How Rules Represent Causal Knowledge: Causal Modeling with Abductive Logic Programs

Kilian Rückschloß[1,*,†], Felix Weitkämper[2,3,†]

[1]Eberhard Karls Universität Tübingen, Auf der Morgenstelle 10 (C Bau), 72076 Tübingen, Germany
[2]German University of Digital Science, Marlene-Dietrich-Allee 14, 14482 Potsdam, Germany
[3]Ludwig-Maximilians-Universität München, Oettingenstr. 67, 80538 München, Germany

Abstract

Pearl observes that causal knowledge enables predicting the effects of interventions, whereas descriptive knowledge only permits drawing conclusions from observations. This paper brings Pearl's approach to causality and interventions into stratified abductive logic programming. It shows how stable models of such programs can be given a causal interpretation by building on philosophical foundations developed in recent work by Bochman and Eelink et al. In particular, it translates abductive logic programs into the causal systems of Eelink et al., thereby clarifying the informal causal reading of logic program rules. The main results establish that the stable model semantics for stratified programs conforms to key philosophical principles of causation, including causal sufficiency, natural necessity, and the irrelevance of unobserved effects. This justifies the use of stratified abductive logic programs as a framework for causal modeling and for predicting the effects of interventions.

Keywords

Causal Logic, Stable Model Semantics, Abductive Logic Programming, Interventions, Do-Calculus, Explainable Artificial Intelligence.

1. Introduction

After being a central topic of philosophical inquiry for over two millennia, causality entered the mainstream of artificial intelligence research through the work of Pearl [1]. A key feature of his account is that causal knowledge goes beyond descriptive knowledge in the questions it can address: while descriptive knowledge permits only inferences from observations, causal knowledge enables reasoning about the effects of external interventions such as actions on the modeled system.

Example 1.1. *Consider a road that passes through a field with a sprinkler. Assume the sprinkler is turned on by a weather sensor when it is sunny. Suppose further that it rains whenever it is cloudy, and that the road becomes wet if either it rains or the sprinkler is activated. Finally, assume that a wet road is dangerous.*

Observing that the sprinkler is on, one might conclude that the weather is sunny. However, actively intervening and switching the sprinkler on does not cause the weather to become sunny. To predict the effect of such an intervention, one needs causal—not merely descriptive—knowledge.

Since evaluating the effects of possible actions is one of the primary motivations for modeling in the first place, this has paved the way for the adoption of causal frameworks across a wide variety of domains [2, 3, 4].

Pearl [1], however, develops his theory of causality exclusively within his own formalisms: Bayesian networks and structural causal models.

Example 1.2. *Recall Example 1.1, and denote by c the event that the weather is cloudy, by s the event that the sprinkler is on, by r the event of rain, by w the event that the road is wet, and by d the event that the road is dangerous.*

RuleML+RR'25: Companion Proceedings of the 9th International Joint Conference on Rules and Reasoning, September 22–24, 2025, Istanbul, Turkiye

*Corresponding author.
†These authors contributed equally.
kilian.rueckschloss@uni-tuebingen.de (K. Rückschloß); felix.weitkaemper@german-uds.de (F. Weitkämper)
0000-0002-7891-6030 (K. Rückschloß); 0000-0002-3895-8279 (F. Weitkämper)

Pearl [1] models these causal mechanisms as a system of structural equations:

$$r := c \qquad s := \neg c \qquad w := r \vee s \qquad d := w \tag{1}$$

Since the mechanisms do not specify whether it is cloudy, Pearl [1] treats c as an external variable or error term. The solutions of the corresponding causal model $\mathscr{M}$ are obtained by solving Equations (1) for $c = \top$ and $c = \bot$. If the sprinkler is observed to be on, then according to Equations (1), it must be sunny.

The intervention of manually switching the sprinkler on is represented by modifying the causal mechanism for the sprinkler so that it is on regardless of the weather. This is captured in the structural equations below:

$$r := c \qquad s := \top \qquad w := r \vee s \qquad d := w \tag{2}$$

Again, c is considered an external variable, and the solutions of the corresponding causal model $\mathscr{M}_s$ represent the possible states of the world after the intervention of switching the sprinkler on manually. Note that s is now true independently of the weather.

In philosophy, the idea that causal explanations are given by rules of the form "ϕ *causes* ψ" is well established, for instance, in René Descartes' *Principles of Philosophy* II:37 (see the translation by Miller and Miller [5]). This makes abductive logic programming [6] a natural target formalism for representing causal knowledge. An abductive logic program consists of a set of rules $\mathbf{P}$ and a set of propositions $\mathfrak{A}$, called *abducibles*. Similar to external variables in Pearl's causal models, abducibles are independently assumed to be either true or false.

Rückschloß and Weitkämper [7] apply Clark completion [8] to relate probabilistic logic programming to Pearl's theory of causality, thereby transferring Pearl's notion of intervention. Similarly, abductive logic programs can be translated into causal models, enabling a principled treatment of interventions.

Example 1.3. *The situation in Example 1.1 gives rise to the rules:*

$$r \leftarrow c \qquad s \leftarrow \neg c \qquad w \leftarrow r \qquad w \leftarrow s \qquad d \leftarrow w \tag{3}$$

Since c can be either true or false, it is considered an abducible; that is, $\mathfrak{A} := \{c\}$. Reading Pearl's ":=" sign as logical equivalence "$\leftrightarrow$," the Clark completion [8] implies that the models of the resulting abductive logic program coincide with the solutions of the causal model $\mathscr{M}$ in Equations (1) of Example 1.2.

Intervening and switching the sprinkler on manually results in the rules:

$$r \leftarrow c \qquad s \leftarrow \top \qquad w \leftarrow r \qquad w \leftarrow s \qquad d \leftarrow w \tag{4}$$

Again, Clark completion [8] yields that the models of the resulting abductive logic program coincide with the solutions of the causal model $\mathscr{M}_s$ in Equations (2) of Example 1.2.

Notably, this approach provides an informal semantics for abductive logic programs, where each rule $h \leftarrow b_1, \ldots, b_n$ is interpreted as "$b_1 \wedge \ldots \wedge b_n$ *causes* h". However, translating abductive logic programs into causal models via the Clark completion [8] and replacing logical equivalence "$\leftrightarrow$" with Pearl's ":=" sign can lead to counterintuitive results when cyclic causal relations are involved.

Example 1.4. *Assume h_1 and h_2 are two neighboring houses. Let f_i denote the event that House h_i is on fire, and sf_i the event that House h_i starts burning, for $i \in \{1, 2\}$. It is reasonable to assume that a fire in House h_1 leads to a fire in House h_2, and vice versa.*

This situation can be modeled by the cyclic abductive logic program $\mathscr{P}$, consisting of the abducibles $\mathfrak{A} := \{sf_1, sf_2\}$ and the rules:

$$f_1 \leftarrow sf_1 \qquad f_2 \leftarrow sf_2 \qquad f_2 \leftarrow f_1 \qquad f_1 \leftarrow f_2 \tag{5}$$

Proceeding as in Example 1.3, forming the Clark completion [8] and replacing logical equivalence "$\leftrightarrow$" with Pearl's ":=" sign relates $\mathscr{P}$ to a causal model that admits a solution corresponding to the structure $\omega := \{f_1, f_2\}$. In ω, both houses are on fire even though neither house actually started to burn. This contradicts the intuition that houses do not spontaneously catch fire merely because they influence each other.

This work extends the applicability of Pearl's ideas to stratified abductive logic programs with cyclic causal relations. Building on prior work by Bochman [9] and Eelink et al. [10], it connects logic programming to explanations that satisfy Principles 1–5 from philosophy, as stated below.

Principle 1 (Causal Foundation)**.** *Causal explanations originate from external premises, whose explanations lie beyond a given scope.*

Principle 2 (Natural Necessity)**.** *"...given the existence of the cause, the effect must necessarily follow." (Thomas Aquinas:* Summa Contra Gentiles *II:35.4; translation by Anderson [11])*

Principle 3 (Sufficient Causation)**.** *"...there is nothing without a reason, or no effect without a cause." (Gottfried Wilhelm Leibniz:* First Truths*; translation by Loemker [12], p. 268)*

Principle 4 (Causal Irrelevance [13])**.** *Effects beyond the given scope have no influence on beliefs.*

Principle 5 (Non-Interference)**.** *The impact of interventions is restricted to the direction from causes to effects.*

If propositions can only be explained through other propositions, the resulting explanations are either cyclic or lead to an infinite regress. As Aristotle argues in his *Posterior Analytics* (see Barnes' translation [14, pp. 117–118]), infinite chains and cyclical arguments are not genuinely explanatory. To circumvent this problem, Principle 1 assumes that explanations take place within a given scope, just as explanations in chemistry rely on results from physics. Principle 2 states that every statement that can be explained indeed takes place, and Principle 3 states that every statement that takes place can be explained. Principle 4 states that extending the scope and including further effects does not change the result of the initial explanations. For example, if one adds a rule $i \leftarrow d$ to Equations (3) of Example 1.3, stating that somebody is injured if the road is dangerous, this does not affect the causal explanation of why the road is dangerous. Finally, Principle 5 states that the effect of interventions propagates exclusively from causes to effects. As illustrated in Example 1.1, this means, for instance, that switching on the sprinkler manually has no effect on the weather.

Theorem 4.2 translates abductive logic programs into the causal systems of Eelink et al. [10], thereby relating the stable model semantics to Principles 1, 2, and 3. Theorem 4.3 shows that Principle 4 implies 5. Finally, Theorem 4.4 establishes that stratified programs satisfy Principle 4.

Overall, the results show that stratified abductive logic programs under the stable model semantics conform to these principles, supporting their use in causal modeling and the prediction of effects from external interventions.

2. Preliminaries

This section recalls the foundations of the present work: Pearl's causal models [1], abductive logic programs [6], and the logical theories of causality developed by Bochman [9] and Eelink et al. [10].

2.1. Pearl's Causal Models

Pearl [1] suggests modeling causal relationships with deterministic functions. This leads to the following definition of structural causal models.

Definition 2.1 (Causal Model [1, §7.1.1])**.** *A* ***(structural) causal model*** $\mathscr{M}$ *with* ***internal variables*** $\boldsymbol{V}$ *and* ***external variables*** $\boldsymbol{U}$ *is a system of equations that includes one* ***structural equation*** *of the form* $X := f_X(\mathrm{Pa}(X), \mathrm{Error}(X))$ *for each internal variable* $X \in \boldsymbol{V}$*. Here, the* ***parents*** $\mathrm{Pa}(X) \subseteq \boldsymbol{V}$ *of* X *are a subset of internal variables, the* ***error term*** $\mathrm{Error}(X) \subseteq \boldsymbol{U}$ *is a subset of external variables, and the* ***causal mechanism*** *of* X *is a function* f_X *that maps each assignment of values to* $\mathrm{Pa}(X)$ *and* $\mathrm{Error}(X)$ *to a value of* X*.*

A ***solution*** ω *of the structural causal model* $\mathscr{M}$ *is an assignment of values to the variables in* $\boldsymbol{V} \cup \boldsymbol{U}$ *that satisfies all structural equations.*

Notation 2.2. *The parents* $\mathrm{Pa}(V)$ *and error terms* $\mathrm{Error}(V)$ *of an internal variable* $V \in \boldsymbol{V}$ *are typically evident from the causal mechanism* f_V*. Accordingly, this work omits explicit references to* $\mathrm{Pa}(\cdot)$ *and* $\mathrm{Error}(\cdot)$*.*

Example 2.1. *The causal model* $\mathscr{M}$ *in Example 1.2 has external variables* $\boldsymbol{U} := \{c\}$*, internal variables* $\boldsymbol{V} := \{r, s, w, d\}$*, Structural Equations (1), and solutions:*

ω_1 :	$c = \top$	$r = \top$	$s = \bot$	$w = \top$	$d = \top$
ω_2 :	$c = \bot$	$r = \bot$	$s = \top$	$w = \top$	$d = \top$

In artificial intelligence, causal models are particularly valuable because they can represent external interventions. As explained in Chapter 7 of Pearl [1], the key idea is to construct a modified model that incorporates the minimal changes to the structural equations required to enforce an external intervention.

Definition 2.3 (Modified Causal Model)**.** *Fix a causal model* $\mathscr{M}$*. Let* $\mathbf{I}$ *be a subset of internal variables with a value assignment* $\mathbf{i}$*. The* ***modified model*** *or* ***submodel*** $\mathscr{M}_{\mathbf{i}}$ *is the model obtained from* $\mathscr{M}$ *by replacing, for each variable* $X \in \mathbf{I}$*, the structural equation* $X := f_X(\mathrm{Pa}(X), \mathrm{Error}(X))$ *with* $X := \mathbf{i}(X)$*.*

Notation 2.4. *Let* $V \in \boldsymbol{V}$ *be a Boolean internal variable of a structural causal model* $\mathscr{M}$*. In this case, one writes* $\mathscr{M}_V := \mathscr{M}_{V:=\top}$ *and* $\mathscr{M}_{\neg V} := \mathscr{M}_{V:=\bot}$*.*

Example 2.2. *The causal model* $\mathscr{M}_s$ *from Example 1.2 is the modified model corresponding to the value assignment* $s := \top$*. It has the following solutions:*

ω_1 :	$c = \top$,	$r = \top$,	$s = \top$,	$w = \top$,	$g = \top$
ω_2 :	$c = \bot$,	$r = \bot$,	$s = \top$,	$w = \top$,	$g = \top$

These represent the possible states of the system after manually switching on the sprinkler.

Remark. *As in Example 1.2, actions often force a variable in a causal model to take on a new value. Pearl [1] emphasizes that submodels* $\mathscr{M}_{\mathbf{i}}$ *typically arise from performing actions that set certain variables to specific values, a process formalized by the introduction of the* do*-operator.*

2.2. Abductive Logic Programming

This work adopts standard notation for propositions, (propositional) formulas, and structures. A structure is identified with the set of propositions true in it. The term *world* denotes a consistent set of literals that is maximal with respect to inclusion. Since identifying structures with the set of literals they render true yields a one-to-one correspondence between worlds and structures, the two terms are used interchangeably.

Example 2.3. *A structure* ω *in the propositional alphabet* $\mathfrak{P} := \{c,\ r,\ s,\ w,\ d\}$ *of Example 2.1 is a complete state description such as* ω_1 *in Example 2.1. It is identified with the set of propositions* $\{c,\ r,\ w,\ d\}$ *and the world* $\{c,\ r, \neg s,\ w,\ d\}$*.*

Fix a propositional alphabet $\mathfrak{P}$. Logic programs consist of rules or clauses.

Definition 2.5 (Clauses and Logic Programs)**.** *A* ***(normal) clause*** C *is a formula of the form* $(h \leftarrow (b_1 \wedge (b_2 \wedge (... \wedge b_n))...))$*, which is also denoted as* $h \leftarrow b_1 \wedge ... \wedge b_n$*,* $h \leftarrow b_1, ..., b_n$ *or* $\mathrm{head}(C) \leftarrow \mathrm{body}(C)$*. Here,* $\mathrm{head}(C) := h$ *is an atom, referred to as the* ***head*** *of the clause* C *and* $\mathrm{body}(C) := \{b_1, ..., b_n\}$ *is a finite set of literals, known as the* ***body*** *of* C*. If* $\mathrm{body}(C) = \emptyset$ *and* $C = (h \leftarrow \top)$*, one denotes* C *by* h *and calls* C *a* ***fact****.*

A ***logic program*** $\boldsymbol{P}$ *is a finite set of clauses. The* ***dependence graph*** *of* $\boldsymbol{P}$ *is the directed graph over the alphabet* $\mathfrak{P}$ *defined as follows: there is an edge* $p \to q$ *if and only if there exists a clause* $C \in \boldsymbol{P}$ *such that* $\mathrm{head}(C) = q$ *and* $\mathrm{body}(C) \cap \{p, \neg p\} \neq \emptyset$*. It is denoted by* $\mathrm{graph}(\boldsymbol{P})$*.*

An edge $p \xrightarrow{-} q$ *in* graph($\boldsymbol{P}$) *is* ***negative*** *if there exists a clause* $C \in \boldsymbol{P}$ *such that* head(C) $= q$ *and* $\neg p \in$ body(C). *Similarly, an edge* $p \xrightarrow{+} q$ *is* ***positive*** *if there exists a clause* $C \in \boldsymbol{P}$ *such that* head(C) $= q$ *and* $p \in$ body(C). *Note that an edge may be both negative and positive simultaneously.*

A ***cycle*** *in* graph($\boldsymbol{P}$) *is a finite alternating sequence of nodes and edges of the form* $q \to p_1 \to p_2 \to \cdots \to p_n \to q$ *that begins and ends at the same node* q. *The program* $\boldsymbol{P}$ *is* ***acyclic*** *if its dependence graph* graph($\boldsymbol{P}$) *contains no cycle. It is* ***stratified*** *if its dependence graph does not contain a cycle with a negative edge.*

Clark [8] translates acyclic programs **P** to propositional formulas, stating that a valid proposition in a model ω needs to have a reason, i.e., a support in ω.

Definition 2.6 (Clark Completion, Supported Model Semantics)**.** *Let* $\boldsymbol{P}$ *be a logic program. The* ***Clark completion*** *of* $\boldsymbol{P}$ *is the set of formulas*

$$\text{comp}(\boldsymbol{P}) := \left\{ p \leftrightarrow \bigvee_{C \in \boldsymbol{P},\ \text{head}(C)=p} \bigwedge \text{body}(C) \right\}_{p \in \mathfrak{P}}.$$

A ***supported model*** *of* $\boldsymbol{P}$ *is a model of the Clark completion* $\omega \models \text{comp}(\boldsymbol{P})$.

Remark. *If there is no clause* $C \in \mathbf{P}$ *with* head(C) $= p$ *for a proposition* $p \in \mathfrak{P}$, *then* $p \leftrightarrow \bot \in \text{comp}(\mathbf{P})$, *since the disjunction over the empty set evaluates to false.*

Example 2.4. *Rules (3) define an acyclic logic program* $\boldsymbol{P}$ *whose unique supported model is* ω_2 *in Example 2.1.*

Although the supported model semantics is formally well-defined for general propositional logic programs, i.e., it associates a unique (possibly empty) set of models to each program **P**, it yields counterintuitive results for cyclic programs.

Example 2.5. *Rules (5) define a stratified logic program* $\boldsymbol{P}$ *with two supported models:* $\omega_1 := \emptyset$ *and* $\omega_2 := \{f_1, f_2\}$. *In* ω_2, *both houses are on fire, even though there is no initial cause for either to start burning – contradicting everyday intuition.*

For general, potentially cyclic programs, Gelfond and Lifschitz [15] argue that the stable model semantics provides a more appropriate notion of a model. Rather than adopting the more common formulation via reducts, this work follows the equivalent definition based on *unfounded sets*, originally introduced by Saccà and Zaniolo [16] and listed as Definition D in Lifschitz [17].

Definition 2.7 (Unfounded Sets and Stable Models)**.** *Let* ω *be a structure,* $I \subseteq \omega$ *a non-empty subset of the set of atoms that are true in* ω, *and* $\boldsymbol{P}$ *a logic program. Then,* I *is an* ***unfounded set*** *with respect to* ω *and* $\boldsymbol{P}$ *if, for each* $p \in I$, *every rule in* $\boldsymbol{P}$ *with head* p *has some body literal* b *that is either not true in* ω *or belongs to* I.

A structure ω *is a* ***stable model*** *of* $\boldsymbol{P}$ *if it satisfies every clause of* $\boldsymbol{P}$ *when interpreted as a propositional formula and if there is no unfounded set* $I \subseteq \omega$ *with respect to* ω *and* $\boldsymbol{P}$.

Example 2.6. *In Example 2.5, the only stable model is* ω_1, *as intended. The set* ω_2 *is not stable because it is unfounded with respect to itself and the program* $\boldsymbol{P}$.

Gelfond and Lifschitz [15] prove the following results:

Theorem 2.1 (Supported and Stable Models)**.** *Every stable model of a logic program is also a supported model* □.

Theorem 2.2 (Stable Models of Stratified Programs)**.** *Every stratified program has a unique stable model.* □

Abductive logic programming was identified as a distinct branch of logic programming by Kakas and Mancarella [6], with the goal of providing an explanation for a given set of observations in terms of so-called abducibles.

Definition 2.8 (Abductive Logic Program [18])**.** *An **integrity constraint** IC is an expression of the form* $\bot \leftarrow b_1 \wedge \ldots \wedge b_n$ *also written* $\bot \leftarrow \mathrm{body}(IC)$*, where* $\mathrm{body}(IC)$ *is a finite set of literals.*

*An **abductive logic program** is a triplet* $\mathscr{P} := (\boldsymbol{P}, \mathfrak{A}, \boldsymbol{IC})$ *consisting of a logic program* $\boldsymbol{P}$*, a finite set of integrity constraints* $\boldsymbol{IC}$ *and a set of* $\boldsymbol{abducibles}$ $\mathfrak{A} \subseteq \mathfrak{P}$ *such that no abducible* $u \in \mathfrak{A}$ *is the head of a clause in* $\boldsymbol{P}$*. Finally,* $\mathscr{P}$ *is* $\boldsymbol{acyclic}$ *or* $\boldsymbol{stratified}$ *if the underlying logic program* $\boldsymbol{P}$ *is.*

In the context of databases, integrity constraints serve as sanity checks on data [19, Chapter 9]. In a causal setting, they are used to represent observations; that is, they ensure that the knowledge encoded by the causal rules in the program $\mathbf{P}$ and the explanations in $\mathfrak{A}$ is consistent with the given observations.

Example 2.7. *Let* $\mathbf{P}$ *denote the logic program in Example 2.4. Since the causal knowledge in Example 1.1 is expected to be insufficient to explain whether it is cloudy, c is declared as the only abducible, that is,* $\mathfrak{A} := \{c\}$*. One may also observe that the sprinkler is on, leading to the integrity constraint* $\mathbf{IC} := \{\bot \leftarrow \neg s\}$*. Together, this yields the abductive logic program* $\mathscr{P} := (\mathbf{P}, \mathfrak{A}, \mathbf{IC})$*.*

Lastly, the various semantics of an abductive logic program are recalled.

Definition 2.9 (Models of Abductive Logic Programs)**.** *A* $\boldsymbol{stable}$ *or* $\boldsymbol{supported\ model}$ $\omega \subseteq \mathfrak{P}$ *of the abductive logic program* $\mathscr{P} := (\boldsymbol{P}, \mathfrak{A}, \boldsymbol{IC})$ *satisfies the integrity constraints* $\boldsymbol{IC}$*, i.e.,* $\omega \vDash IC$ *(meaning* $\omega \nvDash \mathrm{body}(IC)$*) for all* $IC \in \boldsymbol{IC}$ *and is a stable or supported model of the program* $\boldsymbol{P} \cup (\omega \cap \mathfrak{A})$*. The set* $\epsilon := \omega \cap \mathfrak{A}$ *is then called the* $\boldsymbol{explanation}$ *of* ω*. In this context, the program* $\mathscr{P}$ *is* $\boldsymbol{consistent}$ *if it has at least one model for every choice of abducibles.*

Remark. *If the abductive logic program* $\mathscr{P} := (\mathbf{P}, \mathfrak{A}, \mathbf{IC})$ *is not consistent, there exists a truth-value assignment* ϵ *on* $\mathfrak{A}$ *such that* $\mathscr{P}$ *has no model with explanation* ϵ*. In this case, one would conclude, against the direction of cause and effect, that* ϵ *is impossible, contradicting Principle 4.*

Example 2.8. *The abductive logic program in Example 1.3 has two stable and supported models, namely* ω_1 *and* ω_2 *from Example 2.1, with explanations* $\epsilon_1 := \{c\}$ *and* $\epsilon_2 := \emptyset$*, respectively.*

Since only ω_2 *is consistent with the observation that the sprinkler is on, expressed by the integrity constraint* $\bot \leftarrow \neg s$*, it is the only supported model of the abductive logic program* $\mathscr{P}$ *in Example 2.7.*

Rückschloß and Weitkämper [7] connect probabilistic logic programming to Pearl's theory of causality. Following their approach, the Clark completion [8] of abductive logic programs without integrity constraints gives rise to causal models, thereby transferring Pearl's notion of an intervention:

Definition 2.10 (CM-Semantics)**.** *Let* $\mathscr{P} := (\boldsymbol{P}, \mathfrak{A}, \emptyset)$ *be an abductive logic program without integrity constraints. The* $\boldsymbol{causal\ model\ semantics}$ *of* $\mathscr{P}$ *is the causal model* $\mathrm{CM}(\boldsymbol{P})$ *that is given by the external variables* $\mathfrak{A}$*, the internal variables* $\mathfrak{P} \setminus \mathfrak{A}$ *and the structural equations* $p := \bigvee_{\substack{C \in \boldsymbol{P} \\ \mathrm{head}(C) = p}} \bigwedge \mathrm{body}(C)$*.*

To represent the intervention of forcing the atoms in $\boldsymbol{I} \subseteq \mathfrak{P} \setminus \mathfrak{A}$ *to attain values according to the assignment* $\boldsymbol{i}$*, the* $\boldsymbol{modified\ (abductive\ logic)\ program}$ $\mathscr{P}_{\boldsymbol{i}} := (\boldsymbol{P_i}, \mathfrak{A}_{\boldsymbol{i}}, \emptyset)$ *is obtained from* $\mathscr{P}$ *by the modifications below:*

> *Remove all clauses C from* $\boldsymbol{P}$ *for which* $\mathit{head}(C) \in \boldsymbol{i}$ *or* $\neg \mathit{head}(C) \in \boldsymbol{i}$*.*
> *Add a fact p to* $\boldsymbol{P_i}$ *whenever* $p \in \boldsymbol{i}$*.*

Remark. *By construction, a structure* ω *is a solution of* $\mathrm{CM}(\mathbf{P})$ *if and only if it is a supported model of* $\mathbf{P}$*. Moreover, by construction,* $\mathrm{CM}(\mathbf{P_i}) = \mathrm{CM}(\mathbf{P})_{\mathbf{i}}$*.*

Example 2.9. *In Example 1.3, Rules (4) correspond to the modified program* $\mathscr{P}_s$ *that corresponds to the assignement* $s := \top$*.*

2.3. Bochman's Logical Theory of Causality and Causal Systems

To verify that the stable model semantics is causally meaningful, this contribution builds upon the work of Bochman [9] and Eelink et al. [10]. Both rely on the idea that causal knowledge should be expressed in the form of rules.

Definition 2.11 (Causal Rules and Causal Theories)**.** *A **(literal) causal rule** R is an expression of the form $b_1 \wedge ... \wedge b_n \Rightarrow l$, also denoted by $\{b_1, ..., b_n\} \Rightarrow l$, where $b_1, ..., b_n, l$ are literals. One calls $b_1 \wedge \cdots \wedge b_n$ the **cause** and l the **effect** of R. Informally, R means that $b_1 \wedge \cdots \wedge b_n$ causes l. If, in addition, $l \in \mathfrak{P}$ is an atom, the rule R is **atomic**. A **default rule** is a causal rule of the form $l \Rightarrow l$ for one literal l. A **causal theory** is a set of causal rules Δ. It is called **atomic** if it contains only atomic causal rules.*

Remark. *Note that causation "$\Rightarrow$" is not reflexive; that is, $l \Rightarrow l$ does not hold for all literals l. Bochman [9] interprets default rules of the form $l \Rightarrow l$ as indicating that the literal l is self-explained, meaning it can serve as a starting point for a causal explanation.*

Example 2.10. *The situation in Example 1.4 gives rise to the following causal theory Δ:*

$$sf_1 \Rightarrow sf_1 \qquad sf_2 \Rightarrow sf_2 \qquad sf_1 \Rightarrow f_1 \qquad sf_2 \Rightarrow f_2 \qquad f_1 \Rightarrow f_2 \qquad f_2 \Rightarrow f_1$$
$$\neg sf_1 \Rightarrow \neg sf_1 \qquad \neg sf_2 \Rightarrow \neg sf_2 \qquad \neg f_1 \Rightarrow \neg f_1 \qquad \neg f_2 \Rightarrow \neg f_2$$

The default rule $\neg f_1 \Rightarrow \neg f_1$ expresses that no explanation is required for House 1 not burning; that is, House 1 is assumed not to burn unless an explanation for f_1 is given. Since both default rules $sf_1 \Rightarrow sf_1$ and $\neg sf_1 \Rightarrow \neg sf_1$ are included in Δ, the truth value of sf_1 can be chosen freely.

Bochman [9] extends the rules in a causal theory to an *explainability relation.*

Definition 2.12. *Let Δ be a causal theory. The binary relation $(\Rrightarrow_\Delta)/2$ of **explainability** is defined inductively from the causal rules as follows:*

*If $\lambda \Rightarrow l$, then $\lambda \Rrightarrow_\Delta l$. **(Causal rules)***
*If $\lambda \Rrightarrow_\Delta l$, then $\lambda \cup \lambda' \Rrightarrow_\Delta l$. **(Literal Monotonicity)***
*If $\lambda' \Rrightarrow_\Delta l$ and $\lambda \cup \{l\} \Rrightarrow_\Delta l'$, then $\lambda \cup \lambda' \Rrightarrow_\Delta l'$. **(Literal Cut)***
*$\{p, \neg p\} \Rrightarrow_\Delta l$ for all propositions p and literals l. **(Literal Contradiction)***

*If $\lambda \Rrightarrow_\Delta l$, it is said that λ **explains** l.*

Remark. *Bochman [9] initially allows causal rules of the form $\phi \Rightarrow \psi$ and explainability relations of the form $\phi \Rrightarrow \psi$, where ϕ and ψ are arbitrary formulas. He argues that explainability $(\Rrightarrow)/2$ satisfies all the properties of material implication "$\rightarrow$", except reflexivity (i.e., $\phi \rightarrow \phi$ does not necessarily hold). Given a causal theory Δ in the sense of Definition 2.11, this work restricts attention to explainability relations as characterized in Definition 2.12. Theorem 4.23 in Bochman [9] then provides the basis for this characterization.*

Example 2.11. *In Example 2.10, one finds that $f_1 \Rrightarrow_\Delta f_1$ and $f_2 \Rrightarrow_\Delta f_2$, even though the causal theory Δ does not explicitly assert that f_1 or f_2 are defaults.*

Bochman's semantics [9] for causal theories is grounded in Principles 2 and 3:

Definition 2.13 (Causal World Semantics)**.** *A **causal world** for a causal theory Δ is a world ω such that, for every literal l, the following formalization of Principles 2 and 3 hold:*

Formalization of Principle 2:
If $\omega \Rrightarrow_\Delta l$, then $l \in \omega$.

Formalization of Principle 3:
If $l \in \omega$, then $\omega \Rrightarrow_\Delta l$.

*The **causal world semantics** $\mathrm{Causal}(\Delta)$ is the set of all causal worlds of Δ.*

Bochman [9] gives the following alternative characterisation for the causal worlds ω of a causal theory Δ.

Definition 2.14 (Completion of Causal Theories)**.** *The* ***completion*** *of a causal theory Δ is the set of formulas* $\mathrm{comp}(\Delta) := \left\{ l \leftrightarrow \bigvee_{\phi \Rightarrow l \in \Delta} \phi \right\}_{l \text{ literal}}$.

Theorem 2.3 (Completion of Causal Theories [20, Theorem 8.115])**.** *The causal world semantics* $\mathrm{Causal}(\Delta)$ *of a causal theory Δ coincides with the set of all models of its completion:*

$$\mathrm{Causal}(\Delta) = \{\omega \textit{ world: } \omega \models \mathrm{comp}(\Delta)\}. \ \square$$

Example 2.12. *In Example 2.10, the theory Δ has the causal world $\omega := \{f_1, f_2\}$, which contradicts everyday causal reasoning as explained in Example 1.4. Note that ω is a causal world of Δ since the framework of causal theories allows for the cyclic explanations in Example 2.11.*

To avoid cyclic explanations as in Example 2.11. Eelink et al. [10] extend Bochman's causal theories [9] to accommodate a set of external premises $\mathscr{E}$ that do not require further explanation. Motivated by the ideas in Aristotles's *Posterior Analytics*, they additionally apply Principle 1. This leads them to the set-up of causal systems:

Definition 2.15 (Causal System)**.** *A* ***causal system CS*** $:= (\Delta, \mathscr{E}, \mathscr{O})$ *consists of a causal theory Δ called the* ***causal knowledge*** *of* ***CS****, a set of literals $\mathscr{E}$ called the* ***external premises*** *of* ***CS*** *and a set of formulas $\mathscr{O}$ called the* ***observations*** *of* ***CS****. The causal system* ***CS*** *is* ***without observations*** *if $\mathscr{O} = \emptyset$. Otherwise, the causal system* ***CS observes something****. The causal system* ***CS*** *applies* ***default negation*** *if every negative literal $\neg p$ for $p \in \mathfrak{P}$ is an external premise, i.e., $\neg p \in \mathscr{E}$ and no external premise is an effect of a causal rule in Δ. Further, the system* ***CS*** *is* ***atomic*** *if Δ is an atomic causal theory.*

The causal theory $\Delta(\mathbf{CS}) := \Delta \cup \{l \Rightarrow l \mid l \in \mathscr{E}\}$ is called the ***explanatory closure*** *of* **CS***. A* ***causally founded explanation*** *is an explanation $\lambda \Rrightarrow_{\Delta(\mathbf{CS})} l$ such that $\lambda \subseteq \mathscr{E}$.*

A ***causally founded world*** *ω is a world such that $\omega \models \mathscr{O}$ and for every literal l the following formalizations of Principles 2 and 3 are satisfied:*

> ***Formalization of Principle 2:***
> *If there exists a causally founded explanation $\omega \cap \mathscr{E} \Rrightarrow_{\Delta(\boldsymbol{CS})} l$, then $l \in \omega$.*
> ***Formalization of Principle 3:***
> *If $l \in \omega$, then there exists a causally founded explanation $\omega \cap \mathscr{E} \Rrightarrow_{\Delta(\boldsymbol{CS})} l$.*

Eelink et al. [10] argue that causally founded explanations can be used to formalize Principle 1.

Formalization 1 (Principle 1)**.** *All causal explanations $\lambda \Rrightarrow_{\Delta(\mathbf{CS})} l$ are causally founded.*

Example 2.13. *Example 1.4 gives rise to the causal system with default negation and without observations, defined as* ***CS*** $:= (\Delta, \mathscr{E}, \emptyset)$*, where $\Delta := \{f_1 \Rightarrow f_2, f_2 \Rightarrow f_1\}$ is an atomic causal theory and $\mathscr{E} := \{sf_i, \neg sf_i, \neg f_i\}_{i=1,2}$ the set of external premises. The explanatory closure $\Delta(\boldsymbol{CS})$ of* ***CS*** *coincides with the causal theory in Example 2.10. Note that the cyclic explanations from Example 2.11 are not causally founded. Hence, the world ω in Example 2.12 is not causally founded.*

3. Problem Statement

Example 1.4 shows that abductive logic programming under the causal model semantics can yield counterintuitive results in the presence of cyclic causal relationships. From the perspective of logic programming, such issues are typically addressed by applying the stable model semantics of Gelfond and Lifschitz [15]. However, it remains an open question whether this approach admits a causally meaningful interpretation that accounts for interventions.

4. Results

Throughout this section, we fix a propositional alphabet $\mathfrak{P}$. We begin by introducing the Bochman transformation, which identifies abductive logic programs with causal systems featuring default negation [10], as defined in Definition 2.15.

Informally, the Bochman transformation interprets clauses $h \leftarrow b_1 \wedge \ldots \wedge b_n$ as "$b_1 \wedge \ldots \wedge b_n$ *causes* h," treats the abducibles as external premises whose explanations lie beyond the given scope, and regards the integrity constraints as observations.

Definition 4.1 (Bochman Transformation)**.** *The* ***Bochman transformation*** *of an abductive logic program $\mathscr{P} := (\boldsymbol{P}, \mathfrak{A}, \boldsymbol{IC})$ is the causal system with default negation $\boldsymbol{CS}(\mathscr{P}) := (\Delta, \mathscr{E}, \mathcal{O})$, where $\Delta := \{\text{body}(C) \Rightarrow \text{head}(C) \mid C \in \boldsymbol{P}\}$, $\mathscr{E} := \mathfrak{A} \cup \{\neg p \mid p \in \mathfrak{P}\}$, and $\mathcal{O} := \boldsymbol{IC}$.*

Example 4.1. *Let $\mathscr{P}$ be the abductive logic program in Example 1.4. The causal system in Example 2.13 is the Bochman transformation $\boldsymbol{CS}(\mathscr{P})$ of $\mathscr{P}$.*

Let $\mathscr{P} := (\mathbf{P}, \mathfrak{A}, \mathbf{IC})$ be an abductive logic program. We show that the stable models of $\mathscr{P}$ correspond to the causally founded worlds of its Bochman transformation $\mathbf{CS}(\mathscr{P}) := (\Delta, \mathscr{E}, \mathcal{O})$. Together with Formalization 1 and Definition 2.15, this supports that the stable model semantics follows from Principles 1, 2, and 3.

We begin by relating unfounded sets to explainability in causal theories.

Definition 4.2 (Internal and External Explanations)**.** *Let Δ be an atomic causal theory, ω a model of the propositional theory obtained by reading the causal rules in Δ as logical implications, and $I \subseteq \omega$ a set of positive literals true in ω.*

*An I-**external explanation** is an expression of the form $\lambda \Rrightarrow_\Delta l$, where $l \in I$ and λ is a set of literals that are true in ω and do not belong to I. An I-**internal explanation** is an expression of the form $\lambda \Rrightarrow_\Delta l$ that is not I-external.*

Let ω be a causally founded world of the Bochman transformation $\mathbf{CS}(\mathscr{P})$. Definition 4.1 ensures that a subset $I \subseteq \omega$ of atoms true in ω can be unfounded with respect to any program extending the underlying logic program $\mathbf{P}$ only if every rule $\lambda \Rightarrow l$ in Δ corresponds to an I-internal explanation $\lambda \Rrightarrow_\Delta l$.

Lemma 4.1. *Let Δ, ω and I be as in Definition 4.2. If every rule $\lambda \Rightarrow l$ in Δ corresponds to an I-internal explanation $\lambda \Rrightarrow_\Delta l$, then every other explanation $\lambda' \Rrightarrow_\Delta l'$ is also I-internal as well.*

Proof. By Definition 2.12, $\lambda' \Rrightarrow_\Delta l'$ follows from the rules in Δ through iterated applications of literal cut, literal monotonicity and literal contradiction. So it suffices to show that as long as the rules in the premises to those three inference rules are I-internal, then so is their consequence.

We use the notation of Definition 2.12.

For literal monotonicity, the statement is clear, since if $\lambda \cup \lambda'$ is a set of literals true in ω and not in I, then so is its subset λ.

Since ω is a structure, q and $\neg q$ can never both be true in ω. Therefore, literals contradiction can never yield an I-external explanation.

So it only remains to consider literal cut. Assume for contradiction that $\lambda \cup \lambda' \Rrightarrow_\Delta l'$ is I-external despite both premises $\lambda' \Rrightarrow_\Delta l$ and $\lambda \cup \{l\} \Rrightarrow_\Delta l'$ being I-internal. Then $\lambda \cup \lambda'$ is a set of literals true in ω and not in I, and therefore so are λ and λ'. Thus, since both premises are I-internal, the atom l must not be in I and also not true in ω. However, this contradicts the fact that ω is a model of the propositional theory corresponding to the rules of Δ. Indeed, note that (literal) cut, monotonicity and contradiction are all true for propositional logic, where the triple arrow is read as logical implication. Thus, since $\lambda \Rrightarrow_\Delta l$ is obtained from rules in Δ using those three axioms, ω is a model of $\lambda \rightarrow l$ and of λ and thus of l, yielding the desired contradiction. □

With Lemma 4.1 at hand, we can now prove our first result.

Theorem 4.2 (Bochman Transformation)**.** *The Bochman transformation is a bijection from abductive logic programs to causal systems with default negation.*

An abductive logic program $\mathscr{P} := (\boldsymbol{P}, \mathfrak{A}, \boldsymbol{IC})$ *has a stable model* ω *if and only if* ω *is a causally founded world with respect to its Bochman transformation* $\boldsymbol{CS}(\mathscr{P}) := (\Delta, \mathscr{E}, \mathscr{O})$.

Proof. By construction the Bochman transformation is a bijection between abductive logic programs and causal systems with default negation. Since the integrity constraints are carried over unchanged by the Bochman transformation as observations, we can assume without loss of generality that $\mathscr{P}$is without integrity constraints.

First, we show that every stable model of $\mathbf{P} \cup (\omega \cap \mathfrak{A})$ is a causal founded world of $\mathbf{CS}(\mathscr{P}) =: (\Delta, \mathscr{E}, \emptyset)$.

Let ω be such a stable model. We need to show for all literals l that $\omega \cap \mathscr{E} \Rrightarrow_{\Delta(\mathbf{CS})} l$ if and only if $l \in \omega$. Note that since the rules in Δ correspond precisely to the clauses of the underlying logic program $\mathbf{P}$, the Clark completion of $\mathbf{P}$ coincides with the completion of the explanatory closure $\Delta(\mathbf{CS})$. Since every stable model is also a supported model, the structure ω is a model of the completion of $\Delta(\mathbf{CS}(\mathscr{P}))$.

For any set Λ of literals, we introduce the notation $\mathscr{C}(\Lambda)$ to indicate the set of all literals l such that $\Lambda \Rrightarrow_{\Delta(\mathbf{CS}(\mathscr{P}))} l$. Thus, Theorem 2.3 states that $\mathscr{C}(\omega) = \omega$ and by literal monotonicity $\mathscr{C}(\omega \cap \mathscr{E}) \subseteq \omega$. It remains to show that $\omega \subseteq \mathscr{C}(\omega \cap \mathscr{E})$, or, in other words, that $I := \omega \setminus \mathscr{C}(\omega \cap \mathscr{E}) = \emptyset$. Note first that since all negated literals are in $\mathscr{E}$, I is a set of positive literals. We show that if it were non-empty, I were an unfounded set. Assume that I is not unfounded. Then, there would be a $p \in I$ and a clause $p \leftarrow b_1, \ldots, b_n$ such that all of $b_1, \ldots, b_n$ are in $\omega \setminus I = \mathscr{C}(\omega \cap \mathscr{E})$. However, this implies that $\omega \cap \mathscr{E} \Rrightarrow_{\Delta(\mathbf{CS}(\mathscr{P}))} b_i$ for $i \in 1, \ldots, n$ and therefore by n-fold iteration of the (literal) cut, that $\omega \cap \mathscr{E} \Rrightarrow_{\Delta(\mathbf{CS}(\mathscr{P}))} p$ and thus that $p \in \mathscr{C}(\omega \cap \mathscr{E})$. This in turn contradicts $p \in I$, and thus concludes the proof that $I = \emptyset$ since stable models contain no non-empty unfounded sets of atoms. Overall, we have shown that $\mathscr{C}(\omega \cap \mathscr{E}) = \omega$ and therefore that ω is a causally founded world of $\mathbf{CS}(\mathscr{P})$.

We turn to the converse direction, showing that every causally founded world of $\mathbf{CS}(\mathscr{P})$ is a stable model of $\mathscr{P}$. By Theorem 2.3, every causally founded world of $\mathbf{CS}(\mathscr{P})$ is a model of the propositional theory corresponding to the clauses of $\mathbf{P}$.

So it remains to show that ω has no unfounded sets of atoms with respect to $\mathbf{P} \cup (\omega \cap \mathfrak{A})$. Assume it does have such a set, say I.

As ω is a causally founded world, we obtain $\omega \cap \mathscr{E} \Rrightarrow_{\Delta(\mathbf{CS}(\mathscr{P}))} p$ for any $p \in I$. Since every abducible atom true in ω corresponds to a fact of $\mathbf{P} \cup (\omega \cap \mathfrak{A})$, the unfounded set I must be disjoint from $\mathfrak{A}$ and thus from $\mathscr{E}$. Therefore, $\omega \cap \mathscr{E} \Rrightarrow_{\Delta(\mathbf{CS}(\mathscr{P}))} p$ is I-external.

By Lemma 4.1, this implies that one of the rules of $\Delta(\mathbf{CS}(\mathscr{P}))$ is I-external and thus that I is not an unfounded subset of ω.

This concludes the proof that ω is a stable model of $\mathbf{P} \cup (\omega \cap \mathfrak{A})$ as claimed. □

According to Theorem 4.2, Principles 1, 2, and 3, together with Formalization 1 and Definition 2.15, entail that a causal interpretation of abductive logic programming necessarily yields the stable model semantics of Gelfond and Lifschitz [15]. This raises the question of whether every abductive logic program admits such a causal interpretation.

Example 4.2. *Let e denote the event that a farmer is ecological, and h the event that it is hot. Further, let s denote that pests survive the weather, p that there are pests in the field, and t that the farmer applies toxin.*

Assume pests survive if it is hot. If no toxin is applied, pests remain; toxin is applied if pests are present and the farmer is not ecological. These relations define an abductive logic program $\mathscr{P}$ *with abducibles* $\mathfrak{A} := \{h, e\}$ *and rules* $\boldsymbol{P}$*:*

$$t \leftarrow p, \neg e, \qquad p \leftarrow \neg t, s, \qquad s \leftarrow h.$$

The program $\mathscr{P}$ *has no stable model with explanation* $\epsilon := \{h\}$*. Hence, it concludes against the causal direction, namely that it cannot be hot if the farmer is not ecological. This contradicts everyday intuition as well as Principle 4.*

We argue that to represent causal knowledge, an abductive logic program must also satisfy Principle 4, as explored by Williamson [13] in the context of Bayesian networks. In abductive logic programming, we interpret Principle 4 as the following semantic constraint:

Formalization 2 (Principle 4)**.** *Let $\mathscr{P} := (\boldsymbol{P}, \mathfrak{A}, \boldsymbol{IC})$ be a consistent abductive logic program. For a set $S \subseteq \mathfrak{P}$ of propositions, let $\mathfrak{P}^{>S}$ denote the set of all propositions $q \notin S$ that are descendants in the dependency graph $G := \mathrm{graph}(\boldsymbol{P})$ of some proposition in S. Finally, denote by $\boldsymbol{P}^{>S}$ the program consisting of all clauses $C \in \boldsymbol{P}$ with $\mathrm{head}(C) \in \mathfrak{P}^{>S}$.*

Then, $\mathscr{P}$ satisfies Principle 4 *if and only if for every set $S \subseteq \mathfrak{P}$ and every $\mathfrak{P} \setminus \mathfrak{P}^{>S}$-structure ω, the program $\boldsymbol{P}^{>S,\omega} := \boldsymbol{P}^{>S} \cup \omega$ has at least one stable model; that is, it is not possible to falsify ω with $\boldsymbol{P}^{>S}$.*

Remark. *Williamson [13] proposes Principle 4 in the context of maximum entropy as a weakening of the Markov assumption in Bayesian networks [1]. Accordingly, the above formalization could be viewed as a deterministic analogue of the Markov assumption.*

Example 4.3. *The program $\mathscr{P}$ in Example 4.2 does not satisfy Principle 4.*

We argue that every abductive logic program $\mathscr{P}$ satisfying Principle 4 under the above formalization admits a causal interpretation under the stable model semantics. This raises the question of whether $\mathscr{P}$ can also be used to predict the effects of external interventions.

According to Pearl [1], the joint act of observing and intervening leads to counterfactual reasoning, which lies beyond the scope of this contribution. Therefore, we restrict our interest to programs without integrity constraints and argue that they admit a meaningful representation of interventions if Principle 5 holds. Finally, the following result shows that Principle 4 implies Principle 5.

Theorem 4.3. *Let $\mathscr{P} := (\boldsymbol{P}, \mathfrak{A}, \emptyset)$ be an abductive logic program without integrity constraints that satisfies Principle 4, and let $\mathbf{i}$ be an assigment on a set of propositions $S \subseteq \mathfrak{P}$.*

Define $\mathfrak{P}^{<S} := \mathfrak{P} \setminus (\mathfrak{P}^{>S} \cup S)$, $\boldsymbol{P}^{<S} := \{C \in \boldsymbol{P} \mid \mathrm{head}(C) \in \mathfrak{P}^{<S}\}$ and $\mathscr{P}^{<S} := (\boldsymbol{P}^{<S}, \mathfrak{A}, \emptyset)$. Then, the following are equivalent:

1. *$\omega^{<S} = \omega \cap \mathfrak{P}^{<S}$ for some stable model ω of $\mathscr{P}$, i.e., $\omega^{<S}$ is a reduct of ω.*
2. *$\omega^{<S}$ is a stable model of $\mathscr{P}^{<S}$.*
3. *$\omega^{<S} = \omega_{\mathbf{i}} \cap \mathfrak{P}^{<S}$ for some stable model $\omega_{\mathbf{i}}$ of $\mathscr{P}_{\mathbf{i}}$, i.e., $\omega^{<S}$ is a reduct of $\omega_{\mathbf{i}}$.*

Proof. The proof rests on the splitting lemma [21]. For any set of propositions $S \subseteq \mathfrak{P}$ denote by $\mathfrak{P}^{\geq S} := \mathfrak{P}^{>S} \cup S$. Let $\mathscr{P}^S$ be the set of all clauses with heads in S, and write $\mathbf{P}^{*S}$ for the set of all clauses with heads in $\mathfrak{P}^{*S}$, where $* \in \{<, \geq, >\}$. Finally, set $\mathscr{P}^{*S} := (\mathbf{P}^{*S}, \mathfrak{A}, \emptyset)$.

Let ω be a world. By abuse of notation, for an abductive logic program $\mathscr{Q} := (Q, \mathfrak{B}, \emptyset)$ we denote the logic program $Q \cup (\omega \cap \mathfrak{B})$ also by $\mathscr{Q}$.

We first show the equivalence of 2 and 3, that stable models of $\mathscr{P}^{<S}$ are precisely the reducts of stable models of $\mathscr{P}_{\mathbf{i}}$. Note first that $\mathscr{P}^{<S} = \mathscr{P}_{\mathbf{i}}^{<S}$. Now consider the splitting $(\mathscr{P}_{\mathbf{i}}^{<S}, \mathscr{P}_{\mathbf{i}}^{S}, \mathscr{P}_{\mathbf{i}}^{>S})$. This is indeed a splitting, since after intervention the propositions in S have no ancestors at all. Therefore, every reduct of a stable model of $\mathscr{P}_{\mathbf{i}}$ to $\mathfrak{P}^{<S}$ is a stable model of $\mathscr{P}_{\mathbf{i}}^{<S} = \mathscr{P}^{<S}$. For the other direction, let ω be a stable model of $\mathscr{P}^{<S} = \mathscr{P}_{\mathbf{i}}^{<S}$. Since $\mathscr{P}_{\mathbf{i}}^{>S,\omega}$ consists only of facts, it clearly has a stable model, say ω_S. By assumption, the program $\mathscr{P}^{>S,\omega_S} = \mathscr{P}_{\mathbf{i}}^{>S,\omega_S}$ has a stable model, which by the splitting lemma is also a stable model of $\mathscr{P}_{\mathbf{i}}$ and extends ω.

Now we turn to the equivalence of 1 and 2. Note first that for any S, $(\mathscr{P}^{<S}, \mathscr{P}^{\geq S})$ is a splitting of $\mathscr{P}$. Therefore, every reduct of a stable model of $\mathscr{P}$ to $\mathfrak{P}^{<S}$ is a stable model of $\mathscr{P}^{<S}$.

So let ω be such a stable model of $\mathscr{P}^{<S}$. We need to show that ω can be extended to a stable model of $\mathscr{P}$. Denote $\mathfrak{P}^{<S}$ by S'. We note that S' is closed under predecessors since $\mathfrak{P}^{\geq S}$ is clearly closed under successors. We employ the splitting $(\mathscr{P}^{S'}, \mathscr{P} \setminus (\mathscr{P}^{S'} \cup \mathscr{P}^{>S'}), \mathscr{P}^{>S'})$. Note that the vocabularies used in $\mathscr{P}^{S'}$ and in $\mathscr{P} \setminus (\mathscr{P}^{S'} \cup \mathscr{P}^{>S'})$ are disjoint, since if a head occurring in $\mathscr{P} \setminus (\mathscr{P}^{S'} \cup \mathscr{P}^{>S'})$ would be a successor of a proposition in S', it would lie in $\mathfrak{P}^{>S'}$, and S' is closed under predecessors. Since if the vocabularies of two logic programs are disjoint, the stable models of their union are precisely the unions

of their stable models and $\mathscr{P}$ is consistent (therefore $\mathscr{P} \setminus (\mathscr{P}^{S'} \cup \mathscr{P}^{>S'})$ has at least one stable model), every stable model of $\mathscr{P}^{S'}$ extends to a stable model of $\mathscr{P} \setminus \mathscr{P}^{>S'}$. The result now follows immediately from Formalization 2. □

Finally, we obtain that stratified abductive logic programs under the stable model semantics satisfy Principle 4, as formulated in Formalization 2.

Theorem 4.4. *Every stratified abductive logic program satisfies Principle 4.*

Proof. If $\mathscr{P} := (\mathbf{P}, \mathfrak{A}, \mathbf{IC})$ is stratified, then so is $\mathbf{P}^{>S,\omega}$ for every $\mathbf{S} \subseteq \mathfrak{P}$ and every S-structure ω. Therefore, the program $\mathbf{P}^{>S,\omega}$ has (precisely) one stable model. □ □

5. Conclusion and Related work

Let $\mathscr{P} := (\mathbf{P}, \mathfrak{A}, \mathbf{IC})$ be an abductive logic program. This work proposes a causal interpretation of $\mathscr{P}$ that reads each clause $h \leftarrow b_1, \ldots, b_n \in \mathbf{P}$ as "$b_1 \wedge \cdots \wedge b_n$ *causes* h," interprets the abducibles $\mathfrak{A}$ as external premises with explanations beyond a given scope, and treats the integrity constraints **IC** as observations. Theorem 4.2 then shows that Principles 1–3, as formulated in Formalization 1 and Definition 2.15, entail the stable model semantics [15].

However, Example 4.2 illustrates that general programs may violate Principle 4 and, therefore, do not admit a causal interpretation.

Fortunately, Theorem 4.4 confirms that stratified programs respect this principle, supporting their causal interpretability under the stable model semantics.

In the absence of observations ($\mathbf{IC} = \emptyset$), Theorem 4.3 shows that Principle 4 implies 5. Hence, stratified programs without integrity constraints support reliable predictions under interventions.

Several authors have explored the relation between causal logic and logic programming. To our knowledge, the earliest results appeared in the context of causally enriched versions of the situation calculus [22, 23, 24]. McCain [25] and Lin and Wang [26] translate causal constraints of such languages into disjunctive logic programs with classical negation. McCain's transformation [25] is extended to a broader class of causal theories, including first-order ones, by Ferraris et al. [27]. This line of work relies on classical negation, thus departing from the standard framework of negation as failure. Its aim is to make causal theories executable, rather than to provide a causal semantics for logic programs.

Conversely, some authors have investigated how logic programs themselves might admit a causal interpretation. This includes work by Giunchiglia et al. [28] and Bochman [29], who, under the stable model semantics, translate a logic programming clause of the form $c \leftarrow \vec{a}, \neg\vec{b}$ into the causal rule $\bigwedge \neg\vec{b} \Rightarrow \left(\bigwedge \vec{a} \rightarrow c\right)$. Compared to the Bochman transformation in Definition 4.1, which yields the rule $\vec{a}, \neg\vec{b} \Rightarrow c$, this formulation is more difficult to interpret. In particular, as noted by Eelink et al. [10, §2.2.2], the use of embedded logical implication within a causal reasoning framework is far less transparent than the use of ordinary propositions. This formulation also implies that logic programs correspond only to causal rules of a highly specific syntactic form. From this perspective, logic programs could not express causal dependencies between positive atoms, which would severely limit their causal expressiveness. Moreover, to our knowledge, these works do not address the feasibility of modeling interventions, nor do they consider Principles 1, 4 and 5.

The Bochman transformation in Definition 4.1 maps a fact h to a rule of the form $\top \Rightarrow h$, whose interpretation remains open [10, Remark 1.1]. Future work should investigate suitable causal readings of such rules, identify the class of programs satisfying Principle 4, and extend the framework to disjunctive programs. While this work addresses whether a program can meaningfully represent the effect of all interventions, it remains to be explored when a program contains sufficient information to represent particular interventions, and whether Principle 5, appropriately formalized, is in fact equivalent to Principle 4.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT in order to: Grammar and spelling check, Paraphrase and reword. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] J. Pearl, Causality, Cambridge University Press, 2000. URL: https://doi.org/10.1017/CBO9780511803161.

[2] S. Arif, M. A. MacNeil, Applying the structural causal model framework for observational causal inference in ecology, Ecological Monographs 93 (2023) e1554. URL: https://doi.org/10.1002/ecm.1554.

[3] C. Gao, Y. Zheng, W. Wang, F. Feng, X. He, Y. Li, Causal inference in recommender systems: A survey and future directions, ACM Transactions on Information Systems 42 (2024) 88:1–88:32. URL: https://doi.org/10.1145/3639048.

[4] X. Wu, S. Peng, J. Li, J. Zhang, Q. Sun, W. Li, Q. Qian, Y. Liu, Y. Guo, Causal inference in the medical domain: a survey, Applied Intelligence 54 (2024) 4911–4934. URL: https://doi.org/10.1007/s10489-024-05338-9.

[5] V. Miller, R. Miller, René Descartes: Principles of Philosophy, Springer Dordrecht, 1982. URL: https://doi.org/10.1007/978-94-009-7888-1.

[6] A. C. Kakas, P. Mancarella, Generalized stable models: A semantics for abduction, in: 9th European Conference on Artificial Intelligence, ECAI 1990, 1990, pp. 385–391. URL: https://www.academia.edu/50323356/Generalized_stable_models_a_semantics_for_abduction.

[7] K. Rückschloß, F. Weitkämper, Exploiting the full power of Pearl's causality in probabilistic logic programming, in: Proceedings of the International Conference on Logic Programming 2022 Workshops, volume 3193 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022. URL: https://ceur-ws.org/Vol-3193/paper1PLP.pdf.

[8] K. L. Clark, Negation as failure, in: Logic and Data Bases, Springer US, 1978, pp. 293–322. URL: https://dl.acm.org/doi/book/10.5555/578615.

[9] A. Bochman, A Logical Theory of Causality, The MIT Press, 2021. URL: https://doi.org/10.7551/mitpress/12387.001.0001.

[10] G. Eelink, K. Rückschloß, F. Weitkämper, How artificial intelligence leads to knowledge why: An inquiry inspired by Aristotle's *Posterior Analytics*, 2025. URL: https://arxiv.org/abs/2504.02430. arXiv:2504.02430.

[11] J. F. Anderson, Summa Contra Gentiles, 2: Book Two: Creation, University of Notre Dame Press, 1956. URL: https://doi.org/10.2307/j.ctvpj74rh.

[12] G. W. Leibniz, First truths, in: Philosophical Papers and Letters, Springer Netherlands, 1989, pp. 267–271. URL: https://doi.org/10.1007/978-94-010-1426-7_31.

[13] J. Williamson, Foundations for Bayesian Networks, Springer Netherlands, 2001, pp. 75–115. URL: https://doi.org/10.1007/978-94-017-1586-7_4.

[14] J. Barnes, The Complete Works of Aristotle. Volume One., Princeton University Press, 1995. URL: https://doi.org/10.2307/j.ctt5vjv4w.

[15] M. Gelfond, V. Lifschitz, The stable model semantics for logic programming, in: Proceedings of International Logic Programming Conference and Symposium, MIT Press, 1988, pp. 1070–1080. URL: http://www.cs.utexas.edu/users/ai-lab?gel88.

[16] D. Saccà, C. Zaniolo, Stable models and non-determinism in logic programs with negation, in: Proceedings of the Ninth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, ACM Press, 1990, pp. 205–217. URL: https://doi.org/10.1145/298514.298572.

[17] V. Lifschitz, Thirteen definitions of a stable model, in: Fields of Logic and Computation, Springer, 2010, pp. 488–503. URL: https://doi.org/10.1007/978-3-642-15025-8_24.

[18] M. Denecker, A. C. Kakas, Abduction in logic programming, in: Computational Logic: Logic Programming and Beyond, Springer, 2002, pp. 402–436. URL: https://doi.org/10.1007/3-540-45628-7_16.
[19] J. Chomicki, G. Saake (Eds.), Logics for databases and information systems, Kluwer Academic Publishers, USA, 1998. URL: https://dl.acm.org/doi/10.5555/294135.
[20] A. Bochman, Explanatory Nonmonotonic Reasoning, World Scientific, 2005. URL: https://doi.org/10.1142/5707.
[21] V. Lifschitz, H. Turner, Splitting a logic program, in: Logic Programming, Proceedings of the Eleventh International Conference on Logic Programming, Santa Marherita Ligure, Italy, June 13-18, 1994, MIT Press, 1994, pp. 23–37. URL: https://www.cs.utexas.edu/~ai-lab/?lif94e.
[22] F. Lin, Embracing causality in specifying the indirect effects of actions, in: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes, Morgan Kaufmann, 1995, pp. 1985–1993. URL: http://ijcai.org/Proceedings/95-2/Papers/123.pdf.
[23] N. McCain, H. Turner, Causal theories of action and change, in: Proceedings of the Fourteenth National Conference on Artificial Intelligence AAAI 97, AAAI Press / The MIT Press, 1997, pp. 460–465. URL: http://www.aaai.org/Library/AAAI/1997/aaai97-071.php.
[24] E. Giunchiglia, V. Lifschitz, An action language based on causal explanation: Preliminary report, in: Proceedings of the Fifteenth National Conference on Artificial Intelligence AAAI 98, AAAI Press / The MIT Press, 1998, pp. 623–630. URL: http://www.aaai.org/Library/AAAI/1998/aaai98-088.php.
[25] N. McCain, Causality in commonsense reasoning about actions, Ph.D. thesis, University of Texas at Austin, 1997.
[26] F. Lin, K. Wang, From causal theories to logic programs (sometimes), in: Logic Programming and Nonmonotonic Reasoning, 5th International Conference, Springer, 1999, pp. 117–131. URL: https://doi.org/10.1007/3-540-46767-X_9.
[27] P. Ferraris, J. Lee, Y. Lierler, V. Lifschitz, F. Yang, Representing first-order causal theories by logic programs, Theory and Practice of Logic Programming 12 (2012) 383–412. URL: https://doi.org/10.1017/S1471068411000081.
[28] E. Giunchiglia, J. Lee, V. Lifschitz, N. McCain, H. Turner, Nonmonotonic causal theories, Artificial Intelligence 153 (2004) 49–104. URL: https://doi.org/10.1016/j.artint.2002.12.001.
[29] A. Bochman, A causal logic of logic programming, in: Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference, AAAI Press, 2004, pp. 427–437. URL: http://www.aaai.org/Library/KR/2004/kr04-045.php.