Artificial Intelligence 248 (2017) 1–8

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Characterizing causal action theories and their implementations in answer set programming

Haodi Zhang [a,*], Fangzhen Lin [b]

[a] Department of Industrial Engineering and Logistics Management, Hong Kong University of Science and Technology, Clearwater Bay, Kowloon, Hong Kong
[b] Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clearwater Bay, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 21 August 2015
Received in revised form 11 October 2016
Accepted 27 February 2017
Available online 2 March 2017

Keywords:
Causal action theories
Action languages
Logic programming

ABSTRACT

We consider a simple language for writing causal action theories, and postulate several properties for the state transition models of these theories. We then consider some possible embeddings of these causal action theories in some other action formalisms, and their implementations in logic programs with answer set semantics. In particular, we propose to consider what we call permissible translations from these causal action theories to logic programs. We identify two sets of properties, and prove that for each set, there is only one permissible translation, under strong equivalence, that can satisfy all properties in the set. We also show that these two sets of conditions are minimal in that removing any condition from each of them will result in multiple permissible mappings. Furthermore, as it turns out, for one set, the unique permissible translation is essentially the same as Balduccini and Gelfond's translation from Gelfond and Lifschitz's action language $\mathcal{B}$ to logic programs. For the other, it is essentially the same as Lifschitz and Turner's translation from the action language $\mathcal{C}$ to logic programs. This work provides a new perspective on understanding, evaluating and comparing action languages by using sets of properties instead of examples. The results in this paper provide a characterization of two representative action languages $\mathcal{B}$ and $\mathcal{C}$ in terms of permissible mappings from our causal action theories to logic programs. It will be interesting to see if other action languages can be similarly characterized, and whether new action formalisms can be defined using different sets of properties.

1. Introduction

Formal reasoning about action has been a central topic in logic-based AI for a long time, and motivated much of the early work on nonmonotonic logics. The main difficulties have been the frame and the ramification problems. Current consensus in the community is that to solve the ramification problem, a notion of causality is needed. As a result, there has been much work on causal action theories (e.g. [1–11]), and a variety of languages and semantics have been proposed. These different approaches basically all agree when the set of causal rules is stratified, and in this case yields a complete action theory that can be represented, for example, by a set of successor state axioms. However, when there are cycles in the rules, it is

* Corresponding author.
E-mail address: zhanghd.ustc@gmail.com (H. Zhang).

http://dx.doi.org/10.1016/j.artint.2017.02.008
0004-3702/

not always clear how these rules are going to be represented according to these different approaches, and what the correct results are supposed to be. For instance, do we allow cyclic rules to produce indeterminate effects on actions?

This motivated us to do a computer experiment that would systematically enumerate all possible causal theories in a small language, and look at their desirable models. As it turned out, what counts as a desirable model depends on what properties we want to have about causal theories. This led us to consider various properties of transition models of causal theories. We then consider how these various properties fit into some existing action languages that allow static causal rules. To make the comparisons systematic, and also for computational reasons, we consider what we call *permissible* translations of our causal action theories to logic programs under the answer set semantics [12,13]. Specifically, given a set of properties on the transition models, we consider all possible permissible translations from our causal theories to logic programs so that the answer sets of these programs, when mapped back to the transition models of the causal theories, will satisfy all properties in the set. In this paper, we identify two such sets of properties. For both of them, our computer experiment shows that when there are at most three fluents in the language, the permissible translations are unique up to a notion of strong equivalence. The general case is proved by induction with the three fluent language as the base case.

These two sets of conditions are minimal in that if any condition is dropped from either of the sets, then there will be multiple permissible translations.

Furthermore, for one set of properties, the translation is essentially the same as Balduccini and Gelfond's translation from Gelfond and Lifschitz's action language $\mathcal{B}$ [8] to logic programs. For the other, the translation is essentially the same as Lifschitz and Turner's translation from the action language $\mathcal{C}$ [14] to logic programs. These results are significant in that they provide a new perspective on understanding, evaluating and comparing action languages by using sets of properties instead of examples. It is possible that other action languages can be similarly characterized, and new action languages defined using different sets of properties.

The rest of the paper is organized as follows. In Section 2 we formally introduce the syntax of the causal action theories that we use in this work. Next we list some reasonable properties that we expect the semantics of the causal action theories to satisfy in Section 3. Then in Section 4 we consider some straightforward mappings of our theories to some existing formalisms such as the action languages $\mathcal{B}$ and $\mathcal{C}$, and the situation calculus. In Section 5 we consider the problem of mapping causal action theories to logic programs under the answer set semantics, define the notion of permissible translations, and give our main results. And finally in Section 6 we conclude the paper. The detailed proofs of the two main theorems are given in the online Appendix.

2. Simple causal action theories

We assume a finite set $\mathcal{F}$ of propositional atoms called fluents. We also assume two distinguished symbols "$\top$" for tautology, and "$\bot$" for contradiction. A *fluent literal* is either f or $\neg f$ where $f \in \mathcal{F}$. So far in work on causal action theory, the focus is on the formalization of the effects of primitive actions, and how the causal rules are used in this formalization. The actions are assumed to be independent, in the sense that the effects of one action are independent of the effects of any other actions. So to make our formalism as simple and to the point as possible, we assume that there is just one unnamed action in our language, and when we talk about the effect of an action, we refer to the effect of this implicitly assumed, unnamed action.

Syntactically, a causal action theory is a pair (S, D), where S is a set of *static causal rules*, and D is a set of *dynamic causal rules*. Both static and dynamic causal rules are pairs of the form (l, G), where l is a fluent literal, and G a set of fluent literals. As a static causal rule, (l, G) means that in every situation, whenever all fluent literals in G hold, l is caused to be true. As a dynamic causal rule, it means that in every situation where all fluent literals in G hold, if the action is successfully executed, then l will be true in the new situation. Thus our dynamic causal rules are essentially direct action effect axioms. Notice that in the dynamic causal rules, the action argument is omitted here as we have assumed that there is just one action. In the following, we call l the head, and G the premise of the static or dynamic causal rule. We assume that G is consistent, i.e. it does not contain both f and $\neg f$, and does not contain $\bot$. We define a premise G to be a set of fluent literals because the order of fluent literals in the premise does not matter. However, we often write G as a conjunction of fluent literals in it. For instance, both $f_1 \wedge f_2$ and $f_2 \wedge f_1$ denote the same premise $\{f_1, f_2\}$. In particular, the empty set $\emptyset$ is denoted by $\top$.

Semantically, a causal action theory specifies a set of transitions. A transition is a pair (s, s'), where both s and s' are states, which are sets of fluents. A state s can be considered as a truth assignment such that a fluent f is true in s iff $f \in s$. For a formula φ, we write $s \models \varphi$ if φ is true in the truth assignment s. Similarly, for a set of fluent literals G, we write $s \models G$ if all fluent literals in G hold in the truth assignment. In particular, $s \models \emptyset$ for any state s. Intuitively, a transition (s, s') means that the action can be successfully executed in s to yield s'.

Definition 1. A state is a set of fluent atoms, and a transition is a pair of states. A semantic function δ is a mapping from causal action theories to sets of transitions.

Thus a semantic function δ gives a semantics to each causal action theory. We say that two causal action theories T_1 and T_2 are equivalent under δ if they have the same transitions: $\delta(T_1) = \delta(T_2)$. In the next section, we are going to discuss some properties about semantic functions.

3. Properties

We assume a fixed semantic function δ below. Thus when we say that (s, s') is a transition of T, we mean that $(s, s') \in \delta(T)$. We list below some interesting properties about δ. Our first property is that the states in a transition must satisfy every static causal rule.

Property 1 *(Static causal rules are state constraints). If (s, s') is a transition of a causal action theory (S, D), then for every static causal rule (l, G) in S, we have that*

$$s \models (l \vee \neg \bigwedge_{l_i \in G} l_i) \quad \text{and} \quad s' \models (l \vee \neg \bigwedge_{l_i \in G} l_i).$$

This property is commonly accepted by most action semantics. So we consider it should be satisfied as a basic property and build it in our permissible mapping.

When reasoning about, say whether a switch is open or closed, we can use fluent *closed* to mean that the switch is closed and represent the fact that the switch is open by $\neg closed$. Or we can do it the other way around, use *open* to mean that the switch is open and represent *closed* by $\neg open$. Our following property says that choosing which one to use as primitive is immaterial as long as one does this systematically.

Property 2 *(Fluent literals are interchangeable). For any causal action theory T, any fluent f, and any pair of states s_1 and s_2, (s_1, s_2) is a transition of T, iff (s_1^f, s_2^f) is a transition of T^f, where for any state s,*

$$s^f = \begin{cases} s \setminus \{f\} & \text{if } f \in s \\ s \cup \{f\} & \text{otherwise} \end{cases}$$

and for any causal action theory T, T^f is the causal action theory obtained from T by replacing every occurrence of f by $\neg f$.

Our next property says that if the premise of a static causal rule contains the negation of its head, then this rule is basically a constraint according to Property 1.

Property 3 *(Immediate negative loops can be eliminated). For any causal action theory (S, D), any $r = (l, \{\neg l\} \cup G) \in S$, and any states s and s' that satisfy all rules in S (i.e. Property 1 holds for S), we have that (s, s') is a transition of (S, D) iff (s, s') is a transition of $(S \setminus \{r\}, D)$.*

Our next property says that if there are no possible interactions between the static and dynamic causal rules, then only the dynamic causal rules can be applied. In other words, there are no ramifications.

Property 4 *(No ramification when static causal rules do not interact with dynamic causal rules). Let $T = (S, D)$ be a causal action theory. If for every $(l, G) \in D$, there is no $(l', G') \in S$ such that the fluent in l occurs in G', then (s, s') is a transition of T iff*

- $s' = (s \setminus s^-) \cup s^+$,
- $s^+ \cap s^- = \emptyset$, *and*
- *both s and s' satisfy the static causal rules in S (Property 1),*

where $s^- = \{f \mid (\neg f, G) \in D, s \models G\}$ and $s^+ = \{f \mid (f, G) \in D, s \models G\}$. In particular, if $s^+ \cap s^- \neq \emptyset$, then T has no transitions from s.

The next property is a weaker version of Property 4. Before giving it we first define the *dependency graph* of a causal action theory. Given a causal action theory $T = (S, D)$, its dependency graph is the directed graph such that

- its vertices are arbitrary fluent literals, and
- it has an edge from l_1 to l_2 iff S contains a static causal rule (l_1, G) such that $l_2 \in G$.

Property 4′ *(No ramification when static causal rules do not have a cycle and do not interact with dynamic causal rules). Let $T = (S, D)$ be a causal action theory whose dependency graph is acyclic. If for every $(l, G) \in D$, there is no $(l', G') \in S$ such that the fluent in l occurs in G', then (s, s') is a transition of T iff*

- $s' = (s \setminus s^-) \cup s^+$,
- $s^+ \cap s^- = \emptyset$, *and*
- *both s and s' satisfy the static causal rules in S (Property 1),*

where $s^- = \{f \mid (\neg f, G) \in D, s \models G\}$ *and* $s^+ = \{f \mid (f, G) \in D, s \models G\}$. *In particular, if* $s^+ \cap s^- \neq \emptyset$, *then* T *has no transitions from* s.

Property 4′ is weaker than Property 4 as the former applies only to "stratified" causal action theories.

The next property concerns the redundancy in premises. If both q and $\neg q$ will cause p, then p must be true regardless. Similarly, if both $q \wedge r$ and $\neg q \wedge r$ causes p, then we can conclude that q is immaterial and that r causes p.

Property 5 *(Premises of static causal rules can be combined). Let* $T = (S, D)$ *be a causal action theory. If both* (f, G_1) *and* (f, G_2) *are in* S, *and for some set* G *of fluent literals, formula*

$$\bigwedge_{l_k \in G} l_k$$

is logically equivalent to formula

$$\bigwedge_{l_i \in G_1} l_i \vee \bigwedge_{l_j \in G_2} l_j,$$

then the set of transitions under T *is the same as the set of transitions under* $T' = (S', D)$, *where* $S' = (S \setminus \{(f, G_1), (f, G_2)\}) \cup \{(f, G)\}$.

4. Embedding causal action theories in $\mathcal{B}$, $\mathcal{C}$, and others

Our causal action theories can be mapped to theories in action languages $\mathcal{B}$, $\mathcal{C}$, and others in a straightforward way. Notice that our language does not have terms for actions. Instead, there is an implicit action. Let's call this action A.

Given a causal action theory $T = (S, D)$, it can be mapped to the following action description $T_{\mathcal{B}}$ in language $\mathcal{B}$ as follows: if (l, G) is in S, then the static causal rule "l **if** G" is in $T_{\mathcal{B}}$, and if (l, G) is in D, then the action rule "A **causes** l **if** G" is in $T_{\mathcal{B}}$. This mapping induces a semantic function δ for our causal action theories: $(s, s') \in \delta(T)$ iff (s, A, s') is a transition of $T_{\mathcal{B}}$ according to $\mathcal{B}$. For this semantic function, it can be seen that Properties 1, 3 and 4 are satisfied (see Theorem 1 below). These three properties give a characterization of $\mathcal{B}$ semantics. Besides Properties 2 and 4′ are also satisfied, but Property 5 is not.

A causal action theory $T = (S, D)$ can also be translated to the following action description $T_{\mathcal{C}}$ in action language $\mathcal{C}$ as follows: if (l, G) is in S, then the static causal rule "**caused** l **if** $\bigwedge_{l_i \in G} l_i$" is in $T_{\mathcal{C}}$, and if (l, G) is in D, then the action rule "**caused** l **if** $\top$ **after** $A \wedge \bigwedge_{l_i \in G} l_i$" is in $T_{\mathcal{C}}$. Again this mapping induces a semantic function: (s, s') is a transition of T if (s, A, s) is a transition of $T_{\mathcal{C}}$ according to $\mathcal{C}$ semantics. For this semantic function, we can see that Properties 1, 2, 3 and 5 are satisfied, but not Property 4. For instance, consider the causal action theory $T = (\{(f_2, f_1), (f_1, f_2)\}, \emptyset)$. If $s = \emptyset$, and $s' = \{f_1, f_2\}$, then both s and s' satisfies all static causal rules, and according to $\mathcal{C}$ semantics, s' is a possible new state after the action A occurs in state s.

A causal action theory $T = (S, D)$ can also be mapped to Lin's causal theory T_L in the situation calculus as follows: if (f, G_0) is in S, let $G = \bigwedge_{l_i \in G_0} l_i$, then the following causal rule is in T_L: $\forall s. Holds(G, s) \supset Caused(f, true, s)$; if $(\neg f, G)$ is in S, then the following causal rule is in T_L: $\forall s. Holds(G, s) \supset Caused(f, false, s)$; if (f, G) is in D, then the following direct effect axiom is in T_L: $\forall s. (Poss(A, s) \supset (Holds(G, s) \supset Caused(f, true, do(A, s))))$; and if $(\neg f, G)$ is in D, then the following direct effect axiom is in T_L:

$\forall s. (Poss(A, s) \supset (Holds(G, s) \supset Caused(f, false, do(A, s))))$.

As in $\mathcal{C}$, under this mapping to Lin's causal theories, Property 4 is not satisfied.

One could do similar mappings for other formalisms. However, an obvious question is why the mappings have to be the way that we described above. For instance, could we map theories here to language $\mathcal{C}$ differently so that Properties 1–4 will be satisfied? If we do not have any restrictions on the mappings, then this is certainly possible as one can always compute the transitions first and then find a theory in $\mathcal{C}$ that have the same transitions. In this paper we consider this problem by choosing answer set logic programs as the target language. This is for two reasons. One is the availability of good ASP solvers for computing answer sets of a logic program. The other reason is that there are already well-studied mappings from languages like $\mathcal{B}$ and $\mathcal{C}$ to answer set logic programs. In the next section, we look at all possible compositional mappings of our causal action theories to logic programs that use a "standard" way of encoding inertia.

5. From causal action theories to answer set programming

There has been much work using logic programs for implementing and/or formalizing causal action theories (e.g. [4, 15–18]). In this paper we consider mapping our causal action theories to logic programs under answer set semantics.

Given a set $\mathcal{F}$ of fluents, and a causal action theory T in $\mathcal{F}$, we consider mapping T to a normal logic program (with constraints and classical negation) $\xi(T)$ in the language $\mathcal{L} = \mathcal{F} \cup \neg\mathcal{F} \cup \mathcal{F}' \cup \neg\mathcal{F}'$, where $\neg\mathcal{F} = \{\neg f \mid f \in \mathcal{F}\}$, $\mathcal{F}' = \{f' \mid f \in \mathcal{F}\}$, and similarly for $\neg\mathcal{F}'$. We assume here that for each $f \in \mathcal{F}$, f' is a new atom. The transitions of T and the answer sets

of $\xi(T)$ will then be related by identifying s with the fluents in $\mathcal{F}$, and s' with those in $\mathcal{F}'$: (s, s') is a transition of T iff there is an answer set A of $\xi(T)$ such that

$$s = A \cap \mathcal{F} \tag{1}$$

$$s' = \{f \mid f' \in A \cap \mathcal{F}'\}. \tag{2}$$

Again, if we do not have any restrictions on what programs we can take as $\xi(T)$, then we can have mappings that satisfy any consistent set of properties under the above correspondence between transitions and answer sets. However, from knowledge representation point of view, we would want the mapping to be compositional, in the sense that it can be composed from some mappings on the static and dynamic causal rules. As we mentioned, there has been much work on using logic programs for reasoning about action, and one of the key ideas has been to use the following rules to encode inertia:

$$f' \leftarrow f,\ \text{not}\ \neg f', \tag{3}$$

$$\neg f' \leftarrow \neg f,\ \text{not}\ f'. \tag{4}$$

Also we want the mapping to at least satisfy Property 1 so that both initial and successor states satisfy the static causal rules as constraints, and that a causal rule (l, G) should be translated to some logic program rules about l. Besides, we want the mapping to be uniform in the sense that the fluent names are not material. More precisely, if σ is a permutation of fluent names, then applying the mapping to T after the fluent names are changed according to σ is the same as applying the mapping to T first, and then changing the fluent names in the logic program according to σ.

To summarize, we study the following class of mappings.

Definition 2. A mapping ξ from causal action theories to logic programs is said to be *permissible* if it is

1. **(Compositional)** For each $T = (S, D)$,

$$\xi(T) = \bigcup_{r \in S} \xi^s(r) \cup \bigcup_{r \in D} \xi^d(r) \cup B, \tag{5}$$

where

- $\xi^s(r)$ is the translation on static causal rules, and for a rule (l, G), $\xi^s(l, G)$ is a logic program consisting of the following constraints:

$$\leftarrow \dot{l}, G, \tag{6}$$

$$\leftarrow \bar{l}', G', \tag{7}$$

and some rules of form

$$l' \leftarrow F_1',\ \text{not}\ F_2'$$

where $\bar{l}$ is the complement of l: $\bar{f} = \neg f$, and $\overline{\neg f} = f$, $W' = l_1', ..., l_n'$ if $W = \{l_1, \cdots, l_n\}$, and F_1' and F_2' are sets of atoms in $\mathcal{F}_G' \cup \neg\mathcal{F}_G'$, where $\mathcal{F}_G$ are all fluents that appear in G, and not $\{l_1', ..., l_k'\}$ is not $l_1', ...,$ not l_k';

- $\xi^d(r)$ is the translation on dynamic causal rules, and for a rule (l, G), $\xi^d(l, G)$ is a set of rules of the form

$$l' \leftarrow F,$$

where F is a set of fluent literals in $\mathcal{F} \cup \neg\mathcal{F}$;

- B, the base, is the following set of rules that are independent of the given theory T: for each $f \in \mathcal{F}$,

$$f \leftarrow \text{not}\ \neg f, \tag{8}$$

$$\neg f \leftarrow \text{not}\ f. \tag{9}$$

$$f' \leftarrow f,\ \text{not}\ \neg f', \tag{10}$$

$$\neg f' \leftarrow \neg f,\ \text{not}\ f'. \tag{11}$$

2. **(Uniform)** Both ξ^s and ξ^d are homomorphic under any permutation σ on the set $\mathcal{F}$ of fluents: $\xi^s(r^\sigma) = (\xi^s(r))^\sigma$, and $\xi^d(r^\sigma) = (\xi^d(r))^\sigma$, where r^σ is obtained from r by a fluent substitution according to σ, and for any logic program P, P^σ is the program obtained from P by a fluent substitution according to σ.

Notice that for the translation $\xi^d(r)$ of a dynamic causal rule, no "negation-as-failure" operator is used. This is because we have the choice rules (8) and (9) in the base that generate all possible complete initial states. Under these choice rules, and given that there is no other rule about f or $\neg f$ in a permissible mapping, fluent literals "f" and "not $\neg f$" are

interchangeable, and so are "$\neg f$" and "not f". Of course, we could allow the negation-as-failure operator in the translation of dynamic causal rules.

Notice that a mapping ξ from causal action theories to logic programs yields a semantic function δ under (1) and (2): $(s, s') \in \delta(T)$ iff there is an answer set A of $\xi(T)$ such that (1) and (2) hold. So in the following, when we say that a mapping satisfies a property, we mean that the semantic function yielded by the mapping satisfies this property.

Proposition 1. *If a mapping ξ is permissible, then ξ satisfies Property 1.*

Our main results are about the uniqueness of admissible mappings that satisfies certain sets of properties. Notice that with the presence of (6) and (7), we can easily prove the following proposition. Understandably, this uniqueness needs to be modulo on strong equivalence [19] as two mappings that always yield strongly equivalent logic programs are indistinguishable under the answer set semantics.

Our first result shows the correspondence between action language $\mathcal{B}$ and permissible mappings that satisfy Properties 1, 3 and 4. In the last section, we described a translation from our causal action theories to causal theories in action language $\mathcal{B}$. Given a causal action theory T, let $T_{\mathcal{B}}$ denote the causal theory in $\mathcal{B}$ according to this translation. We say that (s, s') is a transition of T under action language $\mathcal{B}$ semantics if (s, A, s') is a transition of $T_{\mathcal{B}}$ according to the semantics of action language $\mathcal{B}$.

Theorem 1. *Let $\xi_{\mathcal{B}}$ be the following mapping from causal action theories to logic programs: for each $T = (S, D)$,*

$$\xi_{\mathcal{B}}(T) = \bigcup_{r \in S} \xi^s_{\mathcal{B}}(r) \cup \bigcup_{r \in D} \xi^d_{\mathcal{B}}(r) \cup B, \tag{12}$$

where

- *B is the base,*
- *for each static causal rule (l, G), $\xi^s_{\mathcal{B}}(l, G)$ is the set of rules consisting of constraints (6) and (7) and the following rule*

$$l' \leftarrow G', \tag{13}$$

- *for each dynamic causal rule (l, G), $\xi^d_{\mathcal{B}}(l, G)$ is the singleton set consisting of the following rule*

$$l' \leftarrow G. \tag{14}$$

We have

1. *The mapping $\xi_{\mathcal{B}}$ is permissible and satisfies Properties 1, 3, 4. Furthermore, for any causal action theory T, (s, s') is a transition of T under $\xi_{\mathcal{B}}$ iff it is a transition of T under the $\mathcal{B}$ semantics.*
2. *If ξ is a permissible mapping that satisfies Properties 1, 3, 4, then ξ is strongly equivalent to $\xi_{\mathcal{B}}$ in the following sense: for any causal action theory, if (l, G) is a static causal rule, then $\xi^s(l, G)$ and $\xi^s_{\mathcal{B}}(l, G)$ are strongly equivalent under the base B as defined in (5); and if (l, G) is a dynamic causal rule, then $\xi^d(l, G)$ and $\xi^d_{\mathcal{B}}(l, G)$ are strongly equivalent under the base B as well.*
3. *The set of Properties 1, 3, and 4 is minimal: if any of them is deleted, then there is a permissible mapping ξ that satisfies the remaining properties and that for some static causal rule (l, G), $\xi^s(l, G)$ and $\xi^s_{\mathcal{B}}(l, G)$ are not strongly equivalent under B.*

We have a similar result for the action language $\mathcal{C}$ as well. In the following, given a causal action theory T, we say that (s, s') is a transition of T under $\mathcal{C}$ semantics if (s, A, s') is a transition of $T_{\mathcal{C}}$ according to $\mathcal{C}$, where $T_{\mathcal{C}}$ is the action description in $\mathcal{C}$ translated from T given in the last section.

Theorem 2. *Let $\xi_{\mathcal{C}}$ be the following mapping from causal action theories to logic programs: for each $T = (S, D)$,*

$$\xi(T) = \bigcup_{r \in S} \xi^s_{\mathcal{C}}(r) \cup \bigcup_{r \in D} \xi^d_{\mathcal{C}}(r) \cup B, \tag{15}$$

where

- *B is the base,*
- *for each static causal rule (l, G), $\xi^s_{\mathcal{C}}(l, G)$ is the set of rules consisting of constraints (6) and (7) and the following rule*

$$l' \leftarrow \text{not } \overline{G'}, \tag{16}$$

- *for each dynamic causal rule (l, G), $\xi^d_{\mathcal{C}}(l, G)$ is the singleton set consisting of the following rule*

$$l' \leftarrow G. \tag{17}$$

We have

1. *The mapping $\xi_{\mathcal{C}}$ is permissible and satisfies Properties 1, 2, 3, 4′ and 5. Furthermore, for any causal action theory T, (s, s') is a transition of T under $\xi_{\mathcal{C}}$ iff it is a transition of T under the $\mathcal{C}$ semantics.*
2. *If ξ is a permissible mapping that satisfies Properties 1, 2, 3, 4′ and 5, then ξ is strongly equivalent to $\xi_{\mathcal{C}}$ in the following sense: for any causal action theory if (l, G) is a static causal rule, then $\xi^s(l, G)$ and $\xi^s_{\mathcal{C}}(l, G)$ are strongly equivalent under the base B as defined in (5); and if (l, G) is a dynamic causal rule, then $\xi^d(l, G)$ and $\xi^d_{\mathcal{C}}(l, G)$ are strongly equivalent under the base B as well.*
3. *The set of Properties 1, 2, 3, 4′ and 5 is minimal: if any of them is deleted, then there is a permissible mapping ξ that satisfies the remaining properties and that for some static causal rule (l, G), $\xi^s(l, G)$ and $\xi^s_{\mathcal{C}}(l, G)$ are not strongly equivalent under B.*

The complete proofs of these two theorems are given in the online appendix. They are done inductively. The base case is the language with three fluents. For the base case, the results are verified by a computer program that for each representative dynamic or static causal rule, enumerates all of its possible permissible mappings, and for each possible mapping checks if the respective properties are satisfied. The inductive step is proved manually. As permissible mappings are compositional, we can decompose the proof for causal action theories into that for single causal rules. For each type of static/dynamic causal rules, to satisfy the properties, a permissible mapping has to map it to a logic program strongly equivalent with that in the theorems. We sketch the inductive step in the proof of Theorem 1 to illustrate. First, we prove that for any permissible mapping ξ and any dynamic rule r, if ξ satisfies Property 4′, $\xi^d(r)$ is strongly equivalent with $\xi^d_{\mathcal{B}}(r)$ in Theorem 1 (and also with $\xi^d_{\mathcal{C}}(r)$ in Theorem 2) under base B. Then we successively prove that for any permissible mapping ξ, for each r of the following types of static rules, if ξ satisfies the properties in Theorem 1, then $\xi^s(r)$ is strongly equivalent with $\xi^s_{\mathcal{B}}(r)$ under base B and the corresponding state constraints of form (6) and (7):

- static rules without any fluent loop
- static rules with negative fluent loop
- static rules with positive fluent loop

The definition of positive/negative fluent loop is given in the online appendix.

While the proof of the inductive step is tedious and non-trivial, the base case is conceptually more important. In fact, the theorems and the properties were "discovered" using our computer program. This is similar to computer-aided theorem discovery that we used earlier for some other problems [20–22].

These two theorems provide interesting insights into the relationships between the two action languages $\mathcal{B}$ and $\mathcal{C}$. First of all, they both satisfy Property 1, thus all states in their transition models must satisfy all static causal rules. They also satisfy Property 3, thus treat static causal rules with negative loops as constraints.

However, while $\mathcal{B}$ seems to treat causal theories with or without positive dependency loops uniformly, $\mathcal{C}$ assigns special meanings to static causal rules with positive loops. This can be seen from the fact that while $\mathcal{B}$ satisfies Property 4, $\mathcal{C}$ only satisfies a weaker version of that, Property 4′, which applies only to stratified theories. As an example, consider $T = (\{(f_1, f_2), (f_2, f_1)\}, \emptyset)$. The two causal rules cause a cycle between f_1 and f_2. Under $\mathcal{B}$'s semantics, for $(\emptyset, s)$ to be a transition, it must be the case that $s = \emptyset$. In other words, when both f_1 and f_2 are false initially, then they must both stay false in the successor state. However, under $\mathcal{C}$'s semantics, $(\emptyset, \{f_1, f_2\})$ is also a transition of T. In fact, static causal rules with immediate positive cycles such as (f, f) (or **caused** f **if** f in $\mathcal{C}$'s language) have a special role: they are used to express the assumption that the fluent f is a default fluent.

Another difference between $\mathcal{B}$ and $\mathcal{C}$ can be seen from Property 5. The language $\mathcal{C}$ satisfies this property, thus allows premises from multiple static causal rules to be merged together and simplified by logical equivalence. However, this is in general not allowed in $\mathcal{B}$. Consider the following causal action theories

$$T = (\{(f_1, f_2 \wedge f_3), (f_1, f_2 \wedge \neg f_3), (f_3, f_1 \wedge f_2)\}, \{(f_2, \top)\}).$$

The premises of the first two static causal rules can be merged and simplified according to $\mathcal{C}$: T is equivalent to T' below:

$$T' = (\{(f_1, f_2), (f_3, f_1 \wedge f_2)\}, \{(f_2, \top)\}).$$

But under $\mathcal{B}$'s semantics, these two theories are not equivalent. It can be seen that $(\emptyset, \{f_1, f_2, f_3\})$ is a transition of T' but not T under $\mathcal{B}$.

A corollary of these two theorems is that there is no permissible mapping that can satisfy all the properties.

Corollary 3. *There does not exist a permissible mapping ξ such that it satisfies Properties 1, 2, 3, 4 and 5.*

6. Concluding remarks

We have proposed a "minimalistic" language for causal action theories, and postulated some properties for them. We have considered possible embeddings of these causal action theories in some other action formalisms, and their implementations in logic programs with answer set semantics. In particular, we have proposed to consider what we call permissible

translations from causal action theories to logic programs. Our computer experiments show that when there are only three fluents in the language, the only permissible translation under strong equivalence that satisfies Properties 1, 3 and 4 is Balduccini and Gelfond's translation from the action descriptions in Gelfond and Lifschitz's $\mathcal{B}$ language to logic programs; and the only permissible translation under strong equivalence that satisfies Properties 1, 2, 3, 4′ and 5 is Lifschitz and Turner's translation from the action language $\mathcal{C}$ to logic programs. Furthermore, all these properties are necessary for the results to hold. Dropping any of them will result in multiple permissible mappings. These results suggest that the action languages $\mathcal{B}$ and $\mathcal{C}$ are characterized by the two sets of properties, respectively.

For future work, it will be interesting to see whether similar properties and alternative notions of "permissible" translations can be identified for other action formalisms, e.g. MAD and ALM. It is also interesting to investigate some other properties and see how combinations of these properties can yield different action formalisms.

Acknowledgements

This work was supported in part by HK RGC under GRF 616013.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.artint.2017.02.008.

References

[1] V. Lifschitz, Formal theories of action (preliminary report), in: J.P. McDermott (Ed.), Proceedings of IJCAI-87, Morgan Kaufmann, 1987, pp. 966–972.
[2] F. Lin, Embracing causality in specifying the indirect effects of actions, in: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95), 1995, pp. 1985–1993.
[3] N. McCain, H. Turner, A causal theory of ramifications and qualifications, in: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95), 1995, pp. 1978–1984.
[4] C. Baral, Reasoning about actions: nondeterministic effects, constraints, and qualification, in: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95), 1995, pp. 2017–2023.
[5] M. Thielscher, Ramification and causality, Artif. Intell. 89 (1997) 317–364.
[6] V. Lifschitz, On the logic of causal explanation (research note), Artif. Intell. 96 (2) (1997) 451–465, http://dx.doi.org/10.1016/S0004-3702(97)00057-X.
[7] H. Turner, A logic of universal causation, Artif. Intell. 113 (1–2) (1999) 87–123.
[8] M. Gelfond, V. Lifschitz, Action languages, Electron. Trans. Artif. Intell. 3 (1998) 195–210. URL http://www.cs.utexas.edu/users/ai-lab/?gel98.
[9] F. Lin, Compiling causal theories to successor state axioms and STRIPS-like systems, J. Artif. Intell. Res. 19 (2003) 279–314.
[10] A. Herzig, I. Varzinczak, Metatheory of actions: beyond consistency, Artif. Intell. 171 (16–17) (2007) 951–984, http://dx.doi.org/10.1016/j.artint.2007.04.013. URL http://www.sciencedirect.com/science/article/pii/S0004370207000781.
[11] J. Lee, V. Lifschitz, F. Yang, Action language $\mathcal{BC}$: preliminary report, in: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI-13), 2013, pp. 983–989. URL http://www.cs.utexas.edu/users/ai-lab/?lee13.
[12] M. Gelfond, V. Lifschitz, The stable model semantics for logic programming, in: Proceeding of Fifth International Conference and Symposium on Logic Programming (ICLP-88), 1988, pp. 1070–1080.
[13] M. Gelfond, V. Lifschitz, Classical negation in logic programs and disjunctive databases, New Gener. Comput. 9 (1991) 365–385.
[14] E. Giunchiglia, V. Lifschitz, An action language based on causal explanation: preliminary report, in: Proceedings of National Conference on Artificial Intelligence (AAAI), AAAI Press, 1998, pp. 623–630. URL http://www.cs.utexas.edu/users/ai-lab/?giu98.
[15] F. Lin, K. Wang, From causal theories to logic programs (sometimes), in: M. Gelfond, N. Leone, G. Pfeifer (Eds.), Logic Programming and Nonmonotonic Reasoning, Proceedings of the 5th International Conference, LPNMR'99, El Paso, Texas, USA, December 2–4, 1999, in: Lecture Notes in Computer Science, vol. 1730, Springer, 1999, pp. 117–131. URL http://dx.doi.org/10.1007/3-540-46767-X_9.
[16] N. McCain, Causality in Commonsense Reasoning about Actions, PhD thesis, University of Texas at Austin, Austin, TX, 1997.
[17] J. Lee, Reformulating action language $\mathcal{C}+$ in answer set programming, in: E. Erdem, J. Lee, Y. Lierler, D. Pearce (Eds.), Correct Reasoning, in: Lecture Notes in Computer Science, vol. 7265, Springer, Berlin, Heidelberg, 2012, pp. 405–421. URL http://dx.doi.org/10.1007/978-3-642-30743-0_28.
[18] P. Ferraris, J. Lee, Y. Lierler, V. Lifschitz, F. Yang, Representing first-order causal theories by logic programs, Theory Pract. Log. Program. 12 (2012) 383–412, http://dx.doi.org/10.1017/S1471068411000081. URL http://journals.cambridge.org/article_S1471068411000081.
[19] V. Lifschitz, D. Pearce, A. Valverde, Strongly equivalent logic programs, ACM Trans. Comput. Log. 2 (4) (2001) 526–541.
[20] F. Lin, Discovering state invariants, in: Proceedings of the Ninth International Conference on Principles of Knowledge Representation and Reasoning (KR2004), 2004, pp. 536–544.
[21] F. Lin, Y. Chen, Discovering classes of strongly equivalent logic programs, J. Artif. Intell. Res. 28 (2007) 431–451.
[22] P. Tang, F. Lin, Discovering theorems in game theory: two-person games with unique pure Nash equilibrium payoffs, Artif. Intell. 175 (14–15) (2011) 2010–2020, http://dx.doi.org/10.1016/j.artint.2011.07.001.