

Where Are the Explanations in Causal AI? Integrating Pearl’s Definitory Account of Causality with Causal Explanations

Kilian Rückschloß¹[0000–1111–2222–3333] and
Felix Weitkämper²[0000–0002–7891–6030]

¹ University of Tübingen, Auf der Morgenstelle 10, 72076 Tübingen,
`kilian.rueckschloss@uni-tuebingen.de`

² German University of Digital Science, Marlene-Dietrich-Allee 14, 14482 Potsdam
`felix.weitkaemper@german-uds.de`

Abstract. In artificial intelligence, Pearl’s influential work conceives causality as a feature allowing for reasoning about sophisticated queries such as the consequences of actions, while philosophical accounts primarily understand causality as providing explanations for why things happen. This contribution proposes the framework of *definitory causal models* to merge these two perspectives. Here, Datalog programs represent Boolean causal relationships that can be composed into causal explanations. *Definitory causal models* arise by augmenting Datalog with Pearl’s structural equations. Within this framework, the notions of *explanation* and *intervention* are established, helping to close the gap between the explanatory account of causality in philosophy and the query-based account in artificial intelligence. Finally, the expressivity of definitory causal models is studied, and an implementation is provided.

Keywords: Causal models · Datalog · Interventions · Explanations

1 Introduction

Having long been a central topic of philosophical inquiry, causality entered the mainstream of artificial intelligence research only recently through the work of Pearl [16]. He argues that causal knowledge goes beyond descriptive knowledge in the kinds of questions it can address. While descriptive knowledge permits only inferences from observations, causal knowledge additionally enables the prediction of intervention effects, such as the consequences of actions applied to the modelled system and counterfactuals.

Example 1. Consider a road that passes through a field with a sprinkler, which is activated by a weather sensor whenever it is sunny.

Observing that the sprinkler is on, one concludes that the weather is sunny. However, actively intervening and manually switching the sprinkler on does not cause the weather to become sunny. According to Pearl [16], predicting the effects of such interventions requires causal, rather than merely descriptive, knowledge.

Since evaluating intervention effects is a primary motivation for modelling, causal frameworks are employed in a wide variety of domains [17,8].

While causality in artificial intelligence is primarily studied through queries about interventions and counterfactuals, in philosophy it is more commonly examined in terms of explaining why things happen. Likewise, Carloni et al. [4] identify the lack of explanations in causal artificial intelligence as a challenge for explainable AI.

Example 2. In Example 1, concluding from sunny weather to the sprinkler being activated follows the direction of cause and effect. Such reasoning therefore yields causal knowledge about why the sprinkler is on.

By contrast, concluding from the sprinkler being on to its cause, namely sunny weather, runs against the causal direction and therefore does not yield causal knowledge about why it is sunny.

This work extends the prior results of Eelink et al. [7] and Rückschloß and Weitkämper [22], which relate reasoning about interventions as developed by Pearl [16] (see Example 1) to causal explanations in philosophy (see Example 2). From a logical perspective, however, all of the aforementioned works are purely propositional. The present paper continues this line of research by lifting the notions of causal explanation of Eelink et al. [7] and intervention of Pearl [16] to a relational setting through the formalism of *definitory causal models*. It also analyses the expressivity of these models and provides an implementation.

Example 3. Inspired by conservation biology, consider a dataset describing plots of land together with the abundances of trees and birds. One wishes to reason about connected pieces of forest. Furthermore, one wishes to express that the number of birds in any plot is directly proportional to the product of the number of trees and the size of the contiguous forest to which the plot belongs. Notably, this level of abstraction cannot be captured in the propositional theories of Pearl [16] and Eelink et al. [7].

Denote by $\text{trees}(X)$ and $\text{birds}(X)$ the abundances of trees and birds in a plot X , respectively. Furthermore, $\text{adjacent}(X, Y)$ indicates that plots are adjacent, while $\text{barrier}(X, Y)$ indicates that there is a barrier between them.

Through the expressive power of logic programs, that is, *causal rules*, definitory causal models allow reasoning about connected pieces of forest as follows:

$$\text{connected}(X, X) \Leftarrow \text{trees}(X) > 0. \quad (1)$$

$$\begin{aligned} \text{connected}(X, Y) \Leftarrow & \text{connected}(X, Z), \text{adjacent}(Z, Y), \\ & \neg \text{barrier}(Z, Y), \text{trees}(Y) > 0. \end{aligned} \quad (2)$$

In *relational structural equations*, they furthermore allow one to express that the number of birds in any plot X is directly proportional, say with factor five, to the product of the number of trees in X and the size of the contiguous forest to which X belongs:

$$\text{birds}(X) := 5 \times \left(\text{trees}(X) \times \text{sum}_Y(\text{size}(Y) \mid \text{connected}(X, Y)) \right). \quad (3)$$

Note that none of these definitions are possible under either the causal interpretation of logic programs in Rückschloß and Weitekämper [22] or the causal models of Pearl [16].

As desired, definitory causal models turn out to be expressive enough to capture the theory of Eelink et al. [7]. Since their semantics is based on causal explanations, the present work bridges the gap between Pearl’s query-based approach to causality in artificial intelligence [16] and the explanation-based approach found in philosophy and explainable AI. A SWI-Prolog implementation using tabled evaluation and constraint logic programming can be found at https://codeberg.org/felix_weitkaemper/definitory_causal_models.

Related Work

Eelink et al. [7] show in Theorem 2.6 that, in the Boolean case, the causal models of Pearl [16] translate into causally interpreted logic programs. In Proposition 2.8, they show that the corresponding notion of intervention coincides with the one given by Pearl [16]. Thus, in the propositional Boolean setting, where parameters take only the values `true` and `false`, they close the gap between query-based and explanation-based accounts of causality under the assumption of default negation.

The present work extends this line of research by lifting causally interpreted logic programs to a relational setting. In doing so, it also encompasses Pearl’s non-Boolean causal models with interventions [16], where default negation is not applicable. In this sense, the present work closes the gap between query-based and explanation-based approaches to causality.

Observe that the causal models of Pearl [16] and the causally interpreted logic programs of Rückschloß and Weitekämper [22] share a common feature: both specify causal knowledge by *defining* elementary causal relations through structural equations or causal rules. In contrast to the notions of causality provided by Hume [12], Granger [10] and Vennekens et al. [23], neither formalism appeals to temporal notions. The present work argues that this yields a *definitory* account of causality and accordingly introduces the resulting generalisation as the formalism of *definitory causal models*.

To the best of our knowledge, Lifschitz [14] was the first to propose a first-order logical theory of causality. Like Bochman [1,2], he employs a different semantics for causal rules from that used by Eelink et al. [7], namely a semantics that allows cyclic causal explanations. Later, Lifschitz and Yang [15] extended this theory by incorporating explanations for equations $X = f$, where X is a variable and f is a function symbol. Finally, Bochman and Lifschitz [3] embedded the causal models of Pearl [16] into this theory.

Notably, the resulting representation of causal knowledge is restricted to finite domains only. It is also not modular; that is, it does not permit combining a causal model with additional logical causal relationships without altering the solutions of the original model. Furthermore, these approaches do not permit variables in structural equations, which therefore remain propositional.

In contrast, the definitory causal models in this work explicitly allow reasoning over infinite domains. As demonstrated in Equation (3), they support variables in structural equations. Finally, they are modular and explicitly combine explanation-based causal reasoning in logic with Pearl’s causal models [16].

2 Preliminaries

Pearl [16] suggests modelling causal relationships with deterministic functions. This leads to the following notion of structural causal models.

Definition 1 (Causal Model [16, §7.1.1]). A (*structural*) *causal model* $\mathcal{M}$ with *internal variables* $\mathbf{V}$ and *external variables* $\mathbf{U}$ is a system of equations that includes one *structural equation* of the form $X := f_X(\text{Pa}(X), \text{Error}(X))$ for each internal variable $X \in \mathbf{V}$. Here, the *parents* $\text{Pa}(X) \subseteq \mathbf{V}$ of X are a subset of internal variables, the *error term* $\text{Error}(X) \subseteq \mathbf{U}$ is a subset of external variables, and the *causal mechanism* of X is a function f_X that maps each assignment of values to $\text{Pa}(X)$ and $\text{Error}(X)$ to a value of X .

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

Remark 1. The parents $\text{Pa}(V)$ and error terms $\text{Error}(V)$ of an internal variable $V \in \mathbf{V}$ are typically evident from the causal mechanism f_V . Accordingly, this work omits explicit references to $\text{Pa}(\cdot)$ and $\text{Error}(\cdot)$.

Causal models are particularly valuable because they can represent external interventions. As explained in Chapter 7 of Pearl [16], 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 (Modified Causal Model). Fix a causal model $\mathcal{M}$. Let $\mathbf{I}$ be a subset of internal variables with a value assignment $\mathbf{i}$. The *modified model* or *submodel* $\mathcal{M}_{\mathbf{i}}$ is the model obtained from $\mathcal{M}$ by replacing, for each internal variable $X \in \mathbf{I}$, the structural equation $X := f_X(\text{Pa}(X), \text{Error}(X))$ with $X := \mathbf{i}(X)$. If $V \in \mathbf{V}$ is a Boolean internal variable of a structural causal model $\mathcal{M}$, one writes $\mathcal{M}_V := \mathcal{M}_{V:=\top}$ and $\mathcal{M}_{\neg V} := \mathcal{M}_{V:=\perp}$.

Remark 2. As in Example 2, actions often force a variable in a causal model to take on a new value. Pearl [16] emphasises that submodels $\mathcal{M}_{\mathbf{i}}$ typically arise from performing actions that set certain variables to specific values, a process formalised by the introduction of the *do*-operator.

Building on Eelink et al. [7], Rückschloß and Weitkämper [22] employ logic programs as a language for causal relationships, which can then be composed into causal explanations.

Fix a propositional alphabet $\mathfrak{P}$, i.e., a set of propositions. A **literal** is either an **atom** $p \in \mathfrak{P}$ or its negation $\neg p$. Logic programs consist of rules or clauses.

Definition 3 (Clauses and Logic Programs). A (*normal*) *clause* C is an expression of the form $(h \Leftarrow (b_1 \wedge (b_2 \wedge (\dots \wedge b_n))))$, also denoted as $h \Leftarrow b_1 \wedge \dots \wedge b_n$, $h \Leftarrow b_1, \dots, b_n$ or $\text{head}(C) \Leftarrow \text{body}(C)$.

Here, $\text{head}(C) := h$ is an atom, referred to as the **head** of the clause C and $\text{body}(C) := \{b_1, \dots, b_n\}$ is a finite set of literals, known as the **body** of C . If $\text{body}(C) = \emptyset$ and $C = (h \Leftarrow \top)$, one denotes C by h and calls C a **fact**.

Finally, a **logic program** P is a finite set of clauses.

Abductive logic programming has the goal of providing an explanation for a given set of observations in terms of so-called abducibles. Alternatively, from a view closer to database theory, they can also be seen as logic programs with an extensional alphabet.

This work thus identifies in the signature of each logic program P a set of *extensional propositions* $\mathcal{E}$ that do not occur as the head of any clause in P .

Rückschloß and Weitekämper [22] give a causal interpretation to such logic programmes. In this view, a clause $h \Leftarrow b_1 \wedge \dots \wedge b_n$ is interpreted as “ $b_1 \wedge \dots \wedge b_n$ causes h ”. Following Eelink et al. [7], these elementary causal relations are then composed into a relation of causal explainability $(\Rightarrow)/2$. Assuming that external propositions in $\mathcal{E}$ do not require explanations, Eelink et al. [7] then derive the semantics of causally founded worlds from the principles that all explained events take place and that every event that takes place can be explained.

Definition 4 (Explainability, Causally Founded Worlds). Let P be a logic program. For sets of literals λ, λ' and literals l, l' , the binary relation $(\Rightarrow_P)/2$ of **explainability** is defined inductively from the rules in P as follows:

Causal Rules: If $\lambda \Rightarrow l \in P$, then $\lambda \Rightarrow_P l$.

Monotonicity: If $\lambda \Rightarrow_P l$, then $\lambda \cup \lambda' \Rightarrow_P l$.

Cut: If $\lambda' \Rightarrow_P l$ and $\lambda \cup \{l\} \Rightarrow_P l'$, then $\lambda \cup \lambda' \Rightarrow_P l'$.

Contradiction: $\{p, \neg p\} \Rightarrow_P l$ for $p \in \mathfrak{P}$ and literals l .

If $\lambda \Rightarrow_P l$, then λ is said to **explain** l .

A **causally founded world** ω of a logic program P with extensional signature $\mathcal{E}$ then is a world satisfying $l \in \omega$ if and only if $\omega \cap \mathcal{E} \Rightarrow_P l$ for every literal l .

In Theorem 4.2, Rückschloß and Weitekämper [22] prove that the causally founded worlds of a logic program P coincide with the stable models of Gelfond and Lifschitz [9]. They further argue that logic programs correctly predict intervention effects whenever these propagate only along the causal direction, which they verify for stratified programs. In this case, the following notion of intervention consistently generalises Pearl’s notion [16], as obtained through Clark completion [5] by Rückschloß and Weitekämper [21].

Definition 5 (Modified Program). Let P be a logic program with external signature $\mathcal{E}$. To represent the intervention of forcing the atoms in $I \subseteq \mathfrak{P} \setminus \mathcal{E}$ to attain values according to the assignment i , the **modified program** P_i is obtained from P by the modifications below:

1. Remove all clauses $C \in P$ with $\text{head}(C) \in I$.
2. Add a fact p to P_i if $p \in i$.

Building on Rückschloß and Weitkämper [22], this contribution integrates general non-Boolean causal models from Pearl [16] into a framework of causally interpreted relational logic programs. To this end, it adopts the framework of *metafinite model theory* introduced by Grädel and Gurevich [11], which extends the finite domain sort on which logic programs usually operate by an additional arithmetic or secondary sort, used to evaluate Pearl’s structural equations.

Definition 6 (Metafinite Signature). A *metafinite signature* is a triplet $\Sigma := (\sigma_A, \sigma_R, \sigma_W)$. Here, σ_A is a relational signature, which may include constant symbols. The set σ_W contains **attribute symbols**, that is, function symbols of positive arity. Finally, σ_R consists of equality ($=$)/2, the relation and function symbols (including constants) intended for the arithmetic or secondary sort, as well as symbols for **aggregation functions**, which do not have a fixed arity.

Example 4. In Example 3, the signature is as follows: σ_A consists of the binary predicates `adjacent/2`, `barrier/2` and `connected/2`; σ_R consists of binary functions $+$ (addition) and $\times$ (multiplication), the binary relation $(>)/2$, equality $(=)/2$ as well as constants 0 and 1 and aggregation function `sum`; σ_W consists of three unary attribute symbols, `size/1`, `trees/1` and `birds/1`.

Definition 7 (Secondary and Primary Terms). Let $\Sigma := (\sigma_A, \sigma_R, \sigma_W)$ be a metafinite signature. **Secondary** and **primary terms** are defined as follows:

1. Variables are primary terms.
2. Constants in σ_A are primary terms.
3. Constants in σ_R are secondary terms.
4. If $F_1, \dots, F_k$ are secondary terms and g is a k -ary function symbol of σ_R , then $g(F_1, \dots, F_k)$ is a secondary term.
5. If $t_1, \dots, t_k$ are primary terms and $w \in \sigma_W$ is a k -ary attribute symbol, then $w(t_1, \dots, t_k)$ is a secondary term.
6. If F is a secondary term, Γ is an aggregation function symbol in σ_R and R is a relation symbol of σ_A , then $\Gamma_{\mathbf{x}}(F \mid R(\mathbf{t}))$ is a secondary term, where $\mathbf{t}$ is a tuple of primary terms corresponding to the arity of R and $\mathbf{x}$ is a tuple of variables occurring in $\mathbf{t}$.

Variables occurring in a term are considered **free**, apart from variables that only occur in a subterm of the form $\Gamma_{\mathbf{x}}(F \mid R(\mathbf{t}))$ where they are not among the $\mathbf{x}$.

A **primary atom** is of the form $P(\mathbf{t})$, where $\mathbf{t}$ is a list of primary terms and $P \in \sigma_A$ a relation symbol of the right arity. Analogously, a **secondary atom** is of the form $P(\mathbf{t})$, where $\mathbf{t}$ is a list of secondary terms and $P \in \sigma_R$ a relation symbol of the right arity. An **atom** $P(\mathbf{t})$ is either a primary or a secondary atom.

A **literal** is either an atom or a negated atom, and is called **primary** or **secondary** depending on whether this atom is **primary** or **secondary**. It is called **ground** if it does not have free variables, and **aggregation-free** if it does not contain aggregation functions. A **minimal secondary atom** is a ground secondary atom of the form $w(\mathbf{a}) = \alpha$ for a constant $\alpha \in R$. A **minimal atom** (resp. **literal**) is either a ground primary atom (resp. literal) or a minimal secondary atom.

Example 5. In Example 3, one can build secondary terms $1 + 1$, $\text{size}(X)$, $\text{size}(X) * (1 + 1) + \text{trees}(Y)$ and $\text{sum}_X(\text{size}(X) \mid \text{adjacent}(X, Y))$.

The expression $\text{adjacent}(X, Y)$ is a primary atom, while $\text{size}(X) > 1 + 1$ is a secondary atom. Both atoms are neither ground nor minimal.

Metafinite signatures are used to reason on metafinite structures.

Definition 8 (Metafinite Structure). A *(metafinite) structure* $\mathcal{D}$ for a metafinite signature $\Sigma := (\sigma_A, \sigma_R, \sigma_W)$ consists of:

1. a finite σ_A -structure $\mathcal{A}$ with domain A , called the **primary part** of $\mathcal{D}$;
2. a finite or infinite σ_R -structure $\mathcal{R}$ with domain R , called the **secondary part** of $\mathcal{D}$;
3. for every aggregation function symbol Γ in σ_R a (permutation-invariant) function $\Gamma^{\mathcal{R}} : R^{<\infty} \rightarrow R$; and
4. a finite set W with an **attribute function** $w : A^k \rightarrow R$ for every k -ary attribute symbol w in σ_W .

Hereby, $R^{<\infty}$ denotes the set of all R -tuples of finite length.

Example 6. In Example 4, a structure consists of directed graphs A and B on the same set of nodes D , modelling patches of land with an adjacency and a barrier relation; the secondary structure $\mathbb{N}$ with the usual interpretation of $+$, $\times$, $>$, $=$, 0 , 1 , and sum ; and functions $\text{size} : D \rightarrow \mathbb{N}$, $\text{trees} : D \rightarrow \mathbb{N}$, and $\text{birds} : D \rightarrow \mathbb{N}$, modelling the size of a plot and the abundance of trees and birds.

3 Results

In this work, we augment metafinite Datalog programs with structural equations to represent causal relations, which are then composed into explanations.

Definition 9 (Metafinite Datalog Program). A *metafinite Datalog program* Π is a set of clauses $H \Leftarrow B_1, \dots, B_n$ where H is a primary atom and $B_1, \dots, B_n$ are aggregation-free literals (primary or secondary). As in classical Datalog, every relation symbol in σ_A occurring in the head of a clause is called **intentional**, and every other relation symbol is called **extensional**. Note that all secondary relations are extensional in this sense.

Example 7. A possible metafinite Datalog program is given by Clauses (1) and (2). For this program, $\text{connected}/2$ is intentional while $\text{adjacent}/2$, $\text{barrier}/2$ and $\text{trees}/1$ are extensional.

Grounding relates Datalog programs to structures.

Definition 10 (Grounding). Let $\Sigma := (\sigma_A, \sigma_R, \sigma_W)$ be a metafinite signature, Π a metafinite Datalog program, and $\mathcal{D} := (\mathcal{A}, \mathcal{R}, W)$ a metafinite structure. Denote by $\sigma_{\mathcal{A}}$ the relational signature σ_A augmented by a constant a for every element in the domain of $\mathcal{A}$.

The **grounding** $\bar{\Pi}$ of Π with respect to $\mathcal{A}$ is the metafinite Datalog program in the signature $\Sigma_{\mathcal{A}} := (\sigma_{\mathcal{A}}, \sigma_R, \sigma_W)$ obtained from Π by substituting all possible (constants representing) elements in $\mathcal{A}$ for the free variables in clauses C of Π .

The stable models of Gelfond and Lifschitz [9] are generalised as follow.

Definition 11 (Stable Model). *Let Π be a metafinite Datalog program and let $\mathfrak{D} := (\mathfrak{A}, \mathfrak{R}, W)$ be a metafinite structure. The **(primary) reduct** of Π to $\mathfrak{D}$ is obtained from the clauses of the grounding $\bar{\Pi}$ of Π with respect to $\mathfrak{A}$ as follows:*

- Remove clauses $H \leftarrow B_1, \dots, B_n$ for which there is a negative (or a secondary) literal in the body which is not satisfied in $\mathfrak{D}$;
- Remove negative (or secondary) body literals from every remaining clause.

Note that secondary literals are treated just like negative (primary) literals in the construction of the reduct of Gelfond and Lifschitz [9].

*The structure $\mathfrak{D}$ is a **(metafinite) stable model** for Π if $\mathfrak{A}$ is a minimal model of the reduct of Π to $\mathfrak{D}$.*

Example 8. The reduct of the program in Example 7 to a structure $\mathfrak{D}$ is the program with clauses $\text{connected}(a, a)$ for every plot of land a in the primary domain that has trees, and clauses $\text{connected}(a, b) \Leftarrow \text{connected}(a, c), \text{adjacent}(c, b)$ for all a, b, c in the primary domain such that b has trees and there is no barrier between c and b .

The primary reduct contains clauses

$$\text{connected}(a, b) \Leftarrow \text{connected}(a, c), \text{adjacent}(c, b), \neg \text{barrier}(c, b)$$

whenever b has at least one tree.

Since stable models of a metafinite Datalog programme can be characterised as stable models of their primary reduct in the sense of Gelfond and Lifschitz [9], we can apply known results and alternative characterisations of stable models.

Fix a metafinite signature $\Sigma := (\sigma_A, \sigma_R, \sigma_W)$ and denote by $\mathcal{F}$ the set of secondary terms. To represent the structural equations of Pearl [16], we introduce a relation symbol $(:=)_{w,F}$ for every attribute function symbol $w \in \sigma_W$ and $F \in \mathcal{F}$.

Definition 12 (Pearl Extension). *Introduce **causal definitions**, that is, $(k + l)$ -ary relation symbols $(:=)_{w,F}/(k + l)$ for every attribute function symbol w/k of arity k and every secondary term $F(X_1, \dots, X_l) \in \mathcal{F}$ with free variables $X_1, \dots, X_l$.*

*The **Pearl extension** $\bar{\Sigma}$ of the metafinite signature Σ then is the metafinite signature that results by adding the causal definitions $(:=)_{w,F}$ to the relational alphabet σ_A , i.e., $\bar{\Sigma} := (\sigma_A \cup \{(:=)_{w,F} \mid w \in \sigma_W, F \in \mathcal{F}\}, \sigma_R, \sigma_W)$.*

*A primary atom $(:=)_{w,F}(t_1, \dots, t_k, s_1, \dots, s_l)$ with primary terms $t_1, \dots, t_k$ and $s_1, \dots, s_l$ is called a **relational structural equation**. It is also denoted by*

$$w(t_1, \dots, t_k) := F(s_1, \dots, s_l).$$

By construction relational structural equations are interpreted as primary atoms in the Pearl extension $\bar{\Sigma}$. Hence, we can use the machinery of metafinite Datalog programs to merge the explanatory interpretation of causality in logic programs with the definitory account in the causal models of Pearl [16].

Definition 13 (Definitory Causal Model). A *definitory causal model* $\mathcal{M}$ is a metafinite Datalog program in the Pearl extension $\bar{\Sigma}$ such that

- no causal definition $(:=)_{w,F}$ is an extensional relation;
- no attribute symbol w for which $(:=)_{w,F}$ occurs in $\mathcal{M}$ also occurs in a secondary body literal of a clause in $\mathcal{M}$.

We call those attribute symbols w for which some causal definition $(:=)_{w,F}$ occurs in $\mathcal{M}$ **intentional**, and the remaining attribute symbols **extensional**.

A **stable model** for $\mathcal{M}$ is a $\bar{\Sigma}$ -structure $\mathfrak{D} := (\mathfrak{A}, \mathfrak{R}, W)$, where $\mathfrak{A}$ is a stable model of $\mathcal{M}$ considered as a Datalog program and $\mathfrak{R} \models w(t_1, \dots, t_k) = F(s_1, \dots, s_l)$ whenever $\mathfrak{A} \models w(t_1, \dots, t_k) := F(s_1, \dots, s_l)$. It **solves equations uniquely** if it is the only stable model for $\mathcal{M}$ with primary part $\mathfrak{A}$ and secondary part $\mathfrak{R}$ which agrees with W on all extensional literals.

Example 9. A definitory causal model is given by Clauses (1) and (2) together with Relational Structural Equation (3).

A sufficient condition for a definitory causal system to solve equations uniquely is when all its equations can be evaluated recursively from the bottom up. To formulate this condition more precisely, we need some more notation.

Definition 14. For a ground secondary term F and a primary structure $\mathfrak{A}$, we define the set of **attributes mentioned by F** , denoted $W(\mathfrak{A}, F)$, by induction on the definition of a secondary term.

- If F is a constant, then $W(\mathfrak{A}, F) = \emptyset$.
- If $F = g(F_1, \dots, F_n)$, then $W(\mathfrak{A}, F) = \bigcup_{i=1, \dots, n} W(\mathfrak{A}, F_i)$.
- If $F = w(t_1, \dots, t_n)$, then $W(\mathfrak{A}, F) = \{w(a_1, \dots, a_n)\}$, where $a_1, \dots, a_n$ are the interpretations of $t_1, \dots, t_n$ in $\mathfrak{A}$ respectively.
- If $F = \Gamma_x(F \mid R(\mathbf{t}))$, then $W(\mathfrak{A}, F) = \bigcup \{W(\mathfrak{A}, F(\mathbf{a})) \mid \mathbf{a} \in A, \mathfrak{A} \models R(\overrightarrow{t(\mathbf{a})})\}$.

Example 10. The term $\text{trees}(a)$ mentions only the attribute $\text{trees}(a)$, the constant term 5 mentions no attributes, while the term $\text{sum}_Y(\text{size}(Y) \mid \text{connected}(a, Y))$ mentions the attributes $\{\text{size}(b) \mid \mathfrak{A} \models \text{connected}(a, b)\}$. Thus, the term

$$5 \times (\text{trees}(X) \times \text{sum}_Y(\text{size}(Y) \mid \text{connected}(X, Y)))$$

mentions the set of attributes $\{\text{size}(b), \text{trees}(a) \mid \mathfrak{A} \models \text{connected}(a, b)\}$ a element of domain of $\mathfrak{A}$.

Definition 15 (Hierarchical equations). For any subset of attribute symbols $\theta \subseteq \sigma_W$ and metafinite structure $\mathfrak{D} := (\mathfrak{A}, \mathfrak{R}, W)$, let $G_{\theta, \mathfrak{D}}$ be the graph whose nodes are terms of the form $w(\mathbf{a})$ for an attribute symbol $w \in \theta$ and $\mathbf{a}$ a matching tuple of elements in the domain of $\mathfrak{A}$, and with an edge $v(\mathbf{b}) \rightarrow w(\mathbf{a})$ whenever $w(\mathbf{a}) := F(\mathbf{b})$ is true in $\mathfrak{A}$ and $v(\mathbf{b}) \in W(\mathfrak{A}, F(\mathbf{b}))$.

Then a model has hierarchical equations if the following two conditions are satisfied:

1. For every $\mathbf{a}$ a matching tuple of elements of the primary sort, there is precisely one term $w(\mathbf{a}) := F(\mathbf{b})$ true in $\mathfrak{A}$.
2. For $\theta \subseteq \sigma_W$ the set of intentional functions, the graph $G_{\theta, \mathfrak{D}}$ is acyclic.

As in the classical theory of functional causal models [16], hierarchical equations imply unique solvability.

Proposition 1. *A stable model of a definitory causal model with hierarchical equations solves equations uniquely.*

Example 11. Since the model of Example 9 only has one intentional attribute symbol and exactly one equation for each symbol, it has hierarchical equations.

An example of a model without hierarchical equations may additionally use the abundance of birds to explain the abundance of trees (e.g. for pollination or dispersal), for instance with an additional relational structural equation such as

$$\text{trees}(X) := \text{size}(X) \times \text{sum}_Y(\text{birds}(Y) \mid \top)(X);$$

indicating that the number of trees in any plot equals the product of the number of birds in the whole study system and the size of the plot.

As the causal models of Pearl [16] can be read as facts in a definitory causal model, where each “variable” as a nullary attribute symbol, definitory causal models jointly generalise Datalog programs and Pearlian causal models.

Interventions

Next, we lift the notion of intervention from causal models in Definition 1 and propositional logic programs in Definition 5 to definitory causal models.

Definition 16 (Modified Definitory Causal Model). *Let $\mathcal{M}$ be a definitory causal model together with a stable model $\mathfrak{D}$ that has hierarchical equations and i a set of minimal literals. The **modified (definitory causal) model** $\mathcal{M}_i$ is obtained by first forming the grounding $\bar{\mathcal{M}}$ of $\mathcal{M}$ and then performing the modification below:*

- If $(\neg)p \in i$ is a primary literal corresponding to the atom p , erase all clauses from $\bar{\mathcal{M}}$ with head p .
- If $p \in i$ is a primary atom, add the fact p .
- If $(w(\mathbf{a}) := \alpha) \in i$ is a minimal secondary atom, erase all clauses mentioning $w(\mathbf{a})$ in their heads and add the fact $w(\mathbf{a}) := \alpha$.

As is clear from the definitions, the notion of intervention in Definition 16 consistently generalises the ones in Definitions 2 and 5.

Example 12. In a grounding $\bar{\mathcal{M}}$ of the definitory causal model $\mathcal{M}$ in Example 3, an extrinsic factor leading to the death of all birds in Plot a can be modeled by replacing the fact $\text{birds}(a) := 5 \times \left(\text{trees}(a) \times \text{sum}_Y(\text{size}(Y) \mid \text{connected}(a, Y)) \right)$ with $\text{birds}(a) := 0$, thereby obtaining the modified model $\mathcal{M}_{\text{birds}(a) := 0}$.

If one connects Plots a and b by a special indirect connection, one removes all clauses with heads $\text{connected}(a, b)$ from $\bar{\mathcal{M}}$ and adds the fact $\text{connected}(a, b)$, thereby obtaining the modified model $\mathcal{M}_{\text{connected}(a, b)}$.

Explanations

Let $\mathfrak{D} := (\mathfrak{A}, \mathfrak{R}, W)$ be a stable model of a definitory causal model $\mathcal{M}$. Since $\mathfrak{A}$ is a stable model of the primary reduct of the metafinite Datalog program given by $\mathcal{M}$, it follows from Rückschloß and Weitzkämper [22] that every primary atom $P(\mathbf{a})$ that is true in $\mathfrak{A}$ has an explanation as a derivation of $P(\mathbf{a})$ grounded in extensional predicates (Definition 4). We would like to extend this notion of explanation to values of attributes. This extension is most transparent and useful in models with hierarchical sets of equations; hence, we assume from now on that $\mathfrak{D} := (\mathfrak{A}, \mathfrak{R}, W)$ has hierarchical equations.

Let $w \in W$ be a k -ary attribute symbol and $\mathbf{a} \in A^k$ such that $\mathfrak{R} \models w(\mathbf{a}) = \alpha$. Extending Definition 4 by allowing relational structural equations of the form $w(\mathbf{a}) := F(\mathbf{b})$ to serve as explanations for equalities of the same form yields the notion of explanation for definitory causal models. To accommodate aggregation functions, we also need to collect groundings of the conditions occurring in them.

Definition 17 (Condition Set, Explainability). *We define the **condition set** of a secondary term as follows: For a term of the form $\Gamma_{\mathbf{x}}(F \mid R(\mathbf{t}))$, the condition set $\text{Cond}(\mathfrak{A}, \Gamma_{\mathbf{x}}(F \mid R(\mathbf{t})))$ with respect to $\mathfrak{A}$ is the union of the sets $\{R(\overrightarrow{t(\mathbf{a})}) \mid \mathfrak{A} \models R(\overrightarrow{t(\mathbf{a})})\}$ and $\{\neg R(\overrightarrow{t(\mathbf{a})}) \mid \mathfrak{A} \not\models R(\overrightarrow{t(\mathbf{a})})\}$. For a general term F , the conditional set $\text{Cond}(\mathfrak{A}, F)$ is the union of the condition sets of all subterms of the above form.*

Let $\bar{\mathcal{M}}$ be the grounding of a definitory causal model $\mathcal{M}$ to its metafinite stable model $\mathfrak{D} := (\mathfrak{A}, \mathfrak{R}, W)$. Let λ be a set of minimal literals of $\Sigma := (\sigma_{\mathfrak{A}}, \sigma_R, \sigma_W)$ and let l be a minimal atom. Then, the following are the basic one-step explanation for each type of minimal literal, relative to the fixed structure $\mathfrak{R}$:

Causal rules: If l is an intentional primary atom and $H \Leftarrow \lambda, \kappa \in \bar{\mathcal{M}}$, where λ is a set of primary literals and κ a set of secondary literals, then $\lambda, \lambda' \Rightarrow_{\Pi} l$ whenever λ' is the set $w(\mathbf{b})_i = \alpha_i$, where $w(\mathbf{b})_i$ ranges over all secondary terms of the form $w(\mathbf{t})$ mentioned in κ , and the (α_i) are such that substituting α_i for $w(\mathbf{b})_i$ in κ makes κ true in $\mathfrak{R}$.

Unexplained literals: If l is a minimal extensional atom or the negation of a primary atom, then $l \Rightarrow_{\Pi} l$.

Defining equations: Let F be a secondary term and w an attribute function symbol, and let $t_1, \dots, t_n$ enumerate $W(\mathfrak{A}, F)$. Let $\lambda := \{t_i(\mathbf{b}) = \alpha_i\}_{1 \leq i \leq n}$ for some $\alpha_i \in R$, and let α be the value of the corresponding substitution of α_i for t_i in F in $\mathfrak{R}$.

Then, $\lambda, \text{Cond}(\mathfrak{A}, F), w(\mathbf{a}) := F(\mathbf{b}) \Rightarrow_{\Pi} w(\mathbf{a}) = \alpha$

Explanations are further composed of One-Step Explanations (Literal Monotonicity), (Literal Cut) and (Literal Contradiction) in Definition 4.

Example 13. Consider the situation of Example 9, which has hierarchical equations with a plot of land a . Then, following the case for defining equations, a one-step explanation for $\text{birds}(a) = \alpha$ is of the form

$$\lambda, \text{Cond}(\mathfrak{A}, F), \text{birds}(a) := F \Rightarrow_{\mathcal{M}} \text{birds}(a) = \alpha.$$

For $F = 5 \times \text{trees}(a) \times \text{sum}_Y(\text{size}(Y) \mid \text{connected}(a, Y))$, λ would be a value for $\text{tree}(a)$ as well as an allocation of values for $\text{size}(Y)$ for the subset of plots connected to a . The condition set is a complete description of connectedness and non-connectedness to a across all plots.

In stable models with hierarchical equations, every true minimal literal has an explanation from extensional atoms and extensional attribute functions.

Theorem 1. *Let $\mathcal{M}$ be the grounding of a definitory causal model to its stable model with hierarchical equations $\mathfrak{D} := (\mathfrak{A}, \mathfrak{R}, W)$. Then, a minimal intentional literal l is true in $\mathfrak{D}$ if and only if $\lambda \Rightarrow_{\mathcal{M}} l$, where λ is the set of all minimal extensional literals and ground negative primary intentional literals true in $\mathfrak{D}$.*

Proof. Note first that $\mathfrak{A}$ is a stable model of the primary reduct of $\mathcal{M}$. Hence, by Theorem 4 of [22], a primary literal is true in $\mathfrak{A}$ if and only if $\lambda \Rightarrow_{\mathcal{M}} l$ for λ the set of extensional or negated literals in the primary reduct of $\mathcal{M}$. Since the primary reduct only removes secondary body literals κ that are true in $(\mathfrak{A}, \mathfrak{R}, W)$, adding the statements $w(\mathbf{b}) = \alpha$ for $w(\mathbf{b}) \in W(\kappa)$, where α is the corresponding value in W , to those in λ allows every causal rule applicable in the primary reduct to be applied in $\mathcal{M}$ itself.

This demonstrates the statement for primary literals. If $w(\mathbf{a}) = \alpha$ is a minimal secondary literal, then there is a single structural equation $w(\mathbf{a}) := F(\mathbf{b})$ true in $\mathfrak{A}$. By the preceding note, this implies that (only) $w(\mathbf{a}) := F(\mathbf{b})$ has a grounded explanation. Hence, we can apply the defining equations rule to explain $w(\mathbf{a}) = \alpha$ from $w(\mathbf{a}) := F(\mathbf{b})$, λ and the condition set as in that clause. Note that all attributes in λ are either extensional or of lower height. So, if α is the value of $w(\mathbf{a})$ under W , then we can proceed to construct the explanation by induction on the height of w .

Theorem 1 shows that definitory causal models bridge the gap between Pearl’s query-based account of causality [16] and the explanation-based account found in philosophy, as announced earlier.

Expressivity

The expressivity of metafinite logical frameworks was studied by Grädel and Gurevich [11], who investigated the descriptive complexity theory of various metafinite logical systems with various secondary parts. Such questions arise equally in the context of metafinite Datalog programs and definitory causal systems. Here, we give just one result showing that on certain arithmetical structures, where the secondary part are natural numbers, definitory causal systems do not go beyond metafinite Datalog programs in recognising classes of extensional structures. Our notation is taken from Grädel and Gurevich [11], with some modifications for technical convenience.

Definition 18. *Let $\Sigma := (\sigma_A, \sigma_R, \sigma_W)$ be a signature, and $\mathfrak{D} := (\mathfrak{A}, \mathbb{N}, W)$ a structure in that signature. We call Σ and $\mathfrak{D}$ **arithmetical** if all functions*

and relations in σ_R can be evaluated in polynomial time on $\mathbb{N}$, and if σ_R contains $\{+, -, *, 0, 1, <, =\}$ as well as a binary function symbol $\text{bit}/2$, such that $\text{bit}(m, n) \in \{0, 1\}$ is the m -th bit of n , and a unary function symbol $\text{bit}/1$ such that $\text{bit}(n)$ is the number of bits required to represent n . We say that $\mathfrak{D}$ is **ranked with ranking** r for a unary attribute symbol $r \in \sigma_W$ if the interpretation $r_W \in W$ is a bijection between the domain A of $\mathfrak{A}$ and the numbers $0, \dots, |A| - 1 \in \mathbb{N}$.

Let K be a class of arithmetical structures with signature Σ . We call K **ranked** if there is a unary attribute symbol $r \in \sigma_W$ such that all $\mathfrak{D} \in K$ are ranked with ranking r . We say that K **has small weights** if there is a $k \in \mathbb{N}$ such that for every $(\mathfrak{A}, \mathbb{N}, W) \in K$, every $w \in \sigma_W$ and every $a_1, \dots, a_j$ in the domain A of $\mathfrak{A}$, $w(a_1, \dots, a_j)$ can be represented using $|A|^k$ bits, where j is the arity of w . Lastly, we say that K is **recognisable in polynomial time** if there is a polynomial time deterministic Turing machine that determines for any metafinite structure $\mathfrak{D}$ with fixed secondary part $\mathbb{N}$ as input whether $\mathfrak{D}$ lies in K , where natural numbers are encoded using their binary representation.

We show that in an appropriate setting, stratified definitory causal systems do not add to the expressive power of metafinite Datalog, and that metafinite Datalog programs themselves capture precisely polynomial-time recognisable classes of structures.

Theorem 2. *Let K be a ranked class of arithmetical structures with small weights in the extensional vocabulary that is closed under isomorphism. We say that K is recognisable by a stratified definitory causal system (resp. metafinite Datalog program) if there is a stratified definitory causal system (resp. metafinite Datalog program) Π and a ground literal l such that K is the class of all metafinite structures ω for which l is true in the unique metafinite stable model of Π applied to ω . Then the following are equivalent:*

1. *K is recognisable by a stratified definitory causal system with hierarchical equations;*
2. *K is recognisable in polynomial time;*
3. *K is recognisable by a metafinite Datalog program where only extensional atoms occur negated.*

Proof. Condition 1 implies Condition 2 since we can compute the stable model of a stratified (metafinite) Datalog program bottom-up in polynomial time, and we can then solve the hierarchical system of equations in polynomial time too as long as all operations occurring in the defining equations are themselves computable in polynomial time.

To prove that Condition 2 implies Condition 3, we follow the general proof idea of [11] for their Theorem and emulate the attributes in the primary part using bit-wise representation. Since K has small weights, there is a natural number k such that for all $(\mathfrak{A}, \mathbb{N}, W) \in K$, every $w \in \sigma_W$ and every $a \in A$, the binary representation of $w(a)$ has at most $|A|^k$ bits. Let furthermore r be the ranking attribute. We represent the numbers between 0 and $|A|^k - 1$ as elements of A^k by assigning to every tuple $b_1, \dots, b_k$ the number $r(b_1) + |A|r(b_2) + \dots + |A|^{k-1}r(b_k)$.

Consider the extended primary signature σ'_A , which adds for each j -ary attribute function w a $j + k$ -ary predicate symbol P_w , and consider the family K' of (not metafinite) σ'_A -structures $\mathfrak{A}_W$ which, for every metafinite structure $(\mathfrak{A}, \mathbb{N}, W) \in K$, consists of the σ'_A -structure $\mathfrak{A}_W$ with $P_w(a_1, \dots, a_j, b_1, \dots, b_k)$ true in $\mathfrak{A}_W$ when the b 'th bit of the binary representation of $w(\mathbf{a})$ is 1.

Then clearly K' is recognisable in polynomial time whenever K is. Thus, by the (Datalog version of) the Immermann-Vardi Theorem [6, Chapters 7 and 8], there is a Datalog program Π' where only extensional atoms occur negated and a ground literal l such that Π' applied to $\mathfrak{A}'$ satisfies l exactly if $\mathfrak{A}' \in K'$. We obtain a metafinite Datalog program Π recognising K by augmenting Π' as follows: Add clauses

$$\text{maxelem}(z) \leftarrow \neg r(y) < r(x). \quad \text{test} \leftarrow l, \text{small}.$$

as well as for every $w \in \sigma_W$, clauses

$$\begin{aligned} P_w(x_1, \dots, x_j, y_1, \dots, y_k) &\leftarrow \text{maxelem}(z), \\ \text{bit}(r(y_1) + (r(z) + 1)r(y_2) + \dots + (r(z) + 1)^{k-1}r(y_k), w(x_1, \dots, x_j)) &= 1. \\ \text{big}_w(x_1, \dots, x_j) &\leftarrow \text{maxelem}(z), \text{bits}(w(x_1, \dots, x_j)) > \text{maxelem}(z)^k. \end{aligned}$$

and, for $\sigma_W = w_1, \dots, w_n$ the clause

$$\text{small} \leftarrow \neg \text{big}_{w_1}(x_1, \dots, x_{j_1}), \dots, \neg \text{big}_{w_n}(x_1, \dots, x_{j_n}).$$

where exponentiation by k is expressed by iterating multiplication. Then, Π applied to a metafinite structure $\mathfrak{A}$ satisfies test precisely if $\mathfrak{A} \in K$ as desired.

As the implication from Condition 3 to Condition 1 is clear from the fact that metafinite Datalog programs are a special case of definitory causal systems with hierarchical equations, this proves the theorem.

4 Conclusion

We augment causally interpreted Datalog programs with the causal models of Pearl [16] and introduce definitory causal models. The notions of intervention and explanation are generalised, their expressivity is analysed, and an implementation is provided.

Future work should incorporate uncertainty, ideally subsuming probabilistic logic programming [20], Markov logic networks [19], and relational Bayesian networks [13]. This would provide a unifying perspective on statistical relational AI [18]. Last but not least, integrating definitory causal reasoning with the temporal causal reasoning of Vennekens et al. [23] and developing relational representations of interventions and explanations are promising directions for further research.

References

1. Bochman, A.: A logic for causal reasoning. In: Gottlob, G., Walsh, T. (eds.) Proc. IJCAI '03. pp. 141–146. Morgan Kaufmann (2003)
2. Bochman, A.: A Logical Theory of Causality. The MIT Press (2021)
3. Bochman, A., Lifschitz, V.: Pearl's causality in a logical setting. In: Bonet, B., Koenig, S. (eds.) Proc. AAAI '15. pp. 1446–1452. AAAI Press (2015)
4. Carloni, G., Berti, A., Colantonio, S.: The role of causality in explainable artificial intelligence. WIREs Data. Min. Knowl. Discov. **15**, e70015 (2025)
5. Clark, K.L.: Negation as failure. In: Logic and Data Bases. pp. 293–322. Springer US (1978)
6. Ebbinghaus, H., Flum, J., Thomas, W.: Einführung in die mathematische Logik. Springer Spektrum, Berlin, DE, 6 edn. (2018)
7. Eelink, G., Rückschloß, K., Weitkämper, F.: How artificial intelligence leads to knowledge why: An inquiry inspired by aristotle's posterior analytics. Internat. J. Approx. Reason. **190**, 109603 (2026)
8. Gao, C., Zheng, Y., Wang, W., Feng, F., He, X., Li, Y.: Causal inference in recommender systems: A survey and future directions. ACM Trans. Inf. Syst. **42**, 88:1–88:32 (2024)
9. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Proc. Int. Log. Program. Conf. Symp. pp. 1070–1080. MIT Press (1988)
10. Granger, C.W.J.: Investigating causal relations by econometric models and cross-spectral methods. Econometrica **37**, 424–438 (1969)
11. Grädel, E., Gurevich, Y.: Metafinite model theory. Inf. Comput. **140**, 26–81 (1998)
12. Hume, D.: An Enquiry Concerning Human Understanding. A. Millar, London (1748), sect. IV
13. Jaeger, M.: Relational bayesian networks. In: Geiger, D., Shenoy, P.P. (eds.) Proc. UAI '97. pp. 266–273. Morgan Kaufmann (1997)
14. Lifschitz, V.: On the logic of causal explanation. Artif. Intell. **96**, 451–465 (1997)
15. Lifschitz, V., Yang, F.: Functional completion. J. Appl. Non Class. Logics **23**, 121–130 (2013)
16. Pearl, J.: Causality. Cambridge University Press (2000)
17. Proserpi, M., Guo, Y., Sperrin, M., Koopman, J.S., Min, J.S., He, X., Rich, S., Wang, M., Buchan, I.E., Bian, J.: Causal inference and counterfactual prediction in machine learning for actionable healthcare. Nat. Mach. Intell. **2**, 369–375 (2020)
18. Raedt, L.D., Kersting, K., Natarajan, S.: Statistical Relational Artificial Intelligence: Logic, Probability, and Computation. Morgan & Claypool Publishers (2016)
19. Richardson, M., Domingos, P.: Markov logic networks. Mach. Learn. **62**, 107–136 (2006)
20. Riguzzi, F.: Foundations of Probabilistic Logic Programming: Languages, Semantics, Inference and Learning. River Publishers (2020)
21. Rückschloß, K., Weitkämper, F.: Exploiting the full power of Pearl's causality in probabilistic logic programming. In: Proc. ICLP Workshops '22. CEUR-WS.org (2022)
22. Rückschloß, K., Weitkämper, F.: How rules represent causal knowledge: Causal modeling with abductive logic programs. In: RuleML+RR: 25 Companion. CEUR Workshop Proceedings, vol. 4083. CEUR-WS.org, Istanbul, Türkiye (2025)
23. Vennekens, J., Denecker, M., Bruynooghe, M.: CP-logic: A language of causal probabilistic events and its relation to logic programming. Theory Pract. Log. Program. **9**, 245–308 (2009)