Local Structure and BE extensions

COMPSCI 276, Spring 2017

Set 5b: Rina Dechter

(Reading: Darwiche chapter 5, dechter chapter 4)

Outline

- Special representations of CPTs
- Bucket Elimination:
  - Finding induced-width
  - Bucket elimination over mixed networks

Outline

- Bayesian networks and queries
- Building Bayesian Networks
- Special representations of CPTs
  - Causal Independence (e.g., Noisy OR)
  - Context Specific Independence
  - Determinism
  - Mixed Networks

Dealing with Large CPTs

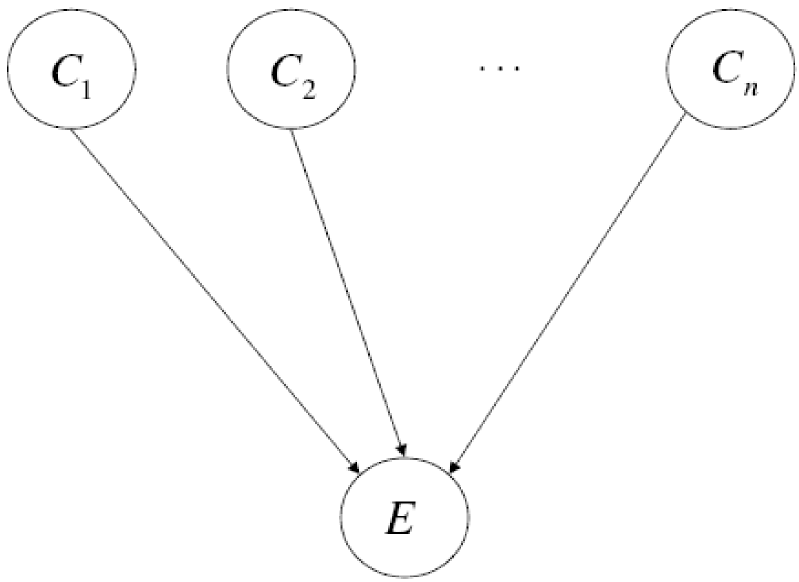
The size of a CPT

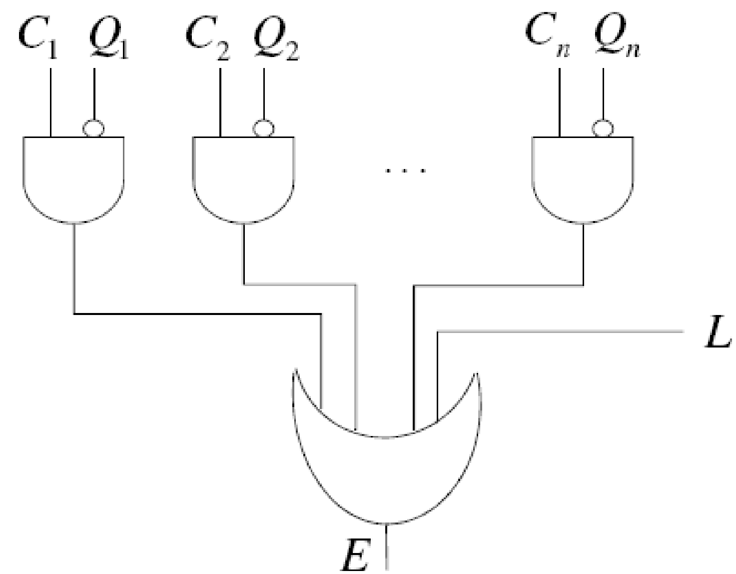
for binary variable E with binary parents $C_1, \ldots, C_n$

Number of Parents: n	Parameter Count: 2^n
2	4
3	8
6	64
10	1024
20	$1,048,576$
30	$1,073,741,824$

Micro Model

Think about headache and 10 different conditions that may cause it.





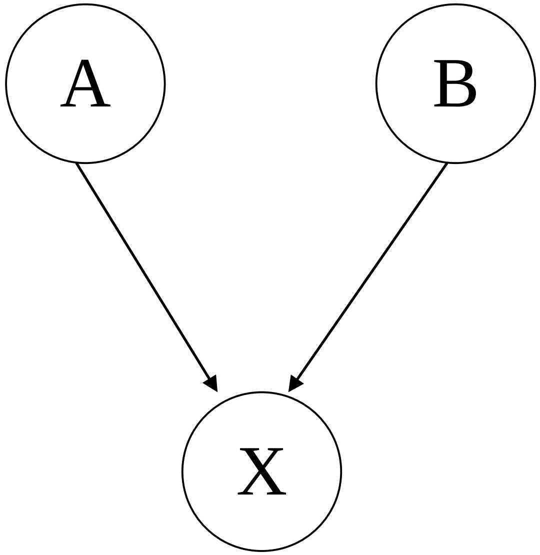
A noisy-or circuit

A micro model

details the relationship between a variable E and its parents $C_1, \ldots, C_n$.

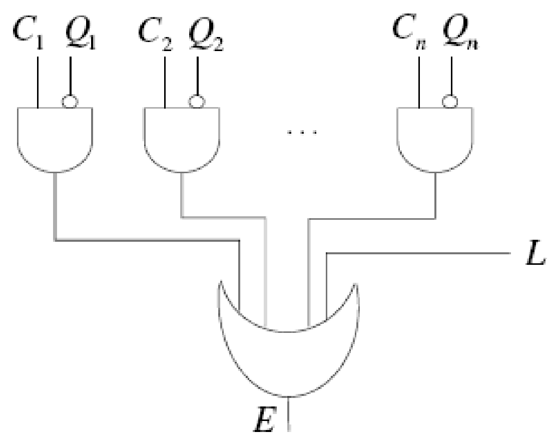
We wish to specify cpt with less parameters

Binary OR



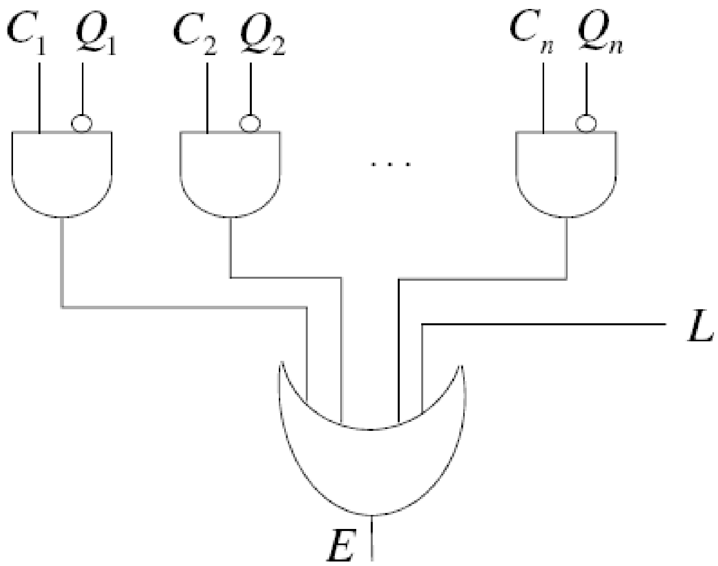
A	B	P(X=0|A,B)	P(X=1|A,B)
0	0	1	0
0	1	0	1
1	0	0	1
1	1	0	1

Noisy-or Model



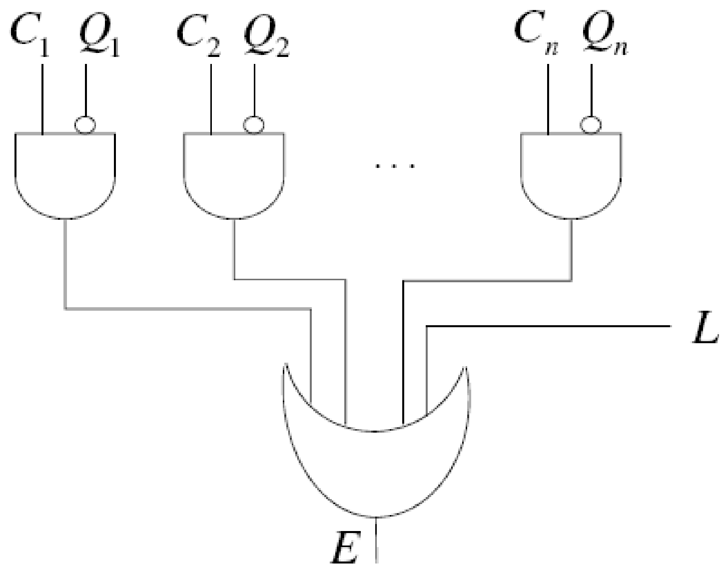
- Cause C_i is capable of establishing effect E, except under some unusual circumstances summarized by suppressor Q_i.
- When suppressor Q_i is active, C_i is no longer able to establish E.
- The leak variable L represents all other causes of E which were not modeled explicitly.
- When none of the causes C_i are active, the effect E may still be established by the leak variable L.

Noisy–or Model



The noisy-or model requires $n + 1$ parameters.

Noisy–or Model



The noisy-or model requires $n + 1$ parameters.

To model the relationship between headache and ten different conditions

- $\theta_{q_i} = \Pr(Q_i = \text{active})$: probability that suppressor of C_i is active.
- $\theta_l = \Pr(L = \text{active})$: probability that leak is active.

Noisy-or Model

- Let I_α be the indices of causes that are active in α.

Noisy-or Model

- Let I_α be the indices of causes that are active in α.
- If

  α: $C_1 =$ active, $C_2 =$ active, $C_3 =$ passive, $C_4 =$ passive, $C_5 =$ active,

  then $I_\alpha = \{1, 2, 5\}$.

Noisy-or Model

- Let I_α be the indices of causes that are active in α.
- If

  α: $C_1 =$ active, $C_2 =$ active, $C_3 =$ passive, $C_4 =$ passive, $C_5 =$ active,

  then $I_\alpha = \{1, 2, 5\}$.
- We then have

$$
\begin{aligned}
\Pr(E = \text{passive} \mid \alpha) &= (1 - \theta_l) \prod_{i \in I_\alpha} \theta_{q_i} \\
\Pr(E = \text{active} \mid \alpha) &= 1 - \Pr(E = \text{passive} \mid \alpha).
\end{aligned}
$$

Noisy-or Model

- Let I_α be the indices of causes that are active in α.
- If

  α: $C_1 = \text{active}$, $C_2 = \text{active}$, $C_3 = \text{passive}$, $C_4 = \text{passive}$, $C_5 = \text{active}$,

  then $I_\alpha = \{1, 2, 5\}$.
- We then have

$$
\begin{aligned}
\Pr(E = \text{passive} \mid \alpha) &= (1 - \theta_l) \prod_{i \in I_\alpha} \theta_{q_i} \\
\Pr(E = \text{active} \mid \alpha) &= 1 - \Pr(E = \text{passive} \mid \alpha).
\end{aligned}
$$

The full CPT for variable E, with its 2^n parameters, can be induced from the $n + 1$ parameters of the noisy-or model.

Noisy-or Model

Example

Sore throat (S) has three causes: cold (C), flu (F), tonsillitis (T).

Noisy-or Model

Example

Sore throat (S) has three causes: cold (C), flu (F), tonsillitis (T).

If we assume that S is related to its causes by a noisy-or model

we can then specify the CPT for S by the following four probabilities:

- The suppressor probability for cold, say .15
- The suppressor probability for flu, say, .01
- The suppressor probability for tonsillitis, say .05
- The leak probability, say .02

Noisy-or Model

Example

Sore throat (S) has three causes: cold (C), flu (F), tonsillitis (T).

Noisy-or Model

Example

Sore throat (S) has three causes: cold (C), flu (F), tonsillitis (T).

The CPT for sore throat is then determined completely as follows:

C	F	T	S	$\theta_{s\|c,f,t}$	
true	true	true	true	0.9999265	$1-(1-.02)(.15)(.01)(.05)$
true	true	false	true	0.99853	$1-(1-.02)(.15)(.01)$
true	false	true	true	0.99265	$1-(1-.02)(.15)(.05)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	
false	false	false	true	.02	$1-(1-.02)$

Noisy/OR CPDs

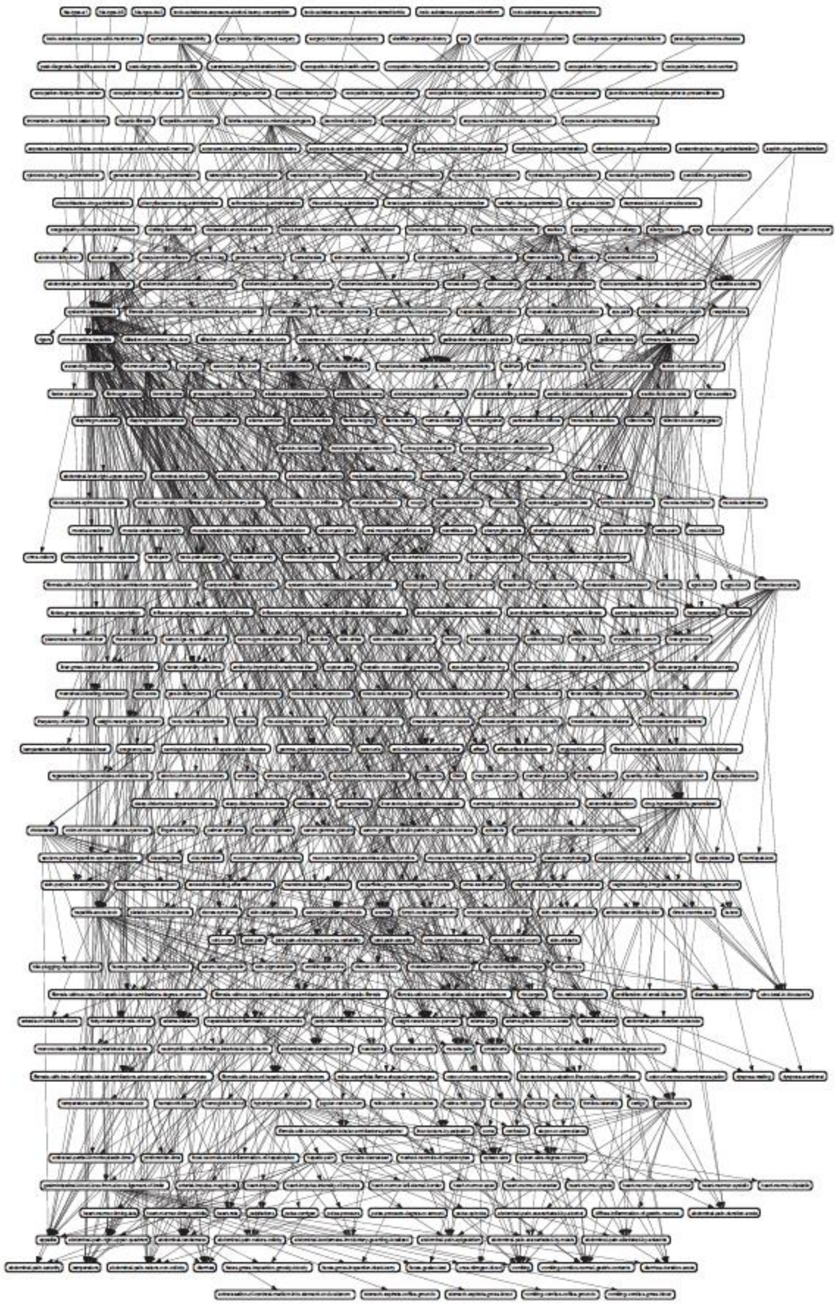
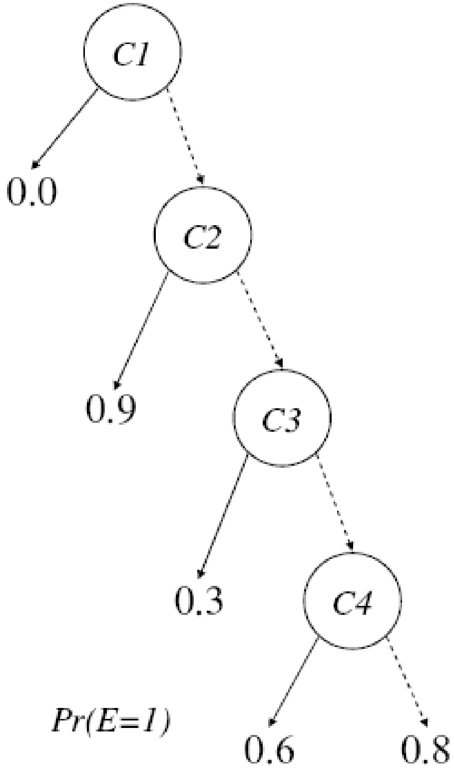


Figure 11: the CPCS network for diagnosis of internal diseases. The network contains 448 nodes, 906 links.

Decision Trees

C1	*C2*	*C3*	*C4*	*Pr(E=1)*
1	1	1	1	0.0
1	1	1	0	0.0
1	1	0	1	0.0
1	1	0	0	0.0
1	0	1	1	0.0
1	0	1	0	0.0
1	0	0	1	0.0
1	0	0	0	0.0
0	1	1	1	0.9
0	1	1	0	0.9
0	1	0	1	0.9
0	1	0	0	0.9
0	0	1	1	0.3
0	0	1	0	0.3
0	0	0	1	0.6
0	0	0	0	0.8



If-Then Rules

A CPT for variable E can be represented using a set of if-then rules of the form

If α_i then $\Pr(e) = p_i$, for each value e of variable E, where α_i is a propositional sentence constructed using the parents of variable E.

If-Then Rules

A CPT for variable E can be represented using a set of if-then rules of the form

If α_i then $\Pr(e) = p_i$, for each value e of variable E, where α_i is a propositional sentence constructed using the parents of variable E.

If $C_1 = 1$	then	$\Pr(E = 1) = 0.0$
If $C_1 = 0 \wedge C_2 = 1$	then	$\Pr(E = 1) = 0.9$
If $C_1 = 0 \wedge C_2 = 0 \wedge C_3 = 1$	then	$\Pr(E = 1) = 0.3$
If $C_1 = 0 \wedge C_2 = 0 \wedge C_3 = 0 \wedge C_4 = 1$	then	$\Pr(E = 1) = 0.6$
If $C_1 = 0 \wedge C_2 = 0 \wedge C_3 = 0 \wedge C_4 = 0$	then	$\Pr(E = 1) = 0.8$

If-Then Rules

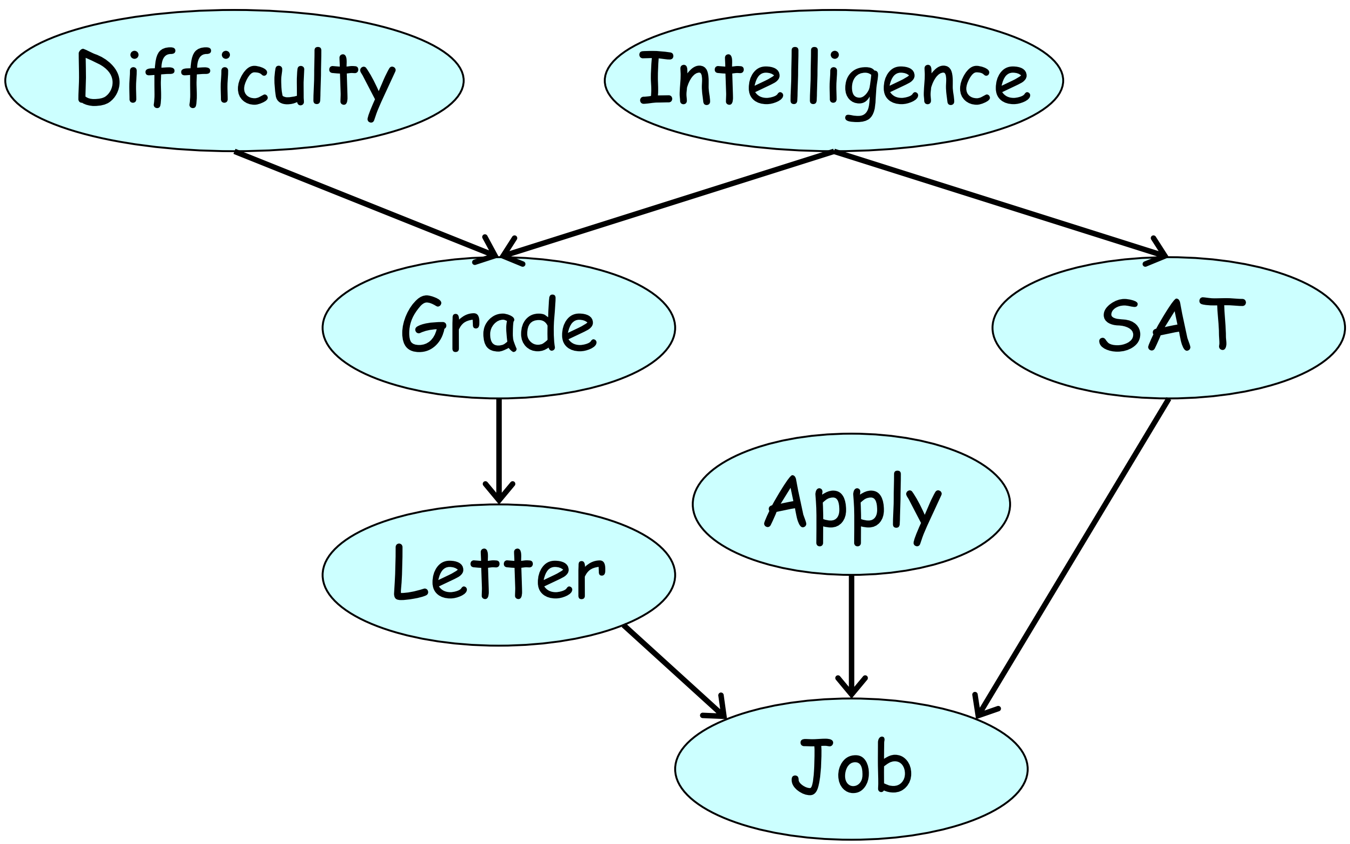
A CPT for variable E can be represented using a set of if-then rules of the form

If α_i then $\Pr(e) = p_i$, for each value e of variable E, where α_i is a propositional sentence constructed using the parents of variable E.

For the rule-based representation to be complete and consistent

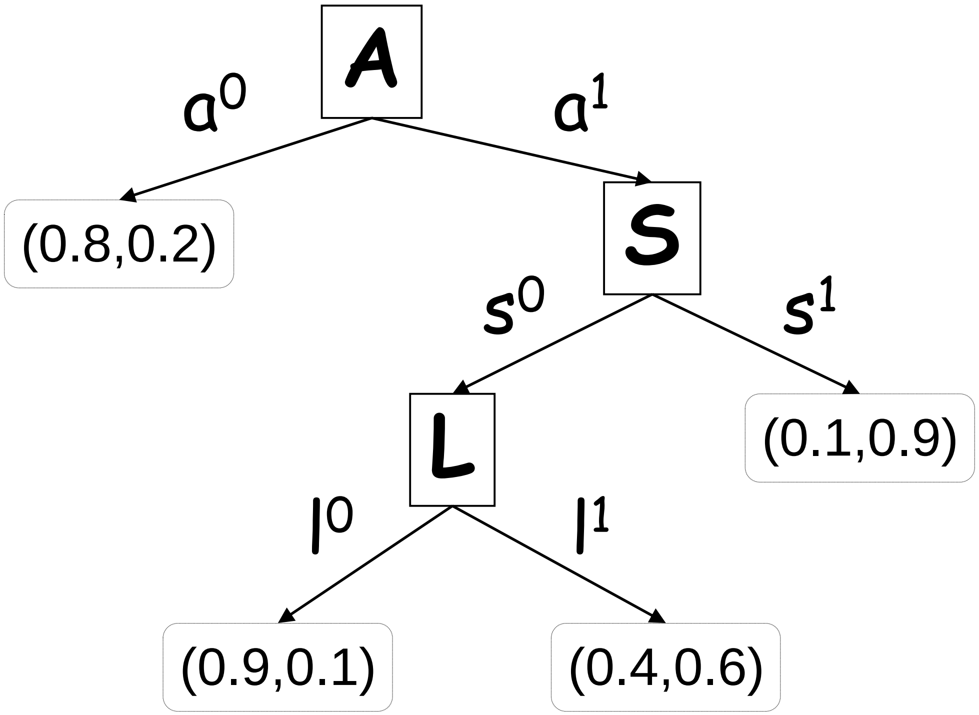
- The premises α_i must be mutually exclusive. That is, $\alpha_i \wedge \alpha_j$ is inconsistent for $i \neq j$. This ensures that the rules will not conflict with each other.
- The premises α_i must be exhaustive. That is, $\bigvee_i \alpha_i$ must be valid. This ensures that every CPT parameter $\theta_{e|\ldots}$ is implied by the rules.

A student's example



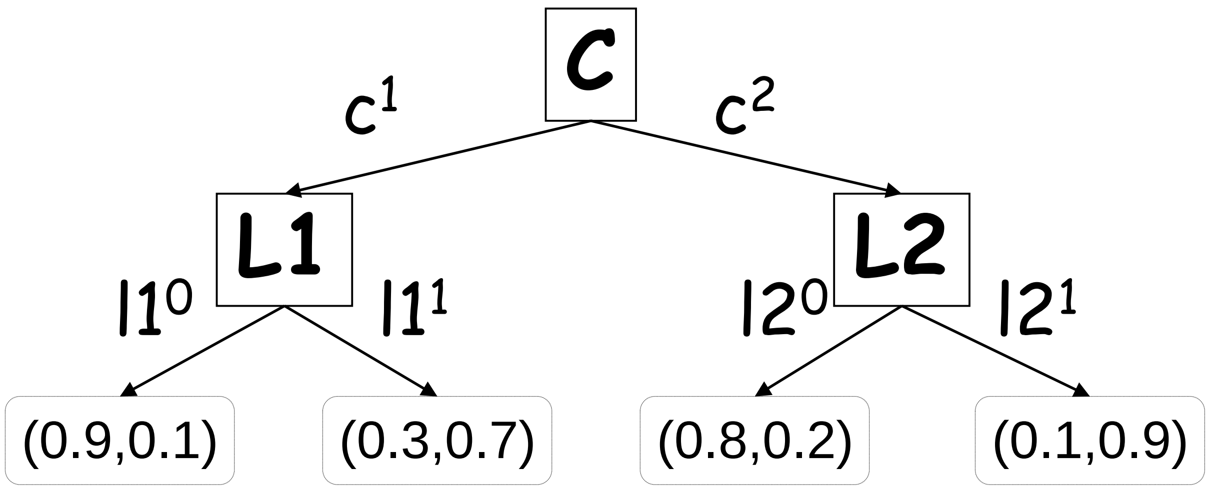
Tree CPD

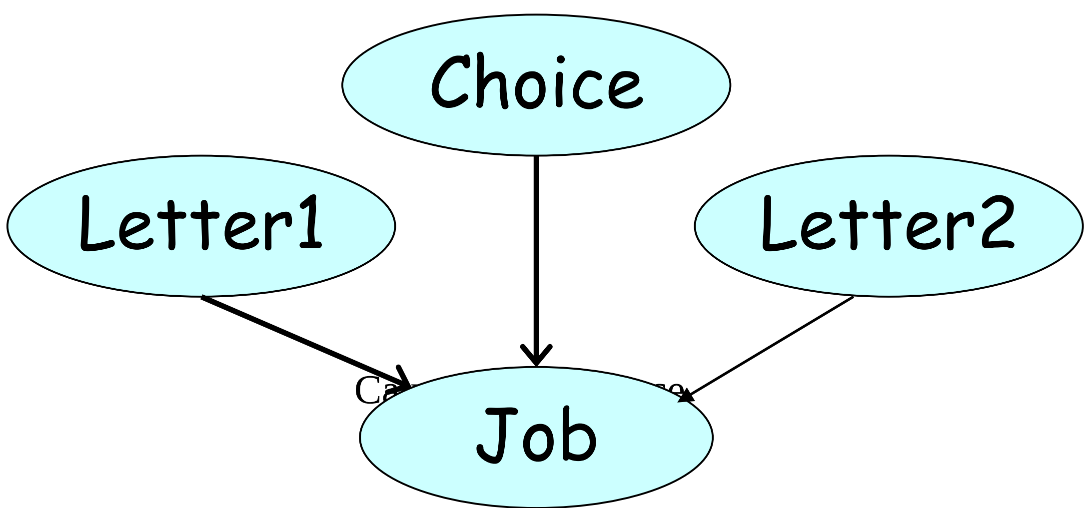
If the student does not *A*pply, *S*AT and *L* are irrelevant



Tree-CPD for job

Captures irrelevant variables

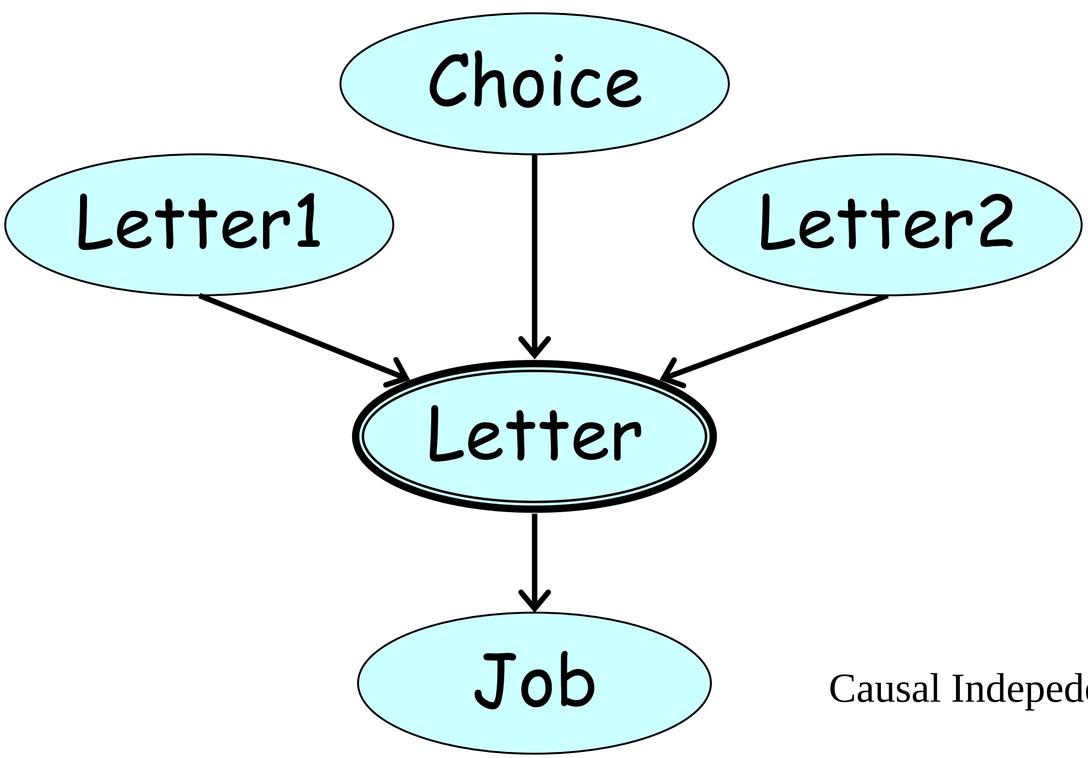




Multiplexer CPD

A CPD P(Y|A,Z1,Z2,…,Zk) is a multiplexer iff Val(A)=1,2,…k, and

P(Y|A,Z1,…Zk)=Z_a



Mixture of trees

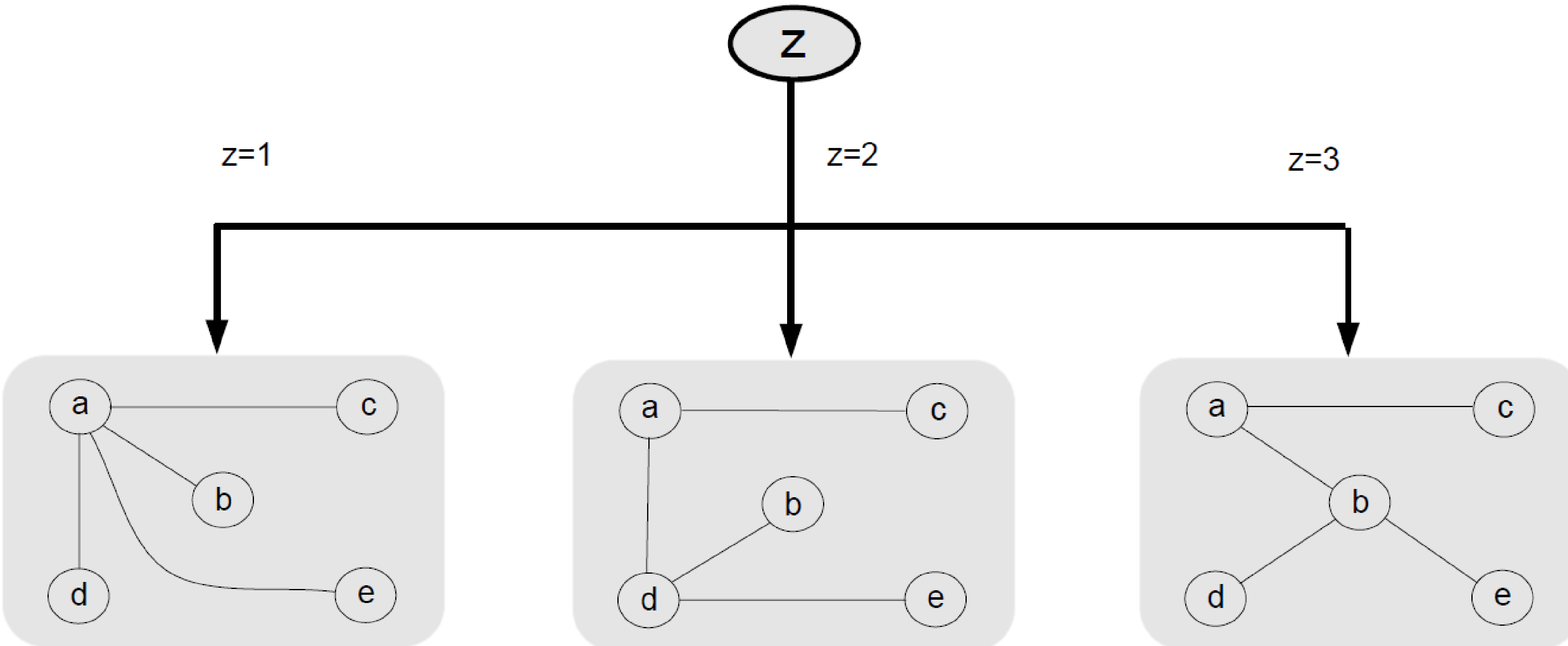


Figure 1: A mixture of trees over a domain consisting of random variables $V = \{a, b, c, d, e\}$, where z is a hidden *choice variable*. Conditional on the value of z, the dependency structure is a tree. A detailed presentation of the mixture-of-trees model is provided in Section 3.

Meila and Jordan, 2000

Mixture model with shared structure

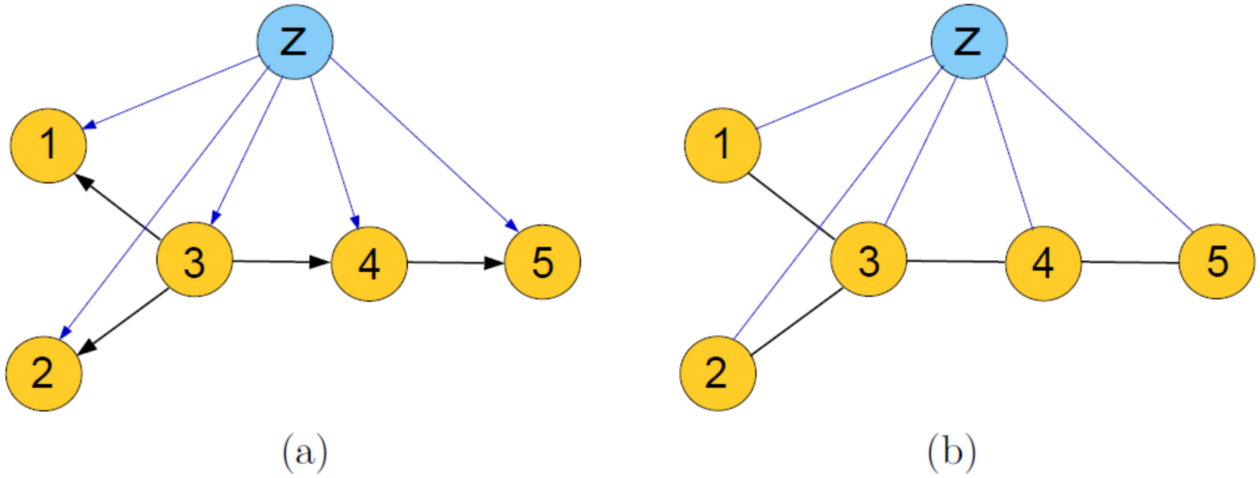


Figure 4: A mixture of trees with shared structure (MTSS) represented as a Bayes net (a) and as a Markov random field (b).

Meila and Jordan, 2000

Deterministic CPTs

Can we use hidden variables?

A	X	C	$\theta_{c\|a,x}$
high	ok	high	0
low	ok	high	1
high	stuckat0	high	0
low	stuckat0	high	0
high	stuckat1	high	1
low	stuckat1	high	1

We can represent this CPT as follows

$$(X = \text{ok} \wedge A = \text{high}) \vee X = \text{stuckat0} \iff C = \text{low}$$
$$(X = \text{ok} \wedge A = \text{low}) \vee X = \text{stuckat1} \iff C = \text{high}$$

Mixed Networks
(Dechter 2013)

Augmenting Probabilistic networks with constraints because:

- Some information in the world is deterministic and undirected ($X \neq Y$)
- Some queries are complex or evidence are complex (cnfs)

Queries are probabilistic queries

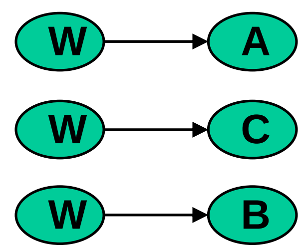
Probabilistic Reasoning

Party example: the weather effect

Alex is-likely-to-go in bad weather

Chris rarely-goes in bad weather

Becky is indifferent but unpredictable



P(A|W=bad)=.9

P(C|W=bad)=.1

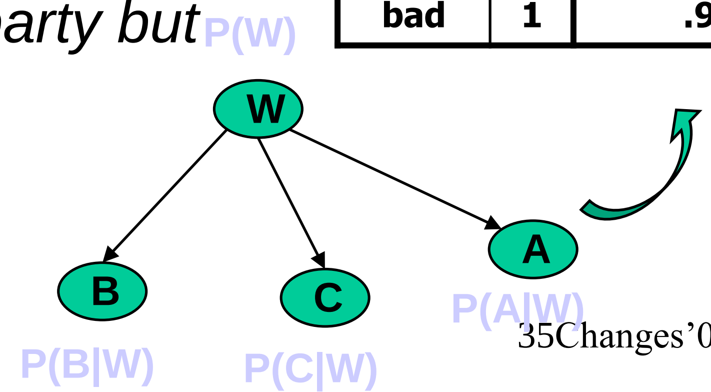
P(B|W=bad)=.5

W	A	P(A|W)
good	0	.01
good	1	.99
bad	0	.1
bad	1	.9

Questions:

Given bad weather, which group of individuals is most likely to show up at the party?

What is the probability that Chris goes to the party but Becky does not?

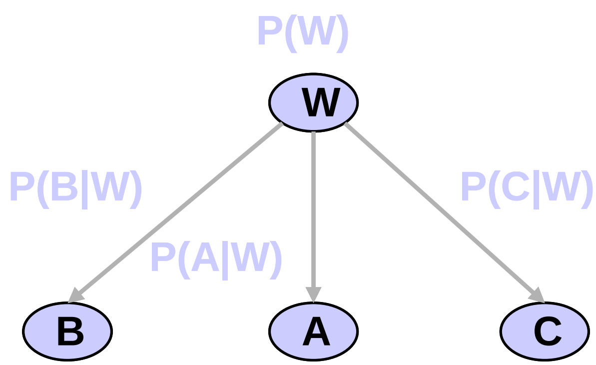


$P(W,A,C,B) = P(B|W) \cdot P(C|W) \cdot P(A|W) \cdot P(W)$

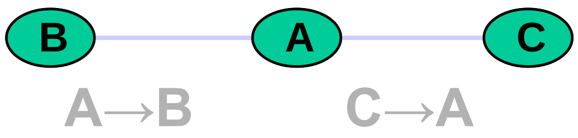
$P(A,C,B|W=bad) = 0.9 \cdot 0.1 \cdot 0.5$

Party Example Again

Bayes Network



Constraint Network



Semantics?

Algorithms?

Query:
Is it likely that Chris goes to the party if Becky does not but the weather is bad?

$$P(C, \neg B \mid w = bad, A \rightarrow B, C \rightarrow A)$$

Outline

- Special representations of CPTs
- Bucket Elimination:
  - Finding induced-width
  - Bucket elimination over mixed networks

Complexity of bucket elimination

Theorem

Given a belief network having n variables, observations e, the complexity of elim-mpe, elim-bel, elim-map along d, is time and space

O(nexp(w*+1)) and O(n exp(w*)), respectively

where $w*(d)$ is the induced width of the moral graph whose edges connecting evidence to earlier nodes, were deleted.

More accurately: O(r exp(w*(d)) where r is the number of cpts.
For Bayesian networks r=n. For Markov networks?

Finding Small Induced-Width
(Dechter 3.4-3.5)

NP-complete

A tree has induced-width of ?

Greedy algorithms:

- Min width
- Min induced-width
- Max-cardinality and chordal graphs
- Fill-in (thought as the best)
- See anytime min-width (Gogate and Dechter)

Type of graphs

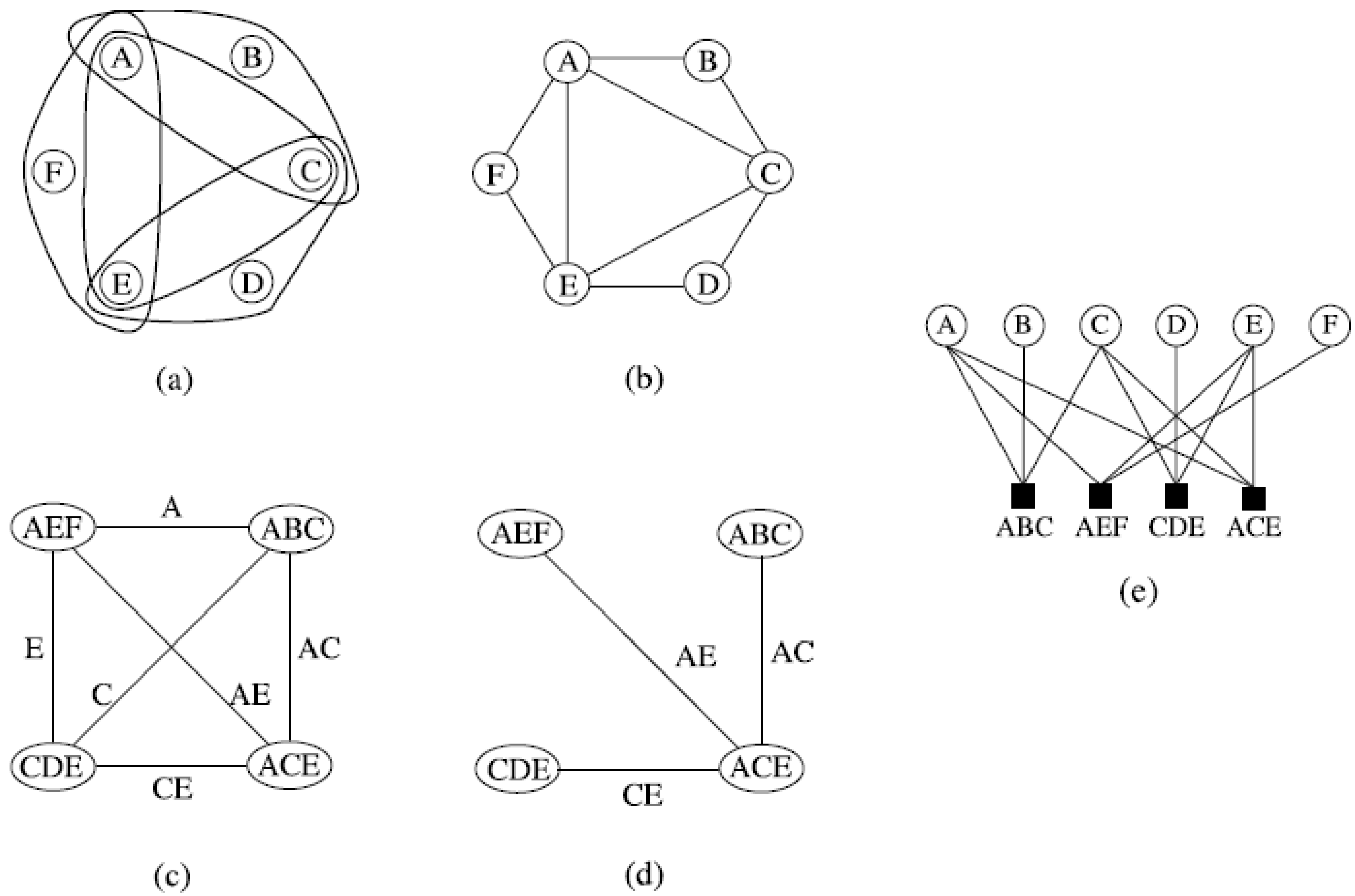


Figure 5.1: (a)Hyper, (b)Primal, (c)Dual and (d)Join-tree of a graphical model having scopes ABC, AEF, CDE and ACE. (e) the factor graph

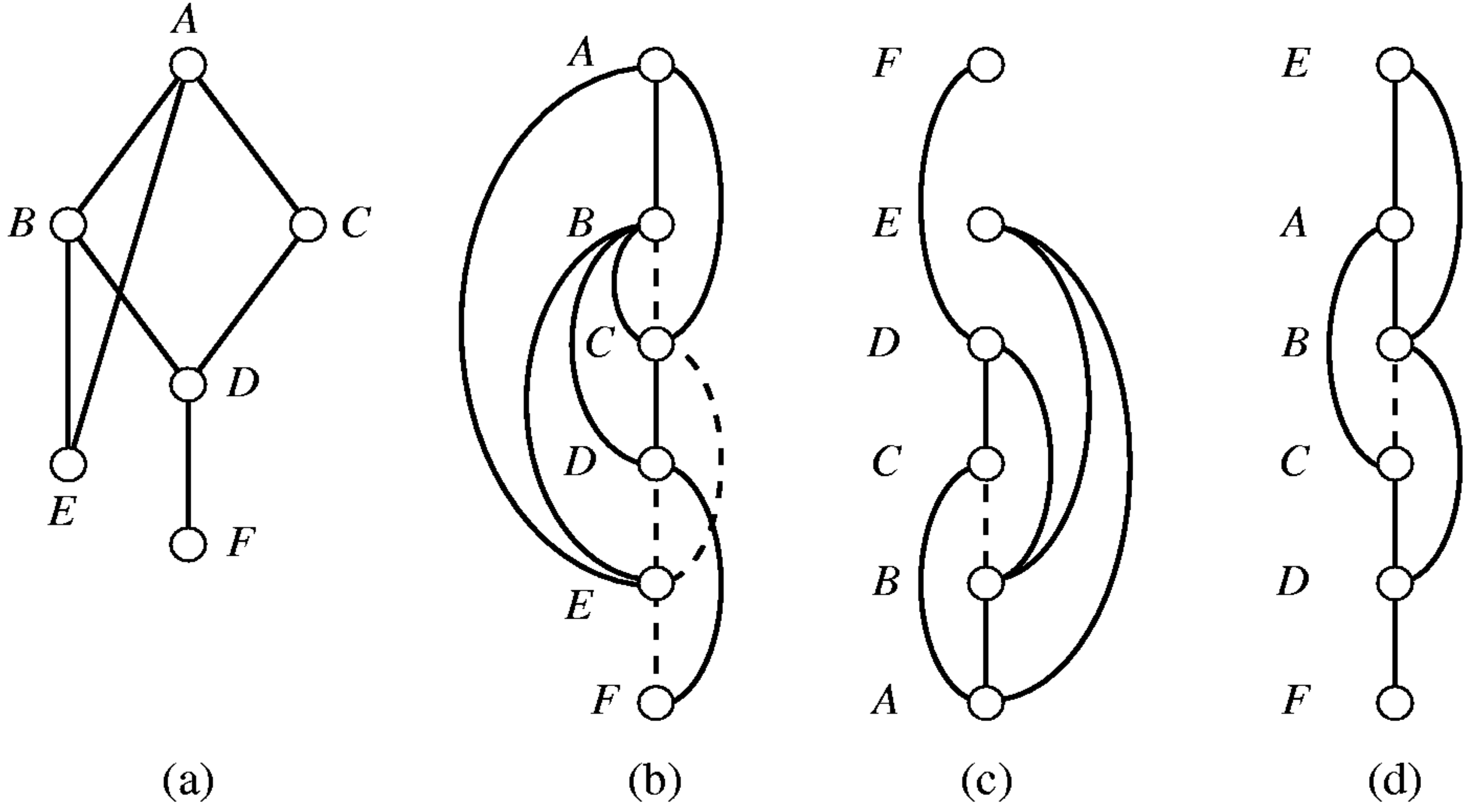
The induced width

Definition 5.2.1 (width) *Given an undirected graph $G = (V, E)$, an* ordered graph *is a pair (G, d), where $V = \{v_1, ..., v_n\}$ is the set of nodes, E is a set of arcs over V, and $d = (v_1, ..., v_n)$ is an ordering of the nodes. The nodes adjacent to v that precede it in the ordering are called its* parents. *The* width of a node *in an ordered graph is its number of parents. The* width of an ordering d *of G, denoted $w_d(G)$ (or w_d for short) is the maximum width over all nodes. The* width of a graph *is the minimum width over all the orderings of the graph.*

Definition 5.2.3 (induced width) *The* induced width of an ordered graph (G, d), *denoted ${w^*}_d$), is the width of the* induced ordered graph *along d obtained as follows: nodes are processed from last to first; when node v is processed, all its parents are connected. The* **induced width** *of a graph, denoted by w^*, is the minimal induced width over all its orderings. Formally*

$$w^*(G) = \min_{d \in orderings} {w^*}_d(G)$$

Different Induced-graphs



Min-Width Ordering

MIN-WIDTH (MW)

input: a graph $G = (V, E)$, $V = \{v_1, ..., v_n\}$

output: A min-width ordering of the nodes $d = (v_1, ..., v_n)$.

1. **for** $j = n$ to 1 by -1 do
2. $r \leftarrow$ a node in G with smallest degree.
3. put r in position j and $G \leftarrow G - r$.
   (Delete from V node r and from E all its adjacent edges)
4. **endfor**

Proposition: (Freuder 1982) algorithm min-width finds a min-width ordering of a graph. Complexity O(|E|)

Greedy Orderings Heuristics

MIN-INDUCED-WIDTH (MIW)

input: a graph $G = (V, E)$, $V = \{v_1, ..., v_n\}$

output: An ordering of the nodes $d = (v_1, ..., v_n)$.

1. **for** $j = n$ to 1 by -1 do
2. $r \leftarrow$ a node in V with smallest degree.
3. put r in position j.
4. connect r's neighbors: $E \leftarrow E \cup \{(v_i, v_j) | (v_i, r) \in E, (v_j, r) \in E\}$,
5. remove r from the resulting graph: $V \leftarrow V - \{r\}$.

Theorem: A graph is a tree iff it has both width and induced-width of 1.

MIN-FILL (MIN-FILL)

input: a graph $G = (V, E)$, $V = \{v_1, ..., v_n\}$

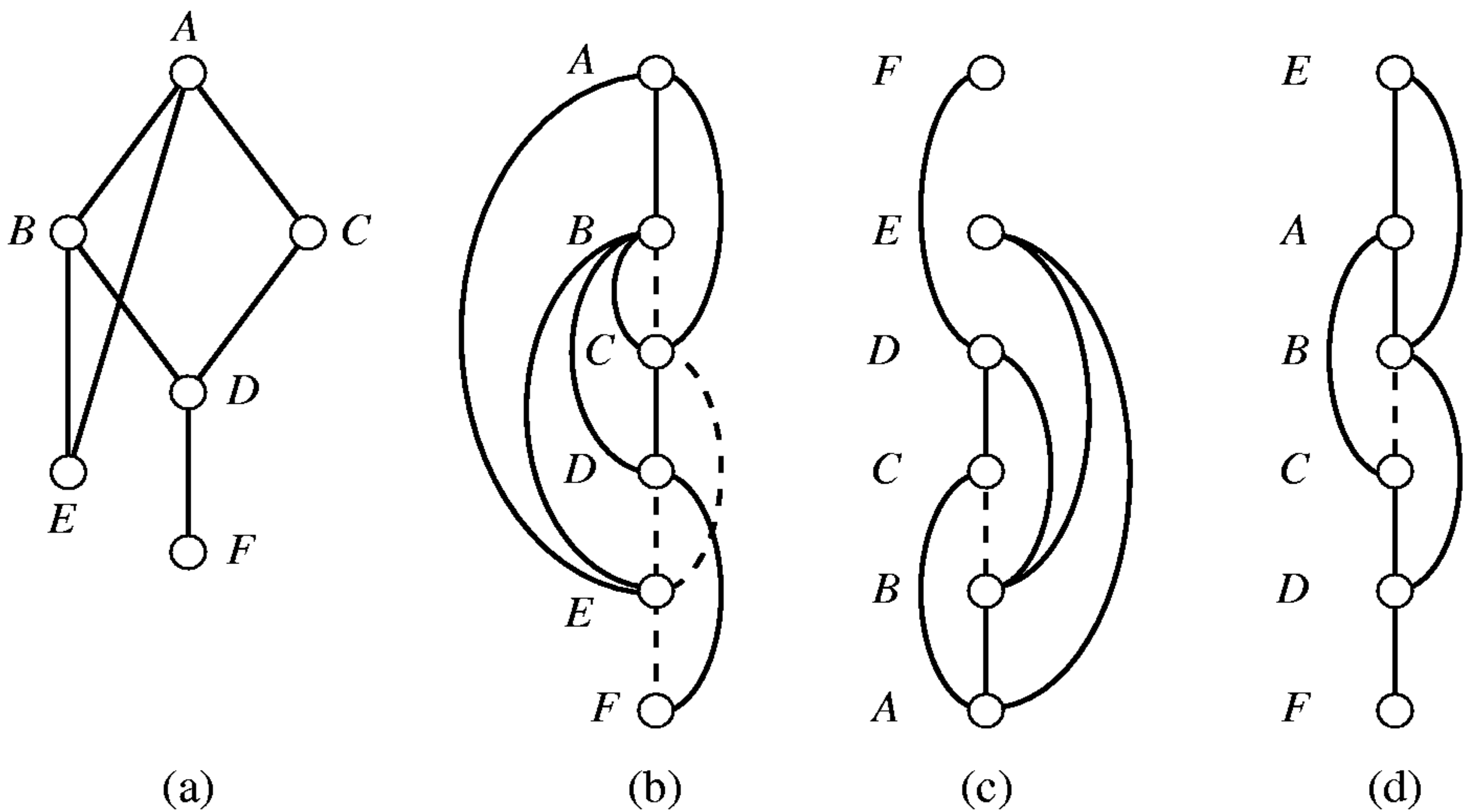
output: An ordering of the nodes $d = (v_1, ..., v_n)$.

1. **for** $j = n$ to 1 by -1 do
2. $r \leftarrow$ a node in V with smallest fill edges for his parents.
3. put r in position j.
4. connect r's neighbors: $E \leftarrow E \cup \{(v_i, v_j) | (v_i, r) \in E, (v_j, r) \in E\}$,
5. remove r from the resulting graph: $V \leftarrow V - \{r\}$.

Complexity?

O(n^3)

Different Induced-Graphs



Induced-width for chordal graphs

Definition: A graph is chordal if every cycle of length at least 4 has a chord

Finding w* over chordal graph is easy using the **max-cardinality ordering:** order vertices from 1 to n, always assigning the next number to the node connected to a largest set of previously numbered nodes. Lets d be such an ordering

A graph along max-cardinality order has no fill-in edges iff it is chordal.

On chordal graphs width=induced-width.

Max-cardinality ordering

MAX-CARDINALITY (MC)

input: a graph $G = (V, E)$, $V = \{v_1, ..., v_n\}$
output: An ordering of the nodes $d = (v_1, ..., v_n)$.
1. Place an arbitrary node in position 0.
2. **for** $j = 1$ to n do
3. $r \leftarrow$ a node in G that is connected to a largest subset of nodes in positions 1 to $j-1$, breaking ties arbitrarily.
4. **endfor**

Proposition 5.3.3 *[56] Given a graph $G = (V, E)$ the complexity of max-cardinality search is $O(n + m)$ when $|V| = n$ and $|E| = m$.*

What is the complexity of min-fill? Min-induced-width? $O(n^3)$

K-trees

Definition 5.3.4 (k-trees) *A subclass of chordal graphs are k-trees. A k-tree is a chordal graph whose maximal cliques are of size* $k+1$*, and it can be defined recursively as follows: (1) A complete graph with* k *vertices is a k-tree. (2) A k-tree with* r *vertices can be extended to* $r+1$ *vertices by connecting the new vertex to all the vertices in any clique of size* k*. A partial k-tree is a k-tree having some of its arcs removed. Namely it will clique of size smaller than* k*.*

Which greedy algorithm is best?

- MinFill, prefers a node who add the least number of fill-in arcs.
- Empirically, fill-in is the best among the greedy algorithms (MW,MIW,MF,MC)
- Complexity of greedy orderings?
  - MW is $O(n^2)$…maybe $O(nlogn + m\ \ )$?
  - MIW: O($O(n^3)$,
  - MF ($O(n^3)$,
  - MC is O(m+n), m edges.

Recent work in my group

Vibhav Gogate and Rina Dechter. "A Complete Anytime Algorithm for Treewidth". *In UAI 2004.*

Andrew E. Gelfand, Kalev Kask, and Rina Dechter. "Stopping Rules for Randomized Greedy Triangulation Schemes" in *Proceedings of AAAI 2011.*

Kalev Kask, Andrew E. Gelfand, Lars Otten, and Rina Dechter. "Pushing the Power of Stochastic Greedy Ordering Schemes for Inference in Graphical Models" in Proceedings of AAAI 2011.

Kask, Gelfand and Dechter, BEEM: Bucket Elimination with External memory, AAAI 2011 or UAI 2011

Potential project

Mixed Networks

Augmenting Probabilistic networks with constraints because:

- Some information in the world is deterministic and undirected ($X \neq Y$).
- Some queries are complex or evidence are complex (cnf formulas)

Queries are probabilistic queries

Mixed Beliefs and Constraints

$$\varphi = (G \vee D) \wedge (\neg D \vee B)$$

$$P(\varphi) = ?$$

If the constraint is a cnf formula

Queries over hybrid network:

Complex evidence structure

$$P(\bar{x} \mid \varphi) = ?$$

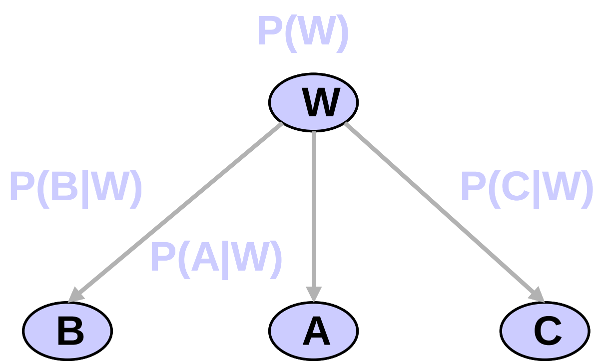
$$P(x_1 \mid \varphi) = ?$$

All reduce to cnf queries over a Belief network:

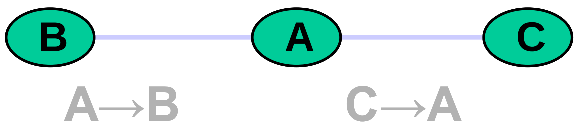
CPE (CNF probability evaluation): Given a belief network, and a cnf formula, find its probability.

Party example again

PN



CN



Query:
Is it likely that Chris goes to the party if Becky does not but the weather is bad?

Semantics?

Algorithms?

$$P(C, \neg B \mid w = bad, A \rightarrow B, C \rightarrow A)$$

Bucket Elimination for Mixed networks

The CPE query

$$P_{\mathcal{B}}(\varphi) = \sum_{\mathbf{x}_\varphi \in Mod(\varphi)} P(\mathbf{x}_\varphi)$$

Using the belief network product form we get:

$$P_{\mathcal{B}}(\varphi) = \sum_{\{\mathbf{x} | \mathbf{x}_\varphi \in Mod(\varphi)\}} \prod_{i=1}^{n} P(x_i | \mathbf{x}_{pa_i}).$$

P((C → B) and P(A → C))

Algorithm 1: BE-CPE

Input: A belief network $\mathcal{M} = (\mathcal{B}, \simeq)$, $\mathcal{B} = \langle \mathbf{X}, \mathbf{D}, \mathbf{P}_G, \prod \rangle$, where $\mathcal{B} = \{P_1, ..., P_n\}$; a CNF formula on k propositions $\varphi = \{\alpha_1, ...\alpha_m\}$ defined over k propositions; an ordering of the variables, $d = \{X_1, \ldots, X_n\}$.

Output: The belief $P(\varphi)$.

1 Place buckets with unit clauses last in the ordering (to be processed first).

`// Initialize`

Partition $\mathcal{B}$ and φ into $bucket_1, \ldots, bucket_n$, where $bucket_i$ contains all the CPTs and clauses whose highest variable is X_i.

Put each observed variable into its appropriate bucket. (We denote probabilistic functions by λs and clauses by αs).

2 **for** $p \leftarrow n$ **downto** 1 **do** `// Backward`

  Let $\lambda_1, \ldots, \lambda_j$ be the functions and $\alpha_1, \ldots, \alpha_r$ be the clauses in $bucket_p$

  `Process-bucket`$_p(\sum, (\lambda_1, \ldots, \lambda_j), (\alpha_1, \ldots, \alpha_r))$

3 **return** $P(\varphi)$ as the result of processing $bucket_1$.

Procedure `Process-bucket`$_p$ $(\sum, (\lambda_1, \ldots, \lambda_j), (\alpha_1, \ldots, \alpha_r))$.

if *bucket*$_p$ *contains evidence* $X_p = x_p$ **then**

1. Assign $X_p = x_p$ to each λ_i and put each resulting function in the bucket of its latest variable
2. Resolve each α_i with the unit clause, put non-tautology resolvents in the buckets of their latest variable and **move any bucket with unit clause to top of processing**

else

$\lambda_p \leftarrow \sum_{\{x_p | \mathbf{x}_{U_p} \in Mod(\alpha_1,\ldots,\alpha_r)\}} \prod_{i=1}^{j} \lambda_i$

Add λ_p to the bucket of the latest variable in S_p, where

$S_p = scope(\lambda_1, ..., \lambda_j, \alpha_1, ..., \alpha_r)$, $U_p = scope(\alpha_1, ..., \alpha_r)$.

(a) Directed acyclic graph (b) Moral graph

Processing Mixed Buckets

We compute:

In Bucket G: $\lambda_G(f,d) = \sum_{\{g|g \vee d = true\}} P(g|f)$

In $Bucket_F$: $\lambda_F(b,c,d) = \sum_f P(f|b,c)\lambda_G(f,d)$

In $Bucket_D$: $\lambda_D(a,b,c) = \sum_{\{d|\neg d \vee \neg b = true\}} P(d|a,b)\lambda_F(b,c,d)$

In $Bucket_B$: $\lambda_B(a,c) = \sum_{\{b|b \vee c = true\}} P(b|a)\lambda_D(a,b,c)\lambda_F(b,c)$

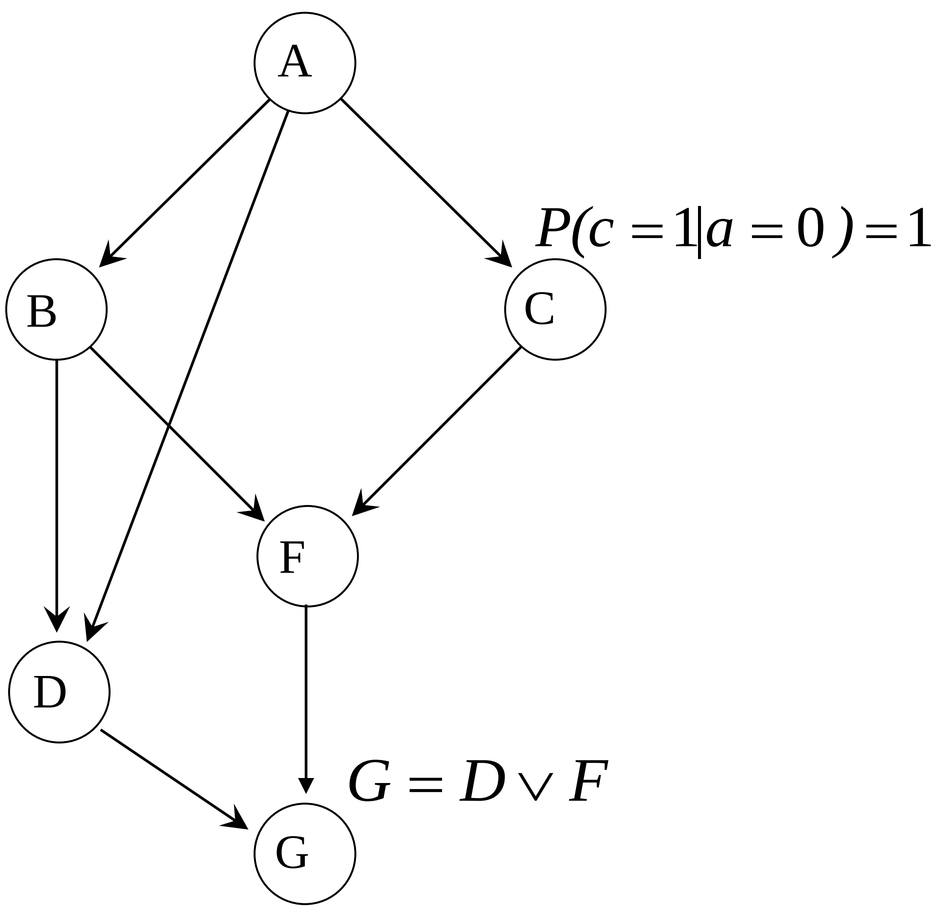
In $Bucket_C$: $\lambda_C(a) = \sum_c P(c|a)\lambda_B(a,c)$

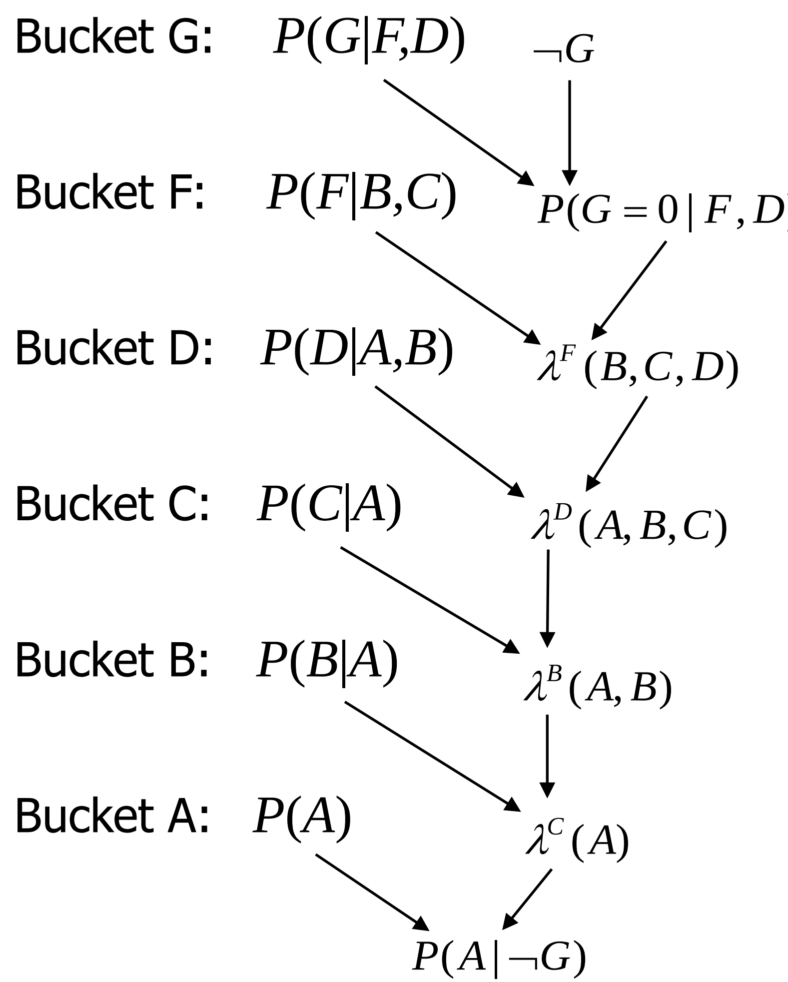
In $Bucket_A$: $\lambda_A = \sum_a P(a)\lambda_C(a)$

$P(\varphi) = \lambda_A$.

For example in $bucket_G$, $\lambda_G(f, d = 0) = P(g = 1|f)$, because if $D = 0$ g must get the value "1", while $\lambda_G(f, d = 1) = P(g = 0|f) + P(g = 1|f)$. In summary, we have the following.

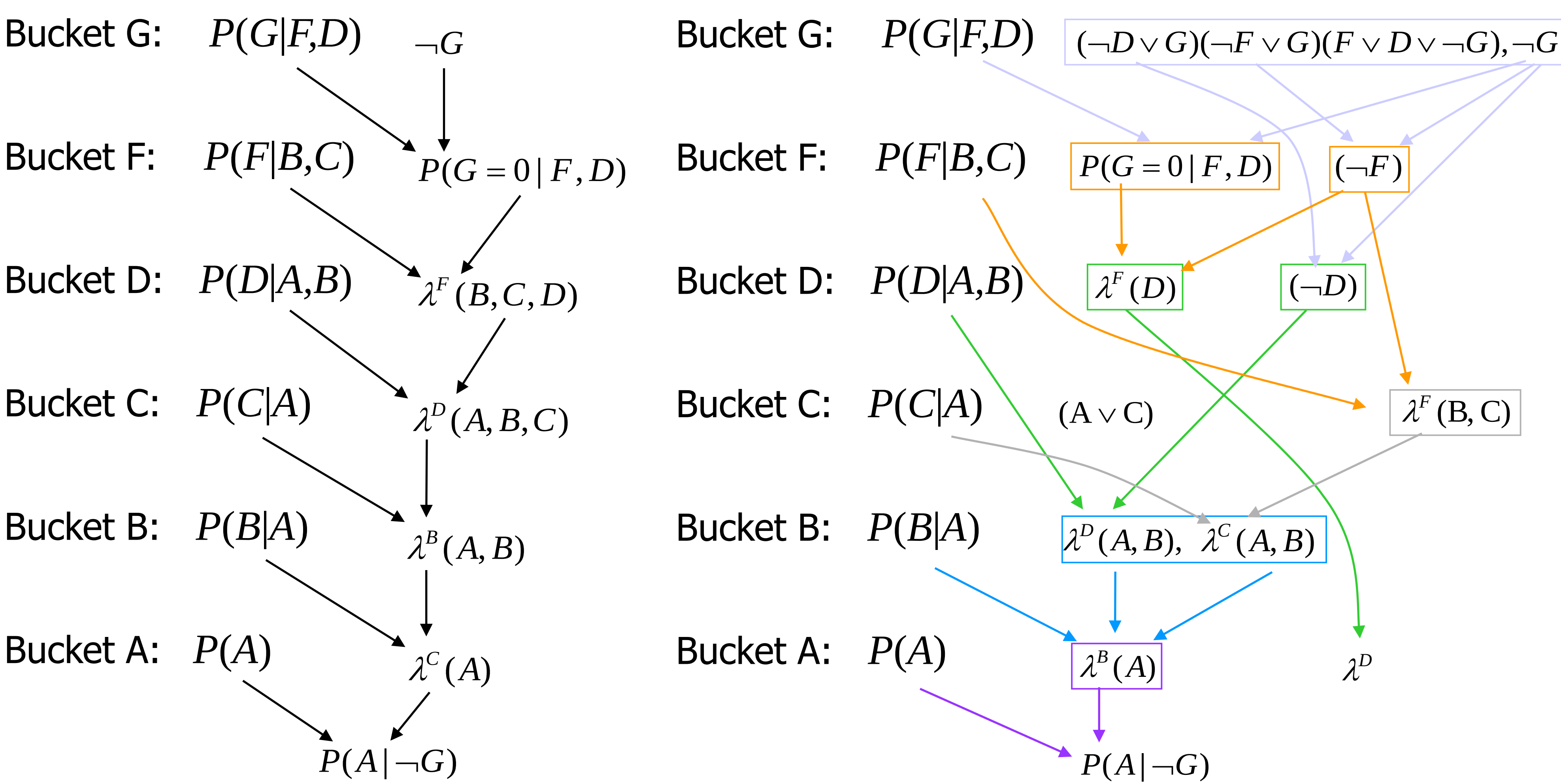
A Hybrid Belief Network





Belief network P(g,f,d,c,b,a)
=P(g|f,d)P(f|c,b)P(d|b,a)P(b|a)P(c|a)P(a)

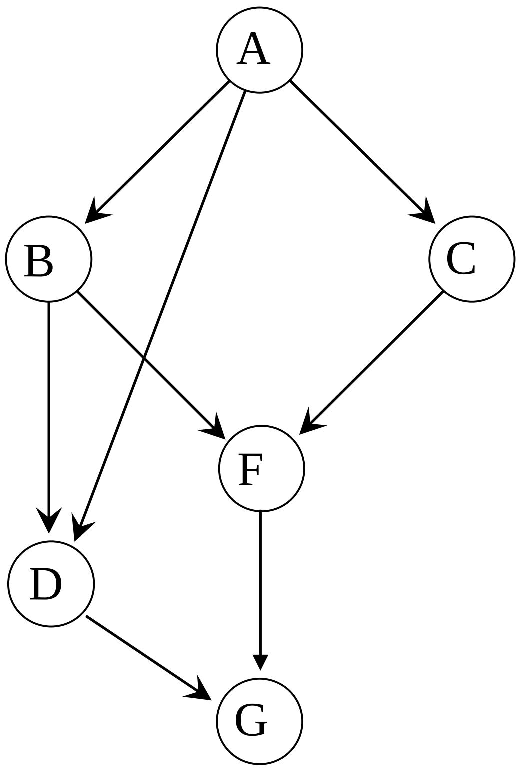
Variable elimination for a mixed network:

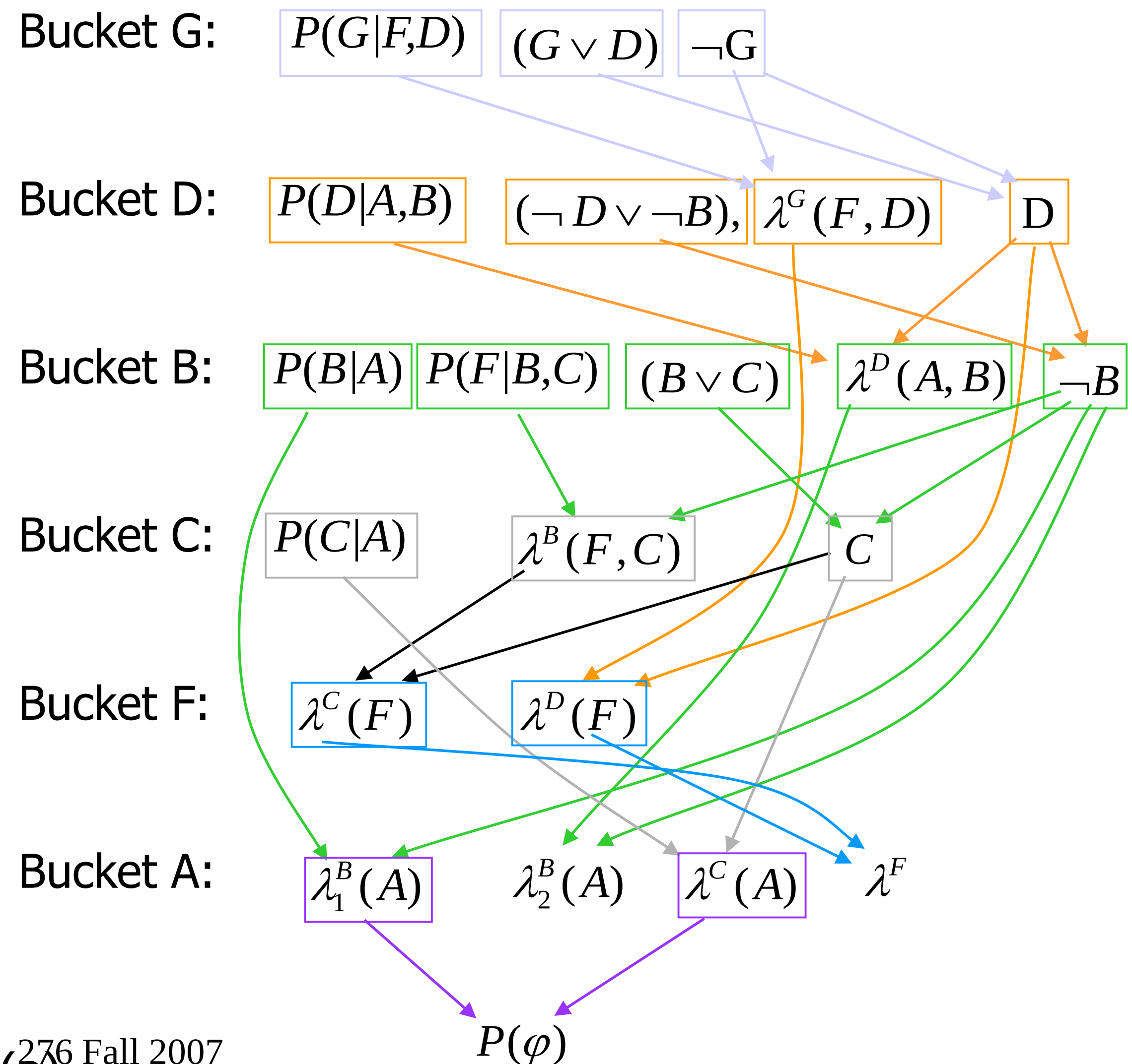


(a) regular Elim-CPE

(b) Elim-CPE-D with clause extraction

Trace of Elim-CPE





Belief network P(g,f,d,c,b,a)
=P(g|f,d)P(f|c,b)P(d|b,a)P(b|a)P(c|a)P(a)

Bucket-elimination example for a mixed network

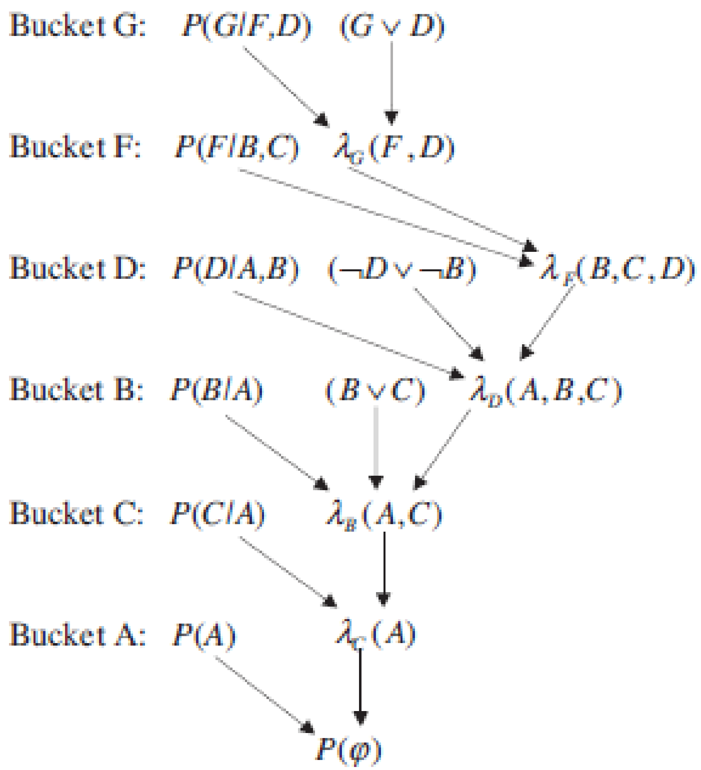


Figure 4.15: Execution of BE-CPE.

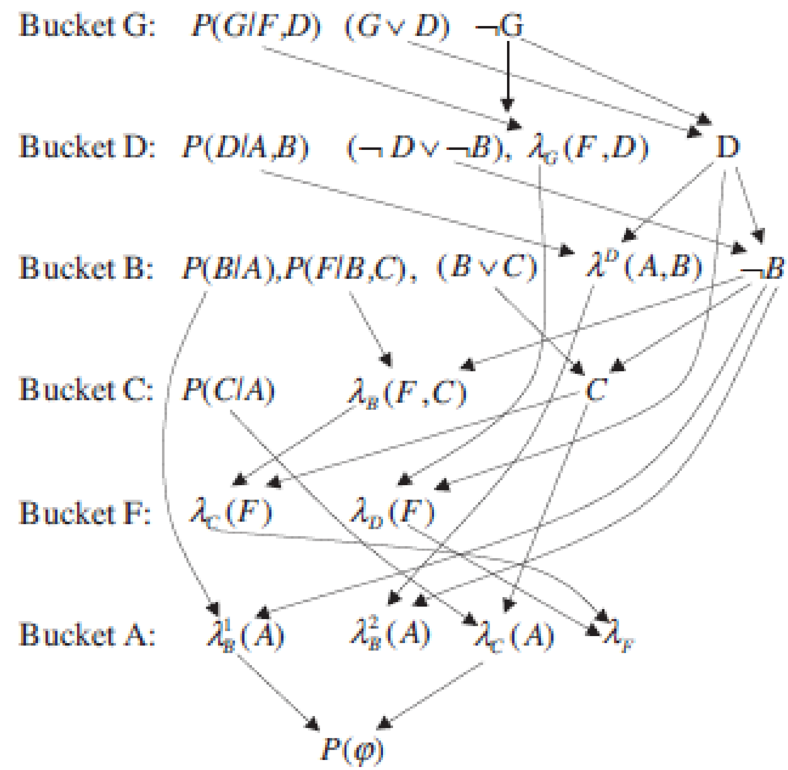


Figure 4.16: Execution of BE-CPE (evidence $\neg G$).

Markov Networks

Definition 2.23 Markov networks. A Markov network is a graphical model $\mathcal{M} = \langle \mathbf{X}, \mathbf{D}, \mathbf{H}, \prod \rangle$ where $\mathbf{H} = \{\psi_1, \ldots, \psi_m\}$ is a set of potential functions where each potential ψ_i is a non-negative real-valued function defined over a scope of variables $\mathcal{S} = \{\mathbf{S}_1, ..., \mathbf{S}_m\}$. $\mathbf{S}_i$. The Markov network represents a global joint distribution over the variables $\mathbf{X}$ given by:

$$P_{\mathcal{M}} = \frac{1}{Z} \prod_{i=1}^{m} \psi_i \quad , \quad Z = \sum_{\mathbf{X}} \prod_{i=1}^{m} \psi_i$$

where the normalizing constant Z is called the partition function.

Dechter, chapter 2

Complexity

Theorem 4.21 Complexity of BE-cpe. *Given a mixed network $M_{\mathcal{B},\varphi}$ having mixed graph is G, with $w^*(d)$ its induced width along ordering d, k the maximum domain size and r be the number of input functions. The time complexity of BE-cpe is $O(r \cdot k^{w^*(d)+1})$ and its space complexity is $O(n \cdot k^{w^*(d)})$. (Prove as an exercise.)*

DEFINITION: An undirected graph $G = (V, E)$ is said to be *chordal* if every cycle of length four or more has a chord, i.e., an edge joining two nonconsecutive vertices.

THEOREM 7: Let G be an undirected graph $G = (V, E)$. The following four conditions are equivalent:

1. G is chordal.

2. The edges of G can be directed acyclically so that every pair of converging arrows emanates from two adjacent vertices.

3. All vertices of G can be deleted by arranging them in separate piles, one for each clique, and then repeatedly applying the following two operations:

   - Delete a vertex that occurs in only one pile.

   - Delete a pile if all its vertices appear in another pile.

4. There is a tree T (called a *join tree*) with the cliques of G as vertices, such that for every vertex v of G, if we remove from T all cliques not containing v, the remaining subtree stays connected. In other words, any two cliques containing v are either adjacent in T or connected by a path made entirely of cliques that contain v.

The running intersection property