Reasoning About Causal Knowledge in Nondeterministic Domains

Shakil M. Khan[1] , **Yves Lespérance**[2] and **Maryam Rostamigiv**[1]
[1]Dept. of Computer Science, University of Regina, Regina, Saskatchewan, Canada
[2]Dept. of Electrical Engineering and Computer Science, York University, Toronto, Ontario, Canada
shakil.khan@uregina.ca, lesperan@eecs.yorku.ca, maryam.rostamigiv@uregina.ca

Abstract

Reasoning about causality and agent causal knowledge is critical for effective decision-making and planning in multi-agent contexts. Previous work in the area generally assumes that the domain is deterministic, but in fact many agents operate in nondeterministic domains where the outcome of their actions depends on unpredictable environment reactions. In this paper, we propose a situation calculus-based framework for reasoning about causal knowledge in nondeterministic domains. In such domains, the agent may not know the environment reactions to her actions and their outcomes, and may be uncertain about which actions caused a condition to come about. But she can perform sensing actions to acquire knowledge about the state and use it to gain knowledge about causes. Our formalization recognizes sensing actions as causes of both physical and epistemic effects. We also examine how regression can be used to reason about causal knowledge.

1 Introduction

Actual or token-level causality [Halpern, 2016] refers to the problem of determining the causes of observed effects from a perceived history of actions. Actual causality plays a crucial role in effective decision-making and planning in multi-agent systems. For instance, when an agent's plans fail to bring about her goals, it would be useful to understand why they failed, as this can aid the task of replanning by allowing the agent to generate better plans. However, for this to work, the agent must be able to reason about her own as well as other agents' causal knowledge. While extensive research has been conducted on actual causality [Pearl, 1998; Pearl, 2000; Halpern, 2000; Eiter and Lukasiewicz, 2002; Bochman, 2003; Hopkins, 2005; Hopkins and Pearl, 2007; Halpern, 2015; Batusov and Soutchanski, 2017; Batusov and Soutchanski, 2018; Bochman, 2018; Khan and Soutchanski, 2020; Beckers, 2021; Gladyshev *et al.*, 2023; Bochman, 2023], only a couple deal with causal knowledge [Chockler *et al.*, 2015; Khan and Lespérance, 2021].

A distinguishing feature of the real world is that change is often unpredictable. Unfortunately, the vast majority of the work in the area has focused on deterministic domains and only recently have researchers started investigating actual causation in nondeterministic domains [Rostamigiv *et al.*, 2024; Beckers, 2024; Khan *et al.*, 2025], where the outcome of the agent's actions depends on unpredictable environment reactions. Importantly, none have studied causal knowledge and its dynamics in such nondeterministic domains.

To deal with this, in this paper we propose a framework for reasoning about causal knowledge in nondeterministic domains. Our formalization is based on the recently proposed nondeterministic situation calculus (NDSC) [De Giacomo and Lespérance, 2021] and a model of actual causation in the situation calculus [Khan and Lespérance, 2021]. In our account, the agent may not know the environment reactions to her actions and their outcomes, and may be uncertain about which actions caused a condition to come about. But she can perform sensing actions to acquire knowledge about the state and use it to gain knowledge about causes. Our formalization supports reasoning about observed epistemic effects, recognizes sensing actions as causes of both physical and epistemic effects, and allows an agent to acquire causal knowledge by performing knowledge-producing actions. We also examine how regression can be used to reason about causal knowledge.

Our contribution in this paper is three-fold. First, we give a new successor-state axiom for the knowledge-accessibility relation and show how one can model knowledge update in the NDSC, and study some of its properties. As we will see, this is quite different from its deterministic situation calculus counterpart, as in such NDSC domains an agent might lose knowledge after executing an action. Secondly, we discuss how a recently proposed account of causation in the situation calculus [Khan and Lespérance, 2021] can be naturally combined with this account of knowledge in the NDSC to model causal knowledge in the NDSC. As mentioned, our formalization supports epistemic effects and recognizes sensing actions as causes. Finally, we extend knowledge regression in the situation calculus [Scherl and Levesque, 2003] and show how one can reason about causal knowledge in the NDSC.

2 Action and Nondeterminism

Situation Calculus (SC). Our base framework is the situation calculus [McCarthy and Hayes, 1969; Reiter, 2001], which we will not cover in detail here, except to remind the reader of the following: S_0 is used to denote the initial situation,

$do(a, s)$ for the successor situation to s resulting from performing the action a, and $do([a_1, \cdots, a_n], s)$ to represent the situation resulting from executing actions $a_1, \cdots, a_n$, starting with situation s. Relational/functional fluents take situation terms as their last argument. $Poss(a, s)$ states that action a is executable in situation s. $s \sqsubset s'$ represents that s' can be reached from situation s by executing some sequence of actions. $s \sqsubseteq s' \doteq s \sqsubset s' \vee s = s'$. $s < s'$ is an abbreviation of $s \sqsubset s' \wedge Executable(s')$, where $Executable(s)$ is defined as $\forall a', s'.\ do(a', s') \sqsubseteq s \supset Poss(a', s')$. $s \leq s'$ is an abbreviation of $s < s' \vee s = s'$.

In the SC, a dynamic domain is specified using a basic action theory (BAT) $\mathcal{D}$ that includes: (i) initial state axioms $\mathcal{D}_{S_0}$; (ii) action precondition axioms $\mathcal{D}_{ap}$ characterizing $Poss(a, s)$; (iii) successor-state axioms $\mathcal{D}_{ss}$ indicating precisely when the fluents change; (iv) unique-names axioms $\mathcal{D}_{una}$ for actions; and (v) foundational axioms Σ, describing the structure of situations [Levesque *et al.*, 1998].

A key feature of BATs is the existence of a sound and complete *regression mechanism* for answering queries about situations resulting from performing a sequence of actions [Pirri and Reiter, 1999; Reiter, 2001]. In a nutshell, the regression operator $\mathcal{R}^*$ reduces a formula ϕ about a particular future situation to an equivalent formula $\mathcal{R}^*[\phi]$ about the initial situation S_0. A formula ϕ is regressable if and only if (i) all situation terms in it are of the form $do([a_1, \ldots, a_n], S_0)$, (ii) in every atom of the form $Poss(a, \sigma)$, the action function is specified, i.e., a is of the form $A(t_1, \ldots, t_n)$, (iii) it does not quantify over situations, and (iv) it does not contain $\sqsubset$ or equality over situation terms. Thus in essence, a formula is regressable if it does not contain situation variables. In Section 4, we will define a one-step variant of $\mathcal{R}^*$, denoted using $\mathcal{R}_{ext}$. $\mathcal{R}^*$ can then be defined as the repeated application of $\mathcal{R}_{ext}$ until further applications leave the formula unchanged. Another key result about BATs is the relative satisfiability theorem [Pirri and Reiter, 1999; Reiter, 2001]: $\mathcal{D}$ is satisfiable if and only if $\mathcal{D}_{S_0} \cup \mathcal{D}_{una}$ is satisfiable (the latter being a purely first-order theory).

We will use uppercase Greek letters Φ, Ψ, etc. for situation-suppressed SC formulae, which are defined as follows:

$$\Phi ::= P(\vec{x}) \mid \neg\Phi \mid \Phi \wedge \Psi \mid \exists x.\ \Phi,$$

where $\vec{x}$ and x are object terms. Also, we will use α and σ, possibly with decorations, to represent ground action and situation terms, respectively. Finally, we use uppercase Latin letters for ground terms, lowercase Latin letters for variables.

Nondeterministic Situation Calculus (NDSC). An important limitation of the standard SC is that atomic actions are deterministic. De Giacomo and Lespérance [2021] (DL21) proposed a simple extension of the framework to handle nondeterministic actions while preserving the solution to the frame problem. For any primitive action by the agent in a nondeterministic domain, there can be a number of different outcomes. (DL21) takes the outcome as being determined by the agent's action and the environment's reaction to this action. This is modeled by having every action type/function $A(\vec{x}, e)$ take an additional environment reaction parameter e, ranging over a new sort *Reaction* of environment reactions. The agent cannot control the environment reaction, so it performs the reaction-suppressed version of the action $A(\vec{x})$ and the environment then selects a reaction e to produce the complete action $A(\vec{x}, e)$. The reaction-suppressed version of the action $A(\vec{x})$ is called an *agent action* and the full version of the action $A(\vec{x}, e)$ is called a *system action*.

We represent nondeterministic domains using action theories called Nondeterministic Basic Action Theories (NDBATs), which can be seen as a special kind of BAT, where (1) every agent action takes an environment reaction parameter; (2) for each agent action we have an agent action precondition formula, $Poss_{ag}(a(\vec{x}), s) \doteq \phi_a^{agPoss}(\vec{x}, s)$; (3) for each agent action we have a reaction independence requirement, stating that the precondition for the agent action is independent of any environment reaction $\forall e.\ Poss(a(\vec{x}, e), s) \supset Poss_{ag}(a(\vec{x}), s)$; (4) for each agent action we also have a reaction existence requirement, stating that if the precondition of the agent action holds then there exists a reaction to it which makes the complete system action executable, i.e., the environment cannot prevent the agent from performing an action when its agent action precondition holds $Poss_{ag}(a(\vec{x}), s) \supset \exists e.\ Poss_{ag}(a(\vec{x}, e), s)$. The above requirements *must* be entailed by an NDBAT.

An NDBAT $\mathcal{D}$ is the union of the following disjoint sets: including (1) foundational axioms, (2) unique-names axioms for actions, (3) axioms describing the initial situation, (4) successor-state axioms (SSA) indicating how fluents change after system actions, and (5) system action precondition axioms, indicating when system actions can occur; while these axioms generally follow the form: $Poss(a(\vec{x}, e), s) \equiv \phi_a^{Poss}(\vec{x}, e, s)$, in practice, these axioms often take the form: $Poss(a(\vec{x}, e), s) \equiv Poss_{ag}(a(\vec{x}), s) \wedge \varphi^{Poss}(\vec{x}, e, s)$, where $Poss_{ag}(a(\vec{x}), s)$ denotes conditions necessary for the agent action $a(\vec{x})$ to occur and $\phi_a^{Poss}(\vec{x}, e, s)$ captures additional conditions influenced by the environment's response.

In the NDSC, executing an agent action in a situation may result in different situations and outcomes depending on the environment reaction. To capture this, (DL21) introduced the defined predicate $Do_{ag}(\vec{a}, s, s')$, meaning that the system may reach situation s' when the agent executes the sequence of agent actions $\vec{a}$ starting in situation s for some environment reactions. A condition Φ may possibly hold after some executions of a sequence of agent actions $\vec{a}$ starting in situation s, denoted by $PAfter(\vec{a}, \Phi, s)$, or it may certainly hold after all executions of $\vec{a}$ in s, denoted by $CAfter(\vec{a}, \Phi, s)$. Similarly, (DL21) define notions of $\vec{a}$ being possibly/certainly executable. See [De Giacomo and Lespérance, 2021] for details.

3 Knowledge in the SC and NDSC

We now extend the framework of NDSC to accommodate knowledge and sensing. For simplicity, in both deterministic SC and NDSC, we assume the existence of two categories of primitive actions: ordinary/non-knowledge-producing actions and binary sensing/knowledge producing actions.

Formalizing Knowledge in SC. Following [Moore, 1985; Scherl and Levesque, 2003], we model knowledge using a possible worlds account adapted to the SC. There can now be multiple initial situations. $Init(s)$ means that s is an initial

situation. The actual initial state is denoted by S_0. $K(s', s)$ is used to denote that in situation s, the agent thinks that she could be in situation s'. Using K, the knowledge of an agent is defined as:[1] $Know(\Phi, s) \doteq \forall s'.\ K(s', s) \supset \Phi[s']$, i.e. the agent knows Φ in s if Φ holds in all of her K-accessible situations in s. We also use the abbreviations $KWhether(\Phi, s) \doteq Know(\Phi, s) \vee Know(\neg\Phi, s)$, i.e., the agent knows whether Φ holds in s and $KRef(\theta, s) \doteq \exists t.\ Know(\theta = t, s)$, i.e., she knows who/what θ refers to. K is constrained to be reflexive and Euclidean (and thus transitive) in the initial situation to capture the fact that the agent's knowledge is true, and that she has positive and negative introspection.

The dynamics of knowledge is specified by a SSA for K that supports knowledge expansion as a result of sensing actions. The information provided by a binary sensing action is specified using the predicate $SF(a, s)$; e.g., we might have $SF(sense_{onTable}(b), s) \equiv onTable(b, s)$, i.e., the action $sense_{ontable}(b)$ will tell the agent whether the block b is on the table in the situation where it is performed. SF is specified to be uniformly true for non-sensing actions. [Scherl and Levesque, 2003] specifies the SSA for K as follows:[2]

$$K(s', do(A(\vec{x}), s)) \equiv \exists s''.\ s' = do(A(\vec{x}), s'') \wedge K(s'', s) \wedge Poss(A(\vec{x}), s'') \wedge [SF(A(\vec{x}), s) \equiv SF(A(\vec{x}), s'')].$$

Thus, after an action happens, the agent learns that it was executable and it has happened. Moreover, if the action is a binary sensing action, the agent acquires knowledge of the associated proposition. As shown in [Scherl and Levesque, 2003], the constraints on K then continue to hold after any sequence of actions since they are preserved by the SSA for K. Scherl and Levesque [Scherl and Levesque, 2003] also showed how one can define regression for knowledge-producing actions.

Thus to model knowledge, one uses a theory that is similar to before, but with modified foundational axioms to allow for multiple initial epistemic states. Also, action preconditions can now include knowledge preconditions and initial state axioms can now include axioms describing the epistemic states of the agents. Finally, the aforementioned axioms for K are included. See [Reiter, 2001] for details.

Accommodating Nondeterminism. In deterministic domains, an agent's knowledge evolves predictably based on her actions. In contrast, in the NDSC, the agent must reason not only about the actions she performed but also about how the environment might react to those actions, *reactions that she does not observe*. For simplicity here, we will assume that the (binary) sensing actions $A(\vec{x})$ are deterministic (unlike the ordinary actions), and thus are coupled with a constant success reaction Suc to produce the associated system actions $A(\vec{x}, Suc)$.

The key to modeling knowledge in such a setting is the SSA for the knowledge fluent K, which specifies how the agent's knowledge evolves after performing an action. We modify it as follows to account for the fact that the agent does not observe the environment reaction:

$$K(s', do(A(\vec{x}, e), s)) \equiv \exists s'', e'.\ s' = do(A(\vec{x}, e'), s'') \wedge K(s'', s) \wedge Poss(A(\vec{x}, e'), s'') \wedge [SF(A(\vec{x}, e'), s) \equiv SF(A(\vec{x}, e'), s'')].$$

Thus after $A(\vec{x}, e)$ happens, the agent learns that for some environment reaction e', $A(\vec{x}, e')$, which was executable, has happened, and just like the deterministic case, if the action is a binary sensing action, the agent acquires knowledge of the associated proposition. This thus models that if there are more than one possible reactions for $A(\vec{x})$, the agent will not know the actual environment reaction.

[Scherl and Levesque, 2003] proved some properties to show that their axiomatization ensures that knowledge changes as appropriate. Let us examine which of these properties still hold unchanged for our axiomatization of knowledge in the NDSC, and which hold in a modified form. First, we still have that knowledge-producing actions do not change the state of the world, and the only fluent whose truth value is altered by a knowledge-producing action is K:

Theorem 1 (Knowledge-Producing Effects - Theorem 1 in [Scherl and Levesque, 2003])**.** *For all situations s, all fluents P (other than K) and knowledge-producing system action terms α, if $P(s)$ then $P(do(\alpha, s))$.*

This follows from the requirement that knowledge-producing actions do not affect fluents other that K.

Secondly, we still have that agents know the consequences of knowledge acquired through knowledge-producing actions:

Theorem 2 (Knowledge Incorporation - Theorem 3 in [Scherl and Levesque, 2003])**.** *For a knowledge-producing system action α, a fluent or the negation of a fluent F, a fluent or the negation of a fluent P, and a situation s, if the axiomatization entails that $Know(F \equiv SF(\alpha), s)$, $F(s)$, $Poss(\alpha, s)$, and $Know(F \supset P, s)$ hold, then $Know(P, do(\alpha, s))$ holds as well.*

We also have the following result showing that ignorance persist unless the agent knows that the action affects the fluent in question or produces knowledge about it:

Theorem 3 (Default Persistence of Ignorance)**.** *For a system action $a(\vec{x}, e)$ and a situation s, if $\neg Know(P, s)$ holds and the axiomatization entails that $\forall e', s.Poss(a(\vec{x}, e'), s) \supset (P(do(a(\vec{x}, e'), s)) \equiv P(s))$ and $\neg Know(\forall e'.[(Poss(a(\vec{x}, e')) \wedge SF(a(\vec{x}, e'))) \supset P], s)$, then $\neg Know(P, do(a(\vec{x}, e), s))$ holds as well.*

This is a modification of Theorem 2 in [Scherl and Levesque, 2003], where we take into account the fact that the agent does not observe the environment reaction.

[Scherl and Levesque, 2003] show (their Theorem 4) that in the deterministic SC, agents never forget, i.e., if an agent knows P in situation s, then P remains known in $do(a, s)$, provided that the action a does not make P false. This property no longer holds in non-deterministic domains. Instead, we have that:

Theorem 4 (Memory)**.** *For all fluents P and situations s, if $Know(P, s)$ holds, then $Know(P, do(a(\vec{x}, e), s))$ holds as long as the axiomatization entails that*

$$\forall e', s.\ Poss(a(\vec{x}, e'), s) \supset (P(do(a(\vec{x}, e'), s)) \equiv P(s)).$$

[1] Φ can contain a placeholder *now* in the place of the situation terms. Also, $\Phi[s]$ denotes the formula obtained by restoring the situation argument s into all fluents in Φ.

[2] For simplicity, in this paper we deal with binary sensing actions exclusively, and thus we use the SF fluent [Levesque, 1996]. We could have generalized this easily.

That is, the knowledge of P persists as long as P persists after the agent action for all possible reactions.

[Scherl and Levesque, 2003] also show (their Theorem 5) that in the deterministic SC, agents know the effects of ordinary actions. Instead, we have that this holds for certain effects of the action (i.e., effects that hold no matter what the environment reaction is):

Theorem 5 (Knowledge of Certain Effects of Actions)**.** *If $a(\vec{x}, e)$ is an ordinary action (not a knowledge-producing action) and if the axiomatization entails that $\forall e', s.\phi[s] \wedge Poss(a(\vec{x}, e'), s) \supset P(do(a(\vec{x}, e'), s))$, where ϕ is an arbitrary formula with situation terms suppressed and P is a fluent or its negation, then the following is also entailed:*

$$Know((\phi \wedge Poss_{ag}(a(\vec{x})), s) \supset Know(P, do(a(\vec{x}, e), s)).$$

Finally, we prove a property that shows that in the NDSC, the agent's knowledge in situation $do(\alpha(e), s)$ can be reduced to the agent's knowledge in s. This will serve as a basis of our proposed extended knowledge-regression operator.

Proposition 1. *Let $\mathcal{D}$ be an NDBAT, $\alpha(\vec{x})$ be a ground agent action term, φ be a situation-suppressed SC formula. Then we have the following. When $\alpha(\vec{x})$ is a regular (non-knowledge-producing) agent action:*

$$\mathcal{D} \models Know(\varphi, do(\alpha(\vec{x}, e'), s)) \equiv \\ Know(\forall e.\ Poss(\alpha(\vec{x}, e)) \supset \varphi[do(\alpha(\vec{x}, e))], s).$$

When $\alpha(\vec{x})$ is a knowledge producing agent action with sensed-fluent axiom $SF(\alpha(\vec{x}, e), s) \equiv \psi[s]$:

$$\mathcal{D} \models Know(\varphi, do(\alpha(\vec{x}, e), s)) \equiv \\ (\psi[s] \supset Know(\psi \wedge Poss(\alpha(\vec{x}, e)) \supset \varphi[do(\alpha(\vec{x}, e))], s)) \wedge \\ (\neg\psi[s] \supset Know(\neg\psi \wedge Poss(\alpha(\vec{x}, e)) \supset \varphi[do(\alpha(\vec{x}, e))], s)).$$

The proofs of Theorems 3, 4, and 5 are similar to that of the corresponding results in [Scherl and Levesque, 2003]. Proposition 1 can be proved easily by applying the definition of $Know$ and using the successor-state axiom for K.

Example. Consider a robot navigating between locations and communicating. The robot communicates only if the location is not risky and it hasn't become vulnerable. The robot moves between connected locations (agent action $move(i, j)$) and communicates from the current location (agent action $comm(i)$). Moving to a risky location may make the robot vulnerable, represented by the system action $move(i, j, e)$ where e can be either Vul (vulnerable) or $NotVul$ (not vulnerable). The communication action has a single successful outcome, represented by the system action $comm(i, e)$ where $e = Suc$. The precondition axioms for these are as follows:

$$(1)\ Poss_{ag}(move(i, j), s) \equiv At(i, s) \wedge Connected(i, j),$$
$$(2)\ Poss_{ag}(comm(i), s) \equiv \neg Vul(s) \wedge \neg Risky(i, s),$$
$$(1')\ Poss(move(i, j, e), s) \equiv Poss_{ag}(move(i, j), s) \\ \wedge (Risky(j, s) \supset (e = Vul \vee e = NotVul)) \\ \wedge (\neg Risky(j, s) \supset e = NotVul),$$
$$(2')\ Poss(comm(i, e), s) \equiv Poss_{ag}(comm(i), s) \wedge e = Suc.$$

The fluents in this example are $Vul(s)$, which denotes that the robot is vulnerable in situation s, $At(i, s)$, which states that the robot is at location i in s, and $Risky(i, s)$, which indicates that location i is risky in s. Certain locations are risky initially and they remain the only risky ones as actions are performed. Additionally, there is a non-fluent $Connected(i, j)$ indicating a path from location i to j.

The SSA for these fluents are as follows:

$$(3)\ At(j, do(a, s)) \equiv \exists i, e.\ a = move(i, j, e) \\ \vee (At(j, s) \wedge \forall j', e'.\ \neg(a = move(j, j', e'))),$$
$$(4)\ Vul(do(a, s)) \equiv \exists i, j.\ a = move(i, j, Vul) \vee Vul(s),$$
$$(5)\ Risky(i, do(a, s)) \equiv Risky(i, s).$$

We also have the following initial state axioms:

$$(6)\ \neg Vul(S_0),\ (7)\ At(I_0, S_0),\ (8)\ Risky(i, S_0) \equiv i = I_2.$$

There are four locations I_0 to I_3 in this domain, and the interconnections between these are given by the $Connected(i, j)$ predicate, where (i, j) are the pairs $(I_0, I_1), (I_1, I_0), (I_1, I_2), (I_2, I_1), (I_2, I_3)$, and (I_3, I_2). We will refer to this NDBAT as $\mathcal{D}_1$.

We assume the following initial knowledge:

$$(9)\ Know(At(I_0), S_0),\ (10)\ Know(\neg Vul, S_0),$$
$$(11)\ \neg KWhether(Risky(I_1), S_0),$$
$$(12)\ \neg KWhether(Risky(I_2), S_0),$$
$$(13)\ Know((Risky(I_1) \wedge \neg Risky(I_2)) \\ \vee (\neg Risky(I_1) \wedge Risky(I_2)), S_0).$$

Initially, there are two possible worlds related to S_0: S_0, where the robot is at location I_0, is not vulnerable, and I_2 only is risky, and S_0^1, where the robot is at location I_0, is not vulnerable, and I_1 only is risky. We also have sensing actions $sense_R(j, e)$ and $sense_V(e)$, and the following sensed-fluent axioms: $sense_R(j, e)$ senses whether location j is $Risky$:

$$(14)\ \forall s, j.\ SF(sense_R(j, Suc), s) \equiv Risky(j, s),$$

$sense_V(e)$ senses whether the robot has become vulnerable:

$$(15)\ \forall s.\ SF(sense_V(Suc), s) \equiv Vul(s).$$

Let $\mathcal{D}_1^K$ denote this axiomatization with knowledge. In this domain, we can show the following results:

Proposition 2.

$$\mathcal{D}_1^k \models \neg KWhether(Vul, do(move((I_0, I_1), e), S_0)),$$
$$\mathcal{D}_1^k \models Know(Vul \supset Risky(I_1) \wedge \neg Risky(I_2), \\ do([move((I_0, I_1), e), sense_V(Suc)], S_0)).\quad \square$$

4 Reasoning About Causal Knowledge

Causal Knowledge in the SC. Actual causation is the problem of identifying the relevant actions within a given history of events (known as the scenario), those that were responsible for causing an observed effect. When the effect is assumed to be initially false, this problem is referred to as *achievement causation*. Building on Batusov and Soutchanski's [2018] definition of actual causes, Khan and Lespérance [2021] (KL21, henceforth) recently formalized causal knowledge in the (deterministic) SC. Both proposals constrain scenarios to be linear sequences of actions. (KL21)'s account can handle causes of epistemic effects, allowing an agent to incorporate knowledge-preconditions of actions and reason

about the causes of newly acquired knowledge. They showed that an agent may be uncertain about a cause, but might know it after performing some knowledge-producing actions.

To formalize reasoning about causal knowledge, (KL21) introduced the concept of an *epistemic dynamic formula*.

Definition 1. *Let $\vec{x}$, θ_a, and $\vec{y}$ range over object terms, system actions, and object/system action variables, respectively. The class of situation-suppressed epistemic dynamic formulae φ is defined by: $\varphi ::= P(\vec{x}) \mid \mathit{Poss}(\theta_a) \mid \mathit{After}(\theta_a, \varphi) \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \exists\vec{y}.\, \varphi \mid \mathit{Know}(\varphi)$.*

Given an epistemic dynamic formula φ, actual causes of φ are defined relative to a *causal setting* $\langle \mathcal{D}, s, \varphi \rangle$, where $\mathcal{D}$ is a BAT and s is the scenario. Since we are dealing with achievement causality exclusively, it is also assumed that $\mathcal{D} \models \mathit{Executable}(s) \wedge \neg\varphi[S_0] \wedge \varphi[s]$. Here $\varphi[s]$ denotes the formula obtained from φ by restoring the appropriate situation argument into all fluents in φ. Formally:

Definition 2.

$$\varphi[s] \doteq \begin{cases} P(\vec{x}, s) & \text{if } \varphi \text{ is } P(\vec{x}) \\ \mathit{Poss}(\theta_a, s) & \text{if } \varphi \text{ is } \mathit{Poss}(\theta_a) \\ \psi'[do(\theta_a, s)] & \text{if } \varphi \text{ is } \mathit{After}(\theta_a, \psi') \\ \neg(\varphi'[s]) & \text{if } \varphi \text{ is } (\neg\varphi') \\ \varphi_1[s] \wedge \varphi_2[s] & \text{if } \psi \text{ is } (\varphi_1 \wedge \varphi_2) \\ \exists\vec{y}.\, (\varphi'[s]) & \text{if } \varphi \text{ is } (\exists\vec{y}.\, \varphi') \\ \forall s'.\, K(s', s) \supset (\varphi'[s']) & \text{if } \varphi \text{ is } \mathit{Know}(\varphi') \end{cases}$$

Since all changes in the SC are result of actions, the potential causes of an effect φ are identified with the set of action terms occurring in a situation s. However, since s may contain multiple instances of the same action, it is necessary to uniquely identify where these actions were performed. To address this, (KL21) required that each situation be associated with an integer timestamp, formalized using the following axioms:
$time(S_0) = 0$,
$\forall a, s, ts.time(do(a, s)) = ts \equiv time(s) = ts - 1$.
Causes are then a set of action-timestamp pairs.

Following [Batusov and Soutchanski, 2018], (KL21) define both primary and indirect notions of causes. Given an effect φ and scenario s, if some action of the action sequence in s triggers the formula φ to change from false to true relative to $\mathcal{D}$, and if there are no actions in s after it that change φ back to false, then this action is an actual cause of achieving φ in s. Such causes are referred to as *primary* causes:

Definition 3 (Primary Cause (KL21))**.**

$$\begin{aligned} &\mathit{CausesDirectly}(a, ts, \varphi, s) \doteq \\ &\quad \exists s_a.\ time(s_a) = ts \wedge (S_0 < do(a, s_a) \leq s) \wedge \\ &\quad \neg\varphi[s_a] \wedge \forall s'.(do(a, s_a) \leq s' \leq s \supset \varphi[s']). \end{aligned}$$

That is, a executed at timestamp ts is the *primary cause* of effect φ in situation s iff a was executed in a situation with timestamp ts in scenario s, a caused φ to change to true, and no subsequent actions on the way to s falsified φ.

Now, note that a (primary) cause a might have been non-executable initially. Also, a might have only brought about the effect conditionally and this context condition might have been false initially. Thus earlier actions in the trace that contributed to the preconditions and the context conditions of a cause must be considered as causes as well. The following definition captures both primary and indirect causes:[3]

Definition 4 (Actual Cause (KL21))**.**

$$\begin{aligned} &\mathit{Causes}(a, ts, \varphi, s) \doteq \\ &\forall P.[\forall a, ts, s, \varphi.(\mathit{CausesDirectly}(a, ts, \varphi, s) \supset P(a, ts, \varphi, s)) \\ &\quad \wedge \forall a, ts, s, \varphi.(\exists a', ts', s'.(\mathit{CausesDirectly}(a', ts', \varphi, s) \\ &\qquad\qquad \wedge time(s') = ts' \wedge s' < s \\ &\qquad\qquad \wedge P(a, ts, [\mathit{Poss}(a') \wedge \mathit{After}(a', \varphi)], s') \\ &\qquad\quad \supset P(a, ts, \varphi, s)) \\ &\quad] \supset P(a, ts, \varphi, s). \end{aligned}$$

Thus, Causes is defined to be the least relation P such that if a executed at timestamp ts directly causes φ in scenario s then (a, ts, φ, s) is in P, and if a' executed at ts' is a direct cause of φ in s, the timestamp of s' is ts', $s' < s$, and $(a, ts, [\mathit{Poss}(a') \wedge \mathit{After}(a', \varphi)], s')$ is in P (i.e. a executed at ts is a direct or indirect cause of $[\mathit{Poss}(a') \wedge \mathit{After}(a', \varphi)]$ in s'), then (a, ts, φ, s) is in P. Here the effect $[\mathit{Poss}(a') \wedge \mathit{After}(a', \varphi)]$ is that a' be executable and φ hold after a'.

Example (cont'd). Consider the (system) action sequence σ_1 and effect φ_1, where $\sigma_1 = do([comm(I_0, Suc),\ move(I_0, I_1, NotVul),\ move(I_1, I_2, Vul),\ move(I_2, I_3, NotVul)], S_0)$ and $\varphi_1 = Vul$. This is depicted in the left branch of the right tree in Figure 1. We can show that:

Proposition 3 (Causes of φ_1 in σ_1)**.**

$$\begin{aligned} \mathcal{D}_1 \models\ & \neg\mathit{Causes}(comm(I_0, Suc), 0, \varphi_1, \sigma_1) \\ & \wedge \mathit{Causes}(move(I_0, I_1, NotVul), 1, \varphi_1, \sigma_1) \\ & \wedge \mathit{Causes}(move(I_1, I_2, Vul), 2, \varphi_1, \sigma_1) \\ & \wedge \neg\mathit{Causes}(move(I_2, I_3, NotVul), 3, \varphi_1, \sigma_1). \end{aligned}$$

Thus, e.g., $move(I_1, I_2, Vul)$ executed at 2 is a cause since it directly caused the effect. Moreover, $move(I_0, I_1, NotVul)$ executed at 1 can be shown to be an indirect cause of φ_1. This is because by axioms (1) and (1′) the primary cause of moving from location I_1 to I_2 i.e. $move(I_1, I_2, Vul)$ is only possible when the robot is at I_1, which in this scenario was brought about by $move(I_0, I_1, NotVul)$. □

With this definition of $\mathit{Causes}(a, t, \psi, s)$, (KL21) showed that one can use it just like any other formula in the context of $Know$. One can state that an agent knows in some situation s that a executed at time t is a cause of an effect ψ, i.e. $\mathit{Know}(\mathit{Causes}(a, t, \psi, now), s)$, which by definition of knowledge means that $\forall s'.\ K(s', s) \supset \mathit{Causes}(a, t, \psi, s')$, i.e. in all her epistemic alternatives s', a at t is a cause of ψ.

Recently, [Khan *et al.*, 2025] showed that Causes in $do(a, s)$ can be reduced to a formula that only mentions Causes in s:

Proposition 4 (Proposition 3 in [Khan *et al.*, 2025])**.**

$$\begin{aligned} &\mathcal{D} \models \mathit{Causes}(b, t, \varphi, do(a, s)) \equiv \\ &\quad (time(s) = t \wedge b = a \wedge \neg\varphi[s] \wedge \varphi[do(a, s)]) \vee \\ &\quad (time(s) > t \wedge \varphi[s] \wedge \varphi[do(a, s)] \wedge \mathit{Causes}(b, t, \varphi, s)) \vee \\ &\quad (time(s) > t \wedge \neg\varphi[s] \wedge \varphi[do(a, s)] \\ &\qquad\qquad \wedge \mathit{Causes}(b, t, \mathit{Poss}(a) \wedge \mathit{After}(a, \varphi), s)). \end{aligned}$$

[3] In this, we need to quantify over situation-suppressed epistemic dynamic formulae. Thus we must encode such formulae as terms and formalize their relationship to the associated SC formulae. This is tedious but can be done essentially along the lines of [De Giacomo *et al.*, 2000]. We assume that we have such an encoding and use formulae as terms directly.

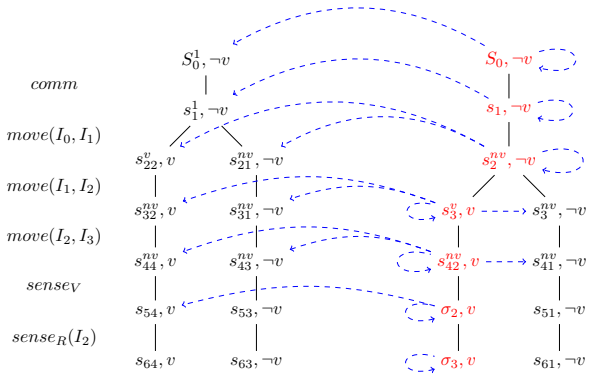

Figure 1: Executions of the agent actions associated with σ_1.

Causal Knowledge in the NDSC. Exploiting our formalization of knowledge for the NDSC from Sec. 3, we can reason about how the agent's causal knowledge changes when her knowledge is updated.

Example (cont'd). Consider the robot's causal knowledge in σ_1. Recall that the robot is unsure about the initial situation and cannot see the environment reactions to her actions; thus she must consider all possible executions of the agent action sequence associated with the scenario σ_1 starting in the initial situations that she considers possible when dealing with causal knowledge. Figure 1 shows these executions of this agent action sequence, i.e. $[comm(I_0),\ move(I_0, I_1),\ move(I_1, I_2),\ move(I_2, I_3)]$, starting in K-alternate initial situations S_0, where location I_2 is risky, and S_0^1, where I_1 is risky. Here, the superscripts v and nv represent environment choices Vul and $NotVul$, respectively, and $v/\neg v$ indicates whether the agent has become vulnerable or not. Thus, e.g., in this tree $s_1 = do(comm(I_0, Suc), S_0)$ and $s_2^{nv} = do(move(I_0, I_1, NotVul), s_1)$, etc. The actual situations and execution is shown in red, and the blue dashed-line shows the (partial) K-accessibility relation. Finally, $s_{42}^{nv} = \sigma_1$.

We can show that:

Proposition 5 (Causal Knowledge in σ_1)**.**

$$\begin{aligned}\mathcal{D}_1^K \models\ & Know(\neg\exists e.Causes(comm(I_0, e), 0, \varphi_1), \sigma_1)\\ & \wedge Know(\neg\varphi_1 \vee \exists e.Causes(move(I_0, I_1, e), 1, \varphi_1), \sigma_1)\\ & \wedge \neg KWhether(\exists e.Causes(move(I_1, I_2, e), 2, \varphi_1), \sigma_1)\\ & \wedge Know(\neg\exists e.Causes(move(I_2, I_3, e), 3, \varphi_1), \sigma_1).\end{aligned}$$

Thus, line 1 and 4 above says that the robot knows that $comm$ and the last $move$ actions are not causes of $\varphi_1 = Vul$ since these do not involve moving to a risky location; line 2, that the robot knows that either she has not become vulnerable, or for some environment reaction e, the first $move$ action (either directly, as in the case when $e = Vul$, or indirectly, when $e = NotVul$) caused φ_1 (note that it can be shown that the robot does not know if she has become vulnerable in σ_1); and line 3, that she does not know whether the second $move$ action is a cause, as even in the case where she became vulnerable, it might have been the first $move$ action that caused it (e.g. as in the case for s_{44}^{nv}). Again, since the robot cannot see the environment reactions and does not know which of S_0 and S_0^1 is the actual initial situation, as far as she is concerned, after executing this agent action sequence, she might be in any of the situations s_{41}^{nv}, s_{42}^{nv} (which is σ_1), s_{43}^{nv}, or s_{44}^{nv}; see Fig. 1.

Moreover, we can show that in addition to the above knowledge, the robot will learn that moving from I_0 to I_1 is indeed a cause of Vul after sensing Vul in σ_1, i.e. in $\sigma_2 = do(sense_V(Suc), \sigma_1)$:

Proposition 6 (Causal Knowledge in σ_2 (Partial))**.**

$$\mathcal{D}_1^K \models Know(\exists e.Causes(move(I_0, I_1, e), 1, \varphi_1), \sigma_2).$$

This is because after sensing for Vul, she will drop all situations from K where Vul does not hold, and will thus have s_{54} and σ_2 in K, in each of which the second move is a cause.

Finally, after sensing whether location I_2 is risky, she will also know the following in $\sigma_3 = do(sense_R(I_2, Suc), \sigma_2)$:

Proposition 7 (Causal Knowledge in σ_3 (Partial))**.**

$$\begin{aligned}\mathcal{D}_1^K \models\ & Know(Causes(move(I_0, I_1, NotVul), 1, \varphi_1), \sigma_3)\\ & \wedge Know(Causes(move(I_1, I_2, Vul), 2, \varphi_1), \sigma_3).\end{aligned}$$

To see this, note that since the robot initially knew that either I_0 or I_1 but not both is risky (by Axiom (13)), knows that (non)risky locations remain (non)risky (by Axiom (5)), and just learned that I_2 is risky, she will also know that I_1 is not risky. Consequently, σ_3 is the only K-accessible situation in σ_3. The result follows from this and the fact that $move(I_0, I_1, NotVul)$ and $move(I_1, I_2, Vul)$ are the only causes of φ_1 in σ_3 (similar to in Proposition 3). □

Extending Regression for Reasoning About Causal Knowledge in the NDSC. We start by extending the set of regressable formulae to include $Causes(b, t, \varphi, do(a, s))$ and $Know(\phi, do(a, s))$, with the same restrictions on their arguments as imposed in the original definition of regressable formula [Scherl and Levesque, 2003]. We also extend $\mathcal{R}$ to define regression of these additional constructs. We denote this one-step extended regression operator using $\mathcal{R}_{ext}$. As usual, we use $\mathcal{R}_{ext}^*$ to denote the repeated application of $\mathcal{R}_{ext}$ until further applications leave the argument formula unchanged.

Definition 5 (The Extended Regression Operator $\mathcal{R}_{ext}$)**.**
(1) When ϕ is a non-fluent atom, including equality atoms without functional fluents as arguments, or when ϕ is a fluent atom, whose situation argument is S_0, $\mathcal{R}_{ext}[\phi] = \phi$.
(2a) For a non-functional fluent F, whose successor-state axiom in $\mathcal{D}$ is $F(\vec{x}, do(a, s)) \equiv \Phi_F(\vec{x}, a, s)$, $\mathcal{R}_{ext}[F(\vec{t}, do(\alpha, \sigma))] = \Phi_F(\vec{t}, \alpha, \sigma)$.
(2b) For an equality literal with a functional fluent f, whose successor-state axiom is $f(\vec{x}, do(a, s)) = y \equiv \Phi_f(\vec{x}, y, a, s)$, $\mathcal{R}_{ext}[f(\vec{t}, do(\alpha, \sigma)) = t'] = \Phi_f(\vec{t}, t', \alpha, \sigma)$.
(2c) For a Poss literal with the action precondition axiom of the form $Poss(A(\vec{x}), s) \equiv \Pi_A(\vec{x}, s)$, $\mathcal{R}_{ext}[Poss(A(\vec{t}), \sigma)] \equiv \mathcal{R}_{ext}[\Pi_A(\vec{t}, \sigma)]$.
(3) For any non-atomic formulae, regression is defined inductively: $\mathcal{R}_{ext}[\neg\phi] = \neg\mathcal{R}_{ext}[\phi]$, $\mathcal{R}_{ext}[\phi_1 \wedge \phi_2] = \mathcal{R}_{ext}[\phi_1] \wedge \mathcal{R}_{ext}[\phi_2]$, $\mathcal{R}_{ext}[\exists v.\ \phi] = \exists v.\ \mathcal{R}_{ext}[\phi]$.
(4) If ϕ is an extended regressable formula of the form $Causes(b, t, \varphi, do(a, s))$, then:

$$\begin{aligned}&\mathcal{R}_{ext}[Causes(b, t, \varphi, do(a, s))] =\\ &(time(s) = t \wedge b = a \wedge \neg\varphi[s] \wedge \mathcal{R}_{ext}[\varphi[do(a, s)]]) \vee\\ &(time(s) > t \wedge \varphi[s] \wedge \mathcal{R}_{ext}[\varphi[do(a, s)]] \wedge Causes(b, t, \varphi, s)) \vee\\ &(time(s) > t \wedge \neg\varphi[s] \wedge \mathcal{R}_{ext}[\varphi[do(a, s)]] \wedge\\ &\qquad Causes(b, t, Poss(a) \wedge \mathcal{R}_{ext}[\varphi[do(a, s')]]^{-1}, s)).\end{aligned}$$

(5a) The regression of the Know operator when $\alpha(\vec{x})$ is not a knowledge producing action:

$$\mathcal{R}_{ext}[Know(\phi, do(\alpha(\vec{x}, e'), s))] = \\ Know(\forall e.Poss(\alpha(\vec{x}, e)) \supset \mathcal{R}_{ext}[\phi[do(\alpha(\vec{x}, e), s')]]^{-1}, s).$$

(5b) The regression of the Know operator when $\alpha(\vec{x})$ is a knowledge producing action with sensed-fluent axiom $SF(\alpha(\vec{x}, e), s) \equiv \psi[s]$:

$$\mathcal{R}_{ext}[Know(\phi, do(\alpha(\vec{x}, Suc), s))] = \\ (\psi[s] \wedge Know(\psi \wedge Poss(\alpha(\vec{x}, Suc)) \supset \\ \mathcal{R}_{ext}[\phi[do(\alpha(\vec{x}, Suc), s')]]^{-1}, s)) \wedge \\ (\neg\psi[s] \wedge Know(\neg\psi \wedge Poss(\alpha(\vec{x}, Suc)) \supset \\ \mathcal{R}_{ext}[\phi[do(\alpha(\vec{x}, Suc), s')]]^{-1}, s))$$

Cases 4 and 5 can be justified directly using Propositions 4 and 1 above. Intuitively, these reduce causes in scenario $do(a, s)$ to that in scenario s via reasoning by cases and using the definition of causes and regression;[4] and reduce knowledge in $do(\alpha(\vec{x}, e), s)$ to that in s.

With this definition in hand, we now present our key result.

Theorem 6. *If ϕ is an extended regressable formula and $\mathcal{D}$ is an NDBAT, then $\mathcal{D} \models \phi$ iff $\mathcal{D}_{S_0} \cup \mathcal{D}_{una} \models \mathcal{R}^*_{ext}[\phi]$.*

The proof is similar to that of the knowledge regression theorem in [Scherl and Levesque, 2003], but uses Prop. 1 and 4.

Example (cont'd). Let $\sigma^* = do([move(I_0, I_1, NotVul), move(I_1, I_2, Vul)], S_0)$. Note that:

$$\mathcal{D}_1^K \models Know(\neg\varphi_1 \vee \exists e.Causes(move(I_0, I_1, e), 0, \varphi_1)), \sigma^*).$$

We can show the following:

Proposition 8 (Example: Regression of Causal Knowledge)**.**

$$\mathcal{R}_{ext}[Know(\neg\varphi_1 \vee \exists e.Causes(move(I_0, I_1, e), 0, \varphi_1)), \sigma^*)] = \\ Know(\forall e'.\ Poss(move(I_1, I_2, e')) \supset \psi^*, \sigma^{**}), \\ \text{where, } \sigma^{**} = do(move(I_0, I_1, NotVul), S_0) \\ \text{and, } \psi^* = \neg(e' = Vul \vee Vul) \vee \\ \exists e.[[time > 0 \wedge Vul \wedge Causes(move(I_0, I_1, e), 0, Vul)] \vee \\ [time > 0 \wedge \neg Vul \wedge e' = Vul \\ \wedge Causes(move(I_0, I_1, e), 0, \\ Poss(move(I_1, I_2, e')) \wedge (e' = Vul \vee Vul))].$$

Thus the robot's knowledge in σ^* that $\neg\varphi_1 \vee \exists e.Causes(move(I_0, I_1, e), 0, \varphi_1)$ can be reduced to her knowledge in the previous situation σ^{**} that for all environment reactions e' for which the agent action $move(I_1, I_2)$ is possible, ψ^* holds, where the latter means that either e' is $NotVul$ and she is not currently vulnerable; or she is vulnerable, and $move(I_0, I_1)$ (directly) caused Vul for some e; or she is not vulnerable, e' is Vul, and for some e, $move(I_0, I_1)$ caused $move(I_1, I_2, e')$ to be possible and make Vul hold afterwards.

Repeated applications of $\mathcal{R}_{ext}$ yields a formula containing knowledge about S_0 that can be checked against the initial state axioms and $\mathcal{D}_{una}$ in $\mathcal{D}_1^K$. □

[4] Note that the formulae inside the context of $Causes$ and $Know$ are situation-suppressed. On the other hand, the regression operator $\mathcal{R}_{ext}$ requires a situation argument. To deal with this, here we introduce a new situation variable s' and use the φ^{-1} operator from [Scherl and Levesque, 2003] that suppresses the situation argument of φ by removing the last (situation) argument of all the fluents in φ. Thus, e.g., $\mathcal{R}_{ext}[\varphi[do(\alpha(\vec{x}, e), s')]]^{-1}$ introduces the situation term $do(\alpha(\vec{x}, e), s')$ to the situation suppressed formula φ, performs the regression, and then suppresses the situation argument in the result.

5 Conclusion

Motivated by the utility of actual causality in multiagent systems and the nondeterministic nature of real world action and change, in this paper we studied reasoning about causal knowledge in the NDSC [De Giacomo and Lespérance, 2021]. We first adapted [Scherl and Levesque, 2003] to obtain a formalization of knowledge change and sensing in the NDSC. We then combined this with a previously proposed definition of causal knowledge in the deterministic SC [Khan and Lespérance, 2021] to obtain a formalization of causal knowledge in the NDSC. Finally, we also defined a regression operator in the NDSC to reason about causal knowledge.

Recently, Khan, Lespérance, and Rostamigiv [2025] studied actual causality in the non-epistemic NDSC. As well as system actions as causes, they considered agent actions as causes and showed that one can define two different notions of causes for the latter, "certainly causes" and "possibly causes". But unlike us they did not study knowledge update in the NDSC or investigate causal knowledge dynamics.

To the best of our knowledge, our formalization is the first to deal with causal knowledge in nondeterministic domains. The only other account of causal knowledge besides [Khan and Lespérance, 2021] that we are aware of is the one proposed by Chockler et al. [2015], who formalized causal knowledge while defining responsibility/blame in legal cases. In that framework, an agent's uncertainty of the causal setting is modeled using an "epistemic state", which is a pair (K, Pr), where K is a set of causal settings and Pr is a probability distribution over K. The proposal is based on structural equations [Simon, 1977], and thus has limited expressiveness [Glymour *et al.*, 2010; Hopkins, 2005; Hopkins and Pearl, 2007]. Also, unlike us, they take events to be deterministic. In contrast, our framework is more expressive (first-order) and incorporates a formal model of domain dynamics and knowledge change. This allows for an interesting interplay between causality and knowledge. In addition, we incorporate nondeterminism and formalize reasoning about causal knowledge.

In this work, we focused on knowledge and not belief. The relation between actual causes, nondeterminism, and causal knowledge becomes more intricate if we incorporate the latter. Dealing with this is future work. Finally, in the future we would like to investigate reasoning that involves concurrent actions by multiple agents [De Giacomo *et al.*, 2016] and define responsibility and blame using our formalization [Yazdanpanah *et al.*, 2019; Naumov and Tao, 2020; Parker *et al.*, 2023].

Acknowledgements

This work is partially supported by the National Science and Engineering Research Council of Canada, by the University of Regina, and by York University.

References

[Batusov and Soutchanski, 2017] Vitaliy Batusov and Mikhail Soutchanski. Situation calculus semantics for actual causality. In Andrew S. Gordon, Rob Miller, and György Turán, editors, *Proceedings of the Thirteenth International Symposium on Commonsense Reasoning, COMMONSENSE 2017, London, UK, November 6-8, 2017*, volume 2052 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2017.

[Batusov and Soutchanski, 2018] Vitaliy Batusov and Mikhail Soutchanski. Situation calculus semantics for actual causality. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1744–1752. AAAI Press, 2018.

[Beckers, 2021] Sander Beckers. The counterfactual NESS definition of causation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6210–6217. AAAI Press, 2021.

[Beckers, 2024] Sander Beckers. Nondeterministic causal models. *CoRR*, abs/2405.14001, 2024.

[Bochman, 2003] Alexander Bochman. A logic for causal reasoning. In Georg Gottlob and Toby Walsh, editors, *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*, pages 141–146. Morgan Kaufmann, 2003.

[Bochman, 2018] Alexander Bochman. Actual causality in a logical setting. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 1730–1736. ijcai.org, 2018.

[Bochman, 2023] Alexander Bochman. Default logic as a species of causal reasoning. In Pierre Marquis, Tran Cao Son, and Gabriele Kern-Isberner, editors, *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning, KR 2023, Rhodes, Greece, September 2-8, 2023*, pages 117–126, 2023.

[Chockler *et al.*, 2015] Hana Chockler, Norman E. Fenton, Jeroen Keppens, and David A. Lagnado. Causal analysis for attributing responsibility in legal cases. In Ted Sichelman and Katie Atkinson, editors, *Proceedings of the 15th International Conference on Artificial Intelligence and Law, ICAIL 2015, San Diego, CA, USA, June 8-12, 2015*, pages 33–42. ACM, 2015.

[De Giacomo and Lespérance, 2021] Giuseppe De Giacomo and Yves Lespérance. The nondeterministic situation calculus. In Meghyn Bienvenu, Gerhard Lakemeyer, and Esra Erdem, editors, *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021, Online event, November 3-12, 2021*, pages 216–226, 2021.

[De Giacomo *et al.*, 2000] Giuseppe De Giacomo, Yves Lespérance, and Hector J. Levesque. ConGolog, a concurrent programming language based on the situation calculus. *Artificial Intelligence*, 121(1-2):109–169, 2000.

[De Giacomo *et al.*, 2016] Giuseppe De Giacomo, Yves Lespérance, and Adrian R. Pearce. Situation calculus game structures and GDL. In *ECAI*, pages 408–416, 2016.

[Eiter and Lukasiewicz, 2002] Thomas Eiter and Thomas Lukasiewicz. Complexity results for structure-based causality. *Artificial Intelligence*, 142(1):53–89, 2002.

[Gladyshev *et al.*, 2023] Maksim Gladyshev, Natasha Alechina, Mehdi Dastani, Dragan Doder, and Brian Logan. Dynamic causality. In Kobi Gal, Ann Nowé, Grzegorz J. Nalepa, Roy Fairstein, and Roxana Radulescu, editors, *ECAI 2023 - 26th European Conference on Artificial Intelligence, September 30 - October 4, 2023, Kraków, Poland - Including 12th Conference on Prestigious Applications of Intelligent Systems (PAIS 2023)*, volume 372 of *Frontiers in Artificial Intelligence and Applications*, pages 867–874. IOS Press, 2023.

[Glymour *et al.*, 2010] Clark Glymour, David Danks, Bruce Glymour, Frederick Eberhardt, Joseph D. Ramsey, Richard Scheines, Peter Spirtes, Choh Man Teng, and Jiji Zhang. Actual causation: A stone soup essay. *Synthese*, 175(2):169–192, 2010.

[Halpern, 2000] Joseph Y. Halpern. Axiomatizing causal reasoning. *Journal of Artificial Intelligence Research*, 12:317–337, 2000.

[Halpern, 2015] Joseph Y. Halpern. A modification of the halpern-pearl definition of causality. In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 3022–3033. AAAI Press, 2015.

[Halpern, 2016] Joseph Y. Halpern. *Actual Causality*. MIT Press, 2016.

[Hopkins and Pearl, 2007] Mark Hopkins and Judea Pearl. Causality and counterfactuals in the situation calculus. *Journal of Logic and Computation*, 17(5):939–953, 2007.

[Hopkins, 2005] Mark Hopkins. *The Actual Cause: From Intuition to Automation*. PhD thesis, University of California Los Angeles, 2005.

[Khan and Lespérance, 2021] Shakil M. Khan and Yves Lespérance. Knowing why - on the dynamics of knowledge about actual causes in the situation calculus. In Frank Dignum, Alessio Lomuscio, Ulle Endriss, and Ann Nowé, editors, *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, pages 701–709. ACM, 2021.

[Khan and Soutchanski, 2020] Shakil M. Khan and Mikhail Soutchanski. Necessary and sufficient conditions for actual root causes. In Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang, editors, *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 800–808. IOS Press, 2020.

[Khan *et al.*, 2025] Shakil M. Khan, Yves Lespérance, and Maryam Rostamigiv. Reasoning about actual causes in nondeterministic domains. In *Proceedings of the Thirty-ninth AAAI Conference on Artificial Intelligence, (AAAI-25)*, 2025.

[Levesque *et al.*, 1998] Hector J. Levesque, Fiora Pirri, and Raymond Reiter. Foundations for the situation calculus. *Electronic Transactions on Artificial Intelligence (ETAI)*, 2:159–178, 1998.

[Levesque, 1996] Hector J. Levesque. What is planning in the presence of sensing? In William J. Clancey and Daniel S. Weld, editors, *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI 96, IAAI 96, Portland, Oregon, USA, August 4-8, 1996, Volume 2*, pages 1139–1146. AAAI Press / The MIT Press, 1996.

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

[Moore, 1985] Robert C. Moore. A formal theory of knowledge and action. In *Formal Theories of the Commonsense World*, pages 319–358. Ablex, 1985.

[Naumov and Tao, 2020] Pavel Naumov and Jia Tao. An epistemic logic of blameworthiness. *Artificial Intelligence*, 283:103269, 2020.

[Parker *et al.*, 2023] Timothy Parker, Umberto Grandi, and Emiliano Lorini. Anticipating responsibility in multiagent planning. In Kobi Gal, Ann Nowé, Grzegorz J. Nalepa, Roy Fairstein, and Roxana Radulescu, editors, *ECAI 2023 - 26th European Conference on Artificial Intelligence, September 30 - October 4, 2023, Kraków, Poland - Including 12th Conference on Prestigious Applications of Intelligent Systems (PAIS 2023)*, volume 372 of *Frontiers in Artificial Intelligence and Applications*, pages 1859–1866. IOS Press, 2023.

[Pearl, 1998] Judea Pearl. On the definition of actual cause. Technical Report R-259, University of California Los Angeles, 1998.

[Pearl, 2000] Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

[Pirri and Reiter, 1999] Fiora Pirri and Raymond Reiter. Some contributions to the metatheory of the situation calculus. *J. ACM*, 43(3):325–361, 1999.

[Reiter, 2001] Raymond Reiter. *Knowledge in Action. Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, Cambridge, MA, USA, 2001.

[Rostamigiv *et al.*, 2024] Maryam Rostamigiv, Shakil M. Khan, Yves Lespérance, and Mriana Yadkoo. A logic of actual cause for non-deterministic dynamic domains. In *Working Notes of the 21st European Conference on Multi-Agent Systems, August 26-28th, 2024, Dublin, Ireland*, 2024.

[Scherl and Levesque, 2003] Richard B. Scherl and Hector J. Levesque. Knowledge, action, and the frame problem. *Artificial Intelligence*, 144(1-2):1–39, 2003.

[Simon, 1977] Herbert A. Simon. Causal ordering and identifiability. *Models of Discovery. Boston Studies in the Philosophy of Science*, 54, 1977.

[Yazdanpanah *et al.*, 2019] Vahid Yazdanpanah, Mehdi Dastani, Wojciech Jamroga, Natasha Alechina, and Brian Logan. Strategic responsibility under imperfect information. In Edith Elkind, Manuela Veloso, Noa Agmon, and Matthew E. Taylor, editors, *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, pages 592–600. International Foundation for Autonomous Agents and Multiagent Systems, 2019.