On Explaining Agent Behaviour via Root Cause Analysis: A Formal Account Grounded in Theory of Mind

Shakil M. Khan [a;*] **and Maryam Rostamigiv**[a]

[a]University of Regina, Saskatchewan, Canada

Abstract. Inspired by a novel action-theoretic formalization of actual cause, Khan and Lespérance (2021) recently proposed a first account of causal knowledge that supports epistemic effects, models causal knowledge dynamics, and allows sensing actions to be causes of observed effects. To date, no other study has looked specifically at these issues. But their formalization is not sufficiently expressive to model explanations via causal analysis of mental states as it ignores a crucial aspect of theory of mind, namely motivations. In this paper, we build on their work to support causal reasoning about conative effects. In our framework, one can reason about causes of motivational states, and we allow motivation-altering actions to be causes of observed effects. We illustrate that this formalization along with a model of goal recognition can be utilized to explain agent behaviour in communicative multiagent contexts.

1 Introduction

Actual causality is a long-standing philosophical problem that is fundamental to the task of reasoning about and analysing observations. Given a narrative or history of events and an observed effect, solving this problem involves finding the events or actions from this history that are responsible for producing this effect, i.e. those that caused the effect. Also known as *token-level* causality, this problem is different from *general* or *type-level* causality, where the task is to discover universal causal mechanisms. Actual causality plays a significant role in reasoning about agents. For instance, causal reasoning can be used to explain the behaviour of a group of agents, e.g. via causal analysis of the mental states produced by this behaviour. These mental states may include beliefs and goals of the agents whose actions are the cause of the observed behaviour as well as those of others'.

Pearl [31, 32] was a pioneer in computational enquiry into actual causality. This line of research was later continued by Halpern [11], Halpern and Pearl [14], and others [7, 16, 17, 12, 13]. This "HP approach" is based on the concept of structural equations [40]. HP follows the Humean counterfactual definition of causation, which states that "an outcome B is caused by an event A" is the same as saying that "had A never occurred, B never had existed". This definition suffers from the problem of preemption[1]: it could be the case that in the absence of event A, B would still have occurred due to another event, which in the original trace was preempted by A. HP address this by performing counterfactual analysis only under carefully selected contingencies, which suspend some subset of the model's mechanisms. While their inspirational early work was shown to be useful for some practical applications, their approach based on Structural Equations Models (SEM) has been criticized for its limited expressiveness [16, 17, 10], and researchers have attempted to expand SEM with additional features, e.g. [23]. Note that despite recently reported progresses (e.g. [15]), many of these expressive limitations remain. Also, while there has been much work on actual causality, the vast majority of the work in this area has focused on defining causes from an objective standpoint.

In recent years, researchers have become increasingly interested in studying causation from the perspective of agents. Among other things, this is useful for defining important concepts such as responsibility and blame. Inspired by a novel action-theoretic formalization of actual causation [3], Khan and Lespérance [22] (KL, henceforth) recently proposed a first account of causal knowledge that supports epistemic effects, models causal knowledge dynamics, and allows sensing actions to be causes of observed effects. To date, no other study has looked specifically at these issues. But their formalization is not sufficiently expressive to model explanations via causal analysis of mental states as it ignores a crucial aspect of theory of mind, namely motivations. In this paper, we build on their work to support causal reasoning about conative effects. In our framework, one can reason about causes of motivational states, and we allow motivation-altering actions to be causes of observed effects. We illustrate that this formalization along with a model of goal recognition can be utilized to explain agent behaviour.

Our contribution in this paper is three-fold. First, we show how causal reasoning about goals/intentions can be modeled. Secondly, using an example, we illustrate how this formalization along with a model of goal recognition can be used to explain agent behaviour in communicative multiagent contexts. The generated explanations include both direct causal explanations as well as higher-order and more useful indirect explanations. The latter is grounded in (multiagent) theory of mind-based causal reasoning. Specifically, for this we define explanations as causes of intentions behind other explanations. Finally, while doing this, we extend a previously proposed account of goal change to deal with the *request* communicative action.

The paper is organized as follows. In the next section, we outline the situation calculus, a model of knowledge therein, and the formalization of infinite paths in the situation calculus, and introduce our running example. In Section 3, we discuss the formalization of prioritized goals and intentions, and propose a modification to the goal dynamics in [19] to deal with requests. Based on this, in Section 4, we present our logic of actual cause within the situation calculus that

* Corresponding Author. Email: shakil.khan@uregina.ca.

[1] Preemption happens when two competing events try to achieve the same effect, and the latter of these fails to do so, as the earlier one has already achieved the effect.

can deal with conative effects. In Section 5, we illustrate how our formalization can be utilized to explain agent behaviour. In Section 6, we prove some intuitive properties of our formalization. We conclude with some discussion in Section 7.

2 Action and Knowledge

The Situation Calculus Our base framework for modeling causal reasoning is the situation calculus (SC) [28] as formalized in [34]. Here, a possible state of the domain is represented by a situation. The initial state is denoted by S_0. There is a distinguished binary function symbol do where $do(a,s)$ denotes the successor situation to s resulting from performing the action a. Thus the situations can be viewed forming a tree, where the root of the tree is an initial situation and the arcs represent actions. As usual, a relational/functional fluent takes a situation term as its last argument. There is a special predicate $Poss(a,s)$ used to state that action a is executable in situation s. We will use the abbreviation $do([\alpha_1,\cdots,\alpha_n],S_0)$ to represent the situation obtained by consecutively performing $\alpha_1,\cdots,\alpha_n$ starting from S_0. Also, the notation $s \sqsubset s'$ means that situation s' can be reached from situation s by executing a sequence of actions. $s \sqsubseteq s'$ is an abbreviation of $s \sqsubset s' \vee s = s'$. $s < s'$ is an abbreviation of $s \sqsubset s' \wedge executable(s')$, where $executable(s)$ is defined as $\forall a', s'.\ do(a',s') \sqsubseteq s \supset Poss(a',s')$, i.e. every action performed in reaching situation s was possible in the situation in which it occurred. $s \leq s'$ is an abbreviation of $s < s' \vee s = s'$.

Our framework uses an action theory $\mathcal{D}$ that includes the following set of axioms:[2] (1) action precondition axioms (APA), one per action a characterizing $Poss(a,s)$, (2) successor state axioms (SSA), one per fluent, that succinctly encode both effect and frame axioms and specify exactly when the fluent changes, (3) initial state axioms describing what is true initially, (4) unique names axioms for actions, and (5) domain-independent foundational axioms describing the structure of situations [26].

Knowledge in the Situation Calculus Following [30, 35], 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. $K(d,s',s)$ is used to denote that in situation s, the agent d thinks that it could be in situation s'. Using K, the knowledge of an agent d is defined as:[3] $Know(d,\Phi,s) \stackrel{\text{def}}{=} \forall s'.\ K(d,s',s) \supset \Phi[s']$, i.e. the agent d knows Φ in s if Φ holds in all of its K-accessible situations in s. We also use the abbreviations $Kwhether(d,\Phi,s) \stackrel{\text{def}}{=} Know(d,\Phi,s) \vee Know(d,\neg\Phi,s)$, i.e. d knows whether Φ holds in s and $Kref(d,\theta,s) \stackrel{\text{def}}{=} \exists t.\ Know(d,\theta = t,s)$, i.e. it knows who/what θ refers to in s. 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 it has positive and negative introspection.

In our framework, the dynamics of knowledge is specified using a SSA for K that supports knowledge expansion as a result of sensing actions as well as communication actions. The information provided by a binary sensing action is specified using the predicate $SF(a,s)$. Similarly for non-binary sensing actions, the term $\mathit{sff}(a,s)$ is used to denote the sensing value returned by the action. These are specified using *sensed fluent axioms*; see [25] for details. Shapiro et al. [37] and later Lespérance [24] extended the SSA for K to support variants of the 'inform' communicative action. We will adopt the variant proposed in KL [18]. The preconditions of $inform(\mathit{inf}, agt, \Phi)$, which can be used by inf to inform agt that Φ, are as follows:

$$\begin{aligned}&Poss(inform(\mathit{inf}, agt, \Phi), s) \equiv\\ &\quad Know(\mathit{inf}, \Phi, s) \wedge \neg Know(\mathit{inf}, Know(agt, \Phi, now), s).\end{aligned}$$

We assume that its effects has been specified as in KL [18]. As shown in [35], the constraints on K then continue to hold after any sequence of actions since they are preserved by the SSA for K. A similar result can be shown for the KL [18] variant of the SSA for K.

Thus to model knowledge, we will use 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 and $inform$ are included. See [34] and [18] for details of these. Note that like [35], we assume that actions are fully observable (even if their effects are not). This can be generalized as in [1].

Paths in the Situation Calculus Following KL [21], we will formalize the sort of *paths* in the SC. A path is essentially an infinite sequence of situations, where each situation along the path can be reached by performing some executable action in the preceding situation. We will use $Starts(p,s)$ to denote that s is the earliest situation on path p and $OnPath(p,s)$ to denote that s is on p. $\mathit{Suffix}(p',p,s)$ means that path p' that starts with situation s is a suffix of p. KL [20] showed how one can interpret arbitrary CTL* formulae within SC with paths.[4] We assume that our theory $\mathcal{D}$ includes the axiomatization for paths.

We will use uppercase and lowercase Greek letters for state formulae (i.e. situation-suppressed SC formulae) and path formulae, respectively. These are inductively defined as follows:

$$\begin{aligned}\Phi &::= P(\vec{x}) \mid A\phi \mid \Phi \wedge \Phi \mid \neg\Phi \mid \forall x.\ \Phi\\ \phi &::= \Phi \mid \phi \wedge \phi \mid \neg\phi \mid \forall x.\ \phi \mid \bigcirc\phi \mid \phi\ \mathcal{U}\ \phi\end{aligned}$$

Here, $\vec{x}$ and x are object terms, $P(\vec{x})$ is an arbitrary situation-suppressed SC formula, and $A\phi$ (i.e. *over all paths* ϕ) is a path quantifier. Also, $\bigcirc\phi$ means that ϕ holds *next* over a path while $\phi\ \mathcal{U}\ \psi$ stands for ϕ *until* ψ. Other logical connectives and quantifiers such as $\vee, \supset, \equiv, \exists$ and CTL* operators such as $E\phi$, $\Diamond\phi$, $\phi\ \mathcal{B}\ \psi$, etc. are handled as the usual abbreviations (see below).

Like now in state formulae, path formulae ϕ can also contain an often-suppressed path placeholder $path$ in the place of the path terms. The function $[\![\cdot]\!]$ translates the above-defined formulae into formulae of the SC with paths. We write $\Phi[\![s]\!]$ (and $\phi[\![p]\!]$) to mean that state formula Φ (and path formula ϕ) holds in situation s (and over path p, respectively). In the following, we give the definition of

[2] We will be quantifying over formulae, and thus assume that $\mathcal{D}$ includes axioms for encoding of formulae as first order terms, as in [38].

[3] The state formula Φ can contain a placeholder *now* in the place of the situation terms. We often suppress *now* when the intent is clear from the context. Also, $\Phi[s]$ denotes the formula obtained by restoring the situation argument s into all fluents in Φ.

[4] CTL* [8] is a well-known branching-time temporal logic. It is a superset of Linear Temporal Logic (LTL) [33] and Computational Tree Logic (CTL) [5], and it allows arbitrary mixing of temporal operators and path quantifiers.

$[\![\cdot]\!]$ from [20].

$$P[\![s]\!] \stackrel{\text{def}}{=} P[s], \text{ where } P \text{ is a situation suppressed SC formula,}$$
$$A\phi[\![s]\!] \stackrel{\text{def}}{=} \forall p.\ Starts(p,s) \supset \phi[\![p]\!],$$
$$(\Phi \wedge \Psi)[\![s]\!] \stackrel{\text{def}}{=} \Phi[\![s]\!] \wedge \Psi[\![s]\!],$$
$$\neg\Phi[\![s]\!] \stackrel{\text{def}}{=} \neg(\Phi[\![s]\!]),$$
$$(\forall x.\ \Phi)[\![s]\!] \stackrel{\text{def}}{=} \forall x.(\Phi[\![s]\!]),$$

$$\Phi[\![p]\!] \stackrel{\text{def}}{=} \exists s.\ Starts(p,s) \wedge \Phi[\![s]\!],$$
$$(\phi \wedge \psi)[\![p]\!] \stackrel{\text{def}}{=} \phi[\![p]\!] \wedge \psi[\![p]\!],$$
$$\neg\phi[\![p]\!] \stackrel{\text{def}}{=} \neg(\phi[\![p]\!]),$$
$$(\forall x.\ \phi)[\![p]\!] \stackrel{\text{def}}{=} \forall x.(\phi[\![p]\!]),$$
$$\bigcirc\phi[\![p]\!] \stackrel{\text{def}}{=} \exists s,a,p'.\ Starts(p,s) \wedge Suffix(p',p,do(a,s)) \wedge \phi[\![p']\!],$$
$$(\phi\ \mathcal{U}\ \psi)[\![p]\!] \stackrel{\text{def}}{=} \exists s,s',p'.\ Starts(p,s) \wedge Suffix(p',p,s') \wedge \psi[\![p']\!] \wedge (\forall s^*,p^*.\ s \preceq s^* \prec s' \wedge Suffix(p^*,p,s^*) \supset \phi[\![p^*]\!]).$$

Thus, the situation suppressed situation calculus formula P holds in situation s if $P[s]$ is true, i.e. the formula obtained by reintroducing situation s in P holds in s. $A\phi[\![s]\!]$ holds if ϕ holds over all the paths in the future of s. ϕ holds next over a path p (i.e. $\bigcirc\phi[\![p]\!]$) if ϕ holds over the suffix of p that starts with the successor to the starting situation of p. ϕ until ψ holds over a path p (i.e. $(\phi\ \mathcal{U}\ \psi)[\![p]\!]$) if there is a suffix p' of p that starts with s', ψ holds over p', and for all suffixes p^* of p that start with an earlier situation s^* than s' (i.e. $s^* \prec s'$), ϕ holds over p^*. The rest are self-explanatory.

As mentioned above, other CTL* operators can be defined as usual abbreviations, e.g. *over some path* ϕ (denoted by $E\phi$), *eventually* ϕ (denoted by $\Diamond\phi$), *always* ϕ (denoted by $\Box\phi$), ϕ *unless* ψ (denoted by $\phi\ \mathcal{W}\ \psi$), ϕ *before* ψ (denoted by $\phi\ \mathcal{B}\ \psi$), etc.:

$$E\phi[\![s]\!] \stackrel{\text{def}}{=} \neg\ A\neg\phi[\![s]\!],$$
$$\Diamond\phi[\![p]\!] \stackrel{\text{def}}{=} (\text{True}\ \mathcal{U}\ \phi)[\![p]\!],$$
$$\Box\phi[\![p]\!] \stackrel{\text{def}}{=} \neg\Diamond\neg\phi[\![p]\!],$$
$$(\phi\ \mathcal{W}\ \psi)[\![p]\!] \stackrel{\text{def}}{=} ((\phi\ \mathcal{U}\ \psi) \vee \Box(\phi \wedge \neg\psi))[\![p]\!],$$
$$(\phi\ \mathcal{B}\ \psi)[\![p]\!] \stackrel{\text{def}}{=} \neg(\neg\phi\ \mathcal{U}\ \psi)[\![p]\!].$$

We will use α and σ, possibly with decorations, to represent ground action and situation terms, respectively. Finally, we will use uppercase Latin letters for ground terms, and lowercase Latin letters for variables.

Example For our running example, we consider a couple of simple rescue drone agents D_1 and D_2, a manager agent D_c, and their flight paths from one location to another. At anytime, an agent can be in any of the four locations L_s, L_d, L_1, and L_1'. The geometry of the flight paths is captured using the non-fluent relation $Route(l,l')$, which states that there is a flight path from location l to l' (throughout, we assume that free variables are universally quantified from the outside):[5]

$$(a). Route(l,l') \equiv [(l = L_s \wedge l' = L_1) \vee (l = L_s \wedge l' = L_1') \vee (l = L_1 \wedge l' = L_d) \vee (l = L_1' \wedge l' = L_d)].$$

The controller agent D_c is in charge of the overall mission and warns about potentially unsafe routes. Besides the $inform$ communicative action mentioned above, there are three additional actions in this domain. Action $takeOff(d,l)$ can be used by drone d to take off from location l, $flyTo(d,l,l')$ takes d from l to l', and $land(d,l)$ makes d land at l. There are four fluents in this domain, $At(d,l,s)$, $Flying(d,s)$, $Vis(d,l,s)$, and $TStorm(l,s)$, representing that d is located at l in situation s, that d is flying in s, that d has visited l in s, and that there is an ongoing thunderstorm at l in s.

The action preconditions in this domain are as follows:

$$(b).\ Poss(takeOff(d,l),s) \equiv At(d,l,s) \wedge \neg Flying(d,s),$$
$$(c).\ Poss(flyTo(d,l,l'),s) \equiv At(d,l,s) \wedge Flying(d,s) \wedge Route(l,l') \wedge \neg Know(d, TStorm(l'),s),$$
$$(d).\ Poss(land(d,l),s) \equiv At(d,l,s) \wedge Flying(d,s).$$

Thus, e.g., (c) states that a drone agent d can fly from locations l to l' in situation s iff it is located at l in s, it is flying in s, there is a route from l to l', and it does not know that there is a storm at l' in s.

Moreover, the SSA for the above fluents are as follows.

$$(e).\ At(d,l,do(a,s)) \equiv [\exists l'.\ a = flyTo(d,l',l) \vee (At(d,l,s) \wedge \neg\exists l'.\ a = flyTo(d,l,l'))],$$
$$(f).\ Flying(d,do(a,s)) \equiv [\exists l.\ a = takeOff(d,l) \vee (Flying(d,s) \wedge \neg\exists l.\ a = land(d,l))],$$
$$(g).\ Vis(d,l,do(a,s)) \equiv \exists l'.\ a = flyTo(d,l',l) \vee Vis(d,l,s),$$
$$(h).\ TStorm(l,do(a,s)) \equiv TStorm(l,s).$$

Thus, e.g., Axiom (e) states that d is at location l after executing action a in situation s (i.e. in $do(a,s)$) iff a refers to d's action of flying from some location l' to l, or d was already at l in s and a is not its action of flying to a different location l'. Note that, for simplicity, we essentially treat $TStorm$ in (h) as a non-fluent.

Initially, drone D_1 is at location L_s, is not flying, and has only visited L_s, and it knows these facts. Moreover, it does not know that there is a storm at location L_1, but knows that there are no storms at L_1' and L_d. There is indeed a thunderstorm at location L_1 and the controller agent D_c knows this. Finally, D_c does not know however that the other agents know this fact. These are captured using the following initial state axioms (note that $Know(d,\Phi(now),s) \supset \Phi[s]$):

$$(i).\ Know(D_1, At(D_1,L_s), S_0),$$
$$(j).\ Know(D_1, \neg Flying(D_1), S_0),$$
$$(k).\ Know(D_1, \forall l.\ Vis(D_1,l) \equiv l = L_s, S_0),$$
$$(l).\ \neg Know(D_1, TStorm(L_1), S_0),$$
$$(m).\ Know(D_1, \neg TStorm(L_1'), S_0),$$
$$(n).\ Know(D_1, \neg TStorm(L_d), S_0),$$
$$(o).\ Know(D_c, TStorm(L_1), S_0),$$
$$(p).\ \forall d.\ d \neq D_c \supset \neg Know(D_c, Know(d, TStorm(L_1)), S_0).$$

3 Formalizing Goals and Intentions

To model conative effects in the SC, we adopt the expressive formalization of prioritized goals (p-goals) and intentions proposed by KL [19]. In this framework, each p-goal is specified by its own accessibility relation G. To deal with multiple agents, we modify KL's proposal by adding an agent argument for all goal-related predicates and relations; usually the first argument for this. Given agent

[5] We assume that all agents know all non-fluent facts.

d, a path p is G-accessible at priority level n in situation s, denoted by $G(d,p,n,s)$, iff the goal of d at level n is satisfied over p and p starts with a situation that has the same action history as s. The latter requirement ensures that the agent's p-goal-accessible paths reflect the actions that have been performed so far. A smaller n represents higher priority, with 0 being the highest priority level. Thus the set of p-goals are totally ordered according to priority. We say that d has the p-goal that ϕ at level n in situation s iff ϕ holds over all paths that are G-accessible for d at n in s, i.e. $PGoal(d,\phi,n,s) \stackrel{\text{def}}{=} \forall p.\ G(d,p,n,s) \supset \phi[\![p]\!]$.

We assume that a domain theory $\mathcal{D}$ for our framework also includes the domain-dependent initial goal axioms (see below) and the domain-independent axioms and definitions that appear throughout this paper. As KL, we allow the agent to have infinitely many goals, some of which can be left unspecified.

For instance, assume that initially, our drone agent D_1 has the following two p-goals: $\phi_0 = \Diamond At(D_1, L_d)$, i.e. that it is eventually at L_d, and $\phi_1 = Vis(D_1, L_1)\ \mathcal{B}\ Vis(D_1, L_d)$, i.e. that it visits L_1 before it visits L_d, at level 0 and 1, respectively. Also, D_c does not have any initial p-goals. Then the initial goal hierarchy of D_1 and D_c can be specified using the following axioms:

$$
\begin{aligned}
&(q).\ Init(s) \supset \\
&\quad ((G(D_1,p,0,s) \equiv \exists s'.\ Starts(p,s') \wedge Init(s') \wedge \phi_0[\![p]\!]) \\
&\quad \wedge (G(D_1,p,1,s) \equiv \exists s'.\ Starts(p,s') \wedge Init(s') \wedge \phi_1[\![p]\!])), \\
&(r).\ Init(s) \wedge n \geq 2 \supset \\
&\quad (G(D_1,p,n,s) \equiv \exists s'.\ Starts(p,s') \wedge Init(s')), \\
&(s).\ Init(s) \wedge n \geq 0 \supset \\
&\quad (G(D_c,p,n,s) \equiv \exists s'.\ Starts(p,s') \wedge Init(s')).
\end{aligned}
$$

(q) specifies the p-goals ϕ_0, ϕ_1 (from highest to lowest priority) of D_1 in the initial situations, and makes $G(D_1,p,n,s)$ true for every path p that starts with an initial situation and over which ϕ_n holds, for $n = 0, 1$; each of them defines a set of initial goal paths for a given priority level, and must be consistent. (r) makes $G(D_1,p,n,s)$ true for every path p that starts with an initial situation for $n \geq 2$. Thus at levels $n \geq 2$, D_1 has the trivial p-goal that it be in an initial situation. The case for D_c is similar.

Assume that $\mathcal{D}_{dr}$ denotes our theory for the drone domain. Then in our example, we can show the following:

Proposition 1.

$$
\begin{aligned}
&\text{For } n < 2, \\
&\mathcal{D}_{dr} \models PGoal(D_1, \phi_n \wedge Starts(s) \wedge Init(s), n, S_0). \\
&\text{For } n \geq 2, \\
&\mathcal{D}_{dr} \models PGoal(D_1, Starts(s) \wedge Init(s), n, S_0).
\end{aligned}
$$

Since not all G-accessible paths are realistic in the sense that they start with a K-accessible situation, to filter the unrealistic paths out, KL defined *realistic* p-goal accessible paths:

$$G_R(d,p,n,s) \stackrel{\text{def}}{=} G(d,p,n,s) \wedge \exists s'.\ Starts(p,s') \wedge K(d,s',s).$$

G_R prunes out the paths from G that are known to be impossible, and since intentions are defined in terms of realistic p-goals, this ensures that these are realistic.

Using realistic p-goals-accessible paths, KL defined intentions as the realistic and maximal consistent prioritized intersection of the agent's goal hierarchy. First they specify all paths p that are in this prioritized intersection $G_\cap(d,p,n,s)$:[6]

$$
\begin{aligned}
&G_\cap(d,p,n,s) \equiv \\
&\quad \textbf{if } (n=0) \textbf{ then} \\
&\quad\quad \textbf{if } \exists p'.\ G_R(d,p',n,s) \textbf{ then } G_R(d,p,n,s) \\
&\quad\quad \textbf{else } \exists s'.\ Starts(p,s') \wedge K(d,s',s) \\
&\quad \textbf{else} \\
&\quad\quad \textbf{if } \exists p'.(G_R(d,p',n,s) \wedge G_\cap(d,p',n-1,s)) \\
&\quad\quad\quad \textbf{then } (G_R(d,p,n,s) \wedge G_\cap(d,p,n-1,s)) \\
&\quad\quad \textbf{else } G_\cap(d,p,n-1,s).
\end{aligned}
$$

Using this, they defined what it means for an agent to have an intention at some level n:[7]

$$Int(d,\phi,n,s) \stackrel{\text{def}}{=} \forall p.\ G_\cap(d,p,n,s) \supset \phi[\![p]\!],$$

i.e. an agent d has the intention at level n that ϕ in situation s if ϕ holds over all paths that are in the prioritized intersection of d's set of G_R-accessible paths up to level n in s. Finally, intentions are defined in terms of intentions at n:

$$Int(d,\phi,s) \stackrel{\text{def}}{=} \forall n.\ Int(d,\phi,n,s),$$

i.e. the agent d has the intention that ϕ in s if for any level n, ϕ is d's intention at n in s.

In our example, given the axioms above, since initially ϕ_0 and ϕ_1 are both realistic and are consistent with each other, we can show that initially D_1 has the intention that ϕ_0 and that ϕ_1:

Proposition 2.

$$\mathcal{D}_{dr} \models Int(D_1, \phi_0 \wedge \phi_1, S_0).$$

Goal Dynamics An agent's goals change when its knowledge changes as a result of the occurrence of an action, including exogenous events, or when it adopts or drops a goal. KL showed how this can be formalized by specifying how p-goals change. Intentions are then computed using realistic p-goals in every new situation as above.

Since for our example we only need to model cooperative agents that always respect the controller agent's requests, to simplify, we will modify KL's framework slightly by introducing a request communicative action and by getting rid of the actions for goal adoption and dropping. $req(d,d',\phi)$ can be used by an agent d to request to adopt a p-goal ϕ to another agent d'. The APA for this is as follows:

$$
\begin{aligned}
&Poss(req(d,d',\phi),s) \equiv \\
&\quad \neg Int(d, \neg\exists s', p'.\ Starts(s') \wedge \\
&\quad\quad\quad Suffix(p', do(req(d,d',\phi),s')) \wedge \phi[\![p']\!], s)
\end{aligned}
$$

That is, an agent d can request another agent d' to adopt the p-goal that ϕ if d does not intend in s that it is not the case that it executes the *req* action next and ϕ holds afterwards.

In the following, we specify the dynamics of p-goals by giving the modified SSA for G and discuss each case, one at a time:

$$
\begin{aligned}
&G(d,p,n,do(a,s)) \equiv \\
&\forall d', \phi.(a \neq req(d',d,\phi) \wedge Progressed(d,p,n,a,s)) \\
&\quad \vee \exists d', \phi.(a = req(d',d,\phi) \wedge Requested(d,p,n,a,s,\phi)).
\end{aligned}
$$

[6] The construct **if** ϕ **then** δ_1 **else** δ_2 is an abbreviation for $(\phi \supset \delta_1) \wedge (\neg\phi \supset \delta_2)$.

[7] KL used the term "chosen goals" (C-Goals) for this.

The overall idea for this is as follows. First of all, to handle the occurrence of a non-request (i.e. a regular or a request not directed to d) action a, we progress all of d's G-accessible paths to reflect the fact that a has just happened; this is done using the $Progressed(d, p, n, a, s)$ construct, which replaces each of d's G-accessible path p' with starting situation s', by its suffix p provided that it starts with $do(a, s')$:

$$Progressed(d, p, n, a, s) \stackrel{\text{def}}{=} \exists p', s'.\ G(d, p', n, s) \\ \wedge Starts(p', s') \wedge Suffix(p, p', do(a, s')).$$

Any path over which the next action performed is not a is eliminated from the respective G-accessibility level for d.

Secondly, to handle the request of a p-goal ϕ directed to d, we add a new p-goal level containing the requested p-goal ϕ to d's goal hierarchy at the highest priority by modifying the G-relation accordingly.[8] The G-accessible paths for d at level 0 are the ones that share the same history with $do(a, s)$ and over which ϕ holds. The G-accessible paths for d at all levels below 0 are the ones that can be obtained by progressing the level immediately above it. Thus the agent d acquires the p-goal that ϕ at the highest priority level 0, and all the p-goals in s are pushed down one level in the hierarchy.

$$\begin{array}{l} Requested(d, p, n, a, s, \phi) \stackrel{\text{def}}{=} \\ \quad \textbf{if } (n = 0) \textbf{ then} \\ \quad\quad \exists s'.\ Starts(p, s') \wedge SameHist(s', do(a, s)) \wedge \phi[\![p]\!] \\ \quad \textbf{else } Progressed(d, p, n-1, a, s). \end{array}$$

In our example, we can show that the agent D_1 will have the intention that $\Diamond Vis(D_1, L_1')$ after D_1 takes off from L_s, D_c informs D_1 that there is a thunderstorm at L_1, and D_c requests D_1 to eventually visit L_1', starting in S_0, i.e. in situation $S_3 = do([takeOff(D_1, L_s); inform(D_c, D_1, TStorm\ (L_1)); req(D_c, D_1, \Diamond Vis(D_1, L_1'))], S_0)$; thus:

Proposition 3.

$$\mathcal{D}_{dr} \models Int(D_1, \Diamond Vis(D_1, L_1'), S_3).$$

But D_1 will not have the intention that ϕ_1 as it has become impossible for D_1 to visit L_1 due to its knowledge of the thunderstorm at L_1, i.e.:

Proposition 4.

$$\mathcal{D}_{dr} \models \neg Int(D_1, \phi_1, S_3).$$

Proving the above two propositions involve progressing D_1's G-accessible paths using the SSA for G and then recomputing its intentions in S_3 using the definition of Int and the axiom for $G_\cap$.

4 Handling Conative Effects

Given a trace of events, *actual achievement causes* are the events that are behind achieving an effect.[9] To formalize reasoning about epistemic effects, KL [22] introduced the notion of *epistemic dynamic formulae in the SC*. An effect in their framework is thus an epistemic dynamic formula. We will extend this notion to that of *intentional dynamic formulae* φ to deal with conative effects (see below). Given an effect φ, the actual causes are defined relative to a *narrative* (variously known as a *scenario* or a *trace*) s. When s is ground, the tuple $\langle \varphi, s \rangle$ is often called a *causal setting* [3]. Also, it is assumed that s is executable, and φ was false before the execution of the actions in s, but became true afterwards, i.e. $\mathcal{D} \models executable(s) \wedge \neg\varphi\langle root(s)\rangle \wedge \varphi\langle s \rangle$, where $root(s) \doteq root(s')$, if $\exists a'.\ s = do(a', s')$, and $root(s) = s$, otherwise. Here $\varphi\langle s \rangle$ denotes the formula obtained from φ by restoring the appropriate situation argument into all fluents in φ (see Definition 2).

Note that since all changes in the SC result from actions, the potential causes of an effect φ are identified with a set of action terms occurring in s. However, since s might include multiple occurrences of the same action, one also needs to identify the situations where these actions were executed. To deal with this, KL required that each situation is associated with a time-stamp. Since in the context of knowledge, we will have different K-accessible situations where an action occurs, using time-stamps provides a common reference/rigid designator for the action occurrence. The initial situations start at time 0 and each action increments the time-stamp by one. Thus, our theory includes the following axioms:

$$Init(s) \supset time(s) = 0,$$
$$\forall a, s, t.\ time(do(a, s)) = t \equiv time(s) = t - 1.$$

With this, causes in this framework is a non-empty set of action-time-stamp pairs derived from the trace s given φ.

We now introduce our notion of *intentional dynamic formulae* (IF, henceforth):

Definition 1. *Let $\vec{x}$, θ_a, and $\vec{y}$ respectively range over object terms, action terms, and object and action variables. The class of* situation-suppressed intentional dynamic formulae φ *is defined inductively using the following grammar:*

$$\varphi ::= P(\vec{x}) \mid Poss(\theta_a) \mid After(\theta_a, \varphi) \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \\ \mid \exists \vec{y}.\ \varphi \mid Know(agt, \varphi) \mid Int(agt, \psi).$$

That is, an IF can be a situation-suppressed fluent, a formula that says that some action θ_a is possible, a formula that some IF holds after some action has occurred, a formula that can built from other IF using the usual connectives, or a formula that the agent knows that some IF holds or intends to bring about some path formula ψ. Note that φ can have quantification over object and action variables, but must not include quantification over situations or ordering over situations (i.e. $\sqsubset$) or arbitrary K or G-relations, i.e. those that do not come from the expansion of $Know/Int$. We will use φ for IF.

Note that the argument of Int in the above inductive definition is a path formula ψ. Thus to allow for IF in the context of Int, we need to redefine state formulae Φ to include IF φ:

$$\Phi ::= P(\vec{x}) \mid A\phi \mid \Phi \wedge \Phi \mid \neg\Phi \mid \forall x.\ \Phi \mid \varphi.$$

Also, the following addition to the definition of $\Phi[\cdot]$ is needed:

$$\Phi[s] \stackrel{\text{def}}{=} \varphi\langle s \rangle, \quad \text{if } \Phi \text{ is of the form } \varphi.$$

We define $\varphi\langle\cdot\rangle$ as follows:

[8] For simplicity, we assume that the requested goal is always adopted as the highest priority goal. Other sophisticated models, e.g. one where the requestee adopts the requested goal only if it is from a trusted source, it is consistent with its own set of core goals, and at just below these core goals, could have been modeled as easily.

[9] We do not conceptually distinguish between actions and events.

Definition 2.

$$\varphi\langle s\rangle \overset{\text{def}}{=} \begin{cases} P(\vec{x}, s) & \text{if } \varphi \text{ is } P(\vec{x}) \\ Poss(\theta_a, s) & \text{if } \varphi \text{ is } Poss(\theta_a) \\ \varphi'\langle do(\theta_a, s)\rangle & \text{if } \varphi \text{ is } After(\theta_a, \varphi') \\ \neg(\varphi'\langle s\rangle) & \text{if } \varphi \text{ is } (\neg\varphi') \\ \varphi_1\langle s\rangle \wedge \varphi_2\langle s\rangle & \text{if } \varphi \text{ is } (\varphi_1 \wedge \varphi_2) \\ \exists\vec{y}.\ (\varphi'\langle s\rangle) & \text{if } \varphi \text{ is } (\exists\vec{y}.\ \varphi') \\ \forall s'.\ K(d, s', s) \supset (\varphi'\langle s'\rangle) & \text{if } \varphi \text{ is } Know(d, \varphi') \\ \forall n.\ Int(d, \psi, n, s) & \text{if } \varphi \text{ is } Int(d, \psi). \end{cases}$$

We will now present the definition of causes in the SC. The idea behind how causes are computed is as follows. Given an effect φ and scenario s, if some action of the action sequence in s triggers the formula φ to change its truth value from false to true relative to $\mathcal{D}$, and if there are no actions in s after it that change the value of φ back to false, then this action is an actual cause of achieving φ in s. Such causes are referred to as *primary* causes:[10]

Definition 3 (Primary Cause)**.**

$$\begin{aligned} &CausesDirectly(a, t, \varphi, s) \overset{\text{def}}{=} \\ &\qquad \exists s_a.\ time(s_a) = t \wedge (root(s) < do(a, s_a) \leq s) \\ &\qquad \wedge \neg\varphi\langle s_a\rangle \wedge \forall s'.(do(a, s_a) \leq s' \leq s \supset \varphi\langle s'\rangle). \end{aligned}$$

That is, a executed at time t is the *primary cause* of effect φ in situation s iff a was executed in a situation with time-stamp t in scenario s, a caused φ to change its truth value 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 on the trace that contributed to the preconditions and the context conditions of a cause must be considered as a cause as well. The following definition captures both primary and indirect causes.[11]

Definition 4 (Actual Cause [22])**.**

$$\begin{aligned} &Causes(a, t, \varphi, s) \overset{\text{def}}{=} \\ &\forall P.[\forall a, t, s, \varphi.(CausesDirectly(a, t, \varphi, s) \supset P(a, t, \varphi, s)) \\ &\quad \wedge \forall a, t, s, \varphi.(\exists a', t', s'.(CausesDirectly(a', t', \varphi, s) \\ &\qquad\qquad \wedge time(s') = t' \wedge s' < s \\ &\qquad\qquad \wedge P(a, t, [Poss(a') \wedge After(a', \varphi)], s') \\ &\qquad \supset P(a, t, \varphi, s)) \\ &\quad] \supset P(a, t, \varphi, s). \end{aligned}$$

Thus, $Causes$ is defined to be the least relation P such that if a executed at time t directly causes φ in scenario s then (a, t, φ, s) is in P, and if a' executed at t' is a direct cause of φ in s, the time-stamp of s' is t', $s' < s$, and $(a, t, [Poss(a') \wedge After(a', \varphi)], s')$ is in P (i.e. a executed at t is a direct or indirect cause of $[Poss(a') \wedge After(a', \varphi)]$ in s'), then (a, t, φ, s) is in P. Here the effect $[Poss(a') \wedge After(a', \varphi)]$ requires a' to be executable and φ to hold after a'.

With these simple modifications, the framework is now capable of dealing with conative effects. To see this, consider the following scenario σ in our example, where $\sigma = do([takeOff(D_1, L_s);\ inform(D_c, D_1, TStorm(L_1));\ req(D_c, D_1, \Diamond Vis(D_1, L_1'));\ inform(D_c, D_2, TStorm(L_1));\ req(D_c, D_2, \Diamond Vis(D_2, L_1'));\ flyTo(D_1, L_s, L_1');\ flyTo(D_1, L_1', L_d)], S_0)$. There are 7 actions in this scenario. For convenience, we will use $\vec{\alpha_i}$ to denote the first i actions in this trace, and so $do([\vec{\alpha_5}], S_0)$ is the situation obtained from executing the first 5 actions starting in S_0. Now assume that we want to reason about the causes of the effect $\varphi_1 = Int(D_1, \Diamond Vis(D_1, L_1'))$ in scenario $\sigma_1 = do([\vec{\alpha_5}], S_0))$. Then we can show that:

Proposition 5.

$$\mathcal{D}_{dr} \models Causes(req(D_c, D_1, \Diamond Vis(D_1, L_1')), 2, \varphi_1, \sigma_1),$$

i.e. as expected, D_c's request to D_1 to eventually visit L_1' that was executed at time 2 is the cause of D_1's intention that $\Diamond Vis(D_1, L_1')$.

5 Reasoning about Agent Behaviour

We are now ready to formalize reasoning about agent behaviour via causation. Just like causes, an explanation in our framework is also modeled using an action-time-stamp pair (a, t). Agent behaviour, on the other hand, is captured using a situation s and relative to an observation φ. For this, we use the predicate $Explains(a, t, \varphi, s)$, which means that the action a executed at time t explains the behaviour of the agents captured in situation s relative to the observation φ. For example, $Explains(a_{dr}, t_{dr}, \varphi_2, \sigma)$ states that the behaviour of drones as modeled by situation/scenario σ relative to the effect that $\varphi_2 = Vis(D_1, L_1')$ can be explained by action a_{dr} executed at time t_{dr} (see below for the values/binding of a_{dr} and t_{dr}). Thus, (a_{dr}, t_{dr}) explains why the drone D_1 visited the location L_1'. Note that, just as in the case for achievement causation, we assume here that $\neg\varphi\langle root(s)\rangle \wedge \varphi\langle s\rangle$.

While explaining agent behaviour through the causes of the effect is reasonable, it may not always be insightful. For instance, we can argue that the behaviour of the drone D_1 in σ w.r.t. visiting L_1' can be explained by its action $flyTo(D_1, L_s, L_1')$ executed at time 5. However, this is obvious and is not very useful. A deeper level of explanation requires analyzing the motivations of the involved agents, in particular their intentions behind executing the actions that caused the effect.

To further explain agent behaviour, we will use an intention recognition component, which for this paper is considered to be a black-box module. We use the predicate $RRInt(d, \phi, a, t, s)$ to denote that agent d is recognized to have the relevant intention that ϕ in situation s w.r.t. the action a executed at time t. For instance, $RRInt(D_1, \Diamond Vis(D_1, L_1'), flyTo(D_1, L_s, L_1'), 5, \sigma)$ says that in scenario σ, agent D_1 is recognized to have the intention that $\Diamond Vis(D_1, L_1')$ for executing the action $flyTo(D_1, L_s, L_1')$ at time 5. With this, we can further explain an agent's behaviour via the root-cause analysis of its intentions behind performing actions. In our example, since D_1 flew to L_1' due to its intention that $\Diamond Vis(D_1, L_1')$, it is reasonable to explain its behaviour via the causes of having this intention in the first place. This will reveal that D_1 had this intention due to D_c's request, and thus its behaviour w.r.t. visiting L_1' can be explained by this request action.

We now give the definition for $Explains$:

[10] Definition 3 is a slightly generalized definition than that of [22], where the authors used S_0 instead of $root(s)$.

[11] In this, we need to quantify over situation-suppressed IF. 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 [9]. We assume that we have such an encoding and use formulae as terms directly.

Definition 5.

$$
\begin{aligned}
&\mathit{Explains}(a,t,\varphi,s) \stackrel{\text{def}}{=}\\
&\forall P.[\forall a,t,s,\varphi.(\neg\varphi\langle \mathit{root}(s)\rangle \wedge \varphi\langle s\rangle \wedge \mathit{Causes}(a,t,\varphi,s)\\
&\qquad\qquad \supset P(a,t,\varphi,s))\\
&\quad \wedge \forall a,t,s,\varphi.(\exists a',t',d',s',\psi.(P(a',t',\varphi,s) \wedge \mathit{agent}(a') = d'\\
&\qquad\qquad\qquad \wedge \mathit{RRInt}(d',\psi,a',t',s)\\
&\qquad\qquad\qquad \wedge \mathit{time}(s') = t' \wedge s' < s\\
&\qquad\qquad\qquad \wedge \neg \mathit{Int}(d',\psi,\mathit{root}(s')) \wedge \mathit{Int}(d',\psi,s')\\
&\qquad\qquad\qquad \wedge \mathit{Causes}(a,t,\mathit{Int}(d',\psi),s')\\
&\qquad\qquad \supset P(a,t,\varphi,s))\\
&\quad] \supset P(a,t,\varphi,s).
\end{aligned}
$$

Thus, $\mathit{Explains}$ is defined to be the least relation P such that if action a executed at time t causes φ in scenario s, then (a,t,φ,s) is in P, and if (a',t',φ,s) is in P (i.e. some other action a' executed at time t' explains φ in s), the agent of a' is d', d' is recognized to have the intention that ψ behind performing a' at t' in s, the timestamp of s' is t' , $s' < s$, and a executed at t was the cause of this intention in s', then (a,t,φ,s) is in P. Here $\mathit{agent}(a)$ denotes the agent of the action a; it can be specified as usual by an axiom that returns the agent of a, usually the first argument of a, i.e. $\mathit{agent}(a(d,\vec{x})) = d$. Also, s' is the situation where a' was executed. Finally, the two requirements that the effect be false before the execution of the actions in the scenario and becomes true afterwards, i.e. $\neg\varphi\langle \mathit{root}(s)\rangle \wedge \varphi\langle s\rangle$ and $\neg \mathit{Int}(d',\psi,\mathit{root}(s')) \wedge \mathit{Int}(d',\psi,s')$, are needed to ensure that the achievement causes actually exist.

Put otherwise, agent behaviour relative to the observation that φ in scenario s can be explained by the action a executed at time t iff a at t is a cause of φ in s, or a at t causes the intention behind some explanation of φ in s.

Returning to our example, we can now formally state the two explanations that we mentioned above and give the bindings for a_{dr} and t_{dr}. First, we can show that:

Proposition 6.

$$\mathcal{D}_{dr} \models \mathit{Explains}(\mathit{flyTo}(D_1,L_s,L_1'),5,\varphi_2,\sigma).$$

But perhaps more interestingly, we can show that:

Proposition 7.

$$
\begin{aligned}
&\mathcal{D}^*_{dr} \models \mathit{Explains}(\mathit{req}(D_c,D_1,\Diamond \mathit{Vis}(D_1,L_1')),2,\varphi_2,\sigma),\\
&\text{ where, } \mathcal{D}^*_{dr} \stackrel{\text{def}}{=}\\
&\quad \mathcal{D}_{dr} \cup \{\mathit{RRInt}(D_1,\Diamond \mathit{Vis}(D_1,L_1'),\mathit{flyTo}(D_1,L_s,L_1'),5,\sigma)\}.
\end{aligned}
$$

It is important to note that the scenario s in Definition 5 may and will often include the actions of multiple agents, and thus explanation of agent behaviour may trigger the analysis of the mental states of multiple agents. For example, given another suitable scenario, recognizing the intention behind the controller agent D_c's request to D_1 and analyzing this intention might have in turn exposed the causes behind its actions, e.g. due to its prior commitments to safety, etc. As such, the analysis performed here is truly multiagent in nature. Also, although our example only involves single-action/direct causes and we do not consider epistemic effects, as discussed above, the framework does support secondary causes and causal knowledge dynamics; see [22] for concrete examples of these.

A potential issue with the above definition of explanation is that in some domains, it might produce counter-intuitive results. Indeed, it is not very hard to come up with examples where the causes of the intention behind executing secondary and tertiary (i.e. non-direct) causes might be irrelevant. For example, a precondition of the $\mathit{flyTo}(D_1,L_s,L_1')$ action might have been that the drone D_1 has enough fuel, and thus refueling D_1 might have been an indirect cause of eventually visiting L_1'; however the causes of the intention behind refueling D_1 have nothing to do with visiting L_1'. Thus, while analyzing the causes of the intention behind a primary or direct cause seems useful, this is not always the case for indirect causes. To deal with this, the following variant of $\mathit{Explains}$ can be adopted.

$$
\begin{aligned}
&\mathit{Explains}(a,t,\varphi,s) \stackrel{\text{def}}{=}\\
&\quad (\neg\varphi\langle \mathit{root}(s)\rangle \wedge \varphi\langle s\rangle \wedge \mathit{Causes}(a,t,\varphi,s)) \vee\\
&\quad \mathit{ExplainsDirectly}(a,t,\varphi,s),
\end{aligned}
$$

where $\mathit{ExplainsDirectly}$ is just like $\mathit{Explains}$ but with Causes replaced with $\mathit{CausesDirectly}$.

6 Properties

We next discuss some general properties of our formalization. We start by showing that requests work as expected:

Theorem 1 (Adoption)**.**

$$
\begin{aligned}
&\mathcal{D} \models \neg \mathit{Know}(\mathit{agt},\neg\exists p,p'.\ \mathit{Starts}(p,\mathit{now}) \wedge\\
&\qquad \mathit{Suffix}(p',p,\mathit{do}(\mathit{req}(\mathit{agt}',\mathit{agt},\phi),\mathit{now})) \wedge \phi[\![p']\!],s) \supset\\
&\quad \mathit{Int}(\mathit{agt},\phi,\mathit{do}(\mathit{req}(\mathit{agt}',\mathit{agt},\phi),s)).
\end{aligned}
$$

Proof Sketch. Since by the antecedent and the SSA for K, there is a path that starts with a K-accessible situation in $\mathit{do}(\mathit{req}(\mathit{agt}',\mathit{agt},\phi),s)$ over which ϕ holds, by the SSA for G and the axiom for $G_\cap$, the $G_\cap$-accessible paths at level 0 in situation $\mathit{do}(\mathit{req}(\mathit{agt}',\mathit{agt},\phi),s)$ will only include paths over which ϕ holds. The rest of the proof follows from this, the definition of Int, and the axiom for $G_\cap$. ☐

That is, an agent agt acquires the intention that ϕ when requested by another agent agt' that ϕ in situation s provided that ϕ is not known to be impossible after the request action has happened in s. Note that this holds even if ϕ is inconsistent with agt's current intentions as the requested goal is always adopted at the highest priority level (again, this is by design). However, at all times the agent's intentions remain consistent. Thus if some other intention ϕ^+ at some priority level in s is inconsistent with the newly adopted goal ϕ, ϕ^+ is simply ignored when computing the intentions in $\mathit{do}(\mathit{req}(\mathit{agt}',\mathit{agt},\phi),s)$.

Next, we turn our attention to properties of explanations. Let $\mathcal{D}^* = \mathcal{D} \cup \mathcal{D}_{RRInt}$, where $\mathcal{D}_{RRInt}$ is a fixed set of $\mathit{RRInt}()$ sentences. First, note that explanations include causes, i.e.:

Corollary 1.

$$\mathcal{D}^* \models \mathit{Causes}(a,t,\varphi,s) \supset \mathit{Explains}(a,t,\varphi,s).$$

Moreover, we can show that explanations are not necessarily causes. This can be proven using Proposition 7 and the following lemma, which together produce a counterexample.

Lemma 1.

$$\mathcal{D}^*_{dr} \models \neg \mathit{Causes}(\mathit{req}(D_c,D_1,\Diamond \mathit{Vis}(D_1,L_1')),2,\varphi_2,\sigma).$$

We next show that explaining agent behaviour relative to logically equivalent effects has the same result:

Proposition 8 (Extensionality w.r.t. Effects)**.**

$$\mathcal{D}^* \models (\forall s.\ \varphi_1\langle s\rangle \equiv \varphi_2\langle s\rangle) \supset (\forall a, t, s.\ \mathit{Explains}(a, t, \varphi_1, s) \equiv \mathit{Explains}(a, t, \varphi_2, s)).$$

Proof. This follows from the fact that we are using a possible worlds/paths semantics. ☐

Finally, we study the conditions under which action occurrences do not alter knowledge about explanations.

Theorem 2 (Persistence of Knowledge about Explanations)**.**

$$\begin{aligned}\mathcal{D}^* \models \forall s, s', s^*, a, t, agt.\ & \mathit{executable}(s) \wedge \neg\varphi\langle \mathit{root}(s)\rangle \wedge \varphi\langle s\rangle \\ & \wedge \mathit{Kwhether}(agt, \mathit{Causes}(a, t, \varphi, \mathit{now}), s) \wedge s < s' \\ & \wedge (\forall s^*.\ s \leq s^* \leq s' \supset \mathit{Know}(agt, \varphi, s^*)) \\ & \supset \mathit{Kwhether}(agt, \mathit{Explains}(a, t, \varphi, \mathit{now}), s').\end{aligned}$$

Proof Sketch. We first show that the agent's causal knowledge remains unchanged when the antecedent holds. This can be shown by proving that the causes of φ remain the same in every epistemic alternative as φ remains true in every new situation. Since $\mathcal{D}_{RRInt}$ in $\mathcal{D}^*$ does not change, the rest of the proof then follows from this and the definition of *Explains*. ☐

That is, if an agent knows in s whether an action a executed at time t is a cause of an effect φ (and thus whether a in t explains φ in s), it will continue to know whether a executed at t explains φ in a future situation s', provided that its knowledge of the effect φ does not change between s and s'.

However, this is not the case in general. For instance, if the agent ceases to know that φ, then in this new situation it will not know what actions are causes, and consequently neither what actions explain φ.

7 Related Work and Conclusion

In this paper, we proposed a formal account of causal reasoning about motivations. Using this, we offered a novel take on explainable AI that is grounded in theory of mind: agent behaviour in our framework can be explained via the causal analysis of observed effects, which in turn can trigger the analysis of their mental states.

Recently, the pursuit of transparent and explainable AI systems has led to a renewed interest in the study of cognitive aspects of knowledge representation (KR), as advocates of KR often argue that its declarative nature makes it cognitively more suitable for explanation purpose. There has been some work on formalizing explanation in KR. For instance, in his early work, Shanahan [36] proposed a deductive and an abductive approach to explanation in the situation calculus, both of which are based on default reasoning. More recently, Shvo et al. [39] proposed a belief revision-based account of explanation. In their framework, a formula ϕ explains another formula ψ if revising by ϕ makes the agent believe ψ and the agent's beliefs remain consistent afterwards. In [6], Dennis and Oren used dialogue between the user and a Belief-Desire-Intention (BDI) agent system to explain why the agent has chosen a particular action. Their approach aims to identify any divergence of views that exist between the user and the BDI agent relative to the latter's behaviour and allows for an interactive and user-friendly explanation process. In his SEM-based formalization, Beckers [4] presented formal definitions of various causal notions of explanation and proposed to use actual causation for the purpose of explainable AI. The connections between these notions and the consequences of ignoring the causal structure are explored. Miller [29] proposed a contrastive explanation model based on structural causal models to enhance understanding and trust in AI decision-making. In [27], SEM-based causal models are utilized to generate explanations of the behaviour exhibited by model-free reinforcement learning agents. Finally, Sridharan et al. [41, 42] proposed an explainable robotic architecture by integrating step-wise refinement, non-monotonic reasoning, probabilistic planning, and interactive learning. However, none of the aforementioned work formalize causal analysis of agent motivation or employ such reasoning along with theory of mind for explaining agent behaviour (while Shvo, Klassen, and McIlraith [39] appealed to theory of mind, they did not address actual causation). In fact to the best of our knowledge, our proposal is the first and the only attempt to this end.

Our current formalization is limited in many ways. For instance, our proposal does not comply with many of the desiderata for explanations proposed by [39], in particular those that are related to belief and belief revision, since we only support knowledge and knowledge update. Also, we only allow deterministic and fully observable actions. Scenarios in our framework are linear, i.e. we assume that the order of action occurrence is known. This also means that while our proposal supports multiple agents, the underlying framework assumed by our work must ensure that these agents only act one at a time. When dealing with causation and explanations, we computed achievement causes only. In the literature, other types of causes has also been studied, e.g., *actual maintenance causes*; these are responsible for mitigating the threats to the achieved effect [2]. Incorporating other types of causes thus would have allowed us to explain effects further and in finer details. We leave these for future work.

Acknowledgements

We would like to thank the anonymous reviewers of ECAI, CAKR, and XLoKR 2023 for directing us to the relevant literature and helping us improve this paper considerably. We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), [funding reference number RGPIN-2022-03433].

References

[1] Fahiem Bacchus, Joseph Y. Halpern, and Hector J. Levesque, 'Reasoning about noisy sensors and effectors in the situation calculus', *Artificial Intelligence*, **111**(1-2), 171–208, (1999).

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

[3] Vitaliy Batusov and Mikhail Soutchanski, 'Situation calculus semantics for actual causality', in *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*, eds., Sheila A. McIlraith and Kilian Q. Weinberger, pp. 1744–1752. AAAI Press, (2018).

[4] Sander Beckers, 'Causal explanations and XAI', in *1st Conference on Causal Learning and Reasoning, CLeaR 2022, Sequoia Conference Center, Eureka, CA, USA, 11-13 April, 2022*, eds., Bernhard Schölkopf, Caroline Uhler, and Kun Zhang, volume 177 of *Proceedings of Machine Learning Research*, pp. 90–109. PMLR, (2022).

[5] Edmund M. Clarke and E. Allen Emerson, 'Design and synthesis of synchronization skeletons using branching-time temporal logic', in *Logics of Programs, Workshop, Yorktown Heights, New York, USA, May 1981*, ed., Dexter Kozen, volume 131 of *Lecture Notes in Computer Science*, pp. 52–71. Springer, (1981).

[6] Louise A. Dennis and Nir Oren, 'Explaining BDI agent behaviour through dialogue', *Auton. Agents Multi Agent Syst.*, **36**(1), 29, (2022).

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

[8] E. Allen Emerson and Joseph Y. Halpern, '"sometimes" and "not never" revisited: on branching versus linear time temporal logic', *J. ACM*, **33**(1), 151–178, (1986).

[9] 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).

[10] 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).

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

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

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

[14] Joseph Y. Halpern and Judea Pearl, 'Causes and explanations: A structural-model approach. part i: Causes', *The British Journal for the Philosophy of Science*, **56**(4), 843–887, (2005).

[15] Joseph Y. Halpern and Spencer Peters, 'Reasoning about causal models with infinitely many variables', in *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, (2022).

[16] Mark Hopkins, *The Actual Cause: From Intuition to Automation*, Ph.D. dissertation, University of California Los Angeles, 2005.

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

[18] Shakil M. Khan and Yves Lespérance, 'ECASL: a model of rational agency for communicating agents', in *4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), July 25-29, 2005, Utrecht, The Netherlands*, eds., Frank Dignum, Virginia Dignum, Sven Koenig, Sarit Kraus, Munindar P. Singh, and Michael J. Wooldridge, pp. 762–769. ACM, (2005).

[19] Shakil M. Khan and Yves Lespérance, 'A logical framework for prioritized goal change', in *9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Toronto, Canada, May 10-14, 2010, Volume 1-3*, eds., Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen, pp. 283–290. IFAAMAS, (2010).

[20] Shakil M. Khan and Yves Lespérance, 'Infinite paths in the situation calculus', Technical Report EECS-2015-05, Department of Electrical Engineering and Computer Science, York University, Toronto, Canada, (2015).

[21] Shakil M. Khan and Yves Lespérance, 'Infinite paths in the situation calculus: Axiomatization and properties', in *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016*, eds., Chitta Baral, James P. Delgrande, and Frank Wolter, pp. 565–568. AAAI Press, (2016).

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

[23] Florian Leitner-Fischer and Stefan Leue, 'Causality checking for complex system models', in *Verification, Model Checking, and Abstract Interpretation, 14th International Conference, VMCAI 2013, Rome, Italy, January 20-22, 2013. Proceedings*, eds., Roberto Giacobazzi, Josh Berdine, and Isabella Mastroeni, volume 7737 of *Lecture Notes in Computer Science*, pp. 248–267. Springer, (2013).

[24] Yves Lespérance, 'On the epistemic feasibility of plans in multiagent systems specifications', in *Intelligent Agents VIII, Agent Theories, Architectures, and Languages, 8th International Workshop, ATAL-2001, Seattle, WA, USA, Aug. 1-3, 2001, Revised papers*, eds., J.J.C. Meyer and M. Tambe, LNAI 2333, pp. 69–85. Springer, (2002).

[25] Hector J. Levesque, 'What is planning in the presence of sensing?', in *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI, IAAI, Volume 2*, pp. 1139–1146, (1996).

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

[27] Prashan Madumal, Tim Miller, Liz Sonenberg, and Frank Vetere, 'Explainable reinforcement learning through a causal lens', in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 2493–2500. AAAI Press, (2020).

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

[29] Tim Miller, 'Contrastive explanation: a structural-model approach', *Knowl. Eng. Rev.*, **36**, e14, (2021).

[30] Robert C. Moore, 'A formal theory of knowledge and action', in *Formal Theories of the Commonsense World*, pp. 319–358. Ablex, (1985).

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

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

[33] Amir Pnueli, 'The temporal logic of programs', in *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*, pp. 46–57. IEEE Computer Society, (1977).

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

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

[36] Murray Shanahan, 'Explanation in the situation calculus', in *Proceedings of the 13th International Joint Conference on Artificial Intelligence. Chambéry, France, August 28 - September 3, 1993*, ed., Ruzena Bajcsy, pp. 160–165. Morgan Kaufmann, (1993).

[37] Steven Shapiro, Yves Lespérance, and Hector J. Levesque, 'Specifying communicative multi-agent systems (invited paper)', in *Agents and Multi-Agent Systems Formalisms, Methodologies, and Applications, Based on the AI'97 Workshops on Commonsense Reasoning, Intelligent Agents, and Distributed Artificial Intelligence, Perth, Australia, December 1, 1997*, eds., Wayne Wobcke, Maurice Pagnucco, and Chengqi Zhang, volume 1441 of *Lecture Notes in Computer Science*, pp. 1–14. Springer, (1997).

[38] Steven Shapiro, Yves Lespérance, and Hector J. Levesque, 'Goal change in the situation calculus', *Journal of Logic and Computation*, **17**(5), 983–1018, (2007).

[39] Maayan Shvo, Toryn Q. Klassen, and Sheila A. McIlraith, 'Towards the role of theory of mind in explanation', in *Explainable, Transparent Autonomous Agents and Multi-Agent Systems - Second International Workshop, EXTRAAMAS 2020, Auckland, New Zealand, May 9-13, 2020, Revised Selected Papers*, eds., Davide Calvaresi, Amro Najjar, Michael Winikoff, and Kary Främling, volume 12175 of *Lecture Notes in Computer Science*, pp. 75–93. Springer, (2020).

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

[41] Mohan Sridharan, 'REBA-KRL: refinement-based architecture for knowledge representation, explainable reasoning and interactive learning in robotics', in *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)*, eds., Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pp. 2935–2936. IOS Press, (2020).

[42] Mohan Sridharan and Ben Meadows, 'Towards a theory of explanations for human-robot collaboration', *Künstliche Intell.*, **33**(4), 331–342, (2019).