \wedge\ \text{CIRC}[\text{Ex6.17–23}\ ;\ & \\ & \mathit{InitiationPoint}_2,\ \mathit{TerminationPoint}_2]\end{aligned} \tag{Ex6.25}$$

The causal minimization (Ex6.25) for this example domain, reduces to first order logic (Ex6.26–6.31).

$$\mathit{InitiationPoint}_0(f,t)\ \equiv\ \exists a.(\mathit{Happens}(a,t)\ \wedge\ \mathit{Initiates}(a,f,t))\leftarrow \tag{Ex6.26}$$

$$\mathit{TerminationPoint}_0(f,t)\ \equiv\ \exists a.(\mathit{Happens}(a,t)\ \wedge\ \mathit{Terminates}(a,f,t))\leftarrow \tag{Ex6.27}$$

$$\begin{aligned}&\mathit{InitiationPoint}_1(f,t)\ \equiv((f{=}\mathit{In}3)\ \wedge\ \mathit{InitiationPoint}_0(\mathit{In}1,t))\ \vee\leftarrow \\ &\quad((f{=}\mathit{In}3)\ \wedge\ \mathit{TerminationPoint}_0(\mathit{In}2,t))\leftarrow\end{aligned} \tag{Ex6.28}$$

$$\begin{aligned}&\mathit{TerminationPoint}_1(f,t)\ \equiv\ (f{=}\mathit{In}3)\ \wedge\ (\ \leftarrow \\ &\quad[\mathit{InitiationPoint}_0(\mathit{In}1,t)\ \wedge\ \mathit{HoldsAt}(\mathit{In}2,t)\ \wedge\neg\mathit{TerminationPoint}_0(\mathit{In}2,t)]\ \vee \\ &\quad[\mathit{InitiationPoint}_0(\mathit{In}2,t)\ \wedge\ \mathit{HoldsAt}(\mathit{In}1,t)\ \wedge\neg\mathit{TerminationPoint}_0(\mathit{In}1,t)]\ \vee \\ &\quad[\mathit{InitiationPoint}_0(\mathit{In}1,t)\ \wedge\ \mathit{InitiationPoint}_0(\mathit{In}2,t)]\ )\leftarrow\end{aligned} \tag{Ex6.29}$$

$$\begin{aligned}&\mathit{InitiationPoint}_2(f,t)\ \equiv\ (f{=}\mathit{Light})\ \wedge\ (\ \leftarrow \\ &\quad[\mathit{InitiationPoint}_0(\mathit{In}1,t)\ \wedge\ \mathit{HoldsAt}(\mathit{In}3,t)\ \wedge\neg\mathit{TerminationPoint}_1(\mathit{In}3,t)]\ \vee \\ &\quad[\mathit{InitiationPoint}_1(\mathit{In}3,t)\ \wedge\ \mathit{HoldsAt}(\mathit{In}1,t)\ \wedge\neg\mathit{TerminationPoint}_0(\mathit{In}1,t)]\ \vee \\ &\quad[\mathit{InitiationPoint}_0(\mathit{In}1,t)\ \wedge\ \mathit{InitiationPoint}_1(\mathit{In}3,t)]\ )\leftarrow\end{aligned} \tag{Ex6.30}$$

$$\begin{aligned}&\mathit{TerminationPoint}_2(f,t)\ \equiv((f{=}\mathit{Light})\ \wedge\ \mathit{TerminationPoint}_0(\mathit{In}1,t))\ \vee\leftarrow \\ &\quad((f{=}\mathit{Light})\ \wedge\ \mathit{TerminationPoint}_1(\mathit{In}3,t))\leftarrow\end{aligned} \tag{Ex6.31}$$

Combining each stratum's causal *InitiationPoint* and *TerminationPoint* predicates, according to (C8 and 9), yields a description of full domain causality, (contingent upon events), as described in (Ex6.32 and 6.33).

$$\begin{aligned}\mathit{InitiationPoint}(a,f,t)\ \equiv\ & \mathit{InitiationPoint}_0(a,f,t)\vee\leftarrow \\ & \mathit{InitiationPoint}_1(a,f,t)\vee \\ & \mathit{InitiationPoint}_2(a,f,t)\leftarrow\end{aligned} \tag{Ex6.32}$$

$$\begin{aligned}\mathit{TerminationPoint}(a,f,t)\ \equiv\ & \mathit{TerminationPoint}_0(a,f,t)\vee\leftarrow \\ & \mathit{TerminationPoint}_1(a,f,t)\vee \\ & \mathit{TerminationPoint}_2(a,f,t)\leftarrow\end{aligned} \tag{Ex6.33}$$

So far, only dynamic triggered changes have been described. To complete the translation of the domain description, we must add the static constraints described in (Ex1.9–1.14).

Their mapping is described by S_{ECR}, within Definition 21 which generate (Ex6.34–6.39) in static constraints. We have thereby specified what resembles contrapositive conditions accruing through the logic gates' functionality, both as causation axioms, and now, their associated state constraints.

$$\neg HoldsAt(In3, t) \leftarrow \quad HoldsAt(In1, t) \wedge HoldsAt(In2, t) \leftarrow \qquad \text{(Ex6.34)}$$

$$HoldsAt(In3, t) \ \leftarrow \ \neg HoldsAt(In1, t) \leftarrow \qquad \text{(Ex6.35)}$$

$$HoldsAt(In3, t) \ \leftarrow \ \neg HoldsAt(In2, t) \leftarrow \qquad \text{(Ex6.36)}$$

$$HoldsAt(Light, t) \leftarrow \quad HoldsAt(In1, t) \wedge HoldsAt(In3, t) \leftarrow \qquad \text{(Ex6.37)}$$

$$\neg HoldsAt(Light, t) \ \leftarrow \ \neg HoldsAt(In1, t) \leftarrow \qquad \text{(Ex6.38)}$$

$$\neg HoldsAt(Light, t) \ \leftarrow \ \neg HoldsAt(In3, t) \leftarrow \qquad \text{(Ex6.39)}$$

We may now prove that under initial conditions, the light is off, but most significantly,[9] when action Open1 takes place, the light remains off. Then, when Close2 is performed, the light turns on. The example shows the logic theory to be free of unwanted propagations that can occur in the presence of mutually cancelling effect domains.

$$\neg HoldsAt(Light, 2) \leftarrow \qquad \text{(Ex6.40)}$$

$$\neg HoldsAt(Light, 4) \leftarrow \qquad \text{(Ex6.41)}$$

$$HoldsAt(Light, 6) \leftarrow \qquad \text{(Ex6.42)}$$

(Ex6.40) is proved simply using initial condition (Ex6.3) along with static constraint (Ex6.38). (Ex6.41) and (Ex6.42) are proved using both static constraints (Ex6.34–6.39) and dynamic transitions (Ex6.26–6.33) along with initial conditions (Ex6.3–6.4), event occurrence (Ex6.14) and EC axioms (EC).

5.2 Gearwheel domain

The Gearwheel domain example is significant in that its fluents cannot be partitioned into sets of fluents with primary and derived effects. Each fluent may affect the other, and in turn be the recipient of a direct action's effect. From an intuitive point of view, the causality path is clear. However, just as no fluent is solely primary or derived, the fluents also cannot be laid out in a linear causation chain. As a consequence, proper stratification is not possible using any form of description where the causation paths are stated directly in fluent ramification terms (such as language $\mathcal{E}$ constructs). However, a finer-grained description of propagated causality is produced by the translation of language $\mathcal{E}$ ramifications into EC-R. Owing to its explicit representation of the non-existence of specific named effects through negation, and the use of negation for no other purpose, the stratification test on such a translated theory becomes an appropriate discriminator for intuitively meaningful (non-negative cycling) ramification domains. Figure 5 is axiomatized here to illustrate how such a domain class is handled in EC-R.

[9]This is the case where instantaneous propagated effects may provoke a wrong conclusion.

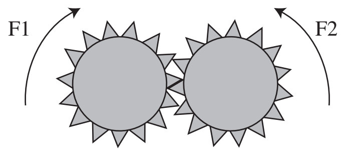

FIGURE 5. Gearwheel domain.

Unique names for fluents and actions are stated in (Ex7.1 and 7.2).

$$\text{UNA[F1, F2]} \tag{Ex7.1}$$

$$\text{UNA[Push1, Push2, Stop1, Stop2]} \tag{Ex7.2}$$

The domain's two fluents have initial conditions of stationary (Ex7.3 and 7.4).

$$\neg HoldsAt(F1, 1) \leftarrow \tag{Ex7.3}$$

$$\neg HoldsAt(F2, 1) \leftarrow \tag{Ex7.4}$$

Causal relations describing direct actions for the initiation and termination of both fluents $F1$ and $F2$ are defined in (Ex7.5–7.8). The action narrative used for example purposes is chosen to be *Push*1 followed by *Stop*1, (Ex7.9–7.10).

$$Initiates(Push1, F1, t) \leftarrow \tag{Ex7.5}$$

$$Initiates(Push2, F2, t) \leftarrow \tag{Ex7.6}$$

$$Terminates(Stop1, F1, t) \leftarrow \tag{Ex7.7}$$

$$Terminates(Stop2, F2, t) \leftarrow \tag{Ex7.8}$$

$$Happens(Push1, 3) \leftarrow \tag{Ex7.9}$$

$$Happens(Stop2, 5) \leftarrow \tag{Ex7.10}$$

Domain closure for direct effect causation and event occurrence are defined by (DD1). In this example, (DD1) is instantiated to become (Ex7.11), where Ξ is a conjunction of (Ex7.1–7.4).

$$\text{CIRC[Ex7.5–8; } Initiates, Terminates] \wedge \text{CIRC[Ex7.9-10 ; } Happens] \wedge \Xi \tag{Ex7.11}$$

The (Ex7.11) minimization is equivalent to first-order completion formulae (Ex7.12 and 7.14) conjoined with Ξ.

$$Initiates(a, f, t) \equiv ((a{=}Push1 \wedge f{=}F1) \vee (a{=}Push2 \wedge f{=}F2)) \leftarrow \tag{Ex7.12}$$

$$Terminates(a, f, t) \equiv ((a{=}Stop1 \wedge f{=}F1) \vee (a{=}Stop2 \wedge f{=}F2)) \leftarrow \tag{Ex7.13}$$

$$Happens(a, t) \equiv ((a{=}Push1 \wedge t{=}3) \vee (a{=}Stop2 \wedge t{=}5)) \leftarrow \tag{Ex7.14}$$

Ramifications for this domain are described by the following **whenever** statements.

F_1 **whenever** F_2
$\neg F_1$ **whenever** $\neg F_2$
F_2 **whenever** F_1
$\neg F_2$ **whenever** $\neg F_1$

Direct action separated causation description theory C_{Sd} is produced by Definition 17 and (Ex7.15) follows directly from C_{Sd}. The indirect action separated causation point theory C_{Si} produced by Definition 18 is then given by (Ex7.16–7.19).

$$InitiationPoint_{F1}(F1, 3) \ \wedge \ \ TerminationPoint_{F2}(F2, 5) \leftarrow \tag{Ex7.15}$$

$$InitiationPoint_{F1}(F1, t) \leftarrow \quad InitiationPoint_{F2}(F2, t) \leftarrow \tag{Ex7.16}$$

$$TerminationPoint_{F1}(F1, t) \leftarrow \quad TerminationPoint_{F2}(F2, t) \leftarrow \tag{Ex7.17}$$

$$InitiationPoint_{F2}(F2, t) \leftarrow \quad InitiationPoint_{F1}(F1, t) \leftarrow \tag{Ex7.18}$$

$$TerminationPoint_{F2}(F2, t) \leftarrow \quad TerminationPoint_{F1}(F1, t) \leftarrow \tag{Ex7.19}$$

C-stratification of theory $C_{Sd} \cup C_{Si}$ is given by Definition 19 using the level mapping function $s = \{(F1, 0), (F2, 0)\}$. C_{ECR} is then produced through Definition 20 using the stratification function. C_{ECR} is identical in structure to $C_{Sd} \cup C_{Si}$, with the exception that predicates have been renamed to create a fully stratified causal theory for the domain (Ex7.20–7.24).

$$InitiationPoint_0(F1, 3) \ \wedge \ \ TerminationPoint_0(F2, 5) \leftarrow \tag{Ex7.20}$$

$$InitiationPoint_0(F1, t) \leftarrow \quad InitiationPoint_0(F2, t) \leftarrow \tag{Ex7.21}$$

$$TerminationPoint_0(F1, t) \leftarrow \quad TerminationPoint_0(F2, t) \leftarrow \tag{Ex7.22}$$

$$InitiationPoint_0(F2, t) \leftarrow \quad InitiationPoint_0(F1, t) \leftarrow \tag{Ex7.23}$$

$$TerminationPoint_0(F2, t) \leftarrow \quad TerminationPoint_0(F1, t) \leftarrow \tag{Ex7.24}$$

Prioritized circumscription (C6) instantiates to (Ex7.25), which in turn reduces to first-order formulae (Ex7.26 and 7.27).

$$\mathrm{CIRC}[\mathrm{Ex7.20–24}\ ;\ InitiationPoint_0,\ TerminationPoint_0] \tag{Ex7.25}$$

$$InitiationPoint_0(f, t) \ \equiv \ ((f{=}F1 \ \wedge \ t{=}3) \vee (f{=}F2 \ \wedge \ t{=}3)) \leftarrow \tag{Ex7.26}$$

$$TerminationPoint_0(f, t) \ \equiv \ ((f{=}F1 \ \wedge \ t{=}5) \vee (f{=}F2 \ \wedge \ t{=}5)) \leftarrow \tag{Ex7.27}$$

(C8 and 9) applied to (Ex7.26 and 7.27) finish the causation description for the domain. Only static constraints remain. Definition 21's *SECR* contains the following constraints (Ex7.28–31) for this domain.

$$HoldsAt(F1, t) \leftarrow \quad HoldsAt(F2, t) \leftarrow \tag{Ex7.28}$$

$$\neg HoldsAt(F1, t) \ \leftarrow \ \neg HoldsAt(F2, t) \leftarrow \tag{Ex7.29}$$

$$HoldsAt(F2, t) \leftarrow \quad HoldsAt(F1, t) \leftarrow \tag{Ex7.30}$$

$$\neg HoldsAt(F2, t) \ \leftarrow \ \neg HoldsAt(F1, t) \leftarrow \tag{Ex7.31}$$

It should now be clear from inspection that we may use the domain description (Ex7.26–7.31) developed above, along with the Event Calculus axioms to perform general reasoning and inference about the state of the domain at time points along the timeline in response to particular initial conditions and events.

This example illustrates how the class of interacting fluent problems such as the Gearwheel domain are handled correctly under the stratified EC-R causal description, while the use of direct stratification of fluents in $\mathcal{E}$'s ramification conditions is not so. Stratification is a syntactical condition, and is thus only able to be conditioned upon what is expressed explicitly. The classical logic causal description following translation to EC-R is at a lower level of abstraction, and consequently more expressive than $\mathcal{E}$'s. It states explicitly in syntax what $\mathcal{E}$ contains in its semantics, and takes account of the fact that domain stability is conditional upon the polarity of fluent dependency (positive or negative) and crucially, specifies explicitly where causation point propagations must not exist.

In the literature, another approach to stratification of ramification theories is described in [45]. Due to the choice of representation of ramifications at the stratification stage, the stratification pre-condition necessarily becomes stronger overall than it is for EC-R. Only acyclic domains are accepted, and no positive recursion is permitted. This prevents the formalism from accepting the mutually coupled instantaneously propagated effect class of domain exemplified here using gearwheels. In EC-R it may be helpful to note that stratification may allocate the same stratum number to multiple fluent predicates. This is actually required to properly represent domains of the type here.

The above described translation from language $\mathcal{E}$ ramification domains, into classical logic EC-R can be proved to be sound and complete with respect to $\mathcal{E}$'s semantics. A full proof is undertaken in the next sections.

6 A correspondence result for $\mathcal{E}$ and EC-R

In this section we build a semantic correspondence between a suitable instantiation of EC-R and language $\mathcal{E}$. The reason for pursuing such a path is to exploit as far as possible the understanding of correct reasoning in ramification domains embodied in $\mathcal{E}$ to demonstrate that EC-R handles ramifications properly.

6.1 *Semantic constraints for translation*

To begin, we first give a set of definitions for conditions (assumptions), which if met, can be used to show that the respective domain description theories of $\mathcal{E}$ and EC-R are equivalent. Definition 22 is a condition for uniqueness of names in T_{EC} whereby in all its models, T_{EC} maintains uniqueness of names for the fluents and actions referred to in D.

Definition 22 (Name-matches)
D names-matches T_{EC} iff for every model $\mathcal{M}$ of T_{EC}, for every F, $F' \in \Phi$, $A, A' \in \Delta$, the following conditions hold,

- if $F \neq F'$ then $\|F\|_{\mathcal{M}} \neq \|F'\|_{\mathcal{M}}$ and
- if $A \neq A'$ then $\|A\|_{\mathcal{M}} \neq \|A'\|_{\mathcal{M}}$.

Definitions 23–25 establishes an isomorphism condition for direct actions. The conditions are expressed relative to the interpretation of *HoldsAt*.

Definition 23 (h-satisfies)
Given a model $\mathcal{M}$ of T_{EC}, a time-point $T \in \Pi$, a set $C \subseteq \Phi^{\pm}$ of language $\mathcal{E}$ fluent literals, then $\mathcal{M}$ h-satisfies C at T iff for all $F \in \Phi$, the following conditions hold,

- if $F \in C$ then $(\|F\|_{\mathcal{M}}, T) \in \|HoldsAt\|_{\mathcal{M}}$, and
- if $\neg F \in C$ then $(\|F\|_{\mathcal{M}}, T) \notin \|HoldsAt\|_{\mathcal{M}}$.

Definition 24 (Initiates-matches)
D Initiates-matches T_{EC} iff for every model $\mathcal{M}$ of T_{EC}, every time point T and every action $\alpha\psi$ and fluent $\phi\psi$ in the domain of discourse of $\mathcal{M}$, $(\alpha, \phi, T) \in \|Initiates\|_{\mathcal{M}}$ iff there exists $F \in \Phi, A \in \Delta$ and $C \subseteq \Phi^{\pm}$ such that the following conditions hold,

- $\alpha = \|A\|_{\mathcal{M}}, \phi = \|F\|_{\mathcal{M}}$,
- $\mathcal{M}$ h-satisfies C at T,
- '*A* **initiates** *F* **when** *C*' $\in\gamma$.

Definition 25 (Terminates-matches)
D Terminates-matches T_{EC} iff for every model $\mathcal{M}$ of T_{EC}, every time point T and every action $\alpha\psi$and fluent $\phi\psi$in the domain of discourse of $\mathcal{M}$, $(\alpha, \phi, T) \in \|Terminates\|_{\mathcal{M}}$ iff there exists $F \in \Phi, A \in \Delta$ and $C \subseteq \Phi^{\pm}$ such that,

- $\alpha = \|A\|_{\mathcal{M}}, \phi = \|F\|_{\mathcal{M}}$,
- $\mathcal{M}$ h-satisfies C at T,
- '*A* **terminates** *F* **when** *C*' $\in \gamma$.

Definitions 26–29 provide the isomorphism conditions for ramification constraints. Before defining the ramification mapping, Definition 26 is needed as a sub-definition specifying four cases where triggering occurs.

Definition 26 (Trigger-subconditions)
For time point $T \in \Pi$ and $F \in \Phi$ in the domain of discourse of $\mathcal{M}$ let statements be defined as follows:

- $\Sigma_i^{\mathcal{M}}(F, T) = (\|F\|_{\mathcal{M}}, T) \in \|InitiationPoint_{s(F)}\|_{\mathcal{M}}$.
- $\Sigma_t^{\mathcal{M}}(F, T) = (\|F\|_{\mathcal{M}}, T) \in \|TerminationPoint_{s(F)}\|_{\mathcal{M}}$.
- $\Omega_{hp}^{\mathcal{M}}(F, T) = \mathcal{M}$ h-satisfies F at T and $(\|F\|_{\mathcal{M}}, T) \notin \|TerminationPoint_{s(F)}\|_{\mathcal{M}}$.
- $\Omega_{hn}^{\mathcal{M}}(F, T) = \mathcal{M}$ does not h-satisfy F at T and $(\|F\|_{\mathcal{M}}, T) \notin \|InitiationPoint_{s(F)}\|_{\mathcal{M}}$.

Definition 27 then conveniently specifies the resulting subdefinition for trigger points of a whenever statement '*L* **whenever** *C*'$\in \rho$.

Definition 27 (Triggered)
For a set C of conditional fluent literals such that $C \neq \emptyset$, let the following subset of a powerset partition be defined between the locally-scoped variables.

$$P'(C) = \{ (X, Y) : X \cup Y = C, X \cap Y = \emptyset, X \neq \emptyset \} \leftarrow$$

C is triggered at time point T with respect to $\mathcal{M}$ iff there exists $(C_1, C_2) \in P'(C)$ (where C_1 is the set of fluent literals about to be satisfied, and C_2 the set of fluent literals already satisfied) such that both the following conditions apply,

- For all $F \in C_1$, it is the case that $\Sigma_i^{\mathcal{M}}(F, T)$ applies, and for all $F \in C_2$, it is the case that $\Omega_{hp}^{\mathcal{M}}(F, T)$ applies.
- For all $\neg F \in C_1$, it is the case that $\Sigma_t^{\mathcal{M}}(F, T)$ applies, and for all $\neg F \in C_2$, it is the case that $\Omega_{hn}^{\mathcal{M}}(F, T)$ applies.

Definitions 26 and 27 can now specify the isomorphism conditions for ramification constraints in terms of trigger points.

Definition 28 (Whenever-maps$_i$)
D whenever-maps$_i$ T_{EC} iff for every model $\mathcal{M}$ of T_{EC}, for every time point T and every fluent ϕ in the domain of discourse of $\mathcal{M}$ the following conditions apply:

(1) $(\phi, T) \in \|InitiationPoint_{s(F)}\|_{\mathcal{M}}$ iff there exists $F \in \Phi$ and $C \subseteq \Phi^{\pm}$ such that,

(a) $\phi = \|F\|_{\mathcal{M}}$,
(b) *There exists* 'F **whenever** C'$\in \rho$, such that C is triggered at T with respect to $\mathcal{M}$.

(2) *For every* 'F **whenever** C' $\in \rho$, if $\mathcal{M}$ h-satisfies C at T then $\mathcal{M}$ h-satisfies F at T.

Definition 29 (Whenever-maps$_t$)
D whenever-maps$_t$ T_{EC} iff for every model $\mathcal{M}$ of T_{EC}, for every time point T and every fluent ϕ in the domain of discourse of $\mathcal{M}$ the following conditions apply:

(1) $(\phi, T) \in \|TerminationPoint_{s(F)}\|_{\mathcal{M}}$ iff there exists $F \in \Phi$ and $C \subseteq \Phi^{\pm}$ such that,

(a) $\phi = \|F\|_{\mathcal{M}}$,
(b) There exists '$\neg F$ **whenever** C' $\in \rho$, such that C is Triggered at T with respect to $\mathcal{M}$.

(2) *For every* '$\neg F$ **whenever** C' $\in \rho$, if $\mathcal{M}$ h-satisfies C at T then $\mathcal{M}$ does not h-satisfy F at T.

The final isomorphism condition, Definition 30, covers event occurrence.

Definition 30 (Happens-matches)
D Happens-matches T_{EC} iff for every model $\mathcal{M}$ of T_{EC}, every time point T and every action α in the domain of discourse of $\mathcal{M}$, $(\alpha, T) \in \|Happens\|_{\mathcal{M}}$ iff there exists A $\in\Delta$ such that $\alpha = \|A\|_{\mathcal{M}}$ and 'A **happens-at** T' $\in\eta$.

Definitions 31, 32 and 33 state together that the constraints imposed on *HoldsAt* by T_{EC} are equivalent to those imposed on $\mathcal{E}$'s interpretations by t-propositions. The extent of *HoldsAt* is increased by inference about events through the EC axioms. Definition 32 forms a function $\mathcal{E}$-projection whose purpose is to define a model of $\mathcal{E}$ equivalent to initial conditions as expressed in T_{EC}. A holds-matches condition is then expressed chiefly in condition 2 of Definition 33 after a condition for matching fluents whose effect is to make condition 2 a complete condition.

Definition 31 (t-model)
An interpretation H of $\mathcal{E}$ is a t-model of D iff for every $F \in \Phi$ and $T, T' \in \Pi$, the following conditions hold,

- for all t-propositions in τ of the form 'F **holds-at** T', $H(F, T) = true$.
- for all t-propositions in τ of the form '$\neg F$ **holds-at** T', $H(F, T) = false$,
- for all r-propositions in ρ of the form 'L **whenever** C', if H satisfies C at T then H satisfies $\{L\}$ at T.

Definition 32 ($\mathcal{E}$-projection)
An $\mathcal{E}$-projection of a model $\mathcal{M}$ of T_{EC} is defined as the following language $\mathcal{E}$ interpretation $H_{\mathcal{M}}$,

$$H_{\mathcal{M}}(F, T) = \{true \text{ if } (\|F\|_{\mathcal{M}}, T) \in \|HoldsAt\|_{\mathcal{M}} \; false \text{ otherwise}\}.$$

Definition 33 (Holds-matches)
D holds-matches T_{EC} iff for every model $\mathcal{M}$ of T_{EC}, the following conditions are satisfied,

(1) For every fluent $\phi\psi$in the domain of discourse of $\mathcal{M}$ there exists $F \in \Phi$ such that $\phi = \|F\|_{\mathcal{M}}$.
(2) The $\mathcal{E}$-projection of $\mathcal{M}$ is a t-model of D.
(3) For every t-model H^t of D there is a model $\mathcal{M}\overset{H^t}{\leftarrow}$of T_{EC} which differs from $\mathcal{E}$ only in the interpretation of *HoldsAt* and is such that H^t is the $\mathcal{E}$-projection of $\mathcal{M}\overset{H^t}{\leftarrow}$.

The final (composite) matching condition is stated in Definition 34.

Definition 34 (Matches)
D matches T_{EC} iff D name-matches, initiates-matches, terminates-matches, whenever-maps$_i$, whenever-maps$_t$, happens-matches and holds-matches T_{EC}.

6.2 Causal correspondence

As a prolog to the main correspondence theorem, this section will demonstrate that a causal correspondence exists between the definition of causal points (i.e. initiation point and termination points) in $\mathcal{E}$'s semantics and those in the Event Calculus semantics. It is the causation point component of the theory which defines transitions of fluent states in response to events and fluents in the environment.

We saw previously that a causation operator $\mathcal{F}$ is induced by the conditions of Definition 10 in $\mathcal{E}$'s semantic specification. We may demonstrate causal correspondence by using induction on successive levels of indirect effect (i.e. n-steps of causation), knowing that there exists a limit ordinal where the least fixed point of $\mathcal{F}$ is reached. This is sufficient to demonstrate that the two theories correspond fully in terms of causation. It is convenient to choose the stratum assignment number to achieve this indexing.

To demonstrate causal correspondence, we must show that the operator $\mathcal{F}_\infty$ corresponds to the unique minimal model of T_{EC}. This requires the descriptions of causation points contained in the extensions of *InitiationPoint* and *TerminationPoint* predicates to be equivalent to those causation points described in $\mathcal{F}_\infty$. This correspondence will be shown using induction on the stratification index. An operator with n steps of causation is represented by $\mathcal{F}_n$.

First, we must unpack the composite operator $\mathcal{F}$ so as to separate the initiation from the termination causation points.

$$\mathcal{F}_n^{init} = \{(F, T) \mid F \in \Phi, T \in \Pi, (F, T, init) \in \mathcal{F}_n\}$$

$$\mathcal{F}_n^{term} = \{(F, T) \mid F \in \Phi, T \in \Pi, (F, T, term) \in \mathcal{F}_n\}\leftarrow$$

Assuming a language $\mathcal{E}$ domain with definition D, and a model $\mathcal{M}_C$ of EC causal axioms T_{EC} causally minimized by hierarchical circumscription (as defined previously) such that equivalence conditions of Definitions 22–34 apply, then causal correspondence is shown to hold if and only if Proposition 1 holds.

Using an inductive step for causal correspondence of indirect actions, we will then show that if there exists a correspondence for n-step causation then the (n+1) stage also corresponds. The fluents are taken in index order to ensure that the intermediate causations are equivalent.

PROPOSITION 1 (Causal correspondence)
For a domain D and a model $\mathcal{M}_C$ of T_{EC} where D matches T_{EC}, then:

$$\|InitiationPoint\|_{\mathcal{M}_C} = \mathcal{F}_{\infty}^{init}$$

$$\|TerminationPoint\|_{\mathcal{M}_C} = \mathcal{F}_{\infty\leftarrow}^{term}$$

PROOF. Proof is by induction over the stratification of fluents. If the base case is proved, and the inductive case is proved, then under the schema of induction, Proposition 1 holds.

(1) Base case: For a domain D, and any model $\mathcal{M}_C$ of T_{EC}, the base case holds iff the following propositions hold.

$$\|InitiationPoint_0\|_{\mathcal{M}_C} = \mathcal{F}_1^{init} \tag{a}$$

$$\|TerminationPoint_0\|_{\mathcal{M}_C} = \mathcal{F}_1^{term} \tag{b}$$

(2) Inductive case: For all $m \leq n$, a domain D, and any model $\mathcal{M}_C$ of T_{EC}, the inductive case holds iff assuming (c) and (d) then (e) and (f) follow.

$$\|InitiationPoint_m\|_{\mathcal{M}_C} = \mathcal{F}_{m+1}^{init} \tag{c}$$

$$\|TerminationPoint_m\|_{\mathcal{M}_C} = \mathcal{F}_{m+1}^{term} \tag{d}$$

$$\|InitiationPoint_{m+1}\|_{\mathcal{M}_C} = \mathcal{F}_{m+2}^{init} \tag{e}$$

$$\|TerminationPoint_{m+1}\|_{\mathcal{M}_C} = \mathcal{F}_{m+2}^{term} \tag{f}$$

Subproof of base case.

- Since Definitions 16 and 17, (C7), and D name-matches, initiates-matches, terminates-matches and happens-matches T_{EC}, then by Definition 10 part 1 (equivalently appendix, (FP1-2), (FP5) and Definition A1) for the Causation Operator, then (a) and (b) hold.

(End of subproof)

Subproof of inductive case.

- Since Definition 18, (C7), and D name-matches, whenever-maps$_i$ and whenever-maps$_t$ T_{EC}, then by Definition 10 part 2 (equivalently appendix (FP3-5) and Definition A1) for the Causation Operator, if (c) and (d) hold, then (e) and (f) hold.

(End of subproof)
(End of proof)

Therefore, by induction, together with the assured existence of a limit ordinal, the least fixed point is reached exactly when there is correspondence in causation points between the two formalisms. Therefore, the theories match completely in terms of *n*-point causation, and hence by (C8-9) fully causally-correspond, and Proposition 1 holds. ■

6.3 *Main proof of correspondence*

Making use of Proposition 1 for causal correspondence, we are now able to present a proof of full correspondence. This is achieved by further use of the previously defined isomorphisms between $\mathcal{E}$ and EC-R.

Proposition 2 (Full correspondence)
For all time points $T \in \Pi$ and for all fluents $F \in \Phi$ in the domain of discourse D such that D matches T_{EC}, then

$$D \models_{\mathcal{E}} F \textbf{ holds-at } T \text{ iff } T_{EC} \cup Det_{EC} \models HoldsAt(F, T) \leftarrow \tag{a}$$

$$D \models_{\mathcal{E}} \neg F \textbf{ holds-at } T \text{ iff } T_{EC} \cup Det_{EC} \models \neg HoldsAt(F, T) \leftarrow \tag{b}$$

Proof. For (a), it is sufficient to prove the following conditions:

(1) (Only-if case) If there exists a model H of D where $H(F, T) = true$, then there also exists a model $\mathcal{M}_H$ of $T_{EC} \cup Det_{EC}$ where $\mathcal{M}_H \Vdash HoldsAt(F, T)$.
(2) (If case) If there exists a model $\mathcal{M}$ of $T_{EC} \cup Det_{EC}$ such that $\mathcal{M} \Vdash HoldsAt(F, T)$, then there exists an $\mathcal{E}$-model $H_{\mathcal{M}}$ of D where $H_{\mathcal{M}}(F, T) = true$.

For (b), it is sufficient to prove the following:

(3) (Only-if case) If there exists a model H of D where $H(F, T) = false$, then there also exists a model $\mathcal{M}_H$ of $T_{EC} \cup Det_{EC}$ where $\mathcal{M}_H \Vdash \neg HoldsAt(F, T)$.
(4) (If case) If there exists a model $\mathcal{M}$ of $T_{EC} \cup Det_{EC}$ such that $\mathcal{M} \Vdash \neg HoldsAt(F, T)$, then there exists an $\mathcal{E}$-model $H_{\mathcal{M}}$ of D where $H_{\mathcal{M}}(F, T) = false$.

Subproof of condition (1):

- Suppose there exists a model H of D where $H(F, T) = true$.
- Since T_{EC} is consistent, then by definitions holds-matches, whenever-maps$_i$ and whenever-maps$_t$, there exists a model $\mathcal{M}_H$ of T_{EC} such that H is the $\mathcal{E}$-projection of $\mathcal{M}_H$.
- Then, $\mathcal{M}_H \Vdash HoldsAt(F, T)$.
- Since T_{EC} does not mention the predicates *Clipped*, *Declipped*, *StoppedIn*, *StartedIn*, then we can assume $\mathcal{M}_H$ satisfies (EC 1, 2, 7, 8).
- Since D name-matches, causal-corresponds and happens-matches T_{EC} then:
  - by condition 1 of Definition 11: model, $\mathcal{M}_H$ satisfies (EC5-6).
  - by condition 2 of Definition 11: model, $\mathcal{M}_H$ satisfies (EC3).
  - by condition 3 of Definition 11: model, $\mathcal{M}_H$ satisfies (EC4).
- Therefore $\mathcal{M}_H$ is a model of $T_{EC} \cup Det_{EC}$.

(End of subproof)

Subproof of condition (2):

- Suppose there exists a model $\mathcal{M}$ of $T_{EC} \cup Det_{EC}$ such that $\mathcal{M} \Vdash HoldsAt(F, T)$, then by definitions whenever-maps$_i$, whenever-maps$_t$ and holds-matches, then the $\mathcal{E}$-projection $H_{\mathcal{M}}$ of $\mathcal{M}$ is a t-model of D and so $H_{\mathcal{M}}(F, T) = true$.
- Having established an equivalence of point-wise constraints arising from initial conditions, it remains to show that $H_{\mathcal{M}}$ satisfies conditions 1, 2 and 3 of Definition 11: model.
- Since D name-matches, causal-corresponds, happens-matches T_{EC} then it follows from:
  - $\mathcal{M} \Vdash$ (EC3), that $H_{\mathcal{M}}$ satisfies condition 2 of {Def 11 model },
  - $\mathcal{M} \Vdash$ (EC4), that $H_{\mathcal{M}}$ satisfies condition 3 of {Def 11 model },
  - $\mathcal{M} \Vdash$ (EC5–6), that $H_{\mathcal{M}}$ satisfies condition 1 of {Def 11 model }.

(End of subproof)

Subproof of condition (3):

- This proof is identical to the proof of condition (1) with the exception that constant *'false'* replaces mentioned instances of *'true'* and literal '$\neg HoldsAt(F, T)$', replaces mentioned instances of '$HoldsAt(F, T)$'.

(End of subproof)

Subproof of condition (4):

- This proof is identical to the proof of condition (2) with the exception that literal '$\neg HoldsAt(F, T)$' replaces mentioned instances of '$HoldsAt(F, T)$' and constant *'false'* replaces mentioned instances of *'true'*.

(End of subproof)
(End of proof)

7 Related work

[59] presented a solution to the ramification problem that specifically used both causal minimization and event minimization to handle the indirect effect of actions. There are several examples presented in which the formalism is shown to be quite expressive, and able to handle domains with mutually interacting effects.

The theory is based on the intuitive notion of introducing new events that update each fluent whose value is dependent on other fluents. Further formulae are then added to ensure that the new events are triggered under appropriate conditions. This method has a similar effect to defining indirect causation points in EC-R, with the difference that Shanahan names indirect events explicitly.

However, under some circumstances, the theory does contain ambiguity in its construction of the physical effect propagation. A partial axiomatization of the domain in Figure 6 containing a single logical AND gate shows a case when this occurs. We choose to omit rules for the termination of F and $F1$, and the initiation of $F2$, for reasons of clarity in this example.

If we set up initial conditions (Ex8.1) so that $F1$ is true, while $F2$ and F are false, then we may restrict the analysis to merely consider the conditions under which F makes a transition from low to high.

$$HoldsAt(F1, t) \wedge \neg HoldsAt(F2, t) \wedge \neg HoldsAt(F, t) \leftarrow \qquad \text{(Ex8.1)}$$

According to Shanahan's [59] method, a new (conditional) event *F-on* is introduced which upon occurrence causes fluent F to be initiated. For the purposes of domain manipulation, we will also add two direct actions acting upon input fluents $F1$ and $F2$. (Ex8.2) contains the associated causation relations.

$$Initiates(A1, F1, t) \wedge Terminates(A2, F2, t) \wedge Initiates(F\text{-}on, F, t) \leftarrow \qquad \text{(Ex8.2)}$$

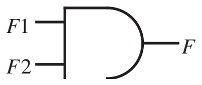

FIGURE 6. Logical AND gate domain

An unconditional event occurrence can be specified for demonstration purposes as $A1$ occurring at time 2 (Ex8.3). To specify the ramification constraint, a further conditional event is added that may occur depending on the state of fluents and other events (Ex8.4).

$$Happens(A1, 2) \leftarrow \tag{Ex8.3}$$

$$\begin{aligned} & Happens(F\text{-}on, t) \leftarrow \\ & \quad [\neg HoldsAt(F, t) \ \vee \ \exists a(Happens(a, t) \wedge Terminates(a, F, t))] \wedge \leftarrow \\ & \quad [HoldsAt(F1, t) \ \vee \ \exists a(Happens(a, t) \wedge Initiates(a, F1, t))] \wedge \leftarrow \\ & \qquad\qquad [\neg \exists a(Happens(a, t) \wedge Terminates(a, F1, t))] \wedge \leftarrow \\ & \quad [HoldsAt(F2, t) \ \vee \ \exists a(Happens(a, t) \wedge Initiates(a, F2, t))] \wedge \leftarrow \\ & \qquad\qquad [\neg \exists a(Happens(a, t) \wedge Terminates(a, F2, t))] \leftarrow \end{aligned} \tag{Ex8.4}$$

A solution to the frame problem is then created using circumscription for causal minimization on event occurrence, minimizing the *Happens* predicate: CIRC[Ex8.3–4 ; *Happens*]. However, the minimization is not unique, forming (Ex8.5 $\vee$ Ex8.6). In the first case (Ex8.5), action *F-on* is triggered at time 2 causing F to become true, however, in the second case (Ex8.6), instead action $A2$ is triggered causing $F2$ to transition to false, while F remains false.

$$Happens(a, t) \ \equiv \ ((a{=}A1 \wedge t{=}2) \ \vee \ (a{=}F\text{-}on \wedge t{=}2)) \leftarrow \tag{Ex8.5}$$

$$Happens(a, t) \ \equiv \ ((a{=}A1 \wedge t{=}2) \ \vee \ (a{=}A2 \wedge t{=}2)) \leftarrow \tag{Ex8.6}$$

Therefore, under this formalism, after action $A1$ occurs, $HoldsAt(F, 3)$ is not entailed. This is a typical competing models problem, as described earlier, trading action *F-on* with action $A2$. Due to the anomalous model occurring on event occurrence rather than causality, it is somewhat harder to see than was the case in the earlier illustrated example where the ambiguity occurred on Initiates and Terminates.

Forth and Shanahan [17] introduce a formalism for ramifications that provides a solution to this problem, using the method of propagated causal effects rather than propagated events as with [59]. However it is a less general solution than provided here. The restrictions are of two types.

First, concurrent or overlapping events are not accommodated. This is a consequence of actions being named explicitly. One action creates a propagated causal effect which can interact with its own ramifications, but not those of other actions. In the present article's formalism, indirect actions are not named explicitly, but rather describe ramifications in terms of causation points. Interaction of propagated effects may then be modelled irrespective of the action causing them. It is therefore possible for two different actions to interact within the same ramification chain, and concurrent actions are thereby supported.

Secondly, in order to provide a suitable pre-condition for the existence of a least fixed point, stratification in [17] is performed directly on a language $\mathcal{E}$←domain description. Although correct, this is overly restrictive in the class of domain descriptions permitted. In this article, by contrast, stratification is performed on a finer-grained causal theory formed after translation into classical logic. This provides a much closer match with the classes of domains that are meaningful in a practical causation sense. As a consequence of these limitations, [17] cannot support the benchmark positive cycling domain Gearwheel, as described in the prior example.

McIlraith [45] presents a method of realizing ramifications in the classical logical language of Situation Calculus [50, 51]. First-order successor state axioms are used as a means to solve the frame problem and an understanding of ramifications is gained through a least fixed point semantics. The formalism can reason about similar classes of domains as [17] above. A domain acceptance pre-condition depends on a direct stratification of fluents through solitary stratified theories. This imposes an acyclic fluent dependency requirement and thus makes the theory overly restrictive. Domains are not permitted to have cyclic fluent dependency, nor concurrent actions. Therefore, valid practical domains belonging to these classes (such as the Gearwheel example) also cannot be represented.

Derecker *et al.* [8] provides a treatment of ramifications in a custom language called $\mathcal{ER}$. This adopts Well-Founded semantics of logic programming [7] to provide an alternative justification of ramifications in which the static constraints can be specified independently from causation. The strength of this formalism is its flexibility in terms of domain acceptance. I inconsistent and unstable domains can be accepted for reasoning, possibly yielding a third level 'unknown' as an output. However, the method also contributes to a weakness because it does not identify and attempt to filter out causally non-sensical domain axiomatizations in a pre-reasoning phase.

In an earlier phase of literature Thielscher [64] ramification formalism (described in the Fluent Calculus) uses a least fixed point understanding of ramifications. Similarly to language $\mathcal{E}$, and [59], it does not directly specify the classes of domains acceptable for reasoning. Due to the least fixed point construction however, the domain specification must implicitly be assumed to be those for which a least fixed point exists. No pre-conditions are given for domain stability nor the associated existence of the least fixed point.

The focus of this article has been on examining how expressed relationships between fluents can be used to determine knock-on or indirect effects of actions. A related, orthogonal issue is how expressed constraints between fluents may be used to determine implicit or extra qualifications for actions. It is outside the intended focus of the article to examine this problem in detail, however we note in passing that extensions of the Event Calculus and its action language equivalents exist which allow this kind of expressivity, and which are perfectly compatible with the techniques included here describing indirect effects, see e.g. [28, 46]. For example [28] shows how to incorporate a 'Prevents' relation between fluents in an Event Calculus-like framework, to express this type of constraint.

Owing to the extent of the literature on formalizing dynamic domains using logical representations, we will here for the sake of completeness augment the citations already given by drawing attention to other related works. Prior to the formulization of Event Calculus in classical logic, the Situation Calculus was also extensively studied. See for example [3, 14, 26, 36, 47, 50, 54, 55]. Logic programming variants have also been developed in (for instance) [4, 15] which take advantage of results in logic programming semantics. Other types of formalizations include [18, 21, 49, 52]. Ramifications have also been addressed in [22, 23, 33, 41, 42, 44, 62]. Beyond general purpose logical representations (and in addition to $\mathcal{E}$), custom logics have been designed to use intuitively clear propositions to represent dynamic domains. [5, 6, 8, 19, 24, 63, 65] are additional references to this line of work.

8 Discussion

8.1 Frame problem

The frame problem was first recognized by [43] and has had a long history of research developments. In essence, the problem is that in a declarative action formalism, a challenge was observed to occur in defining succinctly the large number of values in a domain that do not change as a result of an action. This challenge is important for two reasons. First, a domain description must be created, possibly by hand, requiring that the number of explicit axioms be maintained at a manageable level. Secondly, computational algorithms cannot handle an unbounded number of explicit axioms. A means had to be found therefore to represent and reason about domains in a way that overcame this inherent difficulty. A good review of the Frame Problem and its developments can be found in [57].

Event Calculus uses a non-monotonic logic formalism to make a complete information assumption on the effects of actions, defined through DDA. This (in combination with EC axioms) results in a default assumption of the persistence of fluents. The principle of persistence states that unless a given action specifically affects change through initiation or termination of a fluent, that fluent will retain its prior value.

This is an instance of one of two major approaches to the frame problem employed in the literature. One of these approaches is based on non-monotonic logic, while the other is based on monotonic logic. Event Calculus along with (for example) [35, 40, 44] employ non-monotonicity to make a dynamic complete-knowledge assumption for causation, along with monotonic constructs (DIA in the case of the Event Calculus) to formalize frame axioms specifying the persistence characteristics of a fluent. EC uses circumscription as the non-monotonic formalism to solve the frame and elaboration tolerance problems. The other main approach to the frame problem (monotonic logic based) combines the two functions of effect completion and frame membership into a single axiom called the Successor State Axiom (see [50] for further details).

It may be noted that there has been a close link between the development of action theories and general non-monotonic reasoning, see (for example) [48].

8.2 Representing ramifications

In the 'Introduction', one property of a ramification solution was considered to be the ability to reason about the potentially large number of indirect effects of actions in a sparse way without the need for a full enumeration. It may be useful to consider that the process of causal stratification (c-stratification) involves searching (the space of explicit constraints) for an order in which the ramification constraints may be applied to direct actions. It does not involve searching the full space of effects, nor enumerating (or considering) all the indirect effects themselves of potential actions.

Another property advanced (by [57]) for the purpose of assessing the suitability of a representation of ramifications, is that of *representational parsimony*. This is to say that the representation should be easily understandable by humans, and efficient for automation. For EC-R, the humans would interact with *Whenever* relations, while the automated system would reason with its translation. The size of this translation is linear with the number of *Whenever* statements, and is acceptably compact for automation.

The causal approach to ramifications has been widely used in the literature, however it has been the subject of some criticism. In particular [53] advances an argument detailing an alleged limitation of causal approaches to ramifications. This argument derives from the causal approach's division of the effects of actions into two classes: direct and indirect (sometimes referred to as the main and consequential effects). Direct effects are those considered to be the main result of an action and specified explicitly in an action's definition, while indirect effects are assumed to lie downstream in a causal chain of effects beginning with a direct action. The difficulty is said to occur when domain change propagates in both directions between two or more fluents.

The argument assumes that since causality can only exist in one direction, then domains with bidirectional effects are incompatible with the method. However, the solution to the ramification problem employed by $\mathcal{E}$ and EC-R is able to deal with such domains by defining bidirectional causality, as demonstrated for instance in the Gearwheel example earlier. Such an example illustrates how the argument represents not a fundamental limitation of the causal approach, but rather an opportunity for suitable refinement of the causal approach.

8.3 EC axiom sets

The development of the EC-R ramification theory is modular with respect to the foundational Domain Independent EC Axiom set employed. Alternative axiom sets for various different classes of domains have been previously published in the literature. (EC1–8) were chosen for their application to deterministic domains so that a correspondence between EC-R and $\mathcal{E}$ could potentially be established (conditional upon matching ramification behaviour).

Although event (action) pre-conditions are not specifically expressible in $\mathcal{E}$, they are expressible in Event Calculus generally. It is therefore possible to use an EC axiom set designed with the capability to reason about preconditions as a substitute for the EC axiom set used in this article. Such a change would then allow EC-R to reason about pre-conditions.

The same is true for other extensions such as the representation of non-deterministic actions and concurrent actions. If a suitable (existing) axiom set designed for non-deterministic domains and/or concurrency is chosen (e.g. from [46]), then EC-R may represent domains containing non-determinism and concurrency. Several design choices in EC-R were taken specifically to allow for reasoning in concurrent-event domains.

8.4 Unstable domains

The ramification causal consistency of a domain description is defined to be the representation of those domains whose physical instantiation would never continually oscillate in state. Causal stratification (as defined earlier) serves both to create the foundational arrangement for causal reasoning, but in the process to also perform one part of a check for ramification causal consistency on the domain description. The domain description is ramification causally consistent if and only if it can be stratified and it is found to be statically consistent.

When axiomatizing (or updating) a domain theory, there is a possibility of formalizing a domain which is inherently unstable (continually oscillating with no steady state). It would appear to be advantageous to have a way to detect the presence of such domains.

This problem has been addressed in [66] where the concept of a ‘safe’ domain is specified. When determining safety for ramification domains, there are two approaches. The first, (adopted by [66]) is to exclude conditions which may potentially lead to instability. An example of such a condition, and the choice made by the authors, is to exclude any set of ramification constraints causing a recursive propagation of effects among the fluents. Eliminating such sets will guarantee the absence of instability, but at the cost of excluding many casually meaningful domains (such as those with positive recursion). The second approach is to provide for a more discriminating test, which we have referred to as a causal ramification consistency check. Such a test is sensitive to the causal pathways of influence between fluents established by ramification constraints, and determines if there exists an unstable configuration.

The causal stratification process (c-stratification, defined earlier) in combination with state constraint consistency is one way of realizing such a causal ramification consistency check.

8.5 Knowledge updates

Using EC-R, an agent can perform updates on a knowledge base by adding or deleting the causation predicates *Initiates* and *Terminates* or the formulae corresponding to the translated ramification constraints *Whenever*. Since *Whenever* is merely translated into formulae in a way similar to a macro expansion, each expansion instance can be simply added or removed in just the same way as a *Whenever* statement itself. It may be effective to maintain a reference link between *Whenever* statements and their corresponding translated formulae.

When performing a domain description update, there are two aspects of causal consistency that are of interest. The first is ensuring that within a sequence of actions, there are no actions that interfere with each other in a conflicting way (such as two concurrent actions achieving opposite results on a fluent, or a single action conflicting with a ramification constraint). This problem is addressed by, for example [11–13]. The second problem is identifying any mutually unstable set of ramification constraints.

The first problem is addressed in EC-R by the use of a non-deterministic EC axiom set. The second problem is addressed in EC-R by using the process of c-stratification (defined earlier) and state constraint consistency checking to achieve together a causal ramification consistency check. Extracting from the c-stratification search the ramification constraints that were involved in instability would appear to be relatively straightforward.

The final aspect of knowledge update we should consider is elaboration tolerance. This is the means by which a domain description supports the property that the degree of difficulty in making a knowledge update is only proportional to the changes involved, and not on the total size of the domain. EC-R supports this capability through the use of causal completion that allows changes in the domain description to be made incrementally. This applies both to event-triggered causation and fluent-triggered causation. A new stratification must be found after changes are made to the domain description, however this may be regarded as part of a 'safety check' in the knowledge update process.

9 Conclusion

In comparison with prior literature discussed in section 7, we have shown that a well-known ramification formalism for the Event Calculus has a technical flaw. We have also discussed another classical logic ramification formalism based on the Situation Calculus which displays many desirable qualities as a flexible ramification theory, however, it is prevented from reasoning about positive cycling domains, or domains with concurrent actions.

This article has presented a new version of classical logic Event Calculus EC-R, with the capability to reason about a wide range of ramification domains. In order to obtain some purchase on the new representational power, a correspondence proof has been undertaken with language $\mathcal{E}$'s semantics. This shows that the classes of domains for which the new EC-R theory applies is at least as large as those for which language $\mathcal{E}$ applies.

The relevance and value of this particular formalization is enhanced by its ability to (optionally) handle ramification domains in a way corresponding with language $\mathcal{E}$. The soundness and completeness result, with respect to $\mathcal{E}$'s semantics, provides both a solid justification for the design choices made in EC-R, and also an inheritance pathway between EC-R and the provably correct automated proof procedures [29] developed for $\mathcal{E}$.

A classical logic description is often considered a useful foundation for extensions and incorporation into larger integrated theories. It also serves as a point of comparative reference between alternative formalisms because it avoids the possibility of artefacts associated with specific custom formalisms clouding the picture. Importantly, a classical logic description of ramifications also allows immediate integration with existing EC theory and sub-theory extensions in a modular way.

All solutions to the ramifications problem inherit the characteristics of their underlying action formalisms. Both $\mathcal{E}$ and EC-R share a narrative-based semantics where reasoning occurs with respect to a time line. This has particular advantages when representing multiple concurrent domain events (or actions) with duration. In further work, it would be valuable to examine how EC-R can be extended with compound actions and actions with duration as found in [58] in a way independent of $\mathcal{E}$.

The demonstration of correspondence between the two formalisms is also useful in that the resulting specification of a ramification domain in classical logic may be used to integrate and develop a far wider scope of applications than those expressible solely under $\mathcal{E}$. Owing to the modular way in which the extension has been devised, a full integration of ramifications with many other capabilities engineered into Event Calculus (see [46] for a guide to extensions) can be achieved in a structured way. $\mathcal{E}$ may serve as a high level abstract semantics for the Event Calculus theory presented here, thus integrating and consolidating theoretical development.

Acknowledgements

We would like to thank Murray Shanahan for many useful discussions, Ian Hodkinson for help with the appendix, and also thank the anonymous reviewers for thoughtful and thought-provoking feedback on an earlier draft of the paper.

References

[1] S. Abiteboul, R. Hull and V. Vianu. *Foundations of Databases*. Addison-Wesley, Reading, US, 1995.

[2] P. Aczel. An introduction to inductive definitions. In *Handbook of Mathematical Logic*, Vol. 90 of *Studies in Logic and the Foundations of Mathematics*, J. Barwise, ed., chapter C.7, pp. 739–782. North-Holland Pub. Co., Amsterdam, Netherlands, 1977.

[3] A. B. Baker. A simple solution to the yale shooting problem. In *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning*, H. J. Levesque, R. J. Brachman and R. Reiter, eds, pp. 11–20. Morgan Kaufmann, Toronto, Canada, 1989.

[4] K. R. Apt and M. Bezem. Acyclic programs. In *Proceedings of the Seventh International Conference on Logic Programming*, D. H. D. Warren and P. Szeredi, eds, pp. 617–633. The MIT Press, Jerusalem, 1990.

[5] C. Baral and M. Gelfond. Representing concurrent actions in extended logic programming. In *Proceedings of the Thirteenth International Joint Conference on*

Artificial Intelligence, R. Bajcsy ed., pp. 866–871. Morgan Kaufmann, San Mateo, California, 1993.

[6] C. Baral, M. Gelfond and A. Provetti. Representing actions: Laws, observations and hypotheses. *Journal of Logic Programming*, **31**, 201–243, 1997.

[7] M. Denecker. The well-founded semantics is the principle of inductive definition. In *European Workshop on Logics in Artificial Intelligence (JELIA-98)*, Vol. 1489 of *LNAI*, J. Dix, L. Fariñas del Cerro and U. Furbach, eds, pp. 1–16. Springer, Berlin, 1998.

[8] M. Denecker, D. Theseider and D. Van Belleghem. An inductive definition approach to ramifications. *Electronic Transactions on Artificial Intelligence*, **2**, 25–67. 1998.

[9] K. Doets. *Basic Model Theory*. CSLI Publications, Stanford, California, 1996.

[10] H. Ebbinghaus and J. Flum. *Finite Model Thoery*. Perspectives in Mathematical Logic, Omega Series. Springer, Heidelberg, 1995.

[11] T. Eiter, E. Erdem, M. Fink and J. Senko. Updating action domain descriptions. In *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK*, L. P. Kaelbling and A. Saffiotti, eds, pp. 418–423. Professional Book Center, Edinburgh, Scotland, UK, 2005.

[12] T. Eiter, E. Erdem, M. Fink, and J. Senko. Comparing action descriptions based on semantic preferences. In *JELIA*, Vol. 4160 of *Lecture Notes in Computer Science*, M. Fisher, W. van der Hoek, B. Konev and A. Lisitsa, eds, pp. 124–137. Springer, Liverpool, UK, 2006.

[13] T. Eiter, E. Erdem, M. Fink and J. Senko. Resolving conflicts in action descriptions. In *ECAI 2006, 17th European Conference on Artificial Intelligence, August 29 – September 1, 2006, Riva del Garda, Italy, Including Prestigious Applications of Intelligent Systems (PAIS 2006), Proceedings*, G. Brewka, S. Coradeschi, A. Perini and P. Traverso, eds, pp. 367–371. IOS Press, River del Garden, Italy, 2006.

[14] C. Elkan. Reasoning about action in first-order logic. In *Proceedings of the Conference of the Canadian Society for Computational Studies of Intelligence*, pp. 221–227, British Columbia, Vancouver, 1992.

[15] K. Eshghi and R. A. Kowalski. Abduction compared with negation by failure. In *Logic Programming: Proceedings of the Sixth International Conference*, pp. 234–254. The MIT Press, Cambridge, Massachusetts, 1989.

[16] J. J. Finger. Exploiting constraints in design synthesis. *Technical Report STAN-CS-88-1204* (Thesis), Stanford University, Stanford, California, 1987.

[17] J. Forth and M. Shanahan. Indirect and conditional sensing in the event calculus. In *Proceedings of the 16th European Conference on Artificial Intelligence*, R. Lopez de Mantaras and L. Saitta, eds, pp. 900–904. IOS Press, Valencia, Spain, 2004.

[18] M. Gelfond. Autoepistemic logic and formalization of commonsense reasoning: a preliminary report. In *Second International Workshop on Non-Monotonic Reasoning*, M. Reinfrank, J. de Kleer and E. Sandewall, eds, pp. 177–186. Springer, Berlin, 1987.

[19] M. Gelfond and V. Lifschitz. Representing action and change by logic programs. *Journal of Logic Programming*, **17**, 301–321, 1993.

[20] M. Gelfond and V. Lifschitz. Action languages. *Electronic Transactions on Artificial Intelligence*, **2**, 193–210, 1998.

[21] M. L. Ginsberg and D. E. Smith. Reasoning about action I: a possible worlds approach. In *Readings in Nonmonotonic Reasoning*, M. L. Ginsberg ed., pp. 433–463. Morgan Kaufmann, Los Altos, California, 1987.

[22] E. Giunchiglia, G. N. Kartha and V. Lifschitz. Actions with indirect effects (extended abstract). In *Extending Theories of Action: Formal Theory and Practical Applications: Papers from the 1995 AAAI Spring Symposium*, pp. 80–85. AAAI Press, Menlo Park, California, 1995.

[23] E. Giunchiglia and V. Lifschitz. Dependent fluents. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, C. Mellish ed., pp. 1964–1969. Morgan Kaufmann, San Francisco, 1995.

[24] E. Giunchiglia and V. Lifschitz. An action language based on causal explanation. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and the Ninth Innovative Applications of Artificial Intelligence Conference*, T. Senator and B. Buchanan, eds, pp. 623–628. American Association for Artificial Intelligence, AAAI Press, Menlo Park, California, 1998.

[25] S. Hanks and D. McDermott. Default reasoning, nonmonotonic logics, and the frame problem. *In Proceedings of the Fifth National Conference on Artificial Intelligence*, pp. 328–333. AAAI, Philadelphia, US, 1986.

[26] B. Haugh. Simple causal minimization for temporal persistence and projection. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, K. Forbus and H. Shrobe, eds, pp. 218–223. American Association for Artificial Intelligence, AAAI Press, Menlo Park, California, 1987.

[27] Y. Iwasaki and H. A. Simon. Causality in device behavior. *Artificial Intelligence*, **29**, 3–32, 1986.

[28] A. Kakas, L. Michael and R. Miller. Modular-E: an elaboration tolerant approach to the ramification and qualification problems. In *Logic Programming and Nonmonotonic Reasoning: 8th International Conference*, Vol. 3662 of *LNCS*, C. Baral, G. Greco, N. Leone and G. Terracina, eds, pp. 211–226. Springer, 2005.

[29] A. C. Kakas and R. Miller. A simple declarative language for describing narratives with actions. *Journal of Logic Programming*, **31**, 157–200, 1997.

[30] A. C. Kakas and R. Miller. Reasoning about actions, narratives and ramifications. *Journal of Electronic Transactions on Artificial Intelligence*, **1**, 1998.

[31] A. Kakas, R. Miller and F. Toni. An argumentation framework for reasoning about actions and changes. In *Proceedings of the 5th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-99)*, Vol. 1730 of *LNAI*, M. Gelfond, N. Leone and G. Pfeifer, eds, pp. 78–91. Springer, Berlin, 1999.

[32] A. C. Kakas and F. Sadri. *Computational logic: Logic Programming and Beyond: Essays in Honor of Robert A. Kowalski*, Vol. 2407 of *Lecture Notes in Computer Science and Lecture Notes in Artificial Intelligence*. Springer, New York, NY, USA, 2002.

[33] G. N. Kartha and V. Lifschitz. Actions with indirect effects (preliminary report). In *KR'94: Principles of Knowledge Representation and Reasoning*, J. Doyle, E. Sandewall and P. Torasso, eds, pp. 341–350. Morgan Kaufmann, San Francisco, California, 1994.

[34] R. A. Kowalski and M. J. Sergot. A logic-based calculus of events. *New Generation Computing*, **4**, 67–95, 1986.

[35] J. Lang, F. Lin and P. Marquis. Causal theories of action: a computational core. In *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico*, G. Gottlob and T. Walsh, eds, pp. 1073–1078. Morgan Kaufmann, Acapulco, Mexico, 2003.

[36] V. Lifschitz. Formal theories of action (preliminary report). In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, J. McDermott, ed., pp. 966–972. Morgan Kaufmann, Milan, Italy, 1987.

[37] V. Lifschitz. Circumscription. In *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Nonmonotonic Reasoning and Uncertain Reasoning*, D. Gabbay, C. J. Hogger and J. A. Robinson, eds, pp. 298–352. Oxford University Press, Oxford, UK, 1994.

[38] V. Lifschitz. Two components of an action language. *Annals of Mathematics and Artificial Intelligence*, **21**, 305–320, 1997.

[39] V. Lifschitz and H. Turner. Splitting a logic program. In *Proceedings of the 12th International Conference on Logic Programming, Kanagawa 1995*, MIT Press Series Logic Program, pp. 581–595. MIT Press, Cambridge, Massachusetts, 1995.

[40] F. Lin. Embracing causality in specifying the indirect effects of actions. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, C. Mellish, ed., pp. 1985–1991, Morgan Kaufmann, San Francisco, 1995.

[41] F. Lin and R. Reiter. State constraints revisited. *Journal of Logic and Computation*, **4**, 655–678, 1994.

[42] F. Lin and Y. Shoham. Provably correct theories of action (preliminary report). In *Proceedings of the Ninth National Conference on Artificial Intelligence*, T. Dean and K. McKeown, eds, pp. 349–354. American Association for Artificial Intelligence, The MIT Press, Cambridge, Massachusetts, 1991.

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

[44] N. McCain and H. Turner. A causal theory of ramifications and qualifications. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, C. S. Mellish, ed., pp. 1978–1984. Morgan Kaufmann, San Mateo, 1995.

[45] S. McIlraith. Integrating actions and state constraints: a closed-form solution to the ramification problem. *Artificial Intelligence*, **116**, 87–121, 2000.

[46] R. Miller and M. Shanahan. Some alternative formulations of the event calculus. In *Computations Logic: Logic Programming and Beyond - Essays in Honour of Robert Kowalski, Vol. 2408 of Lecture Notes in Artificial Intelligence*, A. Kakas and F. Sadri, eds, pp. 452–490. Springer Verlag, Berlin, Heidelberg, Germany, 2002.

[47] P. H. Morris. The anomalous extension problem in default reasoning. *Artificial Intelligence*, **35**, 383–399, 1988.

[48] J. Pearl. Embracing causality in default reasoning. *Artificial Intelligence*, **35**, 259–271, 1988.

[49] J. Pinto and R. Reiter. Temporal reasoning in logic programming: a case for the situation calculus. In *Proceedings of the Tenth International Conference on Logic Programming*, D. S. Warren, ed., pp. 203–221. The MIT Press, Budapest, Hungary, 1993.

[50] R. Reiter. The frame problem in the situation calculus: a simple solution (sometimes) and a completeness result for goal regression. In *Artificial Intelligence and Mathematical Theory of Computation*, V. Lifschitz, ed., pp. 359–380. Academic Press, Cambridge, Massachusetts, 1991.

[51] R. Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press, Cambridge, Massachusetts, 2001.

[52] E. Sandewall. *Features and Fluents: A Systematic Approach to the Representation of Knowledge about Dynamical Systems*. Oxford University Press, Oxford, 1994.

[53] E. Sandewall. Comparative assessments of ramification methods that use static domain constraints. In *KR'96: Principles of Knowledge Representation and Reasoning*,

L. C. Aiello, J. Doyle and S. Shapiro, eds, pp. 99–110. Morgan Kaufmann, San Francisco, California, 1996.

[54] R. B. Scherl and H. J. Levesque. The frame problem and knowledge-producing actions. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, R. Fikes and W. Lehnert, eds, pp. 698–695. American Association for Artificial Intelligence, AAAI Press, Menlo Park, California, 1993.

[55] L. Schubert. Monotonic solution of the frame problem in the situation calculus: an efficient method for worlds with fully specified actions. In *Knowledge Representation and Defeasible Reasoning*, Vol. 5 of *Studies in Cognitive Systems*, H. E. Kyburg, R. P. Loui and G. N. Carlson, eds, pp. 23–67. Kluwer Academic Publishers, Dordrecht, Boston, London, 1990.

[56] M. Shanahan. A circumscriptive calculus of events. *Artificial Intelligence*, **77**, 251–284, 1995.

[57] M. Shanahan. *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia*. MIT Press, Cambridge, Massachusetts, USA, 1997.

[58] M. Shanahan. The event calculus explained. *Lecture Notes in Computer Science*, **1600**, 409–434, 1999.

[59] M. Shanahan. The ramification problem in the event calculus. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99-Vol1)*, D. Thomas, ed., pp. 140–146. Morgan Kaufmann Publishers, San Francisco, California, 1999.

[60] Y. Shoham. Nonmonotonic reasoning and causation. *Cognitive Science*, **14**, 213–252, 1990.

[61] E. Ternovskaia. Inductive definability and the situation calculus. *Lecture Notes in Computer Science*, **1472**, 227–248, 1998.

[62] M. Thielscher. Computing ramifications by postprocessing. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, C. S. Mellish, ed., pp. 1994–2000. Morgan Kaufmann, San Mateo, 1995.

[63] M. Thielscher. The logic of dynamic systems. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, C. S. Mellish, ed., pp. 1956–1963. Morgan Kaufmann, San Mateo, 1995.

[64] M. Thielscher. Ramification and causality. *Artificial Intelligence*, **89**, 317–364, 1997.

[65] H. Turner. Representing actions in logic programs and default theories: a situation calculus approach. *Journal of Logic Programming*, **31**, 245–298, 1997.

[66] D. Zhang, S. Chopra and N. Y. Foo. Consistency of action descriptions. In *PRICAI 2002: Trends in Artificial Intelligence, 7th Pacific Rim International Conference on Artificial Intelligence, Tokyo, Japan, Proceedings*, Vol. 2417 of *Lecture Notes in Computer Science*, M. Ishizuka and A. Sattar, eds, pp. 70–79. Springer, Tokyo, Japan, 2002.

Appendix

We introduce Least Fixed Point Logic (LFP) as a means of stating the *specification* of language $\mathcal{E}$ in a formal language. This is useful for gaining further insight and understanding, and for proving properties of the $\mathcal{E}$ formalism. A mapping may be constructed between the resulting LFP logic axiomatization and EC-R using the causation operator induced by LFP logic. A consequence of axiomatizing $\mathcal{E}$'s semantics in such a way is a demonstration that LFP logic has the expressive power to represent ramification domains for the full classes of domains acceptable to $\mathcal{E}$.

A.1 Least Fixed Point Logic

Fixed point logics are a member of the class of definition logics within constructive mathematics, finding applications in database theory among others e.g. [1]. LFP Logic can be used to express properties that are not first-order expressible. A prototypical example is representing the transitive closure of a relation. Refer to [9] and [10] for an in-depth discussion of LFP logic. LFP logic is useful because it is guaranteed to be expressive enough to directly capture the conditions forming $\mathcal{E}$'s semantics.

Language $\mathcal{E}$ causal points are specified inductively through the induced construction of an operator. We will make use of LFP Logic to provide a formal account of the construction of this operator. We begin the formal mapping of language $\mathcal{E}$ semantics into classical logic EC with a direct statement of $\mathcal{E}$'s specification of causation points in LFP Logic.

A.2 Mapping causation points into LFP logic

In this section, we map Definition 10 into a LFP meta-formula. As seen previously, language $\mathcal{E}$'s semantics are based on an inductive definition that leads to the construction of an operator. When iterated sufficiently to form a least fixed point, a complete set of domain causation points are constructed. To facilitate the correspondence proof in later sections, it is useful to use the framework of LFP Logic to provide a formal definition of the operator resulting from the inductive definition specified in $\mathcal{E}$'s semantics. The LFP logic framework also provides insight to conditions under which the least fixed point can be constructed. This is important because it affects the classes of domains over which the entire ramification theory of $\mathcal{E}$ and EC-R can be expected to hold.

Definition 10 (Initiation/Termination point) contained the definition of causation points for direct actions (part1) and indirect actions (part2) with respect to interpretation H. We will rewrite these directly as subformulae of LFP logic. Using uppercase symbols to represent meta-variables, implicitly universally quantified, part 1a and 1b of Definition 10 are FP1 and FP2, respectively.

$$\begin{aligned}
\phi_1(F,T,init) &\overset{\text{def}}{\equiv} (\text{`}A\ \textbf{initiates}\ F\ \textbf{when}\ C\text{'} \in \gamma) \leftarrow \\
&\wedge \Big(\bigwedge_{F' \in C} H(F',T) = true \Big) \wedge \Big(\cancel{\bigwedge}_{\neg F' \in C} \Big(H(F',T) = false \Big) \leftarrow \\
&\wedge (\text{`}A\ \textbf{happens-at}\ T\text{'} \in \eta) \leftarrow
\end{aligned} \tag{FP1}$$

$$\begin{aligned}
\phi_1(F,T,term) &\overset{\text{def}}{\equiv} (\text{`}A\ \textbf{terminates}\ F\ \textbf{when}\ C\text{'} \in \gamma) \leftarrow \\
&\wedge \Big(\bigwedge_{F' \in C} H(F',T) = true \Big) \wedge \Big(\cancel{\bigwedge}_{\neg F' \in C} \Big(H(F',T) = false \Big) \leftarrow \\
&\wedge (\text{`}A\ \textbf{happens-at}\ T\text{'} \in \eta) \leftarrow
\end{aligned} \tag{FP2}$$

The second part of Definition 10 defines causation points generated through the **whenever** statements by indirect actions. Intermediate variable definitions will assist a simple written structure. For formulae FP3 and FP4, and a set C of conditional fluent literals such that $C \neq \emptyset$, let the following subset of a powerset partition be defined between the locally scoped variables.

$$P'(C) = \{\, (X, Y) : X \cup Y = C, X \cap Y = \emptyset, X \neq \emptyset \,\} \leftarrow$$

For a relation *InTe* defining initiates or terminates causation points by the introduction of constants *init* and *term*, parts 2a and 2b of Definition 10, the corresponding LFP logic subformulae are FP3 and FP4, respectively:

$$\phi_2(F, T, init, InTe) \overset{\text{def}}{\equiv} \leftarrow \bigvee_{(C_1, C_2) \in \mathcal{P}'(C)} \Big((\text{`}F \textbf{ whenever } C\text{'} \in \rho \psi$$
$$\wedge \leftarrow \bigwedge_{F'_1 \in C_1} \Big(InTe(F'_1, T, init) \wedge \leftarrow \bigwedge_{\neg F'_1 \in C_1} \Big(InTe(F'_1, T, term)$$
$$\wedge\ \exists T_2 (T \prec T_2\ \wedge\ (\forall T_1 (T \prec T_1 \prec T_2)\ \rightarrow$$
$$\bigwedge_{F'_2 \in C_2} H(F'_2, T_1) = true\ \wedge \bigwedge_{\neg F'_2 \in C_2} \Big(H(F'_2, T_1) = false\ ))) \leftarrow \tag{FP3}$$

$$\phi_2(F, T, term, InTe) \overset{\text{def}}{\equiv} \leftarrow \bigvee_{(C_1, C_2) \in \mathcal{P}'(C)} \Big((\text{`}\neg F \textbf{ whenever } C\text{'} \in \rho \psi$$
$$\wedge \leftarrow \bigwedge_{F'_1 \in C_1} InTe(F'_1, T, init) \wedge \leftarrow \bigwedge_{\neg F'_1 \in C_1} \Big(InTe(F'_1, T, term)$$
$$\wedge\ \exists T_2 (T \prec T_2\ \wedge\ (\forall T_1 (T \prec T_1 \prec T_2)\ \rightarrow\ \leftarrow$$
$$\bigwedge_{F'_2 \in C_2} H(F'_2, T_1) = true\ \wedge \leftarrow \bigwedge_{\neg F'_2 \in C_2} \Big(H(F'_2, T_1) = false\ ))) \tag{FP4}$$

Combining FP1–4 into a single definition formula corresponding to Definition 10 in full, a new variable *Caus* is introduced to represent the type of causation point.

$$(F, T, Caus, InTe) \overset{\text{def}}{\equiv}\ \phi_1(F, T, Caus)\ \vee\ \phi_2(F, T, Caus, InTe) \leftarrow \tag{FP5}$$

Causation is then represented in LFP logic by FP6 where $\mu\psi$ is the fixed point formula designator defining a new predicate $\mu[\ (F, T, Caus, InTe)]$.

$$\mu[\ (F, T, Caus, InTe)](F, T, Caus) \leftarrow \tag{FP6}$$

Definition A1 (Causation-operator)

For a domain description $D = (\gamma, \eta, \tau, \rho)$ and an interpretation $H : \Phi \times \Pi \mapsto \{true, false\}$, the set of constants $Ct = \{init, term\}$ where $init \neq term$, the set $Cp = (\Phi \times \Pi \times Ct)$, the set $W = \mathcal{P}(Cp)$, the LFP logic model $\mathcal{A}$, a recursive variable defining sets of causation points $InTe \subseteq Cp$, a member of fluent literals F, timepoint Φ, T' in Π, then the monotonic operator $\mathcal{F} \leftarrow = \mathcal{F} \leftarrow : W \mapsto W$ is induced according to LFP logic definitions as follows:

$$\mathcal{F}\ (InTe) =\ \{(F, T, Caus) \in Cp \mid (\mathcal{A}, InTe) \Vdash\ (F, T, Caus, InTe)\} \leftarrow$$

The formula $(F, T, Caus, InTe)$ meets the syntactical requirement for monotonicity as required by LFP logic. Therefore, operator $\mathcal{F} \leftarrow$ is monotone and it follows directly that $W_1 \subset W_2 \subset Cp$ implies $\mathcal{F}\ (W_1) \subset \mathcal{F}\ (W_2)$ and that operator $\mathcal{F} \leftarrow$ as defined above gives rise to a sequence of sets $\emptyset, \mathcal{F}\ (\emptyset), \mathcal{F}\ (\mathcal{F}\ (\emptyset)), \ldots$ denoted by $\mathcal{F}_0 = \emptyset$, $\mathcal{F}_1 = \mathcal{F}\ (\emptyset)$ which contains the effects of direct actions, and $\mathcal{F}_{n+1} = \mathcal{F}\ (\mathcal{F}_n)$ which contains the effects of subsequent successive stages of indirect actions. $\mathcal{F}_\infty = InTe^f \in W$ represents the least fixed point of operator $\mathcal{F} \leftarrow$ and is defined as $\mathcal{F}_\infty = \leftarrow \bigcup_{n \geq 0} \mathcal{F}_n$. Through the semantics of LFP logic the model $\mathcal{A}$ is then defined by FP7.

$$\mathcal{A} \Vdash \mu[\ (F, T, Caus, InTe)](F, T, Caus) \leftarrow \text{iff}\quad (F, T, Caus) \in \mathcal{F}_{\infty \leftarrow} \tag{FP7$\leftarrow$}$$

$(F, T, Caus)$ is therefore defined to be a causation point iff $(F, T, Caus) \in \mathcal{F}_\infty$ or equivalently, iff FP6 holds true.

We saw previously that by Definition A1 for the causation operator and FP1–6, operator $\mathcal{F}\leftarrow$is induced by the sentences of the LFP logic. $(F, T, Caus, InTe)$ (FP5) meets LFP logic's requirement for monotonicity, confirming that a LFP exists. A model of a LFP formula is given by the least fixed point $\mathcal{F}_\infty$ of this induced operator.

This operator may form the basis of an inductive model-theoretic soundness and completeness proof of correspondence with the circumscribed classical-logic Event Calculus theory EC-R in exactly the same way as that described in the second half of this article.

Received 23 February 2006