HP-Causality in Multi-agent Planning Tasks

Axel Ind

February 7, 2019

Abstract

HP-Causality (@TODO reference) allows us to determine which variable values are causes of certain other final variable values. The normal HP-Causality approach is limited to a structural equation representations of the problem under consideration. This paper describes a multi-agent planning approach to defining causality. Particular focus is paid to the Modified definition of HP-causality.

1 Introduction

@TODOall mention why HP causality is not useful for action plans (single step etc), mention previous work in planning ethics limited to single-agent cases. Mention value of multi-agent planning (i.e. more realistic).

2 Planning Preliminaries

2.1 Single Agent Planning Tasks

Language

A single-agent planning task is a 4-tuple $\Pi = (V, A, s_0, s_\star)$. Where V denotes the set of variables and maps each to it's allowable values (or domain) such that, for some variable v, $v \in D_v$. A *fact* is a pair $(v, d \in D_v)$. A conjunction of facts $v_1 = d_1, ..., v_k = d_k$ is *consistent* if it contains no contradictory facts. That is, no pair of facts exists in this conjunction such that $v_j = d_j$ and $v_j \neq d_j$.

A denotes the set of *actions* available to the agent. An action $a =< pre, eff >$ consists of a precondition, and an effect. A precondition is a conjunction of facts $(v_1 = d_1, ..., v_k = d_k)$ and an effect is a conjunction of facts in *effect normal form* (@TODOref) of the form $\phi_i \triangleright v_i := d_i$. No contradictory effects are permitted for a single action. That is no pair of conditionals exists such that $\phi_i \triangleright v_j = d_j$ and $\phi_j \triangleright v_j \neq d_j$. A is divided into two types of actions, endogenous actions and exogenous actions. Intuitively endogenous actions describe those actions an agent is able to choose to take. @TODOref add the requirement that the set of endogenous actions always contains the empty action $\epsilon = (\top, \bot)$, this is not a strict requirement in this paper, but is worth noting as a possible

simplifying extension. Exogenous actions describe actions which are outside of the control of the agent and must be performed whenever possible. @TODOref treat exogenous actions as a series of, non-conflicting, actions which are associated with discrete time points $t(a)$ and must be applied as soon as they become applicable and result in a state change. By contrast, this paper drops the requirement that exogenous actions be non-conflicting and instead introduces the permissive and strict approach to dealing with such actions.

The *state* of a classical planning task is the conjunction of all variable pairs. s_0 describes the initial state of the agent, before any actions have been performed. $s_\star$ describes the goal state of the agent, to be achieved by execution of some number of consecutive actions.

Semantics

A action $a =< pre, effect >$ is applicable in a state s iff $s \models pre$ @TODOref. For exogenous actions, it is further required that the current state is the t-th time state associated with that action.

The *change set* $[eff]_s$ of an action is the set of facts that are affected by the conditional. specifically $[eff]_s = \cup_{i=1}^{k}[\phi_i \triangleright v_i := d_i]$, where $[\phi \triangleright v := d]_s = \{v = d\}$ if $s \models phi$ and $\emptyset$ otherwise. The change set never contains contradictory facts.

Applying an actions a to state s results a state s' achieved by replacing all variables in s that occur in $[eff]_s$ with conditional values assigned by that change set.

As with @TODOref, we assume *urgent semantics* for exogenous actions. That is, if an exogenous action is applicable, it must be applied immediately and no other action may precede it. However, because this paper allows conflicting exogenous actions, it is not obvious how to deal with cases where multiple exogenous actions are simultaneously applicable. There are three intuitive solutions to this problem:

1. *Ordering approach*: provide a strict ordering relationship among these actions and always apply the applicable action with the highest precedence first (provided it results in a state change).

2. *Permissive approach*: select some random order of exogenous actions.

3. *Strict approach*: consider every possible permutation of exogenous actions that can be performed.

This paper will concern itself only with the latter two approaches. $\Delta_{exo}(s)$ refers to the set of unique states that may be achieved by application of all relevant exogenous actions using one of the schemes above (Note that, unlike @TODOref, $\Delta_{exo}(s)$ is not closed and does not guarantee a single unique state in the strict approach).

A *plan* $\pi = (a_1, ...a_n)$ is a sequence of endogenous actions. In a typical plan it is sufficient to apply a_1 to the initials state s_0 to achieve the first state s_1 and then apply a_2, etc. However, to incorporate exogenous actions, $\Delta_{exo}(s)$ must

be calculated at each time point in which an exogenous action may occur. If each action in π (supplemented by appropriate intermediate states as a result of $\Delta_{exo}(s)$) is applicable to the preceding state, then the plan is called applicable.

It is worth noting that using the strict approach to exogenous actions above it does not simply suffice to show that some exogenous action in $\Delta_{exo}(s)$ will make the next endogenous action applicable. Instead all applicable exogenous actions must be considered, thus introducing branching complexity into the computation @TODOintroduceexampleofwhythismatters.

2.2 Multi-Agent Action Plans and Planning Tasks

Multi-agent planning tasks extend the intuition of single agent planning tasks as described @TODOsection to allow for consideration of a finite number of agents. Each agent may differ in their allowable actions and goals but are assumed share the same initial state and subsequent states after any endogenous or exogenous action occurs.

A multi-agent planning task Ξ for n agents is a 3-tuple as follows:

$$\Xi =< A, \Pi, T >$$

Where $A = (A_1, ...A_k)$ is an ordered list of all agent names in the planning task. Π is the ordered list of agent planning tasks for each agent in A. $\Pi = (\Pi_1, ..., \Pi_n)$, and T is a scheduling function which determines which agents may act at any given time-point t.

This definition intuitively aggregates a set of disparate planning tasks and makes them accessible via unique labels.

Multi-agent action plans are similar to single-agent action plans, but they also encode information about the agent which performed any given action [1]. Further, only multi-agent action plans in which the order of agent labels could feasibly be generated by T are considered valid.

A multi-agent action plan $\pi = ((A_x, \Pi_x[o]_1), ...(A_y, \Pi_y[o]_n))$. Where A_y is the label of the acting agent, and $\Pi_y[o]_n$ describes some applicable operator for agent y at time-point n.

For the purposes of readability, the rest of this paper will assume unique operator names and omit agent labels and specification of the operators available for that agent. Thus, multi-agent planning tasks will be treated as $\pi = (o_1, ...o_n)$, except where doing so would limit some required expressibility.

2.3 Counterfactual Reasoning in Planning

Counterfactual reasoning concerns the question: "What would happen if some feature of the world were other than it is?", and is an important part of many

[1] I am aware that this information is trivially obtainable when agent actions have unique names, but I feel that it is more robust to also include explicit labelling information. It allows us, for example, to introduce an arbitrary number of identical new agents to a task without having to change their action labels

approaches to causality @TODOrefhp. For example, we can intuitively grasp concepts such as: "*if the boy had not done his homework, he would not have passed*", which deal with a world different from the one in which we exist. Thus, without going into the minutia of defining causality, as will be done later, we have some intuition that not doing his homework, may have been cause of his failure.

@TODOref introduce and justify a simple counterfactual reasoning procedure for single-agent plans. Specifically it asks what would have happened if some action had not occured and the agent had instead done nothing? In the context of a planning task the authors note that is not sufficient to replace the action a_j in question with the empty action $\epsilon = (\top, \bot)$, because that action may have caused variable assignment $v = d$ and without that assignment some later action a_{j+k} may no longer be permissible. The authors, instead, allow impermissible actions to be considered during counterfactual reasoning. That is, they assume that, provided the plan was initially valid, we should allow all other actions to occur as they had before, regardless of violated preconditions.

For this paper we consider a slightly more robust choice of actions. Instead of introducing the empty action, we consider the full set of applicable actions available to an agent in a given state. In order to concisely capture the minutia of this kind of counterfactual reasoning we borrow intuition from the causality framework of @TODOrefHP and introduce our own framework.

Given a single-agent planning task $\Pi = (V, A, s_0, s_\star)$ and plan $\pi < a_1, ..., a_n >$, we define the *causal plan setting* (CSP) (π, Π). Intuitively the CSP corresponds to the full set of information we have involving the problem. It describes the goal, constraints, and exogenous circumstances that drove the sequence of actions that actually occurred.

Let s_n denote the final state of the agent after each action in the planning task has been applied[2]. The variable assignment for which we would like to identify the cause ϕ is always such that $s_n \models \phi$ in the CSP. The relationship between the CSP and ϕ is described as follows:

$$(\pi, \Pi) \models (s_n \models \phi)$$

An *action slot* $q(\pi, k)$ denotes the domain of possible actions at a given position. For example, if $\pi = (a_1, a_2, a_3)$, $q(\pi, 1)$ identifies the position occupied by a_1, thus containing action a_1 and any other action that could have been performed by any agent at that moment.

Let $\pi' \mapsto \pi$ denote a situation in which all elements in π' are action slots in π which could have occurred in the same order or else are the empty action. Specifically, for all $a_k \in \pi'$, $a \in q(\pi, k) \cup \epsilon$. The operation $\pi' \mapsto \pi$ is henceforth abbreviated to the symbol pi'.

$\pi' \leftarrow \vec{o'}$ is the setting of every non-ϵ action slot a_k to the singleton action $\vec{o'}[k]$.

[2]Although we have stated that plans using conflicting exogenous actions do not necessarily result in a single deterministic final state, our intuition for causality allows us to treat the plan and final state as known quantities examined after the fact. Thus we ensure that there is only one state final state s_n

A counterfactual consideration on a CSP asks the question "What if some subset of actions π' in π had been given the values $\vec{o'}$?". If changing the values of $\vec{o'}$ still results in the values ϕ in the final state, we write:

$$(\pi, \Pi) \models [\pi' \leftarrow \vec{o'}](s_n \models \phi)$$

If, on the other hand, ϕ no longer holds in the final state, we write:

$$(\pi, \Pi) \models [\pi' \leftarrow \vec{o'}](s_n \not\models \phi)$$

A *multiagent causal plan setting* (MCPS) works in an analogous way. The MCSP (π, Ξ) is the multi-agent planning task combined with the executed plan. Actions slots now also encompass allowable actions by other agents at a given timeslot (remember that which agents may act at that time slot is determined by the scheduling function T).

A counterfactual consideration on a MCPS in which setting the actions slots described by π' to the actions $\vec{o'}$ still results in the values ϕ in the final state, we write:

$$(\pi, \Xi) \models [\pi' \leftarrow \vec{o'}](s_n \models \phi)$$

And if ϕ no longer holds in the final state, we write:

$$(\pi, \Xi) \models [\pi' \leftarrow \vec{o'}](s_n \not\models \phi)$$

Finally, let $(\pi - \pi')$ denote the plan obtained by setting all non-ϵ actions in π' to ϵ in π.

3 Mathematical Preliminaries HP Causality

@TODOall Discuss the idea of causes. Provide details about the causal setting. Discuss counterfactual reasoning.

4 HP-Causality

This section briefly describes the notion of HP-causality and the three main definitions. @TODO must include 2 other definition and description of how inserting counterfactuals works.

4.1 The Modified definition of HP-Causality

Definition 4.1. $\vec{X} = \vec{x}$ is an actual cause of ϕ in the causal setting $(M, \vec{u})$ iff the following 3 conditions hold:

1. $(M, \vec{u}) \models (\vec{X} = \vec{x})$ and $(M, \vec{u}) \models \phi$.

2. $AC2(a^m)$: There is a set $\vec{W}$ of variables in V and a setting $\vec{x'}$ of the variables in $\vec{X}$ such that if $(M, \vec{u}) \models \vec{W} = \vec{w^*}$, then:

$$(M, \vec{u}) \models [\vec{X} \leftarrow \vec{x'}, \vec{W} \leftarrow \vec{w^*}]\neg\phi$$

3. $\vec{X}$ is minimal. There is no strict subset of $\vec{X}$ that satisfies the previous 2 conditions.

Intuitively the first two requirements are: the possible cause must actually have occurred, the final variable assignment under consideration must also have occurred; and changing the possible cause and fixing some number of other variable assignments can change the final variable assignment under consideration.

5 HP-Causality Planning

The HP causality definitions above are restricted to the identification of variable causes, and as such, are not suitable for use in a planning domain where questions posed may also require the identification of a subset of actions or agents as causes.

For this reasons I propose changed definitions of the HP-causality models which produce comparable conclusions given intuitively identical, but structural different problems. Specifically, my definitions allow for the use of actions plans rather than structural equations as original used.

5.1 But-for Causality Planning

5.2 HP-Causality Modified Planning

But-for Planning

Definition 5.1. Given a multi-agent action plan π with a final state s_n, some ordered subset of the actions in π, $\pi' \mapsto \pi$, $\pi' \leftarrow \vec{o}$ is a but-for cause of some final variable assignment $s_n \models \phi$ iff the following 3 conditions hold:

1. $\vec{o} \mapsto \pi$, $\pi' \models \vec{o}$ and $s_n \models \phi$.
2. $BF2$: There is a setting $\vec{o'}$ of the applicable actions in $\vec{\pi'}$ such that:
$$(\pi, \Xi) \models [\pi' \leftarrow \vec{o'}](s_n \not\models \phi)$$
3. π' is minimal. There is no strict subset of π' that satisfies the previous 2 conditions.

Modified Planning

Definition 5.2. Given a multi-agent action plan π with a final state s_n, some ordered subset of the actions in π, $\pi' \mapsto \pi$, $\pi' \leftarrow \vec{o}$ is a cause of some final variable assignment $s_n \models \phi$ according to the modified planning definition iff the following 3 conditions hold:

1. $\vec{o} \mapsto \pi$, $\pi' \models \vec{o}$ and $s_n \models \phi$.
2. $BF2$: There is a setting $\vec{o'}$ of the applicable actions in $\vec{\pi'}$, and a setting of $W \mapsto (\pi - \pi')$ such that:
$$(\pi, \Xi) \models [\pi' \leftarrow \vec{o'}, W \leftarrow \vec{w^\star}](s_n \not\models \phi)$$
3. π' is minimal. There is no strict subset of π' that satisfies the previous 2 conditions.

(Where $W \leftarrow \vec{w^\star}$ denotes fixing all non-ϵ action slots in W to their original actions.)

Agent Causes

Sometimes we may desire to know if a specific agent or subset of agents caused a particular variable assignment to occur. This allows for statements such as "Person X is solely responsible for event ϕ".

Definition 5.3. A subset of agents $\vec{A}$ is a cause of ϕ iff there exists some cause $\vec{X} = \vec{x}$ of ϕ such that every agent in $\vec{A}$ is a label of at least one action in $\vec{X}$, further, only agents mentioned in $\vec{A}$ are action labels in $\vec{X}$.

Notes on the above definitions

- These definitions are strictly more inclusive than but-for cause checks. This is because when $\vec{W} = \{\}$, the $MC2(a^m)$ precisely models but-for causes (under minimality requirements for the but-for cause).
- The action plan obtained by replacing certain actions may no longer be valid. Much like in the normal HP definition of causality, where impossible variable assignments can be fixed during counter-factual reasoning.
- The most significant remaining concern relates to the treatment of exogenous events. Using a single exogenous event for all non-agent actions makes it very difficult to properly assign causality due to problems it causes with fixing variables. This will be detailed in the next section.
- One might wonder why my definitions do not refer to fixing variables at all, and only focus on actions. I have no strong argument for this decision except the observations that fixing some variable $X = x$ can be done in constant time by the inclusion of an action $s = (\top, X = x)$ in the action plan. This action can then be fixed. This addition can be done in $O(|X|)$ time.

5.3 Planning Causality relations

The following table shows which classes causes my definition of causality can identify. Columns are results ϕ. Rows are causes $\vec{X}$.

	Variable	**Action**
Variable	No[3]	No
Action	Yes	@TODO
Agent	Yes	@TODO

@TODO I would like to address the following questions:

1. What does it mean for an action to be the cause of an action? Is it sufficient to say that without that prior action the next action would not have occurred? What if only one action was available so $X = x' = x$?

2. How would extensions to a game theory approach affect causality? What if changing some action X would always result in a known change to some later action Y? Would we treat the two actions as a single action for the purposes of fixing and changing actions? How would this affect computational complexity?

6 Responsibility and Blameworthiness

6.1 HP Responsibility and Blameworthiness

6.2 Extending Responsibility and Blameworthiness to Planning

7 Introducing Machiavellianism

... every prince should strive to be considered kind rather than cruel.

Based on The Price. Two requirements should not perform any action another agent would consider unethical, should be utilitarian otherwise.

Using epistemic states, leave variable assignments that others cannot see unbounded.

8 Examples

9 Conclusion