Extending the Role of Causality in Probabilistic Modeling

Joost Vennekens, Marc Denecker, Maurice Bruynooghe

{joost, marcd, maurice}@cs.kuleuven.be
K.U. Leuven, Belgium
Causality

- Causality is **central concept** in much of human knowledge & reasoning
- What is its role in **probabilistic modeling**?

### Bayesian networks

- Acyclic Bayesian networks can be given causal interpretation [Pearl, 2000]
- Seems to be important part of success of this language
- However, Bayesian networks are not **inherently** causal
  - Formally: probabilistic independencies, conditional probabilities
  - Causal interpretation is no longer possible for cyclic nets

In this talk, we will

- Present language with causality at the **heart** of its semantics
- Analyse its properties, especially compared to Bayesian nets
Introduction

Formal definition of CP-logic

Bayesian networks in CP-logic

The role of causality

The link to Logic Programming

Conclusion
Basic construct

Express both

- **Causal** relations between propositions
- **Probabilistic events**

### Conditional probabilistic event (CP-event)

If propositions $b_1, \ldots, b_n$ hold, then a probabilistic event will happen that causes at most one of propositions $h_1, h_2, \ldots, h_m$, where the probability of $h_1$ being caused is $\alpha_1$, the probability of $h_2$ is $\alpha_2$, \ldots, and the probability of $h_m$ is $\alpha_m$ (with $\sum_i \alpha_i \leq 1$).

$$(h_1 : \alpha_1) \lor \cdots \lor (h_m : \alpha_m) \leftarrow b_1, \ldots, b_n.$$
Combining CP-events

- Meaning of single CP-event is clear
- But what does a set of CP-events mean?
- Terminology:
  - Set of CP-events is called CP-theory
  - Language of CP-theories is CP-logic
- Meaning of CP-theory is based on two fundamental principles
  - Principle of independent causation
  - Principle of no deus ex machina effects
Principle of independent causation

Every CP-event represents an independent causal process

- Learning outcome of one CP-event
  - May give information about whether another CP-event happens
  - But not about the outcome of another CP-event

- Crucial to have modular representation, that is elaboration tolerant w.r.t. adding new causes

- Compact representation of relation between effect and a number of independent causes for this effect

- Make abstraction of order in which CP-events are executed
No deus ex machina principle

Nothing happens without a cause

- Fundamental principle of causal reasoning
- Especially important for cyclic causal relations
- Compact representations
  - Cases where there is no cause for something can simply be ignored
Semantics

Under these two principles, CP-theory constructively defines probability distribution on interpretations.

**Constructive process**

- Simulate CP-event $(h_1 : \alpha_1) \lor \cdots \lor (h_m : \alpha_m) \leftarrow b_1, \ldots, b_n$.
  - Derive $h_i$ with $\alpha_i$
  - Derive nothing with $1 - \sum \alpha_i$
- Is only allowed if
  - All preconditions $b_1, \ldots, b_n$ have already been derived
  - Event has not been simulated before
- Start from $\{\}$ and simulate as many CP-events as possible

Probability of interpretation is probability of being derived by this process.
Semantics

Theorem

The order in which CP-events are simulated does not matter, i.e., all sequences give same distribution

This follows from:

- Principle of independent causation
- Once preconditions are satisfied, they remain satisfied

Two principles are incorporated into semantics

- Independent causation principle
  - A CP-event always derives $h_i$ with probability $\alpha_i$
- “No deus ex machina” principle
  - Atom is only derived when it is caused by a CP-event with satisfied preconditions
An example

There are two causes for HIV infection: intercourse with infected partner (0.6) and blood transfusion (0.01). Suppose that $a$ and $b$ are partners and $a$ has had a blood transfusion.

\[
\begin{align*}
(hiv(a) : 0.6) &\leftarrow hiv(b). \\
(hiv(b) : 0.6) &\leftarrow hiv(a). \\
(hiv(a) : 0.01).
\end{align*}
\]

- Principle of independent causation
  - Clear, modular, compact representation
  - Elaboration tolerant, e.g., add $(hiv(b) : 0.01)$.
- “No deus ex machina”-principle
  - Cyclic causal relations
  - No need to mention that HIV infection is impossible without transfusion or infected partner
Introduction

Formal definition of CP-logic

Bayesian networks in CP-logic

The role of causality

The link to Logic Programming

Conclusion
Negation

- **Negated** atoms also allowed as preconditions
- Absence of a cause for an atom can cause some other atom
  - Absence of a cause for termination of fluent causes it to persist
  - Absence of a cause for winning/losing game causes it to continue
- Makes representation more **compact**
- But causes **problem** with semantics
  - It is no longer the case that true preconditions remain true, so order of CP-events might matter
  - \((\text{heads} : 0.5) \leftarrow \text{toss.} \)
  - \(\text{win} \leftarrow \neg \text{heads.} \)
  - However, we don’t want to force explicit use of time
  - Most reasonable convention: execute event depending on \(\neg p\) only after all possible causes for \(p\) have been exhausted
Formal solution (for now)

Stratified CP-theories

- Assign level $lvl(p) \in \mathbb{N}$ to each atom $p$
- Such that for all rules $r$
  - If $h \in head_{At}(r)$, $b \in body_{+}(r)$, then $lvl(h) \geq lvl(b)$
  - If $h \in head_{At}(r)$, $b \in body_{-}(r)$, then $lvl(h) > lvl(b)$
- Level of $r$ is $\min_{p \in At(r)} lvl(p)$

- Execute rules with lowest level first
  - By the time we get to rule with precondition $\neg p$, all events that might cause $p$ have already been executed
  - If $p$ has not been derived, it never will
Formal definition of CP-logic

- A CP-theory is a **stratified** set of rules of the form:

  \[(h_1 : \alpha_1) \lor \cdots \lor (h_m : \alpha_m) \leftarrow b_1, \ldots, b_n.\]

- With \(h_i\) atoms, \(b_i\) literals, \(\alpha_i \in [0, 1]\) with \(\sum_i \alpha_i \leq 1\)

- A rule \((h : 1) \leftarrow b_1, \ldots, b_n\). is written as \(h \leftarrow b_1, \ldots, b_n\).
Probabilistic transition system

\[(h_1 : \alpha_1) \vee \cdots \vee (h_m : \alpha_m) \leftarrow b_1, \ldots, b_n.\]

- Tree structure \(\mathcal{T}\) with probabilistic labels
- Interpretation \(\mathcal{I}(c)\) for each node \(c\) in \(\mathcal{T}\)
- Node \(c\) executes rule \(r\) if children are \(c_0, c_1, \ldots, c_n\)
  - for \(i \geq 1\), \(\mathcal{I}(c_i) = \mathcal{I}(c) \cup \{h_i\}\) and \(\lambda(c, c_i) = \alpha_i\)
  - \(\mathcal{I}(c_0) = \mathcal{I}(c)\) and \(\lambda(c, c_0) = 1 - \sum_i \alpha_i\)
- Rule \(r\) is executable in node \(c\) if
  - \(\mathcal{I}(c) \models r\), i.e., \(\text{body}_+(r) \subseteq \mathcal{I}(c)\) and \(\text{body}_-(r) \cap \mathcal{I}(c) = \{\}\)
  - No ancestor of \(c\) already executes \(r\)
Formal semantics of CP-logic

- System $\mathcal{I}$ runs CP-theory $C$
  - $\mathcal{I}(\text{root}) = \{\}$
  - Every non-leaf $c$ executes executable rule $r \in C$ with minimal level
  - No rules are executable in leaves
- Probability of $P_\mathcal{I}(c)$ of leaf $c$ is $\prod_{(a,b) \in \text{root}..c} \lambda(a, b)$
- Probability of $\pi_\mathcal{I}(I)$ of interpretation $I$ is $\sum_{\mathcal{I}(c)=I} P_\mathcal{I}(c)$

**Theorem**
Every $\mathcal{I}$ that runs a CP-theory $C$ has the same $\pi_\mathcal{I}$

- We denote this unique $\pi_\mathcal{I}$ by $\pi_C$
- Defines formal semantics of $C$
Introduction

Formal definition of CP-logic

Bayesian networks in CP-logic

The role of causality

The link to Logic Programming

Conclusion
A Bayesian network expresses

- Conditional probabilities
- Probabilistic independencies

For all nodes $m, n$, such that $n$ is not a successor of $m$, $n$ and $m$ are independent given value for $\text{Parents}(m)$

Can these independencies also be expressed in CP-logic?
Probabilistic independencies in CP-logic

- When can learning the truth of $p$ give direct information about $q$?
  1. $p$ is a precondition to event that might cause $q$
     \[ \exists r : p \in \text{body}(r) \text{ and } q \in \text{head}_{At}(r) \]
  2. $p$ and $q$ are alternative outcomes of the same CP-event
     \[ \exists r : p, q \in \text{head}_{At}(r) \]
- $p$ directly affects $q$ iff (1) or (2) holds
- $p$ affects $q = \text{transitive closure}$

**Theorem**
If $p$ does not affect $q$, then $p$ and $q$ are independent, given an interpretation for the atoms $r$ that directly affect $p$

Independencies of Bayesian network w.r.t. "is parent of"-relation = independencies of CP-theory w.r.t. "directly affects"-relation
(burg : 0.1).
(alarm : 0.9) ← burg, earthq.
(alarm : 0.8) ← burg, ¬earthq.
(earthq : 0.2).
(alarm : 0.8) ← ¬burg, earthq.
(alarm : 0.1) ← ¬burg, ¬earthq.

Can be extended to a general way of representing Bayesian networks in CP-logic
Introduction

Formal definition of CP-logic

Bayesian networks in CP-logic

The role of causality

The link to Logic Programming

Conclusion
Motivation

- CP-logic can express probabilistic knowledge in the same way as Bayesian networks
- Often, this is not the most natural way
- Differences show role of causality
- Arise from the two principles of CP-logic
  - Principle of independent causation
    - Independent causes for the same effect
  - “No deus ex machina”-principle
    - Cyclic causal relations
    - Ignoring cases where nothing happens
Independent causes for the same effect

Consider a game of Russian roulette with two guns, one in the player’s right hand and one in his left. Each of the guns is loaded with a single bullet. What is the probability of the player dying?

\[
\begin{align*}
\text{(death : 1/6)} & \leftarrow \text{fire(left
gun)}. \\
\text{(death : 1/6)} & \leftarrow \text{fire(right
gun)}. \\
\end{align*}
\]

<table>
<thead>
<tr>
<th></th>
<th>left, right</th>
<th>¬ left, right</th>
<th>left, ¬ right</th>
<th>¬ left, ¬ right</th>
</tr>
</thead>
<tbody>
<tr>
<td>death</td>
<td>11/36</td>
<td>1/6</td>
<td>1/6</td>
<td>0</td>
</tr>
</tbody>
</table>
Independent causes for the same effect (2)

\[(\text{death} : 1/6) \leftarrow \text{fire(left\_gun)}.\]
\[(\text{death} : 1/6) \leftarrow \text{fire(right\_gun)}.\]

<table>
<thead>
<tr>
<th></th>
<th>left, right</th>
<th>\neg left, right</th>
<th>left, \neg right</th>
<th>\neg left, \neg right</th>
</tr>
</thead>
<tbody>
<tr>
<td>death</td>
<td>11/36</td>
<td>1/6</td>
<td>1/6</td>
<td>0</td>
</tr>
</tbody>
</table>

- Independence between causes for death is **structural** property
  - \(\text{fire(left\_gun)}, \text{fire(right\_gun)}\) not in same body
  - \(11/36 = 1/6 + 1/6 - 1/6 \cdot 1/6\)

**Qualitative ↔ quantitative knowledge**

- Treated differently, e.g., qualitative knowledge is more robust
- Different origins, e.g.,
  - Quantitative: derived from data
  - Qualitative: from background knowledge about domain
Independent causes for the same effect (3)

\[ \text{death} : 1/6 \leftarrow \text{fire(left\_gun)}. \]
\[ \text{death} : 1/6 \leftarrow \text{fire(right\_gun)}. \]

<table>
<thead>
<tr>
<th></th>
<th>left, right</th>
<th>¬ left, right</th>
<th>left, ¬ right</th>
<th>¬ left, ¬ right</th>
</tr>
</thead>
<tbody>
<tr>
<td>death</td>
<td>11/36</td>
<td>1/6</td>
<td>1/6</td>
<td>0</td>
</tr>
</tbody>
</table>

- Probabilities are **causal** rather than **conditional**
  - More informative: Conditional can be derived from causal
  - Using causal probabilities is more compact
    - For \( n \) guns: \( n \) versus \( 2^n \) entries
  - (Can be partly avoided by introducing new nodes)

- **Elaboration tolerance** w.r.t. adding new causes
  - Player can get heart attack: \( \text{death} : 0.1 \).
Cyclic causal relations

**HIV infection**

\[
\begin{align*}
(hiv(X) : 0.6) & \leftarrow hiv(Y), \text{partners}(X, Y). \\
(hiv(X) : 0.01) & \leftarrow \text{blood\_transfusion}(X).
\end{align*}
\]

- For partners \(a\) and \(b\):
  \[
  (hiv(a) : 0.6) \leftarrow hiv(b).
  \]
  \[
  (hiv(b) : 0.6) \leftarrow hiv(a).
  \]

- **“No deus ex machina”-principle**
  - If no external causes, then neither \(a\) nor \(b\) is infected
  - If \(a\) undergoes blood transfusion, \(a\) is infected with \(0.01\) and \(b\) with \(0.01 \times 0.6\)
  - If both \(a\) and \(b\) have blood transfusion, \(a\) is infected with \(0.01 + 0.01 \times 0.6\)

- Cyclic causal relations require no special treatment
Cyclic causal relations in Bayesian networks

- New nodes $ext(x)$: $x$ has been infected by an external cause

\[
\begin{align*}
\text{bloodtrans}(b) & \rightarrow \text{ext}(b) \rightarrow \text{hiv}(b) \\
\text{bloodtrans}(a) & \rightarrow \text{ext}(a) \rightarrow \text{hiv}(a)
\end{align*}
\]

- $P(ext(a) \mid bloodtrans(a)) = 0.01$
- $P(hiv(a) \mid \neg ext(a), \neg ext(b)) = 0$
- $P(hiv(a) \mid \neg ext(a), ext(b)) = 0.6$
- $P(hiv(a) \mid ext(a), \neg ext(b)) = 1$
- $P(hiv(a) \mid ext(a), ext(b)) = 1$
Ignoring cases where nothing happens

In craps, one keeps on rolling a pair of dice until one either wins or loses. In the first round, one immediately wins by rolling 7 or 11 and immediately loses by rolling 2, 3, or 12. If any other number is rolled, this becomes the player’s so-called “box point”. The game then continues until either the player wins by rolling the box point again or loses by rolling a 7.

(\text{roll}(T+1, 2) : \frac{1}{12}) \lor \cdots \lor (\text{roll}(T+1, 12) : \frac{1}{12}) \leftarrow \neg \text{win}(T), \neg \text{lose}(T).

\text{win}(1) \leftarrow \text{roll}(1, 7).
\text{win}(1) \leftarrow \text{roll}(1, 11).
\text{lose}(1) \leftarrow \text{roll}(1, 2).
\text{lose}(1) \leftarrow \text{roll}(1, 3).
\text{lose}(1) \leftarrow \text{roll}(1, 12).

\text{boxpoint}(X) \leftarrow \text{roll}(1, X), \neg \text{win}(1), \neg \text{lose}(1).

\text{win}(T) \leftarrow \text{boxpoint}(X), \text{roll}(T, X), T > 1.

\text{lose}(T) \leftarrow \text{roll}(T, 7), T > 1.
Ignoring cases where nothing happens (2)

\[
\begin{align*}
\text{Craps} \\
(\text{roll}(T+1, 2) : \frac{1}{12}) \lor \cdots \lor (\text{roll}(T+1, 12) : \frac{1}{12}) & \leftarrow \neg \text{win}(T), \neg \text{lose}(T). \\
\text{win}(T) & \leftarrow \ldots \\
\text{lose}(T) & \leftarrow \ldots
\end{align*}
\]

- Only specify when game is won or lost
- Negation is used to express that game continues otherwise
- The “otherwise”-cases do not need to be explicitly mentioned

<table>
<thead>
<tr>
<th>state_t</th>
<th>(4, 2)</th>
<th>(4, 3)</th>
<th>(4, 4)</th>
<th>(4, 5)</th>
<th>(4, 6)</th>
<th>(4, 7)</th>
<th>(4, 8)</th>
<th>\ldots</th>
</tr>
</thead>
<tbody>
<tr>
<td>Win</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>\ldots</td>
</tr>
<tr>
<td>Lose</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>\ldots</td>
</tr>
<tr>
<td>Neither</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>\ldots</td>
</tr>
</tbody>
</table>
Introduction

Formal definition of CP-logic

Bayesian networks in CP-logic

The role of causality

The link to Logic Programming

Conclusion
An alternative semantics

- An instance of a CP-theory is a normal logic program that results from making a number of independent probabilistic choices.
  - For each rule \((h_1 : \alpha_1) \lor \cdots \lor (h_n : \alpha_n) \leftarrow \text{body}\):
    - Replace rule by \(h_i \leftarrow \text{body}\) with probability \(\alpha_i\).
    - Remove rule with probability \(1 - \sum_i \alpha_i\).
- Interpret such an instance by well-founded semantics.
- Probability of \(I\) is the sum of the probabilities of all instances that have \(I\) as their well-founded model.

**Theorem**

This probability distribution is the same as \(\pi_C\).
An alternative semantics (2)

Historical note

Instance-based semantics was defined first, for Logic Programs with Annotated Disjunctions (LPADs). The interpretation of rules as CP-events and link to causality were discovered later.

Usefulness

- Relax stratification condition
  - New characterization works for all CP-theories s.t. all instances have two-valued well-founded model
  - Weaker requirement
    - Not only static, syntactical stratification
    - But also dynamic, semantical stratification
- Clarify the relation between CP-logic and logic programming
Normal logic programs

\[ h \leftarrow b_1, \ldots, b_m. \]

- For normal logic program \( C \), \( \pi_C(wfm(C)) = 1 \)

**Intuitive meaning of rule**

If propositions \( b_1, \ldots, b_n \) hold, then an event will happen that causes \( h \)

- Interesting link between WFS and causality
  - [Denecker, Ternovska, 2005]: WFS is used to deal with causal ramifications in situation calculus

- WFS formalizes inductive definitions [Denecker, 1998]
  Inductive definition is set of deterministic causal events
Disjunctive logic programs

\[ h_1 \lor \cdots \lor h_n \leftarrow b_1, \ldots, b_m. \]

- Suppose every such rule represents CP-event
  \((h_1 : \alpha_1) \lor \cdots \lor (h_n : \alpha_n) \leftarrow b_1, \ldots, b_m\). with \(\sum_i \alpha_i = 1\)
- \{interpretation \(I \mid \pi_C(I) > 0\}\} does not depend on precise values of \(\alpha_i > 0\)
- This set gives possible world semantics for DLP

**Intuitive meaning of rule**

If propositions \(b_1, \ldots, b_n\) hold, then a non-deterministic event will happen that causes precisely one of \(h_1, h_2, \ldots, h_m\).

- Different from stable model semantics
  - Not about beliefs of an agent, but the outcome of causal events
- For stratified programs, identical to Possible Model Semantics [Sakama,Inoue]
Related work: P-log

Some differences

- **Focus**
  - CP-logic: only concerned with representing probability distribution
  - P-log: various kinds of updates
    - (It seems straightforward to define do-operator for CP-logic)
- In P-log, attributes have dynamic range
  - CP-logic only allows static enumeration of alternatives
- Probabilities are attached to
  - CP-logic: independent causes that might occur together
  - P-log: mutually exclusive circumstances, as in Bayesian networks
Introduction

Formal definition of CP-logic

Bayesian networks in CP-logic

The role of causality

The link to Logic Programming

Conclusion
Conclusion

- Study role of *causality* in probabilistic modeling
- **CP-logic**: sets of conditional probabilistic events
  - Principle of independent causation
  - Principle of no deus ex machina effects
- Can express same knowledge as *Bayesian networks*
- **Differences** in natural modeling methodology for
  - Independent causes for effect
  - Cyclic causal relations
  - Absence of a cause
- Different view on *Logic Programming*