Explaining Actual Causation via Reasoning About Actions and Change

Emily C. LeBlanc
College of Computing and Informatics
Drexel University
Philadelphia, PA
leblanc@drexel.edu

Abstract

In causality, an actual cause is often defined as an event responsible for bringing about a given outcome in a scenario. In practice, however, identifying this event alone is not always sufficient to provide a satisfactory explanation of how the outcome came to be. In this paper, we motivate this claim using well-known examples and present a novel framework for reasoning more deeply about actual causation. The framework reasons over a scenario and domain knowledge to identify additional events that helped to "set the stage" for the outcome. By leveraging techniques from Reasoning about Actions and Change, the approach supports reasoning over domains in which the evolution of the state of the world over time plays a critical role and enables one to identify and explain the circumstances that led to an outcome of interest. We utilize action language $\mathcal{AL}$ for defining the constructs of the framework. This language lends itself quite naturally to an automated translation to Answer Set Programming, using which, reasoning tasks of considerable complexity can be specified and executed. We speculate that a similar approach can also lead to the development of algorithms for our framework.

2012 ACM Subject Classification Computing methodologies → Knowledge representation and reasoning, Computing methodologies → Causal reasoning and diagnostics, Computing methodologies → Temporal reasoning

Keywords and phrases Actual Cause, Explanation, Reasoning about Actions and Change, Action Language, Answer Set Programming, Knowledge Representation and Reasoning

Digital Object Identifier 10.4230/OASIcs.ICLP.2018.16

1 Introduction and Problem Description

The comprehensive goal of this research has been to design, evaluate, and implement a novel causal reasoning framework to discover causal explanations that are in closer agreement with what common sense might lead one to conclude. Identifying actual causation concerns determining how a specified consequence came to be in a given scenario and has long been studied in a diversity of fields, including law, philosophy, and, more recently, computer science. Also referred to as *causation in fact*, actual causation is a broad term that encompasses all possible antecedents that have played a meaningful role in producing the consequence [5]. Consider the well-known Yale Shooting problem [16]:

> *Shooting a turkey with a loaded gun will kill it. Suzy loads the gun and then shoots the turkey. Why is the turkey dead?*

Intuition tells us that Suzy's shooting of the turkey is the *actual cause* of its death. However, if we know for certain that the gun was not loaded at the start of the story, then it is also important to recognize that Suzy's loading the gun played a key role in producing this consequence. On the other hand, if the gun was loaded from the start, then this point may

Technical Communications of the 34th International Conference on Logic Programming (ICLP 2018).
Editors: Alessandro Dal Palu', Paul Tarau, Neda Saeedloei, and Paul Fodor; Article No. 16; pp. 16:1–16:11
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

not be as significant. Moreover, if we build upon this example to say that Tommy handed Suzy the gun at the start of the scenario, then surely we want to identify Tommy's action as a contributory cause of the turkey's death. Hall [11] gives another classic example of actual causation in which two actors have each thrown a rock at a bottle and we wish to determine which actor's throw caused the bottle to break. It is easy to imagine similar extensions to the example that require deeper reasoning about causation to properly explain how the bottle broke – for example, did a third actor instruct the original two to throw their rocks in the first place? Literature examples aside, sophisticated actual causal reasoning has been prevalent in human society and continues to have an undeniable impact on the advancement of science, technology, medicine, and other important fields. From the development of ancient tools to modern root cause analysis in business and industry, reasoning about causal influence in a historical sequence of events enables us to diagnose the cause of an outcome of interest and gives us insight into how to bring about, or even prevent, similar outcomes in future scenarios. Consider problems such as explaining the occurrence of a set of suspicious observations in a monitoring system, reasoning about the efficiency actions taken in an emergency evacuation scenario, or verifying how an automatically generated workflow produces the expected results. It is easy to imagine that in cases such as these, determining surface-level causation (e.g., Suzy shot the turkey) may not be sufficient to provide a satisfactory explanation of how an outcome of interest to be.

In this dissertation work, we claim that reasoning about actual causation in complex scenarios requires the ability to identify more than the existence of a causal relationship. We may want a deeper understanding of the causal mechanism – was the outcome caused directly or indirectly? Did previously occurring events somehow *support* the causing event or the outcome's ability to be caused? To this end, the overall goal of the dissertation work is to investigate and demonstrate the suitability of action language and answer set programming to design and realize a novel approach to automated reasoning about actual causation as described above. The framework leverages techniques from Reasoning about Actions and Change (RAC) to support reasoning over domains that change over time in response to a sequence of events, as well as to answer queries for detailed causal explanations of an *outcome of interest* in a specific scenario. The language of choice for the formalization of knowledge is action language $\mathcal{AL}$ [2] which enables us to represent our knowledge of the direct and indirect effects of actions in a domain.

In the remainder of this summary, we present background on the action language $\mathcal{AL}$ and its semantics, provide an overview of the framework and its behavior on a novel actual causation scenario, survey existing literature, and finally discuss open issues and expected achievements for the dissertation.

2 Preliminaries

As we have already described, this work leverages techniques from Reasoning about Actions and Change [20] to support reasoning over domains that change over time. We assume that knowledge of a domain exists as a set of causal laws called an *action description* describing direct and indirect effects of actions using the action language $\mathcal{AL}$ [2]. These causal laws embody a transition diagram describing all possible world states of the domain and the events that trigger transitions between them. In the thesis investigation, we assume the existence of knowledge in this form, and while the work describes the formalization of the domain descriptions, the matter of the origin of knowledge is beyond the scope of the thesis.

The syntax of $\mathcal{AL}$ builds upon an alphabet consisting of a set $\mathcal{F}$ of symbols for *fluents* and a set $\mathcal{E}$ of symbols for *events*[1]. The $\mathcal{AL}$ is centered around a discrete-state-based representation of the evolution of the domain.

Fluents are boolean properties of the domain whose truth value may change over time. A *(fluent) literal* is a fluent f or its negation $\neg f$. Additionally, we define $\overline{f} = \neg f$ and $\overline{\neg f} = f$. A statement of the form

$$e \textbf{ causes } l_0 \textbf{ if } l_1, l_2, \ldots, l_n \tag{1}$$

is called *dynamic causal law*, and intuitively states that, if event e in $\mathcal{E}$ occurs in a state in which literals $l_1, \ldots, l_n$ hold, then l_0, the *consequence* of the law, will hold in the next state. A statement

$$l_0 \textbf{ if } l_1, \ldots, l_n \tag{2}$$

is called *state constraint* and says that, in any state in which $l_1, \ldots, l_n$ hold, l_0 also holds. This second kind of statement allows for an elegant and concise representation of indirect effects, which increases the flexibility of the language. Finally, an *executability condition* is a statement of the form:

$$e \textbf{ impossible\_if } l_1, \ldots, l_n \tag{3}$$

where e and $l_1, \ldots, l_n$ are as above. (3) states that e cannot occur if $l_1, \ldots, l_n$ hold. A set of statements of $\mathcal{AL}$ is called an *action description*. The semantics of an action description AD is defined by its *transition diagram* $\tau(AD)$, a directed graph $\langle N, E \rangle$ such that:

1. N is the collection of all states of AD;
2. E is the set of all triples $\langle \sigma, e, \sigma' \rangle$ where σ, σ' are states, e is an event executable in σ, and σ, e, σ' satisfy the *successor state equation* [17]:

$$\sigma' = Cn_Z(E(e, \sigma) \cup (\sigma \cap \sigma')) \tag{4}$$

where Z is the set of all state constraints of AD.

The argument of Cn_Z in (4) is the union of the set of direct effects $E(e, \sigma)$ of e, with the set $\sigma \cap \sigma'$ of the facts "preserved by inertia". The application of Cn_Z adds the "indirect effects" to this union. A triple $\langle \sigma, e, \sigma' \rangle \in E$ is called a *transition* of $\tau(AD)$ and σ' is a *successor state of* σ (under e). A sequence $\langle \sigma_1, \alpha_1, \sigma_2, \ldots, \alpha_k, \sigma_{k+1} \rangle$ is a *path of* $\tau(D)$ of length k if every $\langle \sigma_i,\ \alpha_i,\ \sigma_{i+1} \rangle$ is a transition in $\tau(D)$. We refer to state σ_1 of a path p as the *initial state* of p. A path of length 0 contains only an initial state. In the next section, we build upon this formalization to define a query to our framework for representing and reasoning about actual cause.

3 Framework Overview and Foundational Example

In this section, we provide an overview of the causal reasoning framework alongside a novel foundational example that showcases the reasoning capabilities and explanatory power of the framework. It is a straightforward scenario in which an outcome of interest, say θ_E, is not satisfied at the start of the scenario. After the occurrence of three events, say e_1, e_2,

[1] For convenience and compatibility with the terminology from RAC, in this paper we use *action* and *event* as synonyms.

and e_3, the outcome has been caused. Given the outcome of interest, the sequence of events, and knowledge of the domain in which they have occurred, our framework identifies causal explanations for how θ_E may have come to be. In order to explain actual causation, we will aim to characterize *transition events* which tell us the primary cause of an outcome and whether or not it was caused directly or indirectly, as well as *outcome* and *supporting events* which tell us which prior occurring events have contributed to causing the outcome.

Query

A query consists of an action description, a sequence of events, and the outcome of interest. The sequence of three scenario events and the outcome of interest for our example are represented by $v_E = \langle e_1, e_2, e_3 \rangle$, and $\theta_E = \{A, B, C, D, E, F\}$, respectively. The following action description AD_E characterizes events in the scenario's domain:

$$\begin{cases} e_1 \textbf{ impossible\_if } A & (5) \\ e_1 \textbf{ causes } E \textbf{ if } \neg E & (6) \\ e_2 \textbf{ causes } D \textbf{ if } \neg D & (7) \\ e_3 \textbf{ causes } A \textbf{ if } \neg A & (8) \\ e_3 \textbf{ causes } C \textbf{ if } \neg C & (9) \\ e_3 \textbf{ impossible\_if } \neg E & (10) \\ e_3 \textbf{ impossible\_if } \neg F & (11) \\ B \textbf{ if } C & (12) \end{cases}$$

Laws (5) and (6) describe event e_1, telling us that e_1 can only occur when A does not hold and e_1 will cause E if it does not already hold. Law (7) states that e_2 will cause D to hold if it does not already hold. Similar to causal laws (6) and (7), laws (8) and (9) tell us that e_3 will cause A and C to hold if they do not hold. The executability conditions (10) and (11) state that e_3 can only occur when both E and F hold. Finally, the state constraint (12) tells us that B holds whenever C holds. Given the action description AD_E, the sequence of events v_E, and the outcome of interest θ_E, the triple $\mathcal{Q}_E = \langle AD_E, v_E, \theta_E \rangle$ is the *query* for our example. Next, we introduce the concept of a *scenario path*, a unique mapping of the scenario described by a query to a representation of how the state of the world has changed in response to the events.

Scenario Path

Scenario paths represent a unique unfolding of a scenario and provide a convenient representation of how the domain changes over time in response to the events of the scenario. We reason over these paths to explain actual causation.

▶ **Definition 1.** Given a query $\mathcal{Q} = \langle AD, v, \theta \rangle$, a *scenario path* is a path $\rho = \langle \sigma_1, \alpha_1, \sigma_2, ..., \alpha_k, \sigma_{k+1} \rangle$ of $\tau(AD)$ satisfying the following conditions:

1. $\forall i, 1 \leq i \leq k$, $\alpha_i = e_i$
2. $\theta \not\subseteq \sigma_1$
3. $\exists i$, $1 < i \leq k+1$, $\theta \subseteq \sigma_i$

Condition 1 requires that the events in ρ correspond to the events of v, capturing the idea that each event of v represents a transition between states in ρ. Condition 2 requires that the set of fluent literals θ is not satisfied by the initial state of ρ, ensuring that the

Table 1 Tabular representation of the scenario path $\rho_E \in P(\mathcal{Q}_E)$.

State	*Event*	*State Affecting Law(s)*
$\sigma_1 = \{\neg A, \neg B, \neg C, \neg D, \neg E,\ F\}$	$\alpha_1 = e_1$	e_1 **causes** E **if** $\neg E$
$\sigma_2 = \{\neg A, \neg B, \neg C, \neg D,\ E,\ F\}$	$\alpha_2 = e_2$	e_2 **causes** D **if** $\neg D$
$\sigma_3 = \{\neg A, \neg B, \neg C,\ D,\ E,\ F\}$	$\alpha_3 = e_3$	e_3 **causes** A **if** $\neg A$, e_3 **causes** C **if** $\neg C$, B **if** C
$\sigma_4 = \{A,\ B,\ C,\ D,\ E,\ F\}$	–	–

outcome has not already been caused prior to the known events of the story. Condition 3 requires that θ is satisfied in at least one state after the initial state in ρ. Conditions 2 and 3 together ensure that at least one event is responsible for causing θ to hold in ρ. The successor state equation (4) tells us some event in the scenario path must have directly or indirectly caused θ to be satisfied at some point after the initial state. The set of all scenario paths with respect to the query $\mathcal{Q}$ is denoted by $P(\mathcal{Q}) = \{\rho_1, \rho_2, \ldots, \rho_m\}$.

It is clear that there are multiple valid scenario paths in the set $P(\mathcal{Q}_E)$, each representing a valid evolution of state in response to the scenario's events in the domain given by AD_E. For the purposes of this discussion, we choose a path with a complex causal mechanism that will exercise the causal reasoning framework. We will refer to this path as ρ_E. Table 1 shows the evolution of state in ρ_E in response to the events of v_E. The first column lists each state σ_i of ρ_E, and the second column gives the event α_i that caused a transition to the state σ_{i+1}. It is easy to see that ρ_E satisfies the conditions of Definition 1 with respect to AD_E, v_E, and θ_E.

Transition Event

A *transition event* is an event in a scenario path that causes a transition from a state of the world where the outcome θ is not satisfied to a state of the world where θ is satisfied. In this section, we identify transition events and their direct and indirect effects on the outcome.

▶ **Definition 2.** Given a scenario path $\rho = \langle \sigma_1, \alpha_1, \sigma_2, \ldots, \alpha_k, \sigma_{k+1} \rangle$ and an outcome θ, event α_j, where $1 \leq j \leq k$, is a *transition event* of θ in ρ if the following conditions are satisfied by the transition $\langle \sigma_j, \alpha_j, \sigma_{j+1} \rangle$ of ρ:

1. $\theta \not\subseteq \sigma_j$

2. $\theta \subseteq \sigma_{j+1}$

Intuitively, event α_j is a transition event of outcome θ if the outcome was not satisfied when α_j occurred but *was* satisfied after its occurrence. Note that we have defined transition events in such a way that there can be multiple transition events for θ in ρ. Using Table 1, it is straightforward to verify that event e_3 is the only transition event of θ_E in the example scenario path ρ_E, clearly satisfying Conditions 1 and 2 of Definition 2.

Given a query $\mathcal{Q} = \langle AD, v, \theta \rangle$, a scenario path $\rho = \langle \sigma_1, \alpha_1, \sigma_1, \ldots, \alpha_k, \alpha_{k+1} \rangle$ in $P(\mathcal{Q})$, and a transition event α_j for θ, the set of *direct effects of* α_j *in* θ is $d_\theta(\alpha_j, \rho) = \theta \cap E(\alpha_j, \sigma_j)$. Recall that $E(\alpha_j, \sigma_j)$ is the set of all direct effects of event α_j given that it occurs in state σ_j. The set of all direct effects of e_3 with respect to σ_3, then, is $E(e_3, \sigma_3) = \{A, C\}$, in accordance with laws (8) and (9) in AD_E. The direct effects of e_3 in θ_E, then, is given by $d_{\theta_E}(e_3, \rho_E) = \theta_E \cap E(e_3, \sigma_3) = \{A, B, C, D, E, F\} \cap \{A, C\} = \{A, C\}$.

To determine the indirect effects of an event with respect to the outcome, first let $S = E(\alpha_j, \sigma_j) \cup (\sigma_j \cap \sigma_{j+1})$ represent the set of all literals directly caused by the transition event α_j and those preserved by inertia. Given a query $\mathcal{Q} = \langle AD, v, \theta \rangle$, a scenario path

$\rho = \langle \sigma_1, \alpha_1, \sigma_1, \ldots, \alpha_k, \alpha_{k+1} \rangle$ in $P(\mathcal{Q})$, and a transition event α_j for θ, the set of *indirect effects of α_j in θ* is $i_\theta(\alpha_j, \rho) = \theta \cap (\sigma_{j+1} \setminus S)$. Given the set $S_E = E(e_3, \sigma_3) \cup (\sigma_3 \cap \sigma_4) = \{A, C\} \cup \{D, E, F\} = \{A, C, D, E, F\}$ representing the direct effects of e_3 and the literals preserved by inertia, the indirect effects of e_3 in θ_E is

$$\begin{aligned} i_{\theta_E}(e_3, \rho_E) =& \theta_E \cap (\sigma_4 \setminus S_E) \\ =& \{A, B, C, D, E, F\} \cap (\{A, B, C, D, E, F\} \setminus \{A, C, D, E, F\}) \\ =& \{A, B, C, D, E, F\} \cap \{B\} \\ =& \{B\} \end{aligned}$$

This result is intuitive because e_3 directly caused C to hold by law (9) and we know from law (12) that whenever C holds in a certain state, then B holds. We claim that under these conditions, it must be the case the e_3 caused B indirectly.

First Causal Explanation

Both the knowledge of the transition event and its effects on the outcome are represented by the *first causal explanation.* Given the query $\mathcal{Q}_E = \langle AD_E, v_E, \theta_E \rangle$, the scenario path $\rho_E \in P(\mathcal{Q}_E)$, the transition event e_3 in ρ_E, and e_3's direct and indirect effects, $d_{\theta_E}(\rho_E, \theta_E)$ and $i_{\theta_E}(\rho_E, \theta_E)$, respectively, the *first causal explanation* for θ_E in ρ_E is the tuple

$$\begin{aligned} C_E^1 &= \langle \rho_E, e_3, d_{\theta_E}(\rho_E, \theta_E), i_{\theta_E}(\rho_E, \theta_E) \rangle \\ &= \langle \rho_E, e_3, \{A, C\}, \{B\} \rangle \end{aligned}$$

Explanation C_E^1 summarizes our initial findings – the event e_3 caused a transition from a state where the outcome $\{A, B, C, D, E, F\}$ did not hold to a state where it did hold in the scenario path ρ_E. Specifically, literals A and C were direct effects of e_3's occurrence while e_3 caused B indirectly.

While C_E^1 tells us how the set of literals $\{A, B, C\}$ of θ_E were made to hold in scenario path ρ_E, we are still missing information about which, if any, events prior to e_3 caused the remaining literals $\{D, E, F\}$ to hold in state σ_4. We also do not know if any prior occurring events influenced e_3's ability to be a transition event of θ_E. In this work, supporting events are events that have occurred prior to a transition event α_j that enable α_j to be a transition event for the outcome θ. We identify two types of supporting events, *outcome supporting event* (OSEs) and *transition supporting events* (TSEs), both which are presented in the following sections. In order to identify both OSEs and TSEs in a scenario path ρ, we must first introduce the notion that an event α_i *ensures* that a literal l will hold in a specified state σ_j if it is the most recent transition event for l.

▶ **Definition 3.** Given a scenario path $\rho = \langle \sigma_1, \alpha_1, \sigma_2, \ldots, \alpha_k, \alpha_{k+1} \rangle$, event α_i is an *ensuring event* of $l \in \sigma_j$ in ρ if:

1. α_i is a transition event of $\{l\}$ in ρ
2. $i < j$
3. $j - i$ is minimal

Condition 1 leverages Definition 2 to require that event α_i responsible for l holding in some state of ρ. Condition 2 requires that α_i occurs before α_j in ρ. Condition 3 requires that α_i is the most recent transition event of l in ρ. We claim that if no event ensures $l \in \sigma_j$ for a path ρ, this implies that l holds in every state of ρ because there exists no transition $\langle \sigma_i, \alpha_i, \sigma_{i+1} \rangle$ in the path such that $l \notin \sigma_i$. Therefore, l must have held in the initial state and was never changed by a subsequent event prior to α_j's occurrence. Note that because ensuring events are also transition events, it is straightforward to leverage the characterizations of direct and indirect effects of transition events from Section 3 to learn if events ensured l in some state σ due to its direct or indirect effects.

Outcome Supporting Events

In the case where α_j does not set all of the literals of θ, OSEs can be responsible for ensuring that these remaining literals hold by the time α_j occurs in ρ. Finding OSEs requires first identifying if any literals in θ were not set as an effect of the transition event α_j. The set of remaining literals of an outcome θ is given by $R_\theta = \theta \setminus \ (\ d_\theta(\alpha_j, \rho) \ \cup \ i_\theta(\alpha_j, \rho))$. If $|R_\theta| > 0$, then a previously occurring event may have supported the outcome θ by ensuring that the remaining literals held in state σ_{j+1}.

▶ **Definition 4.** Given a query $\mathcal{Q}$, a factual path $\rho \in P(\mathcal{Q})$, a transition event α_j of θ, and a literal $l \in R_\theta$, α_i is an *outcome supporting event (OSE) via* l if α_i ensures $l \in \sigma_{j+1}$.

We denote by O^{supp} the set of OSEs and the literals they ensure. Formally, the tuple $\langle \alpha_i, l \rangle \in O^{supp}$ if α_i is a OSE via l. We denote by O^{init} the set of literals in R_θ that were not ensured by an event in ρ. Given a literal $l \in R_\theta$, $l \in O^{init}$ if:

$$\neg\exists\langle \alpha, l' \rangle \in O^{supp} \ s.t. \ \ l' = l$$

Intuitively, a literal l is in O^{init} when l has is no outcome supporting event in O^{supp}. In our example, we already know that we require additional causal information about the set of remaining outcome literals D, E, and F. Formally, the following literals in the outcome θ_E have not been explained by C^1_E:

$$\begin{aligned} R_{\theta_E} =& \theta_E \setminus (d_{\theta_E}(e_3, \rho_E) \cup i_{\theta_E}(e_3, \rho_E)) \\ =& \{A, B, C, D, E, F\} \setminus (\{A, C\} \cup \{B\}) \\ =& \{A, B, C, D, E, F\} \setminus \{A, C, B\} \\ =& \{D, E, F\} \end{aligned}$$

Because $|R_{\theta_E}| > 0$, there is more causal information to uncover. As covered in the earlier discussion on ensuring events, each literal in R_{θ_E} must either be ensured to hold in state σ_4 by an outcome supporting event or the literal has held consistently from the start of the scenario. Event e_2 is an outcome supporting event because it ensures that literal D held in σ_4. This event meets the three conditions of ensuring $D \in \sigma_4$. First, it is a transition event of $\{D\}$ because the literal D did not hold in state σ_2 but it did hold in σ_3 after e_2's occurrence. It clearly satisfies Conditions 2 because here $i = 2$ and $j = 4$, and so $i < j$. Finally, it satisfies Condition 3 because event e_i is the most recent transition event of $\{D\}$, and so $j - i$ is minimal. Similarly, it is straightforward to verify that e_1 is an outcome supporting event by ensuring that E holds in state σ_4. The set of outcome supporting events is given by $O^{supp}_E = \{\langle e_2, D \rangle, \langle e1, E \rangle\}$. Finally, the set $O^{supp}_E = \{F\}$ because there exists no tuple $\langle \alpha, F \rangle \in O^{supp}_E$, and so F must have held in the initial state of ρ_E and never changed value.

Second Causal Explanation

Knowledge of outcome supporting events and remaining outcome literals that held from the start is represented by the *second causal explanation*. Given the query $\mathcal{Q}_E = \langle AD_E, v_E, \theta_E \rangle$, the scenario path $\rho_E \in P(\mathcal{Q}_{\mathcal{E}})$, and the transition event e_3 for θ_E, the *second causal explanation* for θ_E in ρ_E is

$$\begin{aligned} C_E^2 =& \langle O_E^{supp}, O_E^{init} \rangle \\ =& \langle \{ \langle e_2, D \rangle, \langle e_1, E \rangle \}, \{F\} \rangle \end{aligned}$$

Explanation C_E^2 provides us with information about how the remaining outcome literals $\{D, E, F\} \in \theta_E$ came to hold in the state σ_4. Of these remaining literals, D and E were ensured by events e_2 and e_1, respectively. The remaining literal F held in the initial state and was not ensured in σ_4 by any event prior to e_1.

C_E^2 tells us how the remaining outcome literals came to hold in σ_4, but there is even more causal information to be revealed in this example. Next, we discuss an approach to determining if any other events in scenario path ρ_E contributed to e_3's ability to be a transition event of θ_E.

Transition Supporting Events

TSEs ensure that the preconditions of α_j are satisfied in state σ_j so that α_j could occur and cause θ to be satisfied in σ_{j+1}. The approach to identifying TSEs is conveniently similar to identifying outcome supporting events, and so we will omit the majority of technical details in favor of working out the example in the interest of space. To determine whether or not any prior events supported the transition event e_3, we begin by identifying all preconditions for e_3's occurrence and its ability to produce its effects in ρ_E. We obtain α_j's *preconditions* in ρ by reasoning over the of laws in AD. In the dissertation work, we introduce notation to allow reasoning over the components of laws in an action description AD. For example, given a dynamic causal law λ in AD of form (1), let $e(\lambda) = e$, $c(\lambda) = l_0$, and $p(\lambda) = \{l_1, l_2, \ldots, l_n\}$. We denote by $\mathcal{D}(AD)$ the set of all dynamic causal laws in AD. We use a similar representation for executability conditions, and we introduce a set of conditions under which preconditions can be extracted from these laws. In our example, the literals $\neg A$ and $\neg C$ are in $prec(e_3, \rho_E)$ because of laws (8) and (9) in the action description AD_E. By our definition of precondition, the literals E and F are also in $prec(e_3, \rho_E)$ because of laws (10) and (11) in AD_E. Therefore, the set of preconditions of e_3 in ρ_E is $prec(e_3, \rho_E) = \{\neg A, \neg C, E, F\}$.

Similar to our definition of outcome supporting events, a *transition supporting event* is the most recent transition event for a precondition of the transition event. It is straightforward to verify that the set of transition supporting events is given by $T_E^{supp} = \langle e_1, E \rangle$ and the set of initially set literals is $T_E^{init} = \{\neg A, \neg C, F\}$.

Third Causal Explanation

Knowledge of transition supporting events and precondition literals that held from the start is represented by the *third causal explanation.* Given the scenario path $\rho_E \in P(\mathcal{Q}_{\mathcal{E}})$, the transition event e_3, the set of transition supporting events T_E^{supp}, and the set of uncaused literals T_E^{init} the *third causal explanation* for θ_E in ρ_E is

$$\begin{aligned} C_E^3 =& \langle T_E^{supp}, T_E^{init} \rangle \\ =& \langle \{ \langle e_1, E \rangle \}, \{\neg A, \neg C, F\} \rangle \end{aligned}$$

Explanation C_E^3 tells us about the transition event e_3's preconditions and how they were met by state σ_3. The preconditions literals of event e_3 were $\neg A$, $\neg C$, E, and F. Of these precondition literals, E was ensured in σ_3 by the occurrence of event e_1. The remaining literals $\neg A$, $\neg C$, and F were not ensured in σ_3 by any scenario event. For relative brevity, we will not query further for details about the outcome and transition supporting events. It is easy to see, however, that the framework could tell us that the precondition literal E for e_3 was made to hold as a *direct effect* of e_1's occurrence.

Actual Causal Explanation

As the research intends to prove, there exists a space of possible structures for causal explanation. Recall that when there are remaining outcome literals to explain, there is a second causal explanation. However, if a transition event has no preconditions in the scenario path, then there is no third causal explanation. This implies that the structure of the explanation depends on the information encoded by the corresponding scenario path. We intend to characterize this space of structures in the dissertation. The framework can identify all three causal explanations in our example (i.e., C^1_E, C^2_E, and C^3_E). To summarize, the framework has explained that e_3 was a transition event for θ_E through both direct and indirect effects, e_1 and e_2 were outcome supporting events, and e_1 was a transition supporting event in the scenario path ρ_E.

4 Overview of Existing Literature

While actual causation has been treated in numerous ways in the Artificial Intelligence literature, the most relevant of which we will cover briefly in this section, existing approaches do not possess the fine-granularity of reasoning and explanation required to meet the reasoning needs of the examples discussed here. Many approaches to reasoning about actual cause have been inspired by the human intuition that cause can be determined by hypothesizing about whether or not a removing X from a scenario would prevent Y from being true [19]. Attempts to mathematically characterize actual causation have largely pursued counterfactual analysis of structural equations [22, 13, 15], neuron diagrams [12], and other logical formalisms [18, 23, 4]. It has been widely documented, however, that the counterfactual criteria alone is problematic and fails to recognize causation in some common cases such as preemption, overdetermination, and contributory cause [21, 10]. More recent approaches such as [14] have addressed some of these shortcomings by modifying the existing definitions of actual cause or by modeling change over time with some improved results. However, there is still no widely agreed upon counterfactual definition of actual cause in spite of a considerably large body of work aiming to find one.

The work of [3] departs from the counterfactual approach, using a similar insight to our own that actual causation can be determined by inspecting a specific scenario. Leveraging the Situation Calculus (SC) to formalize knowledge, the approach uses a single step regression approach to identify events deemed relevant to a logical statement becoming true. Although the conceptual approach is similar to our own, the technical approaches differ significantly. For example, [3] identifies a single sequence of causal events without explanation. There are also ramifications due to the choices for the formalization of the domain. Compared to $\mathcal{AL}$ formalizations, SC formalizations incur limitations when it comes to the representations of indirect effects of actions, which play an essential role in our work, and the elaboration tolerance of the formalization. Additionally, SC relies on First-Order Logic, while $\mathcal{AL}$ features an independent and arguably simpler semantics.

5 Open Issues and Expected Achievements

While the core of this framework is fairly well-developed at this stage, there remain some open issues that will be addressed in the dissertation. Evaluation of the framework is a crucial next step, and meaningful progress has been made towards demonstrating the framework's reasoning process when solving examples from causality literature in addition to novel scenarios. We expect to demonstrate that the framework can solve numerous classic examples with finer-grained causal explanations than the current state of the art. Moreover, the dissertation will present a number of empirical studies to compare and evaluate the ability of related approaches to solve the novel example presented in this paper. We expect that related approaches will not be able to explain the causal mechanism of our example in comparable detail. The dissertation will also present a novel set of identified open problems whose investigation can advance the capabilities of the causal reasoning framework. Regarding implementation, the choice of $\mathcal{AL}$ as the underlying formalism has useful practical implications. As demonstrated by a substantial body of literature (see, e.g., [1]), $\mathcal{AL}$ lends itself quite naturally to an automated translation to Answer Set Programming [8, 9], using which, complex reasoning tasks can be specified and executed (see, e.g., [6, 7]). We speculate that a similar approach can also lead to the development of algorithms for our framework, and have begun translating $\mathcal{AL}$ queries, scenario paths, and transition events to ASP.

References

1 Marcello Balduccini and Michael Gelfond. Diagnostic reasoning with A-Prolog. *arXiv preprint cs/0312040*, 2003.

2 Chitta Baral and Michael Gelfond. Reasoning agents in dynamic domains. In *Logic-based artificial intelligence*, pages 257–279. Springer, 2000.

3 Vitaliy Batusov and Mikhail Soutchanski. Situation calculus semantics for actual causality. In *13th International Symposium on Commonsense Reasoning. University College London, UK. Monday, November*, volume 6, 2017.

4 Sander Beckers and Joost Vennekens. A general framework for defining and extending actual causation using CP-logic. *International Journal of Approximate Reasoning*, 77:105–126, 2016.

5 Charles E Carpenter. Concurrent Causation. *University of Pennsylvania Law Review and American Law Register*, 83(8):941–952, 1935.

6 Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. Answer set planning under action costs. *Journal of Artificial Intelligence Research*, 19:25–71, 2003.

7 Esra Erdem, Michael Gelfond, and Nicola Leone. Applications of Answer Set Programming. *AI Magazine*, 37(3), 2016.

8 Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In *ICLP/SLP*, volume 88, pages 1070–1080, 1988.

9 Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New generation computing*, 9(3-4):365–385, 1991.

10 Clark Glymour and David Danks. Actual causation: a stone soup essay. *Synthese*, 175(2):169–192, 2010.

11 Ned Hall. Two concepts of causation. *Causation and counterfactuals*, pages 225–276, 2004.

12 Ned Hall. Structural equations and causation. *Philosophical Studies*, 132(1):109–136, 2007.

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

14 Joseph Y Halpern. *Actual causality.* MIT Press, 2016.

15 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.

16 Steve Hanks and Drew McDermott. Nonmonotonic logic and temporal projection. *Artificial intelligence*, 33(3):379–412, 1987.

17 Patrick J. Hayes and John McCarthy. Some Philosophical Problems from the Standpoint of Artificial Intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, 1969.

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

19 David Lewis. Causation. *The journal of philosophy*, 70(17):556–567, 1974.

20 J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Readings in artificial intelligence*, pages 431–450, 1969.

21 Peter Menzies. Counterfactual theories of causation. *The Stanford Encyclopedia of Philosophy*, 2001.

22 Judea Pearl. On the definition of actual cause, 1998.

23 Joost Vennekens. Actual causation in CP-logic. *Theory and Practice of Logic Programming*, 11(4-5):647–662, 2011.