Causal Analysis of Events Occurring in Trajectories of Dynamic Domains

Michael Gelfond and Evgenii Balai
Texas Tech University, Lubbock, Texas 79409, USA
(*e-mail:* michael.gelfond@ttu.edu)

submitted 1 January 2003; revised 1 January 2003; accepted 1 January 2003

Abstract

In this paper we define the notion of causes of events in trajectories of dynamic domains from the standpoint of an agent acting in this domain. We assume that the agent's knowledge about the domain is axiomatized in P-log with consistency restoring rules – a powerful knowledge representation language combining various forms of logical and probabilistic reasoning. The proposed model of causality is tested on a number of examples of causal domains frequently used in the literature.

KEYWORDS: causality, answer set programming, P-log

1 Introduction

This paper is a contribution to a research program aimed at finding precise mathematical formalization of substantial parts of commonsense knowledge and developing commonsense reasoning methods in knowledge representation languages based on Answer Set Prolog (ASP). We concentrate on *causal reasoning*, which seems to be of vital importance for our understanding of the world. The nature of causality and various causal relations has, for a long time, been debated by philosophers, physicists, statisticians, researchers in AI, etc. For recent AI work see, for instance, (Pearl 2009),(Halpern 2016),(Beckers and Vennekens 2012),(Bochman 2018). But despite the amazing progress, we do not yet have fully adequate understanding of the subject. There are still different interpretations of the intuitive meaning of causality, answers provided to causal questions by various formalisms do not always match the intuition, and some "causal stories" simply cannot be expressed in existing languages. In our approach we address these problems by using rich knowledge representation language capable of expressing non-trivial causal relations as well as various forms of commonsense background knowledge. We opted for logic programming language P-log with consistency-restoring rules (cr-rules) (Baral et al. 2009; Gelfond and Rushton 2010; Balai et al. 2019). It is an extension of ASP with well known methodology for representing defaults and their direct and indirect exceptions, recursive definitions, probability, direct and indirect effects of actions (including parallel and non-deterministic actions), time, etc. Its non-monotonic reasoning system combines standard ASP reasoning, abduction, and probabilistic computation. We are primarily interested in dynamic domains and, as in many theories of action and change, view the agent's knowledge base as a description of possible trajectories of the domain. The events in these trajectories are caused by actions. This is different from a large body of work in which the agent's knowledge is represented by structural equations, causal logic or

other formalisms emphasizing purely causal reasoning at the expense of background knowledge. Usually, but not always, these approaches provide counterfactual account of causality. There is a number of recent approaches (see, for instance, (Batusov and Soutchanski 2018; LeBlanc et al. 2019; Cabalar et al. 2014)) which seem to share our philosophy. There are, however, many substantial differences related to the power of our KR-language and other factors. The multiplicity of interpretations of the word *cause* is partially addressed by concentrating on what is often referred to as actual causes. In our approach time (or at least ordering of events) is an integral part of this notion. We further deal with this problem by dividing causes of events into those which consist of deliberate actions, those which contain at least one accidental (but known) action, and those which include some exogenous actions not native to the agent's model of the world. The methods of testing our definitions and KR methodology are determined by our goal. We view our work as a step in an attempt to pass what J.Pearl calls *Mini-Turing Test* (Pearl and Mackenzie 2018): "*The idea is to take a simple story, encode it on a machine in some way, and then test to see if the machine can correctly answer causal questions that a human can answer*." So, naturally, we use this to test accuracy of our modeling and relationship with other approaches. (Of course, only few of such examples are presented in this paper.) To make sure that wrong answers to these questions are not caused by inadequate representation of the problem we pay serious attention to developing KR methodology which properly combines causal and background knowledge about the domain. The paper is organized as follows. We assume some knowledge of P-log and define notions of an agent's *background theory* and *scenario*, which together form the agent's *causal theory*. The background theory contains general knowledge about the domain while the scenario describes a particular story to be analyzed. Next section contain definitions of three types of causes accompanied by some explanatory examples. Next we give examples of our approach by using it to analyze causal relations in several simple stories, followed by conclusion and future work. We assume reader's familiarity with ASP and some knowledge of P-log.

2 Representing Agent's Knowledge

Knowledge of an agent will be represented by a P-log program tailored for reasoning about effects of actions and causes of events. Regular function symbols of a program will be partitioned into *fluent*, *action*, *static* and *auxiliary* and used to form fluent, action, static and auxiliary terms respectively. We assume that actions are Boolean. The last parameter of functions from the first two groups is of a special sort called *time-step* (usually represented by natural numbers); time-steps are not allowed in statics. Recall that P-log terms formed by *random* are of the form $random(m, f(\bar{x}), p)$. This expression can also be viewed as at atom (a shorthand for $random(m, f, p) = true$), which states that, as the result of a random experiment m which is either genuine or deliberately interfered with, $f(\bar{x})$ should take the value from $\{Y : p(Y) \cap range(f)\}$. In addition, we require that for every time steps t_1 and t_2 occurring in m and f respectively, $t_2 > t_1$ if f is a fluent, and $t_2 \geq t_1$ if f is an action. Finally, both m and $random(m, f(\bar{x}), p)$ are viewed as action terms. Sometimes we say that $f(\bar{x}, t)$ is an instance of abstract fluent $f(\bar{x})$ and that the value y of $f(\bar{x}, t)$ is the value of abstract fluent $f(\bar{x})$ at time-step t. Similarly for abstract actions. Atoms formed by action terms are called *action atoms*. Similarly for fluent and static atoms. In this paper we are especially interested in properties of *events* – statements describing occurrences of actions and values of fluents at given time-steps. More precisely, an *action event* is a collection of action atoms. Similarly for *fluent events*. An *event* is a fluent event or an action event. The program representing the agent's knowledge is divided into two parts: a particular *story* (also called

domain scenario or *recorded history of the domain*) to be analyzed by the agent and *background theory* representing *general knowledge* about agent's domain.

Scenarios: Agent's scenario is a recorded history of time-stepped observations and deliberate (intended) actions which happened in the domain up to the current time-step. The initial time-step of a scenario is usually assumed to be 0. Observations are expressions of the form $obs(atom)$, actions are of the form $\mathbf{do}(a(\bar{x},t)=y)$. Expression $\mathbf{do}(a(\bar{x},t)=true)$ indicates a deliberate execution of action $a(\bar{x},t)$; $\mathbf{do}(a(\bar{x},t)=false)$ indicates a deliberate refusal of such execution. Another form of **do**-operator is very similar to *do*-operator of the original P-log (Baral et al. 2009) and Pearl's causal networks (Pearl 2009). If $random(m,e(\bar{x},t),p)$ is a random experiment and y is a possible value of $e(\bar{x},t)$ then $\mathbf{do}(m,y)$ represents an intervention into random experiment m which forces $e(\bar{x},t)$ to take value y. If the value y of $e(\bar{x},t)$ is simply observed this is recorded as $obs(e(\bar{x},t)=y)$. Whenever possible, we omit **do** from the description of actions in our scenarios.

Background Theory. An agent's background theory T is a P-log program which satisfies the following conditions:

- *T contains no deliberate actions and observations.* Theory T will be used in conjunction with a scenario S which will contain all deliberate actions and observations of the agent.
- The *program T^{reg} obtained from T by removing its cr-rules has a possible world.* This guarantees that the program's description of its dynamic domain does not use rare events. Such events can only be used to resolve unexpected actions and observations from a particular scenario.
- *If a time-step t occurs in the head of rule r from T then time-steps of fluents and actions in the body of r shall not exceed t and $t-1$ respectively.* This is a broadly shared principle of causality which says that the cause must precede its effect.
- *T contains a set of causal mechanisms* or *causal laws*, which have the form

  — – if a is an action:

  $$m:\ a(\bar{x})=y \leftarrow body,\ \text{not}\ ab(m),\ \text{not}\ interfere(a(\bar{x}),y)^{1} \qquad (*)$$

  where m is the mechanism's name, ab and $interfere$ are auxiliary functions; $interfere(a(\bar{x}),y)$ holds if action $a(\bar{x})$ is deliberately assigned value different from y; $ab(m)$ captures indirect exceptions to m.

  – If a is a fluent:

  $$m:\ a(\bar{x})=y \leftarrow body,\ \text{not}\ ab(m)$$

  In both rules *body* is non-empty and contains no default negation.

  — Contingency Axiom

  $$ab(m) \overset{+}{\leftarrow} body$$

  If the causal mechanism is not defeasible, then the contingency axiom and the corresponding "*not* $ab(m)$" from the body of a causal law can be omitted. Intuitively, the first two rules say that normally *body* is a sufficient cause for *head*. The guard $interfere(a(\bar{x}),y)$ present in rule $(*)$ allows deliberate actions to defeat triggering defaults. We will use shorthand

[1] If $a(\bar{x})$ is formed by $random(r,f(\bar{u}),p)$, then m is omitted and the causal mechanism is named r. Random experiments are normally named by action terms.

$interfere(a(\bar{x}))$ to denote $interfere(a(\bar{x}),true)$. The last axiom, referred to as the Contingency Axiom for m, allows causal mechanisms to be defeated by observations. It is a *consistency-restoring rule* of a version of ASP called *CR-Prolog* (Balduccini and Gelfond 2003b). It says that "causal mechanisms m may be disabled, but such a possibility is very rare and, whenever possible, should be ignored". For more details, see the Appendix 1 and/or (Gelfond and Kahl 2014).

The default, expressing the contingency axiom for m, can be accompanied by records of direct exceptions to this default, default preferences, consequences of $ab(m(\bar{x}))$, etc. For instance, if $\bar{x}$ is a strong exception to m, i.e., the head, $a(\bar{x}) = y$ of the rule must be defeated we can add

$$random(name, a(\bar{x})) : \{Y : Y \neq y\})\} \leftarrow ab(m(\bar{x}))$$

or simply

$$\neg a(\bar{x}) \leftarrow ab(m(\bar{x}))$$

if a is Boolean and $y = true$. (Similarly for $y = false$.)

Let us illustrate the notion of agent's background theory by formalizing two informal examples frequently used in the literature on causation.

Example 2.1 (*Firing Squad*)
A certain chain of events is required for a lawful execution of a prisoner. First, the court must order the execution. The order goes to a captain, who signals the soldiers on the firing squad (denoted by a and b) to shoot. We'll assume that they are obedient and expert marksmen, so they only shoot on command, and if either of them shoots, the prisoner dies.

Background theory FS for this example contains abstract actions $court\_order$, $captain\_order$, $shoot(a)$ and $shoot(b)$, inertial abstract fluent $dead$, and standard auxiliary symbols ab and $interfere$. FS consists of causal mechanisms:

$$(1a) \quad [m_1(T)]: \quad captain\_order(T+1) \leftarrow court\_order(T), \text{ not } ab(m_1(T)), \text{ not } interfere(captain\_order(T+1))$$

which is a defeasible version of dynamic causal law used in actions languages. Two other rules:

$$(1b) \quad [m_2(G,T)]: \quad shoot(G,T+1) \leftarrow captain\_order(T), \text{ not } ab(m_2(G,T)), \text{ not } interfere(shoot(G,T+1))$$

$$(1c) \quad [m_3(G,T)]: \quad dead(T+1) \leftarrow shoot(G,T), \text{ not } ab(m_3(G,T))$$

are defeasible triggers. We also have the contingency axioms

$2a \quad ab(m_1(T)) \stackrel{+}{\leftarrow} court\_order(T)$
$2b \quad ab(m_2(G,T)) \stackrel{+}{\leftarrow} captain\_order(T)$
$2c \quad ab(m_3(G,T)) \stackrel{+}{\leftarrow} shoot(G,T)$

The closed world assumptions (CWA) for actions are given by defaults:

$3a \quad \neg shoot(G,T) \leftarrow \text{ not } shoot(G,T)$

$3b \quad \neg captain\_order(T) \leftarrow \text{not } captain\_order(T)$
$3c \quad \neg court\_order(T) \leftarrow \text{not } court\_order(T)$

Since we know that at rare occasions the court can order prisoner's execution we allow to use this possibility to explain some of our observations. This is done by a cr-rule:

$4 \quad court\_order(T) \stackrel{+}{\leftarrow}$

which can be viewed as a typical indirect exception to a default. In principle, similar axioms can be written for (3a) and (3b) but, for simplicity we assume that though captain may, at rare occasions, do not follow the court order or even shoot on his own these events are so rare that we will not use them for explanations. Similarly, for guards. Hence, we will have no indirect exceptions for (3a) and (3b)[2].

We also need inertia axioms for $dead$:

$5a \quad \neg dead(T+1) \leftarrow \neg dead(T), \text{not } dead(T+1)$
$5b \quad dead(T+1) \leftarrow dead(T), \text{not } \neg dead(T+1)$

Note that, despite the fact that the story insists that the guards *only* shoot on command, the corresponding causal law is defeasible. This is essential since we would like to consider scenarios in which, say, guards may refuse to follow the orders or simply fail to do so by unspecified reasons. Similarly for other causal mechanisms.

The next example shows a simple non-deterministic background theory describing a throw of a coin.

Example 2.2 (*Flipping a Coin*)
A theory TC has "transient" fluent *agreed_to_play* (players agreed to start the game), and a "transient" fluent *head* (the coin landed heads). Transient fluents are partial functions which do not satisfy inertia; *head* is defined at time-steps immediately following $flip$. TC also contain action $flip$ (flip the coin). Causal mechanism

$$\begin{aligned} random(flip(T), head(T+1)) \quad \leftarrow \quad & agreed\_to\_play(T), \\ & \text{not } ab(m(T)), \\ & \text{not } interfere(random(flip(T), head(T+1))) \end{aligned}$$

together with the usual contingency axiom states that *agreed_to_play* normally triggers an occurrence of a random experiment $flip$ which ends in heads or in tails.

Agent's Causal Theory and its Models. The agent's knowledge about the domain will be represented by causal theory $T(S)$ which combines general knowledge from T with a particular scenario S. We refer to it as *agent's causal theory*. To define syntax and semantics of $T(S)$, we need to explain the exact encoding of S which is actually used in the definition of $T(S)$, describe special axioms enforcing the intuitive meaning of its constructs and define preference relation between the sets of cr-rules of a program tailored to causal reasoning.

A *scenario S is encoded in P-log* as follows: $obs(A)$, where A is time-stepped by 0 is encoded

[2] Of course this assumption can be easily removed by adding axiom $captain\_order(1) \stackrel{+}{\leftarrow} not\ court\_order(0)$. Similarly for the guards.

by A; If the time-step of A is greater than 0 then $obs(A)$ is encoded by a constraint: "$\leftarrow$ not A". Do-statements of S remain unchanged. We denote this encoding by $enc(S)$.

Causal theory $T(S)$ representing the agent's knowledge is defined as follows:

$$T(S) =_{def} T \cup enc(S) \cup DO.$$

where S is a scenario of T and DO is a collection of special axioms which, together with General Axioms of P-log (see Appendix 2) enforce the intuitive meaning of **do**:

- for every $\mathbf{do}(a(\bar{x},t) = y)$ from S axiom

  $$a(\bar{x},t) = y \leftarrow \mathbf{do}(a(\bar{x},t) = y)$$

  connects **do** with an actual occurrence of an action;
- and

  $$interfere(a(\bar{x},t) = Y) \leftarrow Y \neq y, \mathbf{do}(a(\bar{x},t) = y)$$

  which allows a deliberate action interfere with a defeasible trigger assigning value y to action $a(\bar{x},t)$.
- for every $\mathbf{do}(m,y)$ from S

  $$do(m,a(\bar{x},t),y) \leftarrow \mathbf{do}(m,y)$$

  guarantees that on random experiments **do** coincides with the original do of P-log and

  $$interfere(a(\bar{x},t),Y) \leftarrow Y \neq y, \mathbf{do}(m,y)$$

  to define interference with random experiments.

Usually we omit *enc* and simply write S instead. *We only consider S for which $T(S)$ is a coherent P-log program in which multiplicity of models can only be a result of general axiom (19) from (Balai et al. 2019) for random.*

To complete the definition of $T(S)$ we have to define preference relation on sets of CR-rules of the program. The ultimate choice of such a relation requires further investigation so our definitions will be parameterized with respect to such preference relations. The difference, however, will only show in the strategy employed by an agent in forming explanations of unexpected observations. Other patterns of reasoning will remain unchanged. *In the examples we usually use standard preference relations of CR-Prolog based on ordering of sets by the subset relation. An additional preference, which maybe more suitable for causal reasoning, minimizes the number of of cr-rules which are not relevant to actions and observations from S.* The precise definition of such relevance based preference is given in the appendix. It remains to be seen if it will require further modification.

As any P-log program, $T(S)$ comes with the definition of its possible worlds and probability function. A possible world W of $T(S)$ can be written as

$$W = \langle \sigma(t_f), \alpha(t_f), \ldots, \sigma(i), \alpha(i), \ldots \alpha(t_l - 1), \sigma(t_l) \rangle$$

where $\alpha(i)$ is the set of all action events from W time-stepped by i and $\sigma(i)$ is the set of all fluent atoms of W time-stepped by i and statics from W. It is tempting to view W as a possible *trajectory* of a dynamic system associated with $T(S)$. This is indeed the case when the regular part $T^{reg}(S)$ of $T(S)$ is consistent, i.e., no cr-rules are used in the construction of its world views. If it is not the case, however, then possible worlds are determined by our preference relation over sets of

cr-rules. Note that though all actions from $\alpha(i)$ start at i, their effects may manifest themselves at different time-steps.

Definition 2.1 (*Model and Entailment*)
A *model* of scenario S of T is a possible world of $T(S)$. $T(S)$ *entails* an event E ($T(S) \models E$) if E holds in all models of $T(S)$.

Let us demonstrate this notion by going back to the Firing Squad example.

Example 2.3 (*Firing Squad: models*)
Let us fix background theory FS from Example 2.1. Then, in the model of the scenario

$$S_0 = \langle obs(\neg dead(0)) \rangle$$

the prisoner is alive at every time step of the model. There are no actions. Scenario

$$S_1 = \langle obs(\neg dead(0)), court\_order(0) \rangle$$

has one model, M[3]: (When displaying a model we usually omit negations of actions derived by the CWA and the **do** statements.)

$\neg dead(0), court\_order(0)$, $\neg dead(1), captain\_order(1)$
$\neg dead(2), shoot(a,2), shoot(b,2)$, $dead(3)$, $interfere(court\_order(0), false)$.

The next scenario

$$S_2 = \langle obs(\neg dead(0)), court\_order(0), \neg shoot(a,2), \neg shoot(b,2) \rangle$$

is more interesting. Deliberate actions $\neg shoot(a,2), \neg shoot(b,2)$ from S together with axioms from DO cancel axiom $m_2(G,2)$. The only model M of S_2 is

$\neg dead(0), court\_order(0)$
$\neg dead(1), captain\_order(1)$
$\neg dead(2), interfere(a,2), interfere(b,2), \neg shoot(a,2), \neg shoot(b,2)$,
$\neg dead(3)$

(In what follows we do not show atoms formed by $interfere$ in the models). Next consider scenario

$$S_3 = \langle obs(\neg dead(0)), court\_order(0), obs(\neg dead(3)) \rangle$$

with a non-initial observation which contradicts the effects of our causal mechanisms. The contradiction can be resolved by assuming that the captain was not able to follow the court order or that his order could not have been executed by the guards. This is done by the Contingency Axioms. The contradiction can be avoided by finding abductive support of $FS(S_3)$.

In our case there are two such supports: R_1 consisting of contingency axioms of the form (2b) and R_2 consisting of (2a). The first derives $ab(m_2(a,1))$ and $ab(m_2(b,1))$ and hence disable $m_2(a,1)$ and $m_2(b,1)$. By inertia, FS^{R_1} will conclude $\neg dead(3)$ which leads to the first model M_1 of FS:

$ab(m_2(a,1))$, $ab(m_2(b,1))$
$\neg dead(0), court\_order(0)$

[3] The model can be computed using our prototype P-log solver. For more details, refer to `https://github.com/iensen/plog2.0/tree/master/plogapp/tests/causality`.

$\neg dead(1), captain\_order(1)$
$\neg dead(2), \neg shoot(a,2), \neg shoot(b,2),$
$\neg dead(3)$

Abductive support R_2 disables causal connection between court and captain orders and, by CWA, no order will be given by the captain. This will lead to the second model M_2 of FS:

$ab(m_1(0))$
$\neg dead(0), court\_order(0)$
$\neg dead(1), \neg captain\_order(1)$
$\neg dead(2), \neg shoot(a,2), \neg shoot(b,2),$
$\neg dead(3)$

In both cases $\neg dead(3)$ is proven by inertia.

In scenario

$$S_4 = \langle obs(\neg dead(0)), obs(dead(3))\rangle$$

the only way to satisfy the last observation is to use rule (4) of FS and assume $court\_order(0)$.The resulting model, M, is the same as that of S_1.

Now let us look at models of the "coin domain" scenarios.

Example 2.4 (*Flipping a Coin (Models)*)
It is easy to see that any model of scenario

$$S_0 = \langle obs(agreed\_to\_play(0))\rangle$$

must contain a random experiment $random(flip(0), head(1))$ (which we shorten to $flip(0)$). According to the semantics of P-log the experiment generates two possible outcomes $head(1)$ and $\neg head(1)$. Thus, scenario S_0 has two models: $M_1 = \{agreed\_to\_play(0), flip(0), head(1)\}$ and $M_2 = \{agreed\_to\_play(0), flip(0), \neg head(1)\}$.
M_1 is the only model of scenario $S_1 = \langle agreed\_to\_play(0), obs(head(1))\rangle$. The model M_2 of scenario

$$S_2 = \langle agreed\_to\_play(0), \mathbf{do}(flip(0), true)\rangle$$

is $M_1 \cup \{do(flip(0), head(1), true)\}$.

It is worth noting that causal theories and their entailment relation allows us to answer many interesting queries which are usually answered using structural equations or other similar formalisms. Consider, for instance, several questions related to the Firing Squad Example which are used in (Pearl and Mackenzie 2018) to illustrate several non-trivial types of causal reasoning. First the authors want to know *"if the prisoner would be dead after the court order was given but not otherwise"*. In the language of causal theories to answer the first part of this question we simply check if there is i such that $FS(S_1) \models dead(i)$. The answer, of course, is *yes*. The second part of the question can be understood as a request to show that if the court order is not given, the prisoner will be alive. This may refer to scenario $S_0 = \langle obs(\neg dead(0))\rangle$ or $\mathscr{S}_0 = S_0 \cup \{\neg court\_order(0)\}$. Clearly, in both cases the answer would be *yes*. Next the authors ask: "Will the prisoner be dead or alive if the guard "a" decides to fire on his own initiative, without waiting for the captains command?". This may correspond to scenario $\mathscr{S}_1 = S_0 \cup \{shoot(a,2)\}$. Clearly, $FS(\mathscr{S}_1) \models dead(3)$, i.e., the answer is *yes*. As pointed out by Pearl, the follow up question "Did guard "b" shoot too?" should be answered by the *no*, which is indeed the case in our

approach. Note, that this is rather different from scenario $\mathscr{S}_2 = S_0 \cup \{obs(shoot(a,2))\}$. Given this scenario we will conclude that shooting happen because of the court order and hence guard b did shoot. This is a difference between observing and doing which Pearl and many others (including us) believe to be very important for understanding and modeling causation.

Another interesting question from (Pearl and Mackenzie 2018) is: "Supposing that the court order was given, what would happen if the guard "a" had decided not to shoot?" Since scenario $S_1 = \langle obs(\neg dead(0)), court\_order(0)\rangle$ entails $shoot(a,2)$ the question can be viewed as having a counterfactual character. It can be answered by considering scenario $\mathscr{S}_2 = S_1 \cup \{\neg shoot(a,2)\}$ and extracting actions from its unique model. One can see that the captain will give the order, guard "b" will shoot, but the mechanism $m_2(a,2)$ causing "a" to follow the captain's order and shoot will be defeated by a's action. Clearly, the prisoner will be dead anyway because of $m_3(b,2)$. If both guards were to refuse the order, the prisoner would remain alive.

This concludes our description of casual theories and their scenarios. Now we are ready to discuss the notion of cause.

3 What is a Cause?

There are several reasons which contribute to the difficulty of defining the meaning of the verb "causes". This includes variety of uses of this word in natural language, the ambiguity of the meaning of a given text, and the dependency of identifying causes on the text's formal representation. When different representations correspond to different disambiguations of the text the latter is a good thing. But sometimes the differences are caused by not fully accurate representations or by idiosyncrasies or insufficient expressiveness of a particular formalism. We attempt to alleviate these problems by starting with intuitive understanding of the notion we are trying to formalize, using a powerful KR language, employing general methodology in formalizing particular examples, and by discussing consequences of some of our representational choices. Our framework is based on the following **Basic Assumption**:

> *1. We assume that an agent is supplied with a (fixed) background theory T and a scenario S with the last time step n, which consist of observations and the complete collection of deliberate actions which occurred in the domain up to that time.*
> *2. In addition, we assume that every uninterrupted random experiment of a scenario used for determining causes is immediately followed by the observation of its outcome.*

In what follows we introduce three different types of actual cause: *deliberate*, *accidental* and *exogenous*. The best explanation of an event will be given by finding its deliberate cause. If this is impossible we attempt to find causes which include accidental (i.e. not deliberate) actions explicitly mentioned in the domain. As the last resort we allow causes containing unknown exogenous actions.

Deliberate Cause: Our definition of a deliberate cause can be viewed as a formalization of the following intuition:

"*A cause of $e(\bar{x},k)=y$ is a deliberate action event α which initiates a chain of events bringing about $e(\bar{x},k)=y$. Moreover, α must be in some sense minimal, i.e. no parts of α can be removed without loss of causal information about $e(\bar{x},k)=y$.*"

An expression "chain of events" from our informal description will be modeled by notion of proof. Let Π be a consistent regular P-log program, M be a set of ground literals, and $M^- = \{\text{not } l : l \text{ is not satisfied by } M\}$.

Definition 3.1 (*Proof*)

- A sequence $P = \langle r_0, l_0, r_1, l_1 \ldots, r_n, l_n\rangle$ where ls are literals from M and rs are rules or names of random experiments of program Π is called a *proof of l_n in M from Π ($M, \Pi \vdash l_n$)* if
  - — For every i, $body(r_i)$ is satisfied by $\{l_0, \ldots, l_{i-1}\} \cup M^-$,
  - — l_i is the head of r_i or
  - — l_i is $e(k)=y$ and l_{i-1} is $random(n, e(k), p)$ representing a non-interrupted random experiment (i.e., there is no y such that $do(n, e(k)=y)$ is in S), r_i is n, and $p(y) \in \{l_0, \ldots, l_{i-1}\}$
  - — No proper sub-sequence of P satisfies the above conditions.

  Note that in P-log facts are viewed as rules with the empty bodies. If M is a possible world of Π we simply say that P is a proof of l_n in M;
- A scenario S of background theory T *derives* event $e(k)=y$ in a model M of S if there is a proof of $e(k)=y$ in M from $T(S)$; S *derives* $e(k)=y$ if S derives $e(k)=y$ in every model of S.

The idea of a deliberate (or intentional) action is formalized as follows.

Definition 3.2 (*Deliberate Action*)
Let S' be the result of removing action event $a(i)=y$ from a scenario S of T. We say that $a(i)=y$ is *deliberate* in $T(S)$ if no model of S' contains $a(i)=y$.

As an example consider background theories $T_1 = \{f(1) \leftarrow a(0)\}$ and $T_2 = T_1 \cup \{a(0) \leftarrow \mathit{not}\ \neg a(0)\}$ and scenario $S = \langle a(0)\rangle$. It is easy to check that $a(0)$ is deliberate in $T_1(S)$ but not deliberate in $T_2(S)$.
To define a cause of an event $e(k)=y$ in a scenario S of T we introduce a notion of the event's *inflection point - a time-step $i \le k$ such that $i-1$ is the last time-step of S in which the value of $e(k)$ is not predicted to be y.* To make this precise we need some notation: Let $S[i]$ consists of observations of S made at points not exceeding i and all actions S which occurred at steps not exceeding $i-1$, i.e.

$$S[i] =_{def} \{obs(f(j)=y) \in S : j \le i\} \cup \{a(j)=y \in S : j < i\}$$

Definition 3.3 (*Inflection Point*)
Step i is the *inflection point* for $e(k)=y$ in S if

- $T(S[i-1])$ does not derive $e(k)=y$ and
- for every $j \in [i,k]$, $T(S[j])$ derives $e(k)=y$.

Now we are ready for the main definition. Let R be a collection of cr-rules of program T. By T^R we denote from T by dropping all cr-rules not in R and turning cr-rules from R into regular rules. For precise definition see the appendix.

Definition 3.4 (*Deliberate Cause*)
Let S be a scenario of T, $obs(e(k) = y) \in S$, M be a model of S generated by a (possibly empty) abductive support R, and i be the inflection point of $e(k) = y$ in scenario S of T^R. A non-empty set α of actions from S is called *a cause* of $e(k) = y$ in M (with respect to $T(S)$) if

(a) $T^R(S[i-1] \cup \alpha)$ derives $e(k) = y$ in M,
(b) For every $\beta \subset \alpha$ there is a proof of $e(k) = y$ from $T^R(S[i-1] \cup \alpha)$ in M which is not a proof of $e(k) = y$ from $T^R(S[i-1] \cup \beta)$ in M.

We say that α *is a possible cause of* $e(k) = y$ *in* S if it is a cause of $e(k) = y$ in some model of S. Finally, α *causes* $e(k) = y$ *in* S if it causes $e(k) = y$ in all models of S.

To illustrate the definition consider scenario S_1 from Example 2.3. Clearly, the inflection point for $dead(3)$ in $S_1 \cup \{obs(dead(3))\}$ is 1. It's cause is $court\_order(0)$ with the proof consisting of applications of causal mechanisms of FS. The same results hold for the value of *dead* at, say, point 4.

Let us now look at a theory T with action a, inertial fluent f and causal law

$$f(T) \leftarrow a(T-1)$$

and at scenario

$$S = \langle obs(\neg f(0)), random(m(0), a(1)), obs(a(1)) \rangle.$$

Clearly $M = \{\neg f(0), m(0), a(1), \neg f(1), f(2)\}$ is the only model of S. The only deliberate action $random(m(0), a(1))$ of S is the only cause of $f(2)$ in M. To bring our terminology closer to the one used in natural language we say that $f(2)$ *is caused by the outcome* $a(1)$ *of random experiment* $m(0)$. *This reading will be used throughout the paper.* Next consider theory T with inertial fluent f and actions a and b consisting of a strict causal law

$$(1) \quad f(1) \leftarrow b(0)$$

and a cr-rule

$$(2) \quad b(0) \stackrel{+}{\leftarrow} a(0).$$

A scenario

$$S = \{\neg f(0), a(0), obs(f(1))\}$$

of T has unique model $M = \{\neg f(0), a(0), b(0), f(1)\}$ generated by abductive support R consisting of rule (2). To find a cause of $f(1)$ in M we need first to find the inflection point of $f(1)$ in scenario S of T^R. Clearly $S[0]$ does not contain $a(0)$ and hence $T^R \cup S[0]$ does not derive $f(1)$. But, for every $i > 0$, $T^R \cup S[i]$ does. Hence, the inflection point of $f(1)$ is 1. Now one can easily check that in M (and hence in S), $f(1)$ is caused by $a(0)$.

The next example explains why the second condition of Definition 3.4 cannot be replaced by subset minimality. This is an adaptation of Example from page 31 in (LeBlanc 2019).

Example 3.1 (*LeBlanc*)
Consider a background theory T

1. $d_1(1) \leftarrow e_1(0)$
2. $d_2(1) \leftarrow e_1(0)$
3. $d_3(1) \leftarrow e_2(0)$

$$4.\ l(I) \leftarrow d_1(I)$$

$$5.\ l(I) \leftarrow d_2(I), d_3(I)$$

For $i \in \{1,2\}$

$$6.\neg e_i(0) \leftarrow \text{not } e_i(0)$$

$$7.e_i(0) :+ .$$

where l is a fluent defined in terms of inertial fluents d_1, d_2, d_3 and e_1 and e_2 are actions. Let $Init = \{\neg d_1(0), \neg d_2(0), \neg d_3(0), \neg l(0)\}$ and consider scenario

$$S_0 = Init \cup \{e_1(0), e_2(0)\}.$$

The unique model, M, of S is $Init \cup \{e_1(0), e_2(0), d_1(1), d_2(1), d_3(1), l(1)\}$. The inflection point for $l(1)$ in S_0 is 1. Clearly, $\beta = \{e_1(0)\}$ is a possible cause of $l(1)$; the corresponding causal chain goes from $e_1(0)$ to $l(1)$ using rules (1) and (4). Note, that $e_1(0)$ would also be a cause of l_1 if we were to replace (b) by the subset minimality condition. But, as pointed out by Le Blanc, it seems that $\alpha = \{e_1(0), e_2(0)\}$ also should be a possible cause of $l(1)$ - due to the presence of $e_2(0)$ there is a causal chain from α to $l(1)$ which uses rules (2), (3), and (5). Hence dropping $e_2(0)$ will result in loosing some causal information about $l(1)$ which does not match with our informal semantics. Under standard minimality condition, however, α will not be a case of $l(1)$. But, according to our definition, both β and α are causes of $l(1)$.

Causes Containing Accidental Actions. To see the need for causes with accidental actions consider scenario

$$S_4 = \{obs(\neg dead(0)), obs(dead(3))\}$$

from Example 2.3. Since S_4 contains no deliberate actions, $dead(3)$ has no deliberate cause. One can, however, argue that action $court\_order(0)$ should be a cause of $dead(3)$ in the unique model M of S_4 even though it is not deliberate. This intuition can be justified by the following informal principle:

> *Execution of a non-deliberate (accidental) action is a part of an event's cause if it's deliberate execution is.*

One should, however, be careful in formalizing this intuition. An attempt to make an action deliberate by simply adding it to the original scenario does not work. Since *court_order* is derived in the model of S_0, it is not deliberate in scenario $S_5 = S_4 \cup \{court\_order(0)\}$ and hence our previous definition is not applicable.

One way to avoid the difficulty is to remove from FS consistency-restoring rule (4) (which allows to derive $court\_order(0)$ in our scenario) and consider S_5 to be a scenario of a new theory FS^*; $court\_order(0)$ is deliberate in $FS^*(S_5)$. After this "surgery", somewhat reminiscent of Pearl's surgery on causal networks, $dead(3)$ would indeed be caused by $court\_order(0)$. Here is another example.
Consider theory T_0 with action a, fluent f, causal law

$$(1)\quad f(T) \leftarrow a(T-1)$$

and default

$$(2) \quad a(T) \leftarrow \text{not } ab(T-1)$$

together with a scenario

$$S_0 = \langle obs(\neg f(0)), obs(f(2)) \rangle$$

of T_0. Even though there is no deliberate cause of $f(2)$, one may legitimately say that it is caused by $a(1)$. As in the previous example we may model our informal counterfactual reasoning by considering scenario

$$S_1 = \langle obs(\neg f(0)), a(1), obs(f(2)) \rangle$$

of a new theory T_0^* obtained from T_0 by removing default (2) (which would, otherwise, automatically trigger $a(0)$). Note, that action $a(1)$ would be deliberate in scenario S_1 of T_0^*. One can check that $a(1)$ is a cause of $f(2)$ in S_1, and hence, by our informal principle, it is also a cause of $f(2)$ in S_0.

To correctly deal with this counterfactual argument in scenarios containing random events we need another small operation. The new scenario, obtained from the original one by adding to it a random experiment, should also be extended by the observation of the outcome of this experiment taken from the corresponding model. To illustrate the problem and the solution let us replace the default (2) in T_0 by

$$random(m(0), a(1)) \leftarrow \text{not } ab(0).$$

In this case it is not sufficient to remove the default from the new theory T_1 and expand S_0 by $random(m(0), a(1))$. The resulting scenario will not satisfy our Basic Assumption. To avoid that, S_0 should also be extended by the required observation of the outcome of this experiment - in our case $obs(a(1))$. Note that action $m(0)$ is deliberate in scenario

$$S_0^* = \langle obs(\neg f(0)), random(m(0), a(1)), obs(a(1)), obs(f(2)) \rangle$$

of theory T_1^* consisting of rule (1) of T_0. Moreover, in this scenario the outcome $a(1)$ of $m(0)$ is the only cause of $f(2)$. According to the intuitive principle above, $a(1)$ can be viewed as accidental cause of $f(2)$ in scenario S_0 of T_1.

These observations lead to the following definition. Let M be a model of a scenario S of T, E be an action atom accidental in $T(S)$, $c(E,M)$ be E if E is regular, and $random(m,f)$ followed by the outcome $f = y$ of m in M if E is $random(m,f)$. If α is an action event then $c(\alpha, M) =_{def} \{c(E,M) : E \in \alpha\}$. We say that a rule r *generates* the value of term $a(i)$ if $head(r)$ is $a(i) = y$ for some y or is of the form $random(m, a(i), p)$. By $surg(T, \alpha)$ we denote a theory obtained from T as follows: For every $a(i)$ such that α contains $random(m, a(i), p)$ or $a(i) = y$ for some y remove from T every rule generating the value of $a(i)$.

Definition 3.5 (*Causes Containing Accidental Actions*)
Given a model M of scenario S of T and action event $\alpha \subset M$ we say that α *is an accidental cause of* $e(k) = y$ *in* M *with respect to* $T(S)$ if there is an action event $\beta \subset M$ not deliberate in S such that

- M is a model of scenario $S^* =_{def} S \cup c(\beta, M)$ of $T_1 = surg(T, \beta)$. (Note that, by construction, all actions of S^* are deliberate).
- α is a deliberate cause of $e(k) = y$ in M with respect to $T_1(S^*)$.

Let us apply this definition to the unique model M of scenario S_0 of theory T_0 from the above example and show that, in this model, $f(2)$ is caused by $\alpha = \{a(1)\}$. Let $\beta = \alpha$. Then $S^* = S_0 \cup \{a(1)\}$ and $surg(T_0, \beta)$ consists of rule (1) of T_0. Clearly M is the model of scenario S^* of $surg(T_0, \beta)$. All actions of this scenario are deliberate. It is easy to check that α is indeed the deliberate cause of $f(2)$ in M.
Let us consider scenario S_0 over theory T_1 from the same example. Its unique model M is $\{\neg f(0), random(m(0), a(1)), \neg f(1), a(1), f(2)\}$. To show that the only cause of $f(2)$ in M is the outcome $a(1)$ of random experiment $m(0)$ let β be $random(m(0), a(1))$. The corresponding $S^* =_{def} S_0 \cup \{random(m(0), a(1)), obs(a(1))\}$ and $surg(T_0, \beta)$ consists of rule (1) of T_0. All conditions of Definition 3.5 are clearly satisfied.

Next consider scenario $S_4 = \langle obs(\neg dead(0)), obs(dead(3))\rangle$ from Example 2.3 and show that the cause of $dead(3)$ in model M of S_4 is indeed $court\_order(0)$. The model contains $court\_order(0)$ included in it by cr-rule (4) of theory FS. Let $\alpha = \beta = \{court\_order(0)\}$. it is easy to check that $court\_order(0)$ is a deliberate cause of $dead(3)$ in scenario

$$S_4^* = S_4 \cup \{court\_order(0)\}$$

of $surg(FS, \beta)$ obtained from FS by removing rule (4). Note, that $captain\_order(1)$ will not be a cause of $dead(3)$ in M since M is not a model of $S_4 \cup \{captain\_order(1)\}$.

Next, consider scenario S_3 from Example 2.3. As was shown there it has two models: M_1 in which guards refuse to shoot and M_2 in which captain does not follow his order. One can check that refusal to shoot is the cause of $\neg dead(3)$ in M_1. In M_2 the cause is $\neg captain\_order(1)$.

To see how the definition works for random events consider the unique model M_1 of scenario

$$S_1 = \langle obs(agreed\_to\_play(0)), obs(head(1))\rangle$$

from Example 2.4. The inflection point for $head(1)$ in S_1 is 1. The scenario contains no deliberate actions and hence $head(1)$ has no deliberate cause. It is, however, not difficult to show that, according to the Definition 3.5, $head(1)$ is caused by a random experiment $flip(1)$. In the model M_2 of S_2. however, $head(1)$ is the result of deliberate intervening action $do(head(1), true)$. The difference between Pearl's actions and observations is important not only for computing probability of events but also for discovering their causes.

Finally, let us revisit background theory T from Example 3.1 and expand the original example by considering a scenario

$$S_1 = Init \cup obs(l(1)).$$

It is natural to expect that causes of $l(1)$ in this scenario and scenario $S_0 = Init \cup \{e_1(0), e_2(0)\}$ will coincide. But this may or may not be the case, depending on preference relation between sets of cr-rules. To construct a model $T(S_1)$ one need to explain unexpected observation of $l(1)$. A subset-minimal preference relation allows us to find only one such explanation – $\{e_1(0)\}$. In the unique model of this program (and hence in S_1) $\{e_1(0)\}$ will be the only cause of $l(1)$. Another possible cause will be missing. The relevance based preference gives a different result. Since $l(1)$ depends on $e_1(0)$ and on $e_2(0)$, under this preference $T(S_1)$ will have two models. One containing $e_1(0)$ and another containing $e_1(0)$ and $e_2(0)$. Hence both, $\{e_1(0)\}$ and $\{e_1(0), e_2(0)\}$, will be possible causes of $l(1)$.

In examples we consider so far β from Definition 3.5 was equal to α. It is not always the case.

Consider background theory T consisting of causal law

$$f(1) \leftarrow a_1(0), a_2(0)$$

where f is an inertial fluent and a_1 and a_2 are actions, and cr-rule

$$a_2(0) \stackrel{+}{\leftarrow} .$$

It is easy to see that scenario

$$S = \langle \neg f(0), a_1(0), obs(f(1)) \rangle$$

of T has unique model M in which unexpected observation $f(1)$ is explained by $a_2(0)$. To show that $\alpha = \{a_1(0), a_2(0)\}$ is an accidental cause of $f(1)$ in M we need to select $\beta = \{a_2(0)\}$.

So far, we have been looking for causes of events occurring in a model M among actions from M. Now we slightly modify this approach by allowing to search for causes among a specific type of actions not present in M.

Causes Containing Exogenous Actions.

> If an event has no cause consisting of deliberate and accidental actions then it may be useful to admit causes containing unknown, exogenous actions responsible for bringing about events in the initial state of the scenario and/or abnormality relations in rules of the theory.

This can be done in two steps. First, we increase time-steps in S (and, if necessary, in T) to have a new initial state with no actions and undefined fluents. Denote the new scenario by S^*. For every fluent f such that, for some y, $obs(f(\bar{x}, 1) = y) \in S^*$ and for every $ab(m(i)) \in M$ we expand T by causal laws

$$f(\bar{x}, 1) = y \leftarrow brought\_about(f(\bar{x}, 1) = y, 0)$$

$$ab(m(i)) \leftarrow brought\_about(ab(m(i)), i-1)$$

where *brought_about* is a new action. Denote the result by T^* and the set of all actions formed by *brought_about* by E.

Definition 3.6 (*Causes Containing Exogenous Actions*)
Given a model M of scenario S of T, an action event $\alpha \subset M \cup E$, and an event $e(k) = y$ in M which has no cause consisting of actions from M we say that α is *an exogenous cause of* $e(k) = y$ *in* M if there is a model M^* of $T^*(S^* \cup \alpha)$ such that

- For every atom $u(i) \notin E$, $u(i) \in M$ if and only if $u(i+1) \in M^*$,
- α is a (deliberate or accidental) cause of $e(k+1) = y$ in M^*.

If $e(k) = y$ is caused by $brought\_about(f(\bar{x}) = y, 0)$ we often abuse the language and simply say that $e(k) = y$ is caused by $f(\bar{x}, 0) = y$. Similarly for *ab*.

To illustrate the definition let us go back to the Firing Squad example 2.3 and consider the only model M_0 of scenario

$$S_0 = \langle obs(\neg dead(0)), obs(\neg dead(3)) \rangle$$

and show that $a(0) = \{brought\_about(\neg dead(1),0)\}$ is the cause of $\neg dead(3)$ in S_0. To do that we need to show that $a(1)$ is a (deliberate) cause of $\neg dead(3)$ in scenario

$$S = \langle a(0), obs(\neg dead(1)), obs(\neg dead(4))\rangle$$

of FS^*. The inflection point here is 1. Clearly there is a proof of $\neg dead(3)$ from $FS^*(S[0] \cup a(0))$ which uses axiom $\neg dead(1) \leftarrow brought\_about(\neg dead(1),0)$ and the inertia, and no proof of $\neg dead(3)$ from $S[0]$. Thus, conditions of Definition 3.4 are satisfied.

[p.s. If instead of inertia we had some form of CWA for $\neg dead(I)$ then $\neg dead(3)$ would still have no cause.]

Similarly, given a theory T

$$m(T) \ : \ f(T) \leftarrow \text{not } ab(m(T-1))$$

$$ab(m(T)) \stackrel{+}{\leftarrow}$$

$$\neg f(T) \leftarrow ab(m(T))$$

and scenario

$$S = \langle obs(\neg f(1))\rangle$$

the cause of $\neg f(1)$ in the only model M of S is the exogenous action which brought about $ab(m(0))$.

Let us now consider scenario

$$S_2 = \langle obs(\neg dead(0)), court\_order(0), \neg shoot(a,2), \neg shoot(b,2)\rangle$$

from Example 2.3. In the only model of S_2, $\neg dead(3)$ is true by inertia and hence its only cause is an exogenous action which brought about $\neg dead(0)$. Intuitively, however, action event $\{\neg shoot(a,2), \neg shoot(b,2)\}$ is in some way responsible for the prisoner being alive at the end of the story since it *prevented* prisoner's death, which otherwise would have happened. This is captured by the following definition.

Definition 3.7 (*Event Prevention*)
Let S be a scenario of T and i be a time step such that for some $y_1 \neq y_2$

- $T(s[i-1])$ derives $e(k) = y_1$
- $T(s[i])$ derives $e(k) = y_2$.

We say that an action event $\alpha(i-1)$ from S *prevents* $e(k) = y_1$ from being true in S if

(a) $T(s[i-1] \cup \alpha(i-1))$ derives $e(k) = y_2$
(b) There is no $\beta(i-1) \subset \alpha(i-1)$ such that $T(s[i-1] \cup \beta(i-1))$ derives $e(k) = y_2$

It is not difficult to check that $\{\neg shoot(a,2), \neg shoot(b,2)\}$ indeed prevented $dead(3)$ from being true in S_2. We plan to further investigate the notion of prevention in follow-ups to this paper.

4 Examples

To further illustrate the above definitions, methodology of representing causal knowledge, and the differences between our approach and several others, we discuss a number of examples. To

save space, we often omit names of casual mechanisms, abnormality predicates, and some common axioms. Our first example, (often used to highlight difficulties with counterfactual approach to causation (see, for instance (Halpern and Hitchcock 2014)) deals with so called "late preemption".

Example 4.1 (*Breaking the Bottle*)
Suzy and Billy both throw rocks at a bottle. Suzy's rock arrives first and shatters the bottle. Billy's arrives second and so does not shatter the bottle. Both throws are accurate: Billy's would have shattered the bottle if Suzy's had not.

The background theory, TS of the story, with steps ranging from 0 to 2 contains actions $throw(suzy)$ and $throw(bill)$ with static attributes $duration(suzy) = 1$ and $duration(bill) = 2$ for durations of agents' throws, causal mechanism $m(A,T_1)$

$$shattered(T_2) \leftarrow throw(A,T_1), duration(A) = D, T_2 = T_1 + D, \neg shattered(T_2 - 1), \text{not } ab(m(A,T_1))$$

determining the effect of throwing, contingency axiom for this rule, CWA for actions and the inertia axiom for $shattered$. Consider scenario

$$S_0 = \langle obs(\neg shattered(0)), throw(suzy,0), throw(bill,0), obs(\neg shattered(1))\rangle.$$

The only model M_0 of S_0 is:

$$\neg shattered(0), throw(suzy,0), throw(bill,0), shattered(1), shattered(2)$$

Step 1 is the inflection point for $shattered(1)$ and $shattered(2)$ in M_0 and their only cause is $throw(suzy,0)$. Now consider

$$S_1 = S_0 \cup \{obs(\neg shattered(1))\}$$

and its model M_1:

$$\neg shattered(0), throw(suzy,0), throw(bill,0), ab(m(suzy,0)), \neg shattered(1), shattered(2)$$

(Since the contingency axiom for $ab(m(suzy,0))$ is the only cr-rule the observation $\neg shattered(1)$ in S_1 depends upon, this is the only model of S_1 for every one of our preference relations.)

Step 1 is still the inflection point for $shattered(2)$ in M_1. It is now caused by $throw(bill,0)$.

By Definition 3.6 $\neg shattered(1) \in M_1$ is caused by an exogenous action which brought about $\neg shattered(0)$.

Next consider

$$S_2 = \langle obs(\neg shattered(0)), throw(bill,0), throw(suzy,1), obs(shattered(2))\rangle.$$

It is easy to check that in its only model $shattered(2)$ has two causes: $throw(bill,0)$ and $throw(suzy,1)$.

If we modify causal mechanism $m(A,T)$ by assuming that the bottle get shattered only if it is hit simultaneously by both stones, and use the new theory together with S_2, the timing of throw would become important. Even though both stones arrive simultaneously the shattering of the bottle would be caused by the latest throw. This does not contradict our intuition. However, if Bill and Suzy were to agree that Suzy's throw should follow that of Bill, i.e., if our theory contained a rule $throw(suzy,I+1) \leftarrow throw(bill,I)$, then shattering of the bottle would be caused by Billy.

Here is another example, taken from (Hall 2004).

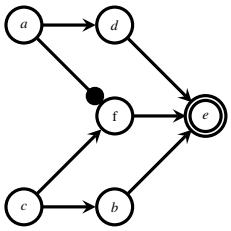

Fig. 1

Example 4.2 (*Hall's Neuron Net*)
Consider a neuron net from figure 1. *If a link from neuron n_1 to neuron n_2 is ended by an arrow, then n_1 stimulates n_2; if it is ended by a bullet, then n_1 inhibits n_2; e is a "stubborn" neuron, requiring two stimulatory signals to fire. For other neurons one stimulating signal is sufficient.*

A background theory *NN* for this example will have sorts for neurons, an action $stim(S)$ which stimulates neurons from the set S, Boolean fluents *stimulated* and *inhibited*, and statics *link* and *stubborn*. The net will be represented by a collection of atoms $link(a,d,stm)$, $link(a,f,inh)$, etc., where $link(X,Y,stm)$ / $link(X,Y,inh)$ indicates that X stimulates/inhibits Y, and facts $stubborn(e) \cup \{\neg stubborn(N) : N \neq e\}$. We will need two time steps: 0 and 1 with 0 being used for the execution of actions and 1 for their effects, and two inputs, s_1 and s_2 of action *stim* defined by statics:

$member(c,s_1)$, $member(c,s_2)$, $member(a,s_2)$.

The causal mechanisms of *NN* are

$$[m_0(X,S)] : \ stimulated(X,1) \leftarrow stim(S,0), member(X,S)$$

$$[m_1(X,Y)] : stimulated(Y,1) \quad \leftarrow \quad \neg stubborn(Y), \neg inhibited(Y,1), \\ link(X,Y,stm), stimulated(X,1)$$

$$[m_2(Y)] : stimulated(Y,1) \quad \leftarrow \quad stubborn(Y), \neg inhibited(Y,1), \\ card\{X : link(X,Y,stm), stimulated(X,1)\} > 1$$

$$[m_3(X,Y)] \ : \ inhibited(Y,1) \leftarrow link(X,Y,inh), stimulated(X,1)$$

We assume that all neuron directly stimulated by *stim* are included in its parameter S, which eliminates the possibility of parallel *stim* actions, i.e. we have

$$\neg stim(S_1,I) \leftarrow stim(S_2,I), S_1 \neq S_2$$

Finally, we need the inertia axiom for the fluents, and CWA with indirect exceptions for action *stim*:

$$\neg stim(S,0) \leftarrow \text{not } stim(S,0)$$

$$stim(S,0) :+$$

Let us consider *NN* together with a scenario

$$S_0 = init \cup \{obs(stimulated(e,1))\}$$

where $init = \{obs(\neg stimulated(X,0)), obs(\neg inhibited(X,0)) : neuron(X)\}$. The regular part of $NN(S_0)$ is inconsistent. There are two ways to restore consistency using abductive supports

$\{stim(s_1)\}$ or $\{stim(s_2)\}$. Note that $\{stim(s_1), stim(s_2)\}$ is not an abductive support of the program. This is obvious for standard minimality based preferences. This is also the case for the relevance based preference since parallel execution of *stim* is prohibited and therefore both cr-rules cannot be activated simultaneously. Thus S_0 has two models – M_1 containing $stim(s_1,0)$ in which e is stimulated via neurons c, f, and b and M_2 which contains $stim(s_2,0)$. In M_2 neuron f is inhibited and e is stimulated via neurons a, c, d, and b. Clearly, in the first model $stimulated(e,1)$ is caused by $stim(s_1,0)$ and in the second by $stim(s_2,0)$. Hence in S_0, $stimulated(e,1)$ has two possible causes.

One can argue that $stim(s_2,0)$ shall not be a possible cause of $stimulated(e,1)$ in S_0 since there is a better "minimally sufficient" candidate $stim(s_1,0)$. Indeed, since $s_1 \subset s_2$, action $stim(s_1,0)$ is simpler than $stim(s_2,0)$ but we believe that this does not preclude $stim(s_2,0)$ from being viewed as a valid possible cause of $stimulated(e,1)$ in S_0. This seems to agree with the Hall's view.

It is also worth noting that our formalization of the story avoids parallel actions. Instead, we consider actions whose parameters are sets of neurons. We believe this to be a reasonable formalization, but it also allows to avoid differences in determination of causes based on varying preference relations of CR-Prolog. Suppose that, instead defining *stim* on sets of neurons we define it on input neurons a and c. The new theory, with obvious causal laws, will be denoted by NN_1. Then consistency of $NN_1(S_0)$ will be restored by abductive support $\{stim(c,0)\}$ independently of preference relation used in the program. However, under relevance based preference the program will have one more abductive support – $\{stim(c,0), stim(a,0)\}$. This means that under standard minimality preference $stimulated(e,1)$ will have only one model and only one cause – $\{stim(c)\}$. Under the relevance based preference there will be two models. In one, $\{stim(c)\}$ will be the only cause. In another both, $\{stim(c)\}$ and $\{stim(a), stim(c)\}$ will be possible causes.

The next example appears in multiple papers. Our version is taken from (Beckers and Vennekens 2012).

Example 4.3 (*Switch*)
An engineer is standing by a switch in the railroad track. A train approaches in the distance. She flips the switch, so that the train travels down the left-hand track, instead of the right. Since the tracks re-converge up ahead, the train arrives at its destination all the same.

This story is slightly more complicated and its causal analysis will depend substantially on parts of the story abstracted out in the process of its formalization (which may explain different views on what constitute proper cause for the train's arrival at its destination in some scenarios). So we use this example to illustrate our step-wise methodology of story formalization and consider several, increasingly more detailed, versions of the story. In a nutshell, the train story consists of two sentences: *"a train appeared"* and *"the train arrived at its destination"*. We can represent this by introducing a sort *loc* for locations and a sort *move* for actions of the type "move". The former will be initially populated by s and f denoting the initial and the final positions respectively. The actions from *move* may have attributes - *source*, *dest*, and *route*. Fluent $pos : loc$ indicates position of the train on a given time-step. For every action $e \in move$ we have a causal law

$$m_0(e, I-1): \quad pos(I) = L \quad \leftarrow \quad \begin{array}{l} e(I-1), \\ dest(e) = L, \\ \text{not } ab(m_0(e, I-1)) \end{array}$$

For move e to be successful its source and destination must be connected. So we add static $connected(s,f)$ and constraint

$$\begin{array}{rl} \leftarrow & source(e) = L_1, \\ & dest(e) = L_2, \\ & \text{not } connected(L_1, L_2) \end{array}$$

We also need CWA

$$\neg e(I) \leftarrow not\ e(I)$$

$$e(I) \stackrel{+}{\leftarrow}$$

We denote these axioms, together with contingency axioms for m_0, and inertia axioms by T_0. Let a with $source = s$ and $dest = f$ be an action from *move*. It is easy to check that $a(0)$ is the (deliberate) cause of $pos(2) = f$ in scenario

$$S_0 = \langle obs(pos(0) = s), a(0), obs(pos(2) = f) \rangle$$

of T_0. This action will also be the cause of $pos(2) = f$ in scenario

$$S_1 = \langle obs(pos(0) = s), obs(pos(2) = f) \rangle$$

This conclusion follows from the contingency axiom for CWA for action a and holds for every one of our preference relations. This time the cause is accidental.

The engineer's action $flip$ can be included in the story by introducing one more location, say $fork$, and new sort *routes* initially populated by two routes, $left$ and *right*, leading from $fork$ to the final destination f. The availability of the route is controlled by action $flip$, via the following causal law:

$$\begin{array}{rlrl} m_1(I-1): & available(I) = left & \leftarrow & flip(I-1), \\ & & & available(I-1) = right, \\ & & & \text{not } ab(m_1(I-1)) \end{array}$$

where $available : \{left, right\}$ is an inertial fluent. Similarly, for switching from $left$ to *right*. Let us denote expansion of T_0 by this law and the corresponding standard axioms (including the CWA with indirect exceptions for $flip$) by T_1. Consider scenario

$$S_2 = S_0 \cup \{obs(available(0) = right), flip(1), obs(available(2) = left)\}$$

One can check that the cause of $pos(2) = f$ in $T_1(S_0)$ is still $a(0)$. But, in addition, we can now inquire about the cause of $available(2) = left$. Clearly the answer would be $flip(1)$.

So far, we abstracted out features of the story which could have allowed us to ask questions about the path the train followed to arrive at its destination. To include the relevant information we expand sort *move* by new actions: a_0 with $source(a_0) = s$ and $dest(a_0) = fork$, a_1 with $source(a_1) = fork$, $dest(a_1) = f$, $route(a_1) = left$ and a_2 with $source(a_2) = fork$, $dest(a_2) = f$, $route(a_2) = right$. Action a can now be viewed as a *complex* action, i.e., a sequence consisting of two consecutive actions - moving from s to $fork$ and from $fork$ to the destination via the available route. This can be formalized by removing a from the sort *move* which only consists of simple actions (not sequences of actions); a will still belong to sort *actions*. This will remove axiom $m_0(a,0)$ from our theory but still allow $a(0)$ to be viewed as a deliberate action in scenario S_0. Next axioms define actions triggered by $a(I)$:

$$a_0(I) \leftarrow a(I)$$

$$a_1(I) \leftarrow a_0(I-1), available(I) = left$$

$$a_2(I) \leftarrow a_0(I-1), available(I) = right$$

A new Boolean fluent $arrived\_by(D,R)$ holds when the train arrives at its destination D by route R. For every action $e \in move$, location d and route r the new theory will include causal law connecting this fluent with e:

$$m_2(I-1): \quad arrived\_by(d,r,I) \quad \leftarrow \quad \begin{array}{l} e(I-1), \\ dest(e) = d, \\ route(e) = r, \\ \text{not } ab(m_2(I-1)) \end{array}$$

The expansion of T_1 by these axioms together with standard axioms for new laws, fluents and actions will be denoted by T_2. Let $arr = \{\neg arrived\_by(d,r,0) : d \in loc \text{ and } r \in routes\}$ and consider scenario

$$S_3 = S_0 \cup arr \cup \{obs(available(0) = right), obs(arrived\_by(f, right, 2))\}$$

One can check that the cause of $arrived\_by(f, right, 2)$ in scenario S_3 is $a(0)$. (Note, that $a_2(1)$ is not a cause since it is not a deliberate action). The cause of $arrived\_by(f, left, 2)$ in scenario

$$S_4 = S_0 \cup arr \cup \{obs(available(0) = right), obs(arrived\_by(f, left, 2))\}$$

is $\{a(0), flip(0)\}$. The latter is derived by abductive reasoning which resolves the contradiction with the last observation by assuming that $flip(0)$ is an indirect exception to CWA for $flip$. Note that the cause of $pos(2) = f$ is still $a(0)$. Even though $flip(0)$ changes the route the train used to reach its destination, it is not responsible for the train being there at the end. If $flip(0)$ did not occur, the train will be there anyway.

The next example deals with a story which, in addition to regular causal mechanisms, involves *intentions* and random experiments.

Example 4.4 (*A shooting story*)
Consider a variant of Yale Shooting story: *Fred has a loaded gun and intends to kill a turkey for dinner. Mary can unload the gun before Fred's departure without being noticed by him. Mary is tormented by the situation. On the one hand she wants to save the turkey. On the other hand she is reluctant to deprive Fred from having his dinner. She decides to rely on chance and flip a coin. If the coin shows "heads" she will unload the gun. Otherwise, she will not interfere.*

Our background theory Y has regular actions *shoot* (which is impossible if the gun is unloaded), and *unload*, random action $random(flip(I), head(I+1))$, where *head* (the coin lands heads) is a transient fluent, and inertial fluents *loaded* and *alive*. To talk about Fred's intent we need a mental action $intend(shoot(T))$ and non-procrastination axiom

$$shoot(T) \leftarrow intend(shoot(T), I), I \leq T, \text{ not } \neg shoot(T)$$

from (Baral and Gelfond 2005). Next two laws specify the effects of *shoot* and *unload*

$$[m_1(T)]: \ \neg alive(T+1) \leftarrow shoot(T), \text{not } ab(m_1(T))$$

$$[m_2(T)]: \ \neg loaded(T+1) \leftarrow unload(T), \text{not } ab(m_2(T))$$

We also need executability condition for *shoot*

$$\neg shoot(T) \leftarrow \neg loaded(T)$$

causal mechanism describing the Mary's strategy

$$[m_3(T)]:\ unload(T) \leftarrow head(T), \text{not } ab(m_3(T))$$

and the inertia and the contingency axioms for causal laws. In theory of intentions one must be careful with closed world assumption - it is only applicable if the action is not intended. So CWA for *shoot* will look as follows:

$$\neg shoot(T) \leftarrow \neg intend(shoot, T), \text{not } shoot(T).$$

CWA for other actions are regular. For simplicity we do not allow indirect exceptions to any of our CWA.

Suppose now that the agent recorded the story together with observations of the Mary's experiment and the value of fluent *alive* at time-step 3. First assume that the coin shows *head* and, as expected, $alive(3)$ is observed to be true, i.e., we have scenario

$$S_1 = S \cup \{obs(head(1)), obs(alive(3))\}$$

where $S = \langle obs(alive(0)), obs(loaded(0)), intend(shoot(2),0), random(flip(0), head(1))\rangle$.
One can check that
$M_1 = \{alive(0), loaded(0), intend(shoot(2),0), flip(0), head(1), loaded(1), alive(1),$
$unload(1), \neg loaded(2), \neg shoot(2), alive(2), \neg loaded(3), alive(3)\}$
is the only model of S_1.

To find a cause of $alive(3) \in M_1$ first notice that its inflection point is 1. Indeed, $S_1[0]$ has two models: one containing $head(1)$ and $alive(3)$ and another containing $\neg head(1)$ and $\neg alive(3)$. Thus, $S_1[0]$ does not derive $alive(3)$. For any $0 < i \leq 3$, M_1 is the only model of $S_1[i]$ and hence $S_1[i]$ derives $alive(3)$. Both conditions of Definition 3.3 are satisfied. Since no deliberate action happen in S after time-point 0, Definition 3.4 is not applicable. In fact, no action occurring in the model changes the value of $alive(0)$ and hence $alive(3)$ is caused by an exogenous action which brought about $alive(0)$.

Note that, since our theory does not allow indirect exceptions to CWA for *load* and *shoot*, scenario in which the agent observes $head(1)$ and $\neg alive(3)$ is inconsistent and will not be considered. We have two other possibilities:

$$S_2 = S \cup \{obs(\neg head(1)), obs(\neg alive(3))\}$$

and

$$S_3 = S \cup \{obs(\neg head(1)), obs(alive(3))\}$$

The first, expected one, has unique model M_2:

$alive(0), loaded(0), intend(shoot(2),0), flip(0), \neg head(1), loaded(1), alive(1),$
$loaded(2), alive(2), shoot(2), loaded(3), \neg alive(3)\}$

The inflection point for $\neg alive(3)$ is 1 and, by Definition 3.5, $\neg alive(3)$ is caused by $shoot(2)$. The only model of S_3 contains $ab(m_1(2))$ which, by Definition 3.6, is caused by an unknown exogenous event responsible for malfunctioning of the gun; by the same definition the turkey is alive due to an unknown exogenous event which caused the turkey to be alive at the beginning of the story.

The following example – similar to one from (Halpern 2016) – illustrates our treatment of preferences between causal mechanisms.

Example 4.5 (*Multiple Orders*)
Suppose that a captain and a sergeant stand before a private, both shout "Charge" at the same time, and the private charges. We follow Halpern and assume that "Orders from higher-ranking officers trump orders from lower-ranking officers". This will be represented by background theory T containing sorts $person = \{c, s, p\}$, $rank = \{1, 2, 3\}$, $command = \{advance, retreat\}$, static $rank(c) = 3, rank(s) = 2$ and $rank(p) = 1$. There are actions $do(A, C)$ (A executes command C) and $order(A, B, C)$ (A orders B to do C). A causal mechanism

$$[m(A,P,C,I-1)]: \quad do(B,C,I) \leftarrow order(A,B,C,I-1), \text{not } ab(m(A,B,C,I-1))$$

indicates that, as a rule, person B executes command C given to him by A. We, of course, assume that commands are legal, e.g.

$$\neg order(A,B,C,I) \leftarrow rank(A) <= rank(B),$$

that at most one command is given by one person at the time, etc. Multiple orders, however, can be given by different people. In this case one is supposed to execute the order given by the person of the highest rank (we assume that there is only one such person) and ignore all the others. This will be modeled by stating preferences between the defaults. This can be done by following standard ASP methodology. In our case we simply say:

$$ab(m(A_1,B,C_1,I)) \leftarrow order(A_1,B,C_1,I-1), order(A_2,B,C_2,I-1), rank(A_1) < rank(A_2).$$

In a scenario

$$S_0 = \langle order(c,p,attack,0), order(s,p,retreat,0), obs(do(p,attack,1))\rangle$$

event $do(p, attack, 1)$ is caused by the order of c. The same is true in scenario

$$S_0 = \langle order(c,p,attack,0), order(s,p,attack,0), obs(do(p,attack,1))\rangle$$

So far, we have only considered stories which contain no specific information about probabilities of outcomes of random events. Of course, we could ask and answer some probability related questions typical for P-log, e.g., "What is the probability of the coin showing head in scenario $\langle agreed\_to\_play(0)\rangle$ from Example 2.2?" By the indifference principle built into the semantics of P-log the answer would be 0.5[4]. If, however, we learn that the coin is not fair and lands heads with probability 0.6, we expend the program by $pr_{flip(T-1)}(head(T)) = 0.6$. This will change the answer to 0.6.

It is more difficult to reason about likelihood of causes in theories which use cr-rules. This will be later addressed in the full paper.

5 Related Work

In this section we briefly discuss the relationship between our approach and several others. We start with, currently best known and influential, Halpern-Pearl (HP) definition of causality. There are several basic differences between our approach and that of HP.

[4] For more details see Appendix 2 and/or (Balai et al. 2019)

- HP is based on counterfactual interpretation of causality and, as all counterfactual based definitions, takes as its starting point the "but for" test. A is a cause of B if, but for A, B would not have happened. Our intuition is more direct. It's starting point, based on theories of actions and their effects, views A as a cause of B if A initiates a chain of events which, according to universal causal laws, brings about B.
- Both approaches recognize the dependence of mathematical definition of causality on formal language used to represent agent's knowledge. The HP approach concentrates primarily on causal knowledge and bases its definitions on the language of structural equations (ref). In its pure form the language is propositional. We believe that, in many cases, correct causal reasoning requires substantial amount of non-causal background knowledge, such as explicit and recursive definitions, wide range of defaults and their exceptions, ontological knowledge, time, etc. Consequently, our language of choice is P-log, which uses variables and covers these and other types of knowledge.

These foundational differences lead to different treatment of particular examples. Let us illustrate this by looking at Forest Fire Example from (Halpern 2016) in which the agent is trying to determine whether a forest fire was caused by lightning or arsonist. The domain is modeled by three Boolean variables: ff for forest fire, l for lightning, and md for match dropped (by arsonist). In the conjunctive model both the match and lightning are needed to start the fire. This is expressed by causal equation: $ff = md \wedge l$. An exogenous variable u is a context which determines the values of l and md in particular scenarios. If $u = (1,1)$ the lightning strikes and arsonist drops the match; $u = (1,0)$ corresponds to lightning but no match, etc. According to (all three) HP definitions considered in the book both the lighting (l) and the dropped match (md) are causes of the fire; if either one had not occurred the fire would not have happened; $l \wedge md$ however is not a fire's cause.

In our approach the story will be modeled by causal mechanism

$$m(0) \ : \ ff(1) \leftarrow md(0), l(0), \text{not } ab(m(0))$$

where ff, md, and l are actions, and scenario

$$S = \{md(0), l(0), obs(ff(1))\}.$$

One can easily check that the cause of $ff(1)$ in our setting is action $\{md(0), l(0)\}$ which corresponds to $l(0) \wedge md(0)$; $l(0)$ however is not a cause of $ff(1)$. This fits well with our informal understanding of causality. Execution of $\{md(0), l(0)\}$ does "bring about" $ff(1)$ while execution of $l(0)$ does not. We need both actions.

The difference is even more pronounced in the disjunctive model given by structural equation $ff = l \vee md$. First, in this case there are differences between three HP definitions presented in the book. With the last, so called modified, definition neither l nor md is a cause of ff. Instead, its cause is $l(0) \wedge md(0)$. The authors acknowledge that calling the conjunction $l \wedge md$ a cause of ff "does not seem to accord with natural language usage". As one of possible ways to address this concern they suggest that "it may be better to think of parts of causes as coming closer to what we call causes in natural language". None of these problems exist in our approach, where the disjunctive structural equation is modeled by causal mechanisms

$$m_1(0) \ : \ ff(1) \leftarrow md(0), \text{not } ab(m_1(0))$$

$$m_2(0) \ : \ ff(1) \leftarrow l(0), \text{not } ab(m_2(0))$$

Now $ff(1)$ from scenario S has two causes: $l(0)$ and $md(0)$. We believe that this is the view which is in tune with everyday language use. Note, that the same answer is given by two earlier versions of HP definition.

There is a number of papers aimed at giving definition of causality based on theory of action and change. In (Batusov and Soutchanski 2018) the authors represent knowledge needed for causal reasoning in Situation Calculus - an action formalism proposed in (Mccarthy and Hayes 1969) and further elaborated in (Reiter 2001). The basic intuition behind definition of causality is similar to ours. It is formulated as follows: "If some action α of the action sequence σ triggers the formula $\phi(s)$ to change its truth value from false to true and if there is no action in σ after α that changes the value of $\phi(s)$ back to false, then α is an actual cause of achieving $\phi(s)$ in σ." If such action exists it can be found by a form of backward chaining. Even though in some respects Situation Calculus is more expressive then the language of structural equation the authors realize its limitations. They write: "It is clear that a broader definition of actual cause requires more expressive action theories that can model not only sequences of actions, but can also include explicit time and concurrent actions." This is what is done in our approach. We have explicit time, concurrent actions, triggers, can represent defaults, recursive definitions, state constraints, intentions, etc. The rich power of P-log allows to reason about incomplete scenarios, explain unexpected observations, and compute likelihoods of causes.

Another work which is similar in spirit to that of (Batusov and Soutchanski 2018) as well as to our work is (LeBlanc et al. 2019). Here background theory is formulated in action language $\mathscr{AL}$ (Gelfond and Inclezan 2009) which, unlike the form of Situation Calculus used in (Batusov and Soutchanski 2018), allows state constraints and defined fluents. In the authors' approach the distinction between direct and indirect effects of actions plays an important role in determining causes of events. More effort is needed for serious study of the relationship but, as in the case of Situation Calculus, our background theory is stated in a much richer language. In addition, the authors acknowledge, that their framework identifies partially counter-intuitive subsets of elementary events as indirect causes. This is illustrated by the following example from (LeBlanc 2019) adopted to our language. Consider background theory T:

$$m_1 : \ d_1(1) \leftarrow e_1(0)$$

$$m_2 : \ d_2(1) \leftarrow e_2(0)$$

$$f_1(1) \leftarrow d_1(1)$$

$$f_2(1) \leftarrow d_2(1)$$

and scenario

$$S = \{e_1(0), e_2(0)\}.$$

Then, "Due to the definitions of static chains and the improved definition of indirect cause, both $e_1(0)$ and $\{e_1(0), e_2(0)\}$ will be identified as indirect causes of $f_1(1)$, even though intuitively one might expect only $e_1(0)$ to be an indirect cause." Our definition gives the correct answer.

We are also currently investigating a close relationship which seems to exist between our work and that of (Cabalar et al. 2014). In this approach an agent's knowledge is represented by a normal logic program and causes are defined as causal graphs – directed graphs of rule labels that reflect some order of applications of rules which could have been used in constructing a particular model. In this way, each true atom of the model is assigned the value that contains justifications

for its derivation from the existing rules. We tentatively conjecture that for programs without cr-rules, random events, and observations, causal chains leading from the inflection points to an event in our approach correspond to causal chains which can be extracted from causal graphs. There are, however, substantial differences between the approaches. Signatures of our causal theories include fluents and actions, with actions playing special role in the definition of causes. There is no such distinction in (Cabalar et al. 2014). There is also no separation between background knowledge and scenarios and between deliberate, accidental and exogenous actions, etc. We believe that these features shed some light on the nature of causes, may increase elaboration tolerance and readability of programs and help with methodology of knowledge representation. Of course the possibility to express indirect exceptions to default increases expressive power of the language and allows to reason with unexpected observations which cannot be explained by explicitly stated exceptions as well as with deliberate actions. Similarly with probabilistic reasoning. However, one important feature hinted at in (Cabalar et al. 2014) and further elaborated in (**?**) is missing from our approach. While we concentrate on using knowledge to determine various cause-effect relations our language does not provide the means to use these relations to derive new conclusions. We cannot say "if a causes f then g is true". Languages discussed in the cited papers above can. We believe this to be an important feature and hope to be able to include it in our approach.

This is certainly not a complete list of works on causality in logic programming and theory of action and change but, to the best of our knowledge, it contains papers which are closest to our work. We plan to eventually expand the section but this is left for the future work.

6 Conclusion and Future Work

The paper reports on the ongoing work aimed at analysis of causal events in trajectories of dynamic domains. In our approach the background knowledge of an agent is formulated in powerful knowledge representation language P-log which combines reasoning in Answer Set Prolog and abductive reasoning with consistency restoring rules and causal probabilistic reasoning. The division of agent's knowledge into general part describing properties of a dynamic domain and a particular history of actions and observations recorded by the agent allows for higher degree of elaboration tolerance (McCarthy 1998) and more transparent representation. It also allows useful distinction between deliberate and accidental causes. We believe that determination of causes depends substantially on formal representation of informal knowledge and try the best we can to use general knowledge representation methodology developed in logic programming, nonmonotonic reasoning, and theory of action and change communities in the last twenty years. In fact, our definitions of causes is given in terms of causal laws developed in this area of research as opposed to counterfactual approach powerfully advocated by Pearl, Halpern, and others. Discovery of precise relationship between this and other contemporary approaches to causality with our work is an interesting and important open problem. Even though in some respects our formalism is a more powerful modeling tool than that of structural equations and graphical models advocated by Pearl and many others it remains to be seen if it can also expand their computational power. There are many open mathematical problems associated with causal theories. Just as an example we mention the need for mathematical theory of causal equivalence. Here is a suggested definition: We say that two background theories T_1 and T_2 are causally equivalent with respect to scenario S if causes of observations and actions from S are the same in $T_1(S)$ and $T_2(S)$; T_1 and T_2 are strongly causally equivalent if they are causally equivalent with respect to every sce-

nario S. Can we find reasonable sufficient conditions to guarantee these properties of programs? Of course we need to develop and implement algorithms for computing causal queries, further develop methodology for representing causal knowledge, better understand what preference relation (or relations) is most suitable for combining diagnostic and causal reasoning, etc. The list is large and serious progress may only be possible with more people getting interested in the subject.

7 Appendix 1 – Cr-Prolog

In what follows we give a brief description of the syntax and semantics of CR-Prolog. A program of the language is a four-tuple consisting of

1. A (possibly sorted) signature.
2. A collection of regular rules of ASP.
3. A collection of rules Π of the form
$$l_0 \stackrel{+}{\leftarrow} l_1, \ldots, l_k, \text{not } l_{k+1}, \ldots, \text{not } l_n \tag{1}$$
where ls are literals. Rules of type (1) are called **consistency restoring rules (cr-rules)**.
4. A partial order, $\leq$, defined on sets of cr-rules. This partial order is often referred to as a **preference relation**.

Whenever signature can be extracted from the rules and the preference relation is clear from the context we refer to this program as Π. Intuitively, rule (1) says that if the reasoner associated with the program believes the body of the rule, then it "may possibly" believe its head; however, this possibility may be used only if there is no way to obtain a consistent set of beliefs by using only regular rules of the program. The partial order over sets of cr-rules is used to select preferred possible resolutions of the conflict. Currently the inference engines of CR-Prolog support two such relations. One is based on the set-theoretic inclusion ($R_1 \leq_1 R_2$ holds iff $R_1 \subseteq R_2$). Another is defined by the cardinality of the corresponding sets ($R_1 \leq_2 R_2$ holds iff $|R_1| \leq |R_2|$). Unfortunately, neither is fully adequate if we want explanations which do not lose causal relations relevant to our domain. To see the problem consider a simple program T consisting of rules

$$f(1) \leftarrow e_1(0)$$
$$f(1) \leftarrow e_2(0)$$
$$e_1(0) \stackrel{+}{\leftarrow}$$
$$e_2(0) \stackrel{+}{\leftarrow}$$

The first two rules is a simplified representation of the fact that symptom $f(1)$ has exactly two possible causes - $e_1(0)$ and $e_2(0)$. According to the mentioned above minimality criteria program T used together with a scenario $S = \langle obs(\neg f(0)), obs(f(1))\rangle$ has two possible worlds:

$W_1 = \{\neg f(1), e_1(0), f(1)\}$
$W_2 = \{\neg f(1), e_2(0), f(1)\}$

There is no possible world in which both $e_1(0)$ and $e_2(0)$ happen simultaneously. This, however, can be important since the presence of an additional cause can influence treatment of the patient or have some other important causal consequences, and/or influence probability of the domain events.

So in this paper we introduce another preference relation, based on the idea of relevance. First,

some definitions. Given terms $f(\bar{x}_1)$ and $p(\bar{x}_2)$ of program T we say that $f(\bar{x}_1)$ *directly depends on* $p(\bar{x}_2)$ in Π if

- there is a rule r in Π such that $f(\bar{x}_1)$ occurs in the head of r and $p(\bar{x}_2)$ occurs in r's body, or
- head of r has the form $random(m, f(\bar{x}_1), p)$, where $p(\bar{x}_2)$ is an instance of $p(X)$;

$f(\bar{x}_1)$ *depends on* $p(\bar{x}_2)$ in Π if it belongs to reflexive, transitive closure of this relation; $f(\bar{x}) =$ *y depends on* cr-rule r in Π if $f(\bar{x})$ depends on a term occurring in the head of r. A cr-rule is called *irrelevant* to the observations of program Π if no observation from Π depends on r. Let R be a set of cr-rules from Π. By $rank(R)$ we denote the number of rules of R irrelevant to observations from Π + 1 if $R \neq \{\ \}$ and 0 otherwise.

Definition 7.1 (*Relevance Based Preference Relation*)
Let R_1 and R_2 be sets of cr-rules of program Π. We say that R_1 is *preferred to* R_2 (and write $R_1 < R_2$) if $rank(R_1) < rank(R_2)$.

To give the precise semantics we need more terminology and notation.

The set of regular rules of a CR-Prolog program Π is denoted by Π^{reg}; the set of cr-rules of Π is denoted by Π^{cr}. By $\alpha(r)$ we denote a regular rule obtained from a consistency restoring rule r by replacing $\stackrel{+}{\leftarrow}$ by $\leftarrow$; α is expanded in a standard way to a set R of cr-rules, i.e.,

$$\alpha(R) =_{def} \{\alpha(r)\ :\ r \in R\}.$$

Finally

$$T^R =_{def} \Pi^{reg} \cup \alpha(R).$$

As in the case of ASP, the semantics of CR-Prolog is given for ground programs. A rule with variables is viewed as a shorthand for a schema of ground rules.

Definition 7.2
(Abductive Support)
A minimal (with respect to the preference relation of the program) collection R of cr-rules of Π such that T^R is consistent (i.e. has an answer set) is called an **abductive support** of Π.

Definition 7.3
(Answer Sets of CR-Prolog)
A set A is called an *answer set* of Π if it is an answer set of a regular program T^R for some abductive support R of Π.

Consider, for instance, the following CR-Prolog program:

$$p(a) \leftarrow not\ q(a).$$
$$\neg p(a).$$
$$q(a) \stackrel{+}{\leftarrow}.$$

It is easy to see that the regular part of this program (consisting of the program's first two rules) is inconsistent. The third rule, however, provides an abductive support which allows to resolve inconsistency. Hence the program has one answer set $\{q(a), \neg p(a)\}$. This example has only one possible resolution of the conflict and hence its abductive support does not depend on the

preference relation of the program. This is of course not always the case. Consider program T and scenario S described above. Then program $T(S) = T \cup S$ has three possible worlds

$W_1 = \{\neg f(1), e_1(0), f(1)\}$
$W_2 = \{\neg f(1), e_2(0), f(1)\}$
$W_3 = \{\neg f(1), e_1(0), e_2(0), f(1)\}$

each generated by a collection of cr-rules of rank 1. Note, that if we were to expand T by $e_3(0) \stackrel{+}{\leftarrow}$ then the rank of this rule in the new program $T_1(S)$ would be 2. In other words this addition will not change possible worlds of the program.

8 Appendix 2 – P-log

Terms of a sorted signature of a P-log program are divided onto

- Standard *arithmetic terms*.
- *Regular terms* - expressions of the form $f(\bar{x})$ where f is a function symbol with signature $s_1, \ldots, s_n \rightarrow s$.
- *Special terms* - listed below.
  - — $do(r, f(\bar{x}), y)$ – "a random experiment r assigning value to $f(\bar{x})$ is deliberately interfered with and $f(\bar{x})$ is assigned the value y".
  - — $obs(f(\bar{x}), y)$ – "the value of $f(\bar{x})$ is observed to be y". If f is boolean we sometimes write $obs(f(\bar{x}))$ and $obs(\neg f(\bar{x}))$ for $y = true$ and $y - false$ respectively.
  - — $random(r, f(\bar{x}), p)$ – "$f(\bar{x})$ may take the value from the set $\{X : p(X)\} \cap range(f(\bar{x}))$ as the result of a random experiment r which is either genuine or deliberately interfered with."
  - — $truly\_random(r, f(\bar{x}))$ – "$f(\bar{x})$ takes value as the result of the genuine random experiment r (i.e., the one without any outside interference)".

The range of all special terms is **boolean**. Arithmetic atoms are defined as usual; non-arithmetic atoms are of the form $t = y$ where t is a term and y is its possible value. An atom $t = y$ is called *special* if t is a special term, otherwise it is called *regular*. Atoms formed by *obs* and *do* are called *observation* and A *P-log rule*, r, is of the form:

$$head(r) \leftarrow body(r) \tag{2}$$

where $head(r)$ is an atom and $body(r)$ is a collection of atoms possibly preceded by default negation *not* . If $head(r)$ is an observation or an action, we require $body(r)$ to be empty and the rule is called *an activity record*. If $head(r)$ is of the form $random(rn, a, p)$, the rule is called *a random selection rule* with name rn, a is referred to as *random attribute term* of the program containing such a rule. Otherwise a rule is called *regular*.

By a *P-log program* we mean a pair consisting of a signature Σ and a collection R of P-log rules and *causal probability statements* (also called *pr-atoms*) – expressions of the form

$$pr(r, f(\bar{x}) = y \mid B) = v \tag{3}$$

where $f(\bar{x})$ is a regular term such that $y \in range(f(\bar{x}))$, B is a set of atoms, possibly preceded by *not* , and $v \in [0, 1]$ is a rational number. The statement says that "if the value of $f(\bar{x})$ is generated randomly by experiment r and B holds then the probability of the selection of y for the value

of $f(\bar{x})$ is v. We will refer to $f(\bar{x}) = y$ as the *head* of the pr-atom and to B as the *body* of the *pr*-atom. We will refer to v as the *probability assigned to* $f(\bar{x}) = y$ *by the pr-atom*.

In addition, every P-log program contains the following rules, called *general P-log axioms*.

General P-log axioms:

- For every regular term $f(\bar{x})$ of Σ the rules

$$\leftarrow not\ f(\bar{x}) = Y,\ obs(f(\bar{x}), Y, true) \tag{4}$$

$$\leftarrow not\ f(\bar{x}) \neq Y,\ obs(f(\bar{x}), Y, false). \tag{5}$$

  The rules are often referred to as *reality check axioms*. Intuitively, they prohibit observations of undefined attribute terms as well as observations which contradict the agent's beliefs.
- For every atom $random(r, f(\bar{x}), p)$ of Σ such that $range(f(\bar{x})) = \{y_1, \ldots, y_k\}$, the rules :

$$f(\bar{x}) = y_1 \text{ or } \ldots \text{ or } f(\bar{x}) = y_k \leftarrow random(r, f(\bar{x}), p) \tag{6}$$

$$\begin{array}{rcl} truly\_random(r, f(\bar{x})) & \leftarrow & random(r, f(\bar{x}), p), \\ & & not\ do(r, f(\bar{x}), y_1), \ldots, not\ do(r, f(\bar{x}), y_k) \end{array} \tag{7}$$

$$\leftarrow f(\bar{x}) = Y,\ not\ \ p(Y), random(r, f(\bar{x}), p) \tag{8}$$

$$\begin{array}{rcl} \leftarrow & & random(r, f(\bar{x}), p), \\ & & not\ f(\bar{X}) = Y, \\ & & do(r, f(\bar{X}), Y). \end{array} \tag{9}$$

  Intuitively, the rules (6) and (8) guarantee that if $random(r, f(\bar{x}), p)$ is true, then $f(\bar{x})$ is assigned the value satisfying condition p by experiment r, rule (7) makes sure that $truly\_random(r, f(\bar{x}))$ is true iff the value of $f(\bar{x})$ is assigned as the result of a truly random experiment r, i.e., an experiment without any intervention, and rule (9) guarantees that the atoms made true by interventions are indeed true if $random(r, f(\bar{x}), p)$ is true.

In addition, for every rule r which is not a general axiom, we disallow literals formed by *obs*, *do*, *truly_random* and *random* to occur in the body of r. Elements of a program such as terms, atoms, rules, programs, etc., are called *ground* if they contain no free occurrence of any variable and no names of arithmetic functions. As usual, the semantics will be defined for ground programs. In addition, we only allow programs in which no two random selection rules have the same name.

The semantics of a P-log program Π is given by the collection $\mathscr{W}(\Pi)$ of possible worlds of Π and the probability function P_Π[5], defined on the sets of these worlds. The former correspond to possible sets of beliefs of a rational agent associated with the program, and the latter specifies degrees of such beliefs. The following examples illustrate the definition. Recall that, in addition to the explicitly stated rules, every program below contains general P-log axioms 4–9. To simplify the specification of programs' signatures, we use standard mathematical declarations of functions, $f : s_1, \ldots, s_n \rightarrow s$ and sorts $s = \{t_1, \ldots, t_m\}$. If $n = 0$, we simply write $f : s$. We follow the input language of P-log implementation – start the sort names with # and replace $\neg$ by -.

[5] When Π is clear from the context we may simply write P instead of P_Π.

Example 8.1
Consider the program Π_1

```
a,b,c: #boolean.
random(r1,a).
b :- c, -a.
do(r1,a, false).
random(r2,c).
```

It is not difficult to see that Π_1 has two possible worlds W_1 and W_2:

$$W_1 = \{\neg a, b, c, do(r1, a, false), random(r1, a), truly\_random(r2, c), random(r2, c)\}$$

and

$$W_2 = \{\neg a, \neg c, do(r1, a, false), random(r1, a), truly\_random(r2, c), random(r2, c)\}.$$

In both worlds $r1$ is interfered with, while $r2$ is truly random.

Example 8.2
Consider the program Π_2:

```
a: #boolean.
obs(a,true).
```

It has no possible worlds. Note that $W = \{obs(a, true)\}$ is not a possible world, because of the reality check axiom (4). If we expand the program by cr-rule $a \stackrel{+}{\leftarrow}$ the new program Π_3 will have possible world $W = \{obs(a, true), a\}$.

The precise definition of probability function defined by a P-log program is rather involved, but the intuition behind the definition is simple. So we illustrate it by an example.

Consider a program Π:

```
#s = {1,2,3}.
a : #s.
b : #boolean.
random(a).
random(b).
pr(a=1) = 1/2.
```

The program has six possible worlds corresponding to choices of values for a and b. For every possible world W and every attribute term a s.t. $truly\_random(a) \in W$ we first define *causal probability*, $P(W, a = y)$ for every possible outcome y of a (for the precise definition of possible outcome we refer the reader to (Baral et al. 2009)). Let us consider a possible world W_1 containing $a = 1$ and $b = true$. Causal probability $P(W_1, a = 1)$ is directly determined by the *pr*-atom of the program and is equal to $1/2$; the value of $P(W_1, b = true)$ is determined by the indifference principle which says that possible values of random attribute term are assumed to be equally probable if we have no reason to prefer one of them to any other, i.e., $P(W_1, b = true) = 1/2$. Now consider a possible world W_2 containing $a = 2$ and $b = true$. From the pr-atom of the program we know that the causal probability of a being equal to 2 or to 3 is $1/2$. Hence, by the indifference principle we have $P(W_2, a = 2) = 1/4$. Now we are ready to compute unnormalized

measure, $\hat{\mu}_\Pi(W)$ defined as the product of causal probabilities of random atoms from W. In our case, $\hat{\mu}_\Pi(W_1) = 1/2 \times 1/2 = 1/4$ and $\hat{\mu}_\Pi(W_2) = 1/4 \times 1/2 = 1/8$. As expected, the measure, $\mu(W)$ is the unnormalized probability of W divided by the sum of the unnormalized probabilities of all possible worlds of Π. Suppose now that Π is a P-log program having at least one possible world with nonzero unnormalized probability. The *probability*, $P_\Pi(E)$, of a set E of possible worlds of program Π is the sum of the measures of the possible worlds from E. The *probability* with respect to program Π of a literal l of Π, $P_\Pi(l)$, is the sum of the measures of the possible worlds of Π that satisfy l.

The corresponding functions are only defined for programs which satisfy three conditions on possible worlds of a program Π. Roughly speaking, the conditions ensure that for every W there is at most one random selection rule determining possible values of a in W, at most one pr-atom which can be used to define $P(W, a = y)$, and that $P(W, a = y)$ is not defined for y outside of the set of a's possible values determined by the random selection rule for a. For more information see (Balai et al. 2019).

References

BALAI, E., GELFOND, M., AND ZHANG, Y. 2019. P-log: refinement and a new coherency condition. *Annals of Mathematics and Artificial Intelligence 86,* 1-3, 149–192.

BALDUCCINI, M. AND GELFOND, M. 2003a. Diagnostic reasoning with a-prolog. *Journal of Theory and Practice of Logic Programming (TPLP) 3,* 4–5 (Jul), 425–461.

BALDUCCINI, M. AND GELFOND, M. 2003b. Logic programs with consistency-restoring rules. In *International Symposium on Logical Formalization of Commonsense Reasoning, AAAI 2003 Spring Symposium Series*. Vol. 102.

BARAL, C. 2003. *Knowledge Representation, Reasoning, and Declarative Problem Solving*. Cambridge University Press.

BARAL, C. AND GELFOND, M. 2005. Reasoning about intended actions. In *Proceedings of the National Conference on Artificial Intelligence*. Vol. 2. 689–694.

BARAL, C., GELFOND, M., AND RUSHTON, J. N. 2009. Probabilistic reasoning with answer sets. *Theory and Practice of Logic Programming 9,* 1, 57–144.

BARAL, C., GELFOND, M., AND RUSHTON, N. 2004. Probabilistic reasoning with answer sets. In *Proceedings of International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR)*.

BARAL, C. AND HUNSAKER, M. 2007. Using the probabilistic logic programming language p-log for causal and counterfactual reasoning and non-naive conditioning. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence*. 243–249.

BATUSOV, V. AND SOUTCHANSKI, M. 2018. Situation calculus semantics for actual causality. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

BECKERS, S. AND VENNEKENS, J. 2012. Counterfactual dependency and actual causation in cp-logic and structural models: a comparison. In *Proceedings of the Sixth Starting AI Researchers Symposium*. Vol. 241. IOS Press, 35–46.

BOCHMAN, A. 2018. Actual causality in a logical setting. In *IJCAI*. 1730–1736.

CABALAR, P., FANDINNO, J., FINK, M., LEUSCHEL, M., AND SCHRIJVERS, T. 2014. Causal graph justifications of logic programs. *Theory and Practice of Logic Programming 14,* 4-5, 603.

FANDINNO, J. 2016. Deriving conclusions from non-monotonic cause-effect relations. *CoRR abs/1608.00867*.

GELFOND, M. AND INCLEZAN, D. 2009. Yet another modular action language. In *In Proceedings of the Second International Workshop on Software Engineering for Answer Set Programming10*. 64–78.

GELFOND, M. AND KAHL, Y. 2014. *Knowledge representation, reasoning, and the design of intelligent agents: The answer-set programming approach*. Cambridge University Press.

GELFOND, M. AND RUSHTON, N. 2010. Causal and probabilistic reasoning in p-log. *Heuristics, probabilities and causality. A tribute to Judea Pearl*, 337–359.

GLYMOUR, C., DANKS, D., GLYMOUR, B., EBERHARDT, F., RAMSEY, J., SCHEINES, R., SPIRTES, P., MAN, C., ZHANG, T. J., GLYMOUR, B., EBERHARDT, F., RAMSEY, J., SCHEINES, R., SPIRTES, P., TENG, C. M., AND ZHANG, J. 2008. Synthese doi 10.1007/s11229-009-9497-9 actual causation: a stone soup essay.

HALL, N. 2004. *Two concepts of causation*. MIT Press, 225–276.

HALPERN, J. 2015. A modification of the halpern-pearl definition of causality. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

HALPERN, J. 2016. *Actual Causality*. MIT Press.

HALPERN, J. Y. AND HITCHCOCK, C. 2011. Actual causation and the art of modeling. *arXiv preprint arXiv:1106.2652*.

HALPERN, J. Y. AND HITCHCOCK, C. 2014. Graded causation and defaults. *The British Journal for the Philosophy of Science 66,* 2, 413–457.

HALPERN, J. Y. AND PEARL, J. 2005. Causes and explanations: A structural-model approach. part i: Causes. *The British journal for the philosophy of science 56,* 4, 843–887.

INCLEZAN, D. AND GELFOND, M. 2016. Modular action language. *TPLP 16,* 2, 189–235.

LEBLANC, E., BALDICCINI, M., AND VENNEKENS, J. 2019. Explaining actual causation via reasoning about actions and change. In *JELIA*.

LEBLANC, E. C. 2018. Explaining actual causation via reasoning about actions and change. In *Technical Communications of the 34th International Conference on Logic Programming (ICLP 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

LEBLANC, E. C. 2019. Explaining actual causation via reasoning about actions and change. Ph.D. thesis, Drexel University.

MCCARTHY, J. 1998. Elaboration tolerance. In *In Proc. of the 4th Symposium on Logical Formalizations of Commonsense Reasoning (Common Sense 98). London, UK, 198–217. Updated version at http://www-formal.stanford.edu/jmc/elaboration.ps.*

MCCARTHY, J. AND HAYES, P. J. 1969. Some philosophical problems from the standpoint of artificial intelligence. In *Machine Intelligence*. Edinburgh University Press, 463–502.

PEARL, J. 2009. *Causality*. Cambridge university press.

PEARL, J. AND MACKENZIE, D. 2018. *The Book of Why*. Basic Books.

REITER, R. 2001. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press.