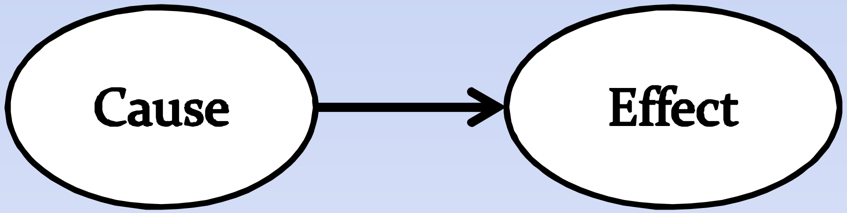
Discovering Cyclic Causal Models with Latent Variables: A General SAT-Based Procedure

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



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

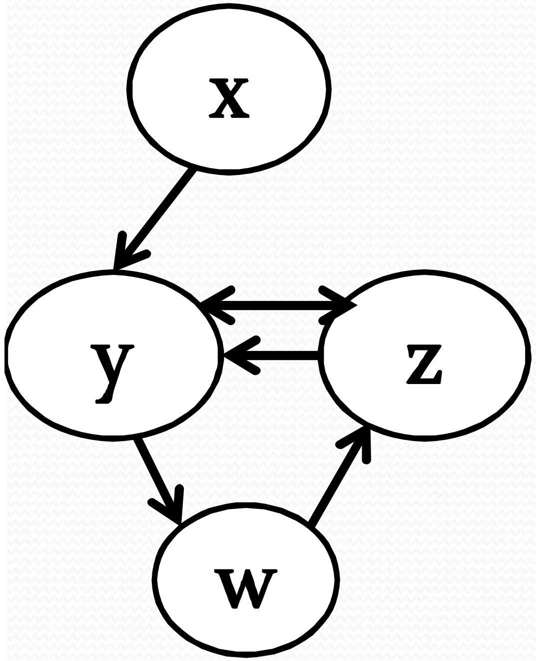
6. Conclusion

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

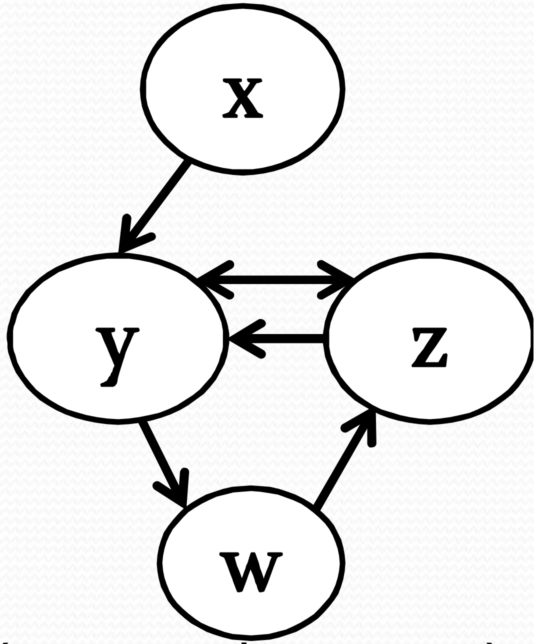
WORLD



DATA

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
...	...	...	...

CAUSAL STRUCTURE

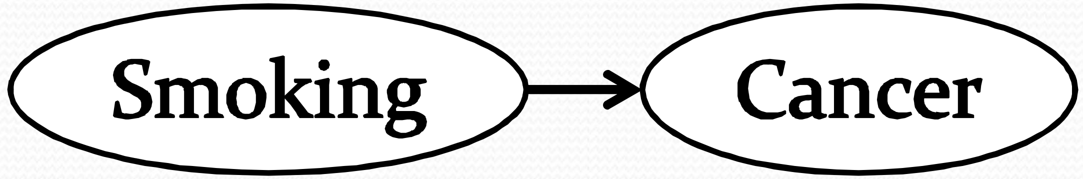


(e.g. Causal Bayes Net)

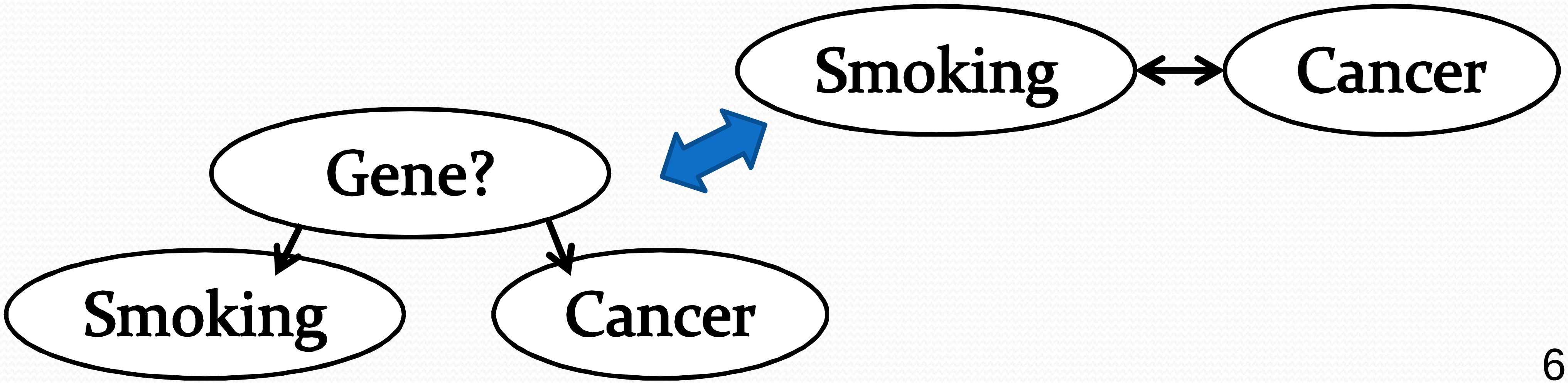
- Nodes represent random variables for measurements.
- Directed edges represents direct causal relationships.
- Bidirected arcs represent unobserved common causes.
- For example: different measurements of blood, life habits and acquired deceases.

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

CAUSAL STRUCTURE

(IN)DEPENDENCE RELATIONS

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
...	...	...	...

$x \not\perp w$

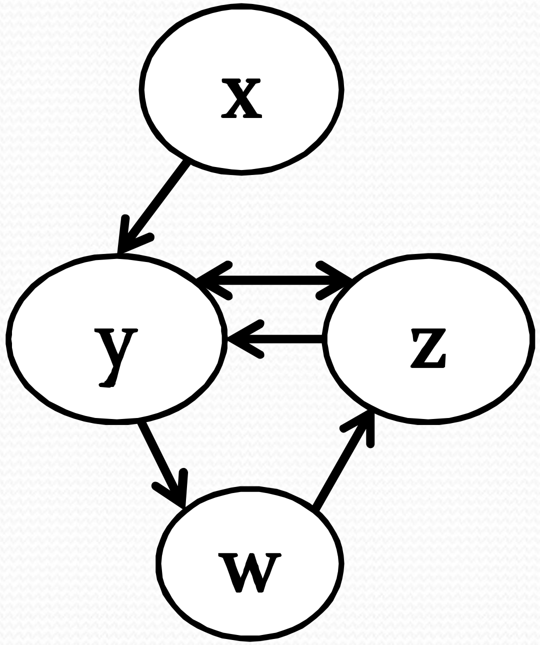
$x \perp w \mid y$

$x \not\perp w \mid y, z$

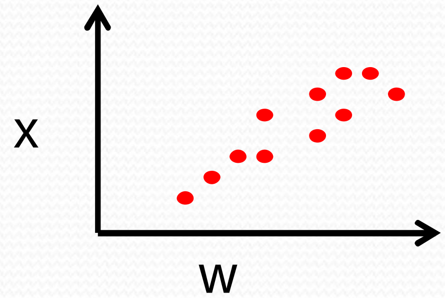
$z \perp x$

...

(cmp. orthogonal vectors)



- 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.



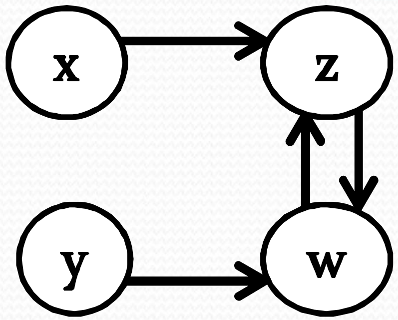
- Motivation: Complete generality of causal relations (continuous, discrete, nonlinear)

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:



- INPUT: (In)dependence relations tested from data....

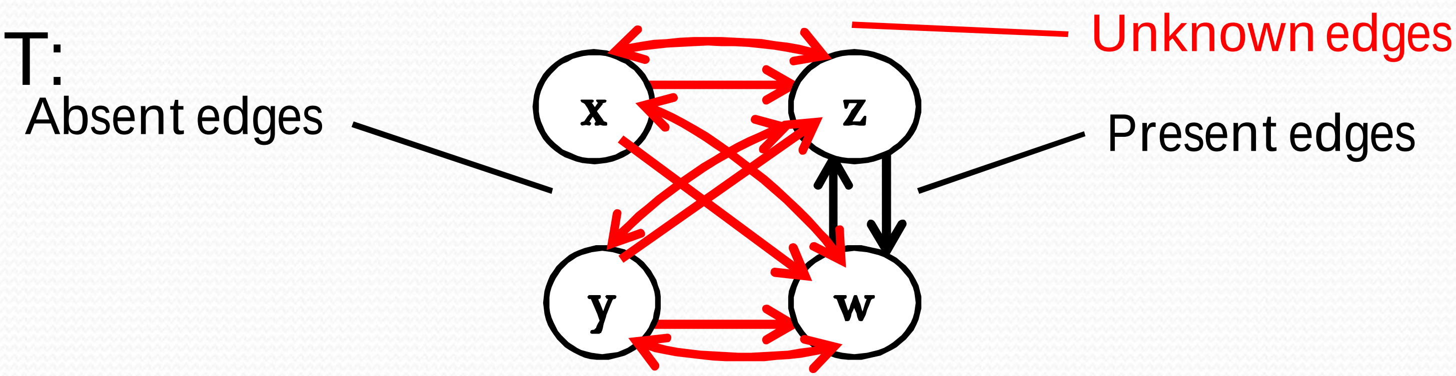
$x \not\perp z$

$x \perp y \mid z, w$

$x \not\perp w \mid z$

...

- OUTPUT:



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).

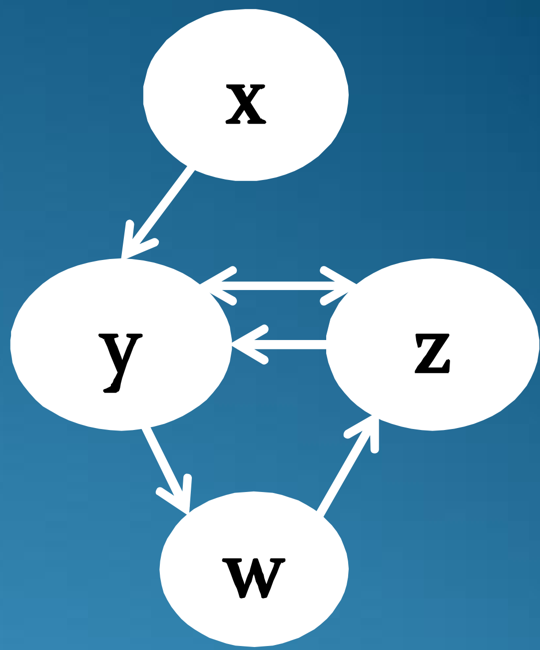
$x \not\perp w$

$x \perp w \mid y$

$x \not\perp w \mid y, z$

$z \perp x$

…



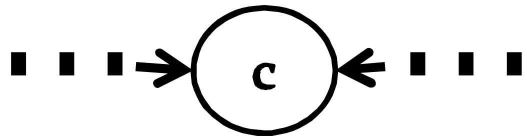
1. Introduction
2. Problem Statement

3. Graphs and Dependencies

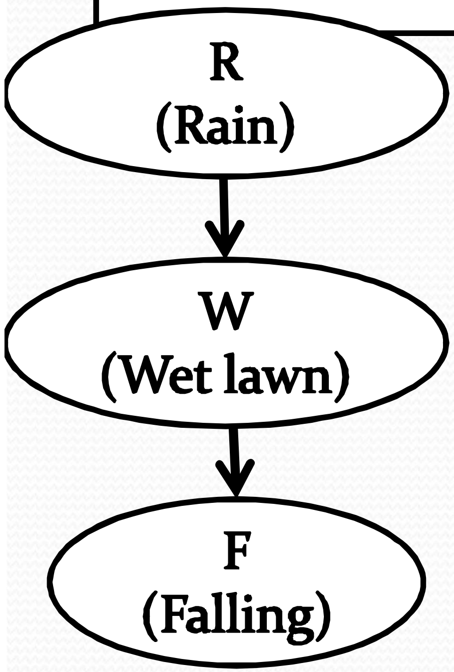
4. Encoding D-connection
5. Algorithm
6. Conclusion

D-connection (1)

- 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 path such that
  - Every collider node c (=node connected with heads) on the path is in C.



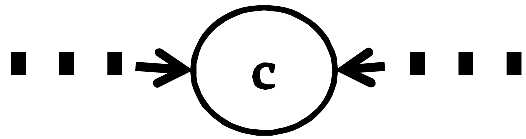
  - Other nodes on the path are not in C.



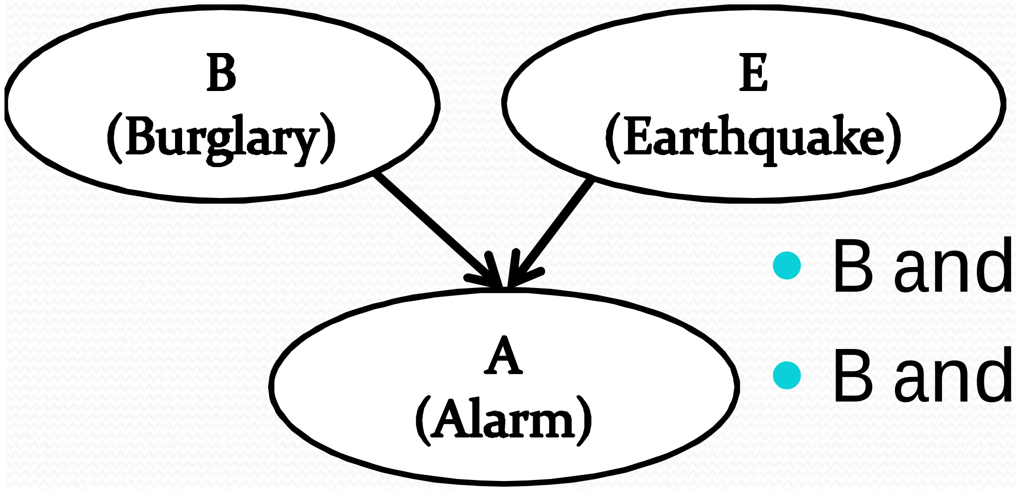
- R and W dependent given C={}? YES.
- R and F dependent given C={}? YES.
- R and F dependent given C={W}? NO.

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 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.

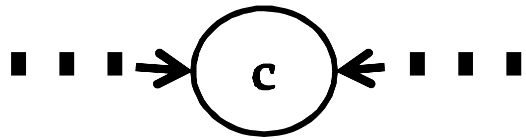
1. Introduction
2. Problem Statement
3. Graphs and Dependencies

4. Encoding D-connection

5. Algorithm
6. Conclusion

Encoding D-connection in Prop. Logic

- Random variables x and y are dependent given C, $x \not\perp\!\!\!\perp y | C$, 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 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.

Encoding D-connection in Prop. Logic

Dependence:

$[u \not\perp v \,|\, \mathbf{C}]$

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

Boolean variables TRUE iff the edge is present in the solution.

Graph:

$[x \rightarrow y]$ $[x \leftrightarrow y]$

Encoding D-connection in Prop. Logic

Dependence:

$$[u \not\perp v \,|\, \mathbf{C}]$$

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.

$$[x \overset{l}{\underset{\mathbf{C}}{-\cdots>}} y]$$

$$[x \overset{l}{\underset{\mathbf{C}}{-\cdots-}} y]$$

$$[x \overset{l}{\underset{\mathbf{C}}{<\cdots>}} y]$$

Graph:

$$[x \rightarrow y]$$

$$[x \leftrightarrow y]$$

Encoding D-separation in Prop. Logic

Dependence:

$$[u \not\perp v \,|\, \mathbf{C}] \Leftrightarrow \bigvee_{l=1}^{l_{\max}} \left([u \overset{l}{\underset{\mathbf{C}}{-\cdots>}} v] \vee [v \overset{l}{\underset{\mathbf{C}}{-\cdots>}} u] \vee [u \overset{l}{\underset{\mathbf{C}}{<\cdots>}} v] \vee [u \overset{l}{\underset{\mathbf{C}}{-\cdots-}} v] \right)$$

Paths:

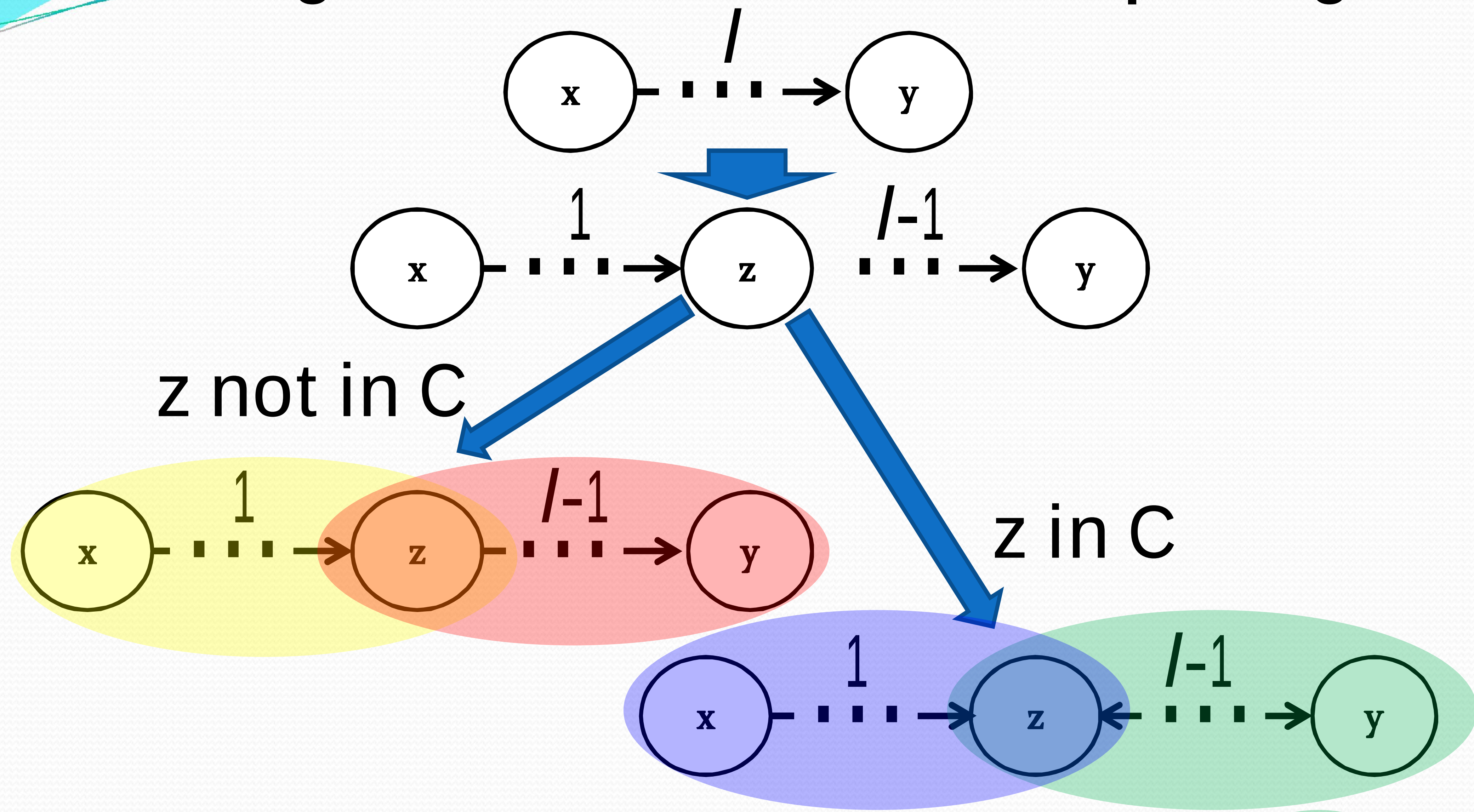
$$[x \overset{l}{\underset{\mathbf{C}}{-\cdots>}} y]$$

$$[x \overset{l}{\underset{\mathbf{C}}{-\cdots-}} y] \qquad [x \overset{l}{\underset{\mathbf{C}}{<\cdots>}} y]$$

Graph:

$$[x \overset{1}{\underset{\mathbf{C}}{-\cdots>}} y] \Leftrightarrow [x \rightarrow y] \qquad [x \overset{1}{\underset{\mathbf{C}}{<\cdots>}} y] \Leftrightarrow [x \leftrightarrow y] \qquad [x \overset{1}{\underset{\mathbf{C}}{-\cdots-}} y] \Leftrightarrow 0$$

Encoding D-connection in Prop. Logic



$$[x \underset{\mathbf{C}}{\overset{l}{- \cdots >}} y] \quad \Leftrightarrow \quad \bigvee_{z \notin \mathbf{C}} \left([x \underset{\mathbf{C}}{\overset{1}{- \cdots >}} z] \wedge [z \underset{\mathbf{C}}{\overset{l-1}{- \cdots >}} y] \right) \vee \bigvee_{z \in \mathbf{C}} \left([x \underset{\mathbf{C}}{\overset{1}{- \cdots >}} z] \wedge [z \underset{\mathbf{C}}{\overset{l-1}{< \cdots >}} y] \right)$$

Encoding D-connection in Prop. Logic

Dependence:

$$[u \not\perp v \,|\, \mathbf{C}] \quad \Leftrightarrow \quad \bigvee_{l=1}^{l_{\max}} \left([u \overset{l}{\underset{\mathbf{C}}{-\cdots>}} v] \vee [v \overset{l}{\underset{\mathbf{C}}{-\cdots>}} u] \vee [u \overset{l}{\underset{\mathbf{C}}{<\cdots>}} v] \vee [u \overset{l}{\underset{\mathbf{C}}{-\cdots-}} v] \right)$$

Paths:

$$[x \overset{l}{\underset{\mathbf{C}}{-\cdots>}} y] \quad \Leftrightarrow \quad \bigvee_{z \notin \mathbf{C}} \left([x \overset{1}{\underset{\mathbf{C}}{-\cdots>}} z] \wedge [z \overset{l-1}{\underset{\mathbf{C}}{-\cdots>}} y] \right) \vee \bigvee_{z \in \mathbf{C}} \left([x \overset{1}{\underset{\mathbf{C}}{-\cdots>}} z] \wedge [z \overset{l-1}{\underset{\mathbf{C}}{<\cdots>}} y] \right)$$

$$[x \overset{l}{\underset{\mathbf{C}}{-\cdots-}} y] \quad \Leftrightarrow \quad \ldots$$

$$[x \overset{l}{\underset{\mathbf{C}}{<\cdots>}} y] \quad \Leftrightarrow \quad \ldots$$

Graph:

$$[x \overset{1}{\underset{\mathbf{C}}{-\cdots>}} y] \quad \Leftrightarrow \quad [x \to y]$$

$$[x \overset{1}{\underset{\mathbf{C}}{<\cdots>}} y] \quad \Leftrightarrow \quad [x \leftrightarrow y]$$

$$[x \overset{1}{\underset{\mathbf{C}}{-\cdots-}} y] \quad \Leftrightarrow \quad 0$$

1. Introduction
2. Problem Statement
3. Graphs and Dependencies
4. Encoding D-connection
5. Algorithm
6. Conclusion

Practical Details (1)

When discovering a network of 10 variables

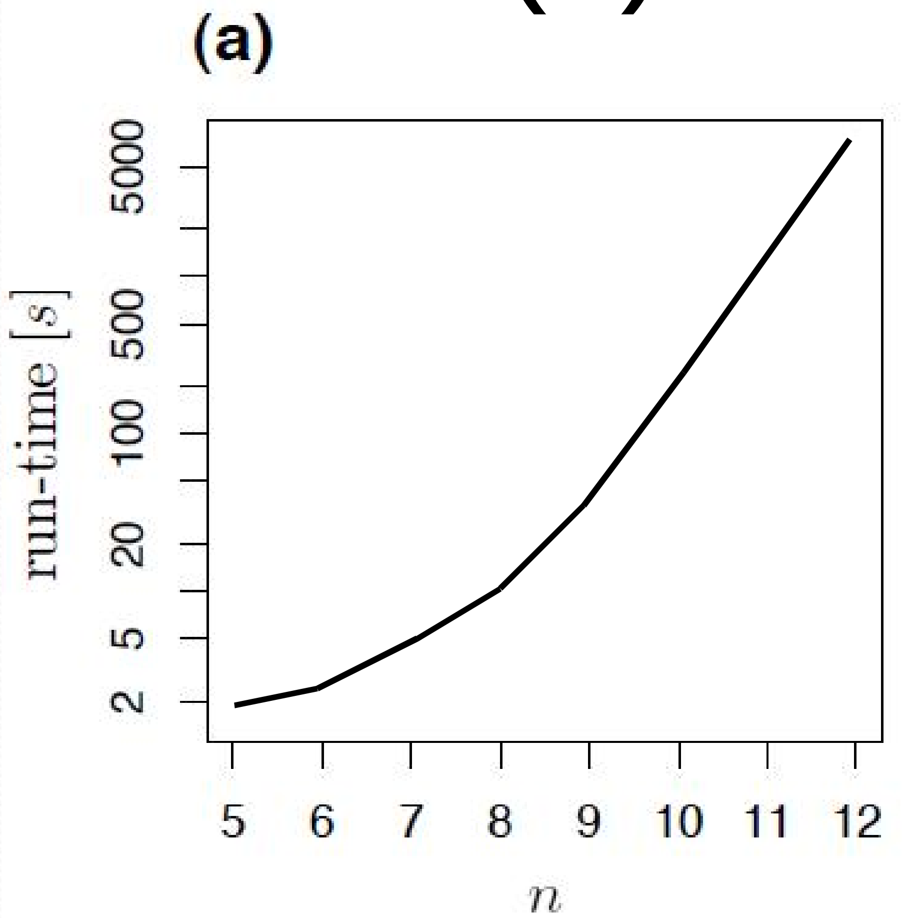
- (10*9+10*9/2)=135 possible edges
- 2^135~ 10^40 different graphs
- For a data set:
  - 2^10=1024 different conditioning sets
  - Longest d-connecting path that needs to be considered is l_{max}=16 edges
  - (10*10+10*10/2+10*10/2)*1024*16= 4 915 200 path variables
  - Gigabytes of CNF formulas.

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

Practical Details (2)



- Build over MiniSAT 2.2. Code is available.
- 8-12 variables, dependending on the settings.

Practical Details (3)

- How to handle errorneous 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

x	y	z	w
0.4	0.56	4	120
0.5	0.23	100	130
...	...	...	...

(IN)DEPENDENCE RELATIONS

$x \not\perp w$

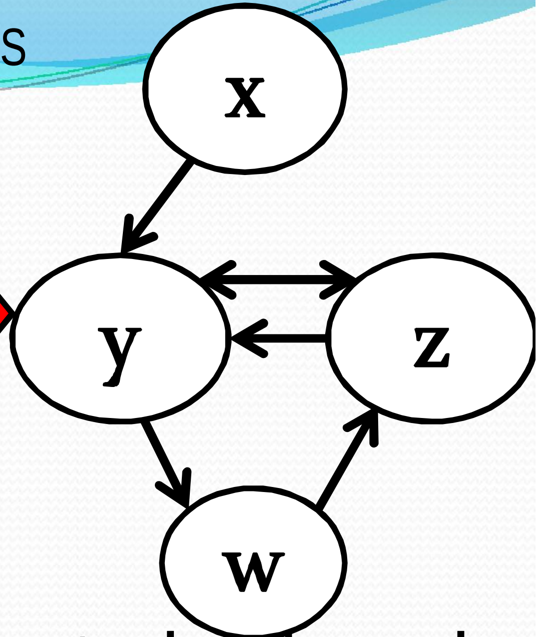
$x \perp w \mid y$

$x \not\perp w \mid y, z$

$z \perp x$

...

CAUSAL STRUCTURE



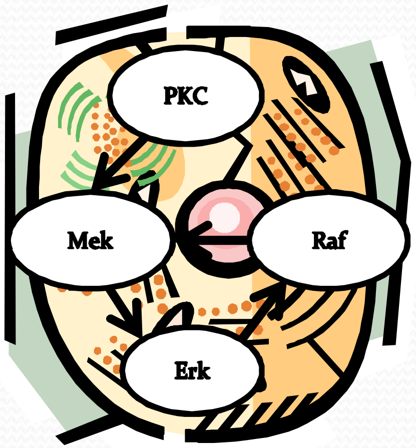
- New application area for SAT technology: constraint-based causal discovery
- SAT-solving to allow for a very general learning setting: cycles, latent variables, several data sets with manipulations
- Encoding, Algorithm exploiting incremental backbone computation
- How both to scale up and handle errorneous constraints?

1. Introduction
2. Problem Statement
3. Graphs and Dependencies
4. Encoding D-connection
5. Algorithm
6. Conclusion

7. EXTRA SLIDES

Real Example

WORLD

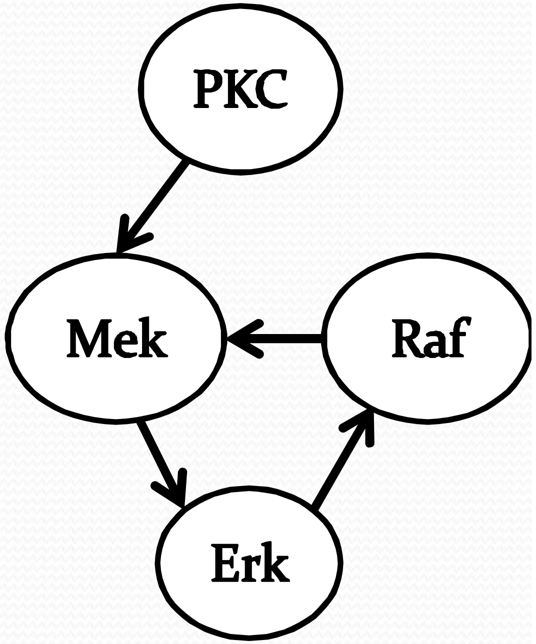


Sachs et al. (Science 2005)

DATA

	Raf	Mek	Erk	PKC
Cell 1	0.4	0.56	4	120
Cell 2	0.5	0.23	100	130
Cell 3	0.1	0.01	34	123
Cell 4	0.23	0.03	52	23
...	...	...	...	...

CAUSAL MODEL/STRUCTURE



- 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.