

The Computational Complexity of Structure-Based Causality

Gadi Aleksandrowicz

*IBM Research Lab
Haifa, Israel*

GADIA@IL.IBM.COM

Hana Chockler

*Department of Informatics
King's College
London, UK*

HANA.CHOCKLER@KCL.AC.UK

Joseph Y. Halpern

*Computer Science Department
Cornell University, Ithaca, NY 14853, USA*

HALPERN@CS.CORNELL.EDU

Alexander Ivrii

*IBM Research Lab
Haifa, Israel*

ALEXI@IL.IBM.COM

Abstract

Halpern and Pearl introduced a definition of actual causality; Eiter and Lukasiewicz showed that computing whether $X = x$ is a cause of $Y = y$ is NP -complete in binary models (where all variables can take on only two values) and Σ_2^P -complete in general models. In the final version of their paper, Halpern and Pearl slightly modified the definition of actual cause, in order to deal with problems pointed out by Hopkins and Pearl. As we show, this modification has a nontrivial impact on the complexity of computing whether $\vec{X} = \vec{x}$ is a cause of $Y = y$. To characterize the complexity, a new family D_k^P , $k = 1, 2, 3, \dots$, of complexity classes is introduced, which generalizes the class D^P introduced by Papadimitriou and Yannakakis (D^P is just D_1^P). We show that the complexity of computing causality under the updated definition is D_2^P -complete.

Chockler and Halpern extended the definition of causality by introducing notions of *responsibility* and *blame*, and characterized the complexity of determining the degree of responsibility and blame using the original definition of causality. Here, we completely characterize the complexity using the updated definition of causality. In contrast to the results on causality, we show that moving to the updated definition does not result in a difference in the complexity of computing responsibility and blame.

1. Introduction

There have been many attempts to define *causality* going back to Hume (1739), and continuing to the present (see, e.g., (Collins, Hall, & Paul, 2004; Pearl, 2000) for some recent work). The standard definitions of causality are based on counterfactual reasoning. Very roughly speaking, this means that A is a cause of B if (counterfactually) had A not occurred, then B would not have occurred (although, in fact, both A and B did occur). In this paper, we focus on one such definition, due to Halpern and Pearl, that has proved quite influential recently.

The definition was originally introduced in 2001 (Halpern & Pearl, 2001), but then modified in the final journal version (Halpern & Pearl, 2005a) to deal with problems pointed out by Hopkins

and Pearl (2003). (For ease of reference, we call these definitions “the original HP definition” and “the updated HP definition” in the sequel.) In general, what can be a cause in both the original HP definition and the updated definition is a conjunction of the form $X_1 = x_1 \wedge \dots \wedge X_k = x_k$, abbreviated $\vec{X} = \vec{x}$; what is caused can be an arbitrary Boolean combination φ of formulas of the form $(Y = y)$. This should be thought of as saying that the fact that $X_1 = x_1$ and \dots and $X_k = x_k$ is the cause of φ being true. As shown by Eiter and Lukasiewicz (2002) and Hopkins (2001), under the original HP definition, we can always take causes to be single conjuncts, that is, expressions of the form $X = x$ (also referred to as *singletons* through the paper). However, as shown by Halpern (2008), this is not the case for the updated HP definition.

Using the fact that causes can be taken to be single conjuncts, Eiter and Lukasiewicz (2002) showed that deciding causality (i.e., deciding whether $X = x$ is a cause of φ) is *NP*-complete in binary models (where all variables can take on only two values) and Σ_2^P -complete in general models.¹ As we show here, this is no longer the case for the updated HP definition. Indeed, we completely characterize the complexity of causality for the updated HP definition. To do so, we introduce a new family of complexity classes that may be of independent interest. Papadimitriou and Yannakakis (1984) introduced the complexity class D^P , which consists of all languages L_3 such that there exists a language L_1 in *NP* and a language L_2 in *co-NP* such that $L_3 = L_1 \cap L_2$. We generalize this by defining D_k^P to consist of all languages L_3 such that there exists a language $L_1 \in \Sigma_k^P$ and a language $L_2 \in \Pi_k^P$ such that $L_3 = L_1 \cap L_2$.²

Since Σ_1^P is *NP* and Π_1^P is *co-NP*, D_1^P is Papadimitriou and Yannakakis’s D^P . We then show that deciding causality under the updated HP definition is D_2^P complete. Papadimitriou and Yannakakis (1984) showed that a number of problems of interest were D^P complete, both for binary and general causal models. To the best of our knowledge, this is the first time that a natural problem has been shown to be complete for D_2^P .

Although, in general, causes may not be single conjuncts, as observed by Halpern (2008), in many cases (in particular, in all the standard examples studied in the literature), they are. In an effort to understand the extent to which the difficulty in deciding causality stems from the fact that causes may require several conjuncts, we consider what we call the *singleton cause* problem; that is, the problem of deciding if $X = x$ is a cause of φ (i.e., where there is only a single conjunct in the cause). We show that the singleton cause problem is simpler than the general causality problem (unless the polynomial hierarchy collapses): it is Σ_2^P complete for both binary and general causal models. Thus, if we restrict to singleton causes (which we can do without loss of generality under the original HP definition), the complexity of deciding causality in general models is the same under the original and the updated HP definition, but in binary models, it is still simpler under the original HP definition.

Causality is a “0–1” concept; $\vec{X} = \vec{x}$ is either a cause of φ or it is not. Now consider two voting scenarios: in the first, Mr. G beats Mr. B by a vote of 11–0. In the second, Mr. G beats Mr. B by a vote of 6–5. According to both the original and the updated HP definition, all the people who voted for Mr. G are causes of him winning. While this does not seem so unreasonable, it does not capture the intuition that each voter for Mr. G is more critical to the victory in the case of the 6–5 vote than in the case of the 11–0 vote. The notion of *degree of responsibility*, introduced by Chockler and Halpern (2004), does so. The idea is that the degree of responsibility of $X = x$ for φ is $1/(k + 1)$,

1. Note that we said “deciding whether $X = x$ is a cause of φ ”; in general, there may be more than one cause of φ .

2. $D^P = BH_2$; that is, D^P is, by definition, level 2 in the *Boolean hierarchy* (Chang & Kadin, 1996). However, it is not hard to show that D_k^P is not in the Boolean hierarchy for $k > 1$.

where k is the least number of changes that have to be made in order to make $X = x$ critical. In the case of the 6–5 vote, no changes have to be made to make each voter for Mr. G critical for Mr. G’s victory; if he had not voted for Mr. G, Mr. G would not have won. Thus, each voter has degree of responsibility 1 (i.e., $k = 0$). On the other hand, in the case of the 11–0 vote, for a particular voter to be critical, five other voters have to switch their votes; thus, $k = 5$, and each voter’s degree of responsibility is $1/6$. This notion of degree of responsibility has been shown to capture (at a qualitative level) the way people allocate responsibility (Gerstenberg & Lagnado, 2010; Lagnado, Gerstenberg, & Zultan, 2013).

Chockler and Halpern further extended the notion of degree of responsibility to *degree of blame*. Formally, the degree of blame is the expected degree of responsibility. This is perhaps best understood by considering a firing squad with ten excellent marksmen. Only one of them has live bullets in his rifle; the rest have blanks. The marksmen do not know which of them has the live bullets. The marksmen shoot at the prisoner and he dies. The only marksman that is the cause of the prisoner’s death is the one with the live bullets. That marksman has degree of responsibility 1 for the death; all the rest have degree of responsibility 0. However, each of the marksmen has degree of blame $1/10$. The complexity of determining the degree of responsibility and blame using the original definition of causality was completely characterized (Chockler & Halpern, 2004; Chockler, Halpern, & Kupferman, 2008). Again, we show that changing the definition of causality affects the complexity, and completely characterize the complexity of determining the degree of responsibility and blame with the updated definition.

The rest of this paper is organized as follows. In Section 2, we review the relevant definitions of causality. In Section 3, we briefly review the relevant definitions from complexity theory and define the complexity classes D_k^P . In Section 4 we prove our results on complexity of causality. We conclude in Section 6. Some proofs are deferred to the appendices.

2. Causal Models and Causality: A Review

In this section, we review the details of Halpern and Pearl’s definition of causal models and causality, describing both the original definition and the updated definition. This material is largely taken from (Halpern & Pearl, 2005a), to which we refer the reader for further details.

2.1 Causal Models

A *signature* is a tuple $\mathcal{S} = \langle \mathcal{U}, \mathcal{V}, \mathcal{R} \rangle$, where \mathcal{U} is a finite set of *exogenous* variables, \mathcal{V} is a finite set of *endogenous* variables, and \mathcal{R} associates with every variable $Y \in \mathcal{U} \cup \mathcal{V}$ a finite nonempty set $\mathcal{R}(Y)$ of possible values for Y . Intuitively, the exogenous variables are ones whose values are determined by factors outside the model, while the endogenous variables are ones whose values are ultimately determined by the exogenous variables. A *causal model* over signature \mathcal{S} is a tuple $M = \langle \mathcal{S}, \mathcal{F} \rangle$, where \mathcal{F} associates with every endogenous variable $X \in \mathcal{V}$ a function F_X such that $F_X : (\times_{U \in \mathcal{U}} \mathcal{R}(U) \times (\times_{Y \in \mathcal{V} \setminus \{X\}} \mathcal{R}(Y))) \rightarrow \mathcal{R}(X)$. That is, F_X describes how the value of the endogenous variable X is determined by the values of all other variables in $\mathcal{U} \cup \mathcal{V}$. If $\mathcal{R}(Y)$ contains only two values for each $Y \in \mathcal{U} \cup \mathcal{V}$, then we say that M is a *binary causal model*.

We can describe (some salient features of) a causal model M using a *causal network*. A causal network is a graph with nodes corresponding to the random variables in \mathcal{V} and an edge from a node labeled X to one labeled Y if F_Y depends on the value of X . Intuitively, variables can have a causal effect only on their descendants in the causal network; if Y is not a descendant of X , then a change

in the value of X has no affect on the value of Y . For ease of exposition, we restrict attention to what are called *recursive* models. These are ones whose associated causal network is a directed acyclic graph (that is, a graph that has no cycle of edges). Actually, it suffices for our purposes that, for each setting \vec{u} for the variables in \mathcal{U} , there is no cycle among the edges of the causal network. We call a setting \vec{u} for the variables in \mathcal{U} a *context*. If M is a recursive causal model, then there is always a unique solution to the equations in M (given by **false**), in a given context.

The equations determined by $\{F_X : X \in \mathcal{V}\}$ can be thought of as representing processes (or mechanisms) by which values are assigned to variables. For example, if $F_X(Y, Z, U) = Y + U$ (which we usually write as $X = Y + U$), then if $Y = 3$ and $U = 2$, then $X = 5$, regardless of how Z is set. This equation also gives counterfactual information. It says that, in the context $U = 4$, if Y were 4, then X would be 8, regardless of what value X and Z actually take in the real world. That is, if $U = 4$ and the value of Y were set to 4 (regardless of its actual value), then the value of X would be 8.

While the equations for a given problem are typically obvious, the choice of variables may not be. Consider the following example due to Hall (2004), showing that the choice of variables influences the causal analysis. Suppose that Suzy and Billy both pick up rocks and throw them at a bottle. Suzy's rock gets there first, shattering the bottle. Since both throws are perfectly accurate, Billy's would have shattered the bottle had Suzy not thrown.

In this case, a naive model might have an exogenous variable U that encapsulates whatever background factors cause Suzy and Billy to decide to throw the rock (the details of U do not matter, since we are interested only in the context where U 's value is such that both Suzy and Billy throw), a variable ST for Suzy throws ($ST = 1$ if Suzy throws, and $ST = 0$ if she doesn't), a variable BT for Billy throws, and a variable BS for bottle shatters. In the naive model, whose graph is given in Figure 1, BS is 1 if one of ST and BT is 1.

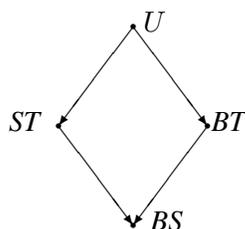


Figure 1: A naive model for the rock-throwing example.

This causal model does not distinguish between Suzy and Billy's rocks hitting the bottle simultaneously and Suzy's rock hitting first. A more sophisticated model might also include variables SH and BH , for Suzy's rock hits the bottle and Billy's rock hits the bottle. It is immediate from the equations that BS is 1 iff one of SH and BH is 1. However, now, SH is 1 if ST is 1, and $BH = 1$ if $BT = 1$ and $SH = 0$. Thus, Billy's throw hits if Billy throws *and* Suzy's rock doesn't hit. This model is described by the following graph, where we implicitly assume a context where Suzy throws first, so there is an edge from SH to BH , but not one in the other direction (and omit the exogenous variable).

Given a causal model $M = (\mathcal{S}, \mathcal{F})$, a (possibly empty) vector \vec{X} of variables in \mathcal{V} , and a vector \vec{x} of values for the variables in \vec{X} , we define a new causal model, denoted $M_{\vec{X} \leftarrow \vec{x}}$, which is identical to M , except that the equation for the variables \vec{X} in \mathcal{F} is replaced by $\vec{X} = \vec{x}$. Intuitively, this is the

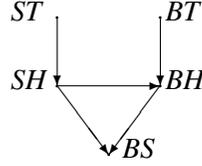


Figure 2: A better model for the rock-throwing example.

causal model that results when the variables in \vec{X} are set to \vec{x} by some external action that affects only the variables in \vec{X} (and overrides the effects of the causal equations). For example, if M is the more sophisticated model for the rock-throwing example, then $M_{ST \leftarrow 0}$ is the model where Suzy doesn't throw.

Given a signature $\mathcal{S} = (\mathcal{U}, \mathcal{V}, \mathcal{R})$, a formula of the form $X = x$, for $X \in \mathcal{V}$ and $x \in \mathcal{R}(X)$, is called a *primitive event*. A *basic causal formula* has the form $[Y_1 \leftarrow y_1, \dots, Y_k \leftarrow y_k]\varphi$, where

- φ is a Boolean combination of primitive events;
- Y_1, \dots, Y_k are distinct variables in \mathcal{V} ; and
- $y_i \in \mathcal{R}(Y_i)$.

Such a formula is abbreviated as $[\vec{Y} \leftarrow \vec{y}]\varphi$. The special case where $k = 0$ is abbreviated as φ . Intuitively, $[Y_1 \leftarrow y_1, \dots, Y_k \leftarrow y_k]\varphi$ says that φ holds in the counterfactual world that would arise if Y_i is set to y_i , for $i = 1, \dots, k$. A *causal formula* is a Boolean combination of basic causal formulas.

A causal formula φ is true or false in a causal model, given a *context*. We write $(M, \vec{u}) \models \varphi$ if φ is true in causal model M given context \vec{u} . $(M, \vec{u}) \models [\vec{Y} \leftarrow \vec{y}](X = x)$ if the variable X has value x in the unique (since we are dealing with recursive models) solution to the equations in $M_{\vec{Y} \leftarrow \vec{y}}$ in context \vec{u} (i.e., the unique vector of values for the exogenous variables that simultaneously satisfies all equations $F_Z^{\vec{Y} \leftarrow \vec{y}}$, $Z \in \mathcal{V} - \vec{Y}$, with the variables in \mathcal{U} set to \vec{u}). We extend the definition to arbitrary causal formulas in the obvious way.

2.2 Causality

We now review the updated HP definition of causality.

Definition 2.1 $\vec{X} = \vec{x}$ is a cause of φ in (M, \vec{u}) if the following three conditions hold:

AC1. $(M, \vec{u}) \models (\vec{X} = \vec{x}) \wedge \varphi$.

AC2. There exist a partition (\vec{Z}, \vec{W}) of \mathcal{V} with $\vec{X} \subseteq \vec{Z}$ and some setting (\vec{x}', \vec{w}) of the variables in (\vec{X}, \vec{W}) such that if $(M, \vec{u}) \models Z = z^*$ for $Z \in \vec{Z}$, then

(a) $(M, \vec{u}) \models [\vec{X} \leftarrow \vec{x}', \vec{W} \leftarrow \vec{w}] \neg \varphi$.

(b) $(M, \vec{u}) \models [\vec{X} \leftarrow \vec{x}, \vec{W}' \leftarrow \vec{w}, \vec{Z}' \leftarrow \vec{z}^*]\varphi$ for all subsets \vec{Z}' of $\vec{Z} \setminus \vec{X}$ and all subsets \vec{W}' of \vec{W} , where we abuse notation and write $\vec{W}' \leftarrow \vec{w}$ to denote the assignment where the variables in \vec{W}' get the same values as they would in the assignment $\vec{W} \leftarrow \vec{w}$, and similarly for $\vec{Z}' \leftarrow \vec{z}^*$. That is, setting any subset \vec{W}' of \vec{W} to the values in \vec{w} should

have no effect on φ as long as \vec{X} has the value \vec{x} , even if all the variables in an arbitrary subset of \vec{Z} are set to their values in the context \vec{u} . The tuple $(\vec{W}, \vec{w}, \vec{x}')$ is said to be a witness to the fact that $\vec{X} = \vec{x}$ is a cause of φ .

AC3. $(\vec{X} = \vec{x})$ is minimal; no subset of \vec{X} satisfies AC2.

If \vec{X} is a singleton, then $X = x$ is said to be a *singleton cause* of φ in (M, \vec{u}) .

AC1 just says that A cannot be a cause of B unless both A and B are true. The core of this definition lies in AC2. The variables in \vec{Z} can be thought of as the variables on a causal path between the variables in \vec{X} and the variables in φ . While this is not always the case (see (Halpern, 2016, Section 2.9) for a counterexample), it is typically the case, and is a helpful intuition.³

AC2(a) is reminiscent of the traditional counterfactual criterion, according to which $X = x$ is a cause of φ if changing the value of X results in φ being false. However, AC2(a) is more permissive than the traditional criterion; it allows the dependence of φ on X to be tested under special *structural contingencies*, in which the variables \vec{W} are held constant at some setting \vec{w} . AC2(b) is an attempt to counteract the “permissiveness” of AC2(a) with regard to structural contingencies. Essentially, it ensures that X alone suffices to bring about the change from φ to $\neg\varphi$; setting \vec{W} to \vec{w} merely eliminates spurious side effects that tend to mask the action of X .

To understand the role of AC2(b), consider the rock-throwing example again. Let M be the model in Figure 1, and let \vec{u} be the context where both Suzy and Billy throw. It is easy to see that both Suzy and Billy are causes of the bottle shattering in (M, \vec{u}) : Let $\vec{Z} = \{ST, BS\}$, and consider the structural contingency where Billy doesn’t throw ($BT = 0$). Clearly $(M, U) \models [ST \leftarrow 0, BT \leftarrow 0](BS = 0)$ and $(M, u) \models [ST \leftarrow 1, BT \leftarrow 0](BS = 1)$, so Suzy is a cause of the bottle shattering. A symmetric argument shows that Billy is also a cause.

But now consider the model M' described in Figure 2; again, u is the context where both Suzy and Billy throw. It is still the case that Suzy is a cause of the bottle shattering in (M', u) . We can take $\vec{W} = \{BT\}$ and again consider the contingency where Billy doesn’t throw. However, Billy is *not* a cause of the bottle shattering in (M', u) . For suppose that we now take $\vec{W} = \{ST\}$ and consider the contingency where Suzy doesn’t throw. AC2(a) holds, since if Billy doesn’t throw (under this contingency), then the bottle doesn’t shatter. However, AC2(b) does not hold. Since $BH \in \vec{Z}$, if we set BH to 0 (its original value), then AC2(b) would require that $(M', u) \models [BT \leftarrow 1, ST \leftarrow 0, BH \leftarrow 0](BS = 1)$, but this is not the case. Similar arguments show that no other choice of (\vec{Z}, \vec{W}) makes Billy’s throw a cause of the bottle shattering in (M', u) .

The original HP definition differs from the updated definition in only one respect. Rather than requiring that $(M, \vec{u}) \models [\vec{X} \leftarrow \vec{x}, \vec{W}' \leftarrow \vec{w}, \vec{Z}' \leftarrow \vec{z}^*]\varphi$ for all subsets \vec{W}' of \vec{W} , it was required to hold only for \vec{W} . That is, the following condition was used instead of AC2(b).

AC2(b') $(M, \vec{u}) \models [\vec{X} \leftarrow \vec{x}, \vec{W} \leftarrow \vec{w}, \vec{Z}' \leftarrow \vec{z}^*]\varphi$ for all subsets \vec{Z}' of \vec{Z} .

The requirement for AC2(b) to hold for all subsets of W in the updated definition prevents situations where W “conceals other causes for φ ”. The role of this requirement is perhaps best understood by considering the following example, due to Hopkins and Pearl (2003) (the description is taken from Halpern & Pearl, 2005a): Suppose that a prisoner dies either if A loads B ’s gun and B shoots, or if C loads and shoots his gun. Taking D to represent the prisoner’s death and making the

3. Interestingly, In the original HP definition, we can in fact always take \vec{Z} to consist of variables that lie on a causal path from a variable in \vec{X} ; see (Halpern, 2016, Section 2.9).

obvious assumptions about the meaning of the variables, we have that $D = (A \wedge B) \vee C$. Suppose that in the actual context u , A loads B 's gun, B does not shoot, but C does load and shoot his gun, so that the prisoner dies. That is, $A = 1$, $B = 0$, and $C = 1$. $C = 1$ is a cause of $D = 1$, since if we set C to 0, then $D = 0$. We would not want to say that $A = 1$ is a cause of $D = 1$, given that B did not shoot (i.e., given that $B = 0$). However, with AC2(b'), $A = 1$ is a cause of $D = 1$. For we can take $\vec{W} = \{B, C\}$ and consider the contingency where $B = 1$ and $C = 0$. It is easy to check that AC2(a) and AC2(b') hold for this contingency, so under the original HP definition, $A = 1$ is a cause of $D = 1$. However, AC2(b) fails in this case, since $(M, u) \models [A \leftarrow 1, C \leftarrow 0](D = 0)$. The key point is that AC2(b) says that for $A = 1$ to be a cause of $D = 1$, it must be the case that $D = 0$ if only some of the values in \vec{W} are set to \vec{w} . That means that the other variables get the same value as they do in the actual context; in this case, by setting only A to 1 and leaving B unset, B takes on its original value of 0, in which case $D = 0$. AC2(b') does not consider this case.

Using AC2(b) rather than AC2(b') has been shown to have a significant benefit (and to lead to more intuitive results) when causality is applied to program verification, with the goal of understanding what in the code is the cause of a program not satisfying its specification (Beer, Ben-David, Chockler, Orni, & Trefler, 2012).

3. Relevant Complexity Classes

In this section, we briefly recall the definitions of the complexity classes that we need for our results, and define the complexity class D_2^k .

Recall that the *polynomial hierarchy* is a hierarchy of complexity classes that generalize the classes NP and co-NP . Let $\Sigma_1^P = NP$ and $\Pi_1^P = \text{co-NP}$. For $i > 1$, define $\Sigma_i^P = NP^{\Sigma_{i-1}^P}$ and $\Pi_i^P = (\text{co-NP})^{\Sigma_{i-1}^P}$, where, in general, X^Y denotes the class of problems solvable by a Turing machine in class A augmented with an oracle for a problem complete for class B . (See Sipser, 2012; Stockmeyer, 1977 for more details and intuition.)

We now define the classes D_k^P as follows.

Definition 3.1 For $k = 1, 2, \dots$,

$$D_k^P = \{L : \exists L_1, L_2 : L_1 \in \Sigma_k^P, L_2 \in \Pi_k^P, L = L_1 \cap L_2\}.$$

For $k = 1$, the class D_1^P is the complexity class D^P defined by Papadimitriou and Yannakakis (1984). It contains *exact* problems such as the language of pairs $\langle G, k \rangle$, where G is a graph that has a maximal clique of size exactly k .⁴ Note that D_1^P is *not* equivalent to $NP \cap \text{co-NP}$ (and, more generally, D_k^P is not equivalent to $\Sigma_k^P \cap \Pi_k^P$). For example, although the exact clique problem is in D_1^P , as we just noted, it is not believed to be in either NP or co-NP , so is not in $NP \cap \text{co-NP}$. As usual, we say that a language L is D_k^P complete if it is in D_k^P and is the ‘‘hardest’’ language in D_k^P , in the sense that there is a polynomial-time reduction from any language $L' \in D_k^P$ to L .

Recall that a *quantified Boolean formula* (QBF) is a generalization of a propositional formula, where some propositional variables are quantified. Thus, for example, $\exists x \forall y (x \vee y)$ is a QBF. A *closed* QBF (CQBF) is one where there are no free propositional variables. A CQBF is either true or false, independent of the truth assignment. The ‘‘canonical’’ languages complete for Σ_2^k and Π_2^k

4. The language of graphs that have a clique of size at least k is in NP , since we can guess a set of vertices of size k and check that it is a clique in polynomial time; in fact, it is a well-known NP -complete problem (Garey & Johnson, 1979). The language of graphs that have a clique of size at most k is in co-NP , since its complement is in NP .

consist of the CQBFs with k alternations of quantifiers starting with \exists (resp., \forall) that are true. In particular, let

$$\begin{aligned}\Sigma_2^P(\text{SAT}) &= \{\exists \vec{X} \forall \vec{Y} \varphi \mid \exists \vec{X} \forall \vec{Y} \varphi \text{ is a CQBF, } \exists \vec{X} \forall \vec{Y} \varphi = \mathbf{true}\} \\ \Pi_2^P(\text{SAT}) &= \{\forall \vec{X} \exists \vec{Y} \varphi \mid \forall \vec{X} \exists \vec{Y} \varphi \text{ is a CQBF, } \forall \vec{X} \exists \vec{Y} \varphi = \mathbf{true}\}.\end{aligned}$$

$\Sigma_2^P(\text{SAT})$ is complete for Σ_2^P and $\Pi_2^P(\text{SAT})$ is complete for Π_2^P (Wrathall, 1976).

The following lemma provides a useful condition sufficient for a language to be D_k^P -complete.

Lemma 3.2 *If L_1 is Σ_k^P -complete and L_2 is Π_k^P -complete, then $L_3 = L_1 \times L_2$ is D_k^P -complete.*

Proof: Without loss of generality, let L_1 and L_2 be languages over some alphabet Φ . The fact that L_3 is in D_k^P is immediate from the definition of D_k^P and the observation that $L_3 = (L_1 \times \Phi^*) \cap (\Phi^* \times L_2)$. For hardness, let L_3' be a language in D_k^P . Then there exist L_1' and L_2' such that $L_1' \in \Sigma_k^P$, $L_2' \in \Pi_k^P$, and $L_3' = L_1' \cap L_2'$. Let f be a polynomial-time (many-to-one) reduction from L_1' to L_1 , and let g be a polynomial-time (many to one) reduction from L_2' to L_2 (the existence of such reductions f and g follows from the fact that L_1 and L_2 are Σ_k^P -complete and Π_k^P -complete, respectively). Then the reduction $\langle f, g \rangle : L_3' \rightarrow L_1 \times L_2$ defined by $\langle f, g \rangle \langle x, y \rangle = \langle f(x), g(y) \rangle$ is a polynomial-time reduction from L_3' to $L_3 = L_1 \times L_2$, as required. \square

Determining whether $\vec{X} = \vec{x}$ is a cause of φ in (M, u) is a decision problem: we define a language and try to determine whether a particular tuple is in that language. (See Section 4 for the formal definition.) Determining degree of responsibility and blame is a different type of problem, since we are determining which number represents the degree of responsibility (resp., blame). Formally, these are *function problems*. For ease of exposition, we restrict attention to functions from strings over some fixed language Σ to strings over Σ (i.e., we are considering functions from Σ^* to Σ^*). For a complexity class A in the polynomial hierarchy, $\text{FP}^{A[\log n]}$ consists of all functions that can be computed by a polynomial-time Turing machine with an A -oracle which on input x asks a total of $O(\log |x|)$ queries (Papadimitriou, 1984). A function $f(x)$ is $\text{FP}^{A[\log n]}$ -hard iff for every function $g(x)$ in $\text{FP}^{A[\log n]}$ there exist polynomially computable functions $R, S : \Sigma^* \rightarrow \Sigma^*$ such that $g(x) = S(f(R(x)))$. A function $f(x)$ is complete in $\text{FP}^{A[\log n]}$ iff it is in $\text{FP}^{A[\log n]}$ and is $\text{FP}^{A[\log n]}$ -hard. Note that since a Σ_k^P -oracle gives positive and negative answers in $O(1)$ time, the class $\text{FP}^{\Pi_k^P[\log n]}$ coincides with $\text{FP}^{\Sigma_k^P[\log n]}$ for all $k \geq 1$ (see also Krentel, 1988).

We remark that, rather than computing the degree of responsibility, which is a function problem, we could have computed the corresponding decision problem “is the degree of responsibility of $\vec{X} = \vec{x}$ for φ in (M, \vec{u}) at least $1/k$ ”. Since $\vec{X} = \vec{x}$ is a cause of φ in (M, \vec{u}) iff its degree of responsibility is greater than 0, this problem must be at least as hard as that of determining causality. It is also no harder, as we could limit our consideration of potential sets \vec{W} in Def. 2.1 to sets of size no larger than $k - 1$ (since the degree of responsibility is defined as $1/(|\vec{W}| + 1)$), making this a variant of the general causality problem, with the reasoning for membership in the relevant complexity classes being the same as that for causality. Hence, it is Σ_2^P -complete for singleton causes and Δ_2^P -complete in general. That said, we believe that the function problem is the more relevant problem for most applications of degree of responsibility. (Similar remarks hold for degree of blame.)

Finally, for a complexity class A in polynomial hierarchy, FP_{\parallel}^A is the class of functions that can be computed by a polynomial-time Turing machine with parallel (i.e., non-adaptive) queries to an A -oracle. (For background on these complexity classes, see Jenner & Toran, 1995; Johnson, 1990.)

4. Complexity for the Updated HP Definition

In this section, we prove our results on the complexity of deciding causality. We start by defining the problem formally. In the definitions, M stands for a causal model, \vec{u} is a context, \vec{X} is a subset of variables of M , and \vec{x} is the set of values of \vec{X} in (M, \vec{u}) :

$$L_{\text{cause}} = \{ \langle M, \vec{u}, \varphi, \vec{X}, \vec{x} \rangle : (\vec{X} = \vec{x}) \text{ is a cause of } \varphi \text{ in } (M, \vec{u}) \}.$$

One of our goals is to understand the cause of the complexity of computing causality. Towards this end, it is useful to define two related languages:

$$\begin{aligned} L_{\text{AC2}} &= \{ \langle M, \vec{u}, \varphi, \vec{X}, \vec{x} \rangle : (\vec{X} = \vec{x}) \text{ satisfies conditions AC1 and AC2 of Def. 2.1 for } \varphi \text{ in } (M, \vec{u}) \}, \\ L_{\text{AC3}} &= \{ \langle M, \vec{u}, \varphi, \vec{X}, \vec{x} \rangle : (\vec{X} = \vec{x}) \text{ satisfies conditions AC1 and AC3 of Def. 2.1 for } \varphi \text{ in } (M, \vec{u}) \}. \end{aligned}$$

It is easy to see that $L_{\text{cause}} = L_{\text{AC2}} \cap L_{\text{AC3}}$.

Let L_{cause}^1 be the subset of L_{cause} where \vec{X} and \vec{x} are singletons; this is the singleton causality problem. We can similarly define L_{AC2}^1 and L_{AC3}^1 . Again, we have $L_{\text{cause}}^1 = L_{\text{AC2}}^1 \cap L_{\text{AC3}}^1$, but, in fact, we have $L_{\text{cause}}^1 = L_{\text{AC2}}^1$, since $L_{\text{AC2}}^1 \subseteq L_{\text{AC3}}^1$; for singleton causality, the minimality condition AC3 trivially holds.

We denote by L_{cause}^B the language of causality for binary causal models (i.e., where the models M in the tuple are binary models), and by L_{AC2}^B and L_{AC3}^B the languages L_{AC2} and L_{AC3} restricted to binary causal models. Again we have that $L_{\text{cause}}^B = L_{\text{AC2}}^B \cap L_{\text{AC3}}^B$. And again, we can define $L_{\text{cause}}^{B,1}$, $L_{\text{AC2}}^{B,1}$, and $L_{\text{AC3}}^{B,1}$, and we have $L_{\text{cause}}^{B,1} = L_{\text{AC2}}^{B,1}$.

We start by considering singleton causality. As we observed, Eiter and Lukasiewicz (2002) and Hopkins (2001) showed that, with the original HP definition, singleton causality and causality coincide. However, for the updated definition, Halpern (2008) showed that it is in fact possible to have minimal causes that are not singletons. Thus, we consider singleton causality and general causality separately. We can clarify where the complexity lies by considering L_{AC2} (and its sublanguages) and L_{AC3} (and its sublanguages) separately.

Theorem 4.1 *The languages L_{AC2} , L_{AC2}^1 , $L_{\text{AC2}}^{B,1}$, and L_{AC2}^1 are Σ_2^P -complete.*

Proof outline: To show that all these languages are in Σ_2^P , given a tuple $\langle M, \vec{u}, \varphi, \vec{X}, \vec{x} \rangle$, checking that AC1 holds, that is, checking that $(M, \vec{u}) \models \vec{X} = \vec{x} \wedge \varphi$, can be done in time polynomial in the size of M , $|\vec{X}|$, and $|\varphi|$ (the length of φ as a string of symbols). For AC2, we need only guess the set \vec{W} and the assignment \vec{w} . The check that assigning \vec{w} to \vec{W} and \vec{x} to \vec{X} indeed falsifies φ is polynomial; we use an NP oracle to check that for all subsets of \vec{W} and all subsets of \vec{Z} , condition AC2(b) holds. (The oracle is used to check if these conditions do *not* hold, which can easily be checked in NP.) The argument is similar in spirit to Eiter and Lukasiewicz's argument that causality is in Σ_2^P for general models with the original HP definition.

For hardness, it suffices to show that $L_{\text{AC2}}^{B,1}$ is Σ_2^P -hard. We do this by reducing Σ_2^P (SAT) to $L_{\text{AC2}}^{B,1}$. Given a CQBF formula $\exists \vec{X} \forall \vec{Y} \varphi$, we show that we can efficiently construct a causal formula ψ , model M , and context u such that $\exists \vec{X} \forall \vec{Y} \varphi = \mathbf{true}$ iff $(M, u, \psi, A, 0) \in L_{\text{AC2}}^{B,1}$. We leave details to Appendix A. \square

Since, as we have observed, AC3 holds trivially in the case of singleton causality, it follows that singleton causality is Σ_2^P -complete.

Corollary 4.2 L_{cause}^1 and $L_{\text{cause}}^{B,1}$ are Σ_2 -complete.

We now show that things are harder if we do not restrict causes to singletons (unless the polynomial hierarchy collapses). As a first step, we consider the complexity of L_{AC3} and L_{AC3}^B .

Theorem 4.3 L_{AC3} and L_{AC3}^B are Π_2^P -complete.

Proof outline: The fact that L_{AC3} and L_{AC3}^B are in Π_2^P is straightforward. Again, given a tuple $\langle M, \vec{u}, \varphi, \vec{X}, \vec{x} \rangle$, we can check that AC1 holds in polynomial time. For AC3, we need to check that for all strict subsets \vec{X}' of \vec{X} , AC2 fails. Since checking AC2 is in Σ_2^P , checking that it fails is in Π_2^P . Checking that it fails for all strict subsets \vec{X}' keeps it in Π_2^P (since it just adds one more universal quantifier).

To prove that these languages are Π_2^P -hard, we show that we can reduce $\Pi_2^P(\text{SAT})$ to L_{AC3}^B . The proof is similar in spirit to the proof of Theorem 4.1; we leave details to the appendix. \square

We are now ready to prove our main result.

Theorem 4.4 L_{cause} and L_{cause}^B are D_2^P -complete.

Proof outline: Membership of L_{cause} (and hence also L_{cause}^B) in D_2^P follows from the fact that $L_{\text{cause}} = L_{\text{AC2}} \cap L_{\text{AC3}}$, $L_{\text{AC2}} \in \Sigma_2^P$, and $L_{\text{AC3}} \in \Pi_2^P$. For hardness, we first consider the language $\Sigma_2^P(\text{SAT}) \times \Pi_2^P(\text{SAT})$, which, by Lemma 3.2 is D_k^P -complete. We then reduce this language to L_{cause}^B using the constructions in Theorems 4.1 and 4.3. We leave details to Appendix B. \square

The fact that there may be more than one conjunct in a cause using the updated HP definition means that checking AC3 becomes nontrivial, and causes the increase in complexity for Σ_2^P to D_2^P . But why is there no dropoff with the updated HP definition when we restrict to binary models, although there is a dropoff from Σ_2^P to NP for the original HP definition? To prove their NP -completeness result, Eiter and Lukasiewicz (2002) showed that for binary models, with the original HP definition, the set \vec{Z} and its subsets can be omitted from the definition of cause. That is, we can replace AC2(b') by

$$\text{AC2(b'')} \quad (M, \vec{u}) \models [\vec{X} \leftarrow \vec{x}, \vec{W} \leftarrow \vec{w}] \varphi$$

to get an equivalent definition. The example that a cause may require more than one conjunct given by Halpern (2008) shows that removing \vec{Z} and its subsets from AC2(b) does not result in an equivalent definition in binary models. But even if it did, the fact that we need to quantify over all subset \vec{W}' of \vec{W} in AC2(b) would be enough to ensure that there is no dropoff in complexity in binary models.

5. Responsibility and Blame

In this section, we review the definitions of responsibility and blame and characterize their complexity. See Chockler and Halpern (2004) for more intuition and details.

5.1 Responsibility

The definition of responsibility given by Chockler and Halpern (2004) was given based on the original HP definition of causality, and thus assumed that causes were always single conjuncts. It is straightforward to extend it to allow causes to have arbitrarily many conjuncts.

Definition 5.1 *The degree of responsibility of $\vec{X} = \vec{x}$ for φ in (M, \vec{u}) , denoted $dr((M, \vec{u}), (\vec{X} = \vec{x}), \varphi)$, is 0 if $\vec{X} = \vec{x}$ is not a cause of φ in (M, \vec{u}) ; it is $1/(k+1)$ if $\vec{X} = \vec{x}$ is a cause of φ in (M, \vec{u}) and there exists a partition (\vec{Z}, \vec{W}) and setting (\vec{x}', \vec{w}) for which AC2 holds such that (a) k variables in \vec{W} have different values in \vec{w} than they do in the context \vec{u} and (b) there is no partition (\vec{Z}', \vec{W}') and setting (\vec{x}'', \vec{w}') satisfying AC2 such that only $k' < k$ variables have different values in \vec{w}' than they do the context \vec{u} .⁵*

Intuitively, $dr((M, \vec{u}), (\vec{X} = \vec{x}), \varphi)$ measures the minimal number of variables whose values have to be changed in order to make φ counterfactually depend on \vec{X} . If there is no partition of \mathcal{V} to (\vec{Z}, \vec{W}) that satisfies AC2, or $(\vec{X} = \vec{x})$ does not satisfy AC3 for φ in (M, \vec{u}) , then the minimal number of changes in Definition 5.1 is taken to have cardinality ∞ , and thus the degree of responsibility of $(\vec{X} = \vec{x})$ is 0 (and hence it is not a cause).

In the original HP model, it was shown that computing responsibility is $\text{FP}^{\text{NP}[\log n]}$ -complete in binary causal models (Chockler et al., 2008) and $\text{FP}^{\Sigma_2^P[\log n]}$ -complete in general causal models (Chockler & Halpern, 2004). First, we prove that the $\text{FP}^{\Sigma_2^P[\log n]}$ -completeness result holds for singleton causes.

Theorem 5.2 *Computing the degree of responsibility is $\text{FP}^{\Sigma_2^P[\log n]}$ -complete for singleton causes in binary and general causal models.*

Proof outline: The proof is quite similar to the proof in (Chockler & Halpern, 2004). We prove membership by describing an algorithm in $\text{FP}^{\Sigma_2^P[\log n]}$ for computing the degree of responsibility. Roughly speaking, the algorithm queries an oracle for the language $\mathcal{R} = \{((M, \vec{u}), (X = x), \varphi, i) \text{ such that } ((M, \vec{u}), (X = x), \varphi) \in L_{\text{cause}}^1 \text{ and the degree of responsibility of } (X = x) \text{ for } \varphi \text{ is at least } i\}$. It follows easily from Corollary 4.2 that \mathcal{R} is Σ_2^P -complete. The algorithm for computing the degree of responsibility performs a binary search on the values of i such that $dr((M, \vec{u}), (X = x), \varphi) \geq i$, using \mathcal{R} as an oracle, dividing the range of possible values for the degree of responsibility by 2 according to the answer given by \mathcal{R} . The number of possible candidates for the degree of responsibility is bounded by n , the size of the input, and thus the number of queries is at most $\lceil \log n \rceil$.

For hardness in binary causal models (which implies hardness in general causal models), we can reduce the $\text{FP}^{\Sigma_2^P[\log n]}$ -complete problem MINQSAT_2 (Chockler & Halpern, 2004) to the degree of responsibility, where $\text{MINQSAT}_2(\exists \vec{X} \forall \vec{Y} \psi)$ is the minimum number of 1's in the satisfying assignment to \vec{X} for $\exists \vec{X} \forall \vec{Y} \psi$ if such an assignment exists, and $|\vec{X}| + 1$ otherwise. The argument is almost the same as that given by Chockler and Halpern in the case of the original HP definition; we refer the reader to (Chockler & Halpern, 2004) for details. \square

5. This definition is a slightly different from the definition of responsibility in (Halpern, 2016); here we define the degree of responsibility for non-singletons $\vec{X} = \vec{x}$; even for singletons, the definitions are slightly different. That said, the two variants have the same complexity, as can be shown using a slight variant of the argument given in the proof of Theorems 5.3.

A perhaps surprising observation is that the complexity of computing the degree of responsibility for general causes is no higher than that for singletons, as the following theorem shows.⁶

Theorem 5.3 *Computing the degree of responsibility is $FP^{\Sigma_2^P[\log n]}$ -complete in binary and general causal models.*

Proof: Hardness follows immediately from hardness of a special case, proven in Theorem 5.2. It remains to show membership in $FP^{\Sigma_2^P[\log n]}$. Recall that the language L_{AC2} , defined in Section 4, is Σ_2^P -complete (see Theorem 4.1). It is easy to show that the following language, which is a modification of L_{AC2} , is in Σ_2^P as well:

$$L_{AC2}^i = \{ \langle (M, \vec{u}), (\vec{X} = \vec{x}), \varphi, i \rangle : \langle (M, \vec{u}), (\vec{X} = \vec{x}), \varphi \rangle \text{ satisfies AC1 and AC2} \\ \text{using a set } \vec{W} \text{ such that } |\vec{W}| \geq i \}.$$

By Theorem 4.3, the language L_{AC3} (also defined in Section 4) is Π_2^P -complete, so its complement $\overline{L_{AC3}}$ is Σ_2^P -complete. Thus, the language $L_{AC2}^i \times \overline{L_{AC3}}$ is in Σ_2^P .

Let \mathcal{R}' be an oracle for the language $L_{AC2}^i \times \overline{L_{AC3}}$. The algorithm for computing the degree of responsibility first queries the oracle \mathcal{R}' for whether $\langle (M, \vec{u}), (\vec{X} = \vec{x}), \varphi \rangle$ in $\overline{L_{AC3}}$ (formally, by asking whether $(a, \langle (M, \vec{u}), (\vec{X} = \vec{x}), \varphi \rangle) \in L_{AC2}^i \times \overline{L_{AC3}}$, where a is a fixed tuple that is trivially in L_{AC2}^i —it is straightforward to construct such a tuple). If the answer is positive, then $(\vec{X} = \vec{x})$ is not a cause of φ in (M, \vec{u}) , as it does not satisfy the minimality condition AC3. Otherwise (i.e., if the answer is negative), the algorithm performs a binary search on the values of i such that $dr((M, \vec{u}), (X = x), \varphi) \geq i$, using the oracle \mathcal{R}' . The number of possible candidates for the degree of responsibility is bounded by n , the size of the input, and thus the number of queries is at most $(\lceil \log n \rceil + 1)$ (the additional 1 comes from the query to $\overline{L_{AC3}}$). \square

5.2 Blame

The notion of blame addresses the situation where there is uncertainty about the true situation or “how the world works”. Blame, introduced in (Chockler & Halpern, 2004), considers the “true situation” to be determined by the context, and “how the world works” to be determined by the structural equations. An agent’s uncertainty is modeled by a pair (\mathcal{K}, Pr) , where \mathcal{K} is a set of pairs of the form (M, \vec{u}) , where M is a causal model and \vec{u} is a context, and Pr is a probability distribution over \mathcal{K} . A pair (M, \vec{u}) is called a *situation*. We think of \mathcal{K} as describing the situations that the agent considers possible before \vec{X} is set to \vec{x} . The degree of blame that setting \vec{X} to \vec{x} has for φ is then the expected degree of responsibility of $\vec{X} = \vec{x}$ for φ in $(M_{\vec{X} \leftarrow \vec{x}}, \vec{u})$, taken over the situations $(M, \vec{u}) \in \mathcal{K}$. Note that the situations $(M_{\vec{X} \leftarrow \vec{x}}, \vec{u})$ for $(M, \vec{u}) \in \mathcal{K}$ are those that the agent considers possible after \vec{X} is set to \vec{x} .

Definition 5.4 *The degree of blame of setting \vec{X} to \vec{x} for φ relative to epistemic state (\mathcal{K}, Pr) , denoted $\text{db}(\mathcal{K}, \text{Pr}, \vec{X} \leftarrow \vec{x}, \varphi)$, is*

$$\sum_{(M, \vec{u}) \in \mathcal{K}} dr((M_{\vec{X} \leftarrow \vec{x}}, \vec{u}), \vec{X} = \vec{x}, \varphi) \text{Pr}((M, \vec{u})).$$

6. Note that the conference version of this paper (Aleksandrowicz, Chockler, Halpern, & Ivrii, 2014) claimed that the problem was $FP^{D_2[\log n]}$ -complete. As we show here, this claim was incorrect.

For the original HP definition of cause, Chockler and Halpern (2004) show that the complexity of computing the degree of blame is best described using the complexity classes $FP_{\parallel}^{\Sigma_2^P}$ and FP_{\parallel}^{NP} , which consist of all functions that can be computed in polynomial time with parallel (i.e., non-adaptive) queries to a Σ_2^P (respectively, NP) oracle. (For background on these complexity classes, see Jenner & Toran, 1995; Johnson, 1990.) The degree of blame for the original definition of cause is proven to be $FP_{\parallel}^{\Sigma_2^P}$ -complete for general causal models and FP_{\parallel}^{NP} -complete for binary causal models. Since the complexity of computing blame is directly derived from the complexity of computing responsibility, we have the following theorem.

Theorem 5.5 *The problem of computing blame in recursive causal models is $FP_{\parallel}^{\Sigma_2^P}$ -complete for singleton and general causes in binary and general causal models.*

Proof outline: The fact that computing blame for singleton causes (and hence also for general causes) is $FP_{\parallel}^{\Sigma_2^P}$ -hard follows directly from the proof of the $FP_{\parallel}^{\Sigma_2^P}$ -completeness of blame for singleton causes, which is the same as the proof for the previous definition of causality (Chockler & Halpern, 2004). The fact that computing blame for general causes (and hence for singleton causes) is in $FP_{\parallel}^{\Sigma_2^P}$ is shown by a parallel computation of the degree of responsibility for all $(M, \vec{u}) \in \mathcal{K}$ and an application of Theorem 5.3. The degree of blame, which is the expected degree of responsibility taken over the situations $(M, \vec{u}) \in \mathcal{K}$ according to the probability distribution Pr , can then be computed in polynomial time, given the degree of responsibility for each situation. \square

6. Conclusion

In this paper, we have examined the complexity of computing whether $\vec{X} = \vec{x}$ is a cause of φ and the degree of responsibility and blame of $\vec{X} = \vec{x}$ for φ according to the updated definition of causality introduced by Halpern and Pearl (2005b). According to Halpern and Pearl's (2005a) original definition of causality, the complexity of computing whether $\vec{X} = \vec{x}$ is a cause of φ was shown by Eiter and Lukasiewicz (2002) to be NP -complete for binary models, and Σ_2^P -complete for general models. For the updated model, we show that the complexity rises to D_2^P -complete, both for binary and general models. On the other hand, the complexity of computing the degree of responsibility and blame does not change as we move from the original definition to the updated definition.

These results suggest that computing causality will be difficult for practical problems. The obvious question is whether there are special cases of the problem that are of practical interest that are more tractable. An examples of a tractable class of interest for the original definition was given by Chockler, Halpern, and Kupferman (2008). For problems in this class, the original and updated definitions agree, so they are tractable for the updated definition as well. Another example was given by Chockler et al. (2015), who used causal models for the analysis of complex legal cases. These models were shown to be highly modular, with each variable affecting only a small part of the model, thus allowing for a modular computation of the degrees of responsibility and blame. It would be of interest to find other such classes.

More recently, Halpern (2015) proposed yet another modification of the HP definition, and showed that checking whether $\vec{X} = \vec{x}$ is a cause of φ is D^P -complete. Alechina, Halpern, and Logan (2016) showed that the problem of determining the degree of responsibility and the degree of blame of $X = x$ for φ is $FP_{\parallel}^{\Sigma_2^P[\log n]}$ -complete, just as for the original and updated definition. Given

the existence of three variants of the definition, clearly more work is needed to understand which is most appropriate and when; see (Halpern, 2016) for more on this issue.

Acknowledgements

A preliminary version of this paper appears in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI-14)*, 2014 (Aleksandrowicz et al., 2014). Joseph Halpern was supported in part by NSF grants IIS-0911036 and CCF-1214844, AFOSR grant FA9550-08-1-0438, ARO grant W911NF-14-1-0017, and by the DoD Multidisciplinary University Research Initiative (MURI) program administered by AFOSR under grant FA9550-12-1-0040.

Appendix A. Proof of Theorem 4.1

As we observed in the main part of the paper, membership is straightforward, so we focus here on hardness. For hardness, we describe a reduction from the language $\Sigma_2^P(\text{SAT})$ to $L_{\text{AC2}}^{B,1}$. In the process, we work with both propositional formulas with propositional variables, and causal formulas that use formulas like $X = 1$ and $X = 0$. We can think of X as a propositional variable here, where $X = 1$ denotes that X is true, and $X = 0$ denotes that x is false. If φ is a propositional formula, let $\tilde{\varphi}$ be the causal formula that results by replacing each occurrence of a propositional variable X by $X = 1$.

Given a CQBF $\exists \vec{X} \forall \vec{Y} \varphi$, consider the tuple $(M, u, \psi, A, 0)$ where $M = (\mathcal{U}, \mathcal{V}, \mathcal{R})$ is a binary causal model and

- $\mathcal{U} = \{U\}$;
- $\mathcal{V} = \{X^0 \mid X \in \vec{X}\} \cup \{X^1 \mid X \in \vec{X}\} \cup \{Y \mid Y \in \vec{Y}\} \cup \{A\}$, where A is a fresh variable that does not appear in \vec{X} or \vec{Y} ;
- for all variables $V \in \vec{\mathcal{V}}$, the structural equation is $V = U$ (i.e. all the variables in \mathcal{V} are set to the value of U);
- $u = 0$;
- $\psi = \psi_1 \vee (\psi_2 \wedge \psi_3)$ where ψ_1, ψ_2, ψ_3 are the following causal formulas:
 - $\psi_1 = \neg (\bigwedge_{X \in \vec{X}} (X^0 \neq X^1))$; ⁷
 - $\psi_2 = \neg (A = 1 \wedge \vec{Y} = \vec{1})$;
 - $\psi_3 = (A = 1) \vee \tilde{\varphi}[\vec{X}/\vec{X}^1]$, where $\tilde{\varphi}[\vec{X}/\vec{X}^1]$ is the result of replacing each occurrence of a variable $X \in \vec{X}$ by X^1 .

We prove that $\exists \vec{X} \forall \vec{Y} \varphi = \mathbf{true}$ iff $A = 0$ is a cause of ψ in (M, u) (which is the case iff $(M, u, \psi, A, 0) \in L_{\text{AC2}}^{B,1}$, since AC3 holds trivially for singleton causes).

First suppose that $\exists \vec{X} \forall \vec{Y} \varphi = \mathbf{true}$. To show that $A = 0$ is a cause of ψ in (M, u) , we prove that AC1 and AC2 hold. Clearly AC1 holds: the fact that $(M, u) \models A = 0$ is immediate, given the equation for A , and $(M, u) \models \psi$ since $(M, u) \models \psi_1$, again by the definition of \mathcal{F} . For AC2, let

⁷ As usual, we take $X^0 \neq X^1$ to be an abbreviation for the causal formula $(X^0 = 1 \wedge X^1 = 0) \vee (X^0 = 0 \wedge X^1 = 1)$.

$\vec{W} = \mathcal{V} - \{A\}$ and define \vec{w} as follows. Let τ be an assignment to the variables in \vec{X} for which $\forall \vec{Y} \varphi = \mathbf{true}$. Using $\vec{w}(X)$ to denote the value of X according to \vec{w} , we define

- $\vec{w}(X^{\tau(X)}) = 1$ for all $X \in \vec{X}$;
- $\vec{w}(X^{1-\tau(X)}) = 0$ for all $X \in \vec{X}$; and
- $\vec{w}(Y) = 1$ for all $Y \in \vec{Y}$.

For AC2(a), note that $(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg \psi_1$ (since \vec{w} assigns different values to X^0 and X^1 for all $X \in \vec{X}$) and, since $\vec{w}(Y) = 1$ for all $Y \in \vec{Y}$, we have that $(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg \psi_2$, so $(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg \psi$. Thus, AC2(a) holds.

It now remains to show that AC2(b) holds. Fix $\vec{W}' \subseteq \vec{W}$. We must show that $(M, u) \models [A \leftarrow 0, \vec{W}' \leftarrow \vec{w}] \psi$. (The condition “for all $\vec{Z}' \subseteq \vec{Z} - \{A\}$ ” is vacuous in this case, since $\vec{Z} = \{A\}$.) Since the definition of M guarantees that $(M, u) \models [A \leftarrow 0, \vec{W}' \leftarrow \vec{w}] \psi$ iff $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \psi$, we focus on the latter condition from here on in.

If $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \psi_1$, we are done. So suppose that $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \neg \psi_1$; that is,

$$(M, u) \models [\vec{W}' \leftarrow \vec{w}] \left(\bigwedge_{X \in \vec{X}} (X^0 \neq X^1) \right). \quad (1)$$

It follows that, for each variable $X \in \vec{X}$, we have that $(M, u) \models [\vec{W}' \leftarrow \vec{w}] (X^1 = \tau(X))$. To see this, note that if $\tau(X) = 1$, then we must have $X^1 \in \vec{W}'$; otherwise, we would have $(M, u) \models [\vec{W}' \leftarrow \vec{w}] (X^1 = 0 \wedge X^0 = 0)$, contradicting (1). And if $\tau(X) = 0$, then since $\vec{w}(X^1) = 0$, we must have $(M, u) \models [\vec{W}' \leftarrow \vec{w}] (X^1 = 0)$, whether or not $X^1 \in \vec{W}'$, so $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \psi_3$. Clearly $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \psi_2$. It follows that $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \psi$, showing that AC2(b) holds.

Finally, we must show that if $A = 0$ is a cause of ψ in (M, u) then $\exists \vec{X} \forall \vec{Y} \varphi = \mathbf{true}$. So suppose that $A = 0$ is a cause of ψ in (M, u) . Then there exists a witness (\vec{W}, \vec{w}, a) . Since we are considering binary models, we must have $a = 1$, so we have

$$(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg \psi. \quad (2)$$

This implies that $(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg \psi_1$, so

$$(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \left(\bigwedge_{X \in \vec{X}} (X^0 \neq X^1) \right).$$

Define τ so that $\tau(X) = b$, where $b \in \{0, 1\}$ is the unique value for which $(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] X^b = 1$.

It also follows from (2) that

$$(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg (\psi_2 \wedge \psi_3).$$

Hence,

$$(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg \psi_2$$

or

$$(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg \psi_3.$$

Since $(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \psi_3$ (because A is assigned to 1), we have that $(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg \psi_2$. Since $\psi_2 = \neg(A = 1 \wedge \vec{Y} = \vec{1})$, we have that $(M, u) \models [A \leftarrow 1, \vec{W} \leftarrow \vec{w}] (\vec{Y} = \vec{1})$. It follows that $Y \in \vec{W}$ and $\vec{w}(Y) = 1$ for all $y \in \vec{Y}$.

Now let ν be an arbitrary assignment to \vec{X} and \vec{Y} such that $\nu|_{\vec{X}} = \tau$. It suffices to show that φ is true under assignment ν . Let $\vec{W}' = \vec{W} - \{Y \in \vec{Y} \mid \nu(Y) = 0\}$; that is, \vec{W}' contains all the variables X^b that are in \vec{W} , and all the variables $Y \in \vec{Y}$ for which $\nu(Y) = 1$. By AC2(b), it follows that $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \psi$. Since \vec{W}' contains all the variables X^b in \vec{W} , we have that $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \neg \psi_1$. Thus, we must have that $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \psi_3$. Since $(M, u) \models [\vec{W}' \leftarrow \vec{w}] (A = 0)$, it follows that $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \varphi[\vec{X}/\vec{X}^1]$.

Note that for $Y \in \vec{Y}$, $\vec{w}(Y) = 1$ iff $\nu(Y) = 1$; moreover, $\vec{w}(X^1) = 1$ iff $\tau(X) = 1$ iff $\nu(X) = 1$. Thus, the fact that $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \psi_3$ implies that φ is satisfied by ν , so we are done.

This completes the proof of the theorem. For future reference, we note that our argument proved more than was strictly necessary, since we showed that $\exists \vec{X} \forall \vec{Y} \varphi = \mathbf{true}$ iff $A = 0$ is a cause of ψ in (M, u) .

Appendix B. Proof of Theorem 4.3

Again, as we observed in the main part of the paper, membership is straightforward, so we focus on hardness. We describe a reduction from the language $\Pi_2(\text{SAT})$ to L_{AC3}^B , which suffices to prove the result. The argument is similar in spirit to that for Theorem 4.1.

Given a CQBF $\forall \vec{Y} \exists \vec{X} \varphi$, consider the tuple $(M, u, \psi, \langle A_1, A_2 \rangle, \langle 0, 0 \rangle)$ where $M = (\mathcal{U}, \mathcal{V}, \mathcal{R})$ is a binary causal model and

- $\mathcal{U} = \{U\}$;
- $\mathcal{V} = \vec{X} \cup \{Y^0 \mid Y \in \vec{Y}\} \cup \{Y^1 \mid Y \in \vec{Y}\} \cup \{A_1, A_2, S\}$, where A_1, A_2 , and S are fresh variables;
- the structural equations for A_1 and A_2 are $A_1 = S$ and $A_2 = S$, and, for all other variables $V \in \mathcal{V}$, the equation is $V = U$;
- $u = 0$;
- $\psi = \psi_1 \vee (\psi_2 \wedge \psi_3) \vee (S = 0)$ where
 - $\psi_1 = \neg(\bigwedge_{Y \in \vec{Y}} (Y^0 \neq Y^1))$;
 - $\psi_2 = \neg(\vec{A} = \vec{1} \wedge \vec{X} = \vec{1})$;
 - $\psi_3 = (A_1 = A_2) \vee \neg \varphi[\vec{Y}/\vec{Y}^1]$.

We now prove that the following are equivalent:

- (a) $\forall \vec{Y} \exists \vec{X} \varphi = \mathbf{true}$;
- (b) $(M, u, \psi, \vec{A}, \vec{0})$ is in L_{AC3}^B ;
- (c) $\vec{A} = \vec{0}$ is a cause of ψ in (M, u) .

The equivalence of (a) and (b) is all we need to prove Theorem 4.3, but the stronger result is useful for proving Theorem 4.4.

We first show that (a) implies (c). So suppose that $\forall \vec{Y} \exists \vec{X} \varphi = \mathbf{true}$. To show that $\vec{A} = \vec{0}$ is a cause of ψ in (M, u) , we must prove that AC1, AC2, and AC3 hold.

For AC1, since $(M, u) \models \vec{Y}^0 = \vec{Y}^1 = \vec{0}$, we have $(M, u) \models \psi_1$, so $(M, u) \models (\vec{A} = \vec{0}) \wedge \psi$.

For AC2, let $\vec{W} = \{S\} \cup \vec{X} \cup \vec{Y}^0 \cup \vec{Y}^1 (= \mathcal{V} - \{A_1, A_2\})$, and define \vec{w} as follows:

- $\vec{w}(S) = 1$;
- $\vec{w}(X) = 1$ for all $X \in \vec{X}$;
- $\vec{w}(Y^1) = 1$ and $\vec{w}(Y^0) = 0$ for all $Y \in \vec{Y}$.

We have that

$$(M, u) \models [\vec{A} \leftarrow \vec{1}, \vec{W} \leftarrow \vec{w}] \left(\bigwedge_{Y \in \vec{Y}} (Y^0 \neq Y^1) \wedge (\vec{A} = \vec{1} \wedge \vec{X} = \vec{1}) \right),$$

so

$$(M, u) \models [\vec{A} \leftarrow 1, \vec{W} \leftarrow \vec{w}] (\neg\psi_1 \wedge \neg\psi_2 \wedge (S = 1)).$$

It follows that $(M, u) \models [\vec{A} \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg\psi$, showing that AC2(a) holds.

For AC2(b), let \vec{W}' be a subset of \vec{W} . We must show that $(M, u) \models [\vec{A} \leftarrow \vec{0}, \vec{W}' \leftarrow \vec{w}] \psi$. Clearly $(M, u) \models [\vec{A} \leftarrow \vec{0}, \vec{W}' \leftarrow \vec{w}] (\vec{A} = \vec{0})$. It is immediate from the definitions of ψ_2 and ψ_3 that $(\vec{A} = \vec{0}) \Rightarrow (\psi_2 \wedge \psi_3)$ is valid, so $(M, u) \models [\vec{W}' \leftarrow \vec{w}] (\psi_2 \wedge \psi_3)$. Hence, $(M, u) \models [\vec{W}' \leftarrow \vec{w}] \psi$, as desired.

To show that AC3 holds, we need to show that neither $A_1 = 0$ nor $A_2 = 0$ is a cause of ψ in (M, u) . We prove that $A_1 = 0$ is not a cause of ψ in (M, u) ; the argument for $A_2 = 0$ not being a cause is identical.

It suffices to prove that AC2 does not hold for $A_1 = 0$. So suppose by way of contradiction that $(\vec{W}, \vec{w}, 1)$ is a witness for $A_1 = 0$ being a cause of ψ in (M, u) . Since AC2(a) holds, there must exist \vec{W} and \vec{w} such that

$$(M, u) \models [A_1 \leftarrow 1, \vec{W} \leftarrow \vec{w}] (\neg\psi_1 \wedge (\neg\psi_2 \vee \neg\psi_3) \wedge (S = 1)). \quad (3)$$

Thus, $S \in \vec{W}$ and $\vec{w}(S) = 1$ (for otherwise $(M, u) \models [A_1 \leftarrow 1, \vec{W} \leftarrow \vec{w}] (S = 0)$). Moreover, since $(M, u) \models [A_1 \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg\psi_1$, for all $Y \in \vec{Y}$, either $Y^0 \in \vec{W}$ and $\vec{w}(Y^0) = 1$ or $Y^1 \in \vec{W}$ and $\vec{w}(Y^1) = 1$, and it is not the case that both Y^0 and Y^1 are in \vec{W} and $\vec{w}(Y^0) = \vec{w}(Y^1)$.

Now consider A_2 . There are three possibilities:

- (i) $A_2 \in \vec{W}$ and $\vec{w}(A_2) = 0$;
- (ii) $A_2 \in \vec{W}$ and $\vec{w}(A_2) = 1$;
- (iii) $A_2 \notin \vec{W}$.

We show that we get a contradiction in each case.

If (i) holds, note that since

$$(M, u) \models [A_1 \leftarrow 1, \vec{W} \leftarrow \vec{w}] (A_2 = 0),$$

it follows that $(M, u) \models [A_1 \leftarrow 1, \vec{W} \leftarrow \vec{w}] \psi_2$, so by (3), $(M, u) \models [A_1 \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg \psi_3$. Since $(M, u) \models [A_1 \leftarrow 1, \vec{W} \leftarrow \vec{w}] (A_1 \neq A_2)$, it follows that $(M, u) \models [A_1 \leftarrow 1, \vec{W} \leftarrow \vec{w}] \bar{\varphi}[\vec{Y}/\vec{Y}^1]$.

Let $Z' = \emptyset$ and let $\vec{W}' = \vec{W} - \{A_2\}$. We show that $(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}] \neg \psi$, so that AC2(b) does not hold. First observe that $(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}] (A_1 \neq A_2)$. Since S and all the variables in \vec{X} , \vec{Y}^0 , and \vec{Y}^1 are in both \vec{W}' and \vec{W} , it follows from (3) that

$$(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}] (\neg \psi_1 \wedge \bar{\varphi}[\vec{Y}/\vec{Y}^1] \wedge (S = 1)).$$

Thus, $(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}] \neg \psi$, and AC2(b) does not hold.

If (ii) or (iii) hold, define an assignment ν to the variables in \vec{Y} by taking $\nu(Y) = 1$ if $Y^1 \in \vec{W}$ and $\vec{w}(Y^1) = 1$ and $\nu(Y) = 0$ if $Y^0 \in \vec{W}$ and $\vec{w}(Y^0) = 1$. (As we observed above, exactly one of these two cases occurs, so ν is well defined.) By assumption, $\forall \vec{Y} \exists \vec{X} \varphi = \mathbf{true}$, so there exists an assignment τ to the variables that makes φ true if $\tau(Y) = \nu(Y)$ for all $Y \in \vec{Y}$.

We again show that AC2(b) does not hold. Let $Z' = \emptyset$ and let $\vec{W}' = \vec{W} - \{X : \tau(X) = 0\}$. Since $S \in \vec{W}'$ and $\vec{w}(S) = 1$, it is easy to see that in both case (b) and (c), we have $(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}] (A_1 \neq A_2)$. (In case (ii), this is because $\vec{w}(A_2) = 1$; in case (iii), this is because we have the equation $A_2 = S$ and $\vec{w}(S) = 1$.) The definition of \vec{W}' ensures that

$$(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}] \bar{\varphi}[\vec{Y}/\vec{Y}^1],$$

so that $(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}] \neg \psi_3$. Hence, $(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}] \neg \psi$, again showing that AC2(b) does not hold. We conclude that AC3 holds for \vec{A} .

This completes the proof that (a) implies (c). The fact that (c) implies (b) is immediate. It remains to show that (b) implies (a). So suppose that $(M, u, \psi, \vec{A}, \vec{0})$ is in L_{AC3}^B . We must show that $\forall \vec{Y} \exists \vec{X} \varphi(\vec{X}, \vec{Y}) = \mathbf{true}$.

Let ν be some assignment to \vec{Y} . Let $\vec{W} = \{S\} \cup \vec{X} \cup \vec{Y}^0 \cup \vec{Y}^1$ and define \vec{w} as follows:

- $\vec{w}(S) = 1$;
- $\vec{w}(X) = 1$ for all $X \in \vec{X}$;
- $\vec{w}(Y^{\nu(y)}) = 1$ and $\vec{w}(Y^{1-\nu(y)}) = 0$ for all $Y \in \vec{Y}$.

Since AC3 holds, $A_1 \leftarrow 0$ cannot be a cause of ψ in (M, u) with witness $(\vec{W}, \vec{w}, 1)$. It is straightforward to check that $(M, u) \models [A_1 \leftarrow 1, \vec{W} \leftarrow \vec{w}] \neg \psi$, using the fact that $\vec{w}(S) = 1$. Hence, AC2(a) holds for $A_1 \leftarrow 0$. AC3 holds trivially, and we have already observed that AC1 holds. Thus, AC2(b) cannot hold for $A_2 \leftarrow 0$, that is, there exist $\vec{W}' \subseteq \vec{W}$ and $\vec{Z}' \subseteq \{A_2\}$ such that $(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}, \vec{Z}' \leftarrow \vec{z}^*] \neg \psi$. It follows that

- $S \in \vec{W}'$ and $\vec{w}(S) = 1$; and
- either $Y^0 \in \vec{W}'$ and $\vec{w}(Y^0) = 1$ or $Y^1 \in \vec{W}'$ and $\vec{w}(Y^1) = 1$, and it is not the case that both Y^0 and Y^1 are in \vec{W}' and $\vec{w}(Y^0) = \vec{w}(Y^1)$.

Since $(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}, \vec{Z}' \leftarrow \vec{z}^*] \psi_2$, it must be the case that $(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}, \vec{Z}' \leftarrow \vec{z}^*] \neg \psi_3$. This, in turn, implies that $(M, u) \models [A_1 \leftarrow 0, \vec{W}' \leftarrow \vec{w}, \vec{Z}' \leftarrow \vec{z}^*] \bar{\varphi}[\vec{Y}/\vec{Y}^1]$. Now define $\tau(X) = 1$ iff $X \in \vec{W}'$. It is immediate that τ satisfies φ if the values of Y are assigned according to ν . It follows that $\forall \vec{Y} \exists \vec{X} \varphi(\vec{X}, \vec{Y}) = \mathbf{true}$, as desired.

Appendix C. Proof of Theorem 4.4

As mentioned in the proof outline, membership is straightforward, so we focus on hardness. Note that it suffices to prove that L_{cause}^B is D_2^P -hard; the fact that L_{cause} is D_2^P -hard follows immediately.

In order to prove that L_{cause}^B is D_2^P -hard, first consider the language $\text{SAT}^2 = \Sigma_2^P(\text{SAT}) \times \Pi_2^P(\text{SAT})$. By Lemma 3.2, SAT^2 is D_2^P -complete. Thus, it suffices to reduce SAT^2 to L_{cause}^B . We do this by combining the constructions of Theorems 4.1 and 4.3 as follows. Given a pair $\langle \exists \vec{X} \forall \vec{Y} \varphi, \forall \vec{Y}' \exists \vec{X}' \varphi' \rangle$, we assume without loss of generality that the formulas in the pair use disjoint sets of variables. For simplicity, we use primed versions of variable names for the second component of the pair. Let $\langle M, u, \psi, A, 0 \rangle$ be the tuple constructed in the proof of Theorem 4.1, and let $\langle M', u', \psi', \langle A'_1, A'_2 \rangle, \langle 0, 0 \rangle \rangle$ be the tuple constructed in the proof of Theorem 4.3 (using primed variable names). Recall that the proof of Theorem 4.1 showed that $\exists \vec{X} \forall \vec{Y} \varphi = \mathbf{true}$ iff $A = 0$ is a cause of ψ in (M, u) , and the proof of Theorem 4.3 showed that $\forall \vec{Y}' \exists \vec{X}' \varphi' = \mathbf{true}$ iff $\vec{A}' = \vec{0}$ is a cause of ψ' in (M', u) .

The reduction from SAT^2 constructs a model M'' that can be viewed as the union of M and M' . That is, the set of endogenous variables in M'' is the union of the endogenous variables in M and in M' , and similarly for the exogenous variables, and the equations for M'' consist of the union of the equations for M and M' . We claim that $A = 0 \wedge \vec{A}' = \vec{0}$ is a cause of $\psi \vee \psi'$ in (M'', u) iff $\langle \exists \vec{X} \forall \vec{Y} \varphi, \forall \vec{Y}' \exists \vec{X}' \varphi' \rangle \in \text{SAT}^2$. (Note that $(M'', u) \models \psi \wedge \psi'$; thus, to make $\psi \vee \psi'$ false, both ψ and ψ' must be false.)

First suppose that $\langle \exists \vec{X} \forall \vec{Y} \varphi, \forall \vec{Y}' \exists \vec{X}' \varphi' \rangle \in \text{SAT}^2$. Then, $A = 0$ is a cause of ψ in (M, u) and $\vec{A}' = \vec{0}$ is a cause of ψ' in (M', u') . It easily follows that $A = 0 \wedge \vec{A}' = \vec{0}$ is cause of $\psi \vee \psi'$ in (M'', u) .

The other direction is similar. If $A = 0 \wedge \vec{A}' = \vec{0}$ is a cause of $\psi \vee \psi'$ in (M'', u) , then, since the non-primed variables affect only the value of ψ , $A = 0$ is a cause of ψ in (M, u) , and similarly, $\vec{A}' = \vec{0}$ is a cause of ψ' in (M', u) . It follows from Theorem 4.1 and 4.3 that $\exists \vec{X} \forall \vec{Y} \varphi \in \Sigma_2^P(\text{SAT})$ and $\forall \vec{Y}' \exists \vec{X}' \varphi' \in \Pi_2^P(\text{SAT})$, therefore $\langle \exists \vec{X} \forall \vec{Y} \varphi, \forall \vec{Y}' \exists \vec{X}' \varphi' \rangle \in \text{SAT}^2$.

References

- Alechina, N., Halpern, J. Y., & Logan, B. (2016). Causality, responsibility, and blame in team plans. Unpublished manuscript.
- Aleksandrowicz, G., Chockler, H., Halpern, J. Y., & Ivrii, A. (2014). The computational complexity of structure-based causality. In *Proc. Twenty-Eighth National Conference on Artificial Intelligence (AAAI '14)*, pp. 974–980.
- Beer, I., Ben-David, S., Chockler, H., Orni, A., & Treffer, R. J. (2012). Explaining counterexamples using causality. *Formal Methods in System Design*, 40(1), 20–40.
- Chang, R., & Kadin, J. (1996). The Boolean hierarchy and the polynomial hierarchy: a closer connection. *SIAM Journal on Computing*, 25(2), 340–354.
- Chockler, H., Fenton, N. E., Keppens, J., & Lagnado, D. A. (2015). Causal analysis for attributing responsibility in legal cases. In *Proceedings of the 15th International Conference on Artificial Intelligence and Law, ICAIL*, pp. 33–42. ACM.
- Chockler, H., Halpern, J. Y., & Kupferman, O. (2008). What causes a system to satisfy a specification?. *ACM Transactions on Computational Logic*, 9(3).

- Chockler, H., & Halpern, J. (2004). Responsibility and blame: a structural-model approach. *Journal of Artificial Intelligence Research (JAIR)*, 22, 93–115.
- Collins, J., Hall, N., & Paul, L. A. (Eds.). (2004). *Causation and Counterfactuals*. MIT Press, Cambridge, MA.
- Eiter, T., & Lukasiewicz, T. (2002). Complexity results for structure-based causality. *Artificial Intelligence*, 142(1), 53–89.
- Garey, M., & Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. Freeman and Co., San Francisco.
- Gerstenberg, T., & Lagnado, D. (2010). Spreading the blame: the allocation of responsibility amongst multiple agents. *Cognition*, 115, 166–171.
- Hall, N. (2004). Two concepts of causation. In Collins, J., Hall, N., & Paul, L. A. (Eds.), *Causation and Counterfactuals*. MIT Press, Cambridge, MA.
- Halpern, J. Y. (2008). Defaults and normality in causal structures. In *Principles of Knowledge Representation and Reasoning: Proc. Eleventh International Conference (KR '08)*, pp. 198–208.
- Halpern, J. Y. (2015). A modification of the Halpern-Pearl definition of causality. In *Proc. 24th International Joint Conference on Artificial Intelligence (IJCAI 2015)*, pp. 3022–3033.
- Halpern, J. Y. (2016). *Actual Causality*. MIT Press, Cambridge, MA.
- Halpern, J. Y., & Pearl, J. (2001). Causes and explanations: A structural-model approach. Part I: Causes. In *Proc. Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI 2001)*, pp. 194–202.
- Halpern, J. Y., & Pearl, J. (2005a). Causes and explanations: a structural-model approach. Part I: Causes. *British Journal for Philosophy of Science*, 56(4), 843–887.
- Halpern, J., & Pearl, J. (2005b). Causes and explanations: A structural-model approach. part 1: Causes. *British Journal for Philosophy of Science*, 56(4), 843–887.
- Hopkins, M. (2001). A proof of the conjunctive cause conjecture. Unpublished manuscript.
- Hopkins, M., & Pearl, J. (2003). Clarifying the usage of structural models for commonsense causal reasoning. In *Proc. AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*.
- Hume, D. (1739). *A Treatise of Human Nature*. John Noon, London.
- Jenner, B., & Toran, J. (1995). Computing functions with parallel queries to NP. *Theoretical Computer Science*, 141, 175–193.
- Johnson, D. S. (1990). A catalog of complexity classes. In Leeuwen, J. v. (Ed.), *Handbook of Theoretical Computer Science*, Vol. A, chap. 2. Elsevier Science.
- Krentel, M. (1988). The complexity of optimization problems. *Journal of the CSS*, 36, 490–509.
- Lagnado, D. A., Gerstenberg, T., & Zultan, R. (2013). Causal responsibility and counterfactuals. *Cognitive Science*, 37, 1036–1073.
- Papadimitriou, C. H. (1984). The complexity of unique solutions. *Journal of ACM*, 31, 492–500.

- Papadimitriou, C. H., & Yannakakis, M. (1984). The complexity of facets (and some facets of complexity). *J. Comput. Syst. Sci.*, 28(2), 244–259.
- Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press, New York.
- Sipser, M. (2012). *Introduction to Theory of Computation* (third edition). Thomson Course Technology, Boston.
- Stockmeyer, L. J. (1977). The polynomial-time hierarchy. *Theoretical Computer Science*, 3, 1–22.
- Wrathall, C. (1976). Complete sets and the polynomial-time hierarchy. *Theoretical Computer Science*, 3(1), 23–33.