Why am I stuck? Causal Logic Models for Token-Level Causal Reasoning

Denver Dash
Intel Corporation
Carnegie Mellon University
denver.h.dash@intel.com

Mark Voortman
University of Pittsburgh
mark@voortman.name

Martijn de Jongh
University of Pittsburgh
mad159@pitt.edu

Abstract

We present a new approach to *token-level* causal reasoning that we call *Causal Logic Models (CLMs)*. A CLM is a first-order representation that allows one to produce token-level causal explanations or predictions in domains which are too vast to generate a complete causal model. CLMs produce explanations/predictions as a dynamic sequence of mechanisms (SoMs) that chain together to propagate causal influence through time. We argue that explanations/predictions of this form are more fundamental than explanations that involve likely states of variables in conjunction with a complete causal model. We compare this approach to the causal explanations of Halpern and Pearl [2005], and show that even on relatively simple real-world physical systems, their method of generating explanations can quickly become intractable. We argue that the SoMs approach is qualitatively closer to the human causal reasoning process, and that for many real problems in AI such as diagnosing why a robot is stuck, CLMs provide more tractable and informative explanations.

1 Introduction

The human faculty of causal reasoning is a powerful tool to form hypotheses by combining limited observational data with pre-existing knowledge. This ability is essential to uncovering hidden structure in the world around us, performing scientific discovery and diagnosing problems in real time. Enabling computers to perform this kind of reasoning in an effective and general way is thus an important sub-goal toward achieving Artificial Intelligence.

The theoretical development of causality in AI has up to now primarily been based on structural equation models (SEMs) [Strotz and Wold, 1960, Simon, 1954, Haavelmo, 1943], a formalism which originated in econometrics and which is still used commonly in the economic and social sciences, and which has been adapted to AI applications by Pearl [2000], Spirtes et al. [2000] and others.

In a previous paper Anonymous [2013] argued that, despite decades of theoretical progress, causal models have not been widely used in the context of core AI applications such as robotics, and they attribute this to the fact that many typical robotics/AI use cases are in need of a means to perform *token-level* as opposed to *type-level* causal reasoning. For example, a type-level model for lung cancer might include all the possible causes, such as: *CigaretteSmoking*, *AsbestosExposure*, *GeneticFactors*, etc., whereas a token-level explanation contains only actual causes: "Bob's lung cancer was caused by that time in the 80s when he snorted asbestos." The distinction, significance and relationship between token-level and type-level causation is well-known in the philosophical literature [Eells, 1991, Kleinberg, 2012, e.g.,], and it sometimes goes by *singular* versus *general* causation, respectively [e.g., Hitchcock, 1995, Davidson, 1967].

Because models in econometrics operate on entire populations of entities, the algorithms and representations coming out of these disciplines generally rely less on token causality and more on type-level reasoning. On the contrary, many applications for causality in AI operate very much on the individual level: a robot gets stuck in a puddle of oil, a person has chest pains and we want to know the specific causes, or a person has cancer and we want to know the particular DNA mutation that is responsible. This disconnect may help explain why causal reasoning has up to now not been widely used in the context of robotics and AI applications.

Anonymous [2013] make the point that token-level explanations generated by Bayesian networks (BNs) and SEMs could be more informative if they were represented by *Sequences of Mechanisms* (SoMs) which chain together across time. By collapsing mechanisms together into single equations or CPTs, which are then used to reason about states of variables, BNs/SEMs may lose information about

particular mechanisms that were active in a particular situation. In this paper, we extend this argument further to show that when producing token-level explanations/predictions from BNs/SEMs on real-world systems, CLMs can produce hypotheses in cases where other methods, such as that of Halpern and Pearl [2005] would be intractable.

Our contributions in this paper are as follows:

1. We define *Causal Logic Models*, a new first-order representation that generalizes the representation of Anonymous [2013] to include a more realistic set of causal mechanisms.

2. We present a new algorithm based on *Mechanistic Bayesian networks*, where the CLM is cast into a special type of time-varying dynamic Bayesian network, which can then be directly used to identify the most likely SoM hypothesis given the data.

3. We show analytically and empirically how our method improves on the SoMs generated by Anonymous, and

4. We present preliminary work on how arbitrary (non-causal) first-order logical constraints can be integrated into this method to improve the applicability of this work.

In Section 2, we motivate our approach by comparing existing work on simple physical systems, in Section 3 we present our representation and reasoning in detail, in Section 4 we present our algorithm for causal reasoning with CLMs, in Section 5 we compare our representation to noisy-OR, in Section 6 we present empirical results on simulated data, and finally in Section 7 we discuss implications and in Section 8 conclusions.

2 Motivation and Previous Work

There has been some work on representation and algorithms for token-level causal reasoning. In particular, Halpern and Pearl [2005] present a fairly influential definition of *causal explanation* which uses the concept of *actual cause* from Halpern and Pearl [2001], based on functional causal models of Pearl [2000], to produce sets of variables which are deemed to be possible explanations for some evidence. We compare their approach to ours in this section.

Kleinberg [2012] also discusses token causality explicitly and presents a measure of significance for a token-level immediate cause given logical formulae which are similar syntactically to our mechanisms; however she does not present an algorithm to find optimal chains of causation which could serve as nontrivial hypotheses. Both Halpern and Pearl's and Kleinberg's approaches are fundamentally propositional in nature, so lack our ability to scale when the number of possible causes is large, as we discuss in this section.

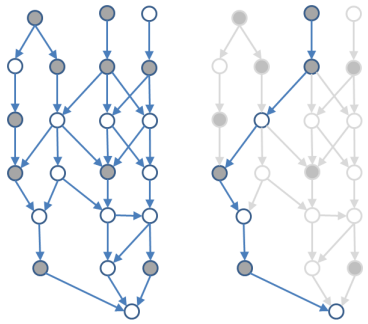
Anonymous [2013] presented an algorithm for generating SoM causal hypotheses which has motivated the present work. We provide a detailed comparison of their algorithm, which we refer to as the *State-Matching* algorithm, when we present our algorithm in Section 4. In this section we reiterate some of the points of Anonymous [2013] for completeness.

Halpern and Pearl discuss causal explanation in the context of SEMs and the concept of actual causality. Acyclic SEMs can be mapped onto the space of Bayesian networks [Druzdzel and Glymour, 1999], so with only a slight loss of generality, we consider BNs models of causality. Actual causality and causal explanation as presented by Pearl and Halpern require an existing causal model (graph) an some observations of some variables of that model.

Despite this limitation, in BNs and SEMs, something similar to token-level reasoning can be performed by instantiating variables in the model to values based on the specific instance at hand. For example, in the lung cancer case of Section 1, one might instantiate *LungCancer* to *True* and *AsbestosExposure* to *True* to indicate the token-level hypothesis that Bob's asbestos snorting adventure caused his lung cancer. However, this *state-space* approach is incomplete: being able to reason only about states and possible causes is different from creating specific hypotheses about how an event was caused. For example, it could very well be that Bob was a smoker, but the smoking was not the cause of his lung cancer. In this case, the BN model is unable to distinguish (and score) between the three hypotheses: *Smoking*→*LungCancer*, *AsbestosExposure*→*LungCancer* and *Smoking & AsbestosExposure*→*LungCancer*.

This problem becomes exacerbated in a more realistic causal model where the number of nodes would be much higher, and where nodes would combine in nontrivial ways. Consider, for example, the BN causal model of Figure 1(a) where all nodes are binary. Given some evidence we can obtain beliefs about the states of all the nodes in the graph (say dark represents *False* and light represents *True*). This representation of a type-level causal graph plus specific states does not necessarily provide us with a clear token-level picture of what is happening causally in this system.

On the other hand, a token-level explanation represented by a subgraph (such as that shown in Figure 1(b) showing the likely causal ancestors of the event of interest provides a much clearer causal picture. If one wanted to consider, for example, which manipulations might change the outcome of the event of interest, the graph of Figure 1(b) would be much more informative. This suggests one possible algorithm to achieve such a token-level explanation which consists of essentially a structure search given data, with two



(a) (b)

Figure 1: The state-space description of specific events (a) can often be difficult to interpret causally (dark nodes are believed to be false and light nodes are believed to be true). A token level explanation (b) of the same event that hypothesizes a particular causal path leading to the event is much easier to understand and visualize.

important differences: (1) the data of interest here is a single record, and (2) we have strong priors on the structure, being provided by the full type-level causal model.

As we pointed out in Anonymous [2013], starting with a "complete" BN model and looking for active sub-graphs as possible token-level hypotheses still lacks the expressiveness required for producing many token-level hypotheses. As an example, consider the following system regarding gum irritation: With some probability p_1 brushing your teeth will cause gum irritation (e.g., because you brush too vigorously). With some probability p_2 brushing your teeth will cause gum irritation to go away (because it will kill bacteria). Furthermore, let's assume that bacteria and rough brushing are latent variables. All we know is that in some token instances, brushing causes irritation, and in some it prevents it. The conditional probability distribution of *GumIrritation* would be determined by a combination of the two mechanisms. Once these mechanisms have been combined into a CPT in a family in a BN, any hypothesis generated from this model will consist of some kind of mixture of two very different mechanisms. It may be possible to construct a BN in such a way that the individual mechanisms are preserved, but the point is, the mechanisms *must* be preserved to do token-level reasoning, and therefore, the mechanisms themselves should play a prominent role in the representation used.

Halpern and Pearl [2005] deal with this issue by searching for *actual causes* given a full causal model and some evidence to explain. An actual cause is a variable which is known to be true, and satisfies various criteria deemed to be essential for causation. The concept of actual cause is then used to generate explanations as the set of variables which are not known for certain, but if they were known would be actual causes.

The difficulty in this approach becomes apparent when one tries to apply it to even relatively simple real-world physical systems. As an archetypal example of a physical causal system, consider a pool table. A typical explanation in a pool table situation, for example, given that a yellow ball has dropped in a side pocket, might be something like: "The cue ball struck the red ball causing it to strike the yellow ball, causing it to go in the side pocket". In order to apply the method of Pearl and Halpern, we must first develop a causal diagram of the pool table. One obstacle to this is the fact that a pool table is a continuous system, so in fact there are infinitely many ways balls may be positioned on the table, may collide with one another with various velocities, not to mention spin etc. However, for simplicity's sake, let us consider a more idealized pool table shown in Figure 2.

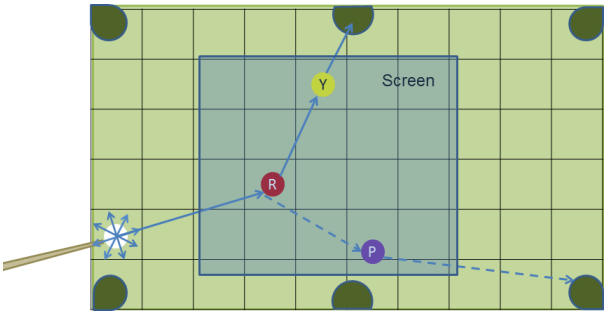


Figure 2: An idealized pool table with 6×10 possible discrete positions, 4 balls, 8 possible velocity directions and 3 possible speeds.

In this example we have 4 balls (including the cue ball), and each ball can occupy a discrete space on the table within a 6×10 rectangular grid, can have 8 possible velocity directions and 3 possible speeds. The events of interest are collisions of balls and balls dropping in pockets. Even with these drastic simplifications, it is apparent that a "complete" model of the causality of this system is hopelessly intractable. Given that the consequences of a collision depend on the locations, speeds and directions of the balls, the number of unique possible collisions is at least equal to the size of the state space which is at least $\binom{60}{4} \times 4^{24} > 10^{20}$ possible ways balls can collide with each other. Yet, in the situation shown in Figure 2, humans would have no trouble (even if some of the balls were non-observable because they were covered by a screen as shown in the figure) determining the token causality of hitting the cue ball followed by the yellow ball dropping in the side pocket. In order to reason about real but basic physical systems like these, a representation and algorithm are required that use the *observed evidence* to construct a *relevant causal model* specific to the situation at hand, rather than relying on a completely specified causal graph depicting all possible scenarios that may be observed in the system.

3 Reasoning and Representation

Causal reasoning as we define it produces hypothesized *sequences of causal mechanisms* that seek to *explain* or *predict* a set of *real or counterfactual* events which have been *observed* or *manipulated*. We therefore maintain that there are three independent dimensions to causal reasoning: *explanation/prediction*, *factual/counterfactual*, *observation/manipulation*. In this section, we look at causal explanation, prediction, counterfactuals, and manipulations. Although these types of reasoning have been discussed at length elsewhere [c.f. Pearl, 2000], here we relate these concepts to token-based causality and in Anonymous [2013] we raised several new issues that arise in this context such as the commutability between observation and manipulation.

In general, causal reasoning is the act of inferring a causal structure relating events in the past or future. The events themselves can be observed, hypothesized (i.e., latent) or manipulated. Given a causal model C and a sequence of events $\mathbf{E_1}, \mathbf{E_2}, \ldots, \mathbf{E_n}$, all causal reasoning can be cast into the problem of finding a most-likely sequence of mechanisms $\hat{S}$ given a set of information:

$$\hat{S} = \arg\max_{S} P(S|C, \mathbf{E_1}, \mathbf{E_2}, \ldots \mathbf{E_n}) \quad (1)$$

We consider sequences of events rather than one big set of events $\mathbf{E} = \cup_i \mathbf{E_i}$ because when we consider manipulation of the system, then it will sometimes be the case that manipulation does not commute with observation. So we need to preserve the sequence in which events are observed and manipulated.

Causal Explanation is the act of explaining a set of observations in terms of a set of mechanisms. This is defined by Equation 1 where the sequence of events $\mathbf{E}$ only contains observations and no manipulations. Causal Prediction is the act of predicting what sequences of cause and effect will occur in the future given evidence observed in the past. Prediction is not restricted to only inferring events in the future. In practice, the events in the past that led to the observations in the present may be relevant for predicting future variables as well, so we must perform inference on past events in order to better predict the future. In general, the distinction between explanations, predictions, and counterfactuals is somewhat arbitrary and can be combined in various ways.

The underlying representation we use to generate hypotheses is nearly equivalent to the representation of Anonymous [2013], with the exception that we explicitly assume persistence across time unless some mechanisms causes a variable to change value. Our representation consists of a collection of causal mechanisms that capture the causal interactions in a dynamic system. These causal mechanisms are formulae encoding and quantifying causal implication, i.e., what is the probability that an effect is true given that all its causes are true. The mechanisms are assumed to *work independently* of each other and the probability specificied with each formula encodes the likelihood of that mechanism causing the effect given that *all other mechanisms are absent*. We say that a formula *matches* when all its causes are present, and when these causes actually bring about the effect we say that it is *active*.

For example, if we construct a one-dimensional pool-table, we might have a set of predicates that look like the following: *At*(*Ball*, *Time*), *Moving*(*Ball*, *Direction*, *Time*), and *Collision*(*Ball*, *Ball*, *Location*, *Time*), etc. Each predicate is indexed by a discrete time index. The set of formulae would take the form, for example: p: $Moving(B_1, D_1, T_1) \wedge Collision(B_1, B_2, T_1) \longrightarrow Moving(B_1, D_2, T_2) \wedge Moving(B_2, D_1, T_2)$, where B_i are balls, D_i are directions, and T_i are times (we have shortened this formulae down for clarity, one would need to include predicates relating time steps and directions to each other, but they do not add substantially to the explanation). The probability p associated with the formulae is used to provide probabilistic semantics to the model, which we discuss in more details below.

Formulae like the above express causal relationships that govern the system of interest with a certain probability, specified by a probability p. We assume that times present on the left side of a formula occur at an earlier time than all times on the right. For simplicity, we assume that causal formulae are drawn from the subset of first-order logic that includes formulae containing a conjunction of (possibly negated) causes relating to (possibly negated) effects. It is assumed that the state of a predicate persists over time if there are no mechanisms actively influencing it. A causal formula is called *promoting* if the effect predicate is not negated. A causal formula is called *inhibiting* if it is not promoting.

If no formula matches a predicate we assume that it will maintain its state. If there is exactly one formula that matches, the probability of the effect is simply defined by the probability associated with the formula; however if a set $\mathbf{F}$ of several mechanisms are active for the same effect, we use a combination rule to produce the probability of the effect. When all mechanisms $F_i \in \mathbf{F}$ being combined are of the same type (promoting or inhibiting), then we apply the well-known noisy-or [Good, 1961, Peng and Reggia, 1986]:

$$P(E|\mathbf{F}) = 1 - \prod_{F_i \in \mathbf{F}} (1 - p_i),$$

where p_i is the probability associated with formula F_i. When $\mathbf{F}$ is made up of inhibiting formulae, then the noisy-or combination determines the complement of E. In the case where we are combining inhibiting formulae ($\mathbf{F}^-$)

with promoting formulae ($\mathbf{F}^+$), we average the noisy-or combination of all promoters with the complement of the noisy-or combination of all inhibitors:

$$P(E|\mathbf{F}^+, \mathbf{F}^-) = \frac{P(E|\mathbf{F}^+) + 1 - P(\neg E|\mathbf{F}^-)}{2}.$$

To tie all these assumptions together into an algorithm capable of causal reasoning, we convert a CLM into a special type of BN that can provide us with token level explanations/predictions by evaluating the likely states of special mechanism variables. As we make observations about our system, these formulae provide possible explanations that can tie those observations together causally. In Section 4 we present more detail about the special BN and an algorithm that accomplishes this by searching for a structure that explains all the observations.

Probabilistic first-order representations have been widely studied in the past decade in the context of graphical models, giving rise to an entire sub-field of AI called *statistical relational AI*[1] with many variants. Many of these variants might be adaptable to produce mechanistic interpretations simply by demanding that rules are comprised of isolated mechanisms, and by producing causal hypotheses that only include lists of ground formulae which relate predicates over time.

One representation in particular, the *CP-Logic* formalism of Vennekens et al. [2009] combines logic programming with causality, and they explicitly discuss this representation in the context of counterfactuals and manipulation [Vennekens et al., 2010]. Our representation is very similar to CP-Logic, with temporal rules and slightly different syntax. To our knowledge CP-Logic has not been used for token-level explanation/prediction previously.

4 Algorithm

In this section we present an algorithm for causal reasoning based on SoMs that is an improved version of an earlier version presented in a previous paper [Anonymous, 2013]. We will briefly describe that algorithm and then show how the new version is an improvement.

The old algorithm is displayed in Figure 3 and takes as input (a) a knowledge-base of mechanisms and a set of observations, and outputs a hypothesis (d) about which mechanisms were active at what time. The basic idea is to convert a causal model and a set of evidence into a BN (b), find (c) the Most Probable Explanation (MPE) for that evidence, and select all the formulae that are consistent with the MPE states to recover (d) the final sequence of grounded mechanisms. We will call this last step pruning because it effectively removes all formulae from the

[1] A good overview of this field is provided by Getoor and Taskar [2007].

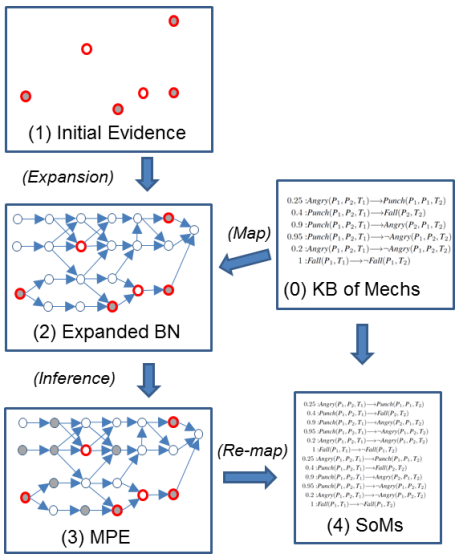


Figure 3: A high-level flow of the SoM algorithm of [Anonymous, 2013].

explanation/prediction that are not relevant. We will now describe each of the three steps in detail.

A set of formulae and a set of evidence are converted into a Bayesian network in the following way. First, for each evidence predicate all the formulae that (partially) match that predicate are instantiated in all possible ways. Consequently, this results in an expanded set of predicates that can then be used to find additional formulae that match, just like in the first step. We call formulae that are added to the model instantiated formulae (because all their free variables are bound). This process continues until no more predicates can be added, and to make this finite, time bounds are used. The CPTs are constructed by following the procedure described in Section 3.

The Most Probable Explanation is a state assignment for all variables that is the most likely out of all possible assignments. This is a well known problem and many algorithms have been developed to efficiently find solutions.

The pruning step selects all the formulae that are consistent with the MPE output. Conceptually speaking, we could iterate through each family of nodes in the BN, and try to instantiate the assignment for the family in each formulae. If all the variables in the formula are bound, we can assess the thruth value of the formula (each predicate in the formula is either true or false because of the MPE step). If the formula is true it will be part of the causal explanation/prediction. If the formula is false or not all variables are bound it will not be part of the causal explanation/prediction.

It is important to emphasize that the SoM algorithm in Figure 3 searches for the MPE states of variables, which are then used to find a set of consistent formulae. We will also call this the State-Matching algorithm. The obvious

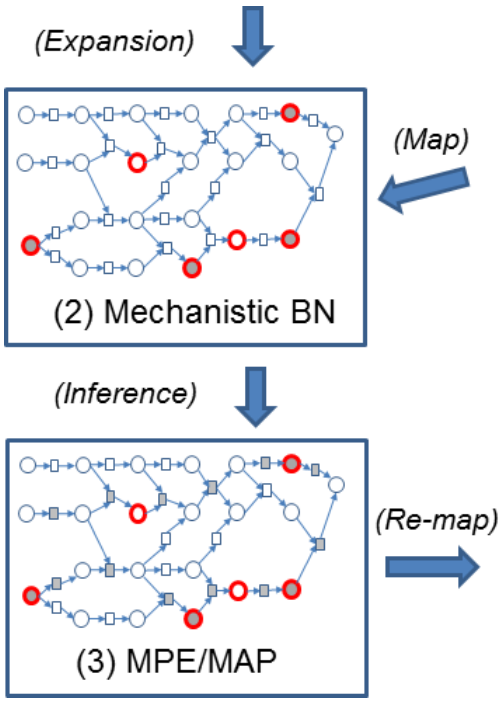


Figure 4: Steps 2 and 3 of the Mechanistic BN SoM algorithm. The active sub-paths are directly apparent by following the path of active mechanism nodes. The remapping step is trivial as each mechanism node is mapped to a ground formula.

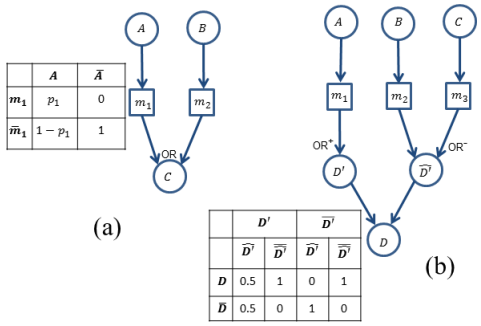


Figure 5: (a) A noisy-or BN representation and (b) the noisy-OR$^+$/OR$^-$.

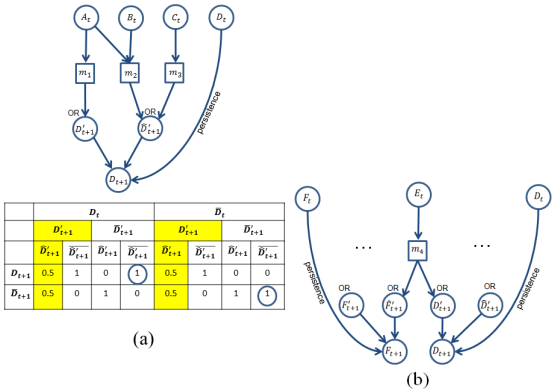


Figure 6: A mechanistic Bayesian network fragment for (a) a simple effect D_{t+1}, and (b) a conjunction of effects $D_{t+1} \wedge \neg F_{t+1}$. These fragments generalize the noisy-OR$^+$/OR$^-$ to allow for multiple parents per mechanism and to model persistence over time.

disadvantage of this approach is that it will select all formulae that are consistent with those variables, even if only one formulae (mechanism) would perfectly explain a set of observations. Therefore, our new algorithm explicitly includes mechanism nodes in the Bayesian network and performs MPE on these variables thereby *directly selecting a set of active mechanisms*. This process is displayed in Figure 4 where the mechanisms are displayed as squares (notice their absence Figure 3) and when they are active there are grayed out. This directly presents you with a sequence of active mechanisms.

5 Comparison to Noisy-OR

In this section we compare noisy-OR to the graph structures we use in our mechanistic approach. Figure 5(a) shows a noisy-OR implementation in a Bayesian network where A and B are causes, C is the effect, and m_1 and m_2 are mechanisms. In a deterministic OR one active cause would with certainty bring about the effect, however, in noisy-OR the mechanisms act as a probabilistic relay where the cause only probabilistically brings about the effect. This is implemented using the CPT in Figure 5(a), e.g., if A is true then m_1 is only true with probability p_1. The mechanisms then feed into a standard OR node. Noisy-ORs could also have a leak probability associated with them, which is defined as the probability of an effect being true with all its causes being false. This could simply be implemented as a separate mechanism for which its parent is always true.

The simplest formulae in our approach, those that have only one cause and one effect, map directly onto noisy-OR. However, if two of these simple formulae have opposing incluences on an effect, i.e., one is a promotor and the other an inhibitor, it is necessary to combine both influences which cannot be done directly with noisy-OR. It is possible, however, to use one noisy-OR for all promoting formulae and one noisy-OR for all inhibiting formulae, which then have to be combined to produce an overall conditional probability distribution for the effect. The graphical structure of this process is shown in Figure 5(b) where m_1 is a promotor and m_2 and m_3 are inhibitors. This approach was described by Zagorecki [2010] and named it noisy-OR$^+$/OR$^-$. He also specified the general form for the combination rule in the following way. If noisy-OR$^+$ is true and noisy-OR$^-$ is false then the effect is true and, conversely, if noisy-OR$^+$ is false and noisy-OR$^-$ is true then the effect is false. When both of them are true or both

of them are false the user has to decide the preferred way of combining them. In our approach, when both noisy-ORs are true we simply use a uniform distribution over the effect states. When both noisy-ORs are false there were no causes for the effect and in that case we assume persistence, so that means the effect variable should also have itself from the previous time step as a parent, which is illustrated in Figure 6(a).

In the previous discussion we assumed that a formula would have only one cause and one effect, but in general each formula can have multiple causes and multiple effects. Figure 6(a) shows a mechanism (m_2) with multiple parents and Figure 6(b) shows a mechanism with multiple children (m_4), although a mechanism could have both at the same time as well. This deviates from noisy-OR and the reason is that in noisy-OR interactions between pairs of variables (cause and effect) are specified, whereas in our approach we define interactions between sets of variables through causal formulae. We are not aware of any other work that generalizes noisy-OR in this way, and we suspect the reason is that it is complicated to define all the joint effects of two variables and more on an effect unless you express it by using formulae.

In the next section we show that our algorithm produces plausible causal explanations/predictions of simulated observations generated from a causal model. Although an explanation/prediction found by our algorithm consists of a set of mechanisms instead of just a set of states, which is an improvement over existing algorithms, it always includes all the mechanisms that are consistent with the MPE states. It is thus possible that a subset of these mechanisms constitute an even better explanation in terms of Occam's Razor; some formulae may be removed from the explanation/prediction leading to fewer parameters in the model and retaining (almost) all the explanatory power. Thus there may be room for improvement of our first SoM algorithm.

6 Experiments

In this section we present an evaluation of our algorithms. The main idea is to start out with a full (ground-truth) causal explanation, and present parts of this explanation as evidence to the algorithms to fill in the gaps. More specifically, we constructed a causal model from which we generated a set of SoMs. For each of the SoMs we then selected a set of predicates that were presented to our causal explanation algorithm to recover the original SoMs. The reconstructed explanations were evaluated by using the precision-recall curve (PR-curve) as a measure of performance.

We examined the performance of the algorithms on two levels: 1) recovering exact matches, requiring all the recovered formulae to exactly match the formulae in the original SoMs, and 2) variable-time matches, where errors are allowed in the time variable, i.e., having the recovered formula occur earlier or later than in the correct SoMs.

6.1 The Airport Model

We evaluated the performance of our algorithms by using the *Airport* model. This model describes several events at an airport, such as collisions, explosions, and terrorist threats. We model aircraft (A), vehicles (V), and time (T).

We defined a set of formulae that link several events together to form SoMs, e.g., an aircraft colliding with a tug vehicle might cause a fuel leak to occur, possibly leading to an explosion. Here are the main formulae:

$$
\begin{aligned}
0.01 &: \mathit{SameLocation}(A_1, V_1, T_1) \longrightarrow \mathit{Collision}(A_1, V_1, T_2) \\
0.1 &: \mathit{Collision}(A_1, V_1, T_1) \longrightarrow \mathit{FuelLeak}(T_2) \\
0.05 &: \mathit{FuelLeak}(T_1) \longrightarrow \mathit{Explosion}(T_2) \\
0.2 &: \mathit{MaintenanceLapse}(A_1, T_1) \longrightarrow \mathit{MechDefect}(A_1, T_2) \\
0.005 &: \mathit{MechDefect}(A_1, T_1) \longrightarrow \mathit{Explosion}(T_2) \\
0.01 &: \mathit{Terrorist}(T_1) \longrightarrow \mathit{Threat}(T_2) \\
0.01 &: \mathit{Terrorist}(T_1) \longrightarrow \mathit{Bomb}(T_2) \\
0.95 &: \mathit{Bomb}(T_1) \longrightarrow \mathit{Explosion}(T_2)
\end{aligned}
$$

6.2 Methodology

To evaluate and compare the performance of our algorithms we used the following procedure:

1. Use the airport model to generate 250 SoMs.
2. For each SoMs select a subset of predicates to present to the algorithm. These included *Explosion*, *Threat*, *SameLocation*, and *MaintenanceLapse*. The original SoMs were stored for evaluation.
3. Run the algorithms on the samples.
4. Calculate the PR-curve using the original SoMs and the SoMs constructed by the algorithm. SoMs are compared up until the first explosion in the orginal SoMs. We calculate 2 types of Precision/Recall scores: 1) *Exact Matches*: formulae in the recovered SoMs have to exactly match the original SoMs, and 2) *Variable-Time Matches*: formulae from recovered SoMs are allowed to occur earlier or later than in the original SoMs.

The predicates that were available for selection as starting predicate were *SameLocation*, *MaintenanceLapse*, and *Terrorist*.

We ran our evaluation procedure three times, varying the number of starting predicates from the set $[1, 2, 3]$ at each run.

6.3 Results

Figure 7 shows the precision-recall results for the two algorithms, the three runs, and for the two levels of comparison of our experiment. In some cases processed samples would result in identical precision-recall pairs. In our figure, a larger dot size corresponds to a larger number of samples that have the exact same precision-recall outcome.

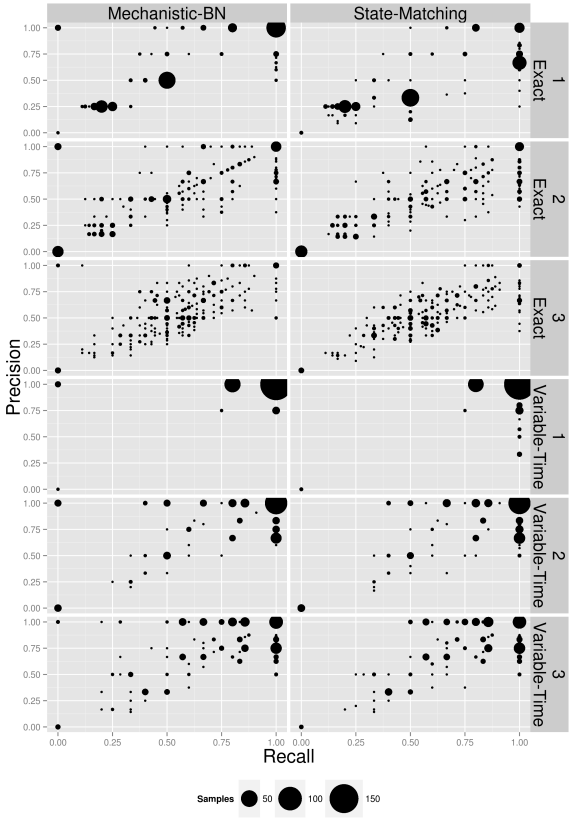


Figure 7: Precision-Recall results for Mechanistic-BN (left) vs State-Matching (right), exact matching (top) vs variable-time matching (bottom), for runs with 1,2, or 3 starting predicates

We found that in any of the examined cases (number of starting predicates vs. exact match/variable-time match) the algorithm was able to achieve reasonable Precision/Recall scores for the majority of the samples, scoring extremely well on the variable-time matching criterion and adequately on the exact matching criterion. In cases where our algorithm did poorly, visual inspection showed that little or no informative evidence was actually presented to the algorithm. In those cases the algorithm picked the mechanism with the highest prior probability, as one would expect. We also found, that due to our persistence mechanism our algorithms scored lower on the exact matching criterion. Figure 8 shows an example of a mismatch due to persistence.

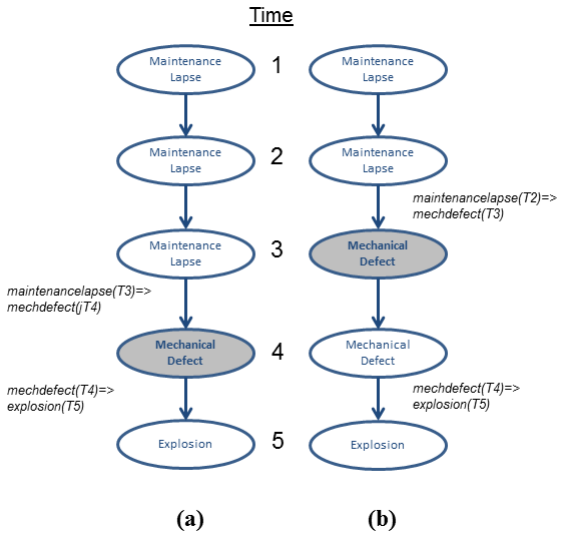


Figure 8: An example where precision on ground formulae suffers despite the algorithm essentially having the correct hypothesis.

Comparing the results of the two algorithms, the Mechanistic-BN algorithm seems to be performing slightly better than the State-Matching algorithm, but we found it ran slower.

7 Discussion

Currently our approach does not support imposing logical constraints, for example, an airplane can only be in one location at one time. We intend to implement logical constraints and in this section we outline what steps would be required. In fact, it is rather straightforward to implement logical OR and AND constraints, which can then be used to implement more complex contraints. As an example consider implementing a logical OR constraint on two variables A and B. This can be achieved by using an auxiliary variable C that results in the graphical structure shown in Figure 9, where C is a deterministic OR node that is set to true. This implies that when A is observed to be false, B automatically because true because of propagated evidence. Using this approach it is possible to implement more elaborate constraints, where OR and AND nodes can feed into other OR and AND nodes and using negation when appropriate. Note that this will never result in a cycle because the nodes are always added as children of existing predicates.

This work takes type-level information as input and outputs token-level hypotheses. An interesting topic for future research is to feedback the token-level hypotheses into the type-level model to allow for abstraction. Song et al. [2009] presented an algorithm similar to this which took individual token-level instances as input and learned time-varying Dy-

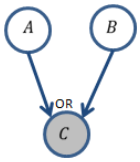


Figure 9: An OR constraint between node A and B is implemented through an auxiliary OR node C that is set to true.

namic Bayesian networks based on assumptions of smoothness across token-level instances which might make a good starting point for this goal.

8 Conclusions

In this paper we presented a new approach to token-level causal reasoning that we call *Sequences Of Mechanisms (SoMs)*, which models causal interactions not as sequences of states of variables causing one another, but rather as a dynamic sequence of active mechanisms that chain together to propagate causal influence through time. This has the advantage of finding explanations that only contain mechanisms that are responsible for an outcome, instead of just knowing a set of variables that constitute many mechanisms that all could be responsible, which is the case for BNs. We presented an improvement on a previous algorithm to discover SoMs that directly searches for active mechanisms. We showed empirically that our algorithm produces plausible and better causal explanations of simulated observations generated from a causal model.

References

Identity Anonymous. Sequences of mechanisms for causal reasoning in Artificial Intelligence. 2013. Under Submission to IJCAI-2013.

D. Davidson. Causal relations. *The Journal of Philosophy*, 64(21):691–703, 1967.

Marek J. Druzdzel and Clark Glymour. Causal inferences from databases: Why universities lose students. In Clark Glymour and Gregory F. Cooper, editors, *Computation, Causation, and Discovery*, pages 521–539, Menlo Park, CA, 1999. AAAI Press.

Ellery Eells. *Probabilistic Causality*. CUP, 1991.

Lise Getoor and Ben Taskar, editors. *Introduction to Statistical Relational Learning*. Adaptive Computation and Machine Learning. The MIT Press, 2007.

I.J. Good. A causal calculus. *British Journal of Philosophy of Science*, pages 305–318, 1961.

Trygve Haavelmo. The statistical implications of a system of simultaneous equations. *Econometrica*, 11(1):1–12, January 1943.

Joseph Halpern and Judea Pearl. Causes and explanations: A structural-model approach — part 1: Causes. In *Proceedings of the Seventeenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 194–202, San Francisco, CA, 2001. Morgan Kaufmann.

Joseph Y. Halpern and Judea Pearl. Causes and explanations: A structural-model approach. part ii: Explanations. *The British Journal for the Philosophy of Science*, 56(4):889–911, December 2005.

Christopher Read Hitchcock. The mishap at reichenbach fall: Singular vs. general causation. *Philosophical Studies*, 78:257–291, 1995.

Samantha Kleinberg. *Causality, Probability, and Time*. 2012.

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

Yun Peng and James A. Reggia. Plausibility of Diagnostic Hypotheses: The Nature of Simplicity. In *Proceedings of the 5th National Conference on AI (AAAI-86)*, pages 140–145, 1986.

Herbert A. Simon. Spurious correlation: A causal interpretation. *Journal of the American Statistical Association*, 49(267):467–479, September 1954.

Le Song, Mladen Kolar, and Eric Xing. Time-Varying Dynamic Bayesian Networks. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1732–1740. 2009.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. The MIT Press, Cambridge, MA, second edition, 2000.

Robert H. Strotz and H.O.A. Wold. Recursive vs. nonrecursive systems: An attempt at synthesis; Part I of a triptych on causal chain systems. *Econometrica*, 28(2):417–427, April 1960.

Joost Vennekens, Marc Denecker, and Maurice Bruynooghe. Cp-logic: A language of causal probabilistic events and its relation to logic programming. *Theory and Practice of Logic Programming*, 9(3):245–308, 2009.

Joost Vennekens, Maurice Bruynooghe, and Marc Denecker. Embracing events in causal modelling: Interventions and counterfactuals in cp-logic. In *The 12th European Conference on Logics in Artificial Intelligence (JELIA-2010*, pages 313–325, 2010.

Adam Zagorecki. *Bayesian networks: knowledge elicitation and inference*. PhD thesis, School of Information Sciences, Pittsbugh, PA, USA, 2010.