Discovering Cyclic Causal Models with Latent Variables: A General SAT-Based Procedure

Pragmatics of SAT, 8.7.2013
Antti Hyttinen,
Patrik O. Hoyer,
Frederick Eberhardt,
Matti Järvisalo

Cause → Effect
1. Introduction
2. Problem Statement
3. Graphs and independencies
4. Encoding D-connection
5. Algorithm
6. Conclusion
1. Introduction
2. Problem Statement
3. Graphs and independencies
4. Encoding D-connection
5. Algorithm
6. Conclusion
Discovering Cyclic Causal Models with Latent Variables: A General SAT-Based Procedure

- Another application field for SAT-solving technology.
- Presentation only submission to this workshop to present the application area, and to get further ideas.
- To be published in International Conference on Uncertainty in Artificial Intelligence 2013, Seattle, USA.
- Hot field: Judea Pearl won the Turing Prize on probabilistic models and causality.
Causal Discovery

Nodes represent random variables for measurements.
Directed edges represent direct causal relationships.
Bidirected arcs represent unobserved common causes.
For example: different measurements of blood, life habits and acquired deceases.

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>z</th>
<th>w</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4</td>
<td>0.56</td>
<td>4</td>
<td>120</td>
</tr>
<tr>
<td>0.5</td>
<td>0.23</td>
<td>100</td>
<td>130</td>
</tr>
<tr>
<td>0.1</td>
<td>0.01</td>
<td>34</td>
<td>123</td>
</tr>
<tr>
<td>0.23</td>
<td>0.03</td>
<td>52</td>
<td>23</td>
</tr>
</tbody>
</table>

(e.g. Causal Bayes Net)
The meaning of the edges

- Directed edges represent causal relationships: When the cause is manipulated, the effect changes.

- Bidirected edge represents an existence of an unobserved common cause: Manipulating either variable does not change the other.
Why SAT?

- Most often data only partly constrains the causal graph structure, part is left undetermined.
- For **restricted cases** there are algorithms exploiting complicated **theory** of this undetermination.
- For the more **general case** we consider here, this may be **impossible**.
- But, **SAT-technology** can be used as a **solving engine** for combining the different constraints!
- We can consider unobserved variables, cycles and several data sets with manipulations, and background knowledge.
1. Introduction

2. Problem Statement

3. Graphs and independencies
4. Encoding D-connection
5. Algorithm
6. Conclusion
Constraint-based Causal Discovery

(IN)DEPENDENCE RELATIONS

\[
\begin{array}{c|c|c|c}
  x & y & z & w \\
  0.4 & 0.56 & 4 & 120 \\
  0.5 & 0.23 & 100 & 130 \\
  0.1 & 0.01 & 34 & 123 \\
  0.23 & 0.03 & 52 & 23 \\
  ... & ... & ... & ...
\end{array}
\]

- Testing statistical conditional independence relations in the data.
  - Dependence #1: x and w correlated.
  - Conditional independence #2: x does not help to predict w if we know the value of y.

Motivation: Complete generality of causal relations (continuous, discrete, nonlinear)
Problem Statement

- **INPUT:** conditional (in)dependence relations \( x \not\perp y | C \) obtained by running statistical independence tests on data set(s) over variables \( V \).

- **OUTPUT:** causal structures consistent with input:
  - for each pair \((x,y)\) of variables in \( V \) and each edge \( x \rightarrow y, y \rightarrow x \) and \( x \leftrightarrow y \) whether it
    - Is **present** (in all causal structures consistent with input)
    - Is **absent** (in all causal structures consistent with input)
    - Is **unknown** (present in some and absent in some causal structures consistent with input).

- **First step:** assume (in)dependence relations can be determined without an error!
Example output

- True causal graph:
  ![Causal Graph](attachment:image)

- INPUT: (In)dependence relations tested from data...
  \[ \begin{align*}
  x \not\perp z \\
  x \perp y \mid z, w \\
  x \not\perp w \mid z \\
  \ldots
  \end{align*} \]

- OUTPUT:
  - Unknown edges
  - Present edges
  - Absent edges
  ![Output Diagram](attachment:image)
Our SAT-based approach

1. Run **conditional independence tests** on the data set(s).

2. **Encode** the dependence and independence relations into the working formula $F$ (assignments of $F$ correspond to graphs consistent with input).

3. Determine the **backbone** of $F$ for the graph properties common to all graphs consistent with input (i.e. which edges are present or absent in all graphs consistent with input).
3. Graphs and Dependencies

1. Introduction
2. Problem Statement

4. Encoding D-connection
5. Algorithm
6. Conclusion
Random variables \( x \) and \( y \) are dependent given \( C \), \( x \not\perp y \mid C \), if and only if there is a d-connecting path given \( C \) between them (Pearl et al. 1990-).

A d-connecting path given \( C \) is path such that
- Every collider node \( c \) (=node connected with heads) on the path is in \( C \).
- Other nodes on the path are not in \( C \).

\[ \begin{align*}
R & \quad \text{(Rain)} \\
\downarrow & \\
W & \quad \text{(Wet lawn)} \\
\downarrow & \\
F & \quad \text{(Falling)}
\end{align*} \]

- \( R \) and \( W \) dependent given \( C={} \)? YES.
- \( R \) and \( F \) dependent given \( C={} \)? YES.
- \( R \) and \( F \) dependent given \( C={} \{W\} \)? NO.
D-connection (2)

- Random variables $x$ and $y$ are dependent given $C$, $x \not\perp y|C$, if and only if there is a **d-connecting path given $C$** between them (Pearl et al. 1990-).

- A **d-connecting path given $C$** is a path such that
  - Every **collider node** $c$ (node connected with heads) on the path is in $C$.
  - Other nodes on the path are not in $C$.

- B and E dependent given $C=\{\}$? NO.
- B and E dependent given $C=\{A\}$? YES.
1. Introduction
2. Problem Statement
3. Graphs and Dependencies
4. Encoding D-connection
5. Algorithm
6. Conclusion
Random variables $x$ and $y$ are dependent given $C$, $x \not\perp y \mid C$, if and only if there is a d-connecting path given $C$ between them (Pearl et al. 1990-).

A d-connecting path given $C$ is path such that:

- Every collider node $c$ (node connected with heads) on the path is in $C$.
- Other nodes on the path are not in $C$. 

\[ \begin{array}{c}
\text{• } \bullet \rightarrow c \leftarrow \bullet \bullet \bullet \\
\end{array} \]
Encoding D-connection in Prop. Logic

Dependence:

Boolean variable TRUE iff the variables u and v are observed dependent given C.

Graph:

Boolean variables TRUE iff the edge is present in the solution.
Encoding D-connection in Prop. Logic

Dependence:

\[ u \not\rightarrow v \mid C \]

Paths:

Boolean variables TRUE iff there is a d-connecting paths given C of length l, with given arrowheads and tails in the outermost edges.

Graph:

\[ [x \rightarrow y] \quad [x \leftrightarrow y] \]
Encoding D-separation in Prop. Logic

Dependence:

\[ [u \not\perp v \mid C] \iff \bigvee_{l=1}^{l_{\text{max}}} \left( [u \leftarrow \frac{l}{C} v] \lor [v \leftarrow \frac{l}{C} u] \lor [u \leftarrow \frac{l}{C} \cdot v] \lor [u \leftarrow \frac{l}{C} \cdot v] \right) \]

Paths:

\[ [x \leftarrow \frac{l}{C} y] \]

\[ [x \leftarrow \frac{l}{C} \cdots y] \]

\[ [x \leftarrow \frac{l}{C} \cdots y] \]

\[ [x \leftarrow \frac{l}{C} \cdots y] \]

Graph:

\[ [x \leftarrow \frac{1}{C} \cdots y] \iff [x \rightarrow y] \]

\[ [x \leftarrow \frac{1}{C} \cdots y] \iff [x \leftrightarrow y] \]

\[ [x \leftarrow \frac{1}{C} \cdots y] \iff 0 \]
Encoding D-connection in Prop. Logic

\[ [x \overset{C}{\rightarrow} y] \iff \bigvee_{z \notin C} \left( [x \overset{1}{\rightarrow} z] \land [z \overset{l-1}{\rightarrow} y] \right) \lor \bigvee_{z \in C} \left( [x \overset{1}{\rightarrow} z] \land [z \overset{l-1}{\leftarrow} y] \right) \]
Encoding D-connection in Prop. Logic

Dependence:

\[ u \not\sim v \mid C \iff \bigvee_{l=1}^{l_{\max}} \left( [u_{-\ldots}^{l} \rightarrow v] \lor [v_{-\ldots}^{l} \rightarrow u] \lor [u_{\ldots}^{l} \leftarrow v] \lor [u_{-\ldots}^{l} \leftarrow v] \right) \]

Paths:

\[ x_{-\ldots}^{l} \rightarrow y \iff \bigvee_{z \notin C} \left( [x_{-\ldots}^{l} \rightarrow z] \land [z_{-\ldots}^{l-1} \rightarrow y] \right) \lor \bigvee_{z \in C} \left( [x_{-\ldots}^{l-1} \rightarrow z] \land [z_{-\ldots}^{l} \rightarrow y] \right) \]

Graph:

\[ x_{-\ldots}^{1} \rightarrow y \iff [x \rightarrow y] \]
\[ x_{-\ldots}^{1} \leftarrow y \iff [x \leftrightarrow y] \]
\[ x_{-\ldots}^{1} \leftarrow y \iff 0 \]
1. Introduction
2. Problem Statement
3. Graphs and Dependencies
4. Encoding D-connection

5. Algorithm
6. Conclusion
When discovering a network of 10 variables

- \((10\times9 + 10\times9/2) = 135\) possible edges
- \(2^{135} \approx 10^{40}\) different graphs

For a data set:

- \(2^{10} = 1024\) different conditioning sets
- Longest d-connecting path that needs to be considered is \(l_{\text{max}} = 16\) edges
- \((10\times10 + 10\times10/2 + 10\times10/2) \times 1024 \times 16 = 4\,915\,200\) path variables
- Gigabytes of CNF formulas.
Our SAT-based approach

1. Run conditional independence tests on the data set(s).
2. Encode the dependence and independence relations into the working formula $F$.
3. Determine the backbone of $F$ for the graph properties common to all graphs consistent with input.

HEURISTIC PRUNING OF UNNECESSARY TESTS
- Build over MiniSAT 2.2. Code is available.
- 8-12 variables, depending on the settings.
Practical Details (3)

- How to handle erroneous constraints?
  - MaxSAT?

- How to achieve better scalability?
  - Bottle neck: Calls to SAT-solver with this many Boolean variables and CNF-formulas.
  - Other types of encodings?
  - More efficient pruning of unnecessary tests?

- How to get both?
1. Introduction
2. Problem Statement
3. Graphs and Dependencies
4. Encoding D-connection
5. Algorithm

6. Conclusion
Conclusion

- New application area for SAT technology: constraint-based causal discovery
- SAT-solving to allow for a very general learning setting: cycles, latent variables, several data sets with manipulations
- Encoding, Algorithm exploiting incremental backbone computation
- How both to scale up and handle erroneous constraints?
1. Introduction
2. Problem Statement
3. Graphs and Dependencies
4. Encoding D-connection
5. Algorithm
6. Conclusion

7. EXTRA SLIDES
Proteins affect concentrations of other proteins.
Why Causal Models?

- Wouldn’t it be enough to learn the probability distribution over the variables? \[ P(x, y, z, w) \]
  - “How do x and w change when we observe different values of y?”
- Deeper, causal understanding allows us to predict given manipulations.
  - “How do x and w change when we manipulate y to different values?”
  - x is unaffected to manipulations of its effect y.
  - Manipulations of y change its effect w.