To appear: *Artificial Intelligence*, vol. 89, 1997.

Ramification and Causality

MICHAEL THIELSCHER[1]
International Computer Science Institute, 1947 Center Street, Berkeley, CA 94704-1198
e-mail: michaelt@icsi.berkeley.edu

Abstract. In formal systems for reasoning about actions, the ramification problem denotes the problem of handling indirect effects. These effects are not explicitly represented in action specifications but follow from general laws describing dependencies among components of the world state. An adequate treatment of indirect effects requires a suitably weakened version of the general law of persistence. It also requires a method to avoid unintuitive changes suggested by the aforementioned dependency laws. We propose a solution to the ramification problem that uses directed relations between two single effects, stating the circumstances under which the occurrence of the first *causes* the second. We argue for the necessity of an approach based on causality by elaborating the limitations of common paradigms employed to handle ramifications—the principle of categorization and the policy of minimal change. Our abstract solution is realized on the basis of a particular action calculus, namely, the fluent calculus.

Keywords. Temporal Reasoning, Ramification Problem, Causality.

1 Introduction

The ability to reason about changing environments, which involves predicting the effects of one's own actions and explaining observed phenomena, serves humans as a basis for understanding the world to an extent sufficient to survive and to act intelligently in their habitat. Formal approaches to model this ability have a long tradition in AI. This research area was initiated by McCarthy [30], who claimed that reasoning about actions plays a fundamental role in common sense.

Drawing conclusions about dynamic environments is grounded on formal specifications of what effects are caused by the execution of a particular action. Since it is obviously infeasible to provide an exhaustive description defining the result of executing an action in each possible state of the world, action specifications should be restricted to the part of the world that they affect—while the rest of the world is subject to the *law of persistence*, i.e., is assumed to remain stable. Yet even this approach becomes unmanageable in complex domains if one tries to put all effects into a single, complete specification. Although an action may cause only a small number of *direct* changes, they in turn may initiate a long chain of *indirect* effects that can be hard to foresee. For instance, consider the action of toggling a switch, which in the first place causes nothing but a change of the switch's position. However, the switch may be part of an electric circuit so that, say, some light bulb is turned off as a side effect, which in turn may cause someone to hurt himself in a suddenly darkened room by running against a chair that, as a consequence, falls into a television set whose implosion activates the fire alarm and so on and

[1] On leave from FG Intellektik, TH Darmstadt

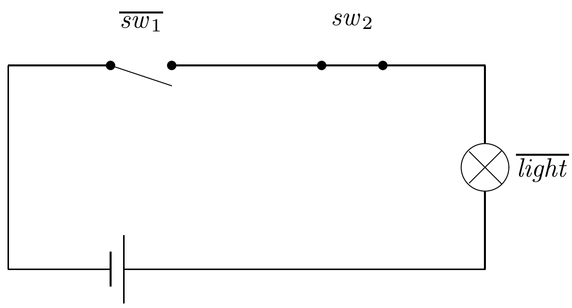

Figure 1: An electric circuit consisting of a battery, two switches, and a light bulb, which is on if and only if both switches are closed. The respective state of each of the three dynamic components involved is described by a unique propositional constant, where negation is denoted by a bar on top of the respective symbol.

so forth.[2]

The task, then, is to design a framework to formalize action scenarios where action specifications are not assumed to completely describe all possible effects. This is called the *ramification problem*.[3] A satisfactory solution to this problem requires a successful treatment of the following two major issues.

First, we need an appropriately weakened version of the aforementioned law of persistence that applies only to those parts of the world description that are unaffected by the action's direct *and* indirect effects. As a solution to this problem, we suggest keeping the law of persistence as it stands while considering the world description obtained through its application merely as an intermediate result. Indirect effects are then accommodated by further reasoning until an overall satisfactory successor state obtains. This method accounts perfectly both for rigorous persistence of unaffected parts and for arbitrarily complex chains of indirect effects.

Second, indirect effects typically are consequences of additional, general knowledge of domain-specific dependencies between world description components (usually called *fluents*)—but not all effects suggested from this perspective are desirable from the standpoint of causality. As an example, consider the simple electric circuit depicted in Figure 1, which consists of a battery, two switches and a light bulb. The obvious connection between these components may formally be described by the logic expression $sw_1 \wedge sw_2 \equiv \mathit{light}$, i.e., the light is on if and only if both switches are on. Now suppose we toggle the first switch in the particular state displayed, where both sw_1 and light are false while sw_2 is true. Then, besides the direct effect of sw_1 becoming true, we also expect that the light bulb turns on. This indirect effect is inspired by the formula

[2] A crucial question in this context concerns the distinction between indirect effects occurring during a single world's state transition step and those which deserve separate state transitions (also called *delayed* effects). E.g., the above may not only be described as "the fire alarm is activated in the successor state after having toggled the switch," but also as, say, "the chair is falling in the successor state (and presumably hits the television set during the next state transition)." As a reasonable, albeit informal, guidance we suggest a single state transition should summarize what happens until someone, e.g. the reasoning agent himself, has the possibility to intervene by acting again (stopping the chair from falling, for instance).

[3] The naming was suggested in [15], inspired by [10]. The latter, however, was not devoted to the ramification problem in exactly the above sense, contrary to what is often claimed; rather, this thesis describes how to exploit logical consequences (called ramifications) of goal specifications in planning problems, with the aim of restricting search space.

just mentioned, which includes the implication $sw_1 \wedge sw_2 \supset \mathit{light}$. However, despite this being the intuitively expected result, the mere knowledge of the connection between the switches and the bulb is not sufficient: Notice that the above formula, $sw_1 \wedge sw_2 \equiv \mathit{light}$, also entails the implication $sw_1 \wedge \overline{\mathit{light}} \supset \overline{sw_2}$, which suggests that instead of the light being turned on, the indirect effect of toggling the first switch is that the second one jumps its position—a result which contradicts our intuition.

The reason for the inadequacy of merely taking into account formalizations of dependencies as pure logical formulas—usually called *domain constraints*—is that these formulas do not include causal information. More precisely, the implication $sw_1 \wedge \overline{\mathit{light}} \supset \overline{sw_2}$ is clearly true in any state and, therefore, contains *evidential* information, that is, if one observes sw_1 to be true and light to be false then it is safe to conclude that sw_2 is false. However, exploiting this implication for reasoning about *causality* is misleading.[4]

As a solution to the second problem, we propose to incorporate causality in the form of so-called *causal relationships*, which formalize statements like

> A change of $\overline{sw_1}$ to sw_1 causes a change of $\overline{\mathit{light}}$ to light, provided sw_2 is true.

After computing the direct effects of executing an action in a particular state of the world, we will apply suitable causal relationships, one by one, to accommodate indirect effects until a satisfactory result obtains. Employing a collection of single causal relationships, each of which only relates two particular effects, accounts for several indirect effects caused by a direct one and also for indirect effects in turn causing further indirect effects. To illustrate the latter, consider the relationship

> A change of $\overline{\mathit{light}}$ to light causes a change of $\overline{\mathit{light\text{-}detector}}$ to $\mathit{light\text{-}detector}$, provided $\mathit{detector\text{-}activated}$ is true.

in addition to the one above. Since we do not expect the designer of a formal domain specification to create a complete set of suitable causal relationships, we will also present an automated procedure to extract them from given domain constraints plus some general information specifying which fluents may possibly influence other fluents. On the basis of a formal theory of actions (to be defined in Section 2), causal relationships and their automatic generation are formally introduced in Section 3.

Yet the purpose of this article is not only to suggest incorporating causal information by means of our causal relationships but also to supply evidence that something like this approach is inevitable when trying to cope with the ramification problem. To this end, in Section 5 we illustrate the limited expressiveness of existing paradigms aimed at handling ramifications, namely, the principle of *categorization* and the policy of *minimal change*. Roughly speaking, categorization-based approaches distinguish fluents that are more likely to change than other fluents (e.g., a change of light is considered more likely than a change of sw_2). However, we will show that some fluents resist any unique categorization (Section 5.1). Even more common is the assumption of minimal change, which amounts to rejecting a resulting state if it is obtained by changing the values of strictly more fluents than necessary. While we will offer a formal proof that our method captures all intuitively expected resulting states with minimal distance to the original state (Section 4), in Section 5.2 we will illustrate that we are also able to find non-minimal solutions, which are perfectly acceptable provided all changes are reasonable from the standpoint of causality. On the basis of these observations, a detailed comparison with related work can be found in the concluding discussion.

[4] See [33] for a general discussion on the different natures of causal and evidential implications.

In the second part of the paper, Section 6, we integrate the concept of causal relationships into a particular action calculus, namely, the *fluent calculus* [19, 20]. While for the sake of simplicity states are described via a set of propositional constants in the first part (see Section 2), our calculus itself employs a more complex notion of fluent, which involves fluent formulas that include quantification. The encoding will be proved correct wrt. the formal semantics induced by causal relationships and their application as a solution to the ramification problem.

2 A Basic Theory of Actions

In the first part of the paper, we employ a suitably simple theory of actions and change, which enables us to stress solely on the problem of domain constraints and ramifications. The basic entities in this theory are *states*. A state is a snapshot of the underlying dynamic system, i.e., the part of the world being modeled, at a particular instant of time. Formally, we describe a state by assigning truth-values to a fixed finite set of propositional constants.

Definition 1 Let $\mathcal{F}$ be a finite set of symbols called *fluent names*. A *fluent literal* is either a fluent name $f \in \mathcal{F}$ or its negation, denoted by $\overline{f}$. A set of fluent literals is *inconsistent* iff it contains some $f \in \mathcal{F}$ along with $\overline{f}$. A *state* is a maximal consistent set of fluent literals. ∎

Notice that formally any combination of truth-values denotes a state, which, however, might be considered impossible due to specific dependencies among some fluents (see below). Throughout the paper we assume the following notational conventions: If ℓ is a fluent literal, then by $|\ell|$ we denote its affirmative component, that is, $|f| = |\overline{f}| = f$ where $f \in \mathcal{F}$. This notation extends to sets of fluent literals S as follows: $|S| = \{|\ell| : \ell \in S\}$. E.g., for each state S we have $|S| = \mathcal{F}$. Furthermore, if ℓ is a negative fluent literal then $\overline{\ell}$ should be interpreted as $|\ell|$. In other words, $\overline{\overline{f}} = f$. Finally, if S is a set of fluent literals then by $\overline{S}$ we denote the set $\{\overline{\ell} : \ell \in S\}$. E.g., $\overline{\mathcal{F}}$ contains all negative fluent literals given a set $\mathcal{F}$ of fluent names.

Example 1 To model the electric circuit depicted in Figure 1, we use the three fluents $\mathcal{F} = \{sw_1, sw_2, light\}$ to denote, respectively, the states of the two switches and the light bulb. The current state displayed in Figure 1, for instance, is represented by $\{\overline{sw_1}, sw_2, \overline{light}\}$. ∎

The elements of an underlying set of fluent names can be considered atoms for constructing (propositional) formulas to allow for statements about states. Truth and falsity of such formulas wrt. a particular state S are based on defining a literal ℓ to be true if and only if $\ell \in S$.

Definition 2 Let $\mathcal{F}$ be a set of fluent names. The set of *fluent formulas* is inductively defined as follows: Each fluent literal in $\mathcal{F} \cup \overline{\mathcal{F}}$ and $\top$ (*tautology*) and $\bot$ (*contradiction*) are fluent formulas, and if F and G are fluent formulas then so are $F \wedge G$, $F \vee G$, $F \supset G$, and $F \equiv G$.[5]

Let S be a state and F a fluent formula, then the notion of F being *true* (resp. *false*) in S is inductively defined as follows:

1. $\top$ is true and $\bot$ is false in S;
2. a fluent literal ℓ is true in S iff $\ell \in S$;
3. $F \wedge G$ is true in S iff F and G are true in S;

[5] As negation can be expressed through negative literals, we omit the standard connective "$\neg$". This is just for the sake of readability as it avoids too many different forms of negation.

4. $F \vee G$ is true in S iff F or G is true in S (or both);

5. $F \supset G$ is true in S iff F is false in S or G is true in S (or both);

6. $F \equiv G$ is true in S iff F and G are true in S, or else F and G are false in S.

■

Fluent formulas provide means to distinguish states that cannot occur due to specific dependencies between particular fluents. Formulas which have to be satisfied in all states that are possible in a domain are also called *domain constraints*.

Example 1 (continued) We employ the fluent formula $sw_1 \wedge sw_2 \equiv \mathit{light}$ as domain constraint to model the intended relation between the two switches and the light bulb. This formula holds, for instance, in the state depicted in Figure 1, viz. $\{\overline{sw_1}, sw_2, \overline{\mathit{light}}\}$, but is false in, say, the state $\{sw_1, sw_2, \overline{\mathit{light}}\}$. ■

The second basic entity in frameworks to reason about dynamic environments are *actions*, whose execution causes state transitions. Again, since stress shall lie on the ramification problem rather than on sophisticated methods of specifying the *direct* effects of actions, we employ a suitably simple, STRIPS-style [9, 23] notion of action specification. Each *action law* consists of

- A *condition* C, which is a set of fluent literals all of which must be contained in the state at hand in order to apply the action law.
- A (direct) *effect* E, which is a set of fluent literals, too, all of which hold in the resulting state after having applied the action law.

For the sake of simplicity, we assume $|C| = |E|$ (that is, condition and effect refer to the very same set of fluent names). This enables us to obtain the state resulting from the direct effect by simply removing set C from the state at hand and adding set E to it. This assumption does not impose a restriction of expressiveness since we allow several laws for a single action.

Definition 3 Let $\mathcal{F}$ be a set of fluent names, and let $\mathcal{A}$ be a finite set of symbols, called *action names*, such that $\mathcal{F} \cap \mathcal{A} = \{\}$. An *action law* is a triple $\langle C, a, E \rangle$ where C, called *condition*, and E, called *effect*, are consistent sets of fluent literals such that $|C| = |E|$; and $a \in \mathcal{A}$.

If S is a state then an action law $\alpha = \langle C, a, E \rangle$ is *applicable* in S iff $C \subseteq S$. The *application* of α to S yields the state $(S \setminus C) \cup E$. ■

Notice that S being a state, C and E being consistent, and $|C| = |E|$ guarantee $(S \setminus C) \cup E$ to be a state again—not necessarily, however, one which satisfies the underlying domain constraints. Notice also that it is possible to construct a set of action laws which contains more than one applicable law for an action name and a state. This can be used to describe actions with non-deterministic effect.[6]

[6] Suppose, as an example, that a gun non-deterministically gets loaded or not when spinning its cylinder [38]. This may be formalized by the two action laws $\langle \{\}, \mathit{spin}, \{\} \rangle$ and $\langle \{\overline{\mathit{loaded}}\}, \mathit{spin}, \{\mathit{loaded}\} \rangle$. Both of them are applicable in the state $\{\overline{\mathit{loaded}}\}$, which suggests two possible outcomes, viz. $\{\overline{\mathit{loaded}}\}$ and $\{\mathit{loaded}\}$. See [45] for details on how to represent and reason about non-deterministic actions on this basis.

Example 1 (continued) Suppose our Electric Circuit domain allows for two actions, namely, toggling either switch. These actions are represented by the action names $toggle_1$ and $toggle_2$, respectively, along with the four action laws

$$\begin{array}{ll} \langle \{\overline{sw_1}\}, \, toggle_1, \, \{sw_1\} \rangle & \langle \{\overline{sw_2}\}, \, toggle_2, \, \{sw_2\} \rangle \\ \langle \{sw_1\}, \, toggle_1, \, \{\overline{sw_1}\} \rangle & \langle \{sw_2\}, \, toggle_2, \, \{\overline{sw_2}\} \rangle \end{array} \tag{1}$$

That is, the only direct effect of these actions is a change of the respective switch's position. When executing, say, $toggle_1$ in the state $S = \{\overline{sw_1}, sw_2, \overline{light}\}$, then the first of the above laws is applicable due to $\{\overline{sw_1}\} \subseteq S$. Its application yields

$$(S \setminus \{\overline{sw_1}\}) \cup \{sw_1\} \; = \; \{sw_1, sw_2, \overline{light}\}$$

Notice that our domain constraint $sw_1 \wedge sw_2 \equiv light$ is false in the resulting state. ■

Our example illustrates that a state obtained through the application of an action law may violate the underlying domain constraints since only direct effects have been specified. In the next section, we develop a method to further modify such a preliminarily resulting, impossible state in order to account for additional, indirect effects.

3 Causal Relationships

The ramification problem arises as soon as it does not suffice to compute the direct effects of actions. This is the case whenever the resulting collection of fluent literals violates underlying domain constraints, which in turn give rise to additional, indirect effects. In theory, we could of course compile all conceivable indirect effects into action laws by exploiting the fact that an arbitrary number of different laws for a single action can be formulated. For instance, in our running example we could replace $\langle\{\overline{sw_1}\}, toggle_1, \{sw_1\}\rangle$ by the two laws $\langle\{\overline{sw_1}, sw_2, \overline{light}\}, toggle_1, \{sw_1, sw_2, light\}\rangle$ and $\langle\{\overline{sw_1}, \overline{sw_2}, \overline{light}\}, toggle_1, \{sw_1, \overline{sw_2}, \overline{light}\}\rangle$. If now $toggle_1$ is applied to the state $\{\overline{sw_1}, sw_2, \overline{light}\}$, then we obtain the desired result, viz. $\{sw_1, sw_2, light\}$. However, this procedure bears two major problems which demonstrate its inadequacy. First, it may require an enormous number of action laws to account for every possible combination of indirect effects. Consider, for example, a model of an electric circuit where a distinguished switch is involved in n sub-circuits each of which additionally contains a switch-bulb pair in a similar fashion as in Example 1. Defining all effects of toggling the separate switch solely by means of action laws would then require 2^{n+1} different laws, one for each possible combination of truth-values assigned to the switch being operated and the other n switches. The second problem with exhaustive action laws is that the introduction of a new domain constraint may cause, in the worst case, a re-definition of the entire set of action laws used before.

3.1 Applying Causal Relationships

As a consequence of the above observations, we keep the given action laws but regard a state obtained after having computed the direct effect merely as intermediate. Any single indirect effect is then obtained according to a directed *causal* relation between specific fluents. For instance, having as direct effect a change of the first switch into its upper position in the state depicted in Figure 1, this is regarded as additionally *causing* the light bulb to change its state also, for the second switch is on. We will formalize causal relationships of this kind by expressions like

$$sw_1 \;\texttt{causes}\; \mathit{light} \;\texttt{if}\; sw_2 \tag{2}$$

Formally, such expressions operate on state-effect pairs (S, E) where S is the current collection of fluent literals and E contains all (direct or indirect) effects that have been considered so far. E.g., let $S = \{sw_1, sw_2, \overline{\mathit{light}}\}$ be the state obtained after having computed the direct effect of toggle_1, as described in the preceding section, then $E = \{sw_1\}$. The causal relationship formalized in (2) gives rise to the indirect effect light, which supersedes $\overline{\mathit{light}}$ in S. This new effect is added to E. Altogether, this results in the state-effect pair $(\{sw_1, sw_2, \mathit{light}\}, \{sw_1, \mathit{light}\})$.

The reason for maintaining the second component, E, is that identical intermediate states S can be reached by different sets of effects E, each of which may require a different, sometimes opposite treatment. Suppose, as an example, there are two switches which are mechanically connected (say, through a spring) such that they cannot be closed simultaneously. The corresponding domain constraint, $\overline{sw_1} \vee \overline{sw_2}$, gives rise to two causal relationships, viz.

$$sw_1 \;\texttt{causes}\; \overline{sw_2} \;\texttt{if}\; \top \tag{3}$$

$$sw_2 \;\texttt{causes}\; \overline{sw_1} \;\texttt{if}\; \top \tag{4}$$

Now, toggling the first switch in the state $\{\overline{sw_1}, sw_2\}$ yields the intermediate state $\{sw_1, sw_2\}$ if we apply the corresponding action law in (1). The very same intermediate state is obtained by toggling the second switch in the state $\{sw_1, \overline{sw_2}\}$. Yet the intended outcomes differ considerably: In the first case, the final result should be $\{sw_1, \overline{sw_2}\}$, while $\{\overline{sw_1}, sw_2\}$ is expected in the second case. This distinction can only be achieved by referring to the differing direct effects, $\{sw_1\}$ and $\{sw_2\}$. The former enables only the application of (3) to the intermediate result, $\{sw_1, sw_2\}$, the latter only the application of (4). In both cases, this leads to the desired unique successor state.[7]

The formal definition of causal relationships and their application is as follows:

Definition 4 Let $\mathcal{F}$ be a set of fluent names. A *causal relationship* is an expression of the form $\varepsilon \;\texttt{causes}\; \varrho \;\texttt{if}\; \Phi$ where Φ is a fluent formula based on $\mathcal{F}$ and ε and ϱ are fluent literals.

Let (S, E) be a pair consisting of a state S and a set of fluent literals E, then a causal relationship $\varepsilon \;\texttt{causes}\; \varrho \;\texttt{if}\; \Phi$ is *applicable* to (S, E) iff $\Phi \wedge \overline{\varrho}$ is true in S and $\varepsilon \in E$. Its application yields the pair (S', E') where $S' = (S \setminus \{\overline{\varrho}\}) \cup \{\varrho\}$ and $E' = (E \setminus \{\overline{\varrho}\}) \cup \{\varrho\}$.

Let $\mathcal{R}$ be a set of causal relationships, then by $(S, E) \leadsto_{\mathcal{R}} (S', E')$ we denote the existence of an element in $\mathcal{R}$ whose application to (S, E) yields (S', E'). ■

In words, a causal relationship is applicable if the associated condition Φ holds, the particular indirect effect ϱ is currently false, and its cause ε is among the current effects. Notice that if S is a state and E is consistent, then $(S, E) \leadsto_{\mathcal{R}} (S', E')$ implies that S' is a state and E' is consistent, too.[8] In what follows, we say that a sequence of causal relationships $r_1, \ldots, r_n$ is *applicable* to a pair (S_0, E_0) iff there exist pairs $(S_1, E_1), \ldots, (S_n, E_n)$ such that for each $1 \leq i \leq n$, r_i is applicable to (S_{i-1}, E_{i-1}) and yields (S_i, E_i). We adopt a standard notation

[7] Other examples that require taking into account how an intermediate state is obtained will be discussed in Sections 3.3 and 5.3.

[8] In order to guarantee consistency of E', we remove the negation $\overline{\varrho}$ from E prior to adding ϱ. This is necessary because $\overline{\varrho}$ may have been generated as a direct or indirect effect before, which has to be withdrawn. An example where this situation occurs will be discussed below, in Section 5.2.

in writing $(S,E) \overset{*}{\rightsquigarrow}_{\mathcal{R}} (S',E')$ to indicate the existence of a (possibly empty) sequence of causal relationships in $\mathcal{R}$ which is applicable to (S,E) and yields (S',E').

Now suppose we are given a suitable set of causal relationships and have obtained a set of fluent literals S after having computed the direct effects of an action via Definition 3. State S may violate the underlying domain constraints. In order to obtain a satisfactory resulting state, we compute additional, indirect effects by (non-deterministically) selecting and (serially) applying causal relationships. If this procedure eventually results in a state satisfying the domain constraints, then this is called a *successor state*.

Definition 5 Let $\mathcal{F}$ be a set of fluent names, $\mathcal{A}$ a set of action names, $\mathcal{L}$ a set of action laws, $\mathcal{D}$ a set of domain constraints, and $\mathcal{R}$ a set of causal relationships. Furthermore, let S be a state satisfying $\mathcal{D}$, and let $a \in \mathcal{A}$ be an action name. A state S' is a *successor state* of S and a iff there exists an applicable (wrt. S) action law $\langle C, a, E\rangle \in \mathcal{L}$ such that

1. $((S \setminus C) \cup E, E) \overset{*}{\rightsquigarrow}_{\mathcal{R}} (S',E')$ for some E', and
2. S' satisfies $\mathcal{D}$.

■

Example 1 (continued) An adequate set of causal relationships for our Electric Circuit domain consists in the following four elements:

$$\begin{array}{llllllll} sw_1 & \texttt{causes} & \mathit{light} & \texttt{if} & sw_2 \qquad & sw_2 & \texttt{causes} & \mathit{light} \;\; \texttt{if} \;\; sw_1 \\ \overline{sw_1} & \texttt{causes} & \overline{\mathit{light}} & \texttt{if} & \top \qquad & \overline{sw_2} & \texttt{causes} & \overline{\mathit{light}} \;\; \texttt{if} \;\; \top \end{array} \tag{5}$$

In words, if either switch gets closed, then this causes the light bulb to turn on provided the other switch is already on. Conversely, opening either switch results in a dark bulb regardless of the other switch.

Applying the first action law in (1) to the state depicted in Figure 1, viz. $\{\overline{sw_1}, sw_2, \overline{\mathit{light}}\}$, yields the state-effect pair $(\{sw_1, sw_2, \overline{\mathit{light}}\}, \{sw_1\})$. Then the first of the given causal relationships is applicable, and its application results in $(\{sw_1, sw_2, \mathit{light}\}, \{sw_1, \mathit{light}\})$. The first component of this pair satisfies the underlying domain constraint, $sw_1 \wedge sw_2 \equiv \mathit{light}$, hence is a successor state. Moreover, it is the unique successor since no other causal relationship in (5) is applicable to $(\{sw_1, sw_2, \overline{\mathit{light}}\}, \{sw_1\})$ and since $(\{sw_1, sw_2, \mathit{light}\}, \{sw_1, \mathit{light}\})$ does not allow for further application of causal relationships. ■

Later in this paper, in Section 5.2, we will see that the order in which causal relationships are applied, might be crucial insofar as a different ordering may allow the application of different relationships. On the other hand, we can prove the following important result of order independence in case a unique set of relationships is applied:

Proposition 6 *Let $\mathcal{F}$ be a set of fluent names, S a state, and E a set of fluent literals. Furthermore, let $r_1, \ldots, r_n$ be a sequence of causal relationships ($n \geq 0$) applicable to (S,E) which yields*

$$(S,E) \rightsquigarrow_{\{r_1\}} (S_1,E_1) \rightsquigarrow_{\{r_2\}} \ldots \rightsquigarrow_{\{r_n\}} (S_n,E_n) \tag{6}$$

Then, for any permutation $r_{\pi(1)}, \ldots, r_{\pi(n)}$ which is also applicable to (S,E) and which yields

$$(S,E) \rightsquigarrow_{\{r_{\pi(1)}\}} (S'_1,E'_1) \rightsquigarrow_{\{r_{\pi(2)}\}} \ldots \rightsquigarrow_{r_{\{\pi(n)\}}} (S'_n,E'_n) \tag{7}$$

we have $S_n = S'_n$ and $E_n = E'_n$.

Proof: For each fluent name $f \in \mathcal{F}$, let k_f^+ be the number of causal relationships $r_i = \varepsilon_i$ `causes` f `if` Φ_i, and let k_f^- be the number of causal relationships $r_i = \varepsilon_i$ `causes` $\overline{f}$ `if` Φ_i $(1 \leq i \leq n)$. The application of a causal relationship ε_i `causes` ϱ_i `if` Φ_i requires that ϱ_i be true in the state at hand. Therefore, the finally resulting truth-value of f depends only on k_f^+ and k_f^-.[9] Since (6) and (7) do not differ in this respect, we know $f \in S_n$ iff $f \in S'_n$. Likewise, we have $f \in E_n$ iff $f \in E'_n$ and $\overline{f} \in E_n$ iff $\overline{f} \in E'_n$.

■

It is important to realize that neither uniqueness of a successor state nor even its existence is guaranteed in general. The former characterizes actions with non-deterministic behavior—examples of this kind will be discussed in detail later, e.g. in Section 5.2. The meaning of the latter will be elucidated in Section 3.3. First of all we raise another crucial issue, namely, how to obtain an adequate set of causal relationships on the basis of given domain constraints.

3.2 Influence Information

Obtaining the intended result by applying causal relationships to compute the indirect effects of actions relies, of course, on a suitable set of these relationships. Any of these sets should be sound insofar as each element represents an intuitively plausible causal relation, and it should also be complete insofar as it covers all conceivable indirect effects.[10] The four causal relationships (5), for instance, constitute a set suitable for the Electric Circuit domain. There is obviously a close correspondence between the elements of this set and the domain constraint underlying this example scenario, which is why the following analysis is devoted to the problem of how an adequate set of causal relationships can be automatically extracted from given domain constraints.

There is, however, a crucial, well-known obstacle to be considered towards this end [24]: Despite the fact that the causal relationships in (5) are inspired by the underlying domain constraint $sw_1 \land sw_2 \equiv \mathit{light}$, this formula would give rise to additional, unintended causal relationships if evaluated purely syntactically. For instance, recall the causal relationship sw_1 `causes` light `if` sw_2 in (5). The fact that if sw_1 becomes true then it is impossible to have both sw_2 and $\overline{\mathit{light}}$, equally well suggests the following causal relationship:

$$sw_1 \;\; \texttt{causes} \;\; \overline{sw_2} \;\; \texttt{if} \;\; \overline{\mathit{light}} \tag{8}$$

This would, however, sanction the state $\{sw_1, \overline{sw_2}, \overline{\mathit{light}}\}$ to be possible successor state of turning on sw_1 in the state depicted in Figure 1. In other words, the second switch would magically jump its position in order to satisfy the domain constraint. Though this is an unintuitive outcome, the mere domain constraint does not provide enough information to rule out (8). Hence, we need some additional domain knowledge to be able to automatically design a suitable set of causal relationships.

More precisely, we will exploit general information of potential influence of some fluents upon others. For instance, we provide the knowledge that changing a switch's position might influence

[9] More precisely, if $f \in S$ then either $k_f^+ = k_f^-$ or $k_f^+ = k_f^- - 1$. In the first case, we have $f \in S_n$ (and $f \in E_n$ iff $k_f^+ > 0$), in the second case $\overline{f} \in S_n$ (and $\overline{f} \in E_n$). The analogue holds for $\overline{f} \in S$.

[10] Here, "conceivable" can—to state the obvious—refer only to what is potentially derivable given the domain constraints. One cannot expect to obtain an indirect effect desired in some scenario if the scenario's formalization does not include a piece of knowledge hinting at the possible existence of this effect.

the state of a light bulb rather than directly causing other switches to move.[11] This kind of information is formalized as follows:

Definition 7 Let $\mathcal{F}$ be a set of fluent names. A binary relation $\mathcal{I} \subseteq \mathcal{F} \times \mathcal{F}$ on these fluent names is called *influence information*. ■

If $(f_1, f_2) \in \mathcal{I}$, then this is intended to denote that a change of f_1's truth-value potentially affects the truth-value of f_2. Regarding the Electric Circuit domain, for instance, we choose $\mathcal{I} = \{(sw_1, light), (sw_2, light)\}$, that is, both switches might influence the light but not vice versa nor do they mutually interfere.

Domain constraints plus influence information provide enough information for automatically generating an adequate set of causal relationships. The basic idea is to investigate all possible violations of a domain constraint and to formalize causal relationships which help to 'correct' this. More precisely, let $\mathcal{D}$ be the set of underlying domain constraints. We first construct the conjunctive normal form (CNF, for short) of $\bigwedge \mathcal{D}$, i.e, of the conjunction of all constraints. Then $\mathcal{D}$ is violated iff some conjunct $\ell_1 \vee \ldots \vee \ell_m$ in the CNF is violated. This in turn means that $\overline{\ell_1} \wedge \ldots \wedge \overline{\ell_m}$ holds. Since the initial state supposedly has satisfied $\mathcal{D}$, the reason for $\overline{\ell_1} \wedge \ldots \wedge \overline{\ell_m}$ being true must be some (direct or indirect) effect $\overline{\ell_j}$, and this 'flaw' can be 'corrected' by changing some other literal $\overline{\ell_k}$ to ℓ_k via a causal relationship—but only in case the fluent $|\ell_j|$ potentially affects the fluent $|\ell_k|$ according to $\mathcal{I}$. All this is formalized in the following definition:

Definition 8 Let $\mathcal{F}$ be a set of fluent names and $\mathcal{D}$ a set of domain constraints. An influence information $\mathcal{I}$ then determines a set of causal relationships $\mathcal{R}$ following this procedure:

1. Let $\mathcal{R} := \{\}$.
2. Let $D_1 \wedge \ldots \wedge D_n$ be the CNF of $\bigwedge \mathcal{D}$. For each $D_i = \ell_1 \vee \ldots \vee \ell_{m_i}$ ($i = 1, \ldots, n$) do the following:
3. For each $j = 1, \ldots, m_i$ do the following:
4. For each $k = 1, \ldots, m_i$, $k \neq j$ such that $(|\ell_j|, |\ell_k|) \in \mathcal{I}$, add this causal relationship to $\mathcal{R}$:

$$\overline{\ell_j} \ \texttt{causes} \ \ell_k \ \texttt{if} \bigwedge_{\substack{l = 1, \ldots, m_i \\ l \neq j, l \neq k}} \overline{\ell_l} \tag{9}$$

■

Notice that none of the causal relationships ε **causes** ϱ **if** Φ generated by this procedure (c.f. (9)) exploits the general expressiveness insofar as the condition Φ is, in any case, a conjunction of literals—whereas Definition 4 allows arbitrary formulas. However, this does not imply that some causal information otherwise being representable cannot be obtained by applying Definition 8. This is so because any causal relationship can be transformed into an equivalent set of causal relationships of the form (9). On the other hand, employing the general notion may lead to considerably more compact representations. More sophisticated means to extract causal

[11] The word "directly" is crucial here since switches do have the ability to influence the position of other switches indirectly, e.g., through activating a relay (see Example 3, below).

relationships from domain constraints without constructing normal forms should be developed to this end; yet this goes beyond the scope of this paper.[12,13]

Example 1 (continued) Given domain constraint $D = sw_1 \wedge sw_2 \equiv \mathit{light}$ and influence information $\mathcal{I} = \{(sw_1, \mathit{light}), (sw_2, \mathit{light})\}$, by applying Definition 8 we obtain causal relationships as follows:

- The CNF of D is $(\overline{sw_1} \vee \overline{sw_2} \vee \mathit{light}) \wedge (sw_1 \vee \overline{\mathit{light}}) \wedge (sw_2 \vee \overline{\mathit{light}})$.
- As regards the first disjunct, $D_1 = \overline{sw_1} \vee \overline{sw_2} \vee \mathit{light}$, we obtain the following:
  - In case $j = 1, k = 2$ we have $(sw_1, sw_2) \notin \mathcal{I}$.
  - In case $j = 1, k = 3$ we have $(sw_1, \mathit{light}) \in \mathcal{I}$, which yields

    $$sw_1 \ \texttt{causes} \ \mathit{light} \ \texttt{if} \ sw_2$$

  - In case $j = 2, k = 1$ we have $(sw_2, sw_1) \notin \mathcal{I}$.
  - In case $j = 2, k = 3$ we have $(sw_2, \mathit{light}) \in \mathcal{I}$, which yields

    $$sw_2 \ \texttt{causes} \ \mathit{light} \ \texttt{if} \ sw_1$$

  - In case $j = 3, k = 1$ we have $(\mathit{light}, sw_1) \notin \mathcal{I}$.
  - In case $j = 3, k = 2$ we have $(\mathit{light}, sw_2) \notin \mathcal{I}$.
- As regards the second disjunct, $D_2 = sw_1 \vee \overline{\mathit{light}}$, we obtain the following:
  - In case $j = 1, k = 2$ we have $(sw_1, \mathit{light}) \in \mathcal{I}$, which yields

    $$\overline{sw_1} \ \texttt{causes} \ \overline{\mathit{light}} \ \texttt{if} \ \top$$

  - In case $j = 2, k = 1$ we have $(\mathit{light}, sw_1) \notin \mathcal{I}$.
- As regards the third disjunct, $D_3 = sw_2 \vee \overline{\mathit{light}}$, we obtain the following:
  - In case $j = 1, k = 2$ we have $(sw_2, \mathit{light}) \in \mathcal{I}$, which yields

    $$\overline{sw_2} \ \texttt{causes} \ \overline{\mathit{light}} \ \texttt{if} \ \top$$

  - In case $j = 2, k = 1$ we have $(\mathit{light}, sw_2) \notin \mathcal{I}$.

Altogether, we obtain exactly the four causal relationships listed in (5), which we have granted above to obtain the desired solution. ■

[12] To be more precise, the task would be to construct, for any two literals ε and ϱ such that $(|\varepsilon|, |\varrho|) \in \mathcal{I}$, a causal relationship $\varepsilon \ \texttt{causes} \ \varrho \ \texttt{if} \ \Psi$ where Ψ is most simple in describing the circumstances under which effect ε gives rise to indirect effect ϱ. This might, for instance, be accomplished by collecting all causal relationships obtained via Definition 8 for ε and ϱ, i.e., $\varepsilon \ \texttt{causes} \ \varrho \ \texttt{if} \ \Phi_1, \ \ldots, \ \varepsilon \ \texttt{causes} \ \varrho \ \texttt{if} \ \Phi_m$, and defining Ψ as the most compact formula equivalent to $\Phi_1 \vee \ldots \vee \Phi_m$.

[13] A related challenge would be to find suitable deductive representations of the way domain constraints in conjunction with influence information give rise to causal relationships. This may, roughly speaking and without going into details, look like $\mathit{Causes}(\varepsilon, \varrho, \Phi) \equiv \mathit{Infl}(|\varepsilon|, |\varrho|) \wedge \forall s \, [\mathit{Holds}(\Phi, s) \wedge \mathit{Holds}(\varepsilon, s) \supset \mathit{Holds}(\varrho, s)]$. Such a deductive characterization could prove useful in a variety of particular action calculi.

A crucial issue regarding the concept of causal relationships is of course its complexity. Notice that in the worst case exponentially many causal relationships have to be generated due to the potential combinatorial explosion of the size of the domain constraints during the CNF construction. Up to quadratic many relationships exist for each resulting conjunct. Despite these pathological cases, there is, however, a decisive characteristic due to which especially in large domains the number of causal relationships is small compared to the worst case: The domain constraints do not interfere. In general, large domains tend to be locally structured insofar as each single domain constraint relates only a small number of fluents. Suppose the maximum size of a domain constraint (i.e., the number of fluent names involved) be fixed and small compared to their overall number n. Then the number of causal relationships being generated is *linear* in n. As an example, recall the situation discussed at the beginning of this section, where a distinguished switch, sw_0, affects n different sub-circuits each containing another switch, sw_i, and a bulb, $light_i$ ($1 \leq i \leq n$). The dependencies are described by n domain constraints of the form $sw_0 \wedge sw_i \equiv light_i$. As argued above, compiling all indirect effects into action laws requires 2^{n+1} different laws for toggling sw_0. In contrast, only $4n$ causal relationships are needed, viz. four for each sub-circuit similar to the ones listed in (5). Notice that it still pays regarding the computational effort when actually computing successor states since in any case at most n causal relationships have to be applied when toggling sw_0. Hence, we have avoided the exponential factor of this example in any respect.

The fact that domain constraints do not interfere in determining causal relationships avoids the second crucial problem mentioned at the beginning. No existing causal relationship has to be modified or removed if new domain constraints are added.

Before we conclude this section, let us stress that influence information always reflects the desired direction of chains of indirect effects. As argued in [41], it may sometimes be desirable to define a direction of reasoning which does not correspond to the physical reality. Consider, as an example, a light bulb being connected with two parallel switches, that is, either switch can be used to turn on the light. This is expressed by the domain constraint $sw_1 \vee sw_2 \equiv light$. Suppose we define an action *turn-on-light* in addition to $toggle_1$ and $toggle_2$ (c.f. (1)). The corresponding action law shall be $\langle \{\overline{light}\}, \mathit{turn\text{-}on\text{-}light}, \{light\} \rangle$. If this action is applied to the state $S = \{\overline{sw_1}, \overline{sw_2}, \overline{light}\}$, then, as argued in [41], one expects ramification to tell us that either sw_1 or else sw_2 becomes true in addition to the direct effect, $light$.[14] Obtaining this by means of causal relationships requires the consideration of additional directions of influence, namely, that a change of $light$ may affect sw_1 and sw_2. This is formally represented by extending the influence information used in Example 1 by $(light, sw_1)$ and $(light, sw_2)$. Based on the resulting $\mathcal{I} = \{(sw_1, light), (sw_2, light), (light, sw_1), (light, sw_2)\}$, the domain constraint $sw_1 \vee sw_2 \equiv light$ gives rise to the following causal relationships according to Definition 8:

$$\begin{array}{llllllllll} sw_1 & \texttt{causes} & light & \texttt{if} & \top & \qquad sw_2 & \texttt{causes} & light & \texttt{if} & \top \\ \overline{sw_1} & \texttt{causes} & \overline{light} & \texttt{if} & \overline{sw_2} & \qquad \overline{sw_2} & \texttt{causes} & \overline{light} & \texttt{if} & \overline{sw_1} \\ light & \texttt{causes} & sw_1 & \texttt{if} & \overline{sw_2} & \qquad light & \texttt{causes} & sw_2 & \texttt{if} & \overline{sw_1} \\ \overline{light} & \texttt{causes} & \overline{sw_1} & \texttt{if} & \top & \qquad \overline{light} & \texttt{causes} & \overline{sw_2} & \texttt{if} & \top \end{array} \tag{10}$$

Recall the state $S = \{\overline{sw_1}, \overline{sw_2}, \overline{light}\}$ and the action *turn-on-light*. The application of the aforementioned action law for *turn-on-light* yields the state-effect pair

$$(\{\overline{sw_1}, \overline{sw_2}, light\}, \{light\})$$

[14] This may not correspond to everyone's intuition, but let us accept it for the sake of argument.

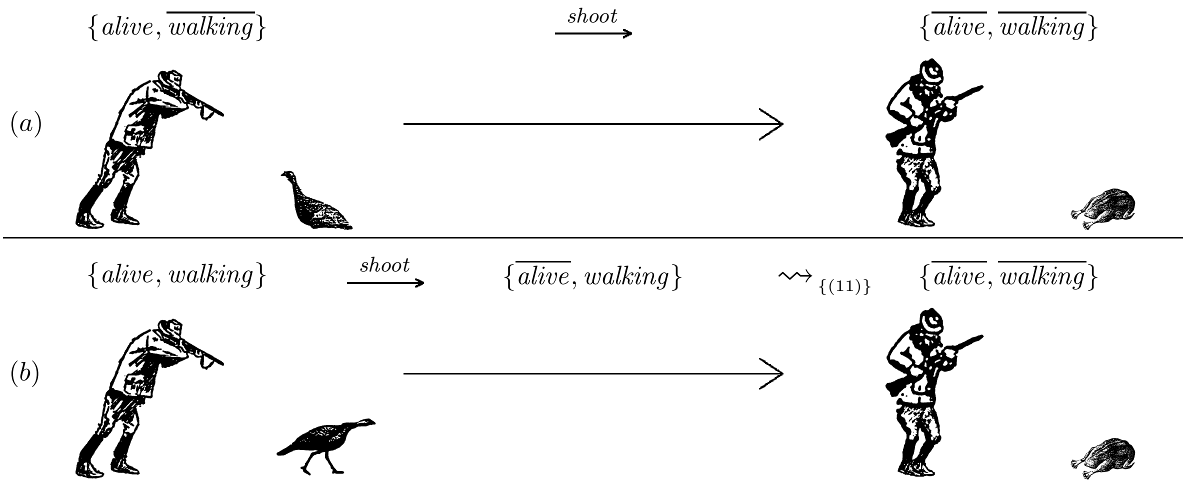

Figure 2: The application of $\langle \{alive\}, shoot, \{\overline{alive}\} \rangle$ to the initial state depicted in (a), where the turkey is not walking, results directly in a state that satisfies the domain constraint $walking \supset alive$. In case the action law is applied to the initial state depicted in (b), an additional ramification step based on (11) has to be computed in order to ensure the turkey stops walking when shot dead.

Its first component violates the underlying domain constraint. The only applicable causal relationships in (10) are the ones in the third row. The resulting state-effect pairs are

$$(\,\{sw_1, \overline{sw_2}, light\}\,,\,\{light, sw_1\}\,) \quad \text{and} \quad (\,\{\overline{sw_1}, sw_2, light\}\,,\,\{light, sw_2\}\,)$$

In both cases, the first component satisfies the domain constraint, hence constitutes a successor state. Notice that no further causal relationships in (10) are applicable to either of these state-effect pairs. Hence, the resulting states are the only successor states. This illustrates that having cyclic influence information does not necessarily imply that there are cyclic, hence infinite, chains of applications of causal relationship. The reader is invited to verify that also if $toggle_1$ or $toggle_2$ (c.f. (1)) are applied to any possible state with fluent names $\{sw_1, sw_2, light\}$, then the causal rules in (10) never support infinite application sequences.

3.3 Indirect Effects vs. Implicit Qualification

Thus far we have seen how domain constraints give rise to additional, indirect effects of actions. However, it has been observed e.g. in [15, 26] that domain constraints might instead give rise to additional, implicit *qualifications* of actions. In the following, we illustrate that the concept of causal relationships along with the notion of potential influence perfectly accounts for this distinction.

Example 2 Consider the following adaption [1] of the Yale Shooting scenario [18]. We intend to hunt a turkey which is either alive or not (described via the fluent name $alive$) and which is walking around or not (fluent name $walking$). The domain constraint $walking \supset alive$ restricts walking turkeys to vivid ones. Let $\mathcal{I} = \{(alive, walking)\}$, that is, a change of $alive$ might affect the truth-value of $walking$ but not vice versa. According to Definition 8, this determines a single causal relationship, viz.

$$\overline{alive} \ \texttt{causes} \ \overline{walking} \ \texttt{if} \ \top \tag{11}$$

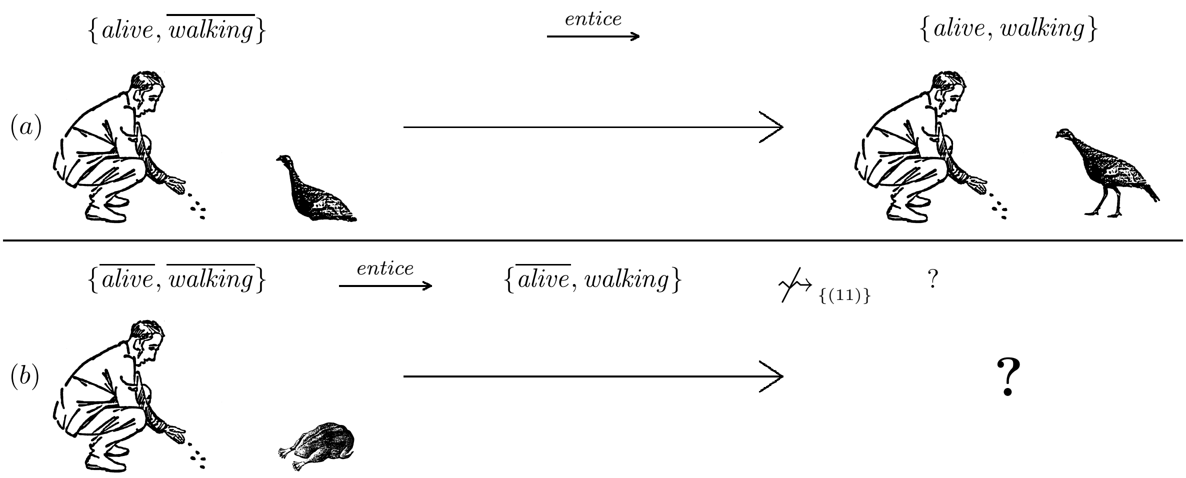

Figure 3: The application of $\langle \{\overline{walking}\}, entice, \{walking\} \rangle$ to the initial state depicted in (a), where the turkey is alive, results directly in a state that satisfies the domain constraint $walking \supset alive$. In case the action law is applied to the initial state depicted in (b), the intermediate state violates the domain constraint. This cannot be 'corrected' on the basis of the given causal relationships, (11). Hence, the action $entice$ cannot be successfully executed in a state where the turkey is not alive.

Consider, now, the action law $\langle \{alive\}, shoot, \{\overline{alive}\} \rangle$. Figure 2 illustrates the respective results of executing $shoot$ in the two states which satisfy $alive$. In case the initial state is $\{alive, \overline{walking}\}$, it is sufficient to compute the direct effect, $\overline{alive}$. If the initial state is $\{alive, walking\}$, then the underlying domain constraint gives rise to an additional, indirect effect via (11)—not only does the turkey drop dead, it also stops walking.

In contrast, suppose we want to entice the turkey if it idles [29]. The corresponding action law is $\langle \{\overline{walking}\}, entice, \{walking\} \rangle$. Figure 3 shows the respective results when $entice$ is applied to the two states which satisfy $\overline{walking}$. Again, if the initial state is $\{alive, \overline{walking}\}$, then the direct effect suffices to obtain a state satisfying the domain constraint. The initial state $\{\overline{alive}, \overline{walking}\}$ is different: Applying the aforementioned action law yields $\{\overline{alive}, walking\}$, which violates the domain constraint. Moreover, (11) is not applicable to the corresponding state-effect pair $(\{\overline{alive}, walking\}, \{walking\})$ since $\overline{alive}$ does not occur as effect. Hence, no successor state exists according to Definition 5. In other words, our domain constraint gives rise to the additional, *implicit* qualification that a turkey must be alive if we want to successfully entice it—which is exactly the desired conclusion. ■

In general, whenever no successor state exists according to Definition 5, then this hints at implicit qualifications of the action under consideration (c.f. the remark at the end of Section 3.1). This shows that providing adequate influence information gives for free, by means of causal relationships, the distinction between ramification and qualification.

4 A Fixpoint Characterization of Ramifications

The successive application of causal relationships can be regarded as a somewhat operational solution to the ramification problem. In this section, we relate our approach to the more static, fixpoint oriented characterization of indirect effects introduced in [29]. This method is based on the idea of minimizing change while respecting causal information. The objective of our comparison is to prove that all successor states satisfying the definition of [29] are also obtained by the application of causal relationships.

As we have argued, an adequate solution to the ramification problem requires more sophisticated information of causal dependencies than provided by the mere domain constraints. This is why it is insufficient to simply minimize change, i.e., to take as possible successor state any state which has minimal distance from the initial state and which satisfies both the direct effect and the underlying domain constraints. Neither does this allow for preventing unintended changes nor does it enable us to distinguish between ramifications and qualifications (c.f. Section 3.3). As a consequence, in [29] it is suggested to replace domain constraints by a suitable set of directed rules, called *causal rules*, which serve as deduction rules and are therefore weaker than the corresponding implications.

Definition 9 [29] Let $\mathcal{F}$ be a set of fluent names. A *causal rule* is an expression $\Phi \Rightarrow \Psi$ where Φ and Ψ are fluent formulas.

Let $\mathcal{C}$ be a set of causal rules. If Θ is a set of fluent formulas, then by $\mathcal{T}_{\mathcal{C}}(\Theta)$ we denote the smallest set of fluent formulas which contains Θ and is deductively closed under $\mathcal{C}$—that is,

1. $\Theta \subseteq \mathcal{T}_{\mathcal{C}}(\Theta)$;
2. for any formula θ such that $\mathcal{T}_{\mathcal{C}}(\Theta) \models \theta$ we have $\theta \in \mathcal{T}_{\mathcal{C}}(\Theta)$; and
3. if $\Phi \Rightarrow \Psi \in \mathcal{C}$ and $\Phi \in \mathcal{T}_{\mathcal{C}}(\Theta)$ then $\Psi \in \mathcal{T}_{\mathcal{C}}(\Theta)$.

If $\theta \in \mathcal{T}_{\mathcal{C}}(\Theta)$ then this denoted by $\Theta \vdash_{\mathcal{C}} \theta$. ■

Example 1 (continued) Consider the singleton set of causal rules $\mathcal{C} = \{sw_1 \wedge sw_2 \Rightarrow \mathit{light}\}$. Let $\Theta = \{sw_1, sw_2\}$, then $\mathcal{T}_{\mathcal{C}}(\Theta)$ includes *light* since the given causal rule is applicable. In contrast, let $\Theta' = \{sw_1, \overline{\mathit{light}}\}$, then $\overline{sw_2} \notin \mathcal{T}_{\mathcal{C}}(\Theta')$ despite $\overline{sw_2}$ follow from sw_1, $\overline{\mathit{light}}$, and $sw_1 \wedge sw_2 \supset \mathit{light}$. ■

Causal rules serve as the basis for a fixpoint characterization of successor states which accounts for indirect effects. Informally speaking, after having executed, in a state S, an action with direct effect E, then a state T is successor iff the following holds: T satisfies E, T is consistent with the set of causal rules, and each change of a truth-value from S to T is grounded on some causal rule. This last condition reflects the idea of minimal change.

Definition 10 [29] Let $\mathcal{F}$ be a set of fluent names, $\mathcal{A}$ a set of action names, $\mathcal{L}$ a set of action laws, and $\mathcal{C}$ a set of causal rules. Furthermore, let S be a state and $a \in \mathcal{A}$ be an action name. A state T is a *minimal change successor* of S and a iff there exists an applicable (wrt. S) action law $\langle C, a, E\rangle \in \mathcal{L}$ such that

$$T \;=\; \{\, \ell : (S \cap T) \cup E \vdash_{\mathcal{C}} \ell \,\} \tag{12}$$

i.e., T is fixpoint of the function $\lambda T. \{\ell : (S \cap T) \cup E \vdash_{\mathcal{C}} \ell\}$ given S and E. ■

Example 1 (continued) An adequate set of causal rules for the Electric Circuit domain consists of the following three elements:

$$\begin{aligned} & sw_1 \wedge sw_2 \Rightarrow \mathit{light} \\ & \overline{sw_1} \Rightarrow \overline{\mathit{light}} \\ & \overline{sw_2} \Rightarrow \overline{\mathit{light}} \end{aligned} \tag{13}$$

Let $S = \{\overline{sw_1}, sw_2, \overline{light}\}$ as depicted in Figure 1, and suppose we apply the action law $\langle\{\overline{sw_1}\}, toggle_1, \{sw_1\}\rangle$. The only minimal change successor is $T = \{sw_1, sw_2, light\}$: We have $(S \cap T) \cup E = \{sw_2\} \cup \{sw_1\}$, and the causal rules in (13) allow to additionally derive $light$. In contrast, the unintended state $T' = \{sw_1, \overline{sw_2}, \overline{light}\}$ does not satisfy equation (12): $(S \cap T') \cup E = \{\overline{light}\} \cup \{sw_1\}$, which does not allow for deriving the missing literal, $\overline{sw_2}$.[15] ■

The following observation justifies the naming "minimal change successor": Each state T satisfying (12) has minimal distance from S, that is, there is no state T' with less (wrt. set inclusion) changes while also satisfying E and the rules in $\mathcal{C}$:[16]

Observation 11 *Let $\mathcal{F}$ be a set of fluent names, $\mathcal{C}$ a set of causal rules, S a state, and E a consistent set of fluent literals. Then for any two states T, T' satisfying (12), if $S \cap T \supseteq S \cap T'$ then $T = T'$.*

> **Proof:** Since each of S, T, T' is a state, $S \cap T = S \cap T'$ implies $T = T'$. Moreover, assuming $S \cap T \supsetneqq S \cap T'$ leads to a contradiction: Let $\ell \in S \cap T$ such that $\ell \notin S \cap T'$, i.e., $\overline{\ell} \in T'$. Since T is consistent and satisfies (12), we have $(S \cap T) \cup E \not\vdash_{\mathcal{C}} \overline{\ell}$. On the other hand, we know $(S \cap T') \cup E \vdash_{\mathcal{C}} \overline{\ell}$ due to $\overline{\ell} \in T'$. Together, $(S \cap T) \cup E \not\vdash_{\mathcal{C}} \overline{\ell}$ and $(S \cap T') \cup E \vdash_{\mathcal{C}} \overline{\ell}$ show that $\mathcal{T}_{\mathcal{C}}(S \cap T') \not\subseteq \mathcal{T}_{\mathcal{C}}(S \cap T)$. This contradicts $S \cap T \supseteq S \cap T'$.[17]
>
> ■

In what follows, we restrict attention to *non-disjunctive* causal rules $\Phi \Rightarrow \Psi$, where Ψ is a conjunction of literals.[18] Since $\Phi \Rightarrow \ell_1 \wedge \ldots \wedge \ell_n$ ($n \geq 0$) is equivalent to the n rules $\Phi \Rightarrow \ell_1, \ldots, \Phi \Rightarrow \ell_n$, we assume, without loss of generality, that the consequent of a rule is a single fluent literal. Moreover, computing some $\mathcal{T}_{\mathcal{C}}(\Theta)$ is only needed in equation (12), where Θ is guaranteed to be a set of literals. Thus, each causal rule of the form $\Phi_1 \vee \Phi_2 \Rightarrow \ell$ can equivalently be replaced by $\Phi_1 \Rightarrow \ell$ plus $\Phi_2 \Rightarrow \ell$. This allows us to assume each causal rule be of the form $\ell_1 \wedge \ldots \wedge \ell_m \Rightarrow \ell$ where $\ell, \ell_1, \ldots, \ell_m$ ($m \geq 0$) are fluent literals. In what follows, for notational convenience we will formally treat the antecedent of a causal rule, Φ, as a set of literals. The conjunction of these literals will be denoted by $\bigwedge \Phi$.

The main result of this section will be a proof that each minimal change successor can be obtained by applying our approach developed in Section 3. Not only does this verify that our method covers all intuitively expected successor states with minimal distance from the original state, it also provides a means to actually *compute* minimal change successors. Notice that, following equation (12), these states have to be guessed prior to testing whether they satisfy the condition of Definition 10.

In view of the intended result, we first present a pseudo-iterative characterization of minimal change successors and prove its adequacy:

[15] It is also interesting to see why $T'' = \{sw_1, sw_2, \overline{light}\}$, where only the direct effect is computed, does not satisfy equation (12): $(S \cap T'') \cup E = \{sw_2, \overline{light}\} \cup \{sw_1\}$ allows to additionally derive $light$ given (13); thus, T'' is not a fixpoint. This illustrates that all formulas $\Phi \supset \Psi$ induced by causal relationships $\Phi \Rightarrow \Psi$ hold in minimal change successors.

[16] Observation 11 is a consequence of a theorem stated and proved in [29], which essentially relates Definition 10 to the basic definition of the *possible models approach* [48]. Below, we provide a direct proof.

[17] Notice that $\mathcal{T}_{\mathcal{C}}$ is obviously monotone, that is, $\Theta \subseteq \Theta'$ always implies $\mathcal{T}_{\mathcal{C}}(\Theta) \subseteq \mathcal{T}_{\mathcal{C}}(\Theta')$ (c.f. Definition 9).

[18] A brief discussion on the nature of disjunctive causal rules can be found at the end of this section.

Proposition 12 *Let $\mathcal{F}$ be a set of fluent names, $\mathcal{C}$ a set of causal rules, S a state, and E a set of literals. For each state T we define*

1. $\Gamma_0(T) := (S \cap T) \cup E$
2. $\Gamma_i(T) := \Gamma_{i-1}(T) \cup \{\ell : \Phi \Rightarrow \ell \in \mathcal{C}$ *and* $\Phi \subseteq \Gamma_{i-1}(T)\}$ *, for* $i = 1, 2, \ldots$ [19]

Then T satisfies (12) iff $T = \bigcup_{i=0}^{\infty} \Gamma_i(T)$ *.*

Proof: We have to prove $\{\ell : (S \cap T) \cup E \vdash_{\mathcal{C}} \ell\} = \bigcup_{i=0}^{\infty} \Gamma_i(T)$.

"*LHS*$\subseteq$*RHS*":

Let $\ell \in \mathit{LHS}$. In case $\ell \in (S \cap T) \cup E$, we find that $\ell \in \Gamma_0(T) \subseteq \bigcup_{i=0}^{\infty} \Gamma_i(T)$. Otherwise, $(S \cap T) \cup E \vdash_{\mathcal{C}} \ell$ implies the existence of a finite sequence $\Phi_1 \Rightarrow \ell_1, \ldots, \Phi_n \Rightarrow \ell_n$ of inference rules in $\mathcal{C}$ ($n \geq 1$) such that $\ell = \ell_n$ and, for each $1 \leq i \leq n$, $\Phi_i \subseteq (S \cap T) \cup E \cup \{\ell_1, \ldots, \ell_{i-1}\}$. Consequently, $\ell \in \Gamma_n(T) \subseteq \bigcup_{i=0}^{\infty} \Gamma_i(T)$.

"*LHS*$\supseteq$*RHS*":

By induction on i , we prove $\Gamma_i(T) \subseteq \{\ell : (S \cap T) \cup E \vdash_{\mathcal{C}} \ell\}$. The base case, $i = 0$, holds by definition since $\Gamma_0(T) = (S \cap T) \cup E$. For $i > 0$ let $\ell \in \Gamma_i(T)$ such that there exists some $\Phi \Rightarrow \ell \in \mathcal{C}$ where $\Phi \subseteq \Gamma_{i-1}(S, T)$. The induction hypothesis for $\Gamma_{i-1}(T)$ implies $(S \cap T) \cup E \vdash_{\mathcal{C}} \bigwedge \Phi$, hence $(S \cap T) \cup E \vdash_{\mathcal{C}} \ell$.

■

This alternative characterization of minimal change successors forms the basis for proving the formal relation between this concept and the application of causal relationships. To this end, each causal rule $\Phi \Rightarrow \ell$ determines a corresponding set of causal relationships. It also induces the domain constraint $\bigwedge \Phi \supset \ell$, which has to be satisfied by a state resulting from the successful application of a series of causal relationships. Besides exploiting Proposition 12, the crucial point in the following proof is to ensure that whenever a causal rule is actually applied to justify an indirect effect, then a corresponding causal relationship ε `causes` ϱ `if` Φ is also applicable. The latter particularly requires ε occur in the current set of previously obtained (direct or indirect) effects E , i.e., the second component of the current sate-effect pair (S, E) .

Theorem 13 *Let $\mathcal{F}$ be a set of fluent names, $\mathcal{A}$ a set of action names, $\mathcal{L}$ a set of action laws, and $\mathcal{C}$ a set of causal rules. Furthermore, let $\mathcal{D} = \{\bigwedge \Phi \supset \ell : \Phi \Rightarrow \ell \in \mathcal{C}\}$ be a set of domain constraints, and let $\mathcal{R}$ be a set of causal relationships containing for each $\{\varphi_1, \ldots, \varphi_n\} \Rightarrow \ell \in \mathcal{C}$ and each $1 \leq i \leq n$ the element*

$$\varphi_i \;\; \texttt{causes} \;\; \ell \;\; \texttt{if} \;\; \varphi_1 \wedge \ldots \wedge \varphi_{i-1} \wedge \varphi_{i+1} \wedge \ldots \wedge \varphi_n \tag{14}$$

Let S be a state which satisfies $\mathcal{D}$, and let $a \in \mathcal{A}$ be an action name, then each minimal change successor T (wrt. $\mathcal{C}$) of S and a is successor state (wrt. $\mathcal{R}$ and $\mathcal{D}$) of S and a .

Proof: Let $\langle C, a, E\rangle \in \mathcal{L}$ be any action law such that $C \subseteq S$. We prove by induction that for each $i \in \mathbb{N}_0$ there exists a pair (S_i, E_i) such that $((S \setminus C) \cup E, E) \overset{*}{\leadsto}_{\mathcal{R}} (S_i, E_i)$ and $\Gamma_i(T) \subseteq S_i$ and $\Gamma_i(T) \setminus S \subseteq E_i$ (c.f. Proposition 12).[20] In what follows, for the sake of readability we abbreviate $\Gamma(T)$ by Γ .

[19] Recall that Φ is considered a set of literals.

[20] The very last condition ensures the aforementioned applicability of all relevant causal relationships as regards the set E_i of previously obtained (direct or indirect) effects.

In case $i = 0$, $S_0 := (S \setminus C) \cup E$ and $E_0 := E$ satisfy the conditions: We have $\Gamma_0 = (S \cap T) \cup E \subseteq (S \setminus C) \cup E$ since $|C| = |E|$ and Γ_0 is consistent. Furthermore, $\Gamma_0 \setminus S = ((S \cap T) \cup E) \setminus S \subseteq E$.

For the induction step, let $i > 0$ and (S_{i-1}, E_{i-1}) satisfy the claim. Then, let

$$\{\ell_1, \ldots, \ell_m\} \;:=\; \{\ell \,:\, \ell \in \Gamma_i \text{ and } \ell \notin \Gamma_{i-1}\} \tag{15}$$

be the set of all literals that are added to Γ_{i-1} to obtain Γ_i. Hence, there exist m causal rules $\Phi_1 \Rightarrow \ell_1, \ldots, \Phi_m \Rightarrow \ell_m \in \mathcal{C}$ such that $\Phi_j \subseteq \Gamma_{i-1}$ for each $1 \leq j \leq m$. Let us consider the first rule, $\Phi_1 \Rightarrow \ell_1$. From the induction hypothesis we conclude $\Gamma_{i-1} \subseteq S_{i-1}$ and, consequently, $\Phi_1 \subseteq S_{i-1}$. Moreover, we can find some $\varphi \in \Phi_1$ such that $\varphi \in E_{i-1}$: Assuming the contrary, i.e., $E_{i-1} \cap \Phi_1 = \{\}$, the induction hypothesis $\Gamma_{i-1} \setminus S \subseteq E_{i-1}$ implies $(\Gamma_{i-1} \setminus S) \cap \Phi_1 = \{\}$. This implies $\Phi_1 \subseteq S$ since $\Phi_1 \subseteq \Gamma_{i-1}$, hence $\ell_1 \in S$ (since S satisfies $\mathcal{D}$, hence $\Phi_1 \supset \ell_1$). From $\ell_1 \in \Gamma_i \subseteq T$ we also know $\ell_1 \in S \cap T \subseteq \Gamma_0$. This contradicts $\ell_1 \notin \Gamma_{i-1}$ (c.f. (15)).

Thus, the causal relationship φ `causes` ℓ_1 `if` $\bigwedge(\Phi_1 \setminus \{\varphi\})$ in $\mathcal{R}$ is applicable to (S_{i-1}, E_{i-1}) provided $\overline{\ell_1} \in S_{i-1}$. But S_{i-1} is a state, that is, if it does not contain $\overline{\ell_1}$ it already contains ℓ_1, and the causal relationship need not be applied. Hence, either we can obtain $((S_{i-1} \setminus \{\overline{\ell_1}\}) \cup \{\ell_1\}, (E_{i-1} \setminus \{\overline{\ell_1}\}) \cup \{\ell_1\})$, or else we keep (S_{i-1}, E_{i-1}) and know $\ell_1 \in S_{i-1}$. Likewise we can successively proceed with all other literals, $\ell_2, \ldots, \ell_m$, provided there is no $k \in \{2, \ldots, m\}$ and $\varphi \in \Phi_k$ such that $\varphi \in E_{i-1}$ but $\varphi \notin (E_{i-1} \setminus \{\overline{\ell_1}, \ldots, \overline{\ell_{k-1}}\}) \cup \{\ell_1, \ldots, \ell_{k-1}\}$. In other words, we have to prove there is no causal relationship φ `causes` ℓ_k `if` $\bigwedge(\Phi_k \setminus \{\varphi\})$ ($2 \leq k \leq m$) which was applicable to (S_{i-1}, E_{i-1}) but is not after first having computed $\ell_1, \ldots, \ell_{k-1}$. To see why this is guaranteed, let us assume the contrary. Then $\varphi \in E_{i-1}$ and $\varphi \notin (E_{i-1} \setminus \{\overline{\ell_1}, \ldots, \overline{\ell_{k-1}}\}) \cup \{\ell_1, \ldots, \ell_{k-1}\}$ imply $\varphi \in \{\overline{\ell_1}, \ldots, \overline{\ell_{k-1}}\}$. This and $\varphi \in \Phi_k \subseteq \Gamma_{i-1}$ assert the existence of some $\ell_j \in \{\ell_1, \ldots, \ell_{m-1}\}$ such that $\overline{\ell_j} \in \Gamma_{i-1}$. In conjunction with $\ell_j \in \Gamma_i$ (c.f. (15)), this contradicts $T \supseteq \Gamma_{i-1} \cup \Gamma_i$ being consistent.

To summarize, having successfully applied all m causal relationships (whenever necessary), we obtain the two sets $S_i := (S_{i-1} \setminus \{\overline{\ell_1}, \ldots, \overline{\ell_m}\}) \cup \{\ell_1, \ldots, \ell_m\}$ and $E_i = (E_{i-1} \setminus \overline{L}) \cup L$ where $L = \{\ell_1, \ldots, \ell_m\} \setminus S$. The pair (S_i, E_i) satisfies the claim: $\Gamma_i \subseteq S_i$ (due to $\Gamma_{i-1} \subseteq S_{i-1}$ and (15)) and $\Gamma_i \setminus S \subseteq E_i$ (due to $\Gamma_{i-1} \setminus S \subseteq E_{i-1}$ and (15)).

Since there exists only a finite number of changes from S to T, we have $\bigcup_{i=0}^{\infty} \Gamma_i = \Gamma_n$ for some $n \in \mathbb{N}_0$. Because $\Gamma_n = T$ is a state, $T \subseteq S_n$ implies $T = S_n$. Consequently, $((S \setminus C) \cup E, E) \overset{*}{\rightsquigarrow}_{\mathcal{R}} (T, E_n)$, that is, T is successor state.

■

Interestingly, the converse of this theorem does not hold, that is, there might exist successor states (in the sense of Definition 5) that cannot be obtained using the fixpoint-based approach. In Section 5.3, we argue that these states are intuitive, and failing to detect them with the approach discussed in this section is due to the policy of minimizing change, which thus might be too restrictive.

Finally, recall that our result is restricted to non-disjunctive causal rules. In the remainder of this section, we briefly discuss the nature of rules involving disjunctions in their consequent, as in

$$\top \;\Rightarrow\; a \vee c \tag{16}$$

Typically, disjunctive rules are used to express non-deterministic behavior. For instance, given $S = \{\overline{a}, \overline{c}\}$, there exist two different minimal change successors wrt. (16) (suppose $E = \{\}$), viz. $T_1 = \{a, \overline{c}\}$ and $T_2 = \{\overline{a}, c\}$, respectively.[21] Notice that $T_3 = \{a, c\}$ is not a minimal change successor since merely having $a \vee c$ does not allow for concluding a nor c. The latter observation suggests that (16) could equivalently be replaced by these two non-disjunctive rules:

$$\begin{aligned} \overline{a} &\Rightarrow c \\ \overline{c} &\Rightarrow a \end{aligned} \tag{17}$$

Indeed, these rules yield the same result as above when applied to the state $S = \{\overline{a}, \overline{c}\}$. This indicates that a disjunctive rule $\Phi \Rightarrow \Psi_1 \vee \ldots \vee \Psi_n$ can often be adequately represented by the n rules

$$\Phi \wedge \overline{\Psi_1} \wedge \ldots \wedge \overline{\Psi_{i-1}} \wedge \overline{\Psi_{i+1}} \wedge \ldots \wedge \overline{\Psi_n} \;\Rightarrow\; \Psi_i$$

where $i = 1, \ldots, n$. However, (16) and (17) are not generally equivalent if additional causal rules are considered. For example, if we add these two rules to (16):

$$\begin{aligned} a \vee c &\Rightarrow a \\ a \vee c &\Rightarrow c \end{aligned} \tag{18}$$

then $T = \{a, c\}$ is minimal change successor of $S = \{\overline{a}, \overline{c}\}$ since $S \cap T = \{\}$ and $\{\} \vdash_{\{(16),(18)\}} a \wedge c$. In contrast, no minimal change successor of S wrt. $\{(17), (18)\}$ exists. Yet this example lacks significance: Notice that antecedent and consequent of the causal rules in (18) share fluent names, which is why it is hard to imagine a meaningful causal relation expressed by these rules. But it is also hard to imagine a more convincing example since adding, say, $a \vee c \Rightarrow d$ instead of (18) does *not* make (16) and (17) behave differently. This strongly suggests that requiring non-disjunctive causal rules means no severe restriction when formalizing causal information.

5 The Necessity of Causal Relationships ...

In this section, we contrast the proposal to employ causal relationships with other abstract concepts that are most widely used (often in slightly different variants) to tackle the problem of undesired indirect effects in the context of the ramification problem. Our aim is to illustrate the restrictive expressiveness of these concepts compared to our method. We accomplish this by discussing some prototypical example scenarios, which every reasonable formalism for reasoning about actions must at least be able to treat correctly.

5.1 ... Compared to Categorization-Based Approaches

The standard approach to avoid intuitively unexpected indirect effects is to introduce some sort of categorization among the underlying set of fluent names. This distinction between different, typically two or three, kinds of fluents comes along with a specific notion of preference as regards changes in one category compared to changes in other ones when computing ramifications—or, less sophisticated, only a particular category is subject to the law of persistence etc. While a variety of names for such fluent classes circulate in literature,[22] the common fundamental

[21] To see why, take $S \cap T_1 = \{\overline{c}\}$, say, which entails the missing literal, a, given $a \vee c$ via (16).

[22] E.g., *frame* vs. *non-frame* fluents [24]; *relevant* vs. *dependent* [4]; *persistent* vs. *non-persistent* [7]; or *persistent*, *remanent* and *contingent* fluents [6].

assumption of categorization-based approaches is that an appropriate categorization always exists. With a simple extension of our Electric Circuit domain, we will illustrate that the role of a fluent might be less obvious in this respect, which causes difficulties in finding a single appropriate category it belongs to. To this end, we employ the following, prototypical categorization-based definition:

Definition 14 Let $\mathcal{F}$ be a set of fluent names. Furthermore, let, for each state S, $\mathcal{F}_p(S)$ (*primary* fluents) and $\mathcal{F}_s(S)$ (*secondary* fluents)[23] be two disjoint subsets such that $\mathcal{F}_p(S) \cup \mathcal{F}_s(S) = \mathcal{F}$. Let S, T, T' be states, then T is *closer to* S than T', written $T \prec_S T'$, iff

1. either $|T \setminus S| \cap \mathcal{F}_p(S) \subsetneqq |T' \setminus S| \cap \mathcal{F}_p(S)$
2. or $|T \setminus S| \cap \mathcal{F}_p(S) = |T' \setminus S| \cap \mathcal{F}_p(S)$ and $|T \setminus S| \cap \mathcal{F}_s(S) \subsetneqq |T' \setminus S| \cap \mathcal{F}_s(S)$.

Let $\mathcal{D}$ be a set of domain constraints, $\mathcal{A}$ a set of action names, and $\mathcal{L}$ a set of action laws. If S is a state and $a \in \mathcal{A}$ an action name, then a state T is a *categorization-based successor* of S and a iff there exists an applicable (wrt. S) action law $\langle C, a, E \rangle \in \mathcal{L}$ such that

1. $E \subseteq T$;
2. T satisfies $\mathcal{D}$; and
3. there is no $T' \prec_S T$ such that $E \subseteq T'$ and T' satisfies $\mathcal{D}$.

■

In words, a state T is closer to some state S than a state T' iff S and T differ in less (wrt. set inclusion) primary fluents than S and T' do, or else S, T and S, T' differ in the same primary fluents but S and T differ in less secondary fluents than S and T' do. For instance, to prefer a change of the light bulb's state compared to a switch magically jumping its position in Example 1, we consider sw_1, sw_2 primary and *light* secondary in any state. Then the application of $\langle \{\overline{sw_1}\}, \mathit{toggle}_1, \{sw_1\} \rangle$ to the state $S = \{\overline{sw_1}, sw_2, \overline{\mathit{light}}\}$ admits, as intended, $T = \{sw_1, sw_2, \mathit{light}\}$ as the unique categorization-based successor since $T \prec_S T'$ for the counter-intuitive state $T' = \{sw_1, \overline{sw_2}, \overline{\mathit{light}}\}$.

Consider, now, the following extension of our electric circuit (see also Figure 4):

Example 3 We augment Example 1 by introducing a third switch, represented by the fluent name sw_3, plus a relay, represented by *relay*. If activated, the relay is intended to force the second switch (sw_2) to jump open. The relay is controlled by the first and third switch. Formally, the dependencies among all these components are described by the following domain constraints:

$$\begin{aligned} sw_1 \wedge sw_2 &\equiv \mathit{light} \\ \overline{sw_1} \wedge sw_3 &\equiv \mathit{relay} \\ \mathit{relay} &\supset \overline{sw_2} \end{aligned} \tag{19}$$

Let S denote the state of the circuit depicted in Figure 4. In order to find an adequate partition of the involved fluent names into primary and secondary fluents, respectively, observe first that we should have $sw_1, sw_2 \in \mathcal{F}_p(S)$ and $\mathit{light} \in \mathcal{F}_s(S)$ as above. For if we toggle either of sw_1 or sw_2, then we prefer a change of *light* instead of a change of the other switch (as regards

[23] The terms "primary" and "secondary," respectively, were inspired by [40].

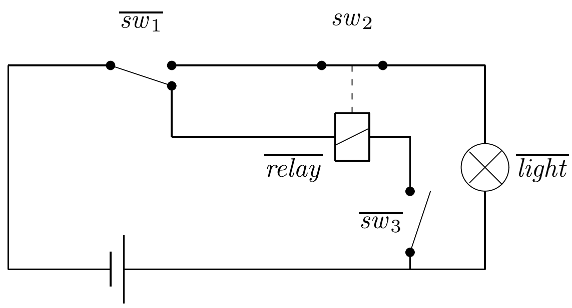

Figure 4: An extended electric circuit described by five fluents. The two possible states of the first switch are up (sw_1 is true) and down (sw_1 is false). The current state is described by $\overline{sw_1}$ (the first switch is down), sw_2 (the second switch is closed), $\overline{sw_3}$ (the third switch is open), $\overline{relay}$ (the relay is deactivated) and $\overline{light}$ (the light bulb is off).

the first domain constraint). Analogously, we should have $sw_1 \in \mathcal{F}_p(S)$ and $relay \in \mathcal{F}_s(S)$. For if we close the third switch, then we prefer a change of *relay* instead of a change of the first switch (as regards the second domain constraint). Hence, we obtain $sw_1, sw_2 \in \mathcal{F}_p(S)$ and $light, relay \in \mathcal{F}_s$.[24]

Suppose, now, we close the third switch, sw_3. Obviously, the expected result is that the relay becomes activated, which in turn causes the second switch, sw_2, jumping its position. Indeed, the corresponding state $\{\overline{sw_1}, \overline{sw_2}, sw_3, relay, \overline{light}\}$ is a categorization-based successor of $S = \{\overline{sw_1}, sw_2, \overline{sw_3}, \overline{relay}, \overline{light}\}$ and $a = toggle_3$ (given the action law $\langle \{\overline{sw_3}\}, toggle_3, \{sw_3\} \rangle$): Besides the direct effect $\{sw_3\}$, the above domain constraints suggest that a second primary fluent, sw_1 or sw_2, must change its truth value since any state with $\overline{sw_1}, sw_2, sw_3$ being true violates (19). However, the observation that a second primary fluent has to be changed suggests another categorization-based successor, namely, where sw_1 changes its truth-value instead of sw_2: The reader is invited to verify that the state $\{sw_1, sw_2, \overline{sw_3}, \overline{relay}, light\}$, too, satisfies the conditions of Definition 14 as it does not violate the domain constraints and has minimal distance to S. Hence, we obtain a second successor state where the first switch magically opens, the relay remains deactivated and the light bulb turns on. ■

The reason for the unexpected second state to occur in this example is that we necessarily fail to assign a unique, appropriate category to fluent sw_2, whose role is twofold: On the one hand, it should be considered primary (regarding the sub-circuit involving sw_1 and *light*), and on the other hand, it behaves like a secondary fluent (as regards the relay). One might suggest that this particular example could be modeled by just introducing an additional category of, say, *tertiary* fluents, $\mathcal{F}_t(S)$, which have even lower priority than secondary fluents. Then, taking $sw_1 \in \mathcal{F}_p(S)$, $sw_2, relay \in \mathcal{F}_s(S)$, and $light \in \mathcal{F}_t(S)$ and extending Definition 14 appropriately yields the expected unique resulting state. However, besides the somehow strange categorization, where similar entities, namely switches, belong to different categories, this particular classification requires a deeper analysis of possible direct and indirect effects in the electric circuit and is far from being intuitively plausible. Moreover, it is not hard to imagine more complex domains requiring more and more categories, which heavily increases the difficulty of deciding to which class a particular fluent name should belong.

[24] Whether sw_3 is considered primary or secondary is irrelevant for our argument.

In contrast, since causal relationships in conjunction with influence information only describe local dependencies, they can easily deal with fluents which behave differently regarding different domain constraints:

Example 3 (continued) The possible influences in the electric circuit depicted in Figure 4 are represented by this relation:

$$\mathcal{I} \;=\; \{(sw_1, \mathit{light}), (sw_2, \mathit{light}), (sw_1, \mathit{relay}), (sw_3, \mathit{relay}), (\mathit{relay}, sw_2)\}$$

Notably, this information concentrates on direct influences only, which, as we shall see, is sufficient; nothing has to be stated about the possibility that sw_3 might indirectly influence sw_2 (through the relay). Notice also that sw_2 occurs both as first and as second argument in $\mathcal{I}$, which encodes its twofold nature in this example. Applying Definition 8 to the domain constraints (19) in conjunction with $\mathcal{I}$ yields the following nine causal relationships:

$$\begin{array}{llllllll}
sw_1 & \texttt{causes} & \mathit{light} & \texttt{if} & sw_2 & sw_2 & \texttt{causes} & \mathit{light} \;\; \texttt{if} \;\; sw_1 \\
\overline{sw_1} & \texttt{causes} & \overline{\mathit{light}} & \texttt{if} & \top & \overline{sw_2} & \texttt{causes} & \overline{\mathit{light}} \;\; \texttt{if} \;\; \top \\
\\
\overline{sw_1} & \texttt{causes} & \mathit{relay} & \texttt{if} & sw_3 & sw_3 & \texttt{causes} & \mathit{relay} \;\; \texttt{if} \;\; \overline{sw_1} \\
sw_1 & \texttt{causes} & \overline{\mathit{relay}} & \texttt{if} & \top & \overline{sw_3} & \texttt{causes} & \overline{\mathit{relay}} \;\; \texttt{if} \;\; \top \\
\\
\mathit{relay} & \texttt{causes} & \overline{sw_2} & \texttt{if} & \top & & &
\end{array} \tag{20}$$

The topmost four relationships are obtained from the domain constraint $sw_1 \wedge sw_2 \equiv \mathit{light}$ as described in detail in Section 3.2; the domain constraint $\overline{sw_1} \wedge sw_3 \equiv \mathit{relay}$ yields, in a similar way, the next four relationships; and, finally, from $\mathit{relay} \supset \overline{sw_2}$ we obtain the last relationship due to $(\mathit{relay}, sw_2) \in \mathcal{I}$.

Suppose given the state depicted in Figure 4, i.e., $S = \{\overline{sw_1}, sw_2, \overline{sw_3}, \overline{\mathit{relay}}, \overline{\mathit{light}}\}$, plus the action law $\langle \{\overline{sw_3}, \mathit{toggle}_3, \{sw_3\}\rangle$. Then the starting point for the application of causal relationships is the state-effect pair

$$(\, \{\overline{sw_1}, sw_2, sw_3, \overline{\mathit{relay}}, \overline{\mathit{light}}\}\, ,\; \{sw_3\}\,) \tag{21}$$

The state violates the second domain constraint in (19). The only applicable causal relationship in (20) is sw_3 `causes` relay `if` $\overline{sw_1}$. Its application yields

$$(\, \{\overline{sw_1}, sw_2, sw_3, \mathit{relay}, \overline{\mathit{light}}\}\, ,\; \{sw_3, \mathit{relay}\}\,)$$

Now the state violates the third domain constraint in (19). The corresponding causal relationship in (20), namely, relay `causes` $\overline{sw_2}$ `if` $\top$, is the only applicable one. Its application yields

$$(\, \{\overline{sw_1}, \overline{sw_2}, sw_3, \mathit{relay}, \overline{\mathit{light}}\}\, ,\; \{sw_3, \mathit{relay}, \overline{sw_2}\}\,)$$

The first argument satisfies the underlying domain constraints and, consequently, denotes a successor state. Since there is no alternative way of applying causal relationships to (21), there are no other successor states. This is exactly the desired conclusion in this example. ■

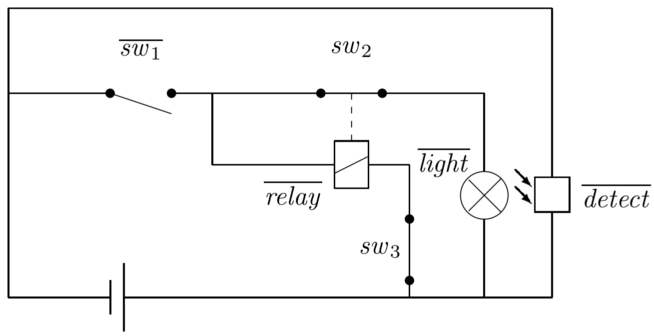

Figure 5: A modified electric circuit (c.f. Figure 4) augmented by a device, represented by *detect* , which registers an activation of the light bulb (this device combines a phototransistor and flipflop). The current state is described as in Figure 4 except that now sw_3 holds (the third switch is closed) and the state of the additional device is assumed to be $\overline{detect}$ (no action of light has occurred).

5.2 ... Compared to the Policy of Minimal Change

A widely accepted assumption concerning the ramification problem says that generating indirect effects ought to satisfy the property of minimal change. Regardless of possible categorizations as discussed above, whenever a 'proper' successor state is strictly closer to the original state than another 'proper' successor state, then the latter should be rejected. Yet while the result proved in Section 4 shows that our method covers all states that confess to the minimal change policy, it is not restricted in this respect. The following example shows the importance of this insofar as requiring minimal change might fail to obtain all intuitively expected possible successor states, hence might lead to unintended conclusions.

Example 4 The extended electric circuit from Example 3 is slightly modified and further augmented by a light detecting device (fluent name *detect*) that becomes (and stays) activated as soon as the light bulb turns on (see Figure 5). The new arrangement is encoded by the following domain constraints:

$$\begin{aligned} sw_1 \wedge sw_2 &\equiv light \\ sw_1 \wedge sw_3 &\equiv relay \\ relay &\supset \overline{sw_2} \\ light &\supset detect \end{aligned} \tag{22}$$

After enhancing the influence information $\mathcal{I}$ used in Example 3 by $(light, detect)$, these domain constraints give rise to the following causal relationships:

$$
\begin{array}{llllllll}
sw_1 & \texttt{causes} & light & \texttt{if} & sw_2 & sw_2 & \texttt{causes} & light \ \ \texttt{if} \ \ sw_1 \\
\overline{sw_1} & \texttt{causes} & \overline{light} & \texttt{if} & \top & \overline{sw_2} & \texttt{causes} & \overline{light} \ \ \texttt{if} \ \ \top \\
\\
sw_1 & \texttt{causes} & relay & \texttt{if} & sw_3 & sw_3 & \texttt{causes} & relay \ \ \texttt{if} \ \ sw_1 \\
\overline{sw_1} & \texttt{causes} & \overline{relay} & \texttt{if} & \top & \overline{sw_3} & \texttt{causes} & \overline{relay} \ \ \texttt{if} \ \ \top \\
\\
relay & \texttt{causes} & \overline{sw_2} & \texttt{if} & \top & light & \texttt{causes} & detect \ \ \texttt{if} \ \ \top
\end{array}
\tag{23}
$$

Suppose we toggle the first switch, sw_1 , in the state depicted in Figure 5. What is the expected outcome? Obviously, the relay gets activated and, then, attracts the second switch, sw_2 . Hence, the latter is open in the finally resulting state. Notice, however, that as soon as the first switch is closed, the sub-circuit involving the light bulb gets closed, too. This may activate the light bulb for an instant, that is, before the second switch jumps its position as a result of activating the relay. If this is indeed the case, then this short-time activation might be registered by the photo device, $detect$. Hence, while it is clear that the light bulb is off in the resulting state, it may or may not be the case that $detect$ becomes true. We therefore should expect two different successor states, viz. $T_1 = \{sw_1, \overline{sw_2}, sw_3, relay, \overline{light}, \overline{detect}\}$ and $T_2 = \{sw_1, \overline{sw_2}, sw_3, relay, \overline{light}, detect\}$. Notice that these states differ only in the value of the fluent $detect$. Notice also that T_1 and the initial state, $S = \{\overline{sw_1}, sw_2, sw_3, \overline{relay}, \overline{light}, \overline{detect}\}$, differ in strictly less fluents than T_2 and S do.

Our set of causal relationships (23) supports this conclusion. The application of $\langle\{\overline{sw_1}\}, toggle_1, \{sw_1\}\rangle$ to S yields the state-effect pair

$$
(\{sw_1, sw_2, sw_3, \overline{relay}, \overline{light}, \overline{detect}\}, \{sw_1\}) \tag{24}
$$

The first component violates both the first and the second domain constraint in (22). There are several possibilities to proceed. First, we can apply sw_1 `causes` $relay$ `if` sw_3 followed by $relay$ `causes` $\overline{sw_2}$ `if` $\top$, which results in

$$
(\{sw_1, \overline{sw_2}, sw_3, relay, \overline{light}, \overline{detect}\}, \{sw_1, relay, \overline{sw_2}\})
$$

The first argument satisfies the underlying domain constraints and, consequently, denotes a successor state (as expected, c.f. T_1 above).

Another possibility to proceed with the state-effect pair (24) is to apply the following chain of causal relationships:

$$
\begin{array}{rlll}
sw_1 & \texttt{causes} & light & \texttt{if} \ \ sw_2 \\
sw_1 & \texttt{causes} & relay & \texttt{if} \ \ sw_3 \\
relay & \texttt{causes} & \overline{sw_2} & \texttt{if} \ \ \top \\
\overline{sw_2} & \texttt{causes} & \overline{light} & \texttt{if} \ \ \top
\end{array}
\tag{25}
$$

In words, we first conclude the light bulb turns on due to the second switch being on. However, since the activation of the relay causes sw_2 to become false, we have to 'turn off' the light bulb again via the finally applied causal relationship. Thus, we obtain the pair

$$
(\{sw_1, \overline{sw_2}, sw_3, relay, \overline{light}, \overline{detect}\}, \{sw_1, relay, \overline{sw_2}, \overline{light}\})
$$

Again, we obtain the successor state T_1 . But we have not considered all possibilities yet. Recall the chain of causal relationships in (25). Since *light* is true and among the current effects after having applied the first of these relationships and remains true until the last one is applied, we can insert the causal relationship light `causes` detect `if` $\top$ somewhere in between. This additionally causes *detect* become true, that is, the finally obtained state-effect pair is

$$(\{sw_1, \overline{sw_2}, sw_3, \mathit{relay}, \overline{\mathit{light}}, \mathit{detect}\} , \{sw_1, \mathit{detect}, \mathit{relay}, \overline{sw_2}, \overline{\mathit{light}}\})$$

Its first component is identical to the second expected successor state, T_2 . No further successor states can be obtained, which is exactly the desired conclusion.

To see why no minimal change-based formalism can possibly obtain this, consider these causal rules $\mathcal{C}$:

$$\begin{array}{rclcrclcrcl} sw_1 \wedge sw_2 & \Rightarrow & \mathit{light} & & \overline{sw_1} & \Rightarrow & \overline{\mathit{light}} & & \overline{sw_2} & \Rightarrow & \overline{\mathit{light}} \\ sw_1 \wedge sw_3 & \Rightarrow & \mathit{relay} & & \overline{sw_1} & \Rightarrow & \overline{\mathit{relay}} & & \overline{sw_3} & \Rightarrow & \overline{\mathit{relay}} \\ \mathit{relay} & \Rightarrow & \overline{sw_2} & & \mathit{light} & \Rightarrow & \mathit{detect} & & & & \end{array}$$

in conjunction with S as above and $E = \{sw_1\}$. While $T_1 = \{sw_1, \overline{sw_2}, sw_3, \mathit{relay}, \overline{\mathit{light}}, \overline{\mathit{detect}}\}$ satisfies equation (12), $T_2 = \{sw_1, \overline{sw_2}, sw_3, \mathit{relay}, \overline{\mathit{light}}, \mathit{detect}\}$ does not since

$$(S \cap T_2) \cup E \;=\; \{sw_3, \overline{\mathit{light}}\} \cup \{sw_1\} \;\not\vdash_{\mathcal{C}}\; \mathit{detect}$$

State T_1 being the only minimal change successor suggests that *detect* necessarily remains false, which is obviously too optimistic a conclusion. ■

This last observation suggests that minimization is not always an adequate concept for distinguishing between possible indirect effects on the one hand, and unfounded changes on the other hand. In fact, the aim of generating ramifications is not to minimize change but to avoid changes that are not caused, which, as we have seen, need not be identical.

5.3 ... Compared to Causal Rules

We have seen in Section 3.3 that domain constraints may give rise to implicit qualifications instead of indirect effects. By Example 2 we have illustrated that sometimes even a single constraint acts in both fashions, depending on which effect actually occurs. Both causal relationships and causal rules allow for modeling this distinction.[25] However, the expressiveness of causal rules with this respect is limited compared to the concept of causal relationships. The reason is that the applicability of a causal rule like $\ell_1 \wedge \ell_2 \Rightarrow \ell$, say, is not restricted to situations where ℓ_1 occurs as effect (causing ℓ as ramification in case ℓ_2 being true), while ℓ_2 occurring as effect (with ℓ_1 being true and ℓ being false) shall indicate an implicit qualification. In contrast, causal relationships support this sophistication,[26] which is required in the following scenario:

Example 5 Let us consider a more subtle, ancient method to hunt turkeys, namely, by using a (manually activated) trapdoor. The state of this trapdoor is described using the fluent name

[25] Regarding Example 2 with domain constraint $\mathit{walking} \supset \mathit{alive}$, this is achieved by taking the causal relationship $\overline{\mathit{alive}}$ `causes` $\overline{\mathit{walking}}$ `if` $\top$ but not $\mathit{walking}$ `causes` alive `if` $\top$; similarly, one would employ the causal rule $\overline{\mathit{alive}} \Rightarrow \overline{\mathit{walking}}$ but not $\mathit{walking} \Rightarrow \mathit{alive}$ [29].

[26] This is why n different relationships are needed to represent a rule with n literals in its antecedent (c.f. (14) in Theorem 13).

trap-open . The fluent name *at-trap* describes whether the turkey is in the dangerous zone or not, and the fluent name *alive* is used as before. The ground underneath the trapdoor is designed such that if the turkey finds itself being *at-trap* and the trapdoor is open, then it cannot be alive. This is represented by the following domain constraint:

$$\mathit{at\text{-}trap} \wedge \mathit{trap\text{-}open} \supset \overline{\mathit{alive}} \tag{26}$$

We can open the trapdoor via the action law $\langle \{\overline{\mathit{trap\text{-}open}}\}, \mathit{open}, \{\mathit{trap\text{-}open}\}\rangle$ and entice the turkey via $\langle \{\overline{\mathit{at\text{-}trap}}, \mathit{alive}\}, \mathit{entice}, \{\mathit{at\text{-}trap}, \mathit{alive}\}\rangle$. While the state of the trapdoor can possibly influence the victim's state of being alive or not, the turkey is alert to the extent that it would never kill itself by moving towards the open trapdoor; thus, *at-trap* can never influence *alive* . The latter is intended to give rise to the implicit qualification $\overline{\mathit{trap\text{-}open}}$ for *entice* . Hence, the adequate influence information is $\mathcal{I} = \{(\mathit{trap\text{-}open}, \mathit{alive})\}$. In conjunction with (26), this determines a single causal relationship according to Definition 8, namely,

$$\mathit{trap\text{-}open} \;\; \texttt{causes} \;\; \overline{\mathit{alive}} \;\; \texttt{if} \;\; \mathit{at\text{-}trap} \tag{27}$$

Given the state $S = \{\mathit{alive}, \mathit{at\text{-}trap}, \overline{\mathit{trap\text{-}open}}\}$ (say, after having enticed the turkey), executing *open* yields $\{\mathit{alive}, \mathit{at\text{-}trap}, \mathit{trap\text{-}open}\}$ as intermediate state, which violates our domain constraint. Since *trap-open* occurred as effect, the causal relationship (27) is applicable, which results in the expected successor state $\{\overline{\mathit{alive}}, \mathit{at\text{-}trap}, \mathit{trap\text{-}open}\}$.

In contrast, consider the state $S = \{\mathit{alive}, \overline{\mathit{at\text{-}trap}}, \mathit{trap\text{-}open}\}$ and the action *entice* , whose execution yields the intermediate state $\{\mathit{alive}, \mathit{at\text{-}trap}, \mathit{trap\text{-}open}\}$, too. But now (27) is not applicable since *trap-open* did not occur as effect, that is, no successor state exists. In other words, $\overline{\mathit{trap\text{-}open}}$ is an additional, implicit qualification for *entice* , which is exactly the intended result. Notice that we are only able to distinguish between these two cases by employing (27) but not the analogous causal relationship $\mathit{at\text{-}trap}$ `causes` $\overline{\mathit{alive}}$ `if` $\mathit{trap\text{-}open}$. For both correspond to identical causal rules, namely, $\mathit{at\text{-}trap} \wedge \mathit{trap\text{-}open} \Rightarrow \overline{\mathit{alive}}$, this distinction goes beyond the expressiveness of causal rules. ■

6 A Fluent Calculus Realization

Having presented, thoroughly discussed, and demonstrated the benefits and expressiveness of our approach to the ramification problem, the second part of the paper is devoted to the development of a suitable, concrete calculus. Our encoding employs the representation technique underlying the *fluent calculus* [19, 20]. Unlike previous fluent calculus-based approaches, however, we do not develop a logic program but exploit the full expressiveness of first-order logic. We begin by defining an appropriate term representation of states in Section 6.1. In Section 6.2, we then model causal relationships and their execution. The resulting encoding is proved adequate wrt. the high-level action semantics proposed in the first part of the paper.

6.1 Reified States

The atomic elements of state descriptions have been restricted to propositional constants throughout the first part of the paper, for the sake of simplicity. For our calculus, we introduce a richer notion of fluents. A fluent is now an n-place predicate with arguments chosen from a given set of objects (or *entities*) [38, 22]. This involves both a generalized concept of action laws and fluent formulas including quantifications. Yet by requiring finiteness of any underlying set of entities, we still guarantee decidability in any respect. The following definition extends Definition 1:

Definition 15 Let $\mathcal{O}$ be a finite set of symbols called *entities*. Let $\mathcal{F}$ denote a finite set of fluent names, each of which is associated with a natural number called *arity*. A *fluent* is an expression $f(o_1, \ldots, o_n)$ where $f \in \mathcal{F}$ is of arity n and $o_1, \ldots, o_n \in \mathcal{O}$. A *fluent literal* is a fluent or its negation, denoted by $\overline{f(o_1, \ldots, o_n)}$.

Let $\mathcal{V}$ be a denumerable set of *variables*. An expression $f(t_1, \ldots, t_n)$ and its negation $\overline{f(t_1, \ldots, t_n)}$ are called *fluent expressions* iff $f \in \mathcal{F}$ is of arity n and $t_i \in \mathcal{O} \cup \mathcal{V}$ ($1 \leq i \leq n$). ■

As before, a *state* is a maximal consistent set of fluent literals. For the sake of readability, from now on we implicitly assume an arbitrary but fixed set $\mathcal{O}$ of entities, a set $\mathcal{F}$ of fluent names, and a set $\mathcal{V}$ of variables, respectively.

Example 1 (continued) The extended expressiveness is exploited to model the Electric Circuit domain as follows. Consider the two entities $\mathcal{O} = \{s_1, s_2\}$ representing the two switches, along with the unary fluent *on* describing the state of its argument. In addition, the nullary fluent *light* denotes the state of the light bulb as before. The state displayed in Figure 1 is then formalized as $S = \{\overline{on\,(s_1)}, on\,(s_2), \overline{light}\}$. ■

As opposed to the situation calculus [32, 37], the fluent calculus employs structured state terms, each of which consists in a collection of all fluent literals that are true in the state being represented. To this end, fluent literals are reified, i.e., formally represented as terms. These terms are connected via a special binary function, which is illustratively denoted by $\circ$ and written in infix notation. For instance, a term representation of the state $S = \{\overline{on\,(s_1)}, on\,(s_2), \overline{light}\}$ is

$$(\,\overline{on\,(s_1)} \circ on\,(s_2)\,) \circ \overline{light} \tag{28}$$

where the bar denoting negative fluent expressions is formally a unary function. It has first been argued in [19] that this representation technique avoids extra axioms (e.g., frame axioms [32, 17]) to encode the general law of persistence: The effects of actions can be modeled by manipulating terms like (28) through removal and addition of sub-terms. Then all sub-terms which are not affected by these operations remain in the state term, hence continue to be true.

Intuitively, the position at which a fluent literal occurs in a state term should be irrelevant. That is, (28) and the term $on\,(s_1) \circ (\overline{light} \circ \overline{on\,(s_1)})$, say, represent identical states. This intuition is modeled by requiring the following formal properties for the connection function $\circ$:

$$\begin{array}{rrcll} \forall x, y, z. & (x \circ y) \circ z & = & x \circ (y \circ z) & \text{(associativity)} \\ \forall x, y. & x \circ y & = & y \circ x & \text{(commutativity)} \\ \forall x. & x \circ \emptyset & = & x & \text{(unit element)} \end{array}$$

where the special constant $\emptyset$ denotes a unit element for $\circ$. This constant represents the empty collection of fluent literals. The above axioms constitute an *equational theory*, which we abbreviate by AC1. Given the law of associativity, from now on we omit parentheses on the level of $\circ$. Notice that the axioms AC1 formalize essential properties of the datastructure "set."[27] For formal reasons, we introduce a function τ which maps sets of fluent expressions $A = \{\ell_1, \ldots, \ell_n\}$ to their term representation $\tau_A = \ell_1 \circ \cdots \circ \ell_n$ (including $\tau_{\{\}} = \emptyset$).

In conjunction with the standard axioms of equality (see (29), below), our equational theory entails the equivalence of two state terms whenever they are built up from an identical collection

[27] The reader may wonder why we do not additionally require the function $\circ$ be idempotent. The (subtle) reason for this is given below, right after Proposition 17.

of fluent literals. These axioms alone, however, do not suffice for our encoding. For it will also be necessary to prove the inequality of two state terms whenever they do not contain the same fluent literals. This requires an extended notion of the standard so-called unique name assumption. More precisely, we adopt the concept of *unification completeness* known from logic programming (see, e.g., [21, 42, 47]). Let E be an equational theory, that is, a set of universally quantified equations. Two terms s and t are said to be *E-equal*, written $s =_E t$, iff $s = t$ is entailed by E plus the standard axioms of equality. A substitution σ is called an *E-unifier* of s and t iff $s\sigma =_E t\sigma$. A set $cU_E(s,t)$ of E-unifiers of s and t is called *complete* if it contains, for each E-unifier of s and t, a more or equally general substitution.[28] Unification completeness is then defined as follows:

Definition 16 Let E be an equational theory. A consistent set of formulas E^* is called *unification complete* wrt. E iff E^* contains the following:

1. The axioms in E.
2. The standard equality axioms, viz.

$$\begin{array}{lll} x = x & \text{(reflexivity)} & \\ x = y \;\supset\; y = x & \text{(symmetry)} & \\ x = y \wedge y = z \;\supset\; x = z & \text{(transitivity)} & (29) \\ x_i = y \;\supset\; f(x_1,\ldots,x_i,\ldots,x_n) = f(x_1,\ldots,y,\ldots,x_n) & \text{(substitutivity I)} & \\ x_i = y \;\supset\; [P(x_1,\ldots,x_i,\ldots,x_n) \equiv P(x_1,\ldots,y,\ldots,x_n) & \text{(substitutivity II)} & \end{array}$$

   for each n-place function symbol f and predicate P, and for each $1 \leq i \leq n$. All variables are universally quantified.

3. Equational formulas, i.e., formulas with "$=$" as the only predicate, such that for any two terms s and t with variables $\widetilde{x}$ the following holds:

   (a) If s and t are not E-unifiable, then $E^* \models \neg\exists\widetilde{x}.\; s \doteq t$.

   (b) If s and t are E-unifiable, then for each complete set of unifiers $cU_E(s,t)$ we have

$$E^* \;\models\; \forall\widetilde{x}\; \left[s = t \;\supset \bigvee_{\sigma\in cU_E(s,t)} \exists\widetilde{y}.\; \sigma_= \right] \tag{30}$$

   where $\widetilde{y}$ denotes the variables which occur in $\sigma_=$ but not in $\widetilde{x}$.[29]

■

As shown in [20], a unification complete theory for our axioms AC1 can be obtained by computing, for any two terms s, t, some complete set $cU_{\mathrm{AC1}}(s,t)$ of AC1-unifiers (see, e.g., [43, 5]) and adding the corresponding equational formula which is to the right of the entailment symbol in (30). In what follows, this unification complete theory will be called *extended unique*

[28] That is, whenever $s\sigma =_E t\sigma$ then there exists some $\sigma' \in cU_E(s,t)$ such that $(\sigma' \leq_E \sigma)|_{\mathcal{V}ar(s)\cup\mathcal{V}ar(t)}$. Here, $\mathcal{V}ar(t)$ denotes the set of variables occurring in term t, and $(\sigma' \leq_E \sigma)|_V$ means the existence of a substitution θ such that $(\sigma'\theta =_E \sigma)|_V$. The latter holds iff for each variable $x \in V$, the two terms $(x\sigma')\theta$ and $x\sigma$ are E-equal.

[29] By $\sigma_=$ we denote the equational formula $x_1 = t_1 \wedge \ldots \wedge x_n = t_n$ constructed from the substitution $\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$.

name assumption, abbreviated by *EUNA*. As an example, consider the terms $on(x) \circ z$ and $\overline{on(s_1)} \circ on(s_2) \circ \overline{light}$. The singleton set $\{\{x \mapsto s_2, z \mapsto \overline{on(s_1)} \circ \overline{light}\}\}$ is a complete set of AC1-unifiers of these terms. Hence,

$$EUNA \models \forall x, z\ [\ on(x) \circ z = \overline{on(s_1)} \circ on(s_2) \circ \overline{light} \supset x = s_2 \wedge z = \overline{on(s_1)} \circ \overline{light}\ ]$$

according to (30).

Before we proceed with the fluent calculus encoding, we prove some crucial properties of *EUNA*. These properties show how the subset relation and the set difference and union operations can be modeled on the term level.

Proposition 17 *Let A, B be two sets of fluent literals.*

1. *IF $A \subseteq B$ then $EUNA \models \exists z.\, \tau_A \circ z = \tau_B$, else $EUNA \models \forall z.\, \tau_A \circ z \neq \tau_B$.*
2. *If $A \subseteq B$ then $EUNA \models \forall z\, [\tau_A \circ z = \tau_B \equiv z = \tau_{B \setminus A}]$.*
3. *If $A \cap B = \{\}$ then $EUNA \models \forall z\, [z = \tau_A \circ \tau_B \equiv z = \tau_{A \cup B}]$.*

Proof:

1. In case $A \subseteq B$, let $Z = B \setminus A$, then $\tau_A \circ \tau_Z$ and τ_B are AC1-equal. According to Definition 16, this implies $EUNA \models \tau_A \circ \tau_Z = \tau_B$, hence $EUNA \models \exists z.\, \tau_A \circ z = \tau_B$. In case $A \not\subseteq B$, $\tau_A \circ z$ and τ_B are not AC1-unifiable. According to Definition 16, this implies $EUNA \models \forall z.\, \tau_A \circ z \neq \tau_B$.
2. Let z be a term then $\tau_A \circ z$ and τ_B are AC1-equal iff each fluent literal occurring in $\tau_A \circ z$ also occurs in τ_B and vice versa and no fluent literal occurs twice or more in $\tau_A \circ z$. This in turn is equivalent to z and $\tau_{B \setminus A}$ being AC1-equal (given $A \subseteq B$), hence $EUNA \models z = \tau_{B \setminus A}$.
3. A term z and the term $\tau_A \circ \tau_B$ are AC1-equal iff each fluent literal occurring in z also occurs in $\tau_A \circ \tau_B$ and vice versa and no fluent literal occurs twice or more in z (given $A \cap B = \{\}$). This in turn is equivalent to z and $\tau_{A \cup B}$ being AC1-equal, hence $EUNA \models z = \tau_{A \cup B}$.

■

For illustration, let $A = \{on(s_2)\}$, $B = \{\overline{on(s_1)}, on(s_2), \overline{light}\}$, and $C = \{on(s_1), light\}$, then $A \subseteq B$ and $A \cap C = \{\}$. Accordingly,

1. $EUNA \models \exists z.\ on(s_2) \circ z = \overline{on(s_1)} \circ on(s_2) \circ \overline{light}$
2. $EUNA \models \forall z\, [\, on(s_2) \circ z = \overline{on(s_1)} \circ on(s_2) \circ \overline{light} \equiv z = \overline{on(s_1)} \circ \overline{light}\,]$
3. $EUNA \models \forall z\, [\, z = on(s_2) \circ on(s_1) \circ light \equiv z = on(s_1) \circ light \circ on(s_2)\,]$

In particular, the equivalence expressed in clause 2 of the above proposition will be used below to model the removal of an action's condition from a state. That is, if C is the condition of some action, S is some state, and z is a term such that $EUNA \models \tau_C \circ z = \tau_S$, then we know that z represents the set $S \setminus C$. Precisely this is the reason why the function $\circ$ is not required to be idempotent. For if it would be, then $\tau_C \circ z = \tau_S$ would not imply $z = \tau_{S \setminus C}$. E.g., if $\circ$ were idempotent, then for any set A we would have $\tau_A \circ \tau_A = \tau_A$. But clearly $\tau_A \neq \tau_{A \setminus A}$

unless $A = \{\}$. In contrast, without $\circ$ being idempotent, *EUNA* entails $\tau_A \circ \tau_A \neq \tau_A$ whenever $\tau_A \neq \emptyset$, i.e., $A \neq \{\}$.

We are now prepared for defining the circumstances under which a term represents a state. We use a many-sorted logic language with four sorts, namely, fluent literals, collections (of fluent literals), action names, and entities. Collections are composed of fluent literals, the constant $\emptyset$, and our connection function $\circ$. Below, variables of the sort "fluent literal" are indicated by ℓ , variables of the sort "action name" by a , and variables of the sort "entity" by x , sometimes with sub- or superscripts. All other variables are of the sort "collection." Free variables are implicitly assumed to be universally quantified.

To begin, the following formula defines a predicate $Holds(\ell, s)$ with the intended meaning that ℓ is contained in s :

$$Holds(\ell, s) \ \equiv \ \exists z.\, \ell \circ z = s \tag{31}$$

Then the following formula determines the constitutional properties of state terms:

$$State(s) \ \equiv \ \forall \ell \, [\, Holds(\ell, s) \equiv \neg Holds(\overline{\ell}, s) \,] \ \wedge \ \forall \ell, z.\, s \neq \ell \circ \ell \circ z \tag{32}$$

In words, s represents a state if it contains each fluent literal or its negation but not both. Furthermore, no fluent literal may occur twice (or more) in s . The following proposition states the adequacy of this formalization:

Proposition 18 *Let s be a collection of fluent literals, then $EUNA, (31), (32) \models State(s)$ iff there exists some state S such that $EUNA \models s = \tau_S$, else $EUNA, (31), (32) \models \neg State(s)$.*

Proof: We have $EUNA, (31), (32) \models State(s)$ iff

$$EUNA, (31), (32) \ \models \ (\, \exists z.\, \ell \circ z = s \ \equiv \ \forall z'.\, \overline{\ell} \circ z' \neq s \,) \ \wedge \ \forall z.\, s \neq \ell \circ \ell \circ z \tag{33}$$

for each fluent literal ℓ .

" $\Rightarrow$ ":

Suppose (33) holds for each fluent literal ℓ . Let $S = \{ \ell : EUNA \models \exists z.\, \ell \circ z = s \}$, then the entailment of the first conjunct in (33) ensures that S is a state. It also ensures that S consists in the fluent literals which occur in s . Moreover, the entailment of the second disjunct in (33) ensures that no fluent literal occurs twice or more in s . Altogether this implies $EUNA \models s = \tau_S$.

" $\Leftarrow$ ":

Suppose $EUNA \models s = \tau_S$ for some state S . Then all fluent literals in τ_S occur exactly once in s . Let ℓ be a fluent literal. S being a state implies that $\ell \in S$ iff $\overline{\ell} \notin S$. Thus, clause 1 of Proposition 17 ensures that the first conjunct in (33) is entailed. Moreover, since s does not contain a literal twice or more, s and $\ell \circ \ell \circ z$ are not AC1-unifiable. This implies $EUNA \models \forall z.\, s \neq \ell \circ \ell \circ z$ according to clause 3(a) of Definition 16.

■

Next, we show how to encode domain constraints in the fluent calculus. In order to exploit the extended notion of a fluent, we allow fluent formulas to quantify over entities.

Definition 19 The set of *fluent formulas* is inductively defined as follows: Each fluent expression and $\top$ and $\bot$ are fluent formulas, and if F and G are fluent formulas then so are $F \wedge G$, $F \vee G$, $F \supset G$, $F \equiv G$, $\exists x. F$, and $\forall x. F$ (where $x \in \mathcal{V}$).

A *closed* formula is a fluent formula without free variables, that is, where each occurring variable is bound by some quantifier. Let S be a state and F a closed fluent formula, then the notion of F being *true* (resp. *false*) in S is inductively defined as follows:

1. $\top$ is true and $\bot$ is false in S;
2. a fluent literal ℓ is true in S iff $\ell \in S$;
3. $F \wedge G$ is true in S iff F and G are true in S;
4. $F \vee G$ is true in S iff F or G is true in S (or both);
5. $F \supset G$ is true in S iff F is false in S or G is true in S (or both);
6. $F \equiv G$ is true in S iff F and G are true in S, or else F and G are false in S;
7. $\exists x. F$ is true in S iff there exists some $o \in \mathcal{O}$ such that $F\{x \mapsto o\}$ is true in S;
8. $\forall x. F$ is true in S iff for each $o \in \mathcal{O}$, $F\{x \mapsto o\}$ is true in S.

Here, $F\{x \mapsto o\}$ denotes the fluent formula resulting from replacing in F all free occurrences of x by o. ■

In our Electric Circuit example, we will use the fluent formula $\forall x. \mathit{on}(x) \equiv \mathit{light}$ as the underlying domain constraint to express the intended relation between the switches and the light bulb. This formula is true, for instance, in the state $S = \{\mathit{on}(s_1), \mathit{on}(s_2), \mathit{light}\}$.

Based on the definition of the *Holds* predicate (c.f. (31)), the encoding of fluent formulas in the fluent calculus is straightforward. In order to state that a fluent formula is true in a state represented by some term s, each fluent literal ℓ occurring in this formula is replaced by the expression $\mathit{Holds}(\ell, s)$. E.g., our domain constraint above becomes

$$\forall x.\ \mathit{Holds}(\mathit{on}(x), s) \ \equiv\ \mathit{Holds}(\mathit{light}, s) \tag{34}$$

For formal reasons, we introduce a function π mapping a fluent formula F and some term s to a formula like (34). This transformation is inductively defined as follows:

$$\begin{array}{rcl}
\pi(\top, s) & = & \top \\
\pi(\bot, s) & = & \bot \\
\pi(\ell, s) & = & \mathit{Holds}(\ell, s) \\
\pi(F \wedge G, s) & = & \pi(F, s) \wedge \pi(G, s) \\
\pi(F \vee G, s) & = & \pi(F, s) \vee \pi(G, s) \\
\pi(F \supset G, s) & = & \pi(F, s) \supset \pi(G, s) \\
\pi(F \equiv G, s) & = & \pi(F, s) \equiv \pi(G, s) \\
\pi(\forall x. F, s) & = & \forall x. \pi(F, s) \\
\pi(\exists x. F, s) & = & \exists x. \pi(F, s)
\end{array} \tag{35}$$

For notational convenience, we will usually write $\mathit{Holds}(F, s)$ instead of $\pi(F, s)$. Given the definition of *Holds*, (31), and the extended unique name assumption, our encoding of fluent formulas is correct:

Proposition 20 *Let F be a fluent formula and S a state, then $EUNA, (31) \models Holds(F, \tau_S)$ iff F is true in S, else $EUNA, (31) \models \neg Holds(F, \tau_S)$.*

> **Proof:** If ℓ is a fluent literal then, according to Definition 19, ℓ is true in S iff $\{\ell\} \subseteq S$. Following Proposition 17, the latter is equivalent to $EUNA \models \exists z. \ell \circ z = \tau_S$, which in turn is equivalent to $EUNA, (31) \models Holds(\ell, \tau_S)$ according to (31) and the definition of π. The claim can then be proved by straightforward induction on the structure of F.

■

In particular, we call *possible* a state term that satisfies a given set of domain constraints $\mathcal{D}$:

$$Possible(s) \equiv \bigwedge_{D \in \mathcal{D}} Holds(D, s) \tag{36}$$

In our example scenario, this would be (c.f. (34))

$$Possible(s) \equiv [\forall x. Holds(on(x), s) \equiv Holds(light, s)]$$

We conclude this section by introducing an extended notion of action laws. An action law may now contain variables, in which case it is considered representative for all of its ground instances. In what follows, the expression $\widetilde{x}$ (resp. $\widetilde{o}$) denotes a finite sequence of variables chosen from the given set $\mathcal{V}$ (resp. entities chosen from $\mathcal{O}$) of arbitrary but fixed length. If $\widetilde{x}$ is a sequence of the variables that occur free in some expression ξ, then this is written $\xi[\widetilde{x}]$. Let $\widetilde{x} = x_1, \ldots, x_n$, then a *ground instance* of some expression $\xi[\widetilde{x}]$ is obtained by applying a substitution $\theta = \{x_1 \mapsto o_1, \ldots, x_n \mapsto o_n\}$ to ξ, where $o_1, \ldots, o_n \in \mathcal{O}$. Let $\widetilde{o} = o_1, \ldots, o_n$, then $\xi[\widetilde{x}]\theta$ is also denoted by $\xi[\widetilde{o}]$.

Definition 21 Let $\mathcal{A}$ be a finite set of action names, each of which is associated with a natural number called *arity*. An *action law* is a triple $\langle C[\widetilde{x}], a(\widetilde{x}), E[\widetilde{x}]\rangle$ where $C[\widetilde{x}]$ and $E[\widetilde{x}]$ are sets of fluent expressions and $a \in \mathcal{A}$ is of arity equal to the length of $\widetilde{x}$. It is assumed that $|C[\widetilde{o}]| = |E[\widetilde{o}]|$ for any sequence $\widetilde{o}$ of entities.

If S is a state, then a ground instance $\alpha[\widetilde{o}]$ of an action law $\alpha[\widetilde{x}] = \langle C[\widetilde{x}], a(\widetilde{x}), E[\widetilde{x}]\rangle$ is *applicable* in S iff $C[\widetilde{o}] \subseteq S$. The *application* of $\alpha[\widetilde{o}]$ to S yields $(S \setminus C[\widetilde{o}]) \cup E[\widetilde{o}]$. ■

Example 1 (continued) For our Electric Circuit domain, we define an action called $toggle(x)$ by the following two laws:

$$\langle \{\overline{on(x)}\}, toggle(x), \{on(x)\} \rangle \qquad \langle \{on(x)\}, toggle(x), \{\overline{on(x)}\} \rangle \tag{37}$$

When executing, say, $toggle(s_1)$ in $S = \{\overline{on(s_1)}, on(s_2), \overline{light}\}$, then the instance $\theta = \{x \mapsto s_1\}$ of the first action law in (37) is applicable due to $\{\overline{on(x)}\}\theta \subseteq S$. The application yields

$$(S \setminus \{\overline{on(s_1)}\}) \cup \{on(s_1)\} = \{on(s_1), on(s_2), \overline{light}\}$$

■

By defining a ternary predicate called $Action$, we encode a given set of action laws $\mathcal{L} = \{\langle C_1[\widetilde{x}_1], a_1(\widetilde{x}_1), E_1[\widetilde{x}_1]\rangle, \ldots, \langle C_n[\widetilde{x}_n], a_n(\widetilde{x}_n), E_n[\widetilde{x}_n]\rangle\}$ ($n \geq 0$) as follows:

$$Action(c, a, e) \equiv \bigvee_{i=1}^{n} \exists \widetilde{x}_i \left[c = \tau_{C_i[\widetilde{x}_i]} \wedge a = a_i(\widetilde{x}_i) \wedge e = \tau_{E_i[\widetilde{x}_i]} \right] \tag{38}$$

The application of action laws according to Definition 21 is modeled in our fluent calculus-based encoding by defining a ternary predicate called $Result$. The intended meaning is that $Result(s, a, s')$ is true iff s' represents the result of applying some instance of some action law for the action name a to the state represented by s:

$$Result(s, a, s') \;\equiv\; \exists c, e, z\, [\, Action(c, a, e) \,\wedge\, c \circ z = s \,\wedge\, s' = z \circ e\,] \tag{39}$$

Notice that the first equation, $c \circ z = s$, ensures that the condition of the action law at hand is contained in the state represented by s (c.f. clause 1 of Proposition 17). This equation also guarantees that z contains all fluent literals in s but not in c (according to clause 2 of Proposition 17). Finally, the second equation encodes the addition of the effect, e, to z (according to clause 3 of Proposition 17). The following proposition states that this encoding is correct:

Proposition 22 *Let $\mathcal{A}$ be a set of action names, $\mathcal{L} = \{\langle C_i[\widetilde{x}_i], a(\widetilde{x}_i), E[\widetilde{x}_i]\rangle : 1 \leq i \leq n\}$ a set of action laws, S a state, $a \in \mathcal{A}$ of arity m, and $\widetilde{o}$ a sequence of entities of length m. Furthermore, let s' be a collection of fluent literals. Then*

$$EUNA, (38), (39) \;\models\; Result(s, a(\widetilde{o}), s') \tag{40}$$

iff there exists an action law $\alpha[\widetilde{x}] = \langle C[\widetilde{x}], a(\widetilde{x}), E[\widetilde{x}]\rangle \in \mathcal{L}$ whose instance $\alpha[\widetilde{o}]$ is applicable in S and whose application yields a state S' such that $EUNA \models s' = \tau_{S'}$, else

$$EUNA, (38), (39) \;\models\; \neg Result(s, a(\widetilde{o}), s')$$

Proof: From (38) and (39) in conjunction with the standard equality axioms, $(29) \in EUNA$, it follows that (40) is true iff there is an instance $\alpha_i[\widetilde{o}]$ of an action law $\alpha_i[\widetilde{x}_i] = \langle C_i[\widetilde{x}_i], a(\widetilde{x}_i), E[\widetilde{x}_i]\rangle$ in $\mathcal{L}$ such that

$$EUNA \;\models\; \exists z\, (\, \tau_{C_i[\widetilde{o}]} \circ z = s \,\wedge\, s' = z \circ E_i[\widetilde{o}]\,)$$

This in turn is true iff

1. $C_i[\widetilde{o}] \subseteq S$ (according to clause 1 of Proposition 17), and
2. $EUNA \;\models\; s' = \tau_{(S \setminus C_i[\widetilde{o}]) \cup E_i[\widetilde{o}]}$ (according to clauses 2 and 3 of Proposition 17 since $(S \setminus C_i[\widetilde{o}]) \cap E_i[\widetilde{o}] = \{\}$).

Following Definition 21, these conditions are equivalent to $\alpha_i[\widetilde{o}]$ being applicable in S and resulting in $S' = (S \setminus C_i[\widetilde{o}]) \cup E_i[\widetilde{o}])$ such that $EUNA \models s' = \tau_{S'}$.

■

As before, however, the resulting state term s' may violate the underlying domain constraints, i.e., $(36) \models \neg Possible(s')$. In this case, the corresponding state term requires further manipulation by means of causal relationships. Their encoding within the fluent calculus is presented in the following section.

6.2 Executing Causal Relationships

Similar to the case of action laws, we may exploit the extended notational expressiveness to formulate causal relationships with variables in their components. These relationships are then considered representatives for all of their ground instances.

Definition 23 A *causal relationship* is an expression of the form ε `causes` ϱ `if` Φ where Φ is a fluent formula and ε and ϱ are (possibly negated) fluent expressions.

Let (S, E) be a pair consisting of a state S and a set of fluent literals E. Furthermore, let $r = \varepsilon$ `causes` ϱ `if` Φ be a causal relationship, and let $\widetilde{x}$ denote a sequence of all free variables occurring in ε, ϱ, or Φ. Then an instance $r[\widetilde{o}]$ is *applicable* to (S, E) iff $S \models \Phi[\widetilde{o}] \wedge \overline{\varrho[\widetilde{o}]}$ and $\varepsilon[\widetilde{o}] \in E$. Its application yields the pair (S', E') where $S' = (S \setminus \{\overline{\varrho[\widetilde{o}]}\}) \cup \{\varrho[\widetilde{o}]\}$ and $E' = (E \setminus \{\overline{\varrho[\widetilde{o}]}\}) \cup \{\varrho[\widetilde{o}]\}$.

Let $\mathcal{A}$ be a set of action names, $\mathcal{L}$ a set of action laws, $\mathcal{D}$ a set of domain constraints, and $\mathcal{R}$ a set of causal relationships. Furthermore, let S be a state satisfying $\mathcal{D}$, $a \in \mathcal{A}$ of arity m, and $\widetilde{o}$ a sequence of entities of length m. A state S' is a *successor state* of S and $a(\widetilde{o})$ iff there exists an applicable (wrt. S) instance $\alpha[\widetilde{o}]$ of an action law $\alpha[\widetilde{x}] = \langle C[\widetilde{x}], a(\widetilde{x}), E[\widetilde{x}] \rangle \in \mathcal{L}$ such that

1. $((S \setminus C[\widetilde{o}]) \cup E[\widetilde{o}], E[\widetilde{o}]) \overset{*}{\rightsquigarrow}_{\mathcal{R}} (S', E')$ for some E', and
2. S' satisfies $\mathcal{D}$.

■

Example 1 (continued) Based on the formalization of our Electric Circuit domain used throughout this section, we define the following two causal relationships:

$$\mathit{on}(x) \ \texttt{causes} \ \mathit{light} \ \texttt{if} \ \forall y.\, \mathit{on}(y) \qquad \overline{\mathit{on}(x)} \ \texttt{causes} \ \overline{\mathit{light}} \ \texttt{if} \ \top \tag{41}$$

Then the instance $\{x \mapsto s_1\}$ of the first one is applicable to $(\{\mathit{on}(s_1), \mathit{on}(s_2), \overline{\mathit{light}}\}, \{\mathit{on}(s_1)\})$ since $\forall y.\, \mathit{on}(y) \wedge \overline{\mathit{light}}$ is true in the first component of this state-effect pair and since $\mathit{on}(s_1)$ occurs in the second component. The application yields $(\{\mathit{on}(s_1), \mathit{on}(s_2), \mathit{light}\}, \{\mathit{on}(s_1), \mathit{light}\})$. Now the first component satisfies the underlying domain constraint (c.f. (34)), thus constitutes a successor state of $\{\overline{\mathit{on}(s_1)}, \mathit{on}(s_2), \overline{\mathit{light}}\}$ and $\mathit{toggle}(s_1)$. ■

Having the extended concept of causal relationships immediately raises the question how Definition 8 can be adapted—that is, how causal relationships with variables can be extracted from domain constraints given an appropriate notion of influence information. Doing this for arbitrary domain constraints, however, turns out to be a non-trivial problem. For instance, suppose we have a domain constraint of the form $\forall x \exists y \forall z.\, p(x, y, z)$. Suppose also that some action causes some instance, say $p(o_1, o_2, o_3)$, to become false and that the domain constraint is violated afterwards. Then we need to decide how this can be 'corrected' by making some other instance $p(o'_1, o'_2, o'_3)$ true. This obviously requires precise information on how the different instances of p interact.

Since we assume finiteness of sets of entities, it is generally possible to rewrite any domain constraint so that it becomes quantifier-free: Each $\forall x.\, F$ is replaced by $\bigwedge_{o \in \mathcal{O}} F\{x \mapsto o\}$, and each $\exists x.\, F$ is replaced by $\bigvee_{o \in \mathcal{O}} F\{x \mapsto o\}$. If the components of the influence information are also restricted to pairs of (positive) ground fluent expressions, then Definition 8 can be directly adopted.

Obviously, however, this method does not exploit the extended expressiveness of causal relationships with variables. As a consequence, the resulting set may contain large subsets, each of which could be represented by a single causal relationship. We therefore develop a generalization of Definition 8 at least for a certain class of domain constraints and influence information. For these domain constraints, the automatic extraction of causal relationships with variables is intuitive and a straightforward extension of the ground case. To be precise, we consider domain constraints in which each occurrence of a quantifier is of the form $\forall\widetilde{x}.\,\ell[\widetilde{x}]$ or of the form $\exists\widetilde{x}.\,\ell[\widetilde{x}]$, where $\ell[\widetilde{x}]$ is a fluent expression with free variables $\widetilde{x}$. Furthermore, the components of the influence information $\mathcal{I}$ are interpreted as follows: If f_1, f_2 are fluent names, then $(f_1, f_2) \in \mathcal{I}$ indicates that a change of any instance $f_1(\widetilde{o}_1)$ may affect any instance $f_2(\widetilde{o}_2)$.

When computing the CNF of a set of domain constraints which is restricted in the above sense, sub-formulas $\forall\widetilde{x}.\,\ell[\widetilde{x}]$ and $\exists\widetilde{x}.\,\ell[\widetilde{x}]$ are treated like ordinary literals. Then each conjunct in the resulting CNF is a disjunction consisting of ground literals and expressions of the form $\forall\widetilde{x}.\,\ell[\widetilde{x}]$ and $\exists\widetilde{x}.\,\ell[\widetilde{x}]$. On this basis, an adequate set of causal relationships is obtained as follows:

Definition 24 Let $\mathcal{D}$ be a set of domain constraints. An influence information $\mathcal{I}$ determines a set of causal relationships $\mathcal{R}$ following this procedure:

1. Let $\mathcal{R} := \{\}$.
2. Let $D_1 \wedge \ldots \wedge D_n$ be the CNF of $\bigwedge \mathcal{D}$. For each $D_i = F_1 \vee \ldots \vee F_{m_i}$ ($i = 1, \ldots, n$) do the following:
3. For each $j = 1, \ldots, m_i$ do the following:
4. For each $k = 1, \ldots, m_i$, $k \neq j$, let

$$\Phi \; := \bigwedge_{\substack{l\,=\,1,\,\ldots\,,\,m_i \\ l\,\neq\,j,\,l\,\neq\,k}} \overline{F_l}$$ [30]

   and add the following causal relationships to $\mathcal{R}$:

   (a) If $F_j = \ell_j$ then
      (a1) if $F_k = \ell_k$ then add $\overline{\ell_j}$ `causes` ℓ_k `if` Φ
      (a2) if $F_k = \forall\widetilde{x}_k.\,\ell_k[\widetilde{x}_k]$ then add $\overline{\ell_j}$ `causes` $\ell_k[\widetilde{x}_k]$ `if` Φ
      (a3) if $F_k = \exists\widetilde{x}_k.\,\ell_k[\widetilde{x}_k]$ then add $\overline{\ell_j}$ `causes` $\ell_k[\widetilde{x}_k]$ `if` $\forall\widetilde{x}_k.\,\overline{\ell_k[\widetilde{x}_k]} \wedge \Phi$
   (b) If $F_j = \forall\widetilde{x}_j.\,\ell_j[\widetilde{x}_j]$ then
      (b1) if $F_k = \ell_k$ then add $\overline{\ell_j[\widetilde{x}_j]}$ `causes` ℓ_k `if` Φ
      (b2) if $F_k = \forall\widetilde{x}_k.\,\ell_k[\widetilde{x}_k]$ then add $\overline{\ell_j[\widetilde{x}_j]}$ `causes` $\ell_k[\widetilde{x}_k]$ `if` Φ
      (b3) if $F_k = \exists\widetilde{x}_k.\,\ell_k[\widetilde{x}_k]$ then add $\overline{\ell_j[\widetilde{x}_j]}$ `causes` $\ell_k[\widetilde{x}_k]$ `if` $\forall\widetilde{x}_k.\,\overline{\ell_k[\widetilde{x}_k]} \wedge \Phi$
   (c) If $F_j = \exists\widetilde{x}_j.\,\ell_j[\widetilde{x}_j]$ then
      (c1) if $F_k = \ell_k$ then add $\overline{\ell_j[\widetilde{x}_j]}$ `causes` ℓ_k `if` $\forall\widetilde{x}_j.\,\overline{\ell_j[\widetilde{x}_j]} \wedge \Phi$
      (c2) if $F_k = \forall\widetilde{x}_k.\,\ell_k[\widetilde{x}_k]$ then add $\overline{\ell_j[\widetilde{x}_j]}$ `causes` $\ell_k[\widetilde{x}_k]$ `if` $\forall\widetilde{x}_j.\,\overline{\ell_j[\widetilde{x}_j]} \wedge \Phi$
      (c3) if $F_k = \exists\widetilde{x}_k.\,\ell_k[\widetilde{x}_k]$ then add $\overline{\ell_j[\widetilde{x}_j]}$ `causes` $\ell_k[\widetilde{x}_k]$ `if` $\forall\widetilde{x}_j.\,\overline{\ell_j[\widetilde{x}_j]} \wedge \forall\widetilde{x}_k.\,\overline{\ell_k[\widetilde{x}_k]} \wedge \Phi$

   In any of these cases, however, add the respective causal relationship only if $(|\ell_j|, |\ell_k|) \in \mathcal{I}$.

■

Example 1 (continued) Recall our domain constraint $D = \forall x.\, on(x) \equiv light$, and let $\mathcal{I} = \{(on, light)\}$. Applying Definition 24 yields the following causal relationships:

[30] If $F_l = \forall\tilde{x}.\,\ell[\tilde{x}]$ then $\overline{F_l}$ denotes the formula $\exists\tilde{x}.\,\overline{\ell[\tilde{x}]}$, and if $F_l = \exists\tilde{x}.\,\ell[\tilde{x}]$ then $\overline{F_l}$ denotes $\forall\tilde{x}.\,\overline{\ell[\tilde{x}]}$.

- The CNF of D is $(\exists x.\, \overline{on(x)} \vee light) \wedge (\forall x.\, on(x) \vee \overline{light})$.
- As regards the first disjunct, $D_1 = \exists x.\, \overline{on(x)} \vee light$, we obtain the following:
  - In case $j = 1, k = 2$ we have $(on, light) \in \mathcal{I}$, which yields

    $$on(x) \ \texttt{causes} \ light \ \texttt{if} \ \forall x.\, on(x)$$

    according to clause (c1).
  - In case $j = 2, k = 1$ we have $(light, on) \notin \mathcal{I}$.
- As regards the second disjunct, $D_2 = \forall x.\, on(x) \vee \overline{light}$, we obtain the following:
  - In case $j = 1, k = 2$ we have $(on, light) \in \mathcal{I}$, which yields

    $$\overline{on(x)} \ \texttt{causes} \ \overline{light} \ \texttt{if} \ \top$$

    according to clause (b1).
  - In case $j = 2, k = 1$ we have $(light, on) \notin \mathcal{I}$.

Altogether, we obtain exactly the two causal relationships granted in (41). ■

The encoding of a given set of causal relationships $\mathcal{R} = \{r_1[\tilde{x}_1] = \varepsilon_1 \ \texttt{causes} \ \varrho_1 \ \texttt{if} \ \Phi_1, \ldots, r_n[\tilde{x}_n] = \varepsilon_n \ \texttt{causes} \ \varrho_n \ \texttt{if} \ \Phi_n\}$ ($n \geq 0$) in the fluent calculus follows Definition 23. We define a predicate $Causes(s, e, s', e')$ which is intended to be true iff there is an instance of a causal relationship in $\mathcal{R}$ which is applicable to (S, E) and whose application yields (S', E')—where s, e, s', e' are term representations of S, E, S', E':

$$Causes(s, e, s', e') \ \equiv \ \bigvee_{i=1}^{n} \exists \tilde{x}_i \left\{ \begin{array}{c} Holds(\Phi_i \wedge \overline{\varrho_i}, s) \ \wedge \ \exists z\, (\, \overline{\varrho_i} \circ z = s \ \wedge \ s' = z \circ \varrho_i \,) \\ \wedge \\ \exists v.\ \varepsilon_i \circ v = e \\ \wedge \\ \left[\begin{array}{c} \forall w.\ \overline{\varrho_i} \circ w \neq e \ \wedge \ e' = e \circ \varrho_i \\ \vee \\ \exists w\, (\, \overline{\varrho_i} \circ w = e \ \wedge \ e' = w \circ \varrho_i \,) \end{array} \right] \end{array} \right\} \tag{42}$$

This definition needs explanation. The first row in the right hand side of the formula encodes the two conditions $\Phi_i \wedge \overline{\varrho_i}$ be true in S and $S' = (S \setminus \{\overline{\varrho_i}\}) \cup \{\varrho_i\}$. The second row encodes the condition $\varepsilon_i \in E$. Finally, to encode the condition $E' = (E \setminus \{\overline{\varrho_i}\}) \cup \{\varrho_i\}$, two cases have to be distinguished: If $\overline{\varrho_i} \notin E$, then we just have to add ϱ_i to the corresponding term e (third row). If, on the other hand, $\overline{\varrho_i} \in E$, then we have to additionally remove the sub-term $\overline{\varrho_i}$ from e (fourth row). The following proposition shows that this encoding is correct given the definition of $Holds$, (31), in conjunction with our encoding of fluent formulas (c.f. (35)):

Proposition 25 *Let $\mathcal{R}$ be a set of causal relationships. Furthermore, let S be a state, E a set of fluent literals, and s', e' two collections of fluent literals. Then*

$$EUNA, (31), (42) \ \models \ Causes(\tau_S, \tau_E, s', e') \tag{43}$$

iff there exist two sets of fluent literals S', E' such that $EUNA \models s' = \tau_{S'} \wedge e' = \tau_{E'}$ and $(S, E) \leadsto_{\mathcal{R}} (S', E')$, else $EUNA, (31), (42) \models \neg Causes(\tau_S, \tau_E, s', e')$.

Proof: From (42) it follows that (43) is true iff there is an instance $r_i[\widetilde{o}]$ of a causal relationship $r_i[\widetilde{x}_i] = \varepsilon_i$ `causes` ϱ_i `if` Φ_i in $\mathcal{R}$ such that the conjunction in the right hand side of (42) is entailed. This in turn holds iff

1. $\Phi_i[\widetilde{o}] \wedge \overline{\varrho_i[\widetilde{o}]}$ is true in S (according to Proposition 20);
2. $s' = \tau_{(S \setminus \{\overline{\varrho_i[\widetilde{o}]}\}) \cup \{\varrho_i[\widetilde{o}]\}}$ (according to clauses 2 and 3 of Proposition 17);
3. $\{\varepsilon[\widetilde{o}]\} \subseteq E$ (according to clause 1 of Proposition 17); and
4. (a) either $\{\overline{\varrho_i[\widetilde{o}]}\} \not\subseteq E$ and $e' = \tau_{E \cup \{\varrho_i[\widetilde{o}]\}}$ (according to clauses 1 and 3 of Proposition 17),
   (b) or else $\{\overline{\varrho_i[\widetilde{o}]}\} \subseteq E$ and $e' = \tau_{(E \setminus \{\overline{\varrho_i[\widetilde{o}]}\}) \cup \{\varrho_i[\widetilde{o}]\}}$ (according to clauses 1, 2, and 3 of Proposition 17).

Following Definition 23, these conditions are equivalent to $(S, E) \leadsto_{\{r_i[\widetilde{o}]\}} (S', E')$ for sets S' and E' such that $EUNA \models s' = \tau_{S'}$ and $EUNA \models e' = \tau_{E'}$.

■

According to Definition 23, a successor state is obtained from an intermediate state by repeatedly applying causal relationships until a state results that does not violate the domain constraints. In order to encode this by means of the fluent calculus, we define a predicate $\mathit{Ramify}(s, e, s')$. It is intended to be true iff the successive application of causal relationships to (S, E) eventually results in a pair whose first component, S', satisfies the domain constraints—where s, e, s' are term representations of S, E, S'. This essentially requires to construct the transitive closure of the Causes relation defined in (42). As this cannot be expressed in first-order logic, we use the standard way of encoding transitive closure using a second-order formula:

$$\mathit{Ramify}(s,e,s') \;\equiv\; \forall \Pi \left\{ \begin{array}{c} \forall s_1, e_1.\ \Pi(s_1,e_1,s_1,e_1) \\ \wedge \\ \left[\begin{array}{l} \forall s_1,e_1,s_2,e_2,s_3,e_3 \\ \quad (\, \Pi(s_1,e_1,s_2,e_2) \wedge \mathit{Causes}(s_2,e_2,s_3,e_3) \\ \qquad\qquad \supset \Pi(s_1,e_1,s_3,e_3)\,) \end{array} \right] \\ \supset \\ \exists e'.\ \Pi(s,e,s',e') \end{array} \right\} \wedge \mathit{Possible}(s') \tag{44}$$

That is, $\mathit{Ramify}(s, e, s')$ is true iff both there is some e' such that (s, e, s', e') is in the transitive closure of Causes, and s' satisfies the domain constraints.

Finally, adding the ramification process to formula (39) completes our encoding. An instance $\mathit{Successor}(s, a, s')$ shall be true iff s' represents a successor state of a and the state represented by s:

$$\mathit{Successor}(s,a,s') \;\equiv\; \exists c,e,z\,[\,\mathit{Action}(c,a,e) \wedge c \circ z = s \wedge \mathit{Ramify}(z \circ e, e, s')\,] \tag{45}$$

To summarize, let $\mathcal{FC}$ denote the union of $EUNA$ with the definitions of Holds (31), $\mathit{Possible}$ (36), Action (38), Causes (42), Ramify (44), and $\mathit{Successor}$ (45), based on given sets of domain constraints, action laws, and causal relationships. As the main result of the second part of this paper we prove that $\mathcal{FC}$ provides a correct encoding of our solution to the ramification problem.

Theorem 26 *Let $\mathcal{FC}$ be the encoding of a set of domain constraints, a set of action laws, and a set of causal relationships. Furthermore, let $\mathcal{A}$ be a set of action names, S, S' two states, $a \in \mathcal{A}$ of arity m, and $\widetilde{o}$ a sequence of entities of length m. Then*

$$\mathcal{FC} \models \mathit{Successor}(s, a(\widetilde{o}), s')$$

iff there is a successor state S' of S and $a(\widetilde{o})$ such that $EUNA \models s' = \tau_{S'}$, else

$$\mathcal{FC} \models \neg \mathit{Successor}(s, a(\widetilde{o}), s')$$

Proof: The result follows from Definition 23, Propositions 22 and 25, and the fact that (44) encodes the transitive closure of the *Causes* relation plus the requirement to satisfy the domain constraints (c.f. (36) in conjunction with Proposition 20).

■

7 Discussion

We have presented a method to accommodate indirect effects of actions which involves the notion of causality to distinguish intuitively conceivable from unmotivated changes. To this end, we have developed the concept of *causal relationships*, each of which connects two effects with the intended meaning that, under specific circumstances, the occurrence of the former causes the occurrence of the latter as indirect effect. Causal relationships are serially applied to the intermediate state resulting from computing the direct effects of an action in order to reach a state that satisfies all underlying domain constraints. Moreover, we have argued that causal relationships can be generated automatically given additional domain-specific knowledge—called *influence information*—of how fluents may generally affect each other.

We have illustrated the expressiveness of our approach regarding the problem of implicit qualifications vs. indirect effects (Section 3.3), fluents which resist any categorization (Section 5.1), domains involving expected non-minimal change (Section 5.2), and regarding domain constraints that require a sophisticated distinction between qualification and ramification (Section 5.3). These results form the basis for comparing the method presented in this paper with existing approaches to the ramification problem.

The necessity of additional information to prevent changes that are suggested syntactically by the mere domain constraints but contradict the intuition, was first observed in [15] in the context of the *possible worlds approach* [14]. There, indirect effects of actions are implicitly obtained by searching for successor states staying as close as possible to the original state while satisfying both the direct effects of the action under consideration and the domain constraints. Although the authors argued that this might yield unintended changes (such as a switch magically jumping its position in the circuit depicted in Figure 1), no solution was offered.

In [24], the first and most elementary categorization-based solution to this problem was formally developed by distinguishing between so-called *frame* and *non-frame* fluents.[31] Only the former are subject to the persistence assumption, and their truth-values completely determine the truth-values of the latter. Similar ideas have been used e.g. in [7], in (the second part of) [4], and in [22]. More sophisticated categorization methods do not simply restrict persistence to one

[31] Earlier, the author of [48] raised the idea of introducing some notion of preference as regards changes of specific fluents to changes of other fluents. Yet her discussion was only informal and took place in the context of an example.

category by allowing arbitrary changes in the other. Instead they exploit different categories to define a partial preference ordering among all possible changes (as in our Definition 14), e.g. [39]. In [40], a systematic framework based on this concept is introduced with the aim of assessing the range of applicability of different approaches that follow the same principles. However, we have already argued in Section 5.1 that even if it is possible to assign an appropriate category to each fluent in a particular domain if only the categorization is suitably fine-grained, the more complex a domain is the more difficult this task becomes as it requires a deep analysis of possible interactions. Besides, despite being the only suitable one, a particular categorization may appear very unnatural even in simple domains, as we have illustrated in the context of Example 3.

More recently, several approaches have been established that take into account specific notions of causality, as does our method, to tackle the problem of unintended changes. The monotonic, situation calculus-based formalism developed in [8] supports specifications of indirect effects by means of complete descriptions of how the truth-value of a particular fluent might be caused to change. As an example, recall the basic Electric Circuit domain, which [8] would encode as

$$\begin{aligned}
&\forall a,s\ [\ \mathit{Causes}(a,s,\mathit{light}) \ \equiv\ (\ \mathit{Causes}(a,s,sw_1) \wedge \mathit{Holds}(sw_2,s)\ ) \\
&\qquad\qquad\qquad\qquad\qquad\quad \vee\ (\ \mathit{Causes}(a,s,sw_2) \wedge \mathit{Holds}(sw_1,s)\ )\ ] \\
&\forall a,s\ [\ \mathit{Cancels}(a,s,\mathit{light}) \ \equiv\ \mathit{Cancels}(a,s,sw_1) \vee \mathit{Cancels}(a,s,sw_2)\ ]
\end{aligned} \tag{46}$$

where $\mathit{Causes}(a,s,f)$ should be read as "executing action a in situation s causes fluent f to become true," $\mathit{Cancels}(a,s,f)$ as "executing action a in situation s causes fluent f to become false," and $\mathit{Holds}(f,s)$ as "fluent f is true in situation s." Suppose we are given the specification of how sw_1 may become true (resp. false), viz.

$$\begin{aligned}
&\forall a,s\ [\ \mathit{Causes}(a,s,sw_1) \ \equiv\ a = \mathit{toggle}_1 \wedge \neg \mathit{Holds}(sw_1,s)\ ] \\
&\forall a,s\ [\ \mathit{Cancels}(a,s,sw_1) \ \equiv\ a = \mathit{toggle}_1 \wedge \mathit{Holds}(sw_1,s)\ ]
\end{aligned}$$

plus this general axiom of persistence:

$$\forall a,s,f\ [\ \mathit{Holds}(f,\mathit{do}(a,s)) \ \equiv\ \mathit{Causes}(a,s,f) \vee (\ \mathit{Holds}(f,s) \wedge \neg \mathit{Cancels}(a,s,f)\ )\ ] \tag{47}$$

where $\mathit{do}(a,s)$ denotes the situation obtained by executing action a in situation s. One then obtains, say, that $\neg \mathit{Holds}(sw_1,s_0) \wedge \mathit{Holds}(sw_2,s_0) \wedge \neg \mathit{Holds}(\mathit{light},s_0)$ implies $\mathit{Causes}(\mathit{toggle}_1,s_0,sw_1)$ and, hence, $\mathit{Causes}(\mathit{toggle}_1,s_0,\mathit{light})$. This in turn implies $\mathit{Holds}(\mathit{light},\mathit{do}(\mathit{toggle}_1,s_0))$, as intended. No effort has to be made to suppress an unwanted change of sw_2 since no causal relation similar to (46) exists that may support this. On the other hand, the use of if-and-only-if descriptions of causal dependencies, as in (46), is restricted to domains where these dependencies are acyclic, i.e., hierarchical. Otherwise, i.e., if fluents depend mutually, unmotivated changes cannot be precluded. To see why, consider the simplest possible cyclic specification, namely,

$$\forall a,s\ [\ \mathit{Causes}(a,s,f_1) \ \equiv\ \mathit{Causes}(a,s,f_2)\ ] \tag{48}$$

Suppose a is an action whose execution in s_0 does not influence f_1 nor f_2, then formula (48) in conjunction with the persistence axiom (47) is too weak to conclude that both f_1 and f_2 keep their truth-values. For neither $\neg \mathit{Causes}(a,s_0,f_1)$ nor $\neg \mathit{Causes}(a,s_0,f_2)$ is entailed. The two mechanically connected switches discussed in Section 3.1, for instance, constitute a simple example which falls into the category of mutual dependencies. A second limitation of [8] compared to our approach stems from the fact that the formula in the right hand side of a definition like (46) either refers to the original state (in case of $Holds(f,s)$) or to the finally resulting successor state (in case of $Causes(a,s,f)$ or $Cancels(a,s,f)$, respectively, in conjunction with the persistence axiom). Consequently, no intermediately occurring fact can possibly trigger an indirect effect; hence, this formalization does not allow for deriving the successor state where a flash of the light bulb is recorded by the light detector in our Example 4.

In [4] and [29], the notion of causality is introduced by defining so-called causal rules (c.f. Definition 9). By virtue of being directed deduction rules, they cannot be applied in a non-causal way (e.g., $sw_1 \wedge sw_2 \Rightarrow light$ has a different meaning than, say, $sw_1 \wedge \overline{light} \Rightarrow \overline{sw_2}$). Besides a far more simple formalization employed in [29] compared to [4], the latter does not allow for deriving implicit qualifications rather than ramifications from domain constraints (c.f. Section 3.3). The reason is that in the approach [4] one always strives for a successor state no matter how many changes are necessary to this end, while [29] additionally requires all changes be explicitly grounded on some causal rule. Apart from this, the two approaches appear closely related. For instance, we strongly presume their equivalence in case of deterministic actions and domain constraints which do not give rise to qualifications.[32] In particular, both methods are grounded on the policy of minimal change, which amounts to rejecting any potential successor state whose distance to the original state is strictly larger than the distance of another proper successor state. As argued in Section 5.2, however, this paradigm might not always capture the intuition insofar as successor states may exist which have non-minimal distance but are equally plausible. A second difference between these two approaches on the one hand and our concept of causal relationships on the other hand, has been elaborated in Section 5.3. Recall that there we have illustrated a lack of expressiveness of causal rules regarding sophisticated distinctions between qualifications and ramifications triggered by one and the same domain constraint. Finally, we want to stress that in the approaches of [4, 29] it is assumed that causal rules be given as part of a domain specification. This requires more design effort than necessary—as can be seen by our suggestion to generate causal relationships automatically by employing more general information on potential influences.

Similar remarks apply to a recently developed integration of causality into the situation calculus-based framework presented in [26], also with the aim of handling indirect effects [25]. There, first-order formulas resembling causal relationships are used to define dependencies between effects and their indirect consequences. These formulas are of the form

$$\forall s \; [\; \Phi(s) \wedge Caused(f_1, v_1, s) \wedge \ldots \wedge Caused(f_n, v_n, s) \;\supset\; Caused(f, v, s) \;] \tag{49}$$

where $Caused(f,v,s)$ should be read as "fluent f is caused to take truth-value v in situation s," and where $\Phi(s)$ describes properties of situation s. As an example, recall the basic Electric Circuit domain, whose encoding by means of the approach [25] would include

$$\forall s \; [\; Holds(sw_2, s) \wedge Caused(sw_1, true, s) \;\supset\; Caused(light, true, s) \;]$$

along with the action definition[33]

$$\forall s \; [\; \neg Holds(sw_1, s) \;\supset\; Caused(sw_1, true, do(toggle_1, s)) \;] \tag{50}$$

The general axiom of persistence employed in this context is

$$\forall a, s, f \; [\; \neg\exists v.\, Caused(f, v, do(a,s)) \;\supset\; (\; Holds(f, do(a,s)) \equiv Holds(f,s) \;) \;]$$

[32] Moreover, a third approach, [11, 12], which is based on a nonmonotonic theory of "conditional entailment," is similar to [4, 29] in using expressions which resemble causal rules. A thorough and formal comparison between these three frameworks, however, has not yet been performed.

[33] For the sake of simplicity we neglect the concept of action preconditions.

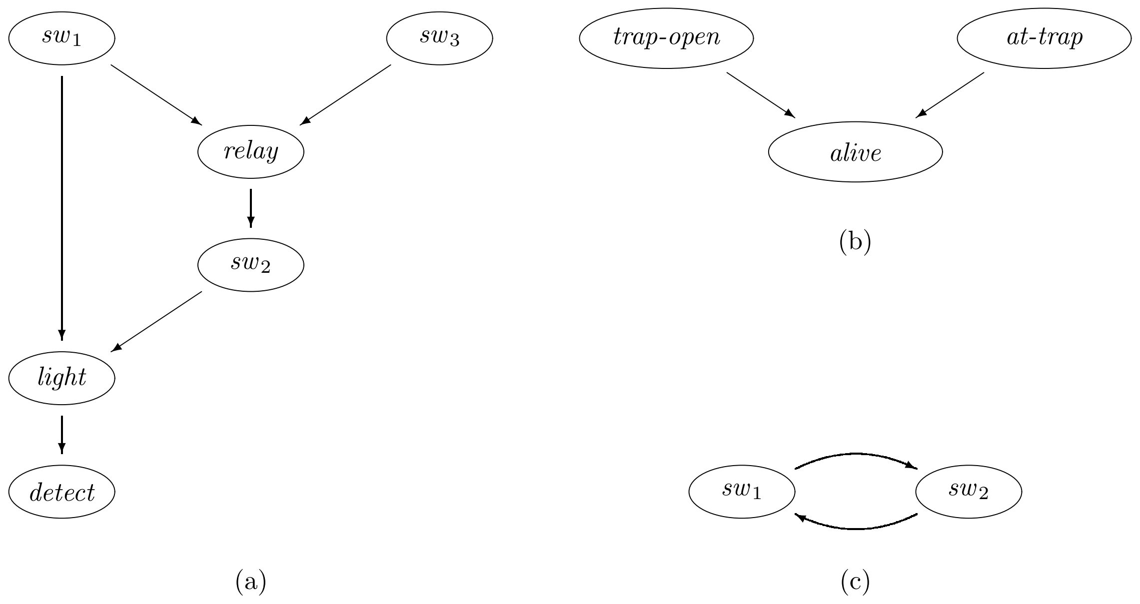

Figure 6: Causal networks representing the structural dependencies between fluents regarding (a) Example 4, (b) Example 5, and (c) the domain constraint $\overline{sw_1} \vee \overline{sw_2}$ (inducing a cycle), respectively.

This axiom is of course useless unless the extension of the predicate *Caused* is minimized given the direct effects of actions (like in (50)) and the laws of causality (each of which is of the form (49)). This is formally achieved by applying circumscription [31]. Hence, besides also leaving the effort of constructing the various causal relations to the designer, this method is grounded on the paradigm of minimal change as well. In fact, this work, too, appears closely related to [4, 29]. Notice, however, that the scheme (49) is expressive enough to allow for the sophisticated distinction between ramification and qualification discussed in Section 5.3. The reason for this is that the predicate *Caused* may occur in the left hand side of the implication (49). For instance, our Example 5 can be solved by employing $\forall s\, [\, \mathit{Holds}(\mathit{at\text{-}trap}, s) \wedge \mathit{Caused}(\mathit{trap\text{-}open}, \mathit{true}, s) \supset \mathit{Caused}(\mathit{alive}, \mathit{false}, s)\,]$ but not $\forall s\, [\, \mathit{Holds}(\mathit{trap\text{-}open}, s) \wedge \mathit{Caused}(\mathit{at\text{-}trap}, \mathit{true}, s) \supset \mathit{Caused}(\mathit{alive}, \mathit{false}, s)\,]$.

In order to account for expected non-minimal changes, more sophisticated means than minimization have to be developed for the approaches discussed above. This requires to carefully distinguish between conceivable changes, triggered by actually occurred (direct or indirect) effects, and unfounded changes. The approach developed in [27], which uses Dijkstra's semantics of programming languages to reason about actions, fails to address this challenge appropriately. In this approach, the ramification problem is tackled by allowing actions to (temporarily) release fluents from being subject to the assumption of persistence. But in case the domain constraints do not completely determine the new values of all in this way released fluents, unexpected effects may be produced (e.g., a turkey magically starts walking if being shot at with an unloaded gun).[34] Our causal relationships account for this problem since they are only applicable if the respective triggering effect either is among the direct effects or has previously been generated as indirect effect.

[34] We thank Vladimir Lifschitz for this observation.

An approach which is considerably different from all methods discussed so far yet still related, is based on networks representing probabilistic causal theories [34]. These networks describe, in the first place, static dependencies among their components. As argued in [35, 36], however, the truth-values of one or more nodes may be re-set dynamically and, then, the values of all depending nodes need to be adjusted according to standard (Bayesian) rules of probability. This can be regarded as generating indirect effects. If probability values are restricted to the binary 0/1-case, then a network whose nodes are fluent names resembles our concept of influence information. For instance, Figure 6(a) depicts a network suitable for Example 4. Although the relation between this approach and the other methods discussed in this section is far from being formally established today, let us point out some restrictions of causal networks compared to the concept of causal relationships. First, recall Figure 6(a). Since the resulting value of a node, after having fixed the direct effects, must not be computed until all new values of its predecessors have been determined, the proposition *detect* necessarily remains false regarding Example 4 since *light* stays false; hence, the non-minimal successor state where a light flash has been detected cannot be obtained. Second, recall Example 5. Since a change of *trap-open* might cause a change of *alive* depending on *at-trap*, the adequate network is the one depicted in Figure 6(b). This, however, does not allow to distinguish between the two situations where either *trap-open* becomes true with *at-trap* being true, or it happens to be the other way round. Hence, the sophisticated distinction whose necessity has been claimed in Section 5.3, is not supported by causal networks—similar to causal rules. Finally, networks representing causal theories are based on acyclic graphs, which means that simple examples like the mechanically connected switches (c.f. Section 3.1; relationships (3) and (4)) cannot be represented (c.f. Figure 6(c)). Aside from these rather specific observations, however, we would consider it most interesting to have a formal result regarding the range of applicability of approaches based on probabilistic networks. The calculus presented in [36], for instance, considers only actions without preconditions, and it merely refers to the *temporal projection* problem, which denotes the task to predict the effects of action sequences. This raises the question whether and how this approach can also be applied in other modes of reasoning like planning or *postdiction* (also called *chronicle completion* in [38]); the latter of which deals with finding explanations for observations made during the execution of action sequences.

This discussion leads us to the question of how to exploit the insights gained in this article. Our formalization in the first part of this paper has been embedded in a high-level, abstract description language and semantics for action scenarios, where we have concentrated on aspects of ramifications only and, to this end, employed a simple form of action specifications as regards direct effects. Three recent, similarly high-level action semantics focus on sophisticated ways to formalize this aspect, namely, the *Action Description Language* [13], the *Ego-World-Semantics* [38], and the framework presented in [46]. These approaches are considered prime candidates for being enhanced by the concept of causal relationships. In case of the Action Description Language, this should be based on the dialect developed in [3], which includes the notion of non-determinism, here needed if more than a single successor state exists. The resulting extension would constitute an alternate to the variant of the Action Description Language presented in [22], which handles ramifications on the basis of categorization and minimization, as does the extension of the Ego-World-Semantics presented in [40].

The main purpose of these three formal frameworks is to provide a uniform semantics for calculi designed to reason about actions and change. Given our formal proposal to handle indirect effects, existing calculi can be extended in such a way that their solution to the ramification problem is provably correct wrt. our semantics. As an example formalism, in the second part of this paper we have adapted the fluent calculus [19, 20]. Our Theorem 26 demonstrates that our extension of this approach is correct wrt. the formal semantics described in the first part. While the work reported in this article has been concentrated solely on the ramification problem, the fluent calculus—besides being closely related, in its basic form, to the *Linear Connection Method* [2] and reasoning about actions based on *Linear Logic* [16, 28]—has shown a wide range of applicability, e.g. regarding postdiction problems and non-deterministic actions [3], reasoning about counterfactual action sequences [44], or concurrent actions in conjunction with (locally) inconsistent specifications [3]. Thus, a main goal of future research consists in combining all these results, each of which focuses on a single ontological aspect, into a uniform and expressive calculus.

Acknowledgments. The author wants to thank Wolfgang Bibel, Jürgen Giesl, Christoph Herrmann, Jana Köhler, Stuart Russell, and two anonymous referees for helpful comments and suggestions on the paper. The paper has also benefited from discussions with Charles Elkan, G. Neelakantan Kartha, Vladimir Lifschitz, Norman McCain, Erik Sandewall, and Hudson Turner. Special thanks to Hans-H. Thielscher.

References

[1] A. B. Baker. Nonmonotonic reasoning in the framework of situation calculus. *Artificial Intelligence Journal*, 49:5–23, 1991.

[2] W. Bibel. A deductive solution for plan generation. *New Generation Computing*, 4:115–132, 1986.

[3] S.-E. Bornscheuer and M. Thielscher. Explicit and implicit indeterminism: Reasoning about uncertain and contradictory specifications of dynamic systems. *Journal of Logic Programming*, 1996. (To appear).

[4] G. Brewka and J. Hertzberg. How to do things with worlds: On formalizing actions and plans. *Journal of Logic and Computation*, 3(5):517–532, 1993.

[5] W. Büttner. Unification in datastructure multisets. *Journal of Automated Reasoning*, 2:75–88, 1986.

[6] M. O. Cordier and P. Siegel. A temporal revision model for reasoning about world change. In B. Nebel, C. Rich, and W. Swartout, editors, *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR)*, pages 732–739, Cambridge, MA, 1992. Morgan Kaufmann.

[7] A. del Val and Y. Shoham. Deriving properties of belief update from theories of action (II). In R. Bajcsy, editor, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 732–737, Chambéry, France, Aug. 1993. Morgan Kaufmann.

[8] C. Elkan. Reasoning about action in first-order logic. In *Proceedings of the Conference of the Canadian Society for Computational Studies of Intelligence*, Vancouver, Canada, May 1992.

[9] R. E. Fikes and N. J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence Journal*, 2:189–208, 1971.

[10] J. J. Finger. *Exploiting Constraints in Design Synthesis.* PhD thesis, Stanford University, CA, 1987.

[11] H. Geffner. Causal theories for nonmonotonic reasoning. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 524–530, Boston, MA, 1990.

[12] H. Geffner. *Default Reasoning: Causal and Conditional Theories.* MIT Press, 1992.

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

[14] M. L. Ginsberg and D. E. Smith. Reasoning about action I: A possible worlds approach. *Artificial Intelligence Journal*, 35:165–195, 1988.

[15] M. L. Ginsberg and D. E. Smith. Reasoning about action II: The qualification problem. *Artificial Intelligence Journal*, 35:311–342, 1988.

[16] J.-Y. Girard. Linear Logic. *Journal of Theoretical Computer Science*, 50(1):1–102, 1987.

[17] C. Green. Application of theorem proving to problem solving. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 219–239, Los Altos, CA, 1969. Morgan Kaufmann.

[18] S. Hanks and D. McDermott. Nonmonotonic logic and temporal projection. *Artificial Intelligence Journal*, 33(3):379–412, 1987.

[19] S. Hölldobler and J. Schneeberger. A new deductive approach to planning. *New Generation Computing*, 8:225–244, 1990.

[20] S. Hölldobler and M. Thielscher. Computing change and specificity with equational logic programs. *Annals of Mathematics and Artificial Intelligence*, 14(1):99–133, 1995.

[21] J. Jaffar, J.-L. Lassez, and M. J. Maher. A theory of complete logic programs with equality. *Journal of Logic Programming*, 1(3):211–223, 1984.

[22] G. N. Kartha and V. Lifschitz. Actions with indirect effects. In J. Doyle, E. Sandewall, and P. Torasso, editors, *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR)*, pages 341–350, Bonn, Germany, May 1994. Morgan Kaufmann.

[23] V. Lifschitz. On the semantics of STRIPS. In M. P. Georgeff and A. L. Lansky, editors, *Proceedings of the Workshop on Reasoning about Actions & Plans*. Morgan Kaufmann, 1986.

[24] V. Lifschitz. Frames in the space of situations. *Artificial Intelligence Journal*, 46:365–376, 1990.

[25] F. Lin. Embracing causality in specifying the indirect effects of actions. In C. S. Mellish, editor, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1985–1991, Montreal, Canada, Aug. 1995. Morgan Kaufmann.

[26] F. Lin and R. Reiter. State constraints revisited. *Journal of Logic and Computation*, 4(5):655–678, 1994.

[27] W. Łukaszewicz and E. Madalińska-Bugaj. Reasoning about action and change using Dijkstra's semantics for programming languages: Preliminary report. In C. S. Mellish, editor, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1950–1955, Montreal, Canada, Aug. 1995. Morgan Kaufmann.

[28] M. Masseron, C. Tollu, and J. Vauzielles. Generating plans in linear logic I. Actions as proofs. *Journal of Theoretical Computer Science*, 113:349–370, 1993.

[29] N. McCain and H. Turner. A causal theory of ramifications and qalifications. In C. S. Mellish, editor, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1978–1984, Montreal, Canada, Aug. 1995. Morgan Kaufmann.

[30] J. McCarthy. Programs with Common Sense. In *Proceedings of the Teddington Conference on the Mechanization of Thought Processes*, London, 1959. (Reprinted in: J. McCarthy, *Formalizing Common Sense*, Ablex, Norwood, New Jersey, 1990).

[31] J. McCarthy. Circumscription—a form of non-monotonic reasoning. *Artificial Intelligence Journal*, 13:27–39, 1980.

[32] J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4:463–502, 1969.

[33] J. Pearl. Embracing causality in default reasoning. *Artificial Intelligence Journal*, 35(2):259–271, 1988.

[34] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.

[35] J. Pearl. Graphical models, causality, and intervention. *Statistical Science*, 8(3):266–273, 1993.

[36] J. Pearl. A probabilistic calculus of actions. In R. L. de Mantaras and D. Poole, editors, *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 454–462, San Mateo, CA, 1994. Morgan Kaufmann.

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

[38] E. Sandewall. *Features and Fluents. The Representation of Knowledge about Dynamical Systems.* Oxford University Press, 1994.

[39] E. Sandewall. Reasoning about actions and change with ramification. In *Computer Science Today*, volume 1000 of *LNCS*. Springer, 1995.

[40] E. Sandewall. Systematic comparison of approaches to ramification using restricted minimzation of change. Technical Report LiTH-IDA-R-95-15, Department of Computer Science, Linköping University, Sweden, 1995.

[41] E. Sandewall. Comparative assessment of update methods using static domain constraints. In U. Sigmund and M. Thielscher, editors, *Reasoning About Actions and Planning in Complex Environments—Proceedings of the Workshop at FAPR'96*, pages I1–I14, Bonn, Germany, June 1996. Technical Report AIDA–96–11, FG Intellektik, TH Darmstadt, Germany.

[42] J. C. Shepherdson. SLDNF-resolution with equality. *Journal of Automated Reasoning*, 8:297–306, 1992.

[43] M. E. Stickel. A unification algorithm for associative commutative functions. *Journal of the ACM*, 28(3):207–274, 1981.

[44] M. Thielscher. An analysis of systematic approaches to reasoning about actions and change. In P. Jorrand and V. Sgurev, editors, *International Conference on Artificial Intelligence: Methodology, Systems, Applications (AIMSA)*, pages 195–204, Sofia, Bulgaria, Sept. 1994. World Scientific.

[45] M. Thielscher. Representing actions in equational logic programming. In P. V. Hentenryck, editor, *Proceedings of the International Conference on Logic Programming (ICLP)*, pages 207–224, Santa Margherita Ligure, Italy, June 1994. MIT Press.

[46] M. Thielscher. The logic of dynamic systems. In C. S. Mellish, editor, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1956–1962, Montreal, Canada, Aug. 1995. Morgan Kaufmann.

[47] M. Thielscher. On the completeness of SLDENF-resolution. *Journal of Automated Reasoning*, 1996. (To appear Fall '96).

[48] M. Winslett. Reasoning about action using a possible models approach. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 89–93, Saint Paul, MN, Aug. 1988.