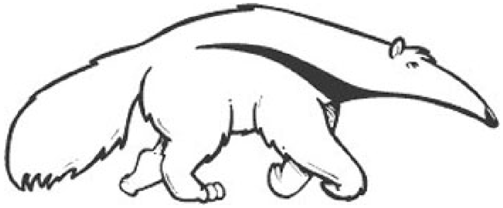
Algorithms for Reasoning with Probabilistic Graphical Models

International Summer School on Deep Learning
July 2017

Prof. Rina Dechter
Prof. Alexander Ihler

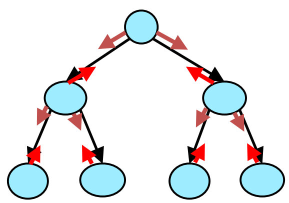


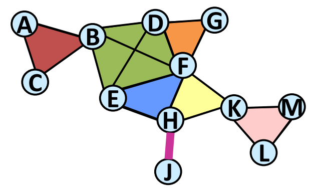
BREN:ICS
INFORMATION AND COMPUTER SCIENCES

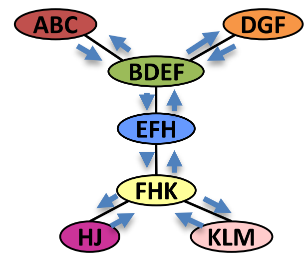
UNIVERSITY of CALIFORNIA IRVINE

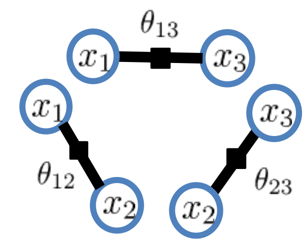
Outline of Lectures

- Class 1: Introduction and Inference

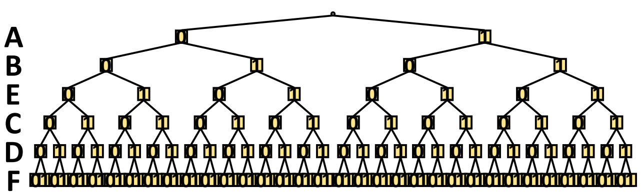


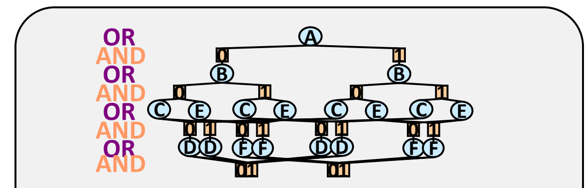






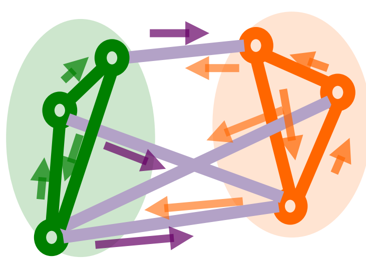
- Class 2: Search

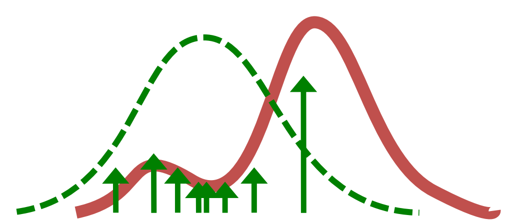




Context minimal AND/OR search graph

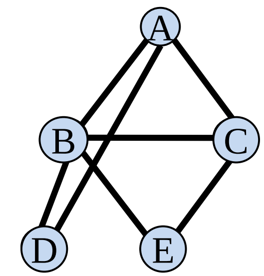
- Class 3: Variational Methods and Monte-Carlo Sampling



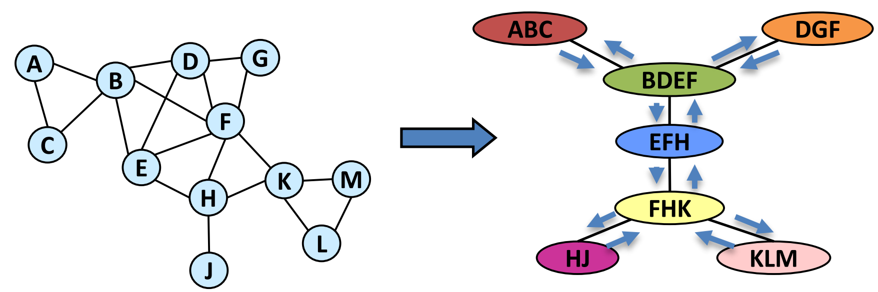


RoadMap: Introduction and Inference

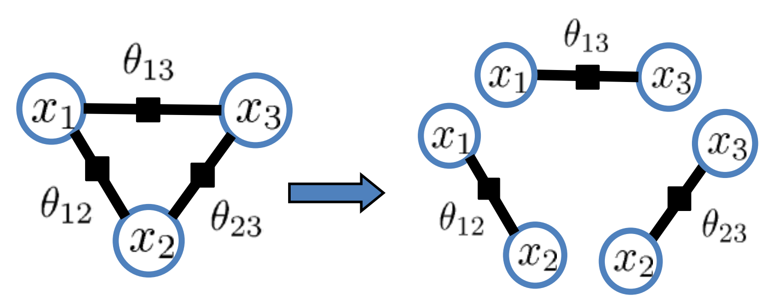
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - Algorithms overview



- Inference algorithms, exact
  - Bucket elimination for trees
  - Bucket elimination
  - Jointree clustering
  - Elimination orders



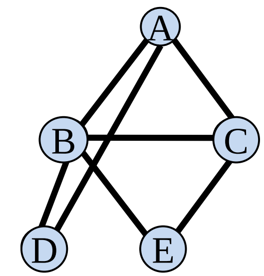
- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - Belief propagation



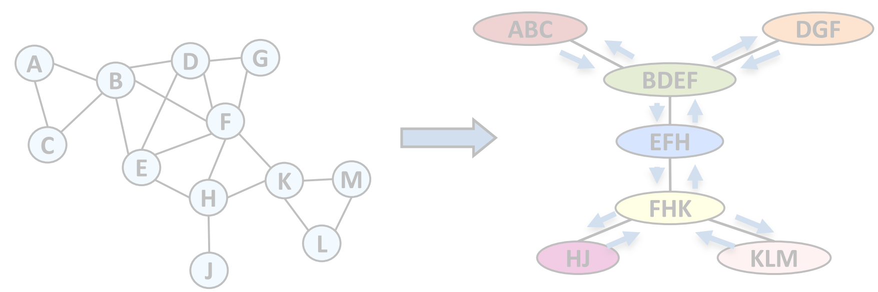
- Summary and Class 2

RoadMap: Introduction and Inference

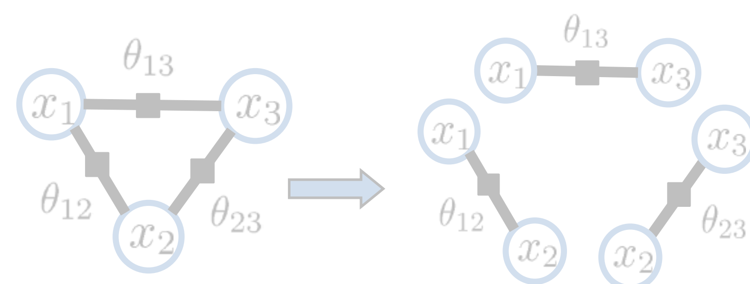
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - Algorithms overview



- Inference algorithms, exact
  - Bucket elimination for trees
  - Bucket elimination
  - Jointree clustering
  - Elimination orders



- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - Belief propagation



- Summary and Class 2

Graphical models

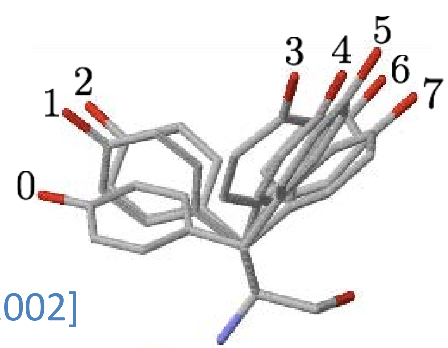
- Describe structure in large problems
  - Large complex system $F(X)$
  - Made of “smaller”, “local” interactions $f_\alpha(x_\alpha)$
  - Complexity emerges through interdependence

Graphical models

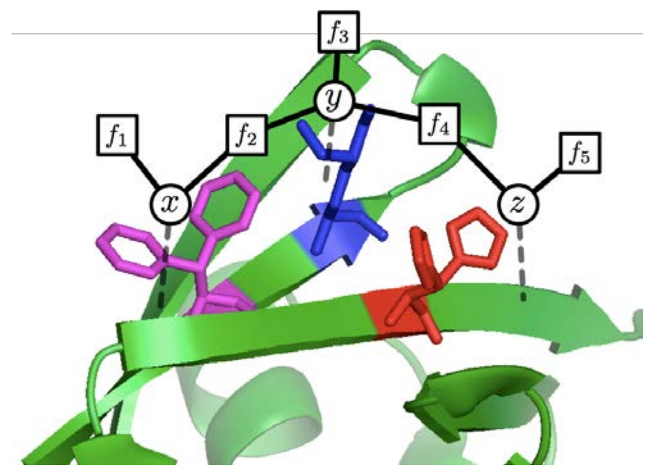
- Describe structure in large problems
  - Large complex system $F(X)$
  - Made of “smaller”, “local” interactions $f_\alpha(x_\alpha)$
  - Complexity emerges through interdependence
- Examples & Tasks
  - Maximization (MAP): compute the most probable configuration

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \prod_\alpha f_\alpha(\mathbf{x}_\alpha) \qquad f(\mathbf{x}^*) = \max_{\mathbf{x}} \prod_\alpha f_\alpha(\mathbf{x}_\alpha)$$

[Yanover & Weiss 2002]



Phenylalanine



Graphical models

- Describe structure in large problems
  - Large complex system $F(X)$
  - Made of "smaller", "local" interactions $f_\alpha(x_\alpha)$
  - Complexity emerges through interdependence
- Examples & Tasks
  - Summation & marginalization

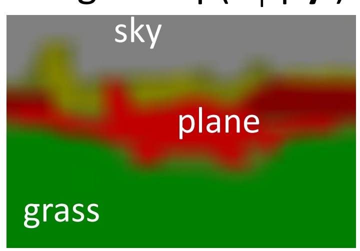
"partition function"

$$p(x_i) = \frac{1}{Z} \sum_{\mathbf{x} \setminus x_i} \prod_\alpha f_\alpha(\mathbf{x}_\alpha) \quad \text{and} \quad Z = \sum_{\mathbf{x}} \prod_\alpha f_\alpha(\mathbf{x}_\alpha)$$

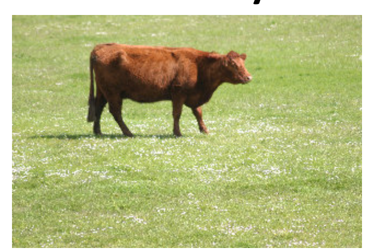
Observation **y**

$\Rightarrow$

Marginals p(x_i | **y**)

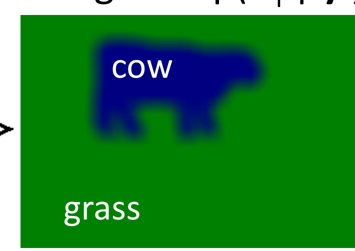


Observation **y**



$\Rightarrow$

Marginals p(x_i | **y**)



e.g., [Plath et al. 2009]

Graphical models

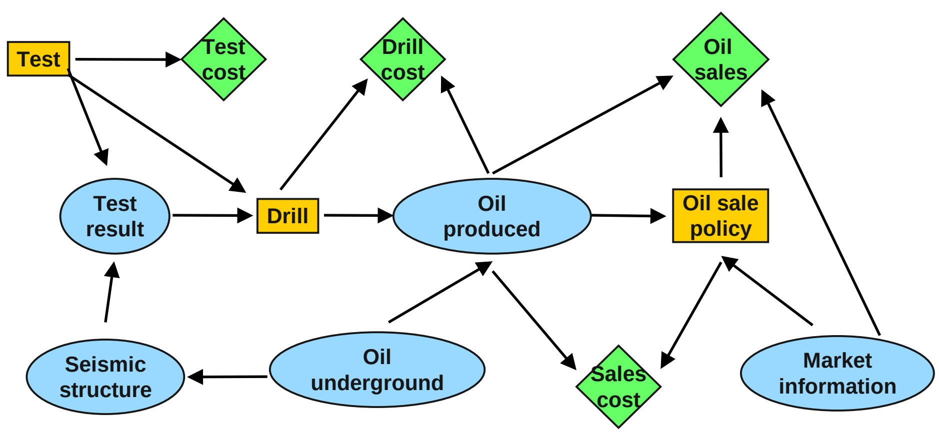
- Describe structure in large problems
  - Large complex system $F(X)$
  - Made of "smaller", "local" interactions $f_\alpha(x_\alpha)$
  - Complexity emerges through interdependence
- Examples & Tasks
  - Mixed inference (marginal MAP, MEU, …)

$$f(\mathbf{x}_M^*) = \max_{\mathbf{x}_M} \sum_{\mathbf{x}_S} \prod_\alpha f_\alpha(\mathbf{x}_\alpha)$$

Influence diagrams &
optimal decision-making

(the "oil wildcatter" problem)

e.g., [Raiffa 1968; Shachter 1986]



Graphical models

A *graphical model* consists of:

$X = \{X_1, \ldots, X_n\}$ -- variables

$D = \{D_1, \ldots, D_n\}$ -- domains (we'll assume discrete)

$F = \{f_{\alpha_1}, \ldots, f_{\alpha_m}\}$ -- functions or "factors"

and a *combination operator*

Example:

$A \in \{0, 1\}$

$B \in \{0, 1\}$

$C \in \{0, 1\}$

$f_{AB}(A, B), \quad f_{BC}(B, C)$

The *combination operator* defines an overall function from the individual factors,

e.g., "+" : $F(A, B, C) = f_{AB}(A, B) + f_{BC}(B, C)$

Notation:

Discrete Xi values called states

Tuple or configuration: states taken by a set of variables

Scope of f: set of variables that are arguments to a factor f

often index factors by their scope, e.g., $f_\alpha(X_\alpha), \quad X_\alpha \subseteq X$

Graphical models

A *graphical model* consists of:

$X = \{X_1, \ldots, X_n\}$ -- variables

$D = \{D_1, \ldots, D_n\}$ -- domains (we'll assume discrete)

$F = \{f_{\alpha_1}, \ldots, f_{\alpha_m}\}$ -- functions or "factors"

and a *combination operator*

Example:

$A \in \{0,1\}$

$B \in \{0,1\}$

$C \in \{0,1\}$

$f_{AB}(A,B), \quad f_{BC}(B,C)$

$$F(A,B,C) = f_{AB}(A,B) + f_{BC}(B,C)$$

For discrete variables, think of functions as "tables"
(though we might represent them more efficiently)

A	B	f(A,B)
0	0	6
0	1	0
1	0	0
1	1	6

$F(A=0, B=1, C=1)$

+

B	C	f(B,C)
0	0	6
0	1	0
1	0	0
1	1	6

=

A	B	C	f(A,B,C)
0	0	0	12
0	0	1	6
0	1	0	0
0	1	1	6
1	0	0	6
1	0	1	0
1	1	0	6
1	1	1	12

= 0 + 6

Canonical forms

A *graphical model* consists of:

$X = \{X_1, \ldots, X_n\}$ -- variables

$D = \{D_1, \ldots, D_n\}$ -- domains

$F = \{f_{\alpha_1}, \ldots, f_{\alpha_m}\}$ -- functions or "factors"

and a *combination operator*

Typically either multiplication or summation; mostly equivalent:

$$f_\alpha(X_\alpha) \geq 0$$

$$F(X) = \prod_\alpha f_\alpha(X_\alpha)$$

log / exp

$$\theta_\alpha(X_\alpha) = \log f_\alpha(X_\alpha) \in \mathbb{R}$$

$$\theta(X) = \log F(x) = \sum_\alpha \theta_\alpha(X_\alpha)$$

Product of nonnegative factors
(probabilities, 0/1, etc.)

Sum of factors
(costs, utilities, etc.)

Graphical visualization

A *graphical model* consists of:

$X = \{X_1, \ldots, X_n\}$ -- variables

$D = \{D_1, \ldots, D_n\}$ -- domains

$F = \{f_{\alpha_1}, \ldots, f_{\alpha_m}\}$ -- functions or "factors"

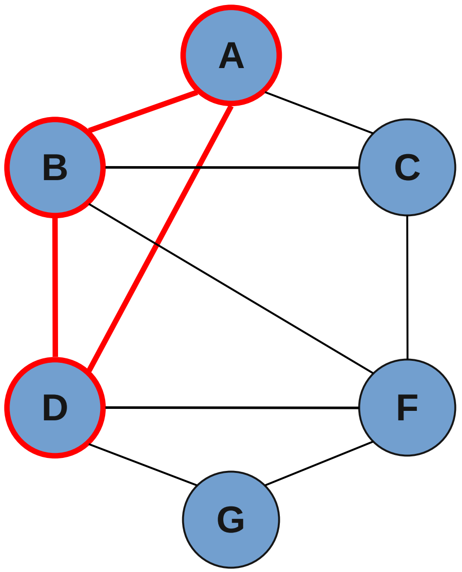
and a *combination operator*

Primal graph:

variables → nodes

factors → cliques

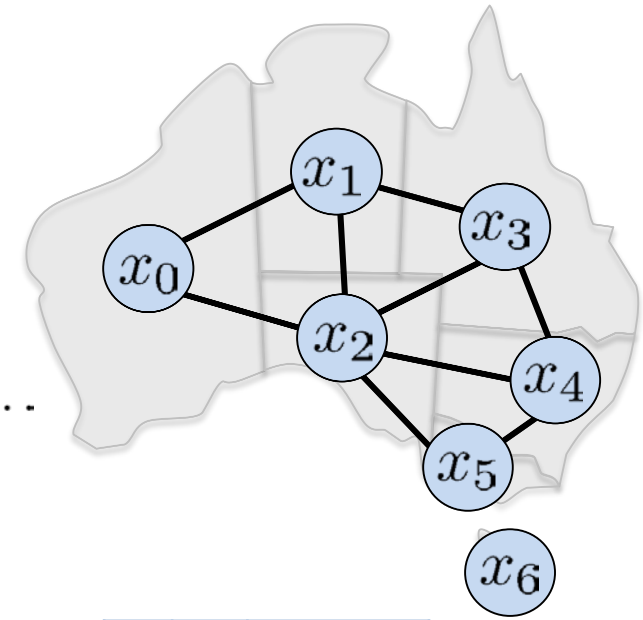
$$F(A,B,C,D,F,G) = f_1(A,B,D) + f_2(D,F,G) + f_3(B,C,F) + f_4(A,C)$$



Example: Map Coloring

$X_i \in \{\color{red}{red}, \color{green}{green}, \color{blue}{blue}\}$

$f_{ij}(X_i, X_j) = (X_i \neq X_j)$ for adjacent regions i,j



Overall function is "and" of individual constraints:

$$F(X) = f_{01}(X_0, X_1) \wedge f_{12}(X_1, X_2) \wedge f_{02}(X_0, X_2) \wedge \ldots$$

"Tabular" form:

$$f_{ij}(X_i, X_j) = \begin{cases} 1.0 & X_i \neq X_j \\ 0.0 & X_i = X_j \end{cases}$$

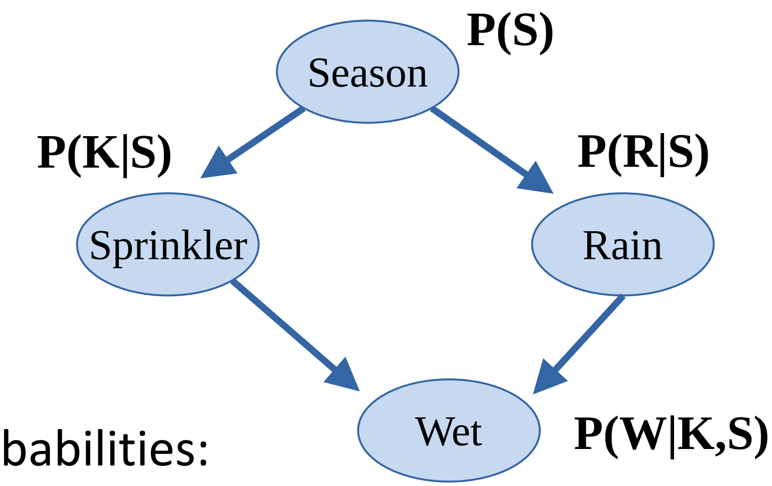
$$F(X) = \prod_{ij} f_{ij}(X_i, X_j) = \begin{cases} 1.0 & \text{all valid} \\ 0.0 & \text{any invalid} \end{cases}$$

X_0	X_1	$f(X_0, X_1)$
0	0	0
0	1	1
0	2	1
1	0	1
1	1	0
1	2	1
2	0	1
2	1	1
2	2	0

Tasks: "max": is there a solution?
"sum": how many solutions?

Example: Bayesian Networks

Random variables S,K,R,W
S has states: {Fall, Winter, Spring, Summer}
R, K, W have states: {True, False}



Overall function is product of conditional probabilities:

$$P(S,K,R,W) = P(S) \cdot P(K|S) \cdot P(R|S) \cdot P(W|K,S)$$

P(W|K,R) =

K	R	W=0	W=1
0	0	1.0	0.0
0	1	0.2	0.8
1	0	0.1	0.9
1	1	0.01	0.99

Typical tasks:

- Observe some variables' outcome
- Reason about the change in probability of others

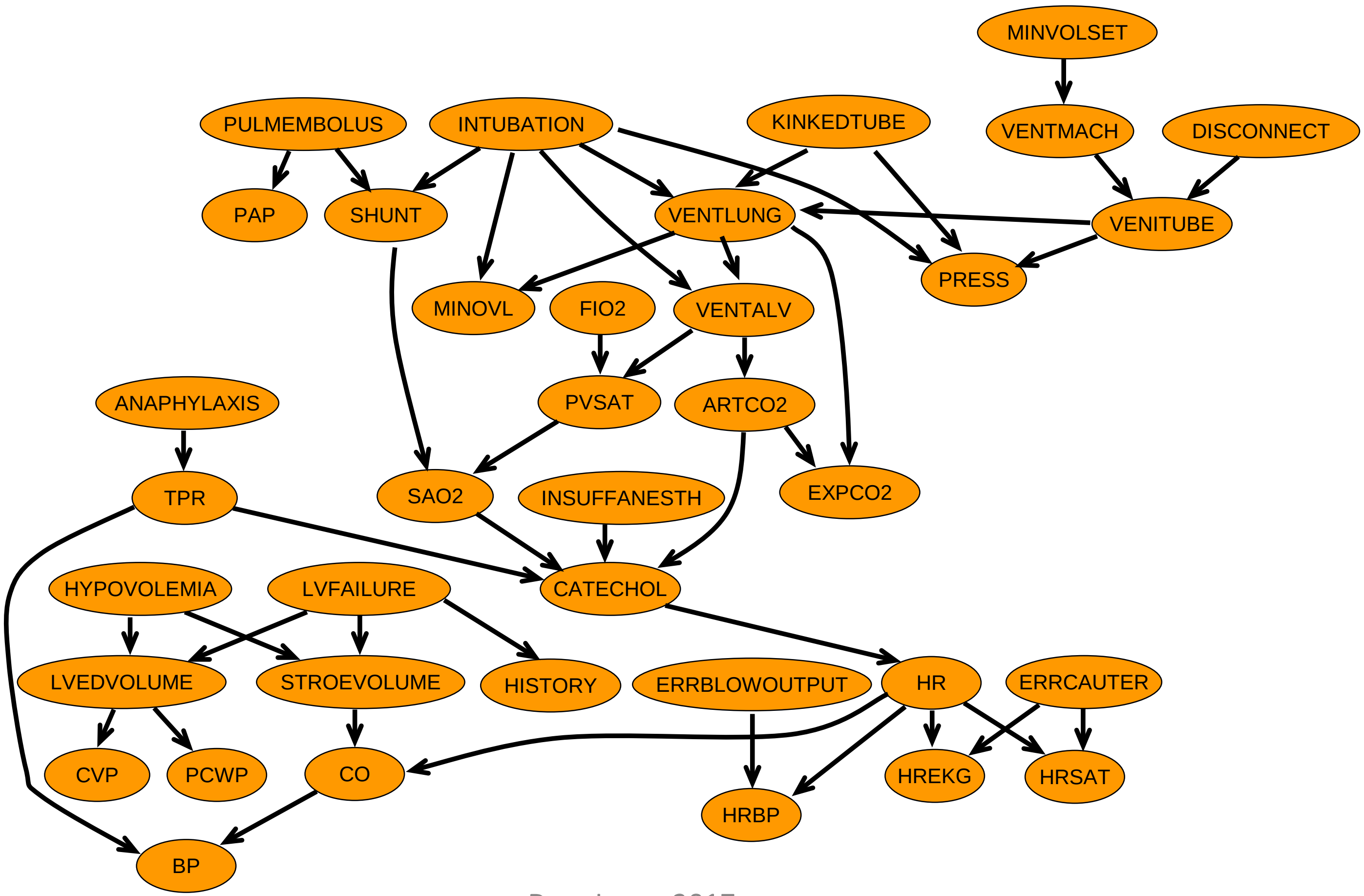
- "max": what's the most probable (MAP) state?
- "sum": what's the probability it rained, given it's wet out?

(sometime called a "belief")

Alarm network [Beinlich et al., 1989]

- Bayes nets: compact representation of large joint distributions

The “alarm” network: 37 variables, 509 parameters (rather than $2^{37} = 10^{11}$!)

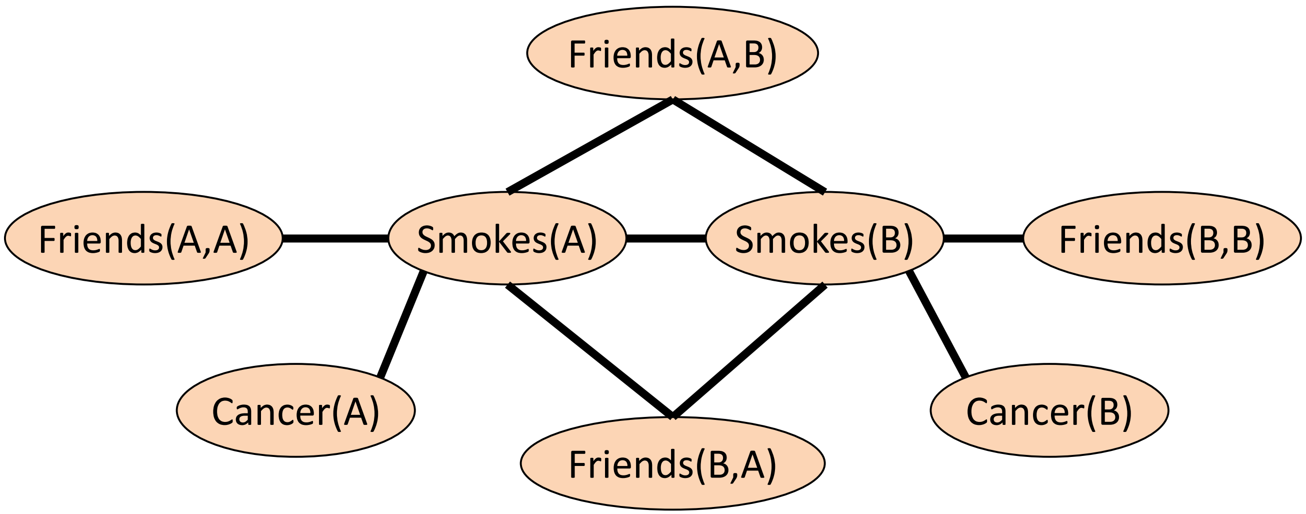


Example: Markov logic

[Richardson & Domingos 2005]

1.5	$\forall x\ Smokes(x) \Rightarrow Cancer(x)$
1.1	$\forall x, y\ Friends(x,y) \Rightarrow (Smokes(x) \Leftrightarrow Smokes(y))$

Two constants: **Anna** (A) and **Bob** (B)



S_A	C_A	$f(S_A,C_A)$
0	0	exp(1.5)
0	1	exp(1.5)
1	0	1.0
1	1	exp(1.5)

F_{AB}	S_A	S_B	f(.)
0	0	0	exp(1.1)
0	0	1	exp(1.1)
0	1	0	exp(1.1)
0	1	1	exp(1.1)
1	0	0	exp(1.1)
1	0	1	1.0
1	1	0	1.0
1	1	1	exp(1.1)

Example domains for graphical models

- Natural Language processing
  - Information extraction, semantic parsing, translation, topic models, …
- Computer vision
  - Object recognition, scene analysis, segmentation, tracking, …
- Computational biology
  - Pedigree analysis, protein folding and binding, sequence matching, …
- Networks
  - Webpage link analysis, social networks, communications, citations, ….
- Robotics
  - Planning & decision making

Graphical visualization

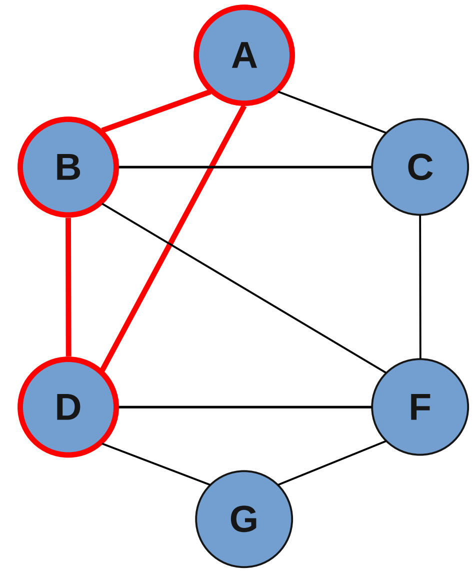
A *graphical model* consists of:

$X = \{X_1, \ldots, X_n\}$ -- variables

$D = \{D_1, \ldots, D_n\}$ -- domains

$F = \{f_{\alpha_1}, \ldots, f_{\alpha_m}\}$ -- functions or "factors"

and a *combination operator*

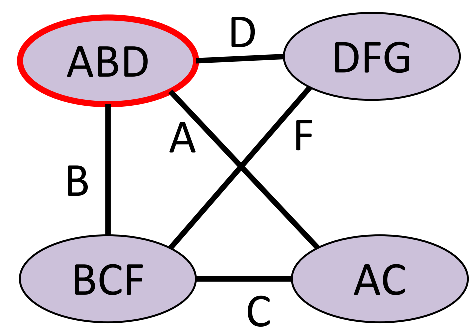


Primal graph:

variables $\rightarrow$ nodes

factors $\rightarrow$ cliques

$$F(A,B,C,D,F,G) = f_1(A,B,D) + f_2(D,F,G) + f_3(B,C,F) + f_4(A,C)$$



Dual graph:

factor scopes $\rightarrow$ nodes

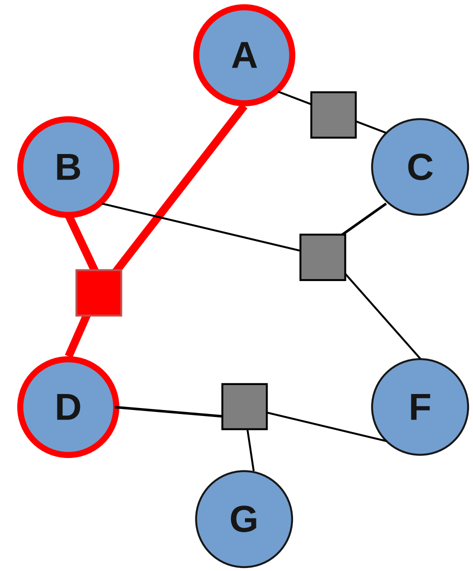
edges $\rightarrow$ intersections (separators)

Graphical visualization

"Factor" graph: explicitly indicate the scope of each factor

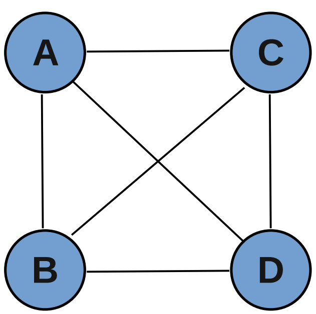
variables $\rightarrow$ circles

factors $\rightarrow$ squares



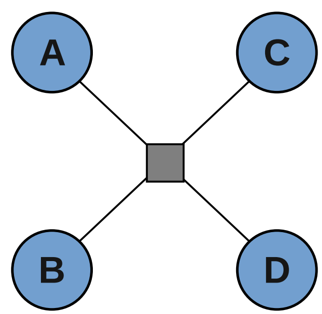
$$F(A,B,C,D,F,G) = \color{red}{f_1(A,B,D)} + f_2(D,F,G) + f_3(B,C,F) + f_4(A,C)$$

Useful for disambiguating factorization:



$\stackrel{?}{=}$

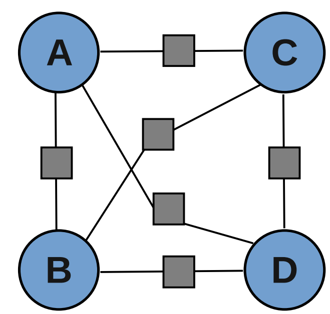
$f_1(A,B,C,D)$



O(d^4)

vs.

$f_1(A,B) + f_2(A,C) + \ldots$



pairwise: O(d^2)

Ex: Boltzmann machines

- Boltzmann machines:

$$p(x) = \frac{1}{Z} \exp \Big[\sum_i a_i x_i + \sum_{ij} w_{ij} x_i x_j \Big]$$

$$= \frac{1}{Z} \prod_i f_i(x_i) \prod_{ij} f_{ij}(x_i, x_j)$$

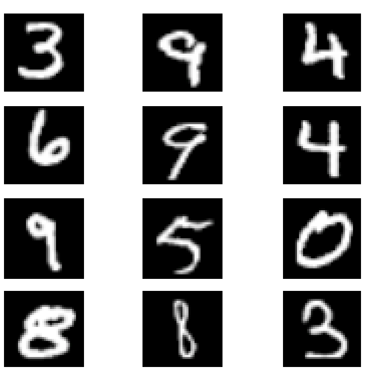
X_i	$f(X_i)$
0	1.0
1	$\exp(a_i)$

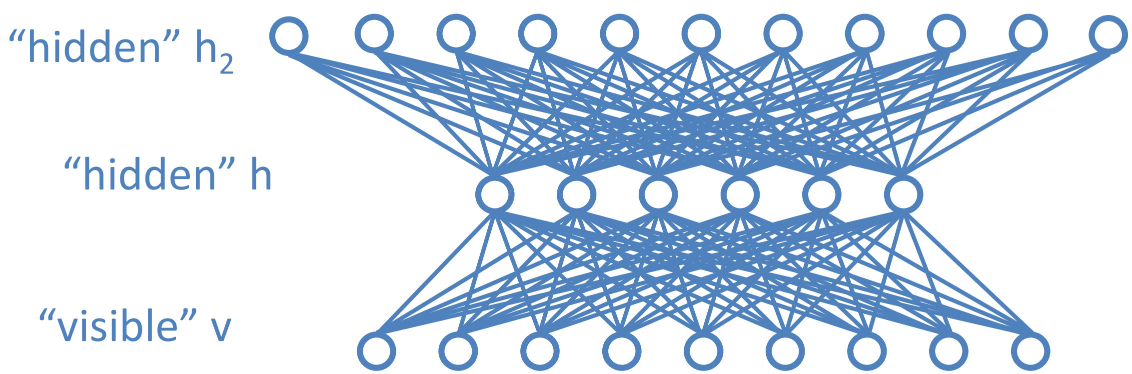
X_i	X_j	$f(X_i,X_j)$
0	0	1.0
0	1	1.0
1	0	1.0
1	1	$\exp(w_{ij})$

- Deep Boltzmann machines:

$$p(v, h_1, h_2) = \frac{1}{Z} \exp \Big[\sum_{ij} w_{1ij} v_i h_{1j} + \sum_{jk} w_{2jk} h_{1j} h_{2k} + \ldots \Big]$$

MNIST:





Graphical visualization

A ***graphical model*** consists of:

$X = \{X_1, \ldots, X_n\}$ -- variables

$D = \{D_1, \ldots, D_n\}$ -- domains

$F = \{f_{\alpha_1}, \ldots, f_{\alpha_m}\}$ -- functions or "factors"

Operators:

combination operator

(sum, product, join, ...)

elimination operator

(projection, sum, max, min, ...)

Types of queries:

Marginal: $Z = \sum_{\mathbf{x}} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$

MPE / MAP: $f(\mathbf{x}^*) = \max_{\mathbf{x}} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$

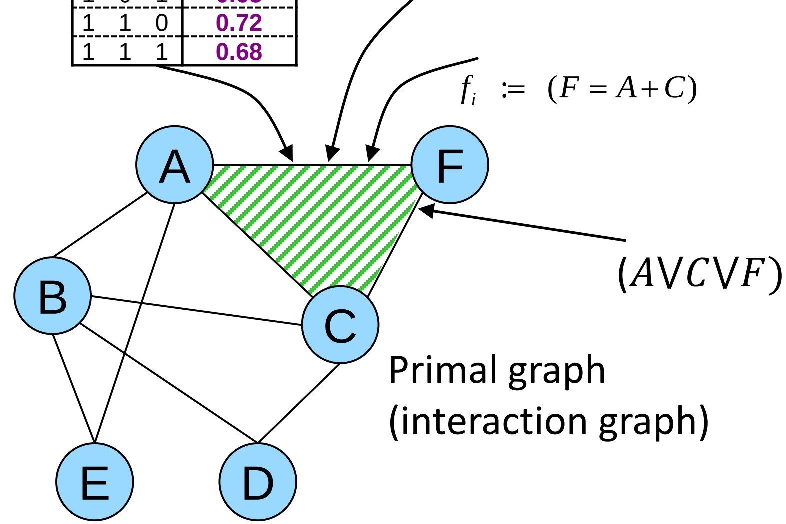
Marginal MAP: $f(\mathbf{x}_M^*) = \max_{\mathbf{x}_M} \sum_{\mathbf{x}_S} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$

Conditional Probability Table (CPT)

A	C	F	P(F|A,C)
0	0	0	0.14
0	0	1	0.96
0	1	0	0.40
0	1	1	0.60
1	0	0	0.35
1	0	1	0.65
1	1	0	0.72
1	1	1	0.68

Relation

A	C	F
red	green	blue
blue	red	red
blue	blue	green
green	red	blue

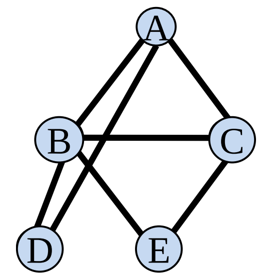


Primal graph (interaction graph)

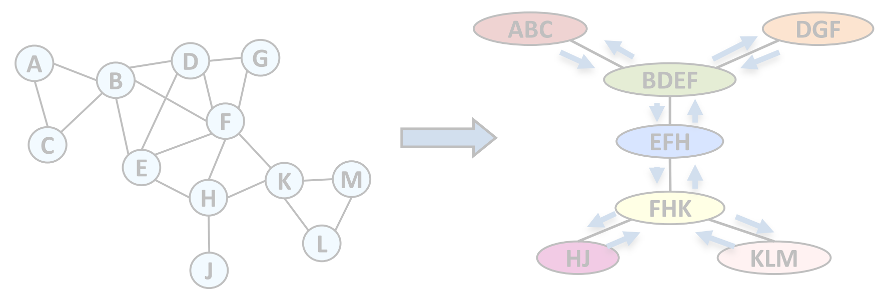
- All these tasks are NP-hard
  - exploit problem structure
  - identify special cases
  - approximate

RoadMap: Introduction and Inference

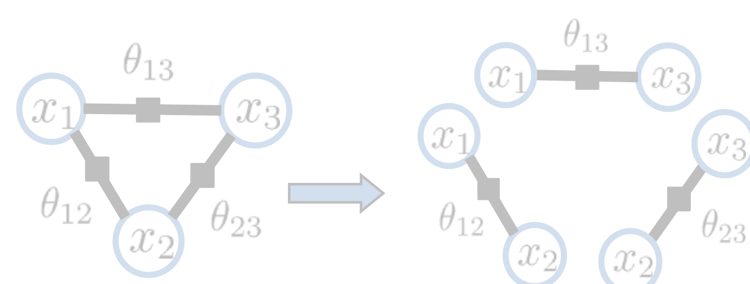
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - **Algorithms overview**



- Inference algorithms, exact
  - Bucket elimination for trees
  - Bucket elimination
  - Jointree clustering
  - Elimination orders



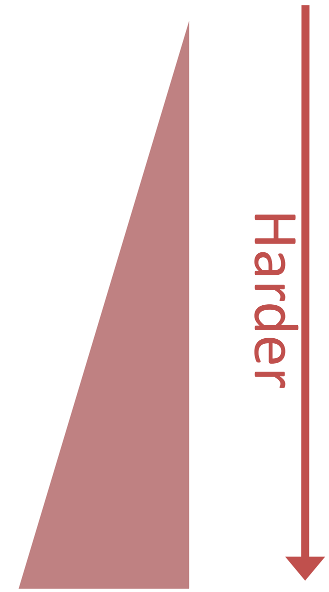
- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - Belief propagation



- Summary and Class 2

Types of queries

Max-Inference	$f(\mathbf{x}^*) = \max_{\mathbf{x}} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$
Sum-Inference	$Z = \sum_{\mathbf{x}} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$
Mixed-Inference	$f(\mathbf{x}_M^*) = \max_{\mathbf{x}_M} \sum_{\mathbf{x}_S} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$

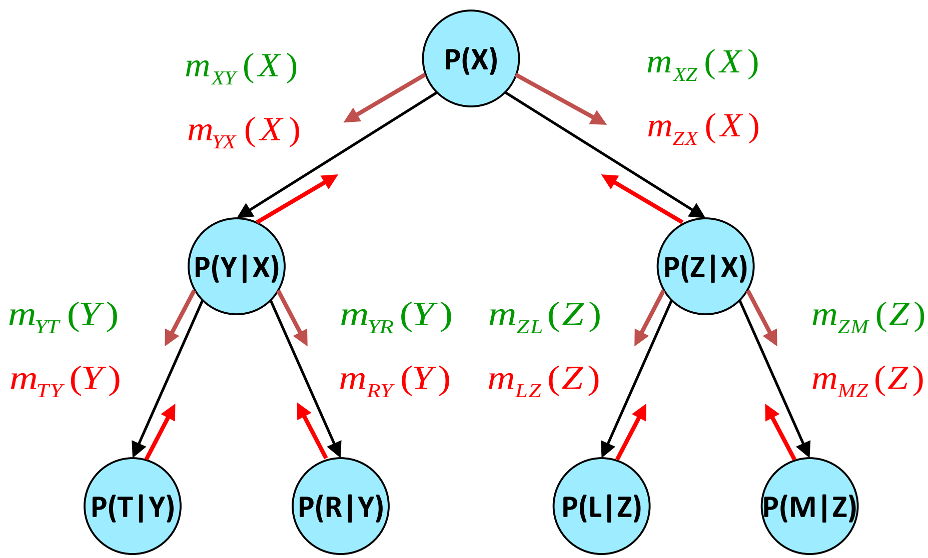


- **NP-hard**: exponentially many terms
- We will focus on **approximation** algorithms
  - **Anytime**: very fast & very approximate ! Slower & more accurate

Tree-solving is easy

Belief updating (sum-prod)

CSP – consistency (projection-join)



MPE (max-prod)

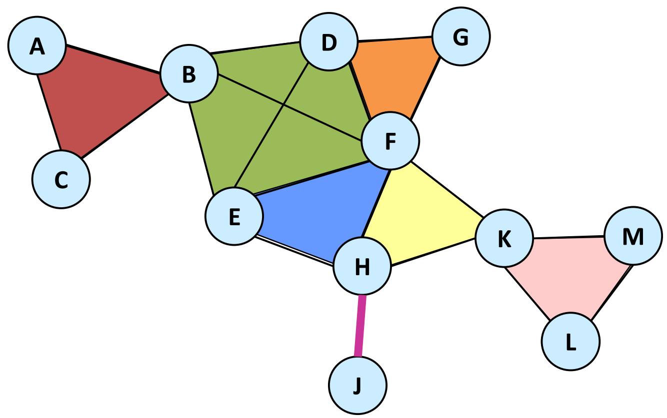
#CSP (sum-prod)

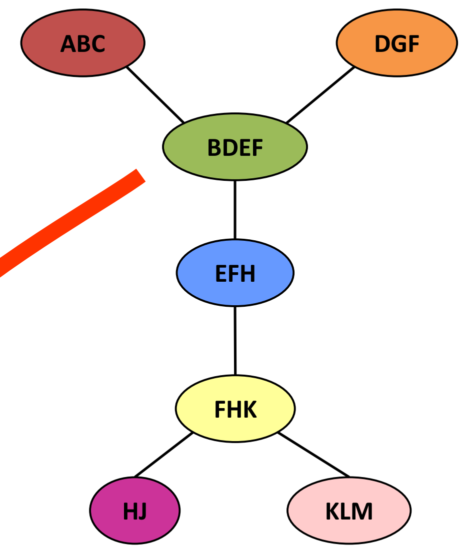
Trees are processed in linear time and memory

Transforming into a Tree

- **By Inference (thinking)**
  - Transform into a single, equivalent tree of sub-problems
- **By Conditioning (guessing)**
  - Transform into many tree-like sub-problems.

Inference and Treewidth





Inference algorithm:

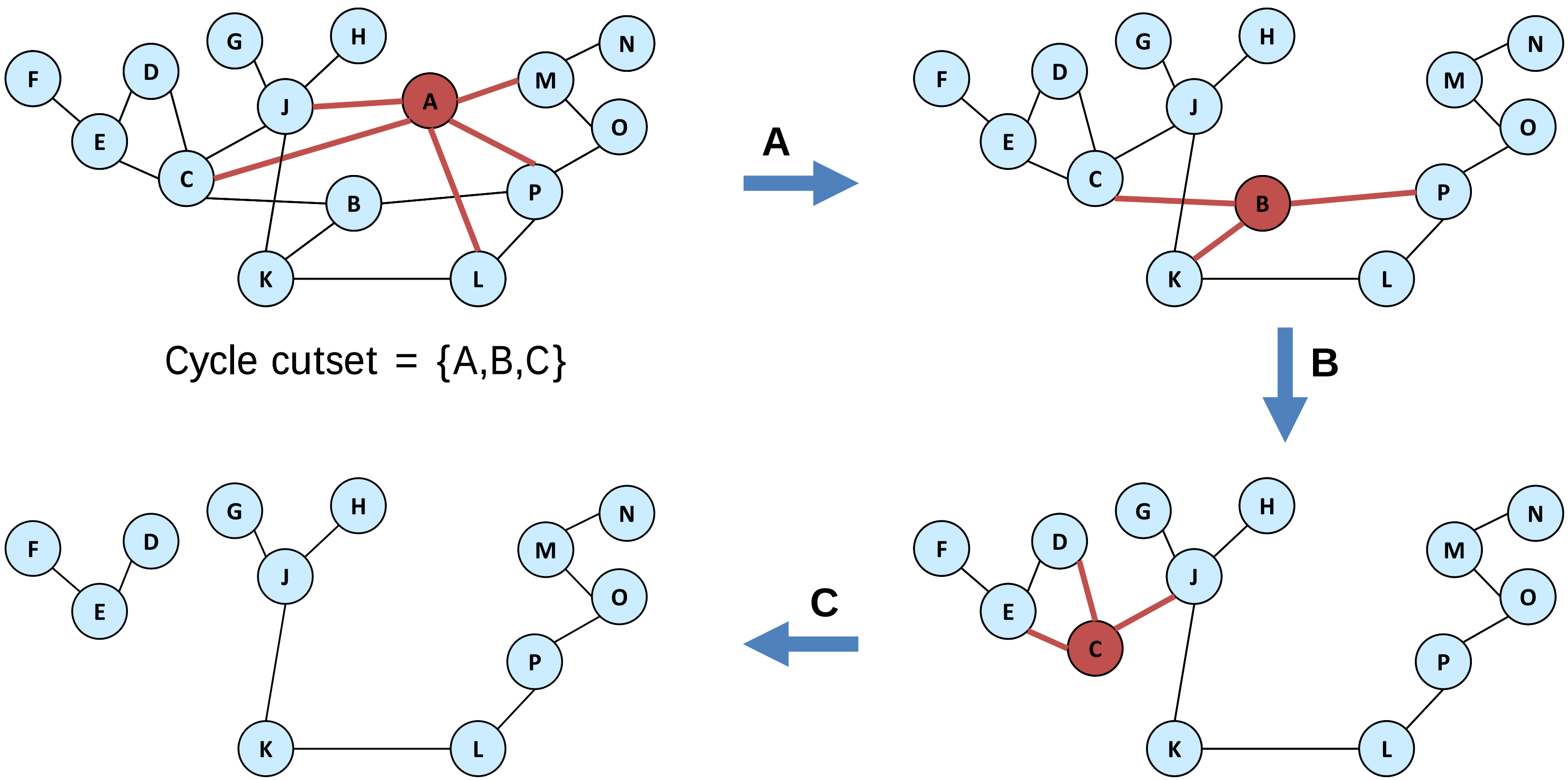
Time: exp(tree-width)

Space: exp(tree-width)

treewidth = 4 - 1 = 3

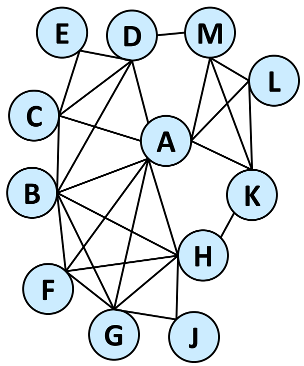
treewidth = (maximum cluster size) - 1

Conditioning and Cycle cutset

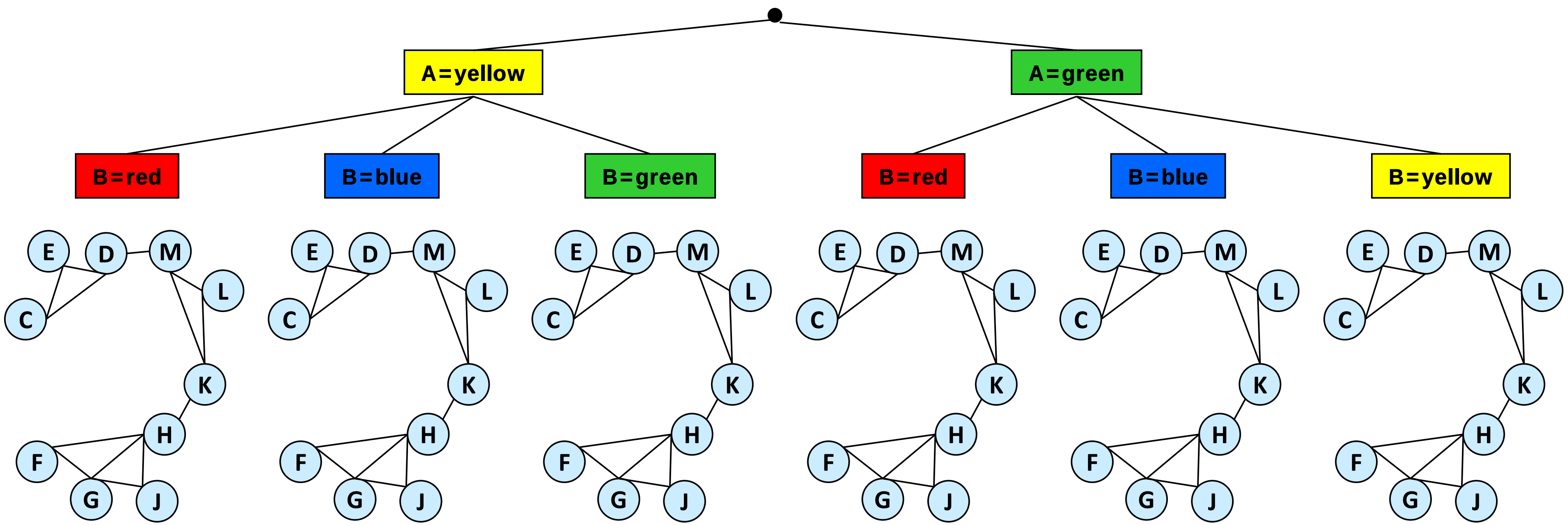


Search over the Cutset

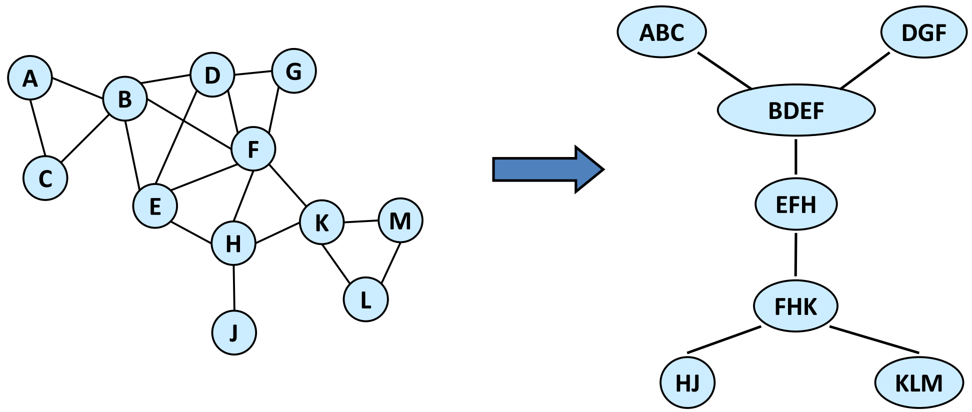
Graph Coloring problem



- Inference may require too much memory
- **Condition** on some of the variables

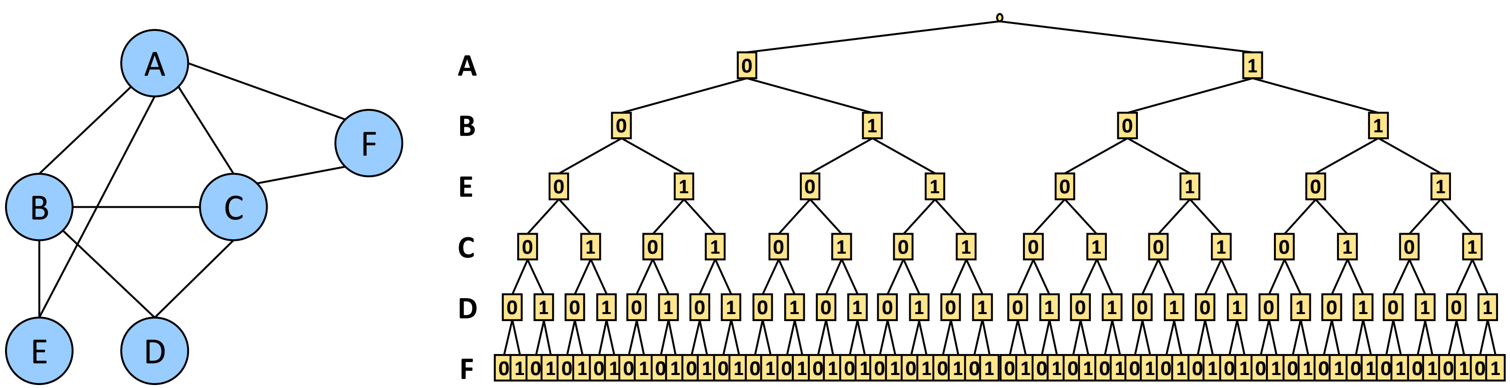


Bird's-eye View of Exact Algorithms



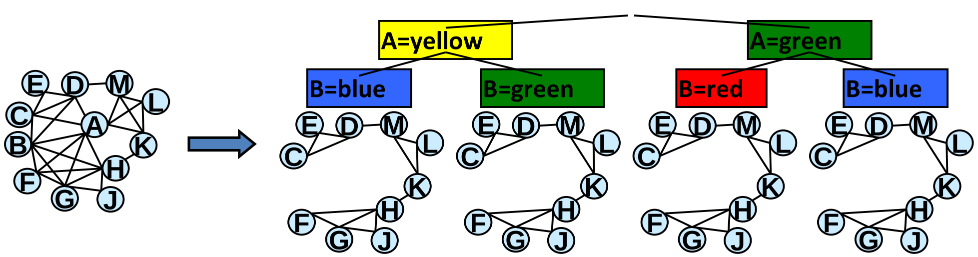
Inference

exp(w*) time/space



Search

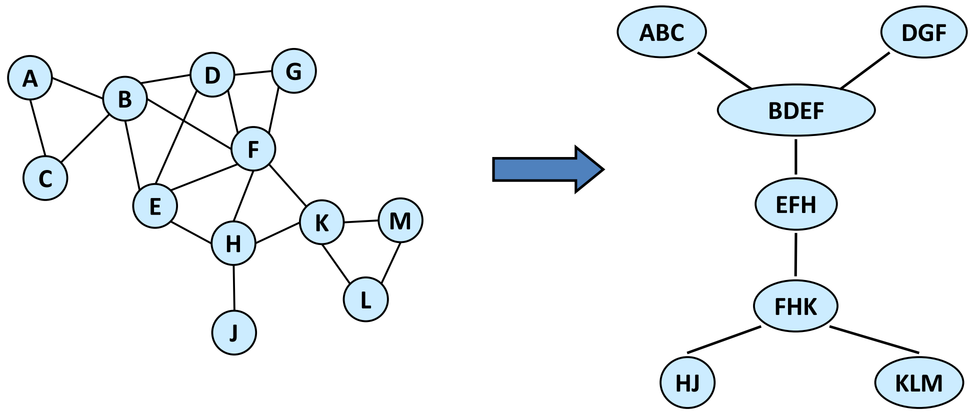
Exp(w*) time
O(w*) space



Search+inference:
Space: exp(q)
Time: exp(q+c(q))

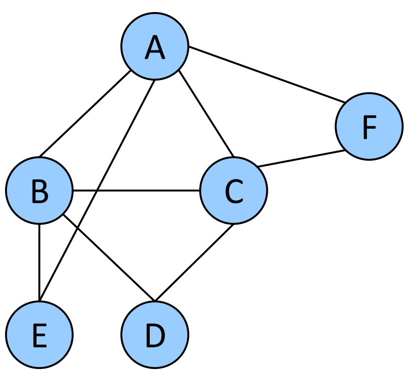
q: user controlled

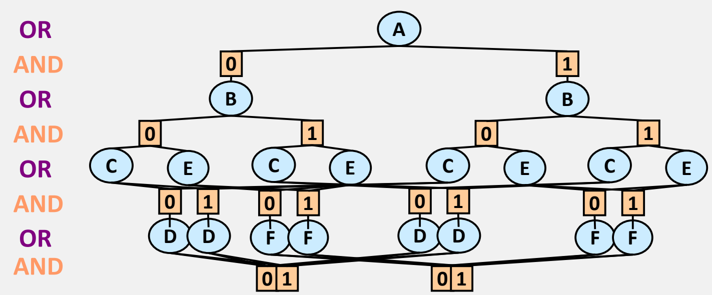
Bird's-eye View of Exact Algorithms



Inference

exp(w*) time/space





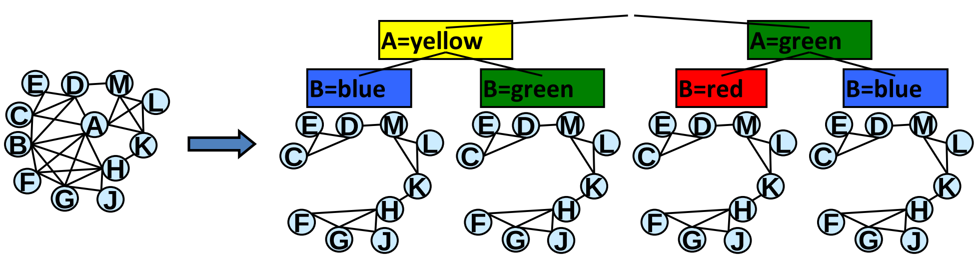
Context minimal AND/OR search graph

18 AND nodes

Search

Exp(w*) time

O(w*) space



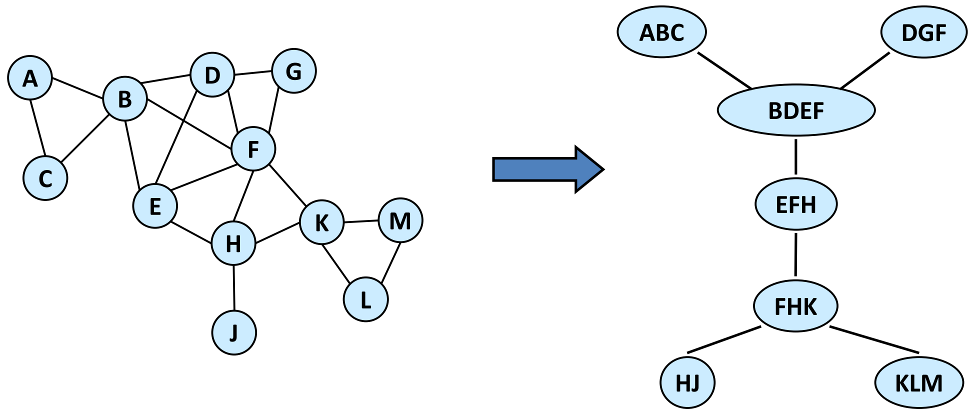
Search+inference:

Space: exp(q)

Time: exp(q+c(q))

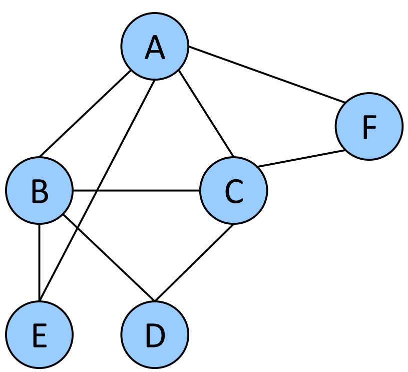
q: user controlled

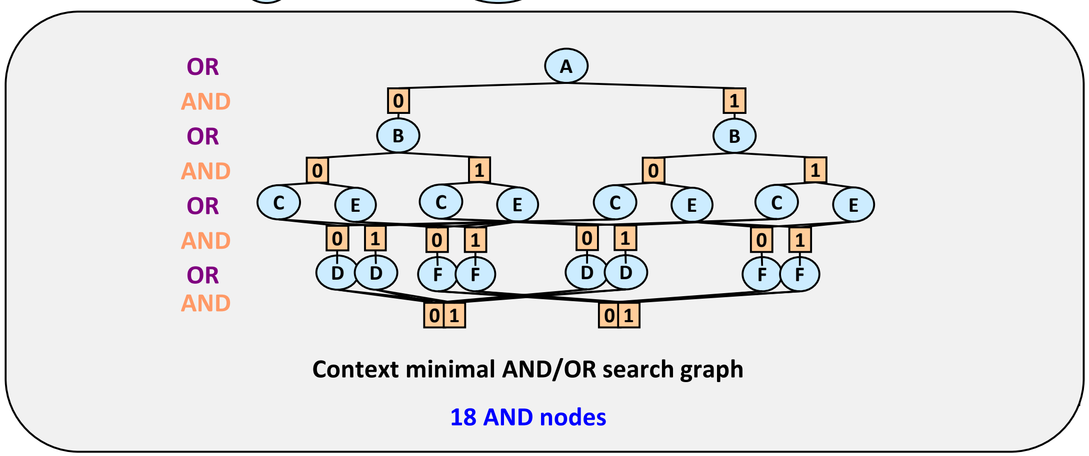
Bird's-eye View of Approximate Algorithms



Inference

Bounded Inference



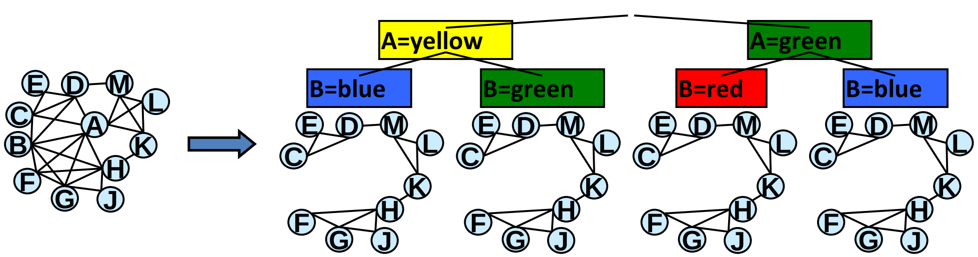


Context minimal AND/OR search graph

18 AND nodes

Search

Sampling



Search + inference:

Sampling + bounded inference

The Conditioning and Elimination Operators

Conditioning on Observations

- Observing a variable's value
  - Reduces the scope of the factor

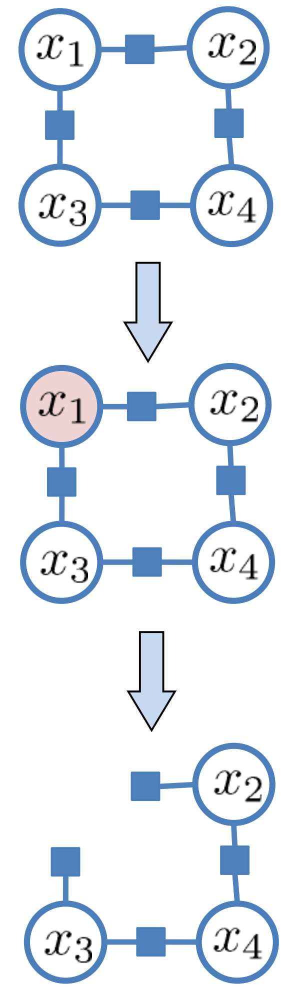
$$p(X) = \frac{1}{Z}\big[f_{12}(X_1,X_2)f_{13}(X_1,X_3)f_{24}(X_2,X_4)f_{34}(X_3,X_4)\big]$$

$$\Downarrow$$

$$p(x_1, X_{2:4}) = \frac{1}{Z}\big[f_{12}(x_1,X_2)f_{13}(x_1,X_3)f_{24}(X_2,X_4)f_{34}(X_3,X_4)\big]$$

$$\Downarrow$$

$$p(X_{2:4}|x_1) = \frac{1}{Z'}\big[\ g_2(X_2)\ \cdot\ g_3(X_3)\ \cdot\ f_{24}(X_2,X_4)f_{34}(X_3,X_4)\big]$$



Conditioning on Observations

A	B	f(A,B)
b	b	4
b	g	5
b	r	1
g	b	2
g	g	6
g	r	3
r	b	1
r	g	1
r	r	6

Assign **A=b**

B	g(B)
b	4
g	5
r	1

Assign **B=r**

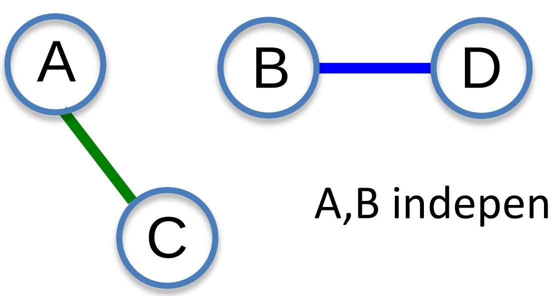
$h_{\varnothing}$
1

Conditional independence

- Undirected graphs have very simple conditional independence
  - X conditionally independent of Y given Z?
  - Check all paths from X to Y
  - A path is “inactive” (blocked) if it passes through a variable node in Z
  - If no path from X to Y, conditionally independent

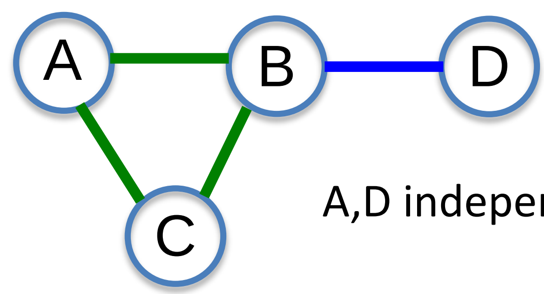
- Examples:

$$p(A)\ p(B)\, p(C \mid A)\, p(D \mid B)$$



A,B independent

$$p(A)\, p(B \mid A)\, p(C \mid A, B)\, p(D \mid B)$$



A,D independent given B

Markov blanket of X: set of variables directly connected to X

Combination of Factors

A	B	f(A,B)
b	b	0.4
b	g	0.1
g	b	0
g	g	0.5

●

B	C	f(B,C)
b	b	0.2
b	g	0
g	b	0
g	g	0.8

A	B	C	f(A,B,C)
b	b	b	0.1
b	b	g	0
b	g	b	0
b	g	g	0.08
g	b	b	0
g	b	g	0
g	g	b	0
g	g	g	0.4

= 0.1 x 0.8

Elimination in a Factor

A	**B**	**f(A,B)**
b	b	4
b	g	6
b	r	1
g	b	2
g	g	6
g	r	3
r	b	1
r	g	1
r	r	6

Elim = sum

Elim(**f**,**B**)

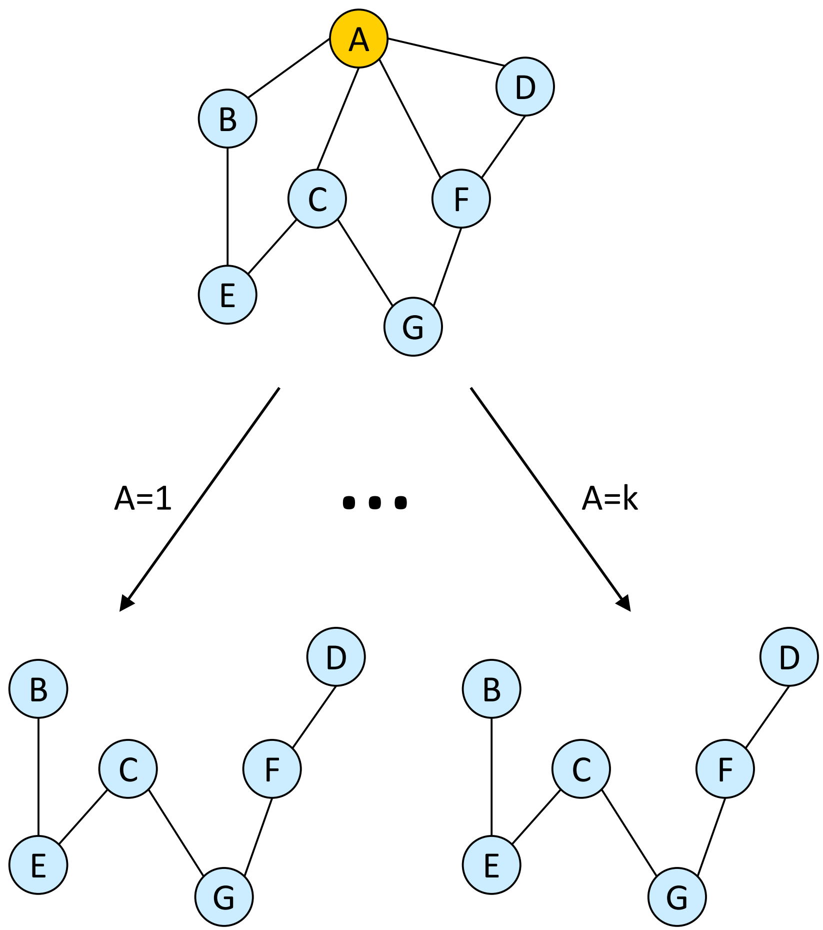
A	**g(A)**
b	11
g	11
r	8

Elim(**g**,**A**)

$\mathbf{h}_{\varnothing}$
30

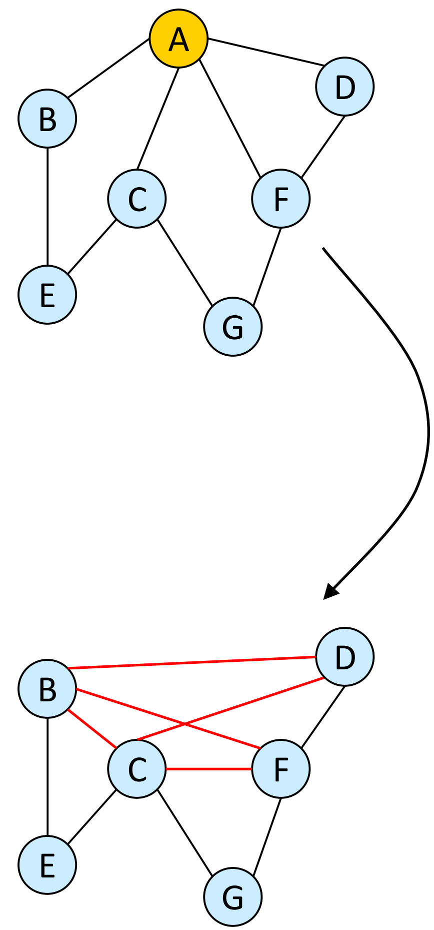
Conditioning versus Elimination

Conditioning (search)



k “sparser” problems

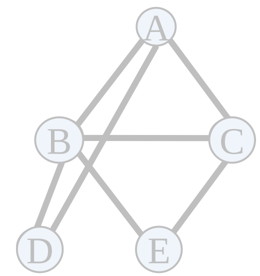
Elimination (inference)



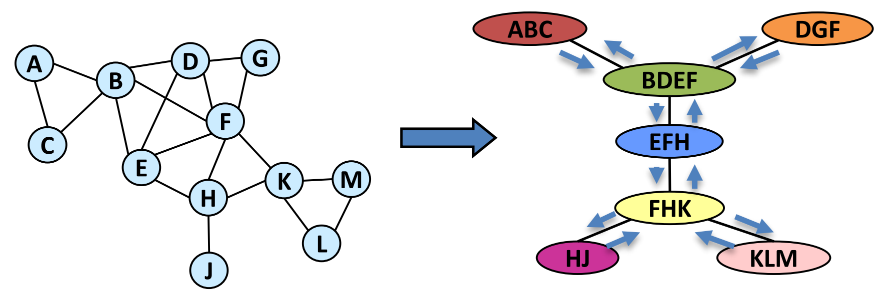
1 “denser” problem

RoadMap: Introduction and Inference

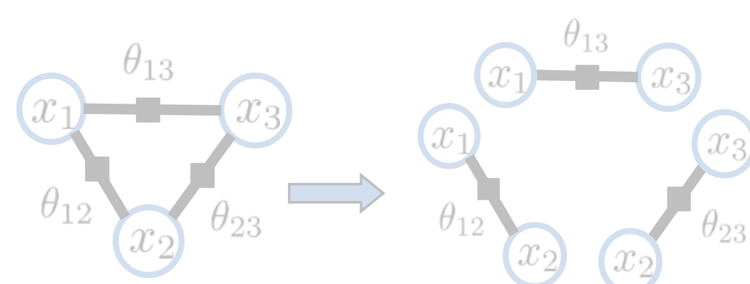
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - Algorithms overview



- Inference algorithms, exact
  - Bucket elimination for trees
  - Bucket elimination
  - Jointree clustering
  - Elimination orders



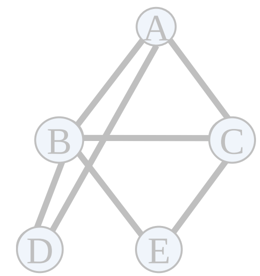
- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - Belief propagation



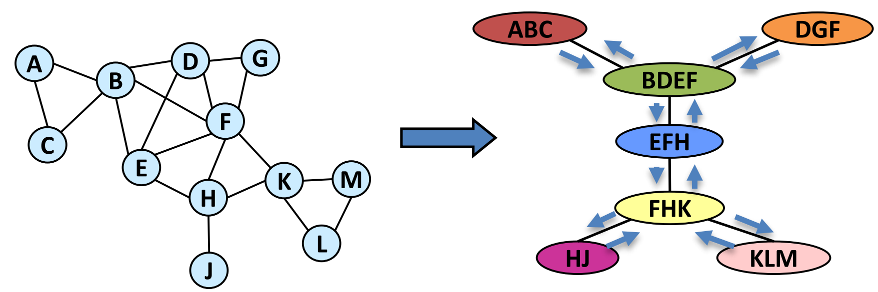
- Summary and Class 2

RoadMap: Introduction and Inference

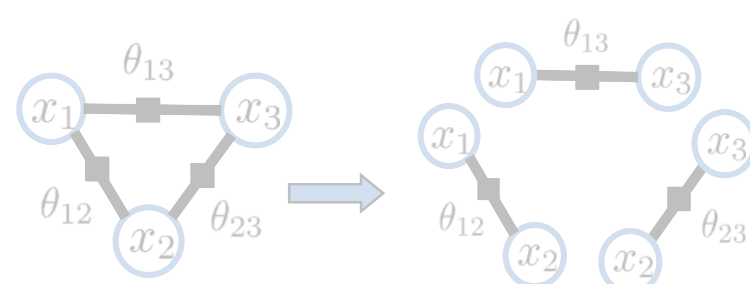
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - Algorithms overview



- Inference algorithms, exact
  - **Bucket elimination for trees**
  - Bucket elimination
  - Jointree clustering
  - Elimination orders



- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - Belief propagation



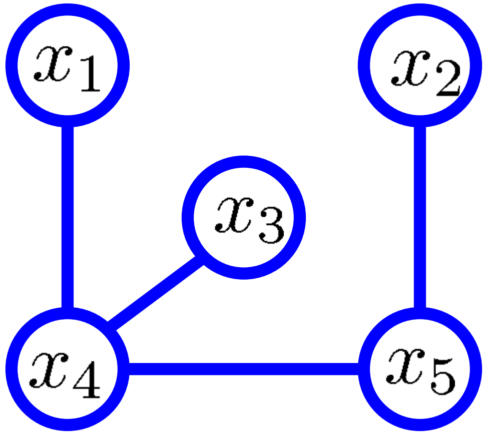
- Summary and Class 2

Variable Elimination in Trees

(Use distributive rule to calculate efficiently:)

$$\max_{x_1 \ldots x_5} f_{14}\ f_{25}\ f_{34}\ f_{45}$$

$$= \max_{x_2 \ldots x_5} f_{25}\ f_{34}\ f_{45} \Big[\max_{x_1} f_{14} \Big]$$



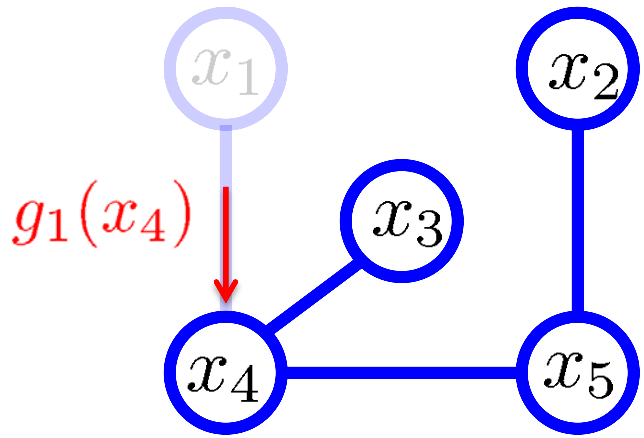
Variable Elimination in Trees

(Use distributive rule to calculate efficiently:)

$$\max_{x_1 \ldots x_5} f_{14}\ f_{25}\ f_{34}\ f_{45}$$

$$= \max_{x_2 \ldots x_5} f_{25}\ f_{34}\ f_{45} \Big[\max_{x_1} f_{14} \Big]$$

$$= \max_{x_3 \ldots x_5} f_{34}\ f_{45}\ g_1(x_4)\ \Big[\max_{x_2} f_{25} \Big]$$



Variable Elimination in Trees

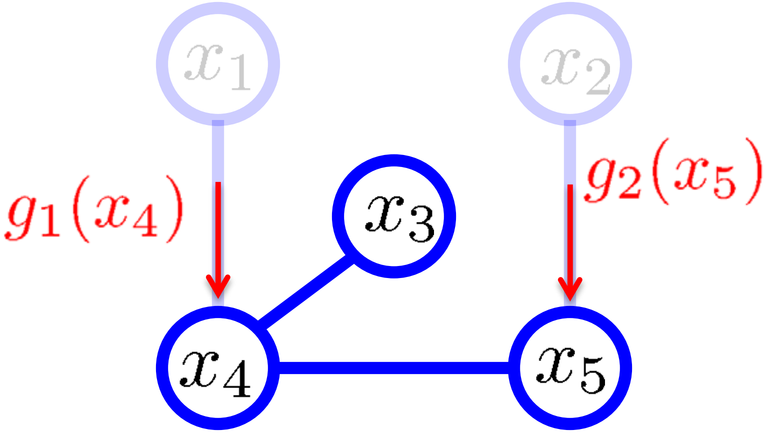
(Use distributive rule to calculate efficiently:)

$$\max_{x_1 \ldots x_5} f_{14}\ f_{25}\ f_{34}\ f_{45}$$

$$= \max_{x_2 \ldots x_5} f_{25}\ f_{34}\ f_{45} \Big[\max_{x_1} f_{14}\Big]$$

$$= \max_{x_3 \ldots x_5} f_{34}\ f_{45}\ g_1(x_4)\ \Big[\max_{x_2} f_{25}\Big]$$

$$= \max_{x_4 \ldots x_5} f_{45}\ g_1(x_4)\ g_2(x_5)\ \Big[\max_{x_3} f_{34}\Big]$$

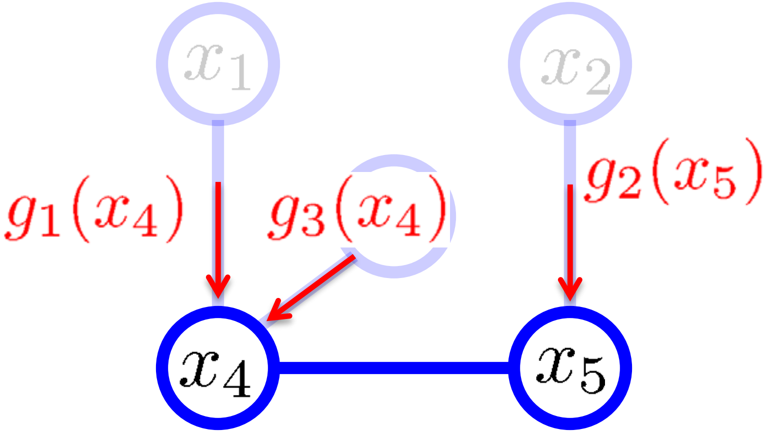


Variable Elimination in Trees

(Use distributive rule to calculate efficiently:)

$$\max_{x_1 \ldots x_5} f_{14}\ f_{25}\ f_{34}\ f_{45}$$

$$\begin{aligned}
&= \max_{x_2 \ldots x_5} f_{25}\ f_{34}\ f_{45} \left[\max_{x_1} f_{14}\right] \\
&= \max_{x_3 \ldots x_5} f_{34}\ f_{45}\ g_1(x_4)\ \left[\max_{x_2} f_{25}\right] \\
&= \max_{x_4 \ldots x_5} f_{45}\ g_1(x_4)\ g_2(x_5)\ \left[\max_{x_3} f_{34}\right] \\
&= \max_{x_5}\ g_2(x_5)\ \left[\max_{x_4} f_{45}\ g_1(x_4)\ g_3(x_4)\right]
\end{aligned}$$



Variable Elimination in Trees

(Use distributive rule to calculate efficiently:)

$$\max_{x_1 \dots x_5} f_{14}\ f_{25}\ f_{34}\ f_{45}$$

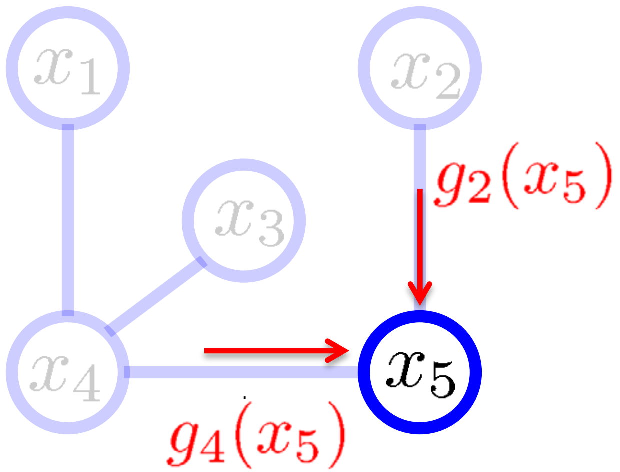
$$= \max_{x_2 \dots x_5} f_{25}\ f_{34}\ f_{45} \Big[\max_{x_1} f_{14}\Big]$$

$$= \max_{x_3 \dots x_5} f_{34}\ f_{45}\ g_1(x_4)\ \Big[\max_{x_2} f_{25}\Big]$$

$$= \max_{x_4 \dots x_5} f_{45}\ g_1(x_4)\ g_2(x_5)\ \Big[\max_{x_3} f_{34}\Big]$$

$$= \max_{x_5}\ g_2(x_5)\ \Big[\max_{x_4} f_{45}\ g_1(x_4)\ g_3(x_4)\Big]$$

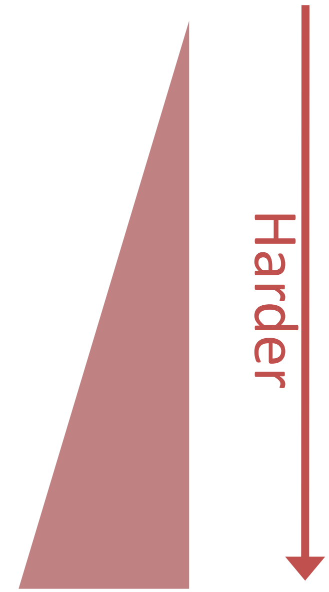
$$= \max_{x_5}\ g_2(x_5)\ g_4(x_5)$$



For trees:
Efficient elimination order (leaves to root);
computational complexity same as model size

Types of queries

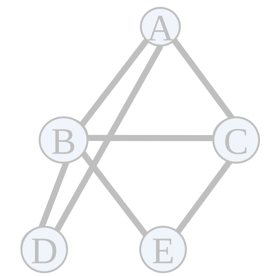
Max-Inference	$f(\mathbf{x}^*) = \max_{\mathbf{x}} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$
Sum-Inference	$Z = \sum_{\mathbf{x}} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$
Mixed-Inference	$f(\mathbf{x}_M^*) = \max_{\mathbf{x}_M} \sum_{\mathbf{x}_S} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$



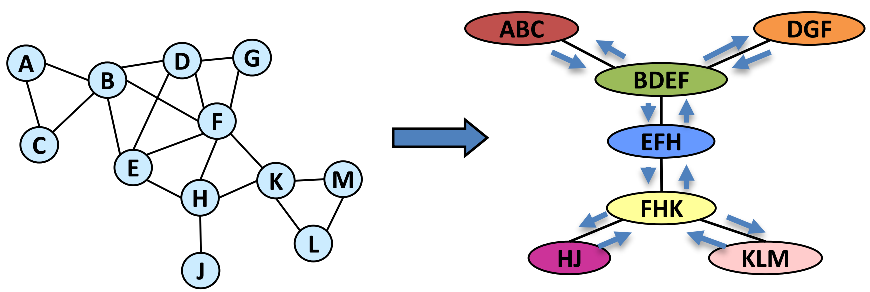
- **NP-hard**: exponentially many terms
- We will focus on **approximation** algorithms
  - **Anytime**: very fast & very approximate ! Slower & more accurate

RoadMap: Introduction and Inference

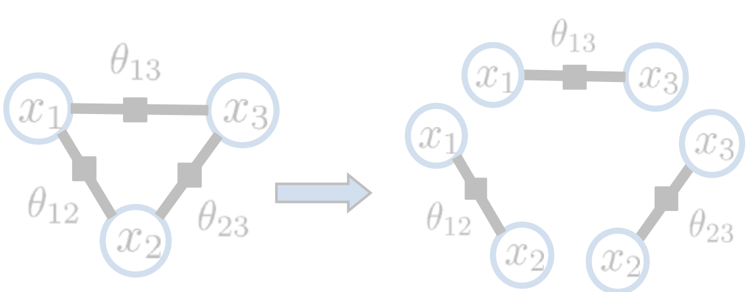
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - Algorithms overview



- Inference algorithms, exact
  - Bucket elimination for trees
  - **Bucket elimination**
  - Jointree clustering
  - Elimination orders



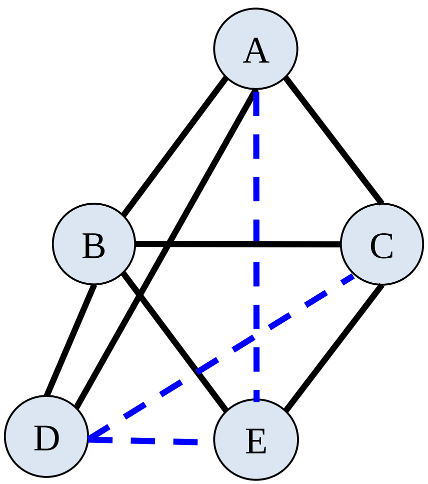
- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - Belief propagation



- Summary and Class 2

Belief Updating

- p(X | Evidence) = ?



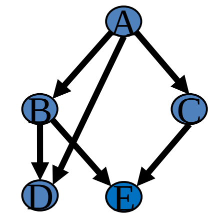
“primal” graph

$$p(A|E=0)$$
$$\propto p(A, E=0)$$
$$= \sum_{e,d,c,b} p(A)\, \underbrace{p(b|A)}\, p(c|A)\, \underbrace{p(d|b,A)\, p(e|b,c)}\, \mathbb{1}[e=0]$$

$$p(A) \sum_{e} \sum_{d} \sum_{c} p(c|A)\, \mathbb{1}[e=0] \underbrace{\sum_{b} p(b|A)\, p(d|b,A)\, p(e|b,c)}_{\lambda_{B\to C}(a,d,c,e)}$$

Variable Elimination

Bucket Elimination



Algorithm *BE-bel* [Dechter 1996]

$$p(A|E=0) = \alpha \sum_{e,d,c,b} p(A)\, p(b|A)\, p(c|A)\, p(d|A,b)\, p(e|b,c)\, \mathbb{1}[e=0]$$

$$\sum_b \prod$$ ← Elimination & combination operators

bucket B: $\overbrace{p(b|A)\;\; p(d|b,A)\;\; p(e|b,c)}$

bucket C: $p(c|A)\quad \lambda_{B\to C}(\underbrace{A,d,c,e})$

bucket D: $\lambda_{C\to D}(A,d,e)$

bucket E: $\mathbb{1}[E=0]\;\lambda_{D\to E}(A,e)$

bucket A: $p(A)\quad \lambda_{E\to A}(A)$

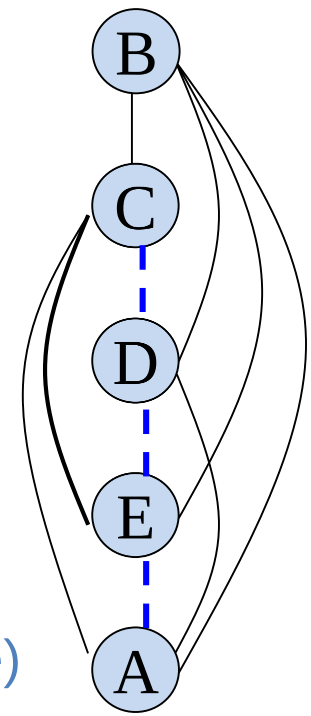
$p(E=0)$

$p(A|E=0) = p(A,E=0)\,/\,p(E=0)$

W*=4

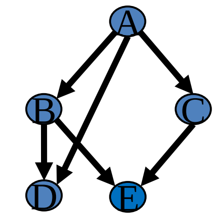
“induced width”

(max clique size)



Bucket Elimination

Algorithm *BE-bel* [Dechter 1996]



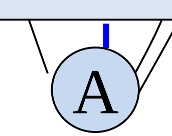
$$p(A|E=0) = \alpha \sum_{e,d,c,b} p(A)\, p(b|A)\, p(c|A)\, p(d|A,b)\, p(e|b,c)\, \mathbb{1}[e=0]$$

$\sum_b \prod$ ← Elimination & combination operators

Time and space exponential in the induced-width / treewidth

bucket A: $p(A)$ $\lambda_{E \to A}(A)$

(max clique size)

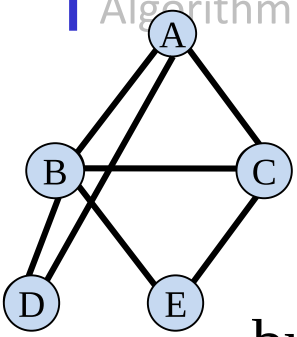


$p(E=0)$

$p(A|E=0) = p(A,E=0) \,/\, p(E=0)$

Finding MPE/MAP

Algorithm BE-mpe (Dechter 1996, Bertele and



$$\text{MPE} = \max_{a,e,d,c,b} p(a)\, p(c|a)\, p(b|a)\, p($$

$$= \max_b p(b|a) \cdot p(d|b,a) \cdot p(e|b,c)$$

$$\underbrace{\hspace{6cm}}_{\max_x \prod}$$

bucket B: $p(b|a)\ \ p(d|b,a)\ \ p(e|b,c)$

bucket C: $p(c|a)\quad \lambda_{B\to C}\underbrace{(a,d,c,e)}$

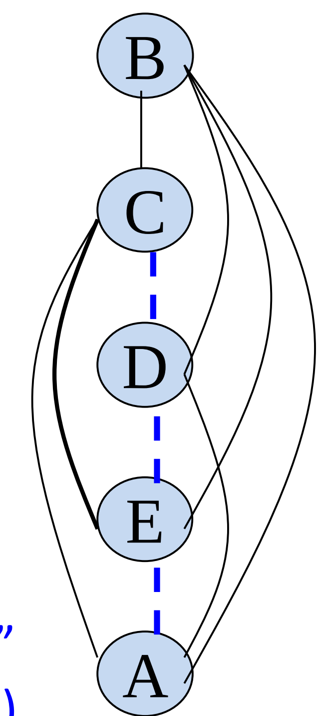
bucket D: $\lambda_{C\to D}(a,d,e)$

bucket E: $\mathbb{1}[e=0]\quad \lambda_{D\to E}(a,e)$

bucket A: $p(a)\quad \lambda_{E\to A}(a)$

OPT

W*=4
"induced width"
(max clique size)



Generating the Optimal Assignment

- Given BE messages, select optimum config in reverse order

$$\mathbf{b}^* = \arg\max_b \, p(b|a^*)\, p(d^*|b,a^*)\, p(e^*|b,c^*)$$

$$\mathbf{c}^* = \arg\max_c \, p(c|a^*)\, \lambda_{B\to C}(a^*,c,d^*,e^*)$$

$$\mathbf{d}^* = \arg\max_d \, \lambda_{C\to D}(a^*,d,e^*)$$

$$\mathbf{e}^* = \arg\max_e \, \mathbb{1}[e=0]\, \lambda_{D\to E}(a^*,e)$$

$$\mathbf{a}^* = \arg\max_a \, p(a)\cdot\lambda_{E\to A}(a)$$

Return optimal configuration (a^*,b^*,c^*,d^*,e^*)

B: $\underbrace{p(b|a)\;\; p(d|b,a)\;\; p(e|b,c)}$

C: $\underbrace{p(c|a) \qquad \lambda_{B\to C}(a,c,d,e)}$

D: $\underbrace{\lambda_{C\to D}(a,d,e)}$

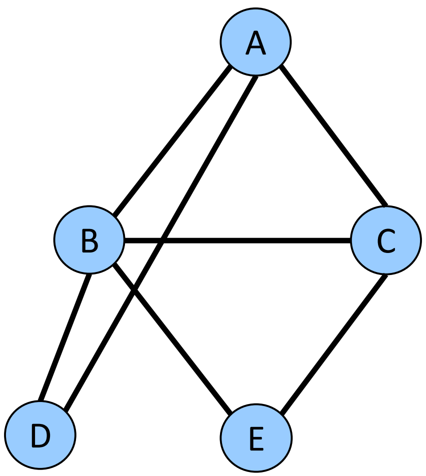
E: $\underbrace{\mathbb{1}[e=0] \quad \lambda_{D\to E}(a,e)}$

A: $\underbrace{p(a)\;\; \lambda_{E\to A}(a)}$

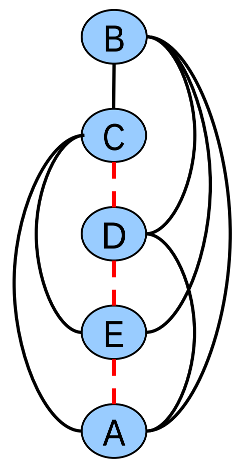
OPT = optimal value

Induced Width

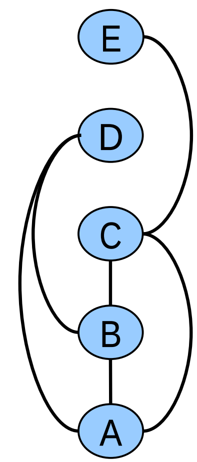
- Width is the max number of parents in the ordered graph
- Induced-width is the width of the induced ordered graph: recursively connecting parents going from last node to first.
- Induced-width $w^*(d)$ is the max induced-width over all nodes in ordering d
- Induced-width of a graph, w^* is the min $w^*(d)$ over all orderings d



primal graph



$w^*(d_1) = 4$



$w^*(d_2) = 2$

Complexity of Bucket Elimination

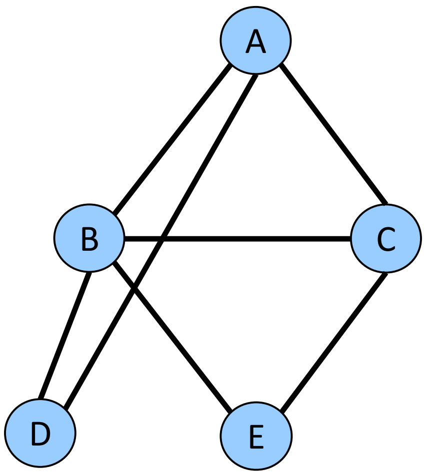
Bucket-Elimination is **time** and **space**

$$O\big(r \exp(w_d^*)\big)$$

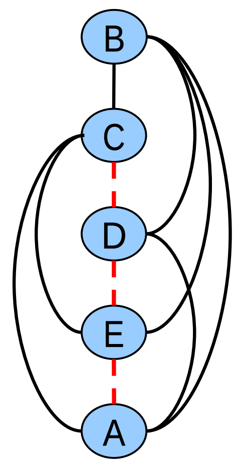
w_d^*: the induced width of the primal graph along ordering d

r = number of functions

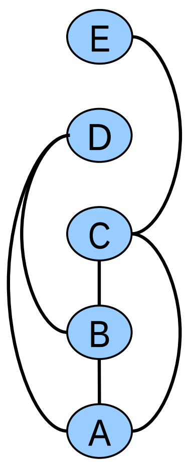
The effect of the ordering:



primal graph



$w^*(d_1) = 4$

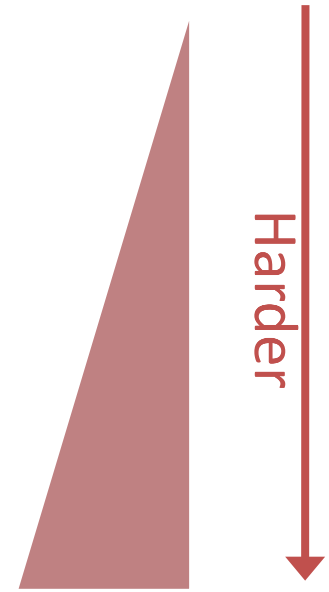


$w^*(d_2) = 2$

Finding smallest induced-width is hard!

Types of queries

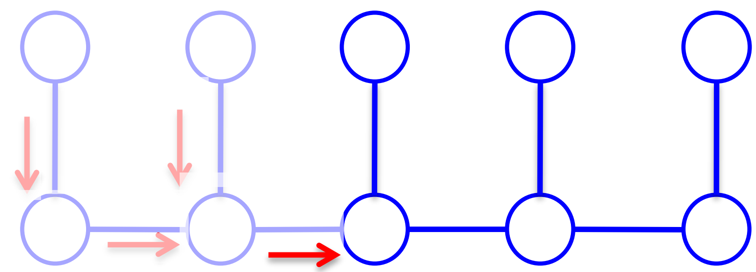
Max-Inference	$f(\mathbf{x}^*) = \max_{\mathbf{x}} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$
Sum-Inference	$Z = \sum_{\mathbf{x}} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$
Mixed-Inference	$f(\mathbf{x}_M^*) = \max_{\mathbf{x}_\mathrm{M}} \sum_{\mathbf{x}_S} \prod_{\alpha} f_\alpha(\mathbf{x}_\alpha)$



- **NP-hard**: exponentially many terms

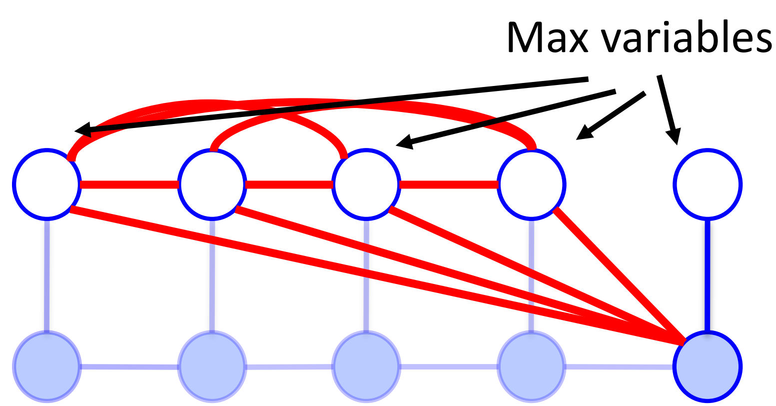
Marginal MAP is not easy on trees

- Pure MAP or summation tasks
  - Dynamic programming
  - Ex: efficient on trees



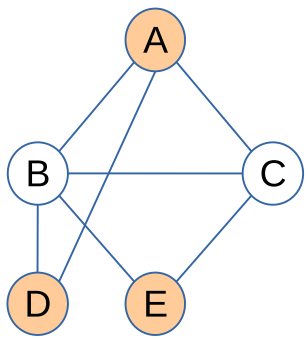
- Marginal MAP
  - Operations do not commute:
  - Sum must be done first!

$$\sum \max \neq \max \sum$$



Bucket Elimination for MMAP

Bucket Elimination



$$\mathbf{X}_M = \{A, D, E\}$$
$$\mathbf{X}_S = \{B, C\}$$

$$\max_{\mathbf{X}_M} \sum_{\mathbf{X}_S} P(\mathbf{X})$$

constrained elimination order

SUM

B: $f(A,B)\, f(B,C)\, f(B,D)\, f(B,E)$

Σ_B

C: $\lambda^B(A,C,D,E)\, f(A,C)\, f(C,E)$

Σ_C

MAX

D: $\lambda^C(A,D,E)\, f(A,D)$

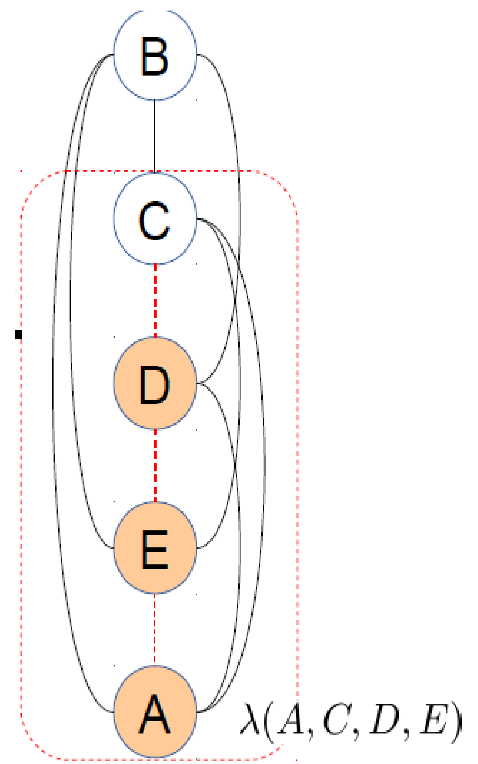
max_D

E: $\lambda^D(A,E)$

max_E

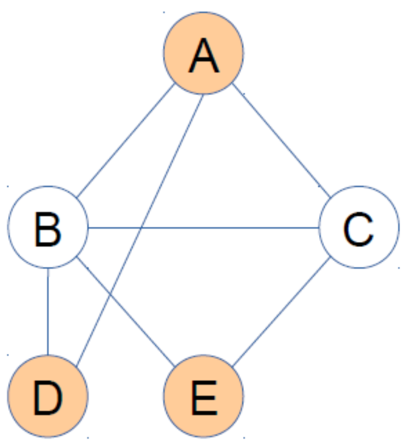
A: $\lambda^E(A)$

MAP* is the marginal MAP value



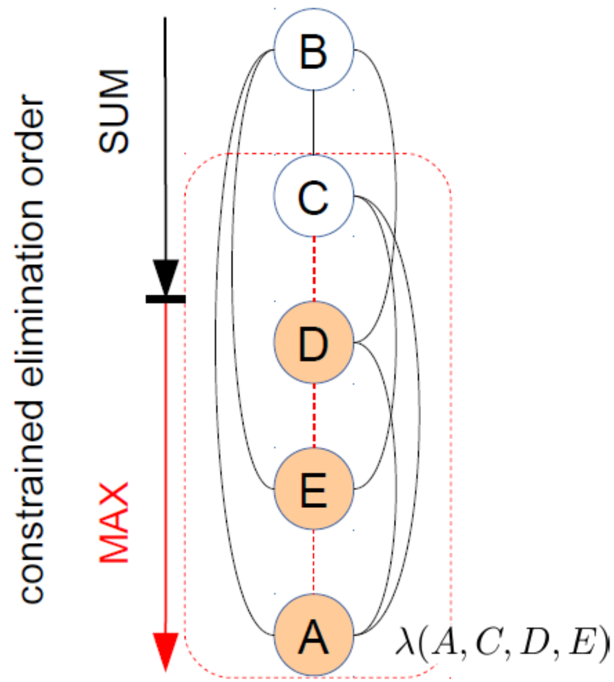
Bucket Elimination for MMAP

Impact of Orderings



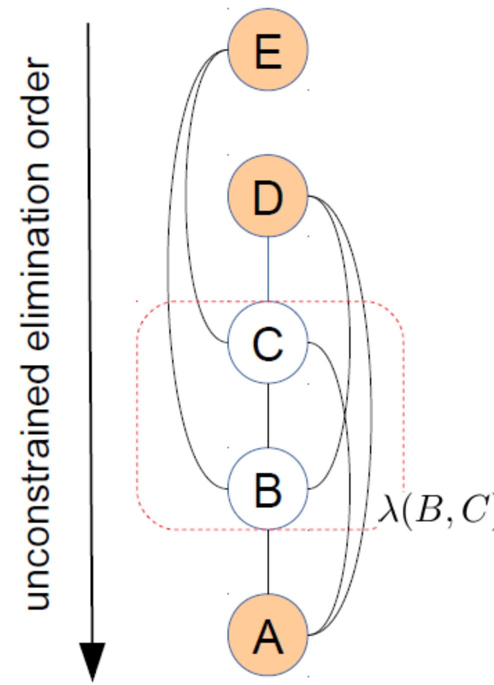
$$\mathbf{X}_M = \{A, D, E\}$$
$$\mathbf{X}_S = \{B, C\}$$

exact



$w^* = 4$

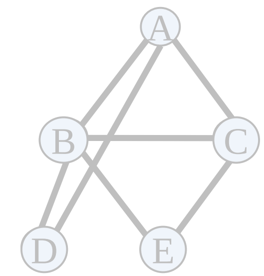
upper bound



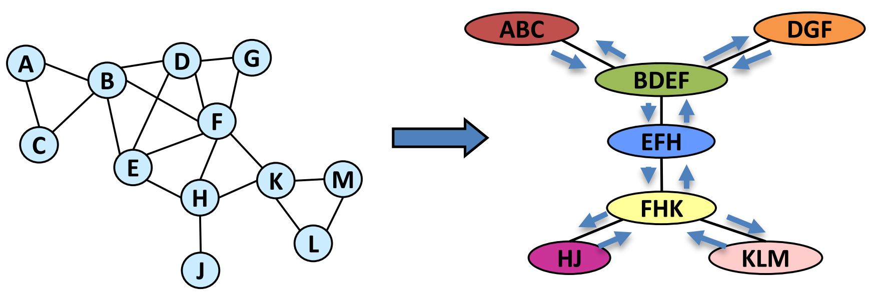
$w^* = 2$

RoadMap: Introduction and Inference

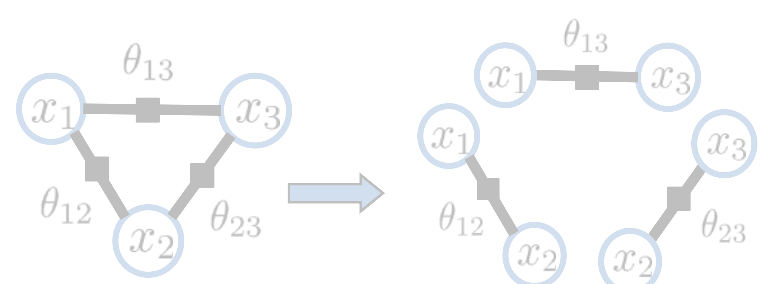
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - Algorithms overview



- Inference algorithms, exact
  - Bucket elimination for trees
  - Bucket elimination
  - **Jointree clustering**
  - Elimination orders

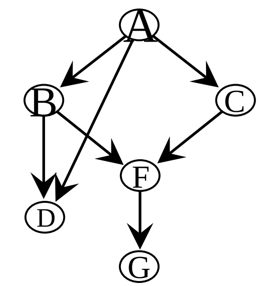


- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - Belief propagation



- Summary and Class 2

From BE to Bucket-Tree Elimination(BTE)



First, observe the BE operates on a tree.

Second, What if we want the marginal on D?

Bucket G: $P(G|F)$

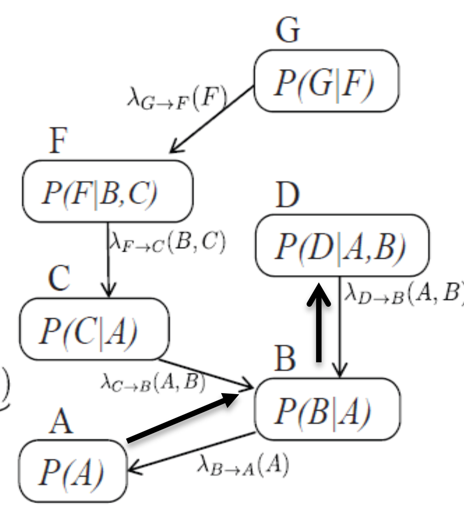
Bucket F: $P(F|B,C)$ $\lambda_{G\to F}(F)$

Bucket D: $P(D|A,B)$

Bucket C: $\underbrace{P(C|A) \quad \lambda_{F\to C}(B,C)}$

Bucket B: $\underbrace{P(B|A) \quad \lambda_{D\to B}(A,B)\, \lambda_{C\to B}(A,B)}$

Bucket A: $P(A)$ $\lambda_{B\to A}(A)$



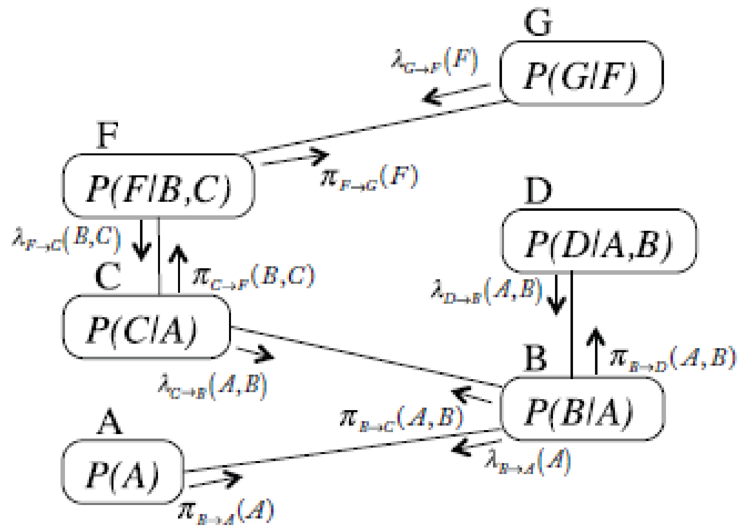
$$\pi_{A\to B}(a) = P(A),$$

$$\pi_{B\to D}(a,b) = p(b|a)\cdot \pi_{A\to B}(a)\cdot \lambda_{C\to B}(b)$$

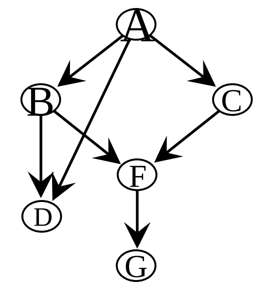
$$bel(d) = \alpha \sum_{a,b} P(d|a,b)\cdot \pi_{B\to D}(a,b).$$

BTE: Allows Messages Both Ways

Initial buckets
+ messages

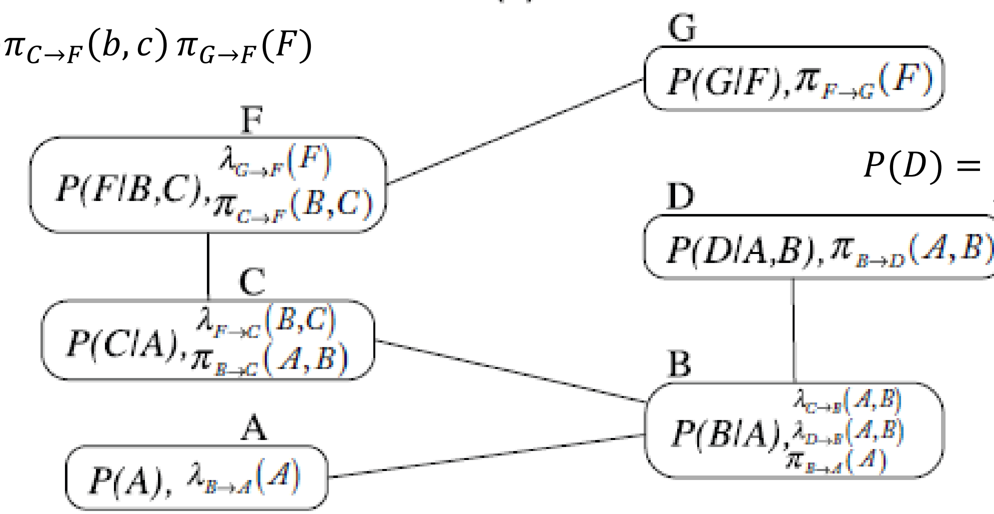


(a)



$$P(F) = \sum_{b,c} P(F|b,c)\pi_{C\to F}(b,c)\,\pi_{G\to F}(F)$$

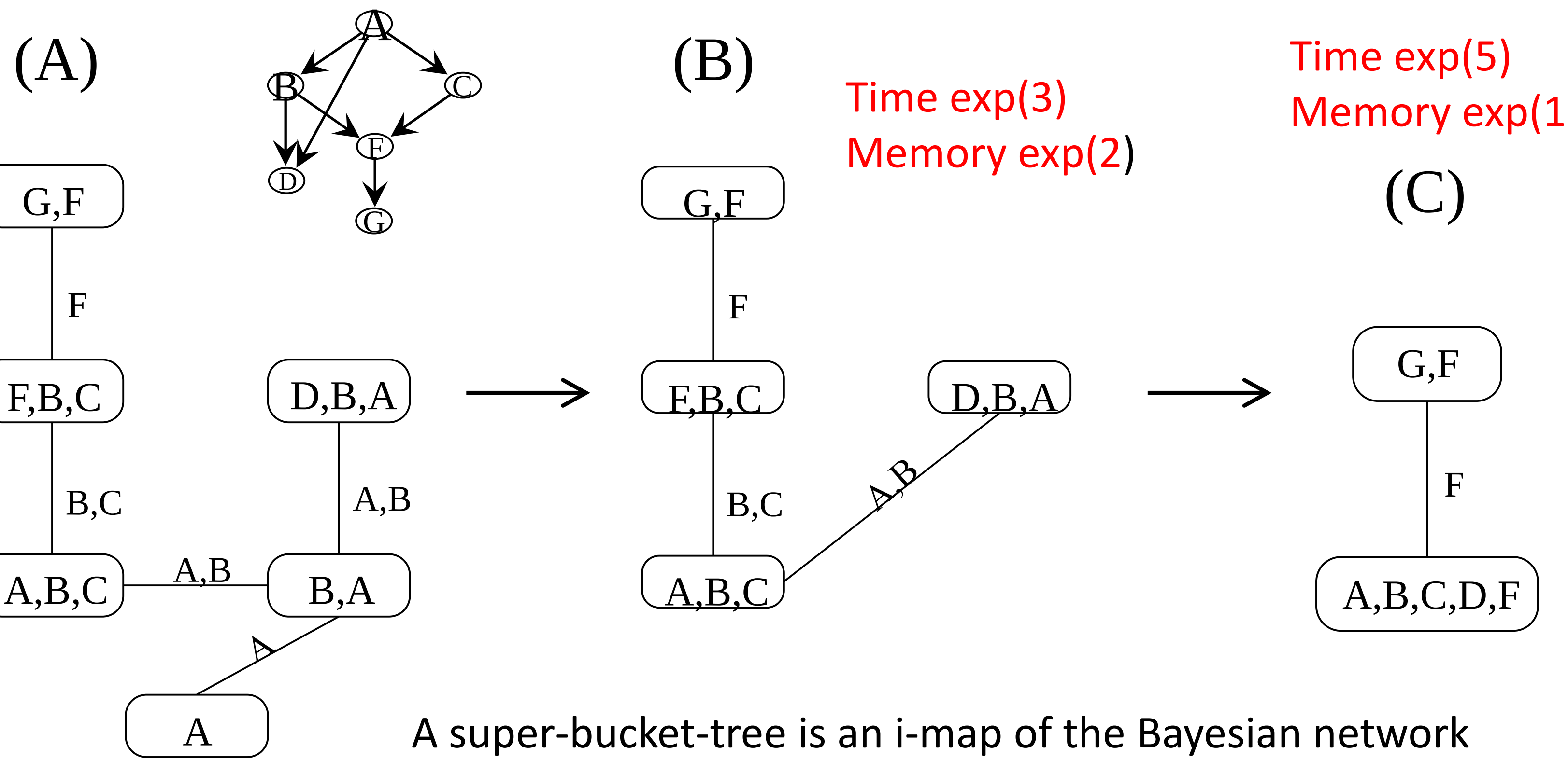
$$P(D) = \sum_{a,b} P(D|a,b)\,\pi_{B\to D}(a,b)$$



(b)

From Buckets to Clusters

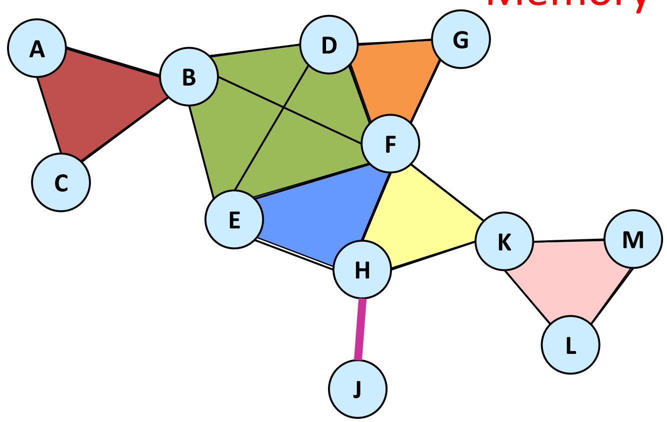
- Merge non-maximal buckets into maximal clusters.
- Connect clusters into a tree: each cluster to one with which it shares a largest subset of variables.
- Separators are variable- intersection on adjacent clusters.

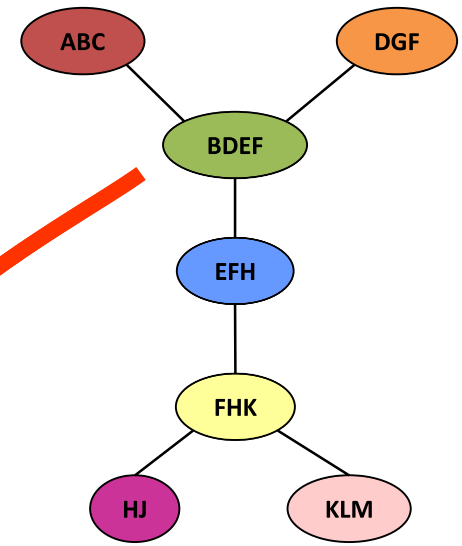


A super-bucket-tree is an i-map of the Bayesian network

The General Tree-Decomposition

We move
Bucket tree to general
Claster tree to trade
Memory for time

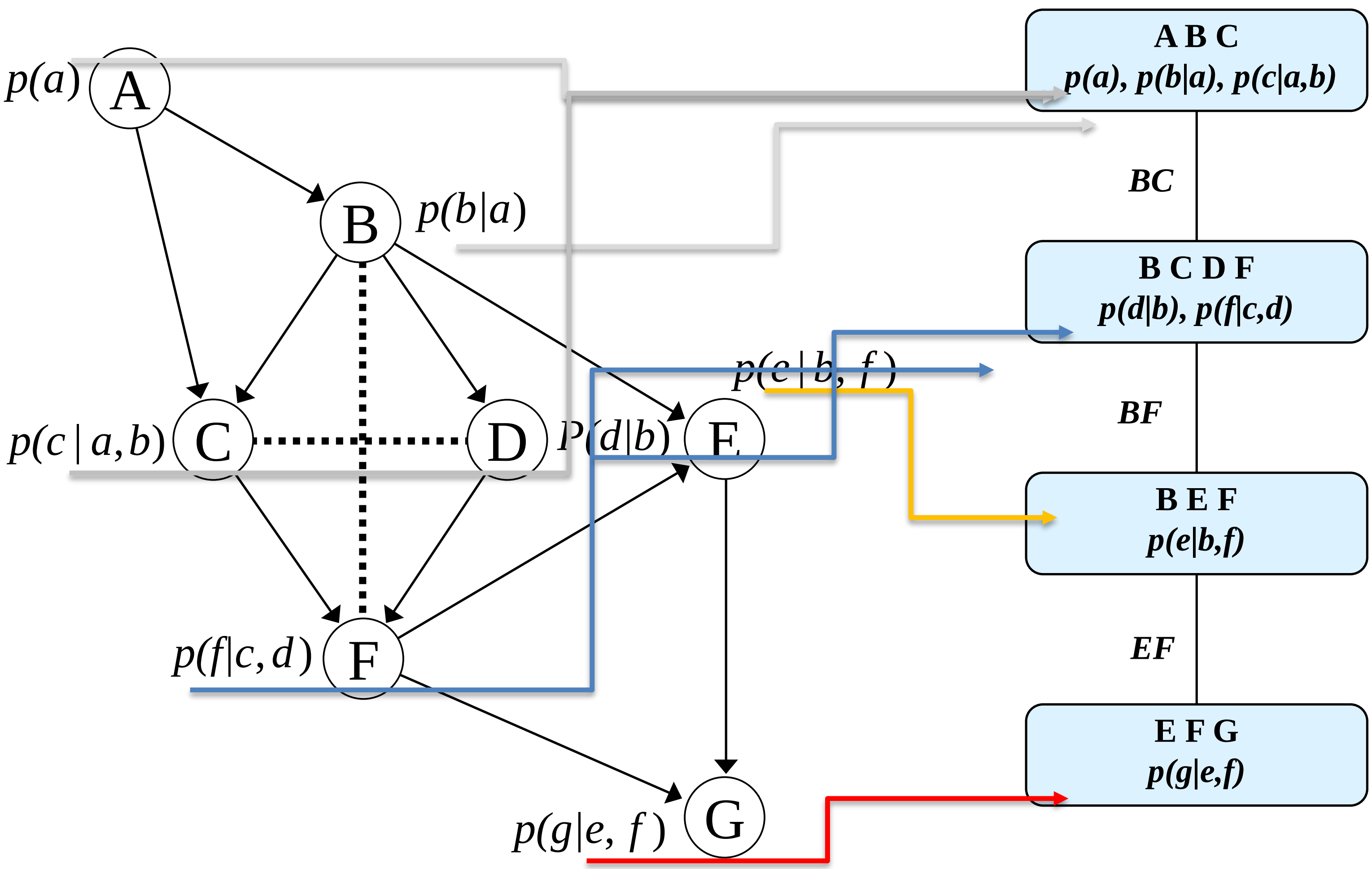




Inference algorithm:
Time: exp(tree-width)
Space: exp(tree-width)

treewidth = 4 - 1 = 3
treewidth = (maximum cluster size) - 1

Example of a Tree Decomposition

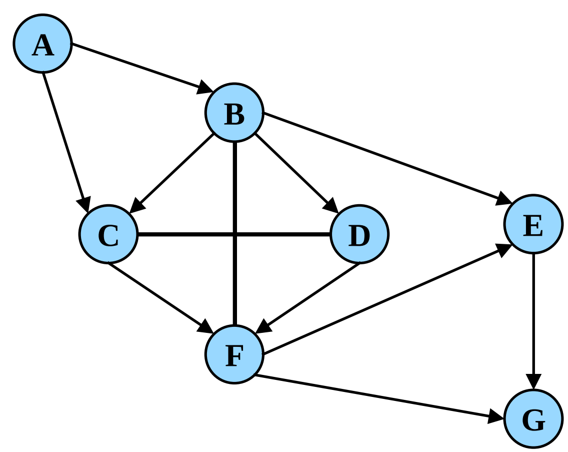


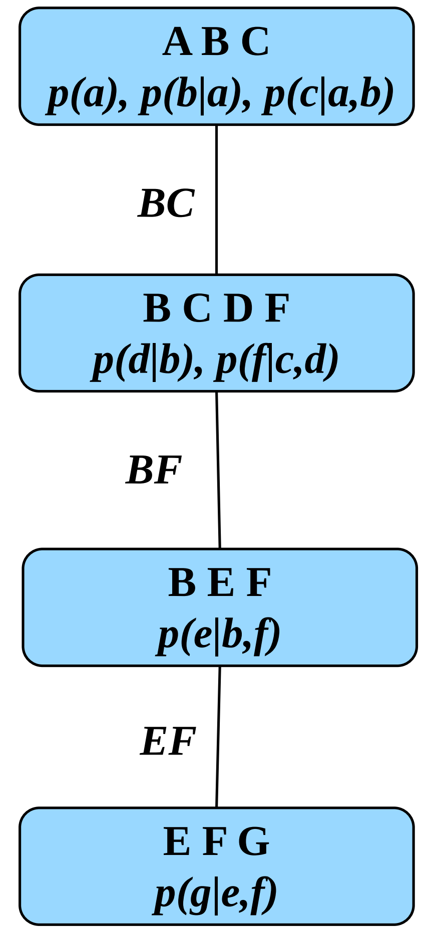
Tree Decompositions

A *tree decomposition* for a graphical model $< X,D,P >$ is a triple $< T, \chi, \psi >$, where $T = (V,E)$ is a tree and χ and ψ are labeling functions, associating with each vertex $v \in V$ two sets, $\chi(v) \subseteq X$ and $\psi(v) \subseteq P$ satisfying :

1. For each function $p_i \in P$ there is exactly one vertex such that $p_i \in \psi(v)$ and $scope(p_i) \subseteq \chi(v)$
2. For each variable $X_i \in X$ the set $\{v \in V | X_i \in \chi(v)\}$ forms a connected subtree (running intersection property)

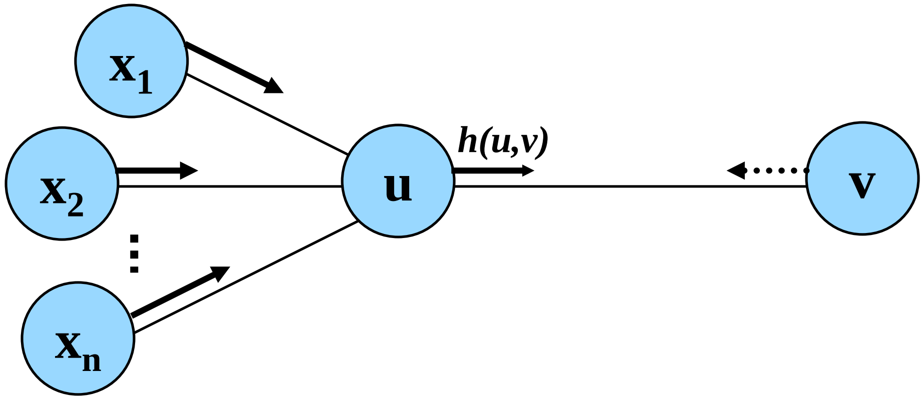
Connectedness, or
Running intersection property





Tree decomposition

Message passing on a tree decomposition



$$cluster(u) = \psi(u) \cup \{h(x_1,u), h(x_2,u), \ldots, h(x_n,u), h(v,u)\}$$

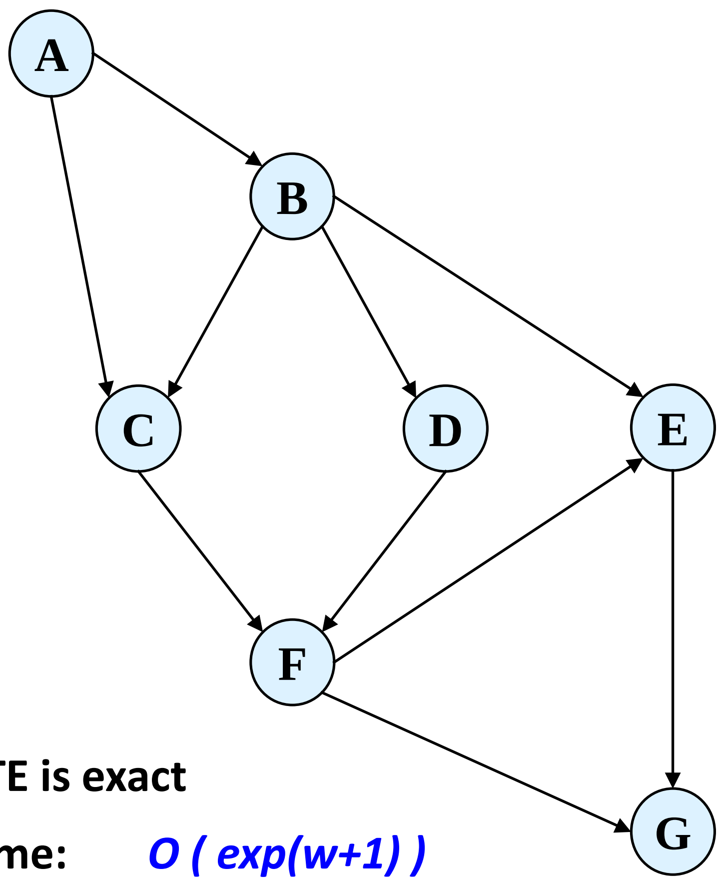
For max-product
Just replace $\sum$
With max.

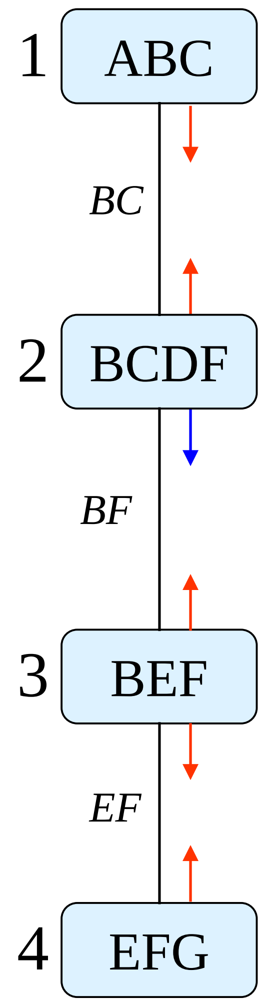
Compute the message :

$$h(u,v) = \sum_{elim(u,v)} \prod_{f \in cluster(u) - \{h(v,u)\}} f$$

Elim(u,v) = cluster(u)-sep(u,v)

Cluster-Tree Elimination (CTE), or Join-Tree Message-passing





$$h_{(1,2)}(b,c) = \sum_{a} p(a) \cdot p(b \mid a) \cdot p(c \mid a,b)$$

$$h_{(2,1)}(b,c) = \sum_{d,f} p(d \mid b) \cdot p(f \mid c,d) \cdot h_{(3,2)}(b,f)$$

$$h_{(2,3)}(b,f) = \sum_{c,d} p(d \mid b) \cdot p(f \mid c,d) \cdot h_{(1,2)}(b,c)$$

$$h_{(3,2)}(b,f) = \sum_{e} p(e \mid b,f) \cdot h_{(4,3)}(e,f)$$

$$h_{(3,4)}(e,f) = \sum_{b} p(e \mid b,f) \cdot h_{(2,3)}(b,f)$$

$$h_{(4,3)}(e,f) = p(G = g_e \mid e,f)$$

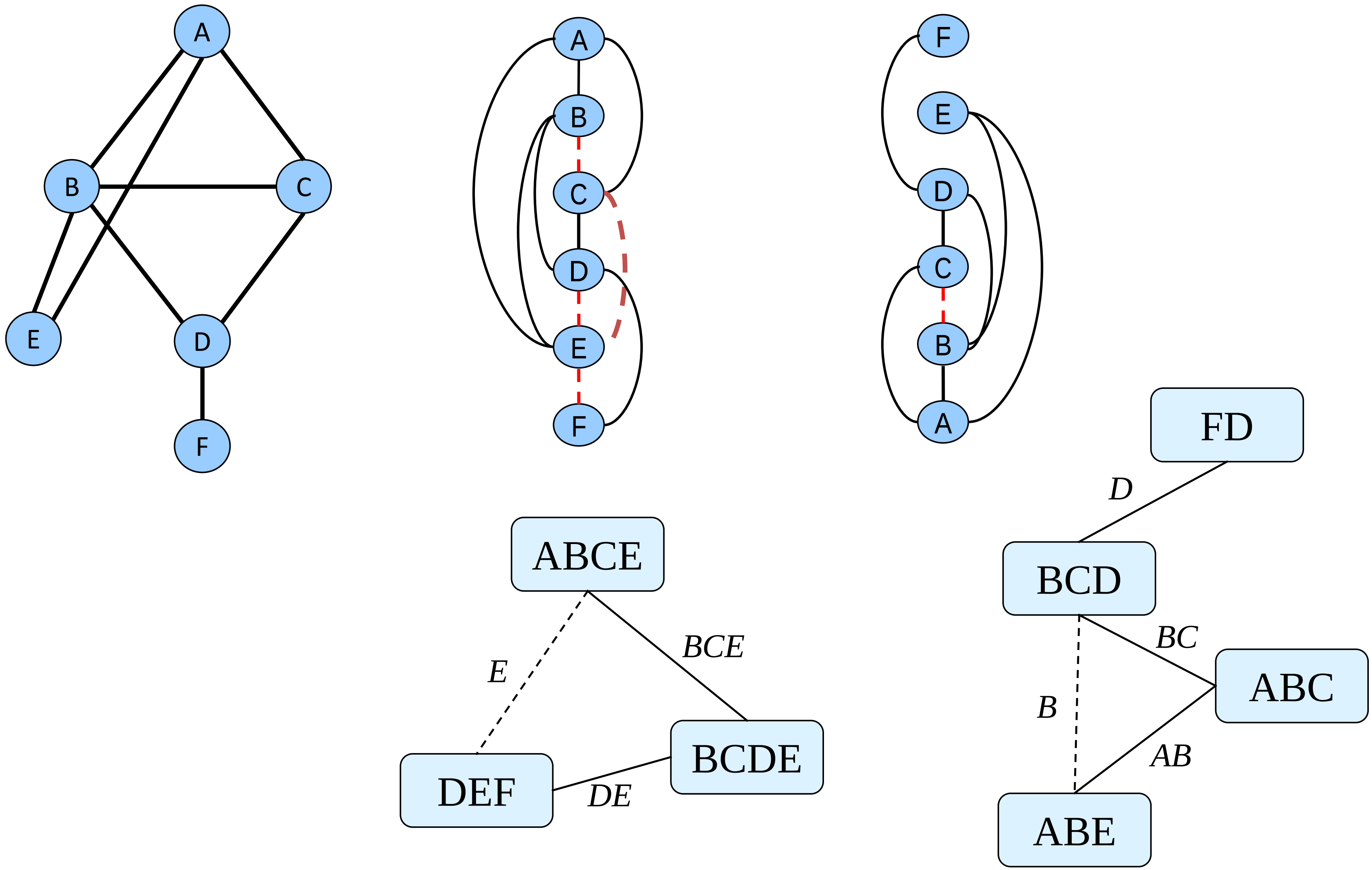
CTE is exact

Time: ***O (exp(w+1))***

Space: ***O (exp(sep))***

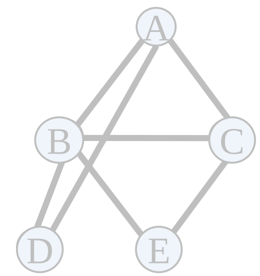
For each cluster P(X|e) is computed, also P(e)

Examples of (Join)-Trees Construction

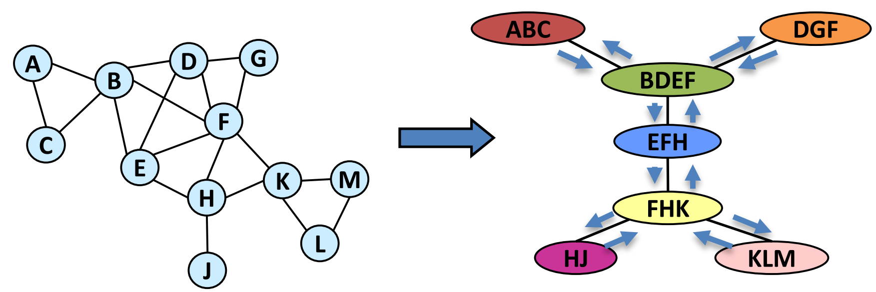


RoadMap: Introduction and Inference

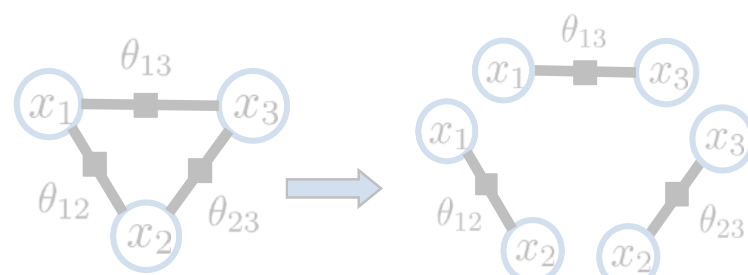
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - Algorithms overview



- Inference algorithms, exact
  - Bucket elimination for trees
  - Bucket elimination
  - Jointree clustering
  - **Elimination orders**



- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - Belief propagation



- Summary and Class 2

Finding a Small Induced-Width

- 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)
- Anytime algorithms
  - Search-based [Gogate & Dechter 2003]
  - Stochastic (CVO) [Kask, Gelfand & Dechter 2010]

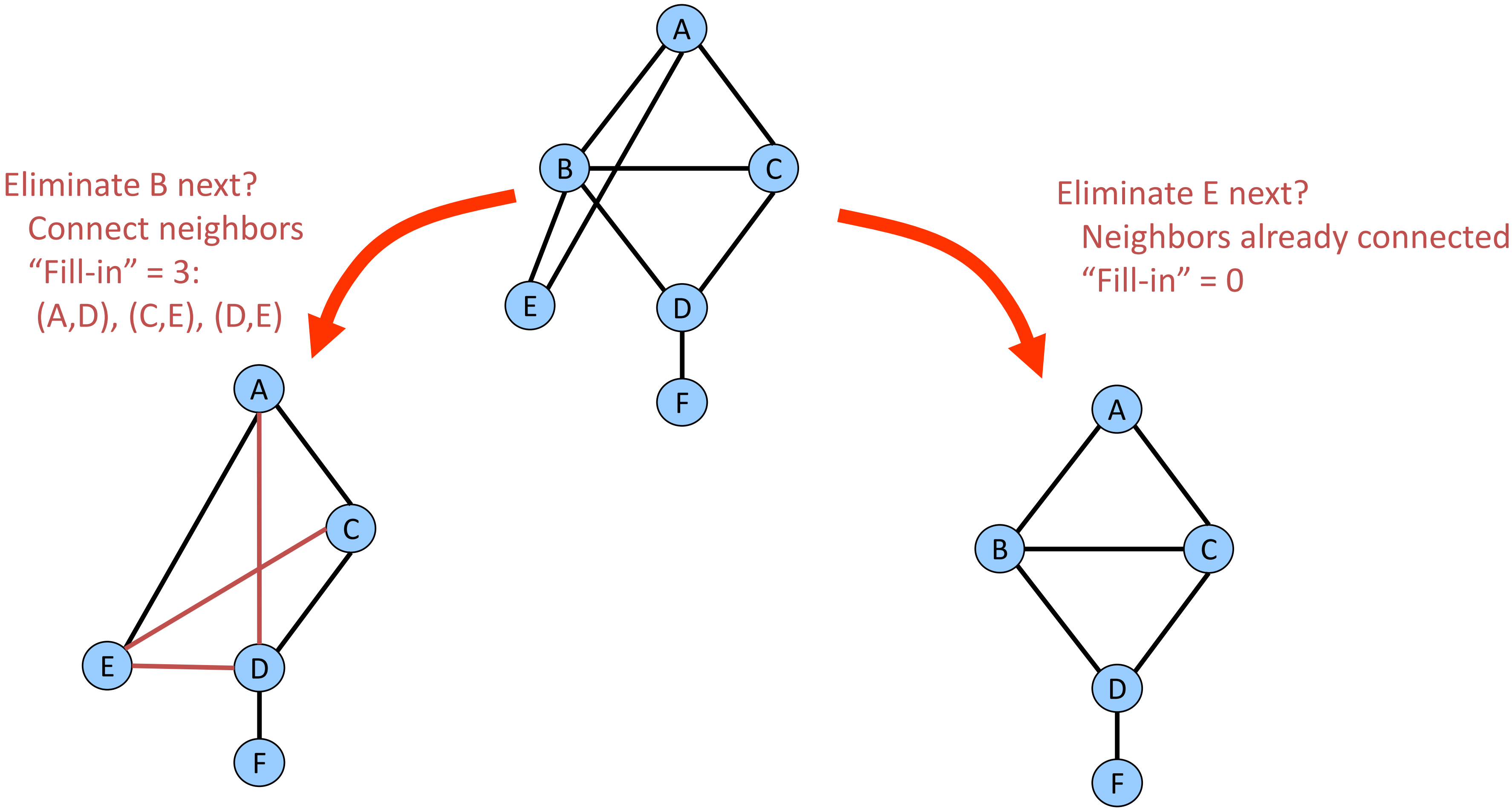
Greedy Orderings Heuristics

- Min-induced-width
  - From last to first, pick a node with smallest width
- Min-Fill
  - From last to first, pick a node with smallest fill-edges

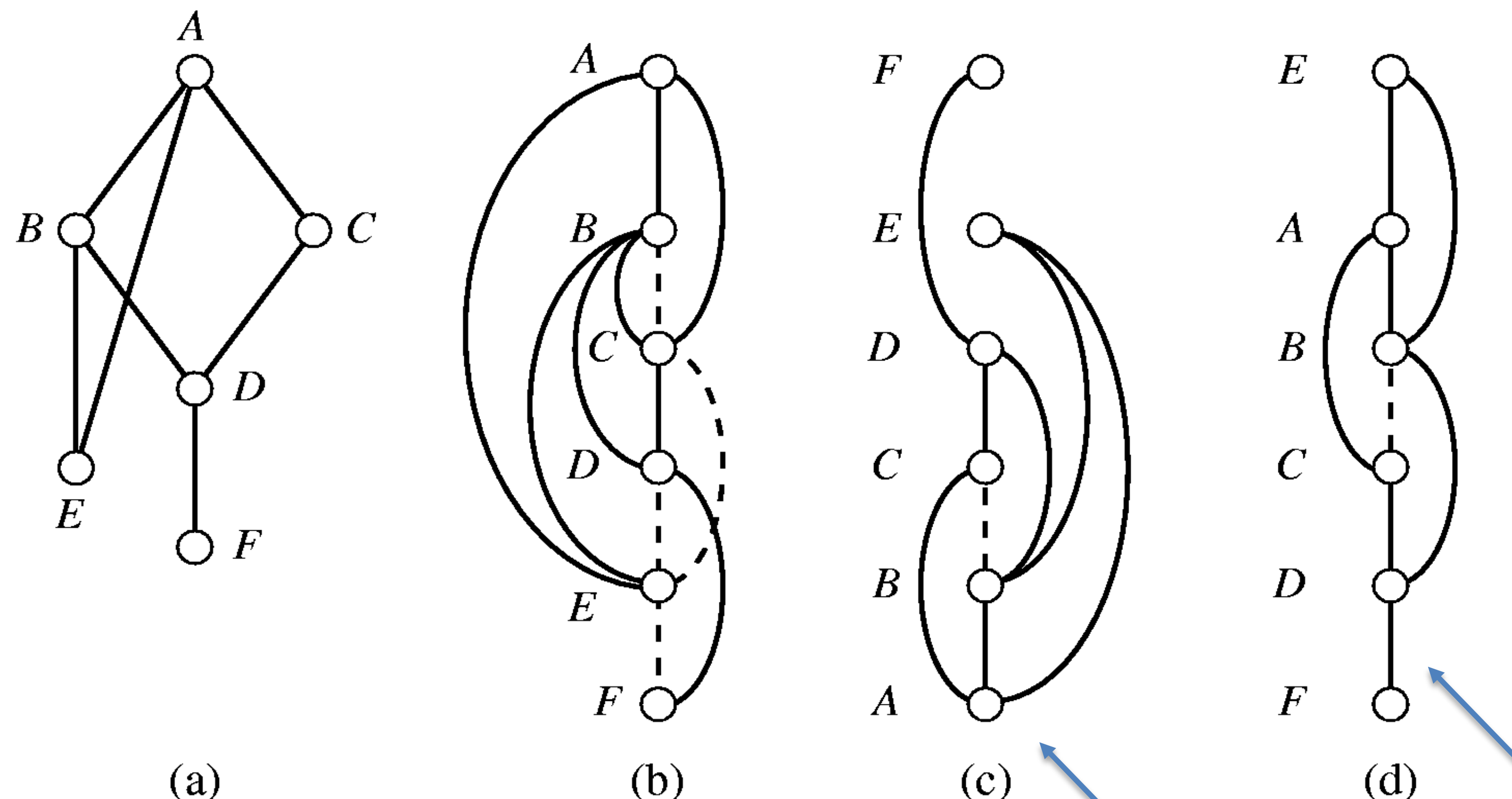
Complexity? O(n^3)

Min-Fill Heuristic

- Select the variable that creates the fewest “fill-in” edges



Different Induced Graphs

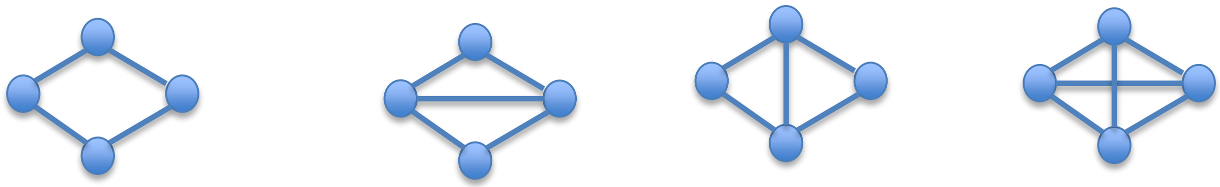


A Min-IW ordering

A Min-fill ordering

Chordal Graphs

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



- Deciding chordality by max-cardinality ordering:
  - from 1 to n, always assigning a next node connected to a largest set of previously selected nodes.

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

- The maximal cliques of chordal graphs form a tree

[Tarjan & Yanakakis 1980]

Greedy Orderings Heuristics

- Min-induced-width
  - From last to first, pick a node with smallest width
- Min-Fill
  - From last to first, pick a node with smallest fill-edges

  Complexity? O(n^3)
- Max-cardinality search
  - From first to last, pick a node with largest neighbors already ordered.

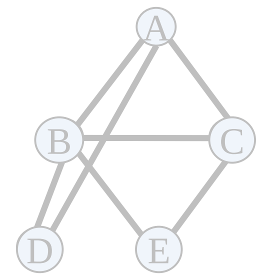
  Complexity? O($n + m$)

Summary Of Inference Scheme

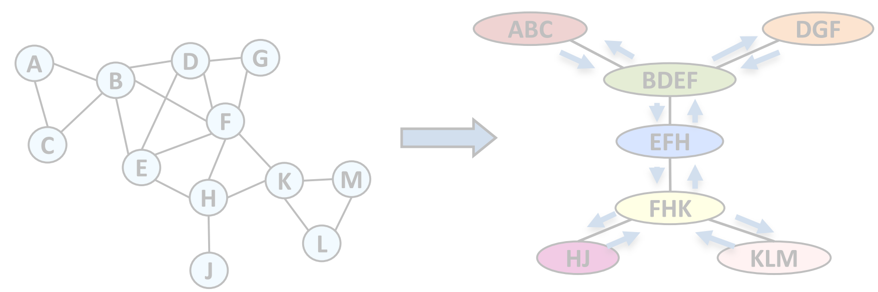
- Bucket elimination is time and memory exponential in the induced-width.
- Join-tree (junction tree) clustering is time O(exp(w*)) and memory O(exp(sep)).
- Bothe solve exactly all queries.
- Finding the w* is hard, but greedy schemes work quit well to approximate. Most popular is fill-edges
- W along d is induced-width. Best induced-width is tree-width.

RoadMap: Introduction and Inference

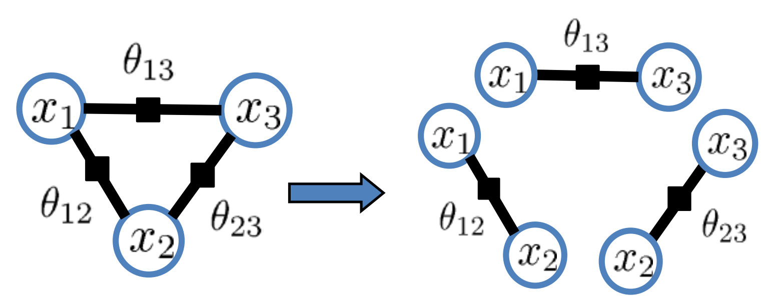
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - Algorithms overview



- Inference algorithms, exact
  - Bucket elimination for trees
  - Bucket elimination
  - Jointree clustering
  - Elimination orders



- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - Belief propagation



- Summary and Class 2

Decomposition bounds

- Upper & lower bounds via approximate problem decomposition
- Example: MAP inference $F(x) = f_1(x) + f_2(x)$

X	F (X)
0	2.0
1	4.0
2	5.0
3	4.0

=

X	f_1(X)
0	1.0
1	2.0
2	3.0
3	4.0

+

X	f_2(X)
0	1.0
1	2.0
2	2.0
3	0.0

$$\max_x F(x) \quad = \quad \max_x \Big[f_1(x) + f_2(x) \Big]$$

$$5.0 \qquad \leq \quad \Big[\max_x f_1(x) \; + \; \max_x f_2(x) \Big] \quad = 4.0 + 2.0 = 6.0$$

- Relaxation: two "copies" of x, no longer required to be equal
- Bound is tight (equality) if f_1, f_2 agree on maximizing value x

Mini-Bucket Approximation

Split a bucket into mini-buckets —> bound complexity

bucket (X) =

$$\left\{ \underbrace{f_1,\ f_2,\ \ldots\ f_r,\ f_{r+1},\ \ldots\ f_n}_{} \right\}$$

$$\lambda_X(\cdot) = \max_x \prod_{i=1}^{n} f_i(x,\ldots)$$

$$\left\{ f_1,\ \ldots\ f_r \right\} \qquad \left\{ f_{r+1},\ \ldots\ f_n \right\}$$

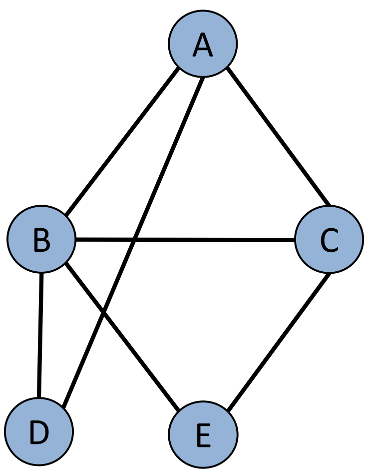
$$\lambda_{X,1}(\cdot) = \max_x \prod_{i=1}^{r} f_i(x,\ldots) \qquad \lambda_{X,2}(\cdot) = \max_x \prod_{i=r+1}^{n} f_i(x,\ldots)$$

$$\lambda_X(\cdot) \ \le\ \lambda_{X,1}(\cdot)\ \lambda_{X,2}(\cdot)$$

Exponential complexity decrease: $O(e^n) \longrightarrow O(e^r) + O(e^{n-r})$

Mini-Bucket Elimination

[Dechter & Rish 2003]



mini-buckets

bucket B: $\underbrace{f(a,b)\ f(b,c)}\quad \underbrace{f(b,d)\ f(b,e)}$

bucket C: $\underbrace{f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,c)}$

bucket D: $\underbrace{f(a,d)\ \ \lambda_{B\to D}(d,e)}$

bucket E: $\underbrace{\lambda_{C\to E}(a,e)\ \lambda_{D\to E}(a,e)}$

bucket A: $\underbrace{f(a)\ \lambda_{E\to A}(a)}$

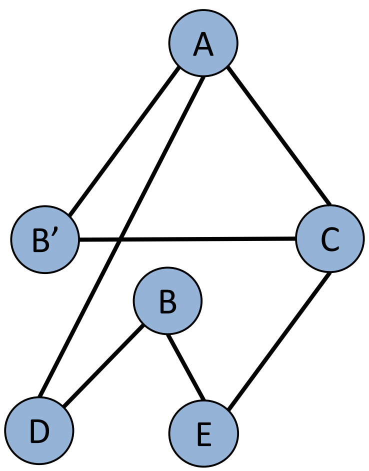
U = upper bound

$$\lambda_{B\to C}(a,c) = \max_b f(a,b)\ f(b,c)$$

$$\lambda_{B\to D}(d,e) = \max_b f(b,d)\ f(b,e)$$

$$\lambda_{C\to E}(a,e) = \max_c \ldots$$

Mini-Bucket Elimination [Dechter & Rish 2003]



mini-buckets

bucket B: $\underbrace{f(a,b')\ f(b',c)}\qquad \underbrace{f(b,d)\ f(b,e)}$

bucket C: $\underbrace{f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,c)}$

bucket D: $\underbrace{f(a,d)\ \ \lambda_{B\to D}(d,e)}$

bucket E: $\underbrace{\lambda_{C\to E}(a,e)\ \lambda_{D\to E}(a,e)}$

bucket A: $\underbrace{f(a)\ \lambda_{E\to A}(a)}$

U = upper bound

$$\lambda_{B\to C}(a,c) = \max_b f(a,b)\ f(b,c)$$

$$\lambda_{B\to D}(d,e) = \max_b f(b,d)\ f(b,e)$$

$$\lambda_{C\to E}(a,e) = \max_c \ldots$$

Can interpret process as “duplicating” B
[Kask et al. 2001, Geffner et al. 2007,
Choi et al. 2007, Johnson et al. 2007]

Mini-Bucket Decoding

- Assign values in reverse order using approximate messages

$$
\begin{aligned}
\mathbf{b}^* &= \arg\max_b \; f(a^*,b)\cdot f(b,c^*) \\
&\qquad\qquad \cdot f(b,d^*)\cdot f(b,e^*) \\
\mathbf{c}^* &= \arg\max_c \; f(c,a^*)\cdot f(c,e^*)\cdot \lambda_{B\to C}(a^*,c) \\
\mathbf{d}^* &= \arg\max_d \; f(a^*,d)\cdot \lambda_{B\to D}(d,e^*) \\
\mathbf{e}^* &= \arg\max_e \; \lambda_{C\to E}(a^*,e)\cdot \lambda_{D\to E}(a^*,e) \\
\mathbf{a}^* &= \arg\max_a \; f(a)\cdot \lambda_{E\to A}(a)
\end{aligned}
$$

Greedy configuration = lower bound

mini-buckets

B: $\overbrace{f(a,b)\;f(b,c)}$ $\overbrace{f(b,d)\;f(b,e)}$

C: $f(c,a)\;f(c,e)\;\lambda_{B\to C}(a,c)$

D: $f(a,d)\;\lambda_{B\to D}(d,e)$

E: $\lambda_{C\to E}(a,e)\;\lambda_{D\to E}(a,e)$

A: $f(a)\;\lambda_{E\to A}(a)$

U = upper bound

Properties of MBE(i)

- **Complexity**: O(r exp(i)) time and O(exp(i)) space
- Yields a lower bound and an upper bound
- **Accuracy**: determined by upper/lower (U/L) bound
- Possible use of mini-bucket approximations
  - As anytime algorithms
  - As heuristics in search
- Other tasks (similar mini-bucket approximations)
  - Belief updating, Marginal MAP, MEU, WCSP, Max-CSP

  [Dechter and Rish, 1997], [Liu and Ihler, 2011], [Liu and Ihler, 2013]

Tightening the bound

- Reparameterization (or, “cost shifting”)
  - Decrease bound without changing overall function

A	B	f_1(A,B)
0	0	2.0
1	0	3.5
0	1	1.0
1	1	3.0

+

B	C	f_2(B,C)
0	0	1.0
0	1	0.0
1	0	1.0
1	1	3.0

$$\max_{a,b} f_1(a,b) + \lambda_{B \to AB}(b) \quad + \quad \max_{b,c} f_2(b,c) + \lambda_{B \to BC}(b)$$

=

$$f_{AB}(a,b) + f_{BC}(b,c)$$

A	B	C	F(A,B,C)
0	0	0	3.0
0	0	1	2.0
0	1	0	2.0
0	1	1	4.0
1	0	0	4.5
1	0	1	3.5
1	1	0	4.0
1	1	1	6.0

$$\lambda_{B \to AB}(b) + \lambda_{B \to BC}(b) = 0$$

(Adjusting functions cancel each other)

=

A	B	f_1(A,B)	λ(B)
0	0	2.0	0
1	0	3.5	
0	1	1.0	+1
1	1	3.0	

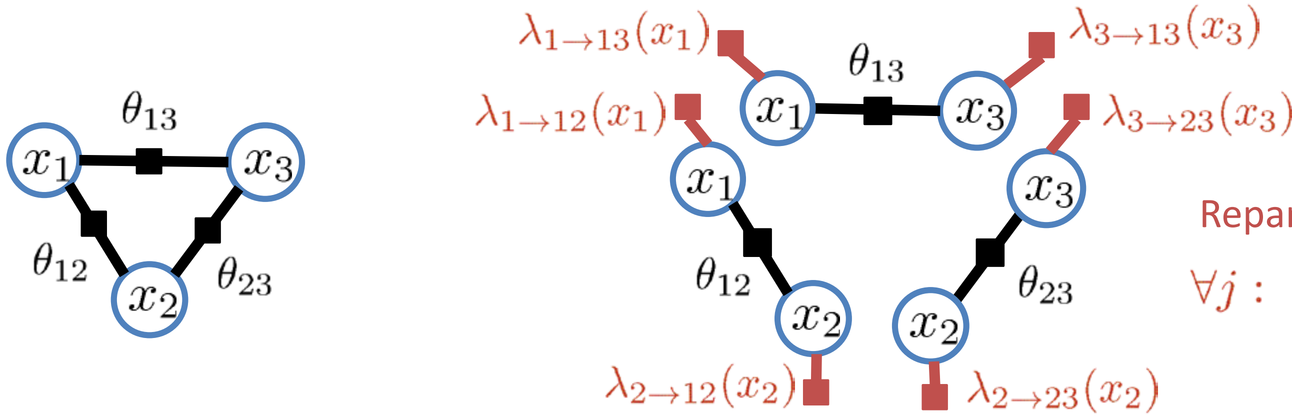
+

B	C	f_2(B,C)	-λ(B)
0	0	1.0	0
0	1	0.0	
1	0	1.0	-1
1	1	3.0	

(Decomposition bound is exact)

Decomposition for MAP

Add factors that "adjust" each local term, but cancel out in total



Reparameterization:

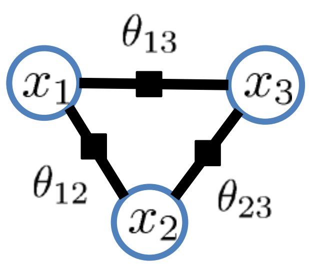
$$\forall j: \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

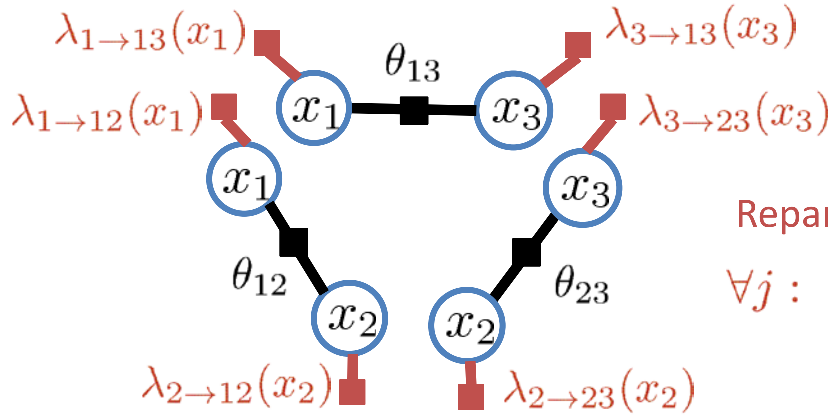
$$\log f(\mathbf{x}^*) = \max_{\mathbf{x}} \sum_{\alpha} \theta_\alpha(\mathbf{x}_\alpha) \quad \le \min_{\{\lambda_{i\to\alpha}\}} \sum_{\alpha} \max_{\mathbf{x}_\alpha} \left[\theta_\alpha(\mathbf{x}_\alpha) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \right]$$

- Bound solution using decomposed optimization
- Solve independently: optimistic bound
- Tighten the bound by reparameterization
  - Enforces lost equality constraints using Lagrange multipliers

Decomposition for MAP

Add factors that "adjust" each local term, but cancel out in total





Reparameterization:

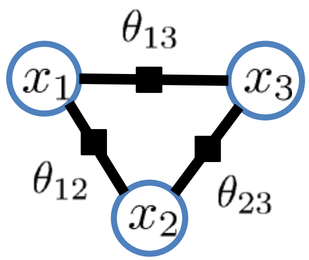
$$\forall j : \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

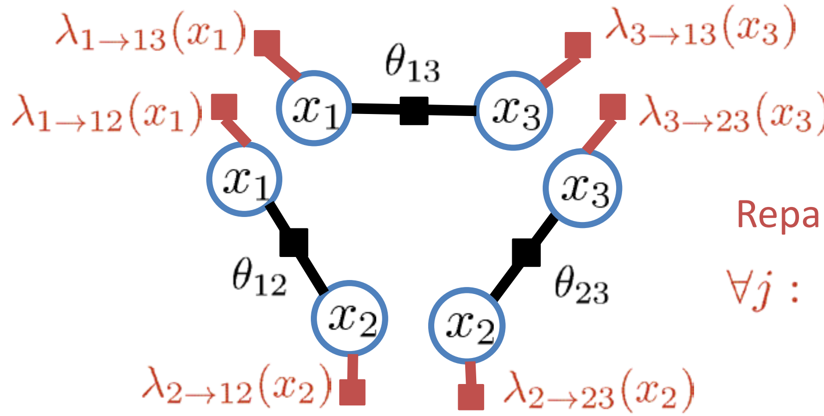
$$\log f(\mathbf{x}^*) = \max_{\mathbf{x}} \sum_{\alpha} \theta_\alpha(\mathbf{x}_\alpha) \quad \le \min_{\{\lambda_{i\to\alpha}\}} \sum_{\alpha} \max_{\mathbf{x}_\alpha} \Big[\theta_\alpha(\mathbf{x}_\alpha) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \Big]$$

- Many names for the same class of bounds
  - Dual decomposition [Komodakis et al. 2007]
  - TRW, MPLP [Wainwright et al. 2005; Globerson & Jaakkola 2007]
  - Soft arc consistency [Cooper & Schiex 2004]
  - Max-sum diffusion [Warner 2007]

Decomposition for MAP

Add factors that "adjust" each local term, but cancel out in total





Reparameterization:

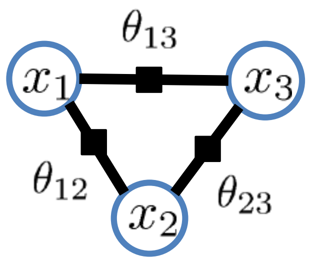
$$\forall j: \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

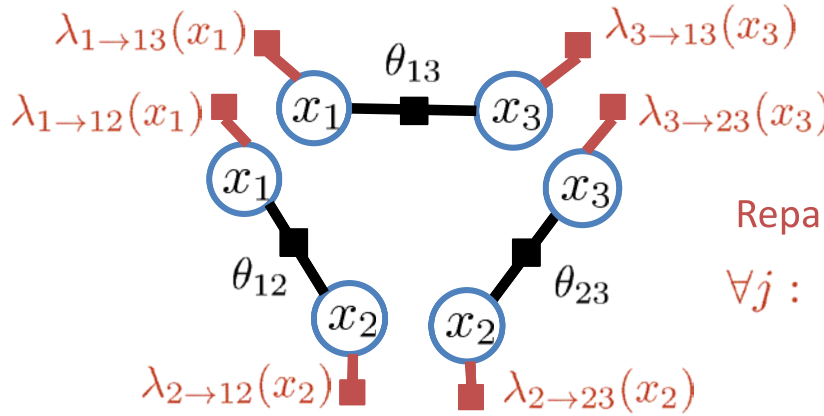
$$\log f(\mathbf{x}^*) = \max_{\mathbf{x}} \sum_{\alpha} \theta_\alpha(\mathbf{x}_\alpha) \quad \leq \min_{\{\lambda_{i\to\alpha}\}} \sum_{\alpha} \max_{\mathbf{x}_\alpha} \Big[\theta_\alpha(\mathbf{x}_\alpha) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \Big]$$

- Many ways to optimize the bound:
  - Sub-gradient descent [Komodakis et al. 2007; Jojic et al. 2010]
  - Coordinate descent [Warner 2007; Globerson & Jaakkola 2007; Sontag 2009; Ihler et al 2012]
  - Proximal optimization [Ravikumar et al. 2010]
  - ADMM [Meshi & Globerson 2011; Martins et al. 2011; Forouzan & Ihler 2013]

Decomposition for MAP

Add factors that "adjust" each local term, but cancel out in total



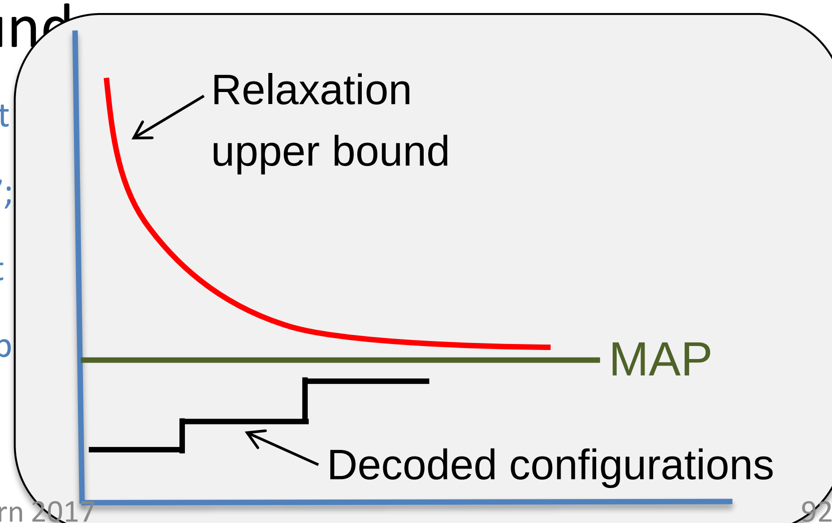


Reparameterization:

$$\forall j : \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

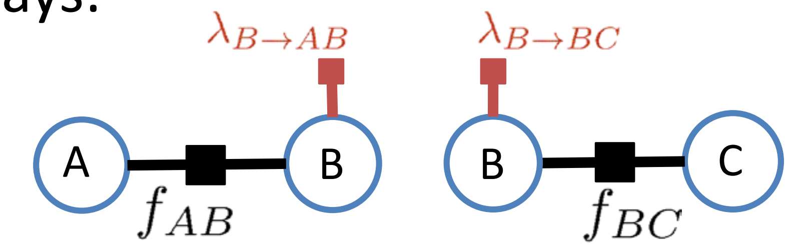
$$\log f(\mathbf{x}^*) = \max_{\mathbf{x}} \sum_{\alpha} \theta_\alpha(\mathbf{x}_\alpha) \leq \min_{\{\lambda_{i\to\alpha}\}} \sum_{\alpha} \max_{\mathbf{x}_\alpha} \left[\theta_\alpha(\mathbf{x}_\alpha) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \right]$$

- Many ways to optimize the bound
  - Sub-gradient descent [Komodakis et
  - Coordinate descent [Warner 2007; ... 12]
  - Proximal optimization [Ravikumar et
  - ADMM [Meshi & Glob ...]



Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent

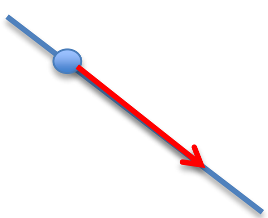


=

A	B	f_1(A,B)	¸(B)
0	0	1.0	0
1	0	0.0	
0	1	0.0	0
1	1	2.5	
0	2	1.0	0
1	2	3.0	

+

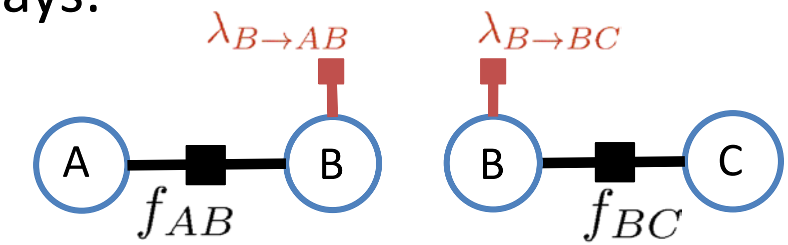
B	C	f_2(B,C)	-¸(B)
0	0	5.0	0
0	1	2.0	
1	0	1.0	0
1	1	1.5	
2	0	0.2	0
2	1	0.0	



$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$

Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent



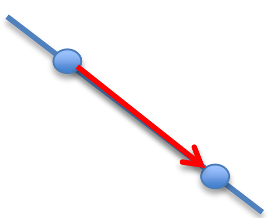
=

A	B	f_1(A,B)	¸(B)
0	0	1.0	+1
1	0	0.0	
0	1	0.0	0
1	1	2.5	
0	2	1.0	-1
1	2	3.0	

\+

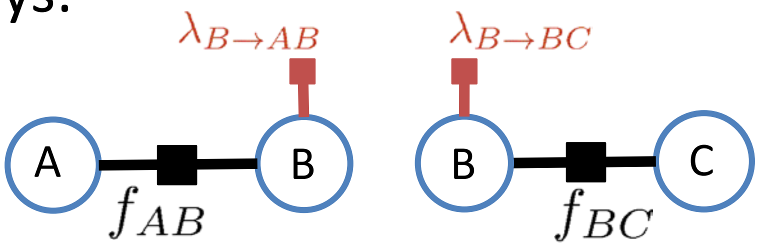
B	C	f_2(B,C)	-¸(B)
0	0	5.0	-1
0	1	2.0	
1	0	1.0	0
1	1	1.5	
2	0	0.2	+1
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$



Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent



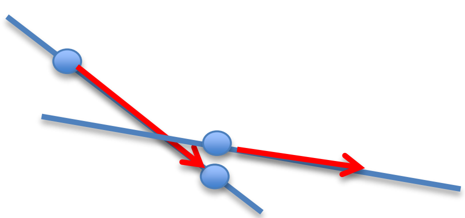
=

A	B	f_1(A,B)	λ(B)
0	0	1.0	+1
1	0	0.0	
0	1	0.0	0
1	1	2.5	
0	2	1.0	-1
1	2	3.0	

+

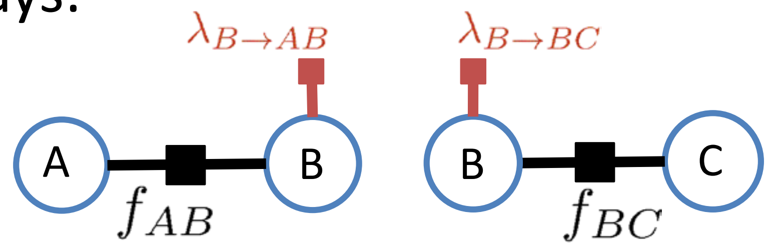
B	C	f_2(B,C)	$-\lambda$(B)
0	0	5.0	-1
0	1	2.0	
1	0	1.0	0
1	1	1.5	
2	0	0.2	+1
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$



Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent

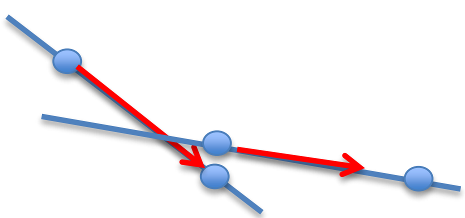


=

A	B	f_1(A,B)	¸(B)
0	0	1.0	+2
1	0	0.0	
0	1	0.0	-1
1	1	2.5	
0	2	1.0	-1
1	2	3.0	

+

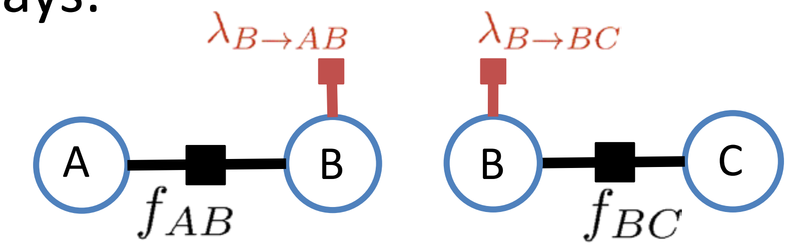
B	C	f_2(B,C)	-¸(B)
0	0	5.0	-2
0	1	2.0	
1	0	1.0	+1
1	1	1.5	
2	0	0.2	+1
2	1	0.0	



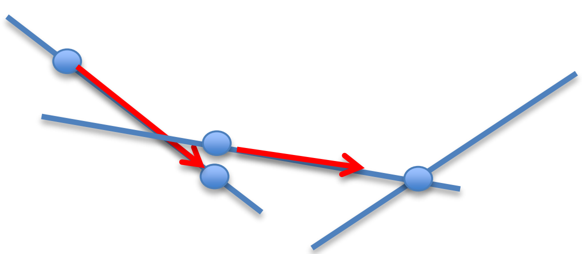
$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$

Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent



Both parts agree on the optima value(s): zero subgradient



=

A	B	f_1(A,B)	λ(B)
0	0	1.0	+2
1	0	0.0	
0	1	0.0	-1
1	1	2.5	
0	2	1.0	-1
1	2	3.0	

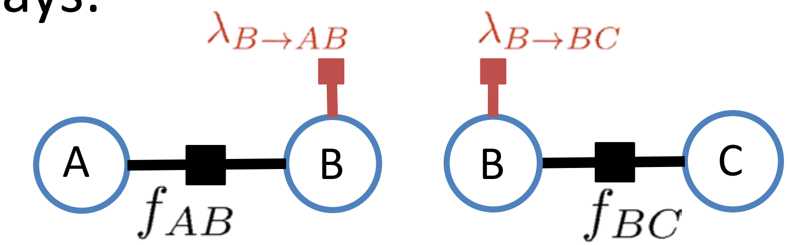
+

B	C	f_2(B,C)	-λ(B)
0	0	5.0	-2
0	1	2.0	
1	0	1.0	+1
1	1	1.5	
2	0	0.2	+1
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$

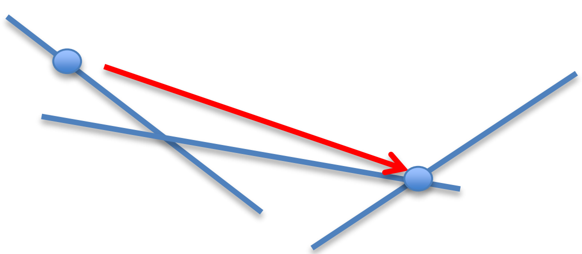
Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent
  - Coordinate descent



Easy to minimize over a single variable, e.g. B:

Find maxima for each B
Match values between f's



=

A	B	f_1(A,B)	¸(B)
0	0	1.0	
1	0	0.0	
0	1	0.0	
1	1	2.5	
0	2	1.0	
1	2	3.0	

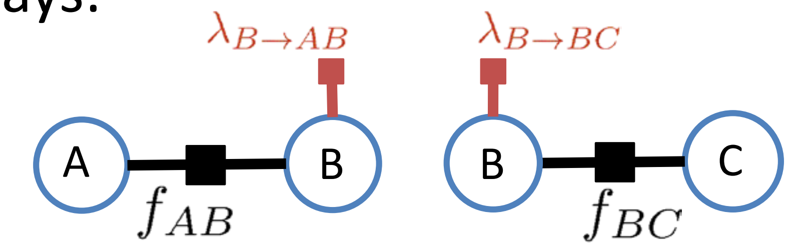
+

B	C	f_2(B,C)	-¸(B)
0	0	5.0	
0	1	2.0	
1	0	1.0	
1	1	1.5	
2	0	0.2	
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$

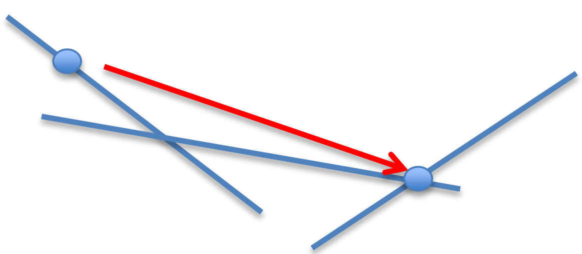
Optimizing the bound

- Can optimize the bound in various ways:
  - (Sub-)gradient descent
  - Coordinate descent



Easy to minimize over a single variable, e.g. B:

Find maxima for each B
Match values between f's



=

A	B	f_1(A,B)	¸(B)
0	0	1.0	- 0.5 +2.5
1	0	0.0	
0	1	0.0	- 1.25 +0.75
1	1	2.5	
0	2	1.0	- 1.5 +0.1
1	2	3.0	

+

B	C	f_2(B,C)	-¸(B)
0	0	5.0	+0.5 - 2.5
0	1	2.0	
1	0	1.0	+1.25 - 0.75
1	1	1.5	
2	0	0.2	+1.5 - 0.1
2	1	0.0	

$$\max_x f_1(a,b) + \lambda_{B\to AB}(b) \quad + \quad \max_x f_2(b,c) + \lambda_{B\to BC}(b)$$

Mini-Bucket as Decomposition

[Ihler et al. 2012]

$$\max_{a,c,b} \log \left[f(a,b) \cdot f(b,c) / \lambda_{B\to C}(a,c) \right] = 0$$

$$\max_{b,d,e} \log \left[f(b,d) \cdot f(b,e) / \lambda_{B\to D}(d,e) \right] = 0$$

$$\max_{a,e,c} \log \left[f(c,a)\, f(c,e)\, \lambda_{B\to C} / \lambda_{C\to E} \right] = 0$$

$$\max_{a,d,e} \log \left[f(a,d)\, \lambda_{B\to D} / \lambda_{D\to E} \right] = 0$$

$$\max_{a,d} \log \left[\lambda_{C\to E}\, \lambda_{D\to E} / \lambda_{E\to A} \right] = 0$$

$$\max_{a} \log \left[f(a)\, \lambda_{E\to A}(a) \right] = \log U$$

mini-buckets

B: $\underbrace{f(a,b)\ f(b,c)}$ $\underbrace{f(b,d)\ f(b,e)}$

C: $\underbrace{f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,c)}$

D: $\underbrace{f(a,d)\ \lambda_{B\to D}(d,e)}$

E: $\underbrace{\lambda_{C\to E}(a,e)\ \lambda_{D\to E}(a,e)}$

A: $\underbrace{f(a)\ \lambda_{E\to A}(a)}$

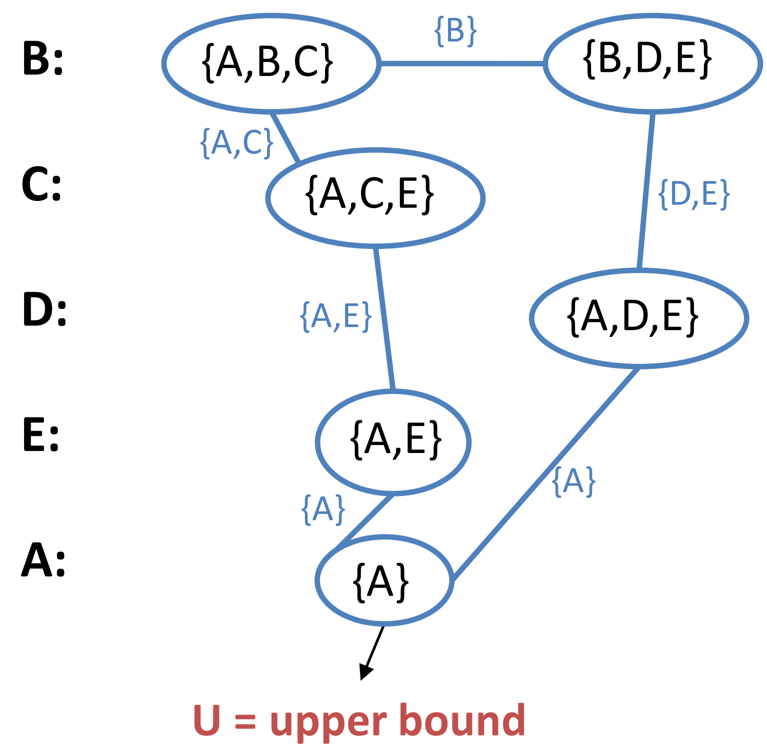
U = upper bound

Mini-Bucket as Decomposition

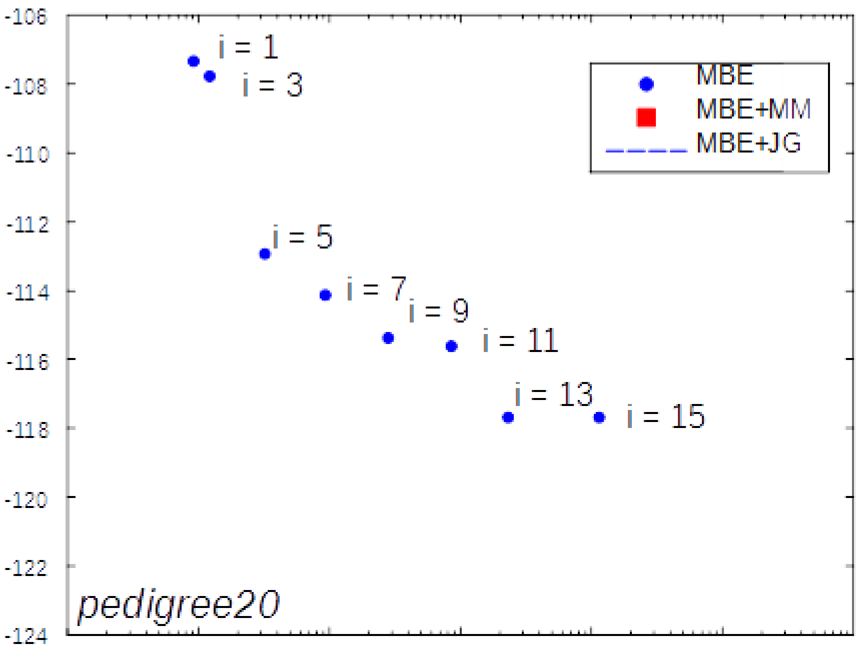
[Ihler et al. 2012]

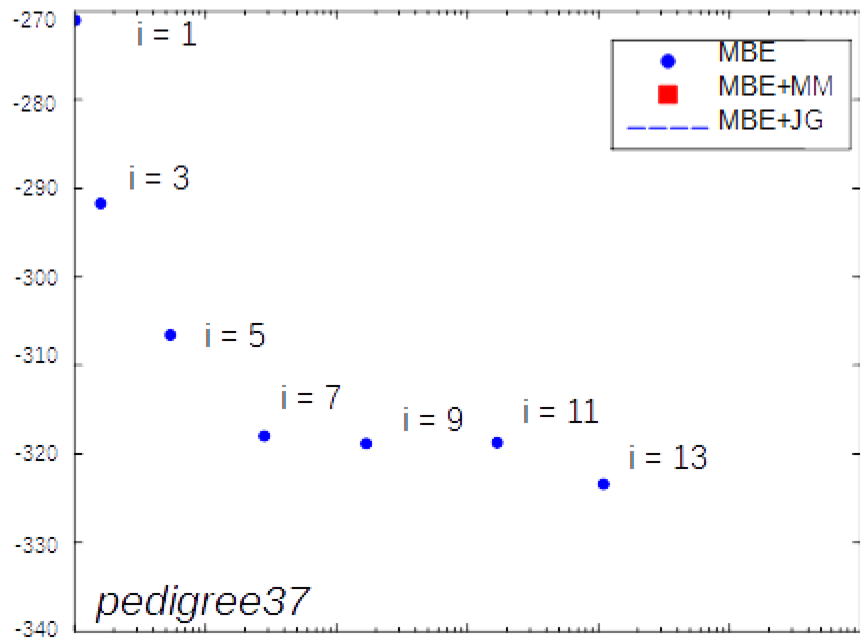
- Downward pass as cost shifting
- Can also do cost shifting within mini-buckets:
  "Join graph" message passing
- "Moment-matching" version:
  One message exchange within each bucket, during downward sweep
- Optimal bound defined by cliques ("regions") and cost-shifting f'n scopes ("coordinates")

Join graph:



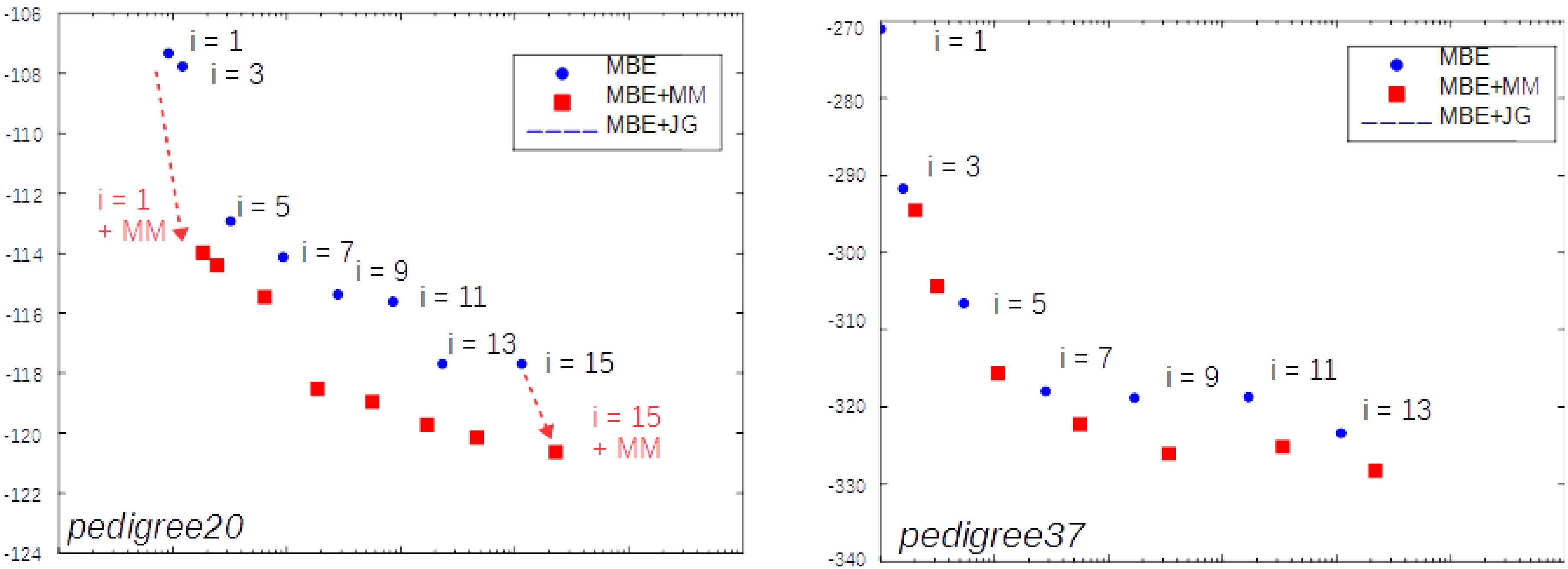
Anytime Approximation





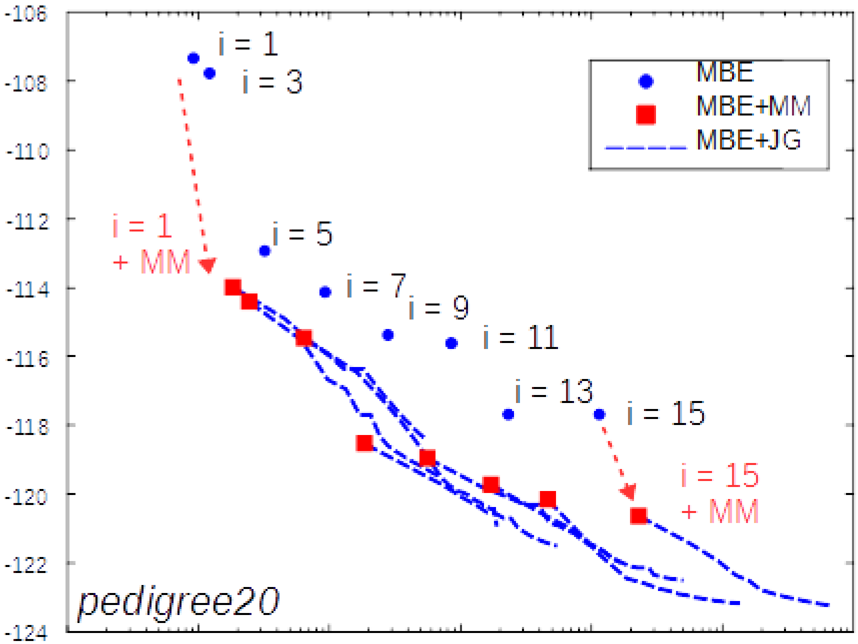
- Can tighten the bound in various ways
  - Cost-shifting (improve consistency between cliques)
  - Increase i-bound (higher order consistency)
- Simple moment-matching step improves bound significantly

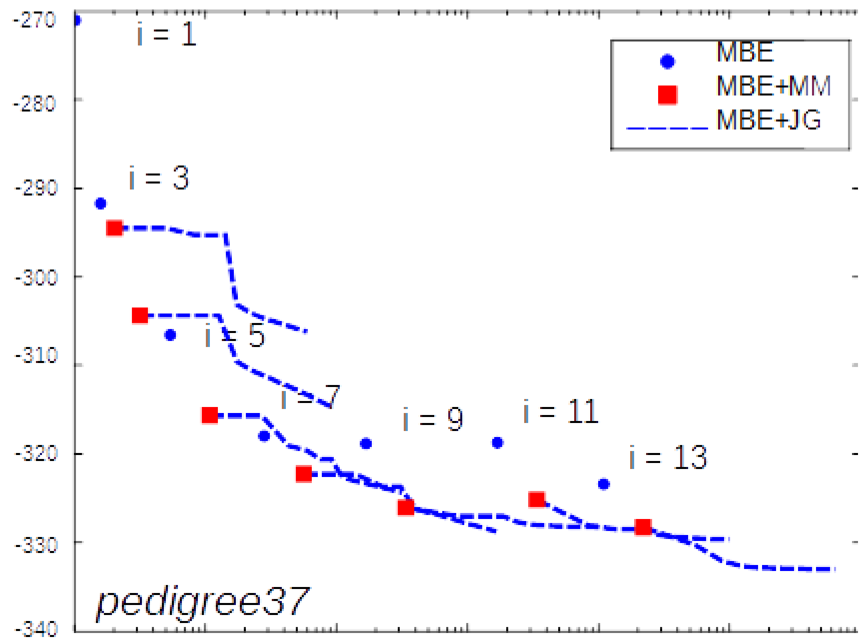
Anytime Approximation



- Can tighten the bound in various ways
  - Cost-shifting (improve consistency between cliques)
  - Increase i-bound (higher order consistency)
- Simple moment-matching step improves bound significantly

Anytime Approximation





- Can tighten the bound in various ways
  - Cost-shifting (improve consistency between cliques)
  - Increase i-bound (higher order consistency)
- Simple moment-matching step improves bound significantly

Decomposition for Sum

$$F(x) = f_1(x) \cdot f_2(x)$$

- Generalize technique to sum via Holder's inequality:

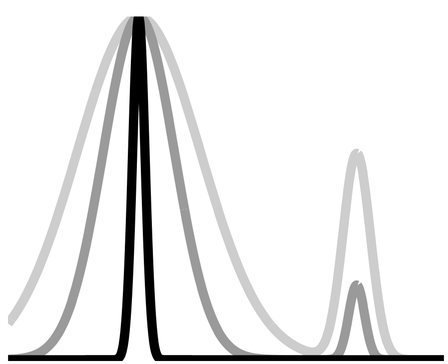
$$\sum_x f_1(x) \cdot f_2(x) \quad \le \quad \Big[\sum_x f_1(x)^{\frac{1}{w_1}}\Big]^{w_1} \cdot \Big[\sum_x f_2(x)^{\frac{1}{w_2}}\Big]^{w_2}$$

$$w_1 + w_2 = 1$$

- Define the weighted (or powered) sum:

$$\sum_{x_1}^{w_1} f(x_1) = \Big[\sum_{x_1} f(x_1)^{\frac{1}{w_1}}\Big]^{w_1}$$

  - "Temperature" interpolates between sum & max:



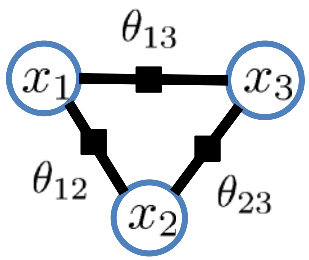
$$\lim_{w \to 0^+} \sum_x^{w} f(x) = \max_x f(x)$$

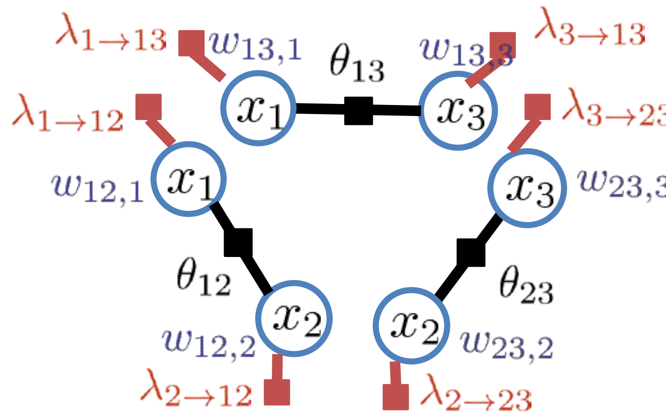
  - Different weights do not commute:

$$\sum_{x_1}^{w_1} \sum_{x_2}^{w_2} f(x_1, x_2) \neq \sum_{x_2}^{w_2} \sum_{x_1}^{w_1} f(x_1, x_2)$$

Decomposition for Sum

[Peng, Liu, Ihler 2015]





Reparameterization:

$$\forall j : \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

$$\log Z = \log \sum_{\mathbf{x}} \exp \Big[\sum_{\alpha} \theta_\alpha(\mathbf{x}_\alpha) \Big] \le \min_{\substack{\{\lambda_{i\to\alpha}\} \\ \{w_{\alpha,i}\}}} \sum_{\alpha} \log \sum_{\mathbf{x}_\alpha}^{\mathbf{w}_\alpha} \exp \Big[\theta_\alpha(\mathbf{x}_\alpha + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \Big]$$

- Fixed elimination order
- Assign weight per clique & variable
- Again, tighten bound by reparameterization
  - Can also optimize over weights

Weights:

$$\forall j : \sum_{\alpha \ni j} \mathbf{w}_{\alpha,j} = 0$$

Ex: w_{12} = [0.5 0.3 -]
w_{13} = [0.5 - 0.6]
w_{23} = [- 0.7 0.4]

Weighted Mini-bucket

[Liu & Ihler 2011]

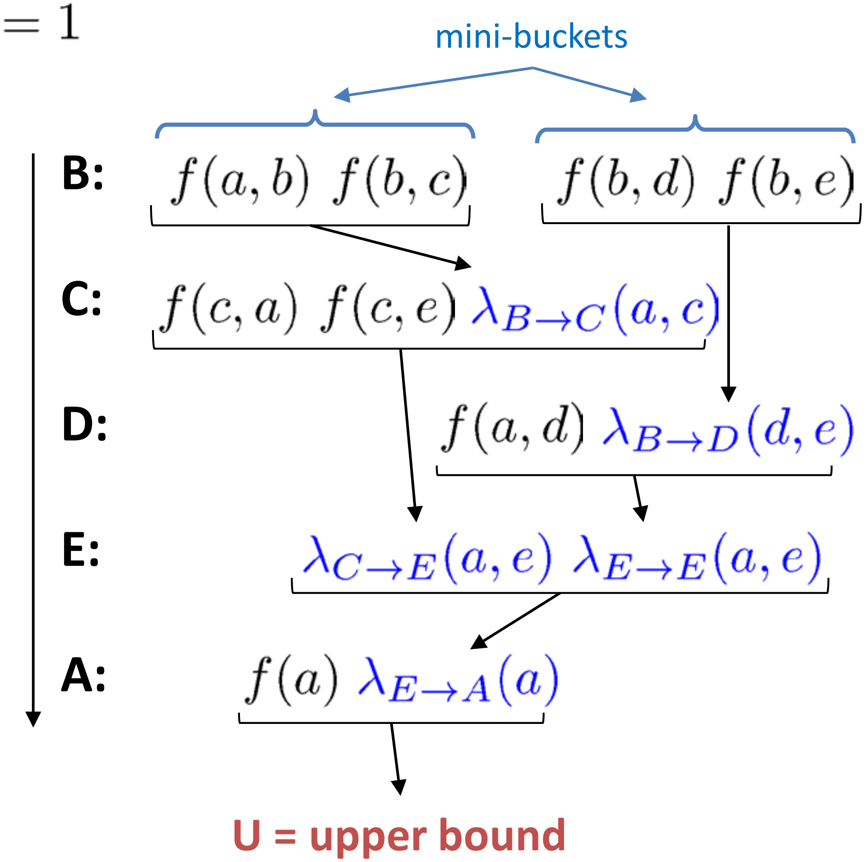
$$\lambda_{B\to C} = \sum_b^{w_{B1}} f(a,b) \cdot f(b,c)$$

$$w_{B1} + w_{B2} = 1$$

$$\lambda_{B\to D} = \sum_b^{w_{B2}} f(b,d) \cdot f(b,e)$$

$$\lambda_{C\to E} = \sum_c f(c,a) \cdot f(c,e) \cdot \lambda_{B\to C}$$

⋮

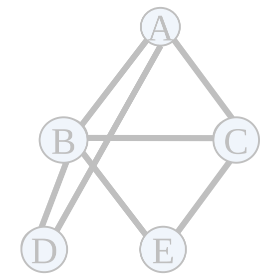


Compute downward messages
using weighted sum

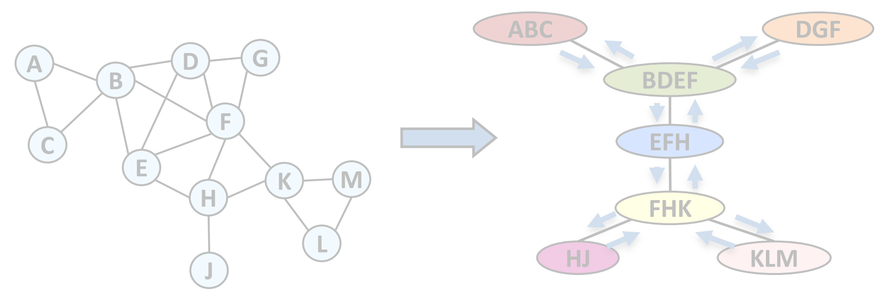
Upper bound if all weights positive
(corresponding lower bound if only one positive, rest negative)

RoadMap: Introduction and Inference

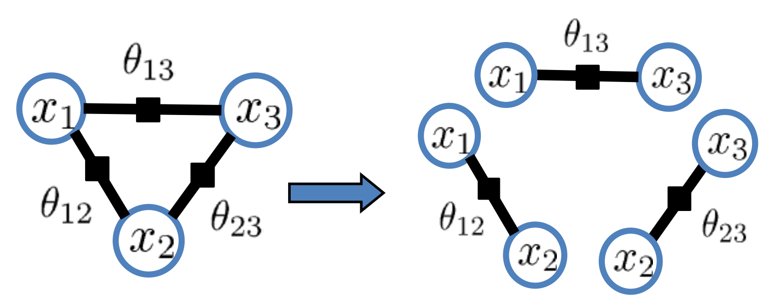
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - Algorithms overview



- Inference algorithms, exact
  - Bucket elimination for trees
  - Bucket elimination
  - Jointree clustering
  - Elimination orders



- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - **Belief propagation**

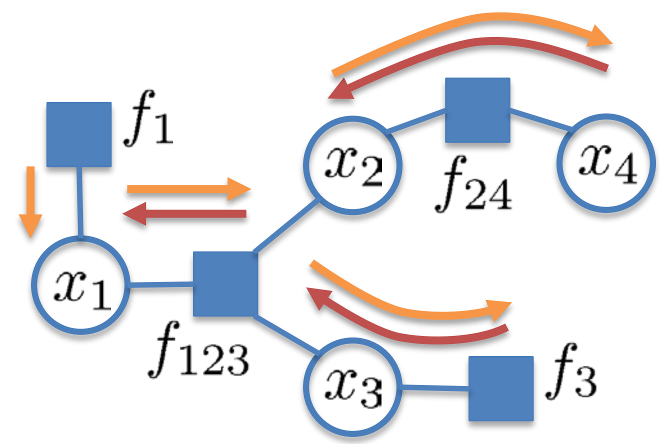


- Summary and Class 2

Variable elimination in trees

Computing all marginal probabilities:

- Two-pass algorithm
  - Pass messages upward to a root
  - Pass messages back downward
  - Messages summarize marginalization of sub-model rooted at that node



- Use messages to compute marginals
- Can also update all messages in parallel, until converged

Calculating messages:

$$m_{i\to\alpha}(x_i) \propto \prod_{\beta \neq \alpha} m_{\beta\to i}(x_i)$$

$$m_{\alpha\to i}(x_i) \propto \sum_{x_\alpha \setminus x_i} f_\alpha(x_\alpha) \prod_{j \neq i} m_{j\to\alpha}(x_j)$$

Calculating marginals:

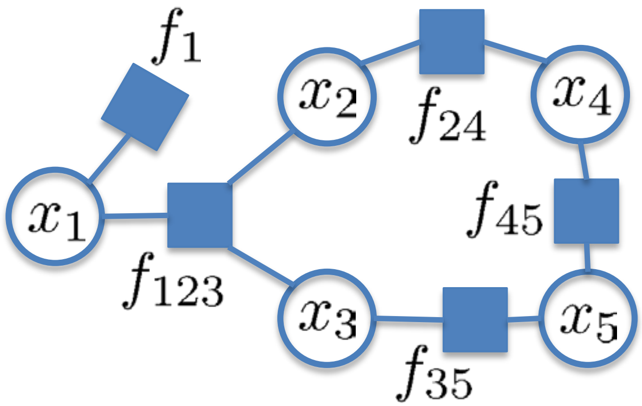
$$p(x_i) \propto \prod_{\alpha \ni i} m_{\alpha\to i}(x_i)$$

$$p(x_\alpha) \propto f_\alpha(x_\alpha) \prod_{i \in \alpha} m_{i\to\alpha}(x_i)$$

Loopy belief propagation

[Pearl 1986]

- Apply the same local updates in arbitrary structure
  - More precisely, often called the "sum-product" algorithm
- Resulting algorithm computes "beliefs" b
  - May not converge
  - Initialization & schedule can matter
  - Is approximate: $b(x_i) \approx p(x_i)$
  - But, is often pretty good in practice
    (quality depends on how "tree-like" the model is)



Calculating messages:

$$m_{i\to\alpha}(x_i) \propto \prod_{\beta\neq\alpha} m_{\beta\to i}(x_i)$$

$$m_{\alpha\to i}(x_i) \propto \sum_{x_\alpha \setminus x_i} f_\alpha(x_\alpha) \prod_{j\neq i} m_{j\to\alpha}(x_j)$$

Calculating marginals:

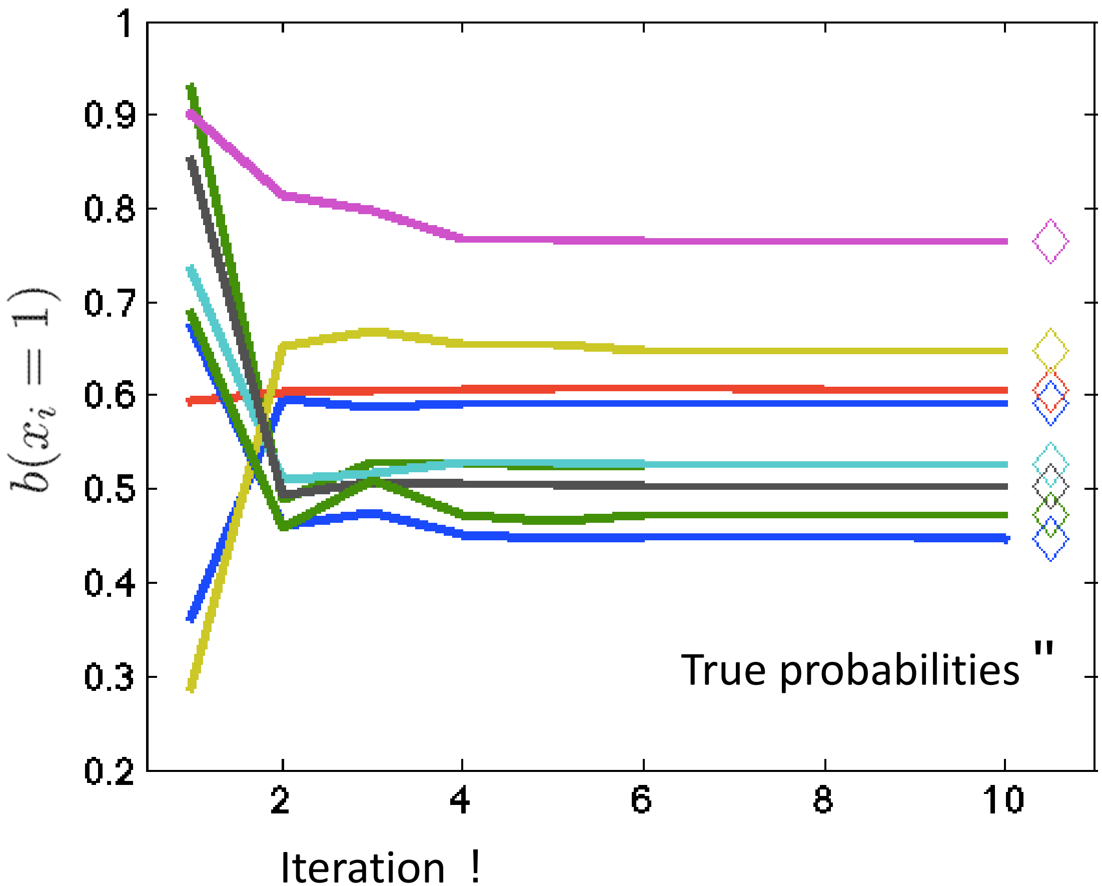
$$b(x_i) \propto \prod_{\alpha\ni i} m_{\alpha\to i}(x_i)$$

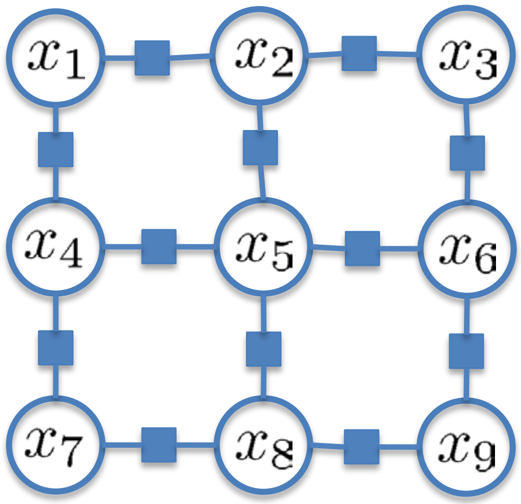
$$b(x_\alpha) \propto f_\alpha(x_\alpha) \prod_{i\in\alpha} m_{i\to\alpha}(x_i)$$

Example: Ising model

- Log-factors

$$\theta_i, \theta_{ij} \sim U[-0.5, 0.5]$$

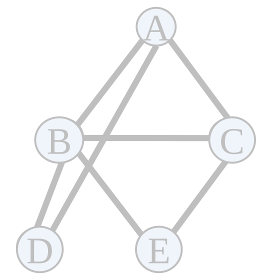




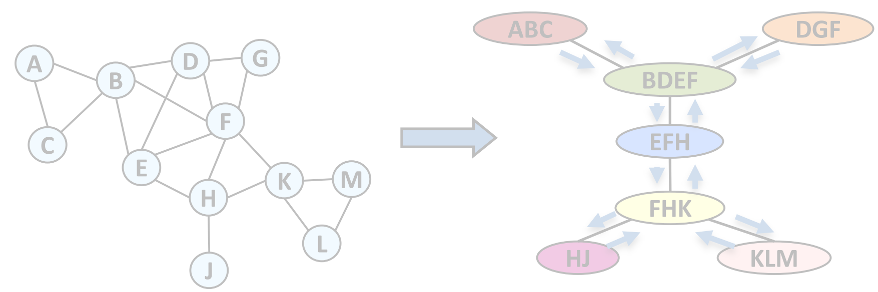
We'll see these ideas again in Class 3…

RoadMap: Introduction and Inference

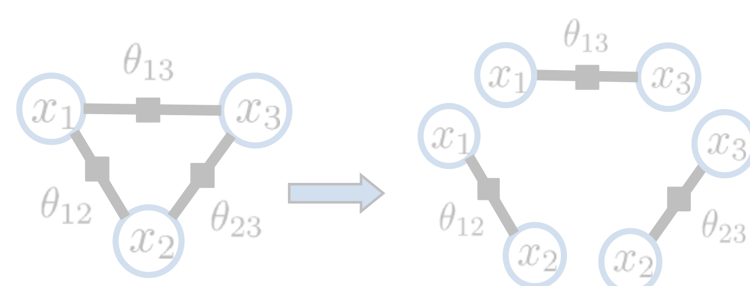
- Basics of graphical models
  - Queries
  - Examples, applications, and tasks
  - Algorithms overview



- Inference algorithms, exact
  - Bucket elimination for trees
  - Bucket elimination
  - Jointree clustering
  - Elimination orders



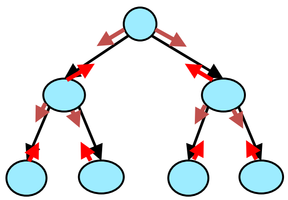
- Approximate elimination
  - Decomposition bounds
  - Mini-bucket & weighted mini-bucket
  - Belief propagation

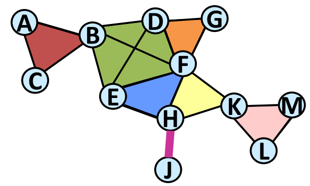


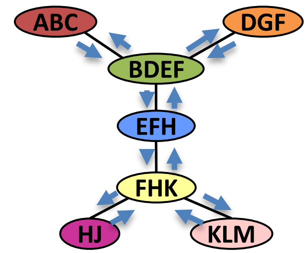
- **Summary and Class 2**

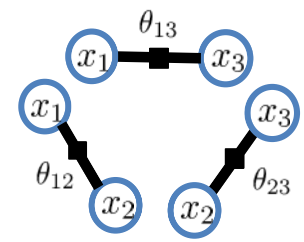
Preview of Class 2

- Class 1: Introduction and Inference

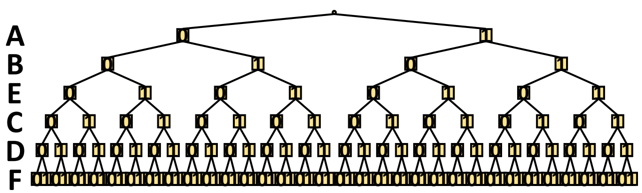


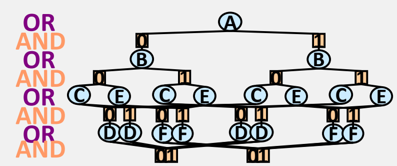






- Class 2: Search





Context minimal AND/OR search graph

- Class 3: Variational Methods and Monte-Carlo Sampling