Diagnosis as Computing Causal Chains from Event Traces

Shakil M. Khan and **Mikhail Soutchanski**
Department of Computer Science
Ryerson University, Toronto, Canada
{shakilmkhan, mes}@scs.ryerson.ca

Abstract

In this paper, we build on previous work on diagnosis in dynamic discrete event systems by proposing to utilize a formal model of causal analysis in the situation calculus to further process reconstructed traces of events. By using the situation calculus, we adopt a language that is well suited for the task of integrated treatment of planning, plan execution and monitoring, plan repair, etc. Also, given a system, our definition enables us to model unwanted inter-component interactions, not just faulty components. We show that our definition has useful properties and is modular.

1. Introduction

Researchers in Artificial Intelligence have previously tackled the problems of planning, plan execution, plan repair, diagnosis, and causal explanation separately. Nevertheless, little previous work addressed these problems within a unified framework. To this end, in this paper we adopt one such possible framework developed within the scope of logical KR, namely the situation calculus (Reiter 2001). In particular, we focus on the problem of diagnosis through causal analysis within the situation calculus.

While the basic goal of diagnosis is to identify the underlying causes that explain a set of observed effects, it is understood differently by control and AI communities, and there are different approaches depending on applications; see (Travé-Massuyès 2014) for a survey. In the approaches of model-based diagnosis (**MBD**) applicable to static systems, the task is to identify from a given system description the minimal sets of faulty components (if any) that are consistent with a given set of observations (Reiter 1987; de Kleer and Williams 1989; de Kleer, Mackworth, and Reiter 1992). On the other hand, diagnosis in discrete event systems (**DES**) deals with dynamic systems where transitions are deterministic and a partially ordered set of observations is available, but some events can be unobservable. The task is to reconstruct the possible traces of (unobservable and observable) events and their ordering wrt each other to explain an effect (Sampath et al. 1995; 1996; Zaytoon and Lafortune 2013). Researchers have long acknowledged a close connection between diagnosis in dynamic DES and planning (McIlraith 1994; Cordier and Thiébaux 1994; McIlraith 1998). However, the diagnosis methods for DES can produce several traces that may include irrelevant events. In complex systems, for each completed trace the question remains which particular sub-sequence of events constitutes the root-causes of an observed failure?

In this paper, we build on the previous work about diagnosis of dynamic DES and propose to further refine the analysis of reconstructed traces by finding the sub-sequences of the events that serve as actual causes. We assume a given logical theory that models how the system responds to all actions/events. Since our proposal builds on the methods for diagnosis in DES, we can assume that a reconstructed event log or trace of the system is available at our disposal. We are looking for causally important events in the log that can actually explain an observed effect. Thereby, we contribute both to the definition of the diagnosis problem and to its solution. We do not conceptually distinguish between agents' actions and exogenous/nature's events.

We show how one can define (our notion of) diagnosis through the computation of causal chains within the situation calculus (SC). Adopting a first-order language like the situation calculus for causality analysis allows us to be more expressive; we can model systems in finer details and build on a recent problem-free definition of causality (Batusov and Soutchanski 2018). Moreover, researchers in KR have enriched the situation calculus by developing mechanisms for planning (Reiter 2001), plan execution and monitoring (De Giacomo, Reiter, and Soutchanski 1998), diagnosis (Sohrabi, Baier, and McIlraith 2010), causal explanation (Batusov and Soutchanski 2017), and handling continuous probability distributions (Belle and Levesque 2018), among other things. Thus, the language of SC is well suited for the task of integrated treatment of these issues. Furthermore, our formalization enables us to detect unwanted inter-component interactions, not just faulty components. Finally, we prove that our definition of diagnosis has some intuitively desirable properties and is modular.

We start by outlining the situation calculus in the next section. In Section 3, we discuss how achievement and maintenance causes can be defined within the situation calculus. In Section 4, we formalize our definition of diagnosis and discuss some properties of our approach. Finally we summarize our results in Section 5.

2. Background

The Situation Calculus (SC). The SC (McCarthy and Hayes 1969) is a popular formalism for modelling and reasoning about dynamic systems. Here, we use a (mostly) first-order sorted version as described by Reiter (2001). There are four basic sorts in the language, *situation*, *action*, *fluent*, and a catch-all *object* sort. A situation represents a sequence of actions. A special constant S_0 is used to denote the initial situation where no actions has yet been performed. There is a distinguished binary function symbol do, where $do(a, s)$ denotes the successor situation to s resulting from performing the action a. Thus the situations can be viewed as a tree, where the root of the tree is S_0 and the arcs represent actions. $do([a_1, \cdots, a_n], s)$ is used to denote the complex situation term obtained by consecutively performing $a_1, \cdots, a_n$ starting from s. Also, the notation $s \sqsubset s'$ means that situation s' can be reached from situation s by executing a sequence of actions. $s \sqsubseteq s'$ is an abbreviation of $s \sqsubset s' \vee s = s'$. Relations whose truth values vary from situation to situation are called relational fluents, and are denoted by predicate symbols taking a situation term as their last argument. There is a special predicate $Poss(a, s)$ used to state that action a is executable in situation s. Finally, a situation s is called *executable* if every action in its history was possible in the situation where it was performed:

$$executable(s) \overset{\text{def}}{=} \forall a', s'.\ do(a', s') \sqsubseteq s \rightarrow Poss(a', s').$$

Following Reiter, we use a basic action theory (**BAT**) $\mathcal{D}$ that includes the following set of axioms: (1) action precondition axioms $\mathcal{D}_{apa}$, one per action a characterizing $Poss(a, s)$, (2) successor-state axioms $\mathcal{D}_{ssa}$, one per fluent, that succinctly encode both effect and frame axioms and specify exactly when the fluent changes, (3) initial state axioms $\mathcal{D}_{S_0}$ describing what is true in S_0, (4) unique name axioms for actions $\mathcal{D}_{una}$, and (5) domain-independent foundational axioms Σ describing the structure of situations.

Example. We use the dining philosophers problem (Hoare 1985) as our running example. We have three philosophers seated around a table. In the centre of the table there is a bowl of spaghetti, and between each two neighbouring philosophers there is a fork. A philosopher p can pick up (and put down) a fork f by executing the $pickUp(p, f)$ ($putDown(p, f)$, resp.) action. A philosopher needs two forks to be able to eat, and can only reach the two forks immediately next to him, each of which is shared with the respective neighbour. Also, to initiate eating, p needs to execute the $eat(p)$ action. A philosopher can only think if he is not holding any forks. There are three fluents in this domain, $hasFork(p, f, s)$, $thinking(p, s)$, and $eating(p, s)$, which mean that philosopher p has fork f in situation s, p is thinking in s, etc. We say that a philosopher p is waiting in situation s if he is not eating or thinking, i.e.:

$$waiting(p, s) \overset{\text{def}}{=} \neg(eating(p, s) \vee thinking(p, s)).$$

Also, we use the non-fluent relation $neighb$ to describe the geometry of the seating arrangement:

$$\forall p, p', f.\ neighb(p, f, p') \leftrightarrow [(p = P_1 \wedge f = F_{12} \wedge p' = P_2) \vee (p = P_2 \wedge f = F_{23} \wedge p' = P_3) \vee (p = P_1 \wedge f = F_{13} \wedge p' = P_3)].$$

This says that philosopher P_1 is a neighbour of P_2 and they share the fork F_{12}, etc. In the following, we give some examples for each type of domain-dependant axioms; others (e.g. the action precondition axiom for action $putDown(p, f)$, the successor-state axiom for fluent $thinking(p, s)$, etc.) can be specified similarly and are assumed to be available. Thus, for instance, the preconditions for $pickUp(p, f)$ and $eat(p)$ can be specified using action precondition axioms (**APA**) as follows (henceforth, all free variables in a sentence are assumed to be universally quantified):

$$(a).\ Poss(pickUp(p, f), s) \leftrightarrow \neg hasFork(p, f, s) \wedge \exists p'.(neighb(p, f, p') \vee neighb(p', f, p)) \wedge \neg hasFork(p', f, s),$$
$$(b).\ Poss(eat(p), s) \leftrightarrow \exists f, f'.\ f \neq f' \wedge hasFork(p, f, s) \wedge hasFork(p, f', s) \wedge \neg eating(p, s).$$

That is, (a) a philosopher p can pick up a fork f in some situation s if and only if p is not already in possession of f in s and f is not being used by the respective neighbour of p in s, and (b) p can eat in s if he is holding two distinct forks and he is not eating in s.

Moreover, the following successor-state axiom (**SSA**) specifies how exactly the fluents $hasFork$ and $eating$ can change value when an action a happens in some situation s:

$$(c).\ hasFork(p, f, do(a, s)) \leftrightarrow a = pickUp(p, f) \vee hasFork(p, f, s) \wedge \neg a = putDown(p, f),$$
$$(d).\ eating(p, do(a, s)) \leftrightarrow a = eat(p) \vee eating(p, s) \wedge \neg \exists f.\ a = putDown(p, f).$$

That is, (c) a philosopher p is in possession of a fork f in the situation resulting from executing action a in situation s (i.e. in $do(a, s)$) if and only if a refers to p's action of picking f up from the table or p already had f in s and a is not the action of p putting it down. The case for the fluent $eating$ in (d) is similar.

Furthermore, the following initial state axioms state that initially (e) all three philosophers are thinking and (f) nobody is eating or is in possession of a fork:

$$(e).\ \forall p.\ thinking(p, S_0) \leftrightarrow p = P_1 \vee p = P_2 \vee p = P_3,$$
$$(f).\ \forall p, f.\ \neg eating(p, S_0) \wedge \neg hasFork(p, f, S_0).$$

Finally, we implicitly assume the domain closure axioms for philosophers; also the following unique name for actions axiom (**UNA**) says that (g) $pickUp$ and $putDown$ refer to different actions, and (h) two $pickUp$ actions refer to the same action if their arguments are the same (these are necessary for the above SSA to work properly):

$$(g).\ \forall p, p', f, f'.\ pickUp(p, f) \neq putDown(p', f'),$$
$$(h).\ \forall p, p', f, f'.\ pickUp(p, f) = pickUp(p', f') \rightarrow p = p' \wedge f = f'.$$

Regression in the SC. BATs employ *regression*, a powerful reasoning mechanism for answering queries about the future. Given a query "does ϕ hold in the situation obtained by performing the ground action α in situation s, i.e. in $do(\alpha, s)$?",[1] the single-step regression operator ρ transforms it into an equivalent query "does ψ hold in situation s?",

[1]A ground term is one whose constituents are ground sub-terms and constants, i.e. that contains no variables.

eliminating action α by compiling it into ψ. The expression $\rho[\phi, \alpha]$ denotes such an equivalent query obtained from the formula ϕ by replacing each fluent atom F in ϕ with the right-hand side of the successor-state axiom for F where the action variable a is instantiated with the ground action α, and then simplified using unique name axioms for actions and constants. One can prove that given a BAT $\mathcal{D}$, a formula $\phi(s)$ uniform in s (meaning that it has no occurrences of $Poss$, $\sqsubseteq$, other situation terms besides s, and quantifiers over situations), and a ground action term α, we have that $\mathcal{D} \models \forall s.\ \phi(do(\alpha, s)) \leftrightarrow \rho[\phi(s), \alpha]$. One can also obtain a similar regression operator $\mathcal{R}$ by repetitive recursive application of ρ. Reiter (2001) showed that for a *regressable* query ϕ, $\mathcal{D} \models \phi$ if and only if $\mathcal{D}_{una} \cup \mathcal{D}_{S_0} \models \mathcal{R}[\phi]$. Regression thus reduces second-order entailment to first-order entailment by compiling dynamic aspects of the theory into the query.

3. Achievement and Maintenance Causes

Given a (reconstructed) trace of all events, *actual achievement causes* are some of the events that are behind achieving an effect while *actual maintenance causes* are those that are responsible for mitigating the threats to the achieved effect. There can be also the cases of subtle interactions of these two. In this section, we review how one can define achievement causality in the SC (Batusov and Soutchanski 2018). An effect in this framework is an SC formula $\phi(s)$ that is uniform in s. Given an effect $\phi(s)$, the actual causes of ϕ are defined relative to a *causal setting* that includes a BAT $\mathcal{D}$ representing the domain dynamics, and a "narrative" (a trace of events) σ, representing the ground situation, where the effect was observed.

Definition 1 (Causal Setting). *A causal setting is a tuple $\langle \mathcal{D}, \sigma, \phi(s) \rangle$, where $\mathcal{D}$ is a BAT, σ is a ground situation term of the form $do([a_1, \cdots, a_n], S_0)$ with ground action functions $a_1, \cdots, a_n$ such that $\mathcal{D} \models executable(\sigma)$, and $\phi(s)$ is an SC formula uniform in s such that $\mathcal{D} \models \phi(\sigma)$.*

As the theory $\mathcal{D}$ does not change, we will often suppress $\mathcal{D}$ and simply write $\langle \sigma, \phi(s) \rangle$. Also, here we require ϕ to hold by the end of the narrative σ, and thus ignore the cases where ϕ is not achieved by the actions in σ, since if this is the case, the achievement cause truly does not exist.

Note that since all changes in the SC result from actions, we identify the potential causes of an effect ϕ with a set of ground action terms occurring in σ. However, since σ might include multiple occurrences of the same action, we also need to identify the situations when these actions were executed. Now, the notion of the achievement cause of an effect suggests that if some action α of the action sequence in σ triggers the formula $\phi(s)$ to change its truth value from false to true relative to $\mathcal{D}$, and if there are no actions in σ after α that change the value of $\phi(s)$ back to false, then α is the actual cause of achieving $\phi(s)$ in σ.

When used together with the single-step regression operator ρ, the above interpretation of achievement condition not only identifies the single action that brings about the effect of interest, but also captures the actions that build up to it. Intuitively, $\rho[\phi, \alpha]$ specifies the weakest condition that must hold in a previous situation (let us call it σ') in order for ϕ to hold after performing the action α in situation σ', i.e. in situation $do(\alpha, \sigma')$. Thus if the action α is an achievement cause of ϕ in situation $do(\alpha, \sigma')$, we can use the single-step regression operator ρ to obtain a formula that holds at situation σ' and constitutes a necessary and sufficient condition for the achievement of $\phi(s)$ via the action α. This new formula may have an achievement cause of its own which, by virtue of the action α, also constructively contributes to the achievement of ϕ. By repeating this process, we can uncover the entire chain of actions that incrementally build up to the achievement of the ultimate effect. At the same time, we must not overlook the conditions that make the execution of the action α in situation σ even possible, which are conveniently captured by the right-hand side of the APA for α and may have achievement causes of their own.

The following inductive definition formalizes this intuition. Let $\Pi_{apa}(\alpha, \sigma)$ be the right-hand side of the APA for action α with the situation term replaced by situation σ.

Definition 2 (Achievement Cause). *A causal setting $\mathcal{C} = \langle \sigma, \phi(s) \rangle$ satisfies the achievement condition of ϕ via the situation term $do(\alpha^*, \sigma^*) \sqsubseteq \sigma$ if and only if there is an action α' and situation σ' such that:*

$$\mathcal{D} \models \neg\phi(\sigma') \land \forall s.\ do(\alpha', \sigma') \sqsubseteq s \sqsubseteq \sigma \rightarrow \phi(s),$$

and either $\alpha^* = \alpha'$ and $\sigma^* = \sigma'$, or the causal setting $\langle \sigma', \rho[\phi(s), \alpha'] \land \Pi_{apa}(\alpha', \sigma') \rangle$ satisfies the achievement condition via the situation term $do(\alpha^*, \sigma^*)$. Whenever a causal setting $\mathcal{C}$ satisfies the achievement condition via situation $do(\alpha^*, \sigma^*)$, we say that the action α^* executed in situation σ^* is an achievement cause *in causal setting $\mathcal{C}$.*

Since the process of discovering intermediary achievement causes using the single-step regression operator ρ cannot continue beyond S_0, it eventually terminates. Moreover, since the narrative σ is a finite sequence, the achievement causes of $\mathcal{C}$ also form a finite sequence of situation-action pairs, which we call the *achievement causal chain of $\mathcal{C}$*.

Example (cont'd). Consider the narrative $\sigma_1 = do([pickUp(P_1, F_{12}), pickUp(P_3, F_{23}), pickUp(P_1, F_{13}), eat(P_1)], S_0)$, i.e. the philosopher P_1 picks up the fork on his left, then P_2 picks up the fork on his right, then P_1 picks up the fork on his right, and then P_1 eats. We are interested in computing the actual causes of the effect $\phi_1 = eating(P_1, s)$. Then according to Definition 2, the causal setting $\langle \phi_1, \sigma_1 \rangle$ satisfies the achievement condition ϕ_1 via the situation term $do(eat(P_1), S_3)$, where $S_3 = do([pickUp(P_1, F_{12}), pickUp(P_3, F_{23}), pickUp(P_1, F_{13})], S_0)$, so the action $eat(P_1)$ executed in situation S_3 is an achievement cause of ϕ_1.

Moreover, let us compute $\rho[eating(P_1, s), eat(P_1)]$ and $Poss(eat(P_1), S_3)$, starting with the former. From right-hand side of the SSA (d) above and by substituting action variable a by $eat(P_1)$, object variable p by P_1, and situation variable s by $\sigma_1 = do(eat(P_1), S_3)$, the result of single-step regression $\rho[eating(P_1, s), eat(P_1)]$ amounts to $(eat(P_1) = eat(P_1)) \lor (eating(P_1, S_3) \land \neg\exists f.\ eat(P_1) \neq putDown(P_1, f))$. Since by the unique names axioms the first disjunct can be replaced with true, the result of ρ can be simplified to true as well. Let us now consider $Poss(eat(P_1), S_3)$; from the right-hand side of APA (b) above and by replacing object variable p with P_1, we have $\exists f, f'.\ hasFork(P_1, f, s) \wedge hasFork(P_1, f', s) \wedge f \neq f' \wedge \neg eating(P_1, s)$. Computing $\rho[eating(P_1, \sigma_1), eat(P_1)] \wedge Poss(eat(P_1), S_3)$ thus gives rise to a new causal setting $\langle S_3, \exists f, f'.\ hasFork(P_1, f, s) \wedge hasFork(P_1, f', s) \wedge f \neq f' \wedge \neg eating(P_1, s)\rangle$. This setting satisfies the achievement condition via the action $pickUp(P_1, F_{13})$, so $pickUp(P_1, F_{13})$ executed in $S_2 = do([pickUp(P_1, F_{12}), pickUp(P_3, F_{23})], S_0)$ is an achievement cause.

Furthermore, this yields yet another setting $\langle \rho(\exists f, f'.\ hasFork(P_1, f, s) \wedge hasFork(P_1, f', s) \wedge f \neq f' \wedge \neg eating(P_1, s), pickUp(P_1, F_{13})) \wedge Poss(pickUp(P_1, F_{13}), S_2), S_2\rangle$. Doing simplifications similar to what we did before, we can arrive at the new causal setting $\langle \exists f.\ f \neq F_{13} \wedge hasFork(P_1, f, s) \wedge \neg eating(P_1, s) \wedge \neg hasFork(P_1, F_{13}, s) \wedge neighb(P_1, P_3, F_{13}) \wedge \neg hasFork(P_3, F_{13}, s), S_2\rangle$, which meets the achievement condition via $pickUp(P_1, F_{12})$ executed in situation S_0 and the analysis terminates.

We can also handle quantified queries. Consider another example; we want to determine the actual causes of $\phi_2 = \forall p.\ waiting(p, s)$ after each philosopher picks up the fork on his left starting in situation S_0, say in the narrative $\sigma_2 = do([pickUp(P_1, F_{12}), pickUp(P_2, F_{23}), pickUp(P_3, F_{13})], S_0)$. Note that using the definition of $waiting$ and the domain closure axiom that says that the only philosophers in this domain are P_1, P_2, and P_3, ϕ_2 can be first simplified as follows: $\neg eating(P_1, s) \wedge \neg thinking(P_1, s) \wedge \neg eating(P_2, s) \wedge \neg thinking(P_2, s) \wedge \neg eating(P_3, s) \wedge \neg thinking(P_3, s)$. Then a similar analysis as above can be used to show that according to our definition the achievement causal chain for this example is as follows: $[(pickUp(P_3, F_{13}), S_2), (pickUp(P_2, F_{23}), S_1), (pickUp(P_1, F_{12}), S_0)]$, where $S_1 = do(pickUp(P_1, F_{12}), S_0)$ and $S_2 = do(pickUp(P_2, F_{23}), S_1)$.

As shown in (Batusov and Soutchanski 2017), one can also define the concept of *maintenance causes* by appealing to a counterfactual notion of potential threats in the causal setting that can possibly flip the truth value of the effect ϕ to false, and actions in the narrative that mitigated those threats. In general, actual causes can be either achievement causes or maintenance causes and the causal chain can include both. However, to keep things simple, in this paper we focus exclusively on actual achievement causes.

4. Diagnosis as Computing Causal Chains

We model the behaviour of the system to be diagnosed as a BAT $\mathcal{D}$ in SC. Also, an *observation* or *effect* is encoded as a SC formula $\phi(s)$ uniform in s, i.e., it has no other situation terms except of s. A *system* $\mathcal{DS}$ in our framework is simply a causal setting $\langle \mathcal{D}, \sigma, \phi(s)\rangle$, where σ is a ground situation and $\phi(s)$ is the effect. Some of the (ground) actions in the ground situation σ can be determined from an explanatory diagnostic system, on which we build on, e.g., by solving a related planning problem to reconstruct unobservable events.

We are ready to give our formal definition of diagnosis:

Definition 3 (Causal Diagnosis). *Given a system* $\mathcal{DS} = \langle \mathcal{D}, \sigma, \phi(s)\rangle$, *a* diagnosis *is a causal chain relative to* $\mathcal{DS}$.

We don't need to impose a notion of minimality here since this is already implied: if α executed in σ is a cause in a diagnosis relative to a system $\mathcal{DS} = \langle \mathcal{D}, \sigma, \phi(s)\rangle$, then α is either necessary or contributes to a condition that is necessary for achieving/maintaining ϕ. Also, we can formally show that our diagnosis for a given system is unique (here $\mathcal{K}_1 = \mathcal{K}_2$ means that causal chains $\mathcal{K}_1$ and $\mathcal{K}_2$ are the same):

Theorem 1 (Uniqueness). *If* $\mathcal{K}_1$ *and* $\mathcal{K}_2$ *are two diagnoses of a system* $\mathcal{DS} = \langle \mathcal{D}, \sigma, \phi(s)\rangle$, *then* $\mathcal{K}_1 = \mathcal{K}_2$.

The proof follows directly from the uniqueness of the narrative σ and that of the output of the regression operator ρ.

Recall diagnosis of DES can provide several different traces, and consequently, each trace may bring about a different causal chain, but as the above theorem shows, for each trace our diagnosis is unique. According to our definitions above, actual causes in a causal chain and consequently a diagnosis can include inter-component communication actions, etc. As such we can model incorrect inter-component interactions and protocols in addition to malfunctioning components.

Moreover, the set of actions in a diagnosis is sufficient for the effect to hold. If $\mathcal{K}$ is a diagnosis of a system $\mathcal{DS} = \langle \mathcal{D}, \sigma, \phi(s)\rangle$, let $\sigma_{\mathcal{K}}$ be the situation obtained by performing the actions in $\mathcal{K}$ in the order they appear in σ, starting from S_0. Then we can show that:

Theorem 2 (Sufficiency). *If* $\mathcal{K}$ *is a diagnosis of a system* $\mathcal{DS} = \langle \mathcal{D}, \sigma, \phi(s)\rangle$, *then* $\mathcal{D} \models executable(\sigma_{\mathcal{K}}) \wedge \phi(\sigma_{\mathcal{K}})$.

Proof sketch (by induction on n, where n is the length of the actions in $\sigma_{\mathcal{K}}$). For the base case when $n = 1$, the situation $\sigma_{\mathcal{K}}$ and thus the causal chain $\mathcal{K}$ has only one action, say b. Thus by Definition 2, ϕ must hold after this action is performed. Also, since there are no other actions in the causal chain $\mathcal{K}$, action b must be possible to execute initially in S_0, and all conditions that are required for ϕ to hold after the execution of b must have been true initially in S_0 as well. Otherwise, since Definition 2 accounts for these conditions, some other actions ensuring the preconditions of b and/or the conditions under which b makes ϕ true must have been included in the causal chain; but this is not the case. Thus, the consequent follows for the base case.

For the inductive step, assume that the theorem holds for all situations σ_N of length $n = N$. We need to prove this for $n = N + 1$. First, note that by the inductive hypothesis, all actions up to b_N in σ_N are executable. Thus, adding one more action, say b_{N+1}, makes $\sigma_{N+1} = do(b_{N+1}, \sigma_N)$ non-executable if and only if b_{N+1} is not executable in σ_N. But this is not possible, since Definition 2 already accounts for this by requiring that all actions that ensure the conditions required for the executability of b_{N+1} (conditions that do not already follow from the initial situation S_0) must be included in the causal chain. Moreover, since b_{N+1} is the last action in σ_{N+1}, by Definition 2, ϕ must hold after it is executed in σ_N. Also, Definition 2 ensures that all those conditions which are required for ϕ to hold after an action b_{N+1} is executed in σ_N, and which do not already follow from the initial situation S_0 have been accounted for by including necessary intermediary actions in σ_N. Thus, the theorem follows. $\square$

However, perhaps somewhat surprisingly, we can show that not every action in a diagnosis is necessary for the effect to follow in the sense that removing any such action from the trace does not necessarily hamper executability or make the effect false. Let $\sigma^{\overline{a',s'}}$ denote the situation that can be obtained by executing the exact sequences of actions as in σ starting in S_0, except for action a' in situation s'.

Theorem 3. *If $\mathcal{K}$ is a diagnosis of a system $\mathcal{DS} = \langle \mathcal{D}, \sigma, \phi(s) \rangle$ and action a$'$ executed in situation s' is a cause in the causal chain $\mathcal{K}$, then:*

$$\mathcal{D} \not\models \neg(executable(\sigma^{\overline{a',s'}}) \wedge \phi(\sigma^{\overline{a',s'}})).$$

Proof (by counter-example). Consider the following example (assume that the corresponding domain theory $\mathcal{D}$ has been specified): there are three actions a_1, a_2, and b, such that a_1 and a_2 are always executable while b can only be executed if either a_1 or a_2 has been previously executed. The execution of b in an executable situation brings about the effect ϕ. Consider a trace $\sigma = do([a_1, a_2, b], S_0)$. Note that although $\{(a_1, S_0), (b, do([a_1, a_2], S_0))\}$ is a diagnosis of the system $\langle \mathcal{D}, \sigma, \phi(s) \rangle$, and therefore (a_1, S_0) is a cause in the corresponding causal chain, removing a_1 from σ does not make it non-executable, namely, $D \models executable(do([a_2, b], S_0)))$. According to our assumptions about the given actions, ϕ still follows if a_1 is omitted from the trace, namely, $D \models \phi(do([a_2, b], S_0)))$. $\square$

Thus, removing a cause a' in s' may not have any affect on $\phi(s)$ as it may be the case that a subsequent action in the trace restores the executability and/or brings about the effect, e.g. one that is currently being preempted by the actual cause.[2] In fact the counter-example in the above proof shows that unlike previous problematic accounts of causality (see Section 5), our base framework does not choose an action (a_2 in this example) as a cause when its effects are preempted by some earlier action (that of a_1 in this case).

When diagnosing problems in large domain, it is often useful to sub-divide the system under consideration into different constituents. In our framework, this can be achieved by specifying the BAT $\mathcal{D}$ for the system in terms of the specification of the BATs for its sub-theories. Thus, assuming that there are N conceptual components in the system, we have $\mathcal{D} \stackrel{\text{def}}{=} \bigcup_{i=1}^{N} \mathcal{D}_i$, where $\mathcal{D}_i$ is the BAT for the i^{th} component. Note that defining diagnosis via causal analysis allows us to take advantage of this modularity: we can show that in our framework causal analysis and hence diagnosis can be performed by examining only the relevant subset of the system specification $\mathcal{D}$. To see this formally, first we assume without loss of generality that all the actions, fluents, and constants in each $\mathcal{D}_i$ have distinct names; if not they can be renamed using the index i of the underlying component.[3] Also, we say that the set of BATs $\mathcal{D}_{\vec{\alpha}} \subseteq \mathcal{D}$ is a *block of a partition of $\mathcal{D}$ relative to narrative* $\sigma = do([\vec{\alpha}], S_0)$ if every action in the action sequence $\vec{\alpha}$ comes from the BATs in $\mathcal{D}_{\vec{\alpha}}$. Finally, we say that ϕ is a *relevant effect relative to block $\mathcal{D}_{\vec{\alpha}}$* if and only if $\mathcal{D}_{\vec{\alpha}} \models \phi(do([\vec{\alpha}], S_0))$, i.e. $\mathcal{D}_{\vec{\alpha}}$ entails that ϕ holds after $\vec{\alpha}$ has been performed starting in situation S_0. Then:

Theorem 4 (Modularity). *If $\mathcal{D}_{\vec{\alpha}}$ is a block of a partition of a system specification $\mathcal{D}$ relative to a ground narrative $\sigma = do([\vec{\alpha}], S_0)$, then for all relevant effects ϕ relative to $\mathcal{D}_{\vec{\alpha}}$ and for all $\mathcal{K}$, $\mathcal{K}$ is a causal chain of setting $\langle \mathcal{D}, \sigma, \phi(s) \rangle$ if and only if $\mathcal{K}$ is a causal chain of setting $\langle \mathcal{D}_{\vec{\alpha}}, \sigma, \phi(s) \rangle$.*

The proof for this follows straightforwardly from the structure of the SSAs for the relevant fluents, the APAs for the relevant actions, and our definition of causal chains. Our notion of diagnosis is thus computationally modular.

We believe that our theory of causality is well behaved with respect to abstraction and refinement of actions in action hierarchies. Since we define diagnosis using causality, we should be able to relate diagnosis at various levels of abstractions. To this end, we can start by specifying system behaviour using modular BATs (Gu and Soutchanski 2008) instead of our current flat representation, and show "equivalence" of causal chains at different levels. Put otherwise, our approach allows us to further benefit from the representational and computational advantages of modular BATs.

5. Discussion

As discussed earlier, there has been much work on diagnosis. In the static systems, one is concerned with the question "what is wrong with the system?" by essentially identifying a minimal set of faulty components that can explain the effect, e.g. (Reiter 1987; de Kleer, Mackworth, and Reiter 1992). These neither work for dynamic systems nor capture the cases where all components of the system function well, but interaction between them is not configured properly.

There have also been much work on fault diagnosis of DES within control theory, e.g. (Sampath et al. 1995; 1996); also see (Lamperti and Zanella 2003; Lamperti, Zanella, and Zhao 2018; Zaytoon and Lafortune 2013) for a review. In those papers, faults are considered to be unobservable events, and the diagnosis problem is concerned with determining those linearly ordered traces of events, including faults, which are consistent with observations and can explain a given observed effect based on the model of the system. AI researchers view this problem of diagnosis in dynamic systems by addressing the question "what has happened?". Such explanatory notion of diagnosis (Cordier and Thiébaux 1994; McIlraith 1998; Schumann, Pencolé, and Thiébaux 2007) where one tries to reconstruct the relevant set of histories can be defined as follows: given a system specification and some observations, determine the sequences of actions each of whose executions entails these observations. Some recent examples include (Sohrabi, Baier, and McIlraith 2010), who viewed diagnosis as preference-based planning, and (Rintanen and Grastien 2007), who reduced diagnosis to the path finding problem implemented using SAT solvers. Recently, there have been work on extending the notion of conflicts from static diagnostic algorithms to dynamic systems and DES (Grastien, Haslum, and

[2]Note that this result does not contradict our earlier claim that every action in the causal chain is either necessary or contributes to a condition that is necessary for achieving the effect. In particular, the Theorem 3 does not hold when all the trailing actions (i.e., actions in the narrative that come after the last action in the causal chain) are ignored. Consider the sequence that is composed of all the actions in the narrative up to and including the last action in the causal chain. It can be shown that removing any action that appears in the causal chain either makes it non-executable or fails to bring about the intended effect.

[3]If two BATs need to share an action, they should be grouped under the same component index.

Thiébaux 2012). This allows the transfer of the efficient conflict-based approaches used for diagnosis in static systems to dynamic DES and handles both types of diagnosis in a uniform way (Haslum and Grastien 2011).

This is where we contribute: we start with the output (i.e. the set of actions, in our case, encoded by a ground situation) of an explanatory diagnostic system. The problem for us is to search in this trace for the exact achievement causes of the given effect to filter out irrelevant events. Thus, our work can be used to extract root causes from the candidate traces (event logs) obtained by planning techniques for solving explanatory diagnosis (Cordier and Thiébaux 1994; McIlraith 1998).

There was an interesting earlier work (Bhandari, Simon, and Siewiorek 1987), where the authors presented an optimal algorithm to do probe selection in causal chains (understood differently from our paper) in the context of diagnostic programs. However, their framework depends on numerical data in the form of time required for probes, probability of errors, etc. Our notion of causality has motivation that is similar to (Gössler and Le Métayer 2015; Wang et al. 2015), who also discuss causal analysis relative to traces. However, they work with less expressive languages, and unlike us their counterfactual definition of causality suffers from the *preemption problem*, which occurs when two competing events try to achieve the same effect, and the latter of these fails to do so, as the earlier one has already achieved the effect. Unlike these, as formally shown in (Batusov and Soutchanski 2018), our definition above can correctly compute actual causes even for the more problematic examples with early preemption and over-determination that create serious difficulties for the well-known structural equations based approach developed in (Pearl 1998; 2000; Halpern and Pearl 2005; Halpern 2016). It is worth mentioning here that the above mentioned approaches to actual causality are not directly related to causal model-based diagnosis studied by the Qualitative Reasoning community in the 1990s (Weld and De Kleer 1990; Travé-Massuyès 2014).

While we propose to utilize and incorporate causal analysis to further refine outputs from diagnostic systems, we do not however discuss how this refinement can be actually done. This in part depends on the DES framework on top of which our work can be applied. We leave this for future work.

There has also been earlier work, e.g. (McIlraith et al. 2000), that deals with diagnosis in hybrid systems where the main concern is a combination of both continuous and discrete dynamic behaviours. While we do not exploit an explicit representation of time in this paper, with some effort our model can be modified to deal with hybrid systems and continuous time, e.g. by using the hybrid SC (Batusov, De Giacomo, and Soutchanski 2018) to model domain dynamics. In particular, the regression operator can be also defined in the hybrid SC. We leave this direction for future work.

Acknowledgements

We thank the reviewers for useful comments on an earlier version. This work was supported in part by the National Science and Engineering Research Council of Canada.

References

Batusov, V., and Soutchanski, M. 2017. Situation calculus semantics for actual causality. In *Proceedings of the 13th International Symposium on Commonsense Reasoning, COMMONSENSE 2017, London, UK, November 6-8, 2017*.

Batusov, V., and Soutchanski, M. 2018. Situation calculus semantics for actual causality. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.

Batusov, V.; De Giacomo, G.; and Soutchanski, M. 2018. Hybrid Temporal Situation Calculus. *ArXiv e-prints*.

Belle, V., and Levesque, H. J. 2018. Reasoning about Discrete and Continuous Noisy Sensors and Effectors in Dynamical Systems. *Artificial Intelligence* 262:189–221.

Bhandari, I. S.; Simon, H. A.; and Siewiorek, D. P. 1987. Optimal Diagnosis for Causal Chains. Technical Report CMU-CS-87-151, Carnegie Mellon University.

Cordier, M.-O., and Thiébaux, S. 1994. Event-Based Diagnosis for Evolutive Systems. In *Proceedings of the 5th International Workshop on Principles of Diagnosis, New Paltz, NY*, 64–69.

De Giacomo, G.; Reiter, R.; and Soutchanski, M. 1998. Execution monitoring of high-level robot programs. In *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), Trento, Italy, June 2-5, 1998.*, 453–465.

de Kleer, J., and Williams, B. C. 1989. Diagnosis with Behavioral Modes. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence. Detroit, MI, USA, August 1989*, 1324–1330.

de Kleer, J.; Mackworth, A. K.; and Reiter, R. 1992. Characterizing Diagnoses and Systems. *Artificial Intelligence* 56(2-3):197–222.

Gössler, G., and Le Métayer, D. 2015. A general framework for blaming in component-based systems. *Science of Computer Programming* 113, Part 3.

Grastien, A.; Haslum, P.; and Thiébaux, S. 2012. Conflict-Based Diagnosis of Discrete Event Systems: Theory and Practice. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012, Rome, Italy, June 10-14, 2012*.

Gu, Y., and Soutchanski, M. 2008. Reasoning about Large Taxonomies of Actions. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*, 931–937.

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.

Halpern, J. Y. 2016. *Actual Causality*. The MIT Press.

Haslum, P., and Grastien, A. 2011. Diagnosis as Planning: Two Case Studies. In *5th Scheduling and Planning Applications Workshop (SPARK-11)*.

Hoare, C. A. R. 1985. *Communicating Sequential Processes*. Prentice-Hall.

Lamperti, G., and Zanella, M. 2003. *Diagnosis of Active Systems: Principles and Techniques*. Springer Netherlands.

Lamperti, G.; Zanella, M.; and Zhao, X. 2018. *Introduction to Diagnosis of Active Systems*. Springer Netherlands.

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

McIlraith, S. A.; Biswas, G.; Clancy, D.; and Gupta, V. 2000. Hybrid Systems Diagnosis. In *Hybrid Systems: Computation and Control, Third International Workshop, HSCC 2000, Pittsburgh, PA, USA, March 23-25, 2000, Proceedings*, 282–295.

McIlraith, S. A. 1994. Towards a Theory of Diagnosis, Testing and Repair. In *Proceedings of the 5th International Workshop on Principles of Diagnosis (DX-94)*, 185–192.

McIlraith, S. A. 1998. Explanatory Diagnosis: Conjecturing Actions to Explain Observations. In *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), Trento, Italy, June 2-5, 1998.*, 167–179.

Pearl, J. 1998. On the definition of actual cause. Technical report, University of California Los Angeles.

Pearl, J. 2000. *Causality: Models, Reasoning, and Inference*. Cambridge University Press.

Reiter, R. 1987. A theory of diagnosis from first principles. *Artificial Intelligence* 32(1):57–95.

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

Rintanen, J., and Grastien, A. 2007. Diagnosability Testing with Satisfiability Algorithms. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, 532–537.

Sampath, M.; Sengupta, R.; Lafortune, S.; Sinnamohideen, K.; and Teneketzis, D. 1995. Diagnosability of Discrete Event Systems. *IEEE Transactions Automatic Control* 40:1555–1575.

Sampath, M.; Sengupta, R.; Lafortune, S.; Sinnamohideen, K.; and Teneketzis, D. 1996. Failure Diagnosis using Discrete-Event Models. *IEEE Trans. Contr. Sys. Techn.* 4(2):105–124.

Schumann, A.; Pencolé, Y.; and Thiébaux, S. 2007. A spectrum of symbolic on-line diagnosis approaches. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, 335–340. AAAI Press.

Sohrabi, S.; Baier, J. A.; and McIlraith, S. A. 2010. Diagnosis as planning revisited. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Twelfth International Conference, KR 2010, Toronto, Ontario, Canada, May 9-13, 2010*.

Travé-Massuyès, L. 2014. Bridging control and artificial intelligence theories for diagnosis: A survey. *Eng. Appl. of AI* 27:1–16.

Wang, S.; Geoffroy, Y.; Gössler, G.; Sokolsky, O.; and Lee, I. 2015. A Hybrid Approach to Causality Analysis. In *RV 2015 - 6th International Conference on Runtime Verification*, volume 9333 of *LNCS*.

Weld, D., and De Kleer, J. 1990. *Readings in Qualitative Reasoning about Physical Systems*. The Morgan Kaufmann series in representation and reasoning. Morgan Kaufmann.

Zaytoon, J., and Lafortune, S. 2013. Overview of fault diagnosis methods for Discrete Event Systems. *Annual Reviews in Control* 37(2):308–320.