BREN:ICS INFORMATION AND COMPUTER SCIENCES   UNIVERSITY of CALIFORNIA IRVINE

Inference and Search for

Probabilistic and deterministic Graphical Models

Rina Dechter

Bren school of ICS, University of California, Irvine

Joint work with Radu Marinescu, Robert Mateescu, Kalev Kask, Irina Rish





HUJI 2012

Outline

- Graphical models: reasoning principles
- Inference
- Search; via AND/OR Search
- Lower Bounding schemes for inference
- Lower-bounding heuristics for AND/OR search
- Experiments

Sample Applications for Graphical Models

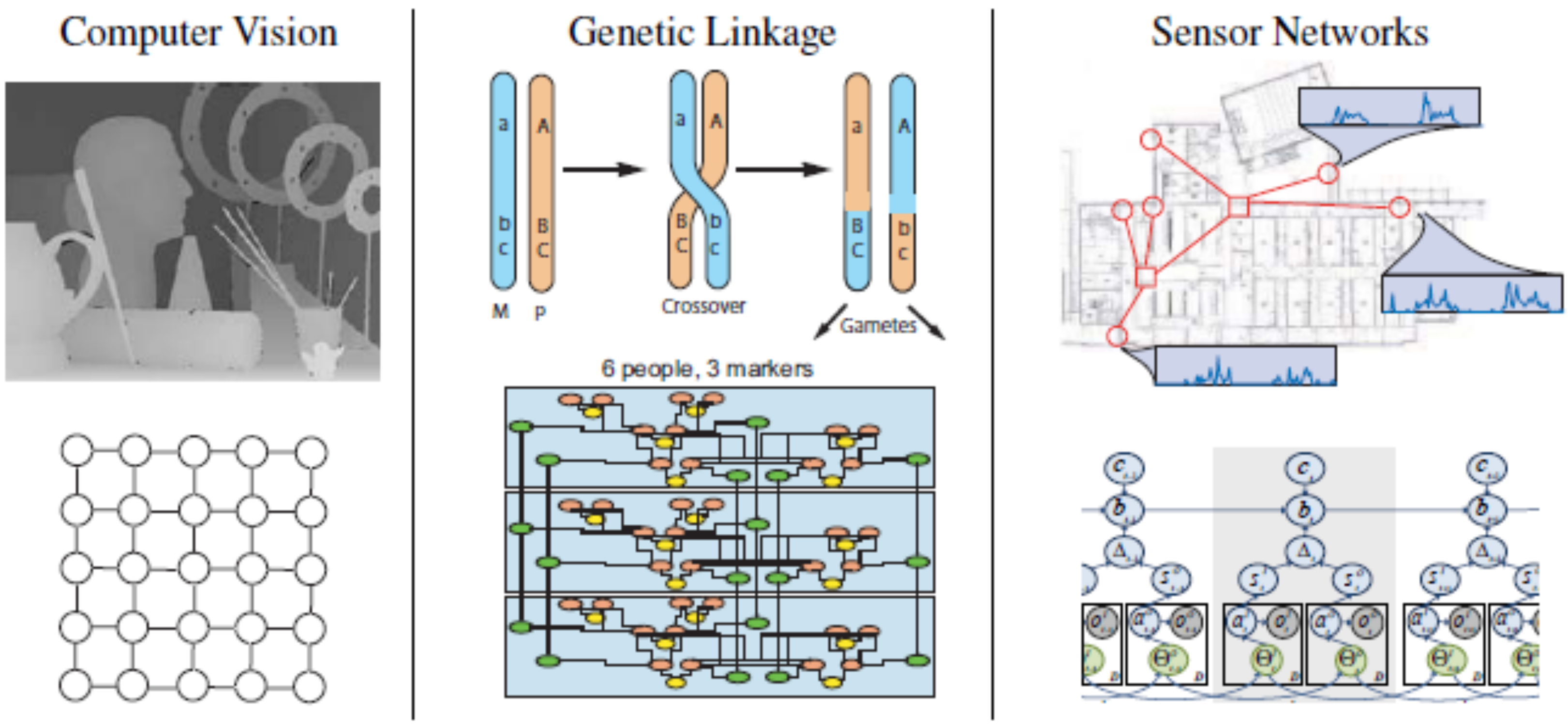


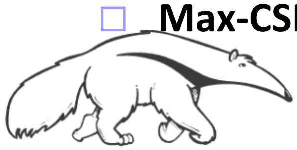
Figure 1: Application areas and graphical models used to represent their respective systems: (a) Finding correspondences between images, including depth estimation from stereo; (b) Genetic linkage analysis and pedigree data; (c) Understanding patterns of behavior in sensor measurements using spatio-temporal models.

Outline

- Graphical models: reasoning principles
- Inference
- Advancing Searchvia AND/OR Search
- Lower Bounding schemes for inference
- Lower-bounding heuristic for AND/OR search
- Experiments

Graphical Models

- A graphical model $(\mathbf{X},\mathbf{D},\mathbf{F})$:
  - $\mathbf{X} = \{X_1,\dots X_n\}$ variables
  - $\mathbf{D} = \{D_1, \dots D_n\}$ domains
  - $\mathbf{F} = \{f_1,\dots,f_r\}$ functions
    (constraints, CPTS, CNFs …)
- Operators:
  - combination
  - elimination (projection)
- Tasks:
  - **Belief updating**: $\Sigma_{X-y} \prod_j P_i$
  - **MPE**: $\max_X \prod_j P_j$
  - **CSP**: $\prod_X \times_j C_j$
  - **Max-CSP**: $\min_X \Sigma_j F_j$



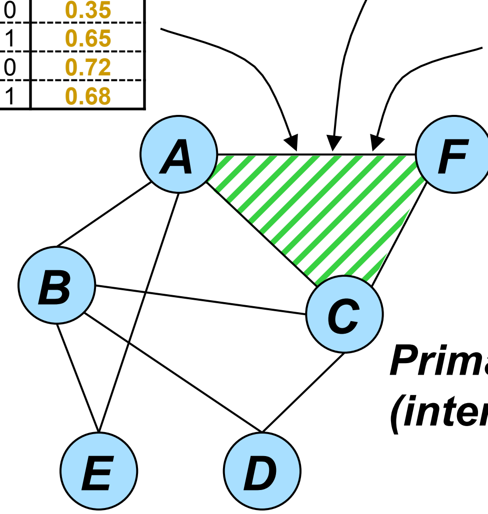
Conditional Probability Table (CPT)

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

$f_i \;:=\; (F = A + C)$



Primal graph (interaction graph)

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

Graphical Models

- A graphical model (**X,D,F**):
  - **X** = {X_1,…X_n} variables
  - **D** = {D_1, … D_n} domains
  - **F** = {f_1,…,f_r} functions
    (constraints, CPTS, CNFs …)

- Tasks:
  - **Belief updating**: $\Sigma_{X-y} \prod_j P_i$
  - **MPE**: $\max_X \prod_j P_j$
  - **CSP**: $\prod_X \times_j C_j$
  - **Max-CSP**: $\min_X \Sigma_j F_j$

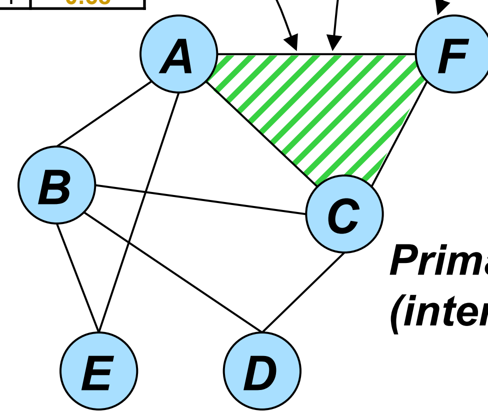
Conditional Probability Table (CPT)

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

$f_i \; := \; (F = A + C)$



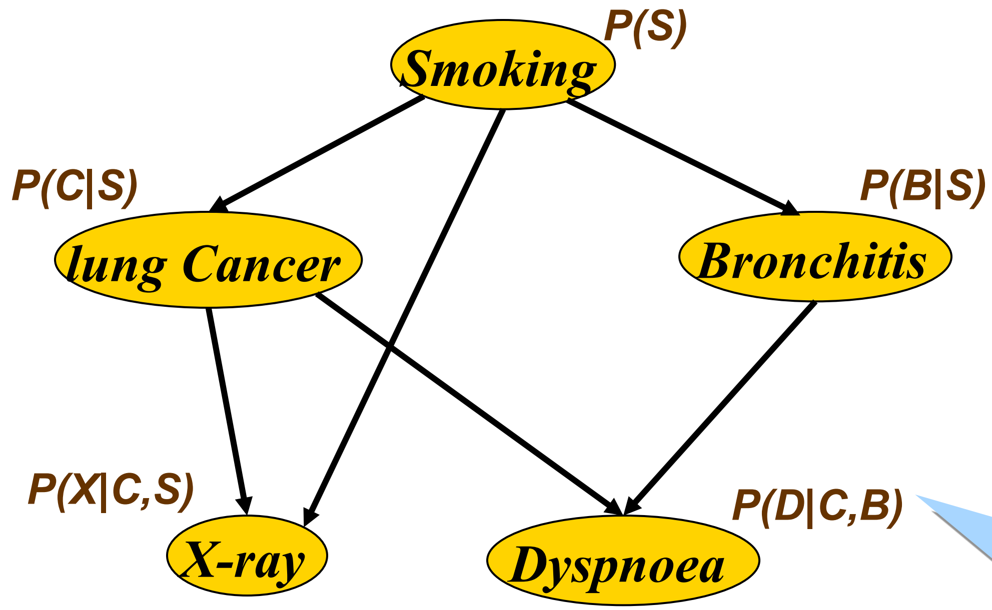
Primal graph (interaction graph)

The MRF

$\boldsymbol{x} = (x_1, \ldots, x_n)$ to all the variables which maximizes the sum of the factors:

$$\mathrm{MAP}(\boldsymbol{\theta}) = \max_{\boldsymbol{x}} \sum_{i \in V} \theta_i(x_i) + \sum_{f \in F} \theta_f(\boldsymbol{x}_f). \qquad (1.1)$$

Bayesian Networks (Pearl 1988)



BN = (G, Θ)

CPD:

C	B	P(D|C,B)	
0	0	0.1	0.9
0	1	0.7	0.3
1	0	0.8	0.2
1	1	0.9	0.1

P(S, C, B, X, D) = *P(S) P(C|S) P(B|S) P(X|C,S) P(D|C,B)*

Combination: Product
Marginalization: sum/ma

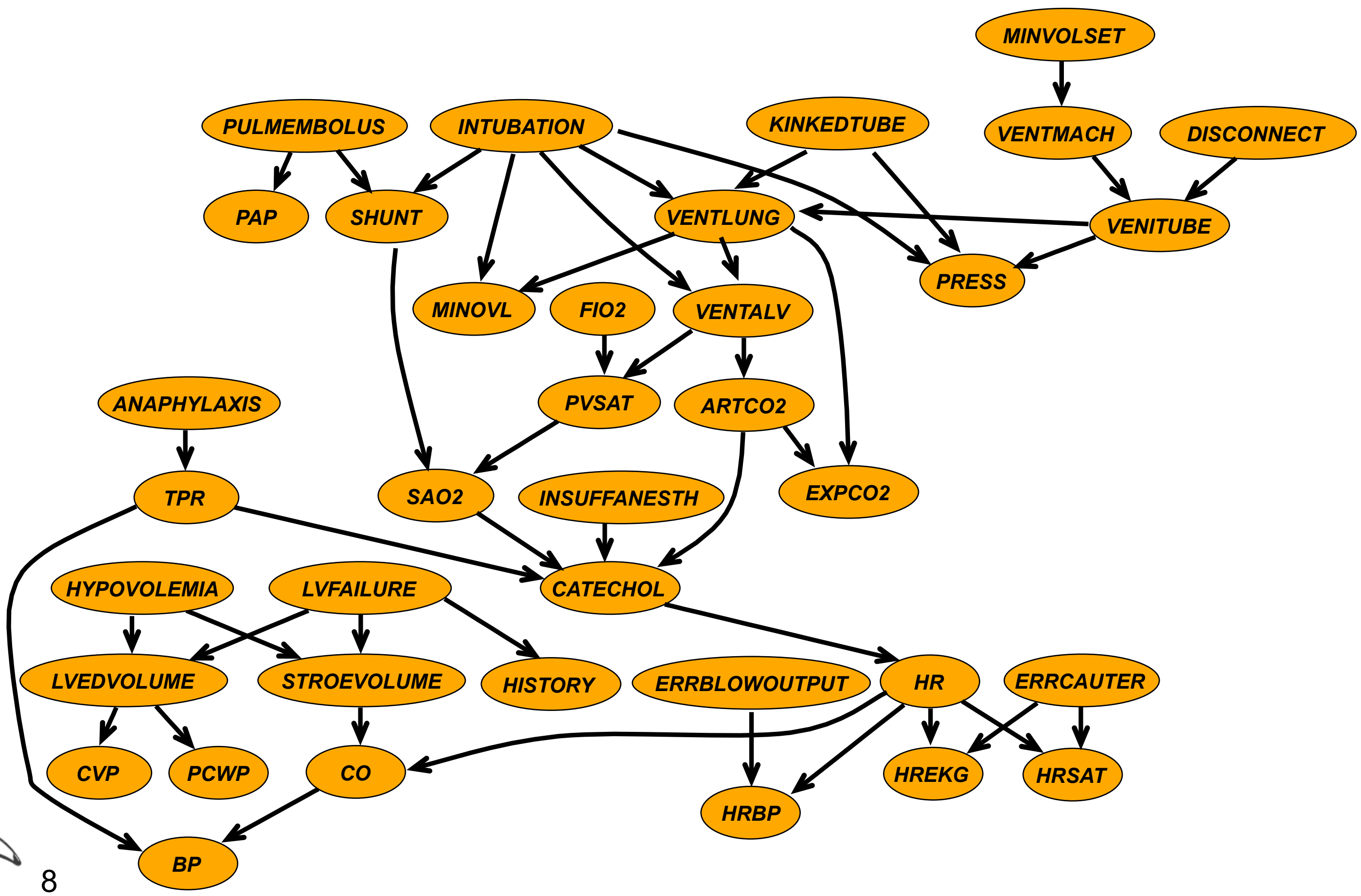
$$P(x_1 \ldots x_n) = \prod_i p(x_i \mid pa(x_i))$$

$$P(e) = \sum_{X-E} \prod_i p(x_i \mid pa(x_i))$$

$$mpe = \max_x P(x)$$

Monitoring Intensive-Care Patients

The “alarm” network - 37 variables, 509 parameters (instead of 2^{37})



Constraint Networks

Map coloring

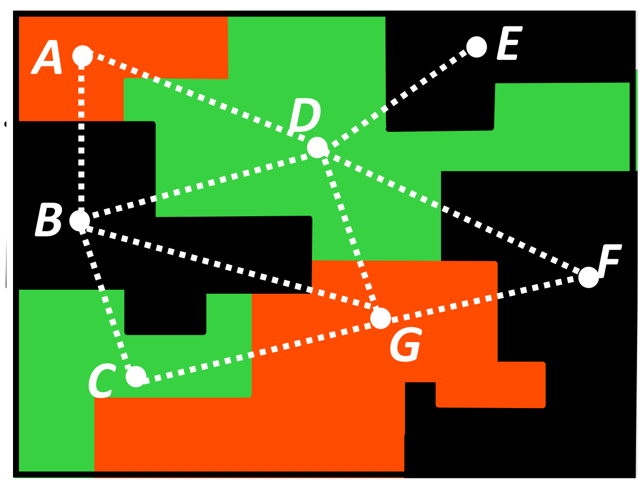
Combination = join
Marginalization = projection

Variables: *countries (A B C etc.)*

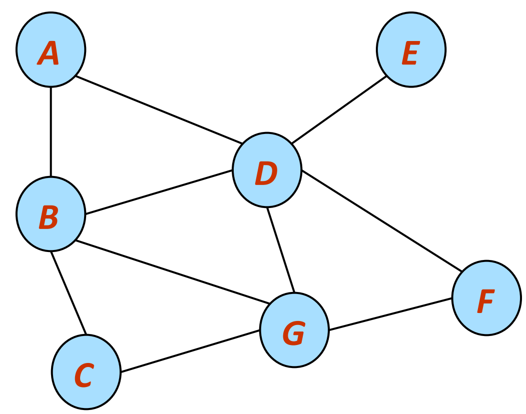
Values: *colors (red green blue)*

Constraints: $A \neq B$, $A \neq D$, $D \neq E$, …

A	*B*
red	*green*
red	*yellow*
green	*red*
green	*yellow*
yellow	*green*
yellow	*red*



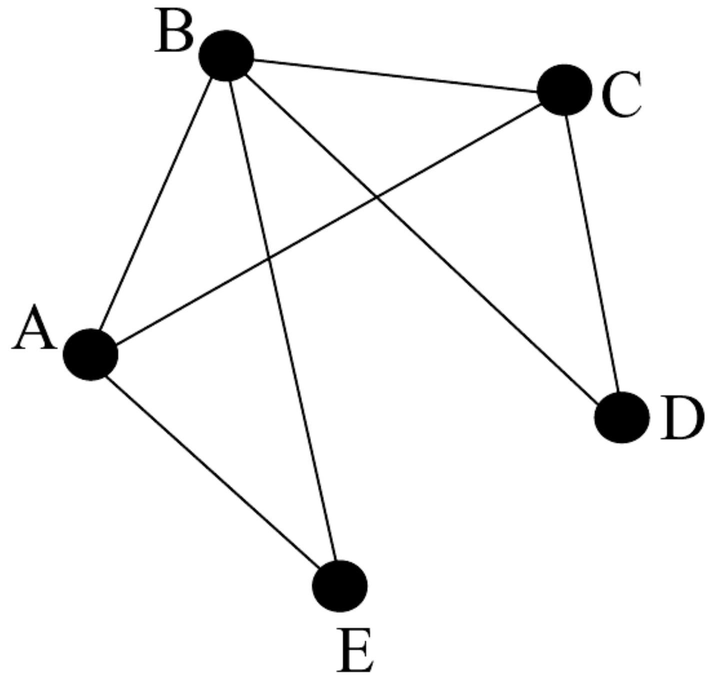
Constraint graph



Queries: Find one solution, all solutions, counting

Propositional Satisfiability

φ = {(¬C), (A v B v C), (¬A v B v E), (¬B v C v D)}.



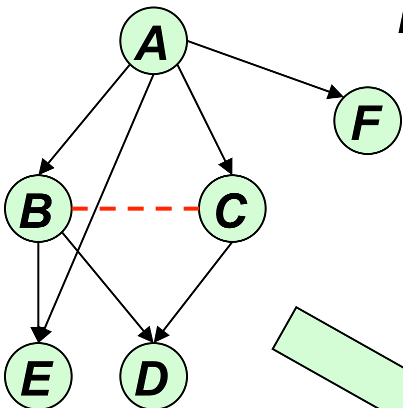
Combination: "AND"
Marginalization: ?
Resolution does combine followedBy project

Mixed Networks

(Mateescu and Dechter, 2004)

Examples: NLP, Linkgage, Software verification, probabilistic languages

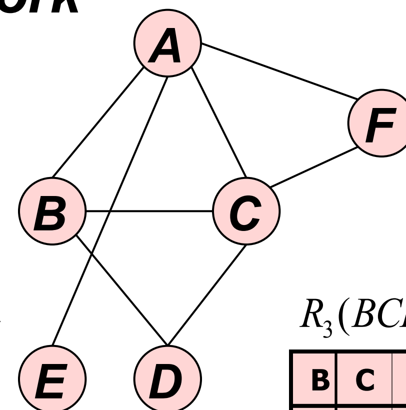
Belief Network



$P(D \mid B, C)$

B	C	D=0	D=1
0	0	.2	.8
0	1	.1	.9
1	0	.3	.7
1	1	.5	.5

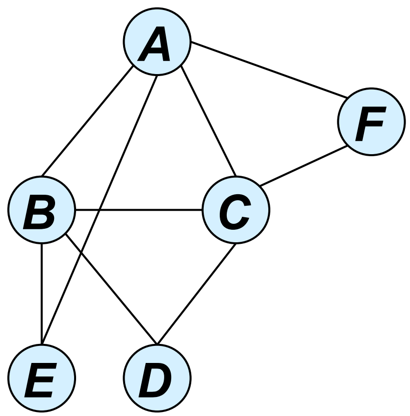
Constraint Network



$R_3(BCD)$

B	C	D
0	0	1
0	1	0
1	1	0

Moral mixed graph



Complex cnf queries: P((A or B) and (~CVD))

$$P_M(\bar{x}) = \begin{cases} P_B(\bar{x} \mid \bar{x} \in \rho) = \dfrac{P_B(\bar{x})}{P_B(\bar{x} \in \rho)}, \text{if } \bar{x} \in \rho \\ 0, \text{otherwise} \end{cases}$$

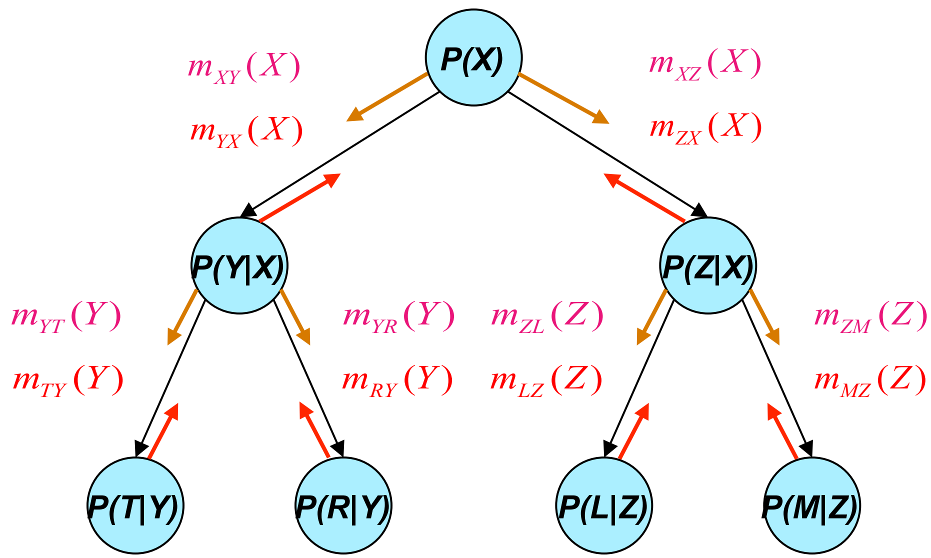
Solution methods

- Solving tree is easy
- Inference: move to trees by clustering
  - (dynamic programming, variable elimination, junction trees)
  - Exploit structure well.
- Search: move to trees by conditioning.
  - Can also exploit structure well.

Tree-solving is Easy

Belief updating (sum-prod)

CSP – consistency (projection-join)



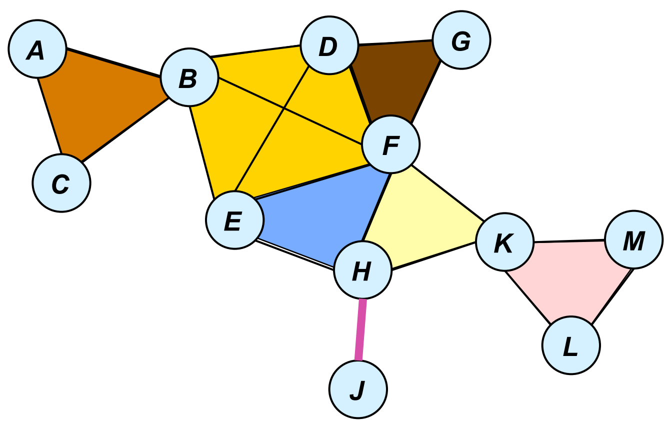
Dynamic Programming, Inference

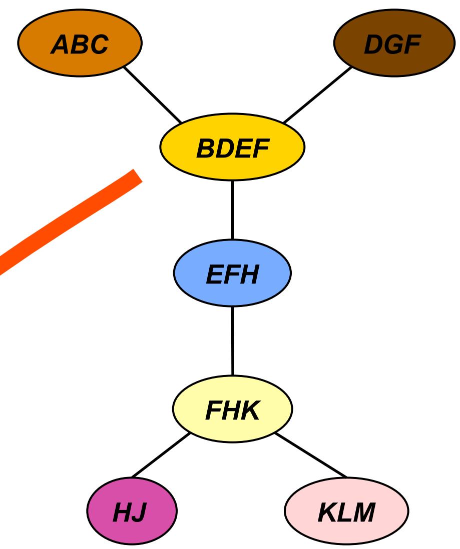
MPE (max-prod)

#CSP (sum-prod)

Trees are processed in linear time and memory
Message-passing

Clustering and Treewidth

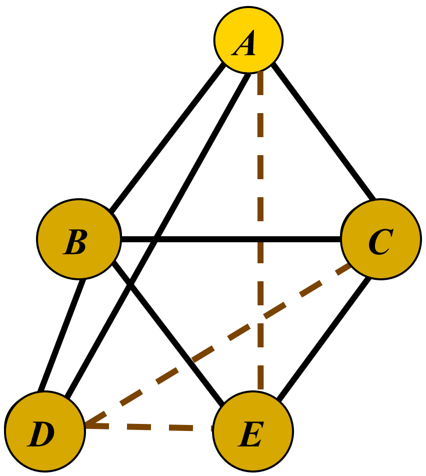




Inference algorithm:
***Time:** exp(tree-width+1)*
***Space:** exp(separator-width)*

treewidth = 4 - 1 = 3
treewidth = (maximum cluster size) – 1
Separator-width=2

Belief updating: P(X|evidence)=?



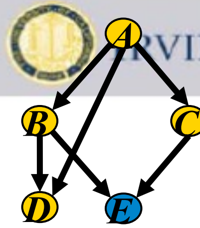
"Moral" graph

$$P(a|e=0) \propto P(a,e=0)=$$

$$\sum_{e=0,d,c,b} P(a)\underbrace{P(b|a)}P(c|a)\underbrace{P(d|b,a)P(e|b,c)}$$

$$P(a)\sum_{e=0}\sum_{d}\sum_{c}P(c|a \underbrace{\sum_{b}P(b|a)P(d|b,a)P(e|b,c)}_{h^{B}(a,d,c,e)}$$

Variable Elimination



Bucket elimination Algorithm *BE-bel* (Dechter 1996)

$$P(A \mid E=0) = \alpha \sum_{E=0,D,C,B} P(A)\cdot P(B \mid A)\cdot P(C \mid A)\cdot P(D \mid A,B)\cdot P(E \mid B,C)$$

$\sum_b \prod$ ← ***Elimination operator***

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

bucket C: $P(c|a)$ $\lambda^B \underbrace{(a,d,c,e)}$

bucket D: $\lambda^C(a,d,e)$

bucket E: $e=0$ $\lambda^D(a,e)$

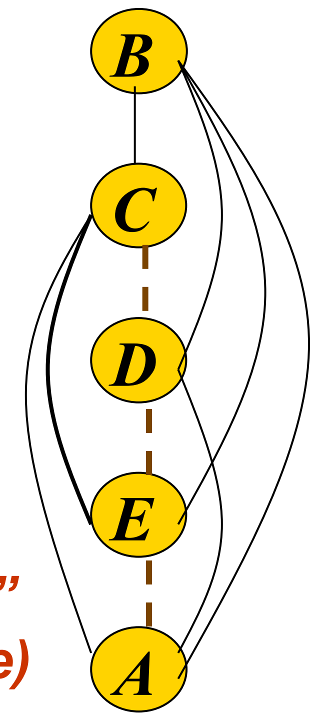
bucket A: $P(a)$ $\lambda^E(a)$

$P(e=0)$

$P(a|e=0)$

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

$$P(a|e=0)=\frac{P(a,e=0)}{P(e=0)}$$



Inference for Optimization: Bucket Elimination

Algorithm BE-mpe (Dechter 1996, Bertele and Briochi, 1977)

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

$$\underbrace{\qquad\qquad\qquad\qquad}^{\max_X \prod}$$

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

bucket C: $P(c|a)$ $h^B(a, \underbrace{d, c, e}_{})$

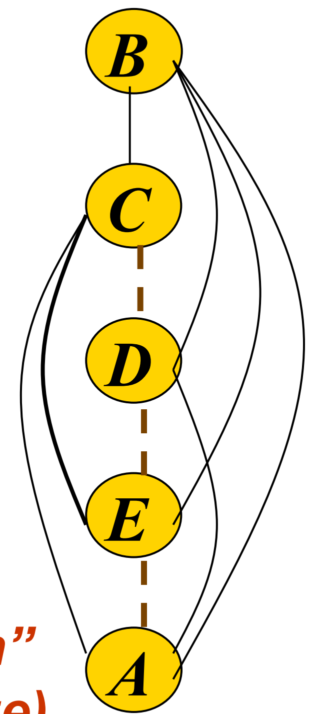
bucket D: $h^C(a,d,e)$

bucket E: $e=0$ $h^D(a,e)$

bucket A: $P(a)$ $h^E(a)$

OPT

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



Generating the MPE-tuple

5. $b' = \arg\max_b P(b \mid a') \times P(d' \mid b, a') \times P(e' \mid b, c')$

4. $c' = \arg\max_c P(c \mid a') \times h^B(a', d', c, e')$

3. $d' = \arg\max_d h^C(a', d, e')$

2. $e' = 0$

1. $a' = \arg\max_a P(a) \cdot h^E(a)$

B: $P(b|a)$ $P(d|b,a)$ $P(e|b,c)$

C: $P(c|a)$ $h^B(a,d,c,e)$

D: $h^C(a,d,e)$

E: $e=0$ $h^D(a,e)$

A: $P(a)$ $h^E(a)$

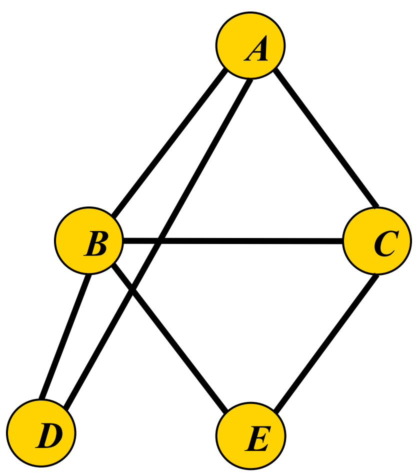
Return (a', b', c', d', e')

Complexity of Elimination

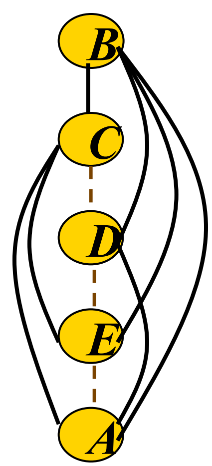
$$O(n \exp(w^*(d)))$$

$w^*(d)$ – the induced width of moral graph along ordering d

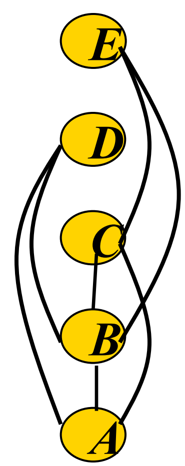
The effect of the ordering:



"Moral" graph



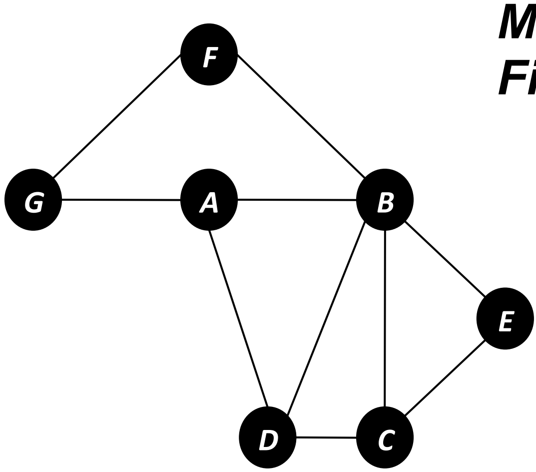
$w^*(d_1) = 4$

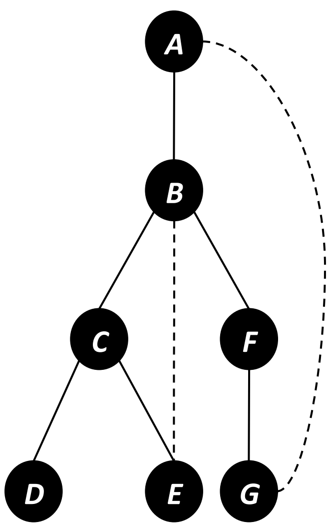


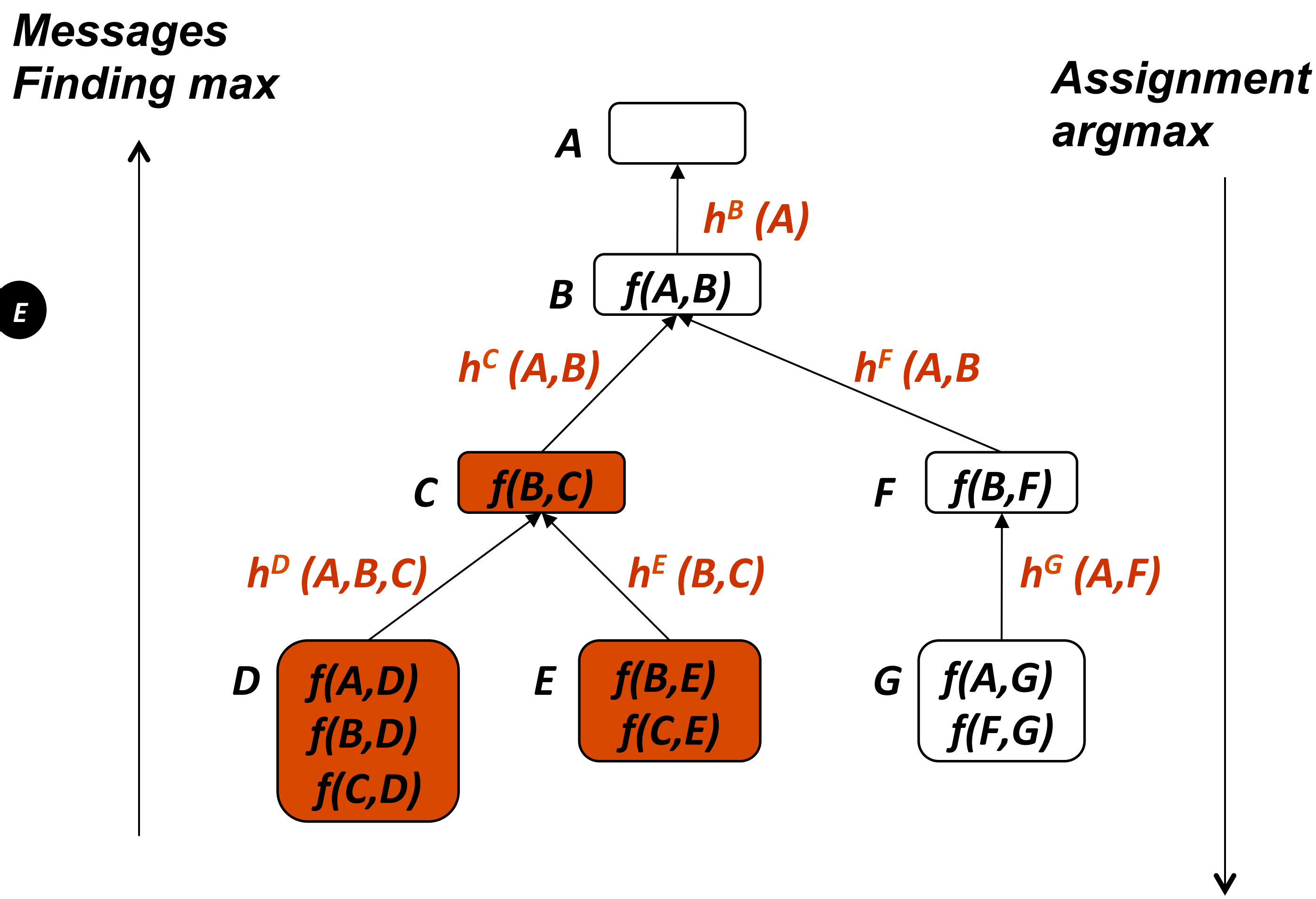
$w^*(d_2) = 2$

Bucket Elimination

$$\min_{a,b,c,d,e,f,g} f(a,b)+f(a,d)+f(b,c)+f(a,d)+f(b,d)+f(c,d)+f(b,e)+f(c,e)+f(b,f)+f(a,g)+f(f,g)=$$





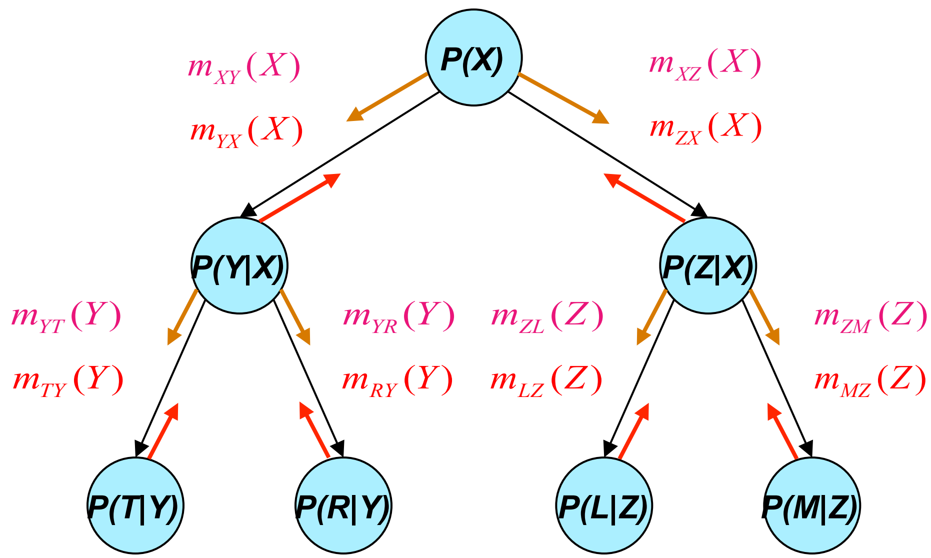


Ordering: (A, B, C, D, E, F, G)

Tree-solving is Easy

Belief updating (sum-prod)

CSP – consistency (projection-join)



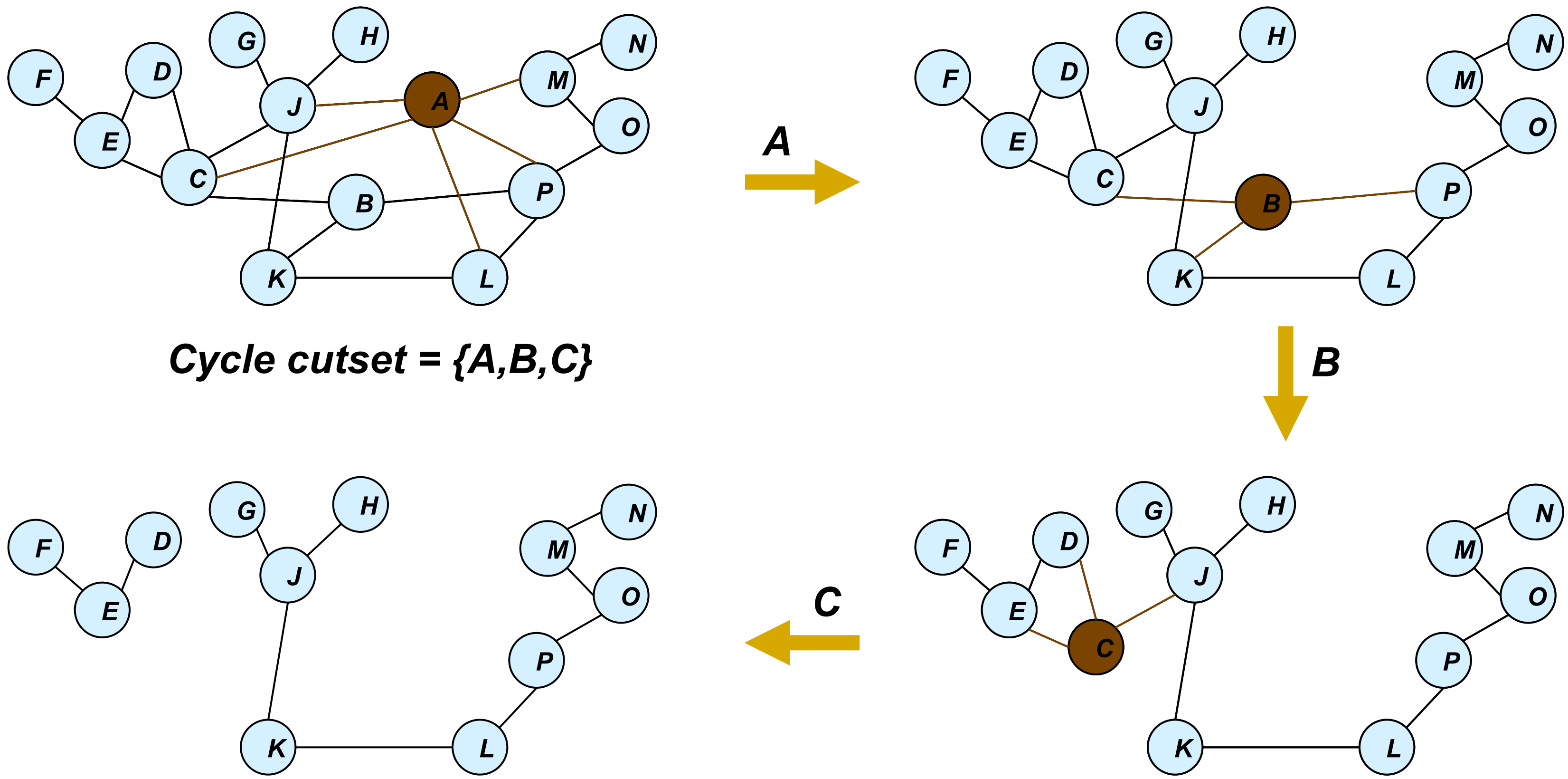
Dynamic Programming, Inference

MPE (max-prod)

#CSP (sum-prod)

Trees are processed in linear time and memory
Message-passing

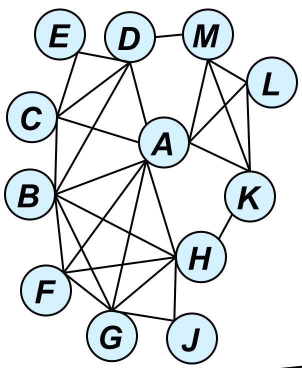
Conditioning and Cycle cutset



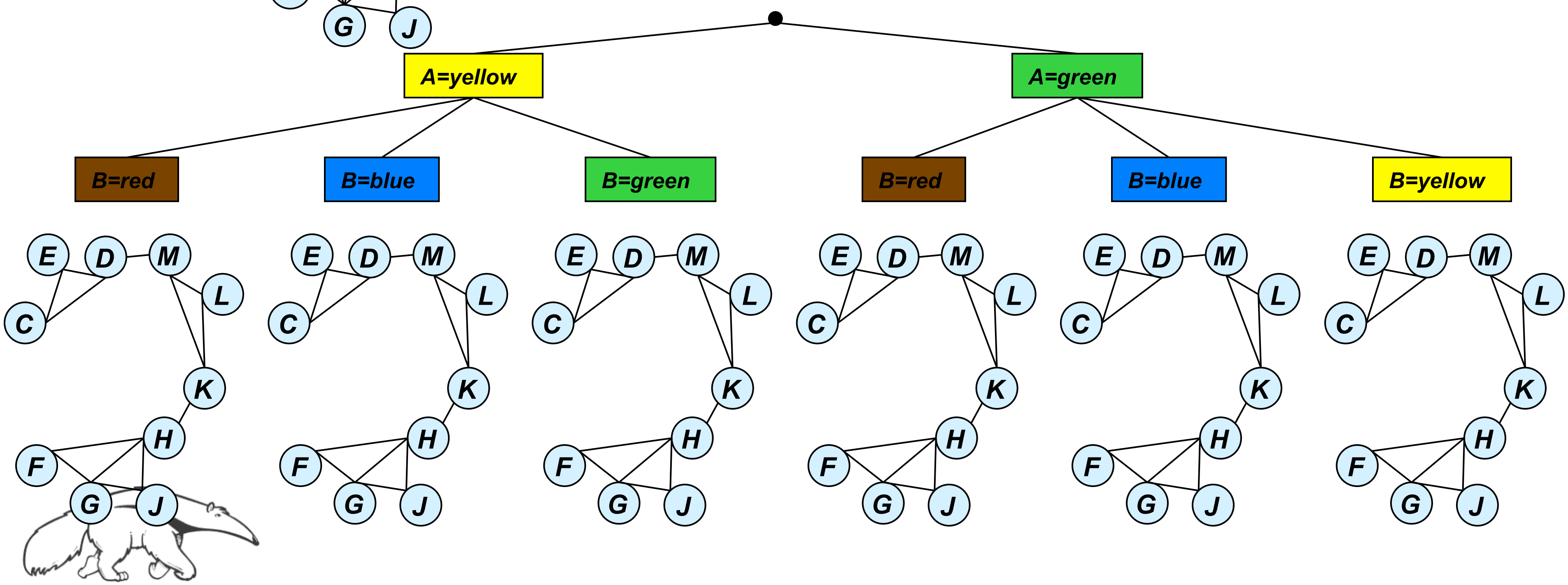
Search over the Cutset (cont)

Time exp in cycle-cutset
Memory-linear

Graph Coloring problem

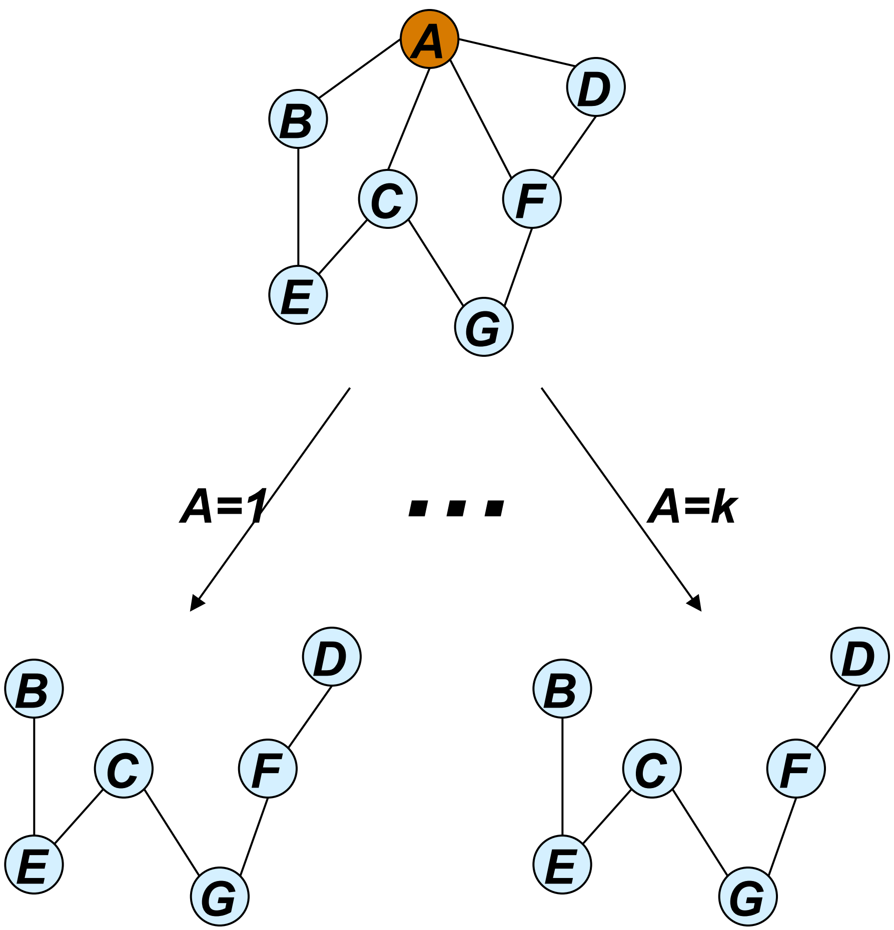


- ***Inference may require too much memory***
- ***Condition on some of the variables***



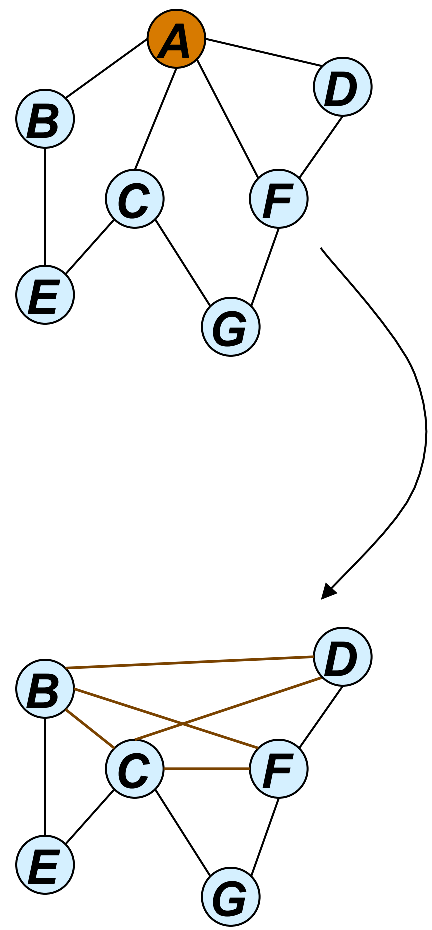
Conditioning vs. Elimination

Conditioning (search)



k "sparser" problems

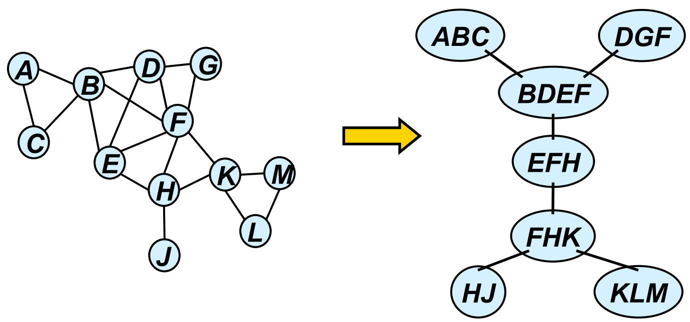
Elimination (inference)



1 "denser" problem

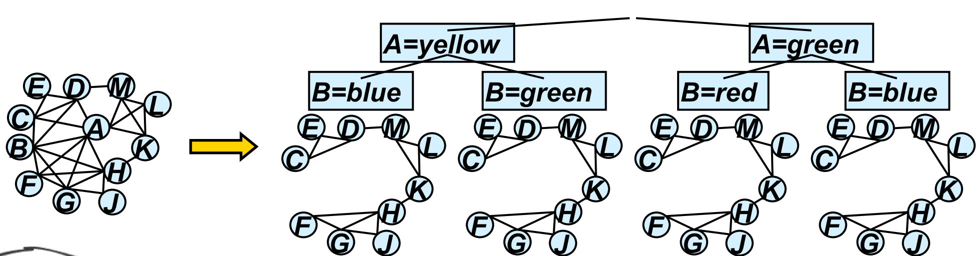
Inference vs. Conditioning

- By Inference (thinking)



Exponential in treewidth
Time and memory

- ***By Conditioning (guessing)***

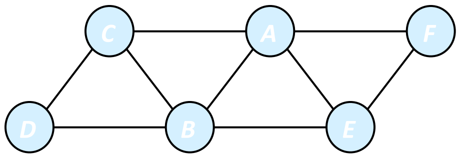


Exponential in cycle-cutset
Time-wise, linear memory

Outline

- Graphical models: reasoning principles
- Inference
- Advancing Search via AND/OR Search
- Lower Bounding schemes for inference
- Lower-bounding heuristic for AND/OR search
- Experiments

The Search Space



A	B	f_1
0	0	2
0	1	0
1	0	1
1	1	4

A	C	f_2
0	0	3
0	1	0
1	0	0
1	1	1

A	E	f_3
0	0	0
0	1	3
1	0	2
1	1	0

A	F	f_4
0	0	2
0	1	0
1	0	0
1	1	2

B	C	f_5
0	0	0
0	1	1
1	0	2
1	1	4

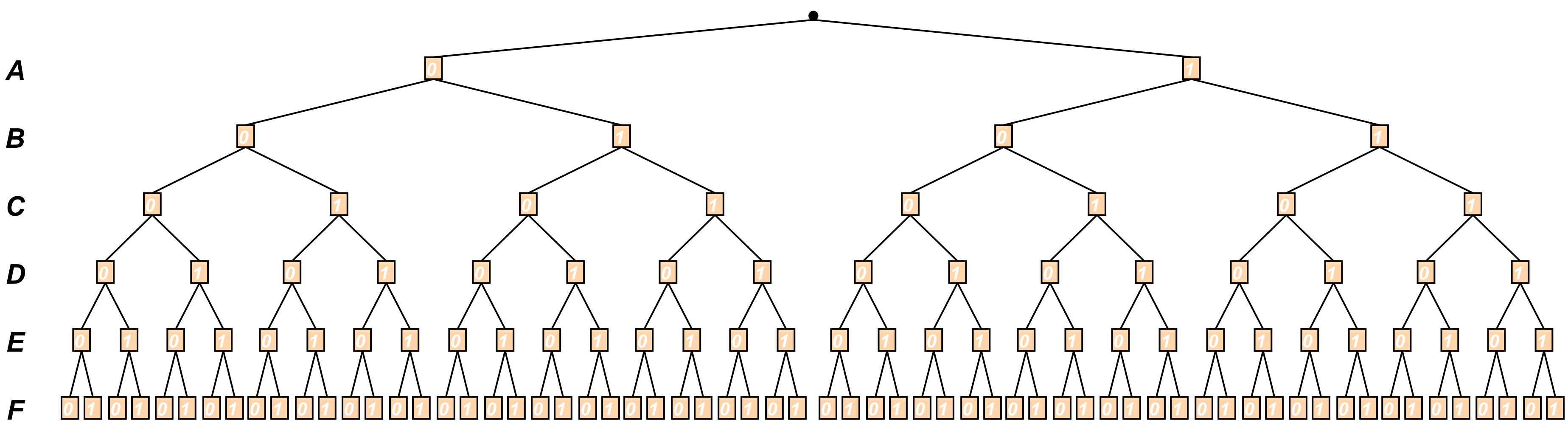
B	D	f_6
0	0	4
0	1	2
1	0	1
1	1	0

B	E	f_7
0	0	3
0	1	2
1	0	1
1	1	0

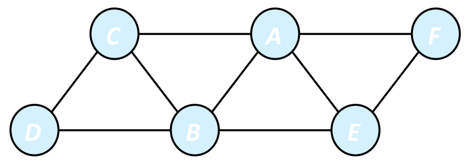
C	D	f_8
0	0	1
0	1	4
1	0	0
1	1	0

E	F	f_9
0	0	1
0	1	0
1	0	0
1	1	2

$$f(X) = \min_X \sum_{i=1}^{9} f_i(X)$$



The Search Space



A	B	f_1
0	0	2
0	1	0
1	0	1
1	1	4

A	C	f_2
0	0	3
0	1	0
1	0	0
1	1	1

A	E	f_3
0	0	0
0	1	3
1	0	2
1	1	0

A	F	f_4
0	0	2
0	1	0
1	0	0
1	1	2

B	C	f_5
0	0	0
0	1	1
1	0	2
1	1	4

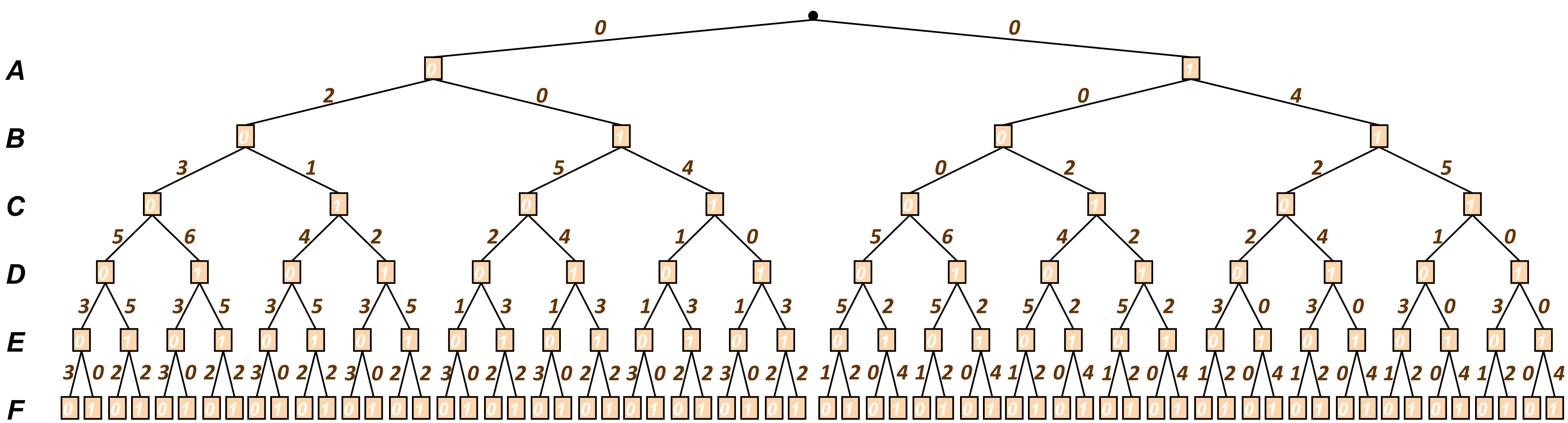
B	D	f_6
0	0	4
0	1	2
1	0	1
1	1	0

B	E	f_7
0	0	3
0	1	2
1	0	1
1	1	0

C	D	f_8
0	0	1
0	1	4
1	0	0
1	1	0

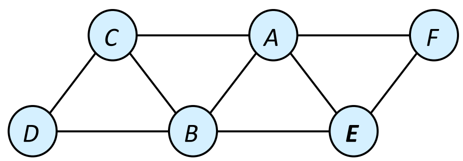
E	F	f_9
0	0	1
0	1	0
1	0	0
1	1	2

$$f(X) = \min_X \sum_{i=1}^{9} f_i(X)$$



are calculated based on cost components

The Search Space



A	B	f_1
0	0	2
0	1	0
1	0	1
1	1	4

A	C	f_2
0	0	3
0	1	0
1	0	0
1	1	1

A	E	f_3
0	0	0
0	1	3
1	0	2
1	1	0

A	F	f_4
0	0	2
0	1	0
1	0	0
1	1	2

B	C	f_5
0	0	0
0	1	1
1	0	2
1	1	4

B	D	f_6
0	0	4
0	1	2
1	0	1
1	1	0

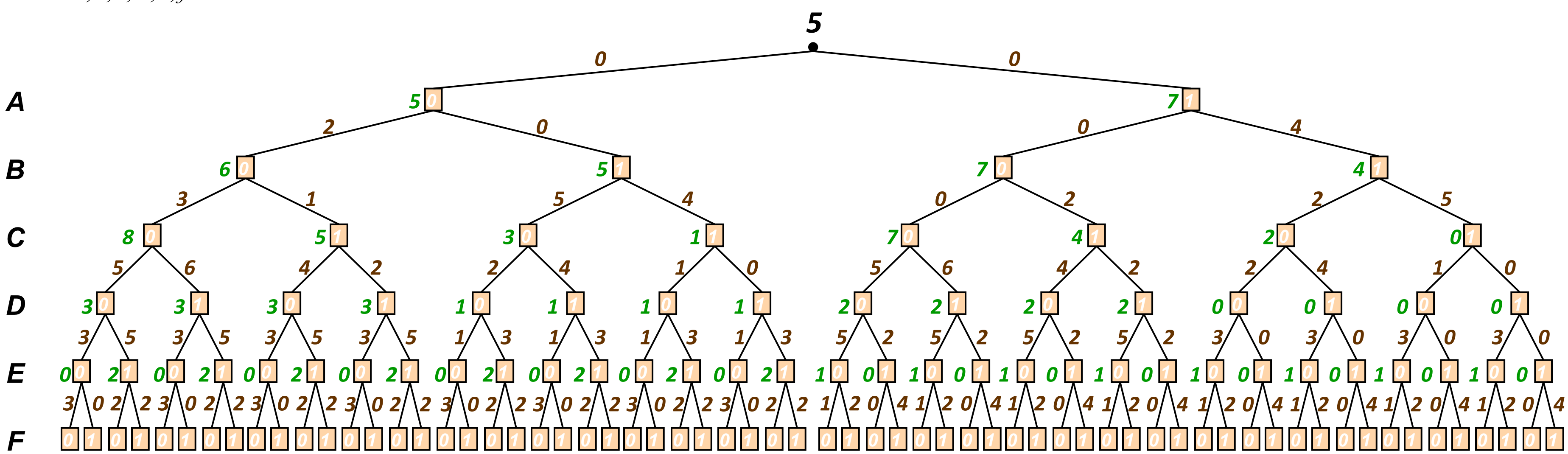
B	E	f_7
0	0	3
0	1	2
1	0	1
1	1	0

C	D	f_8
0	0	1
0	1	4
1	0	0
1	1	0

E	F	f_9
0	0	1
0	1	0
1	0	0
1	1	2

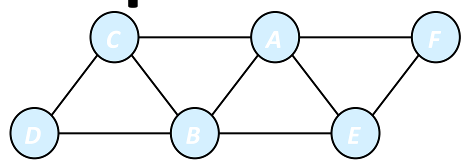
$$f(\mathrm{X}) = \sum_{i=1}^{9} f_i(\mathrm{X})$$

$$\min_{a,b,c,d,e,f} f_1(a,b) + f_2(a,c) + f_3(a,f) + f_4(b,c) + f_5(b,d) + f_6(b,e) + f_7(c,d) + f_8(e,f)$$



***Node value* = minimal cost solution below it**

An Optimal Solution



A	B	f_1
0	0	2
0	1	0
1	0	1
1	1	4

A	C	f_2
0	0	3
0	1	0
1	0	0
1	1	1

A	E	f_3
0	0	0
0	1	3
1	0	2
1	1	0

A	F	f_4
0	0	2
0	1	0
1	0	0
1	1	2

B	C	f_5
0	0	0
0	1	1
1	0	2
1	1	4

B	D	f_6
0	0	4
0	1	2
1	0	1
1	1	0

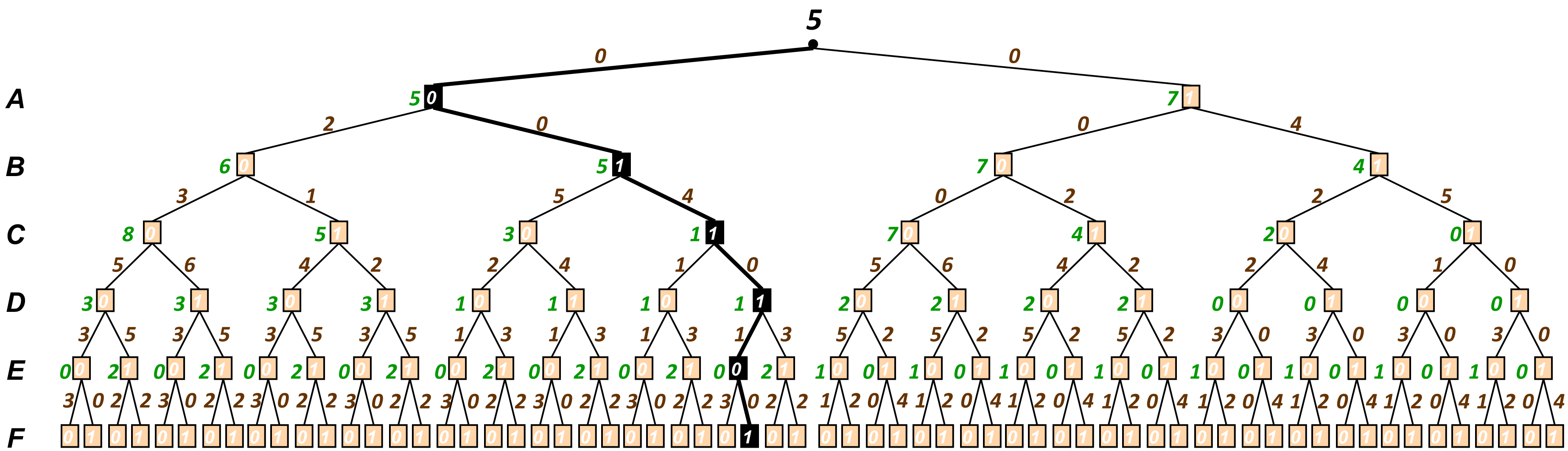
B	E	f_7
0	0	3
0	1	2
1	0	1
1	1	0

C	D	f_8
0	0	1
0	1	4
1	0	0
1	1	0

E	F	f_9
0	0	1
0	1	0
1	0	0
1	1	2

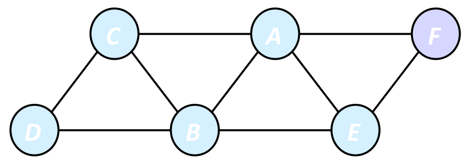
$$f(\mathbf{X}) = \sum_{i=1}^{9} f_i(\mathbf{X})$$

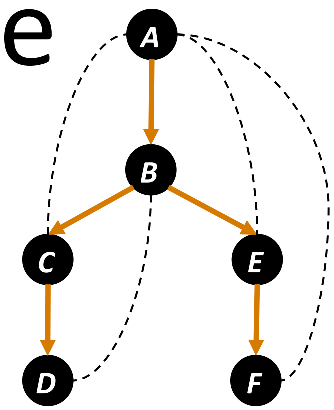
$$\min_{a,b,c,d,e,f} f_1(a,b) + f_2(a,c) + f_3(a,f) + f_4(b,c) + f_5(b,d) + f_6(b,e) + f_7(c,d) + f_8(e,f)$$



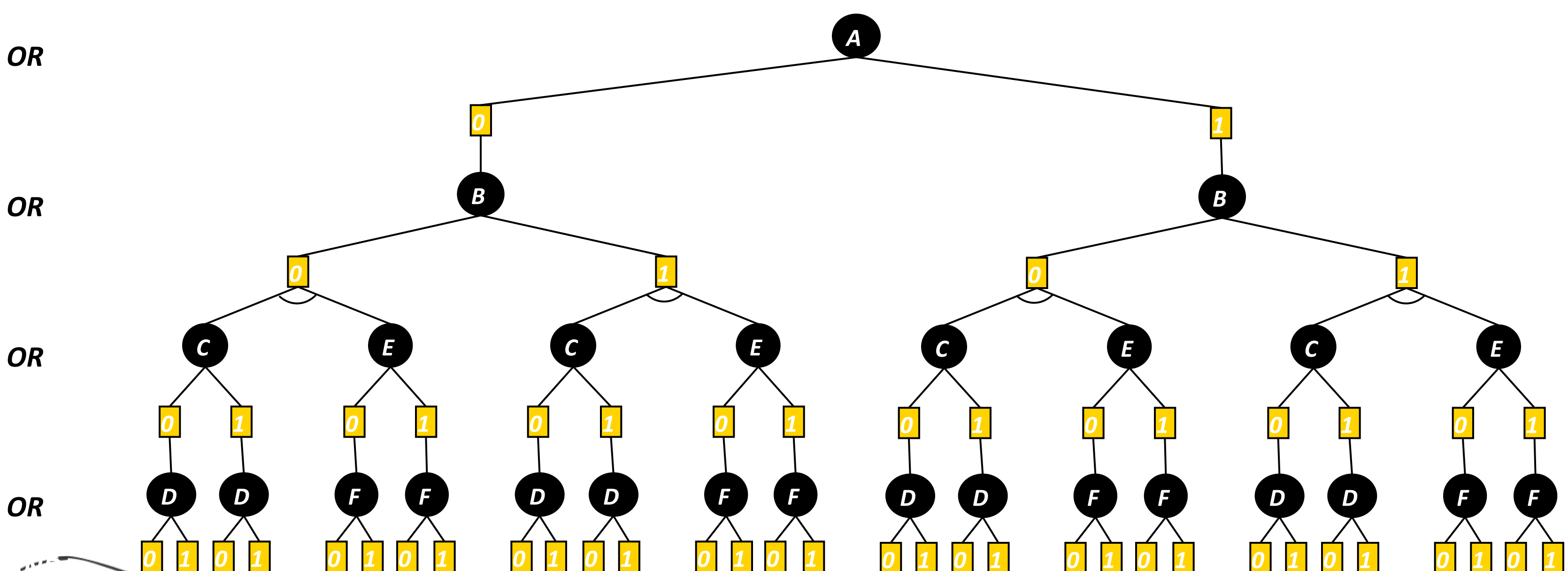
An optimal assignment is A=0, B=1, C=1, D=1, E=0, F=1 with cost 5

The AND/OR Search Tree

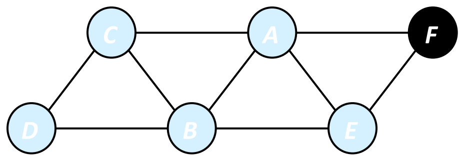


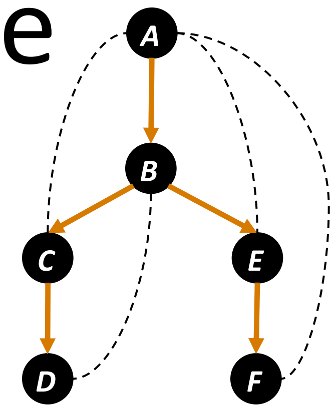


***Pseudo tree** (Freuder & Quinn85)*

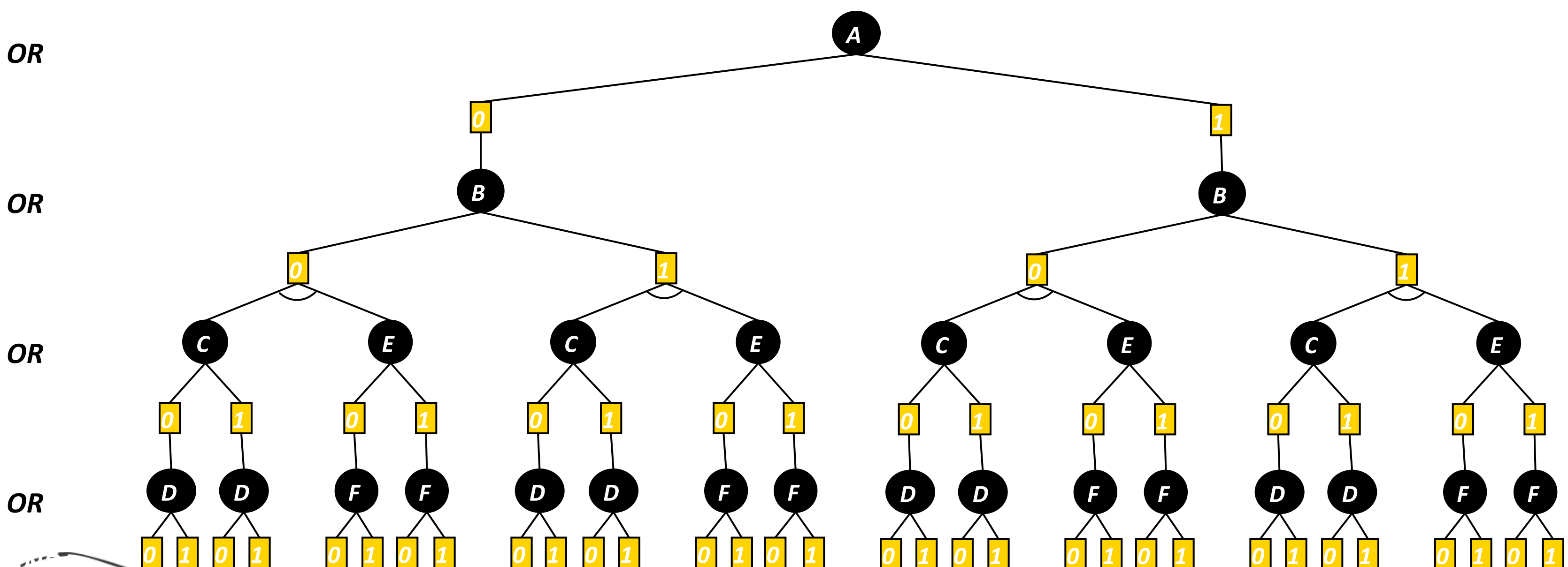


The AND/OR Search Tree



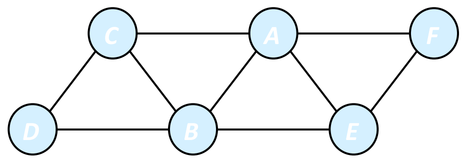


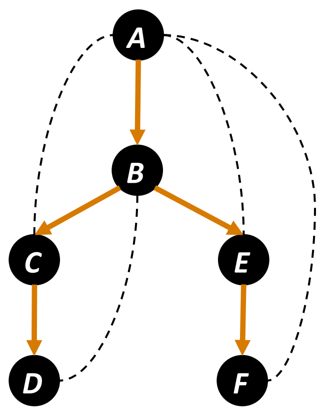
Pseudo tree



A solution subtree is

Weighted AND/OR Search Tree





A	B	f_1
0	0	2
0	1	0
1	0	1
1	1	4

A	C	f_2
0	0	3
0	1	0
1	0	0
1	1	1

A	E	f_3
0	0	0
0	1	3
1	0	2
1	1	0

A	F	f_4
0	0	2
0	1	0
1	0	0
1	1	2

B	C	f_5
0	0	0
0	1	1
1	0	2
1	1	4

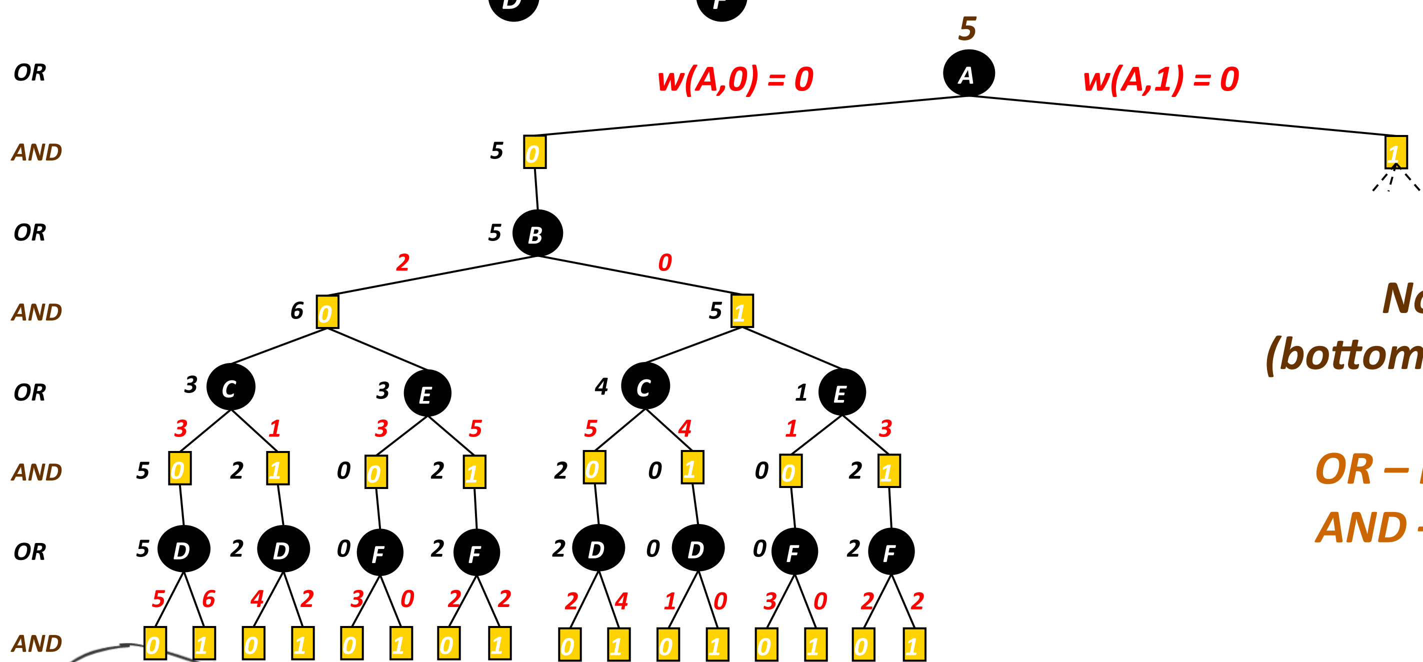
B	D	f_6
0	0	4
0	1	2
1	0	1
1	1	0

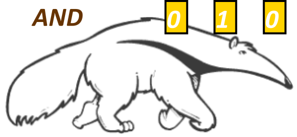
B	E	f_7
0	0	3
0	1	2
1	0	1
1	1	0

C	D	f_8
0	0	1
0	1	4
1	0	0
1	1	0

E	F	f_9
0	0	1
0	1	0
1	0	0
1	1	2

$$f(\mathrm{X}) = \sum_{i=1}^{9} f_i(\mathrm{X})$$

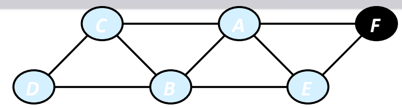


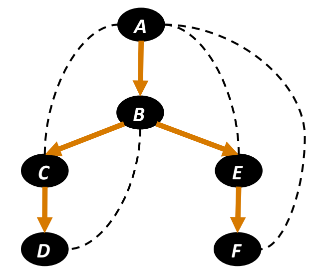


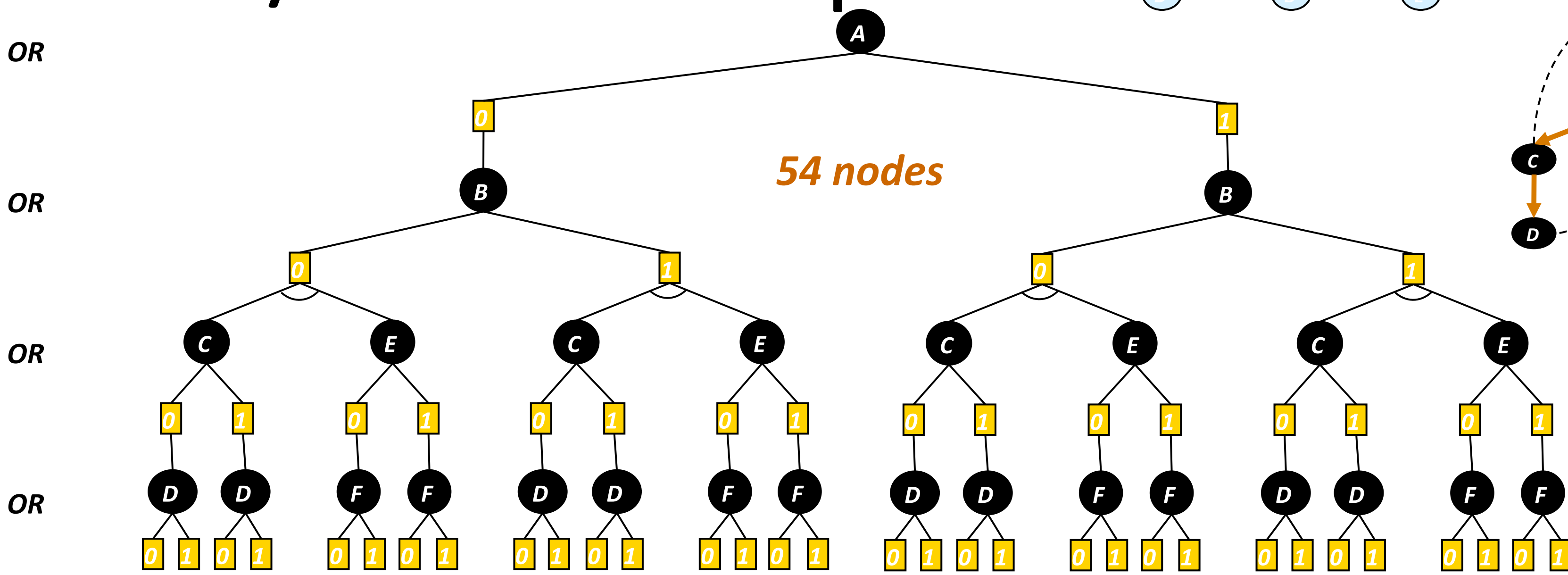
Node Value (bottom-up evaluation)

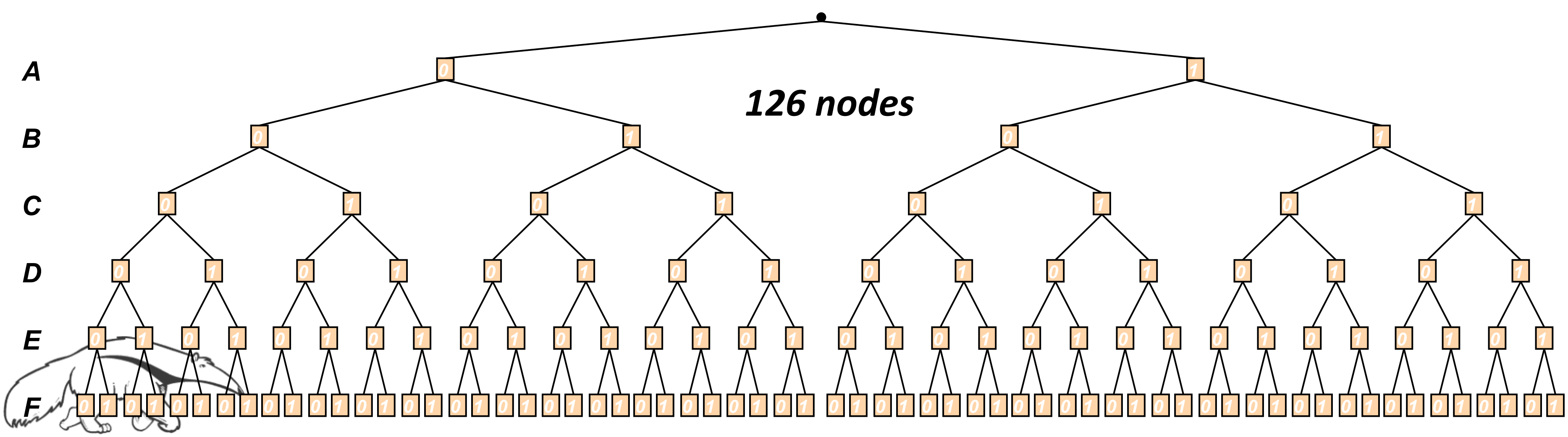
OR – minimization
AND – summation

AND/OR vs. OR Spaces



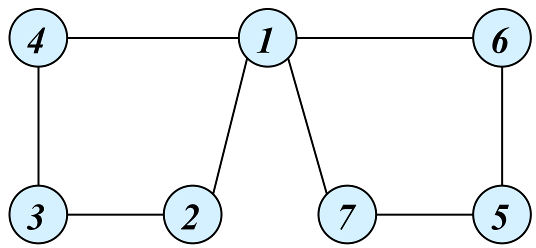




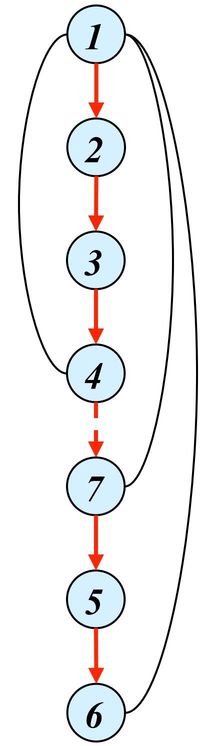


Pseudo-Trees

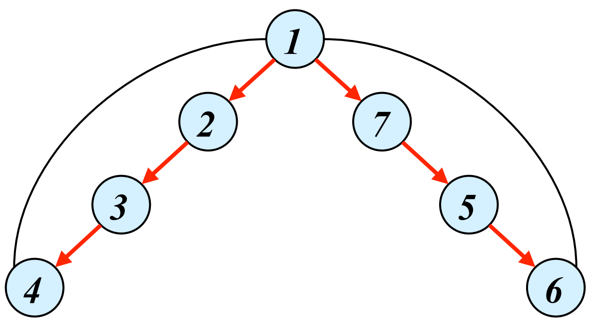
(Freuder 85, Bayardo 95, Bodlaender and Gilbert, 91)

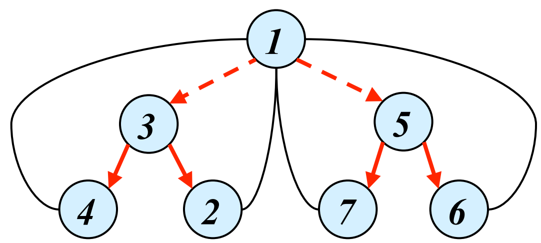


(a) Graph



$m <= w^* \log n$





(b) DFS tree
depth=3

(c) pseudo- tree
depth=2

(d) Chain
depth=6

AND/OR Tree DFS Algorithm (Belief Updating)

$P(E \mid A,B)$

A	B	E=0	E=1
0	0	.4	.6
0	1	.5	.5
1	0	.7	.3
1	1	.2	.8

Evidence: E=0

$P(B \mid A)$

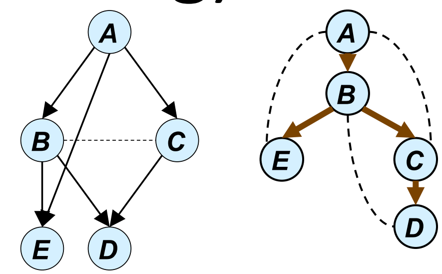
A	B=0	B=1
0	.4	.6
1	.1	.9

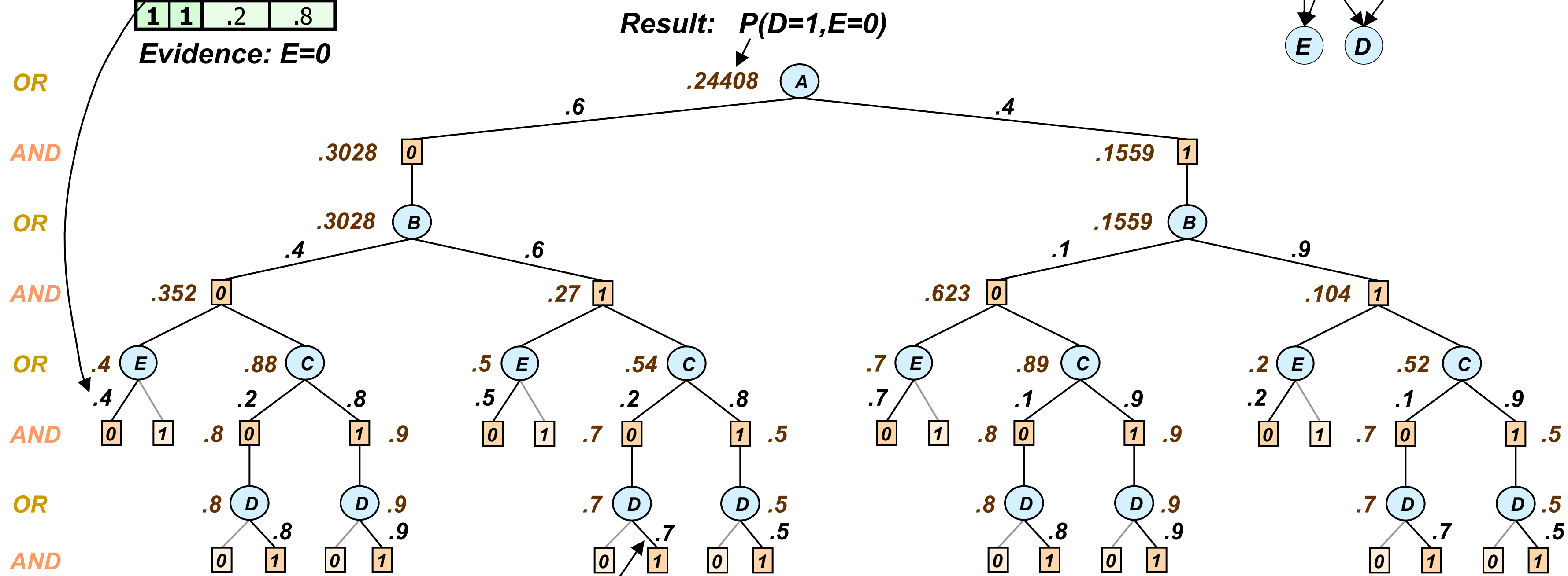
$P(C \mid A)$

A	C=0	C=1
0	.2	.8
1	.7	.3

$P(A)$

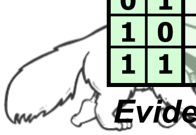
A	P(A)
0	.6
1	.4





$P(D \mid B,C)$

B	C	D=0	D=1
0	0	.2	.8
0	1	.1	.9
1	0	.3	.7
1	1	.5	.5



Evidence: D=1

OR node: Marginalization operator (summation)

AND node: Combination operator (product)

Value of node = updated belief for sub-problem below

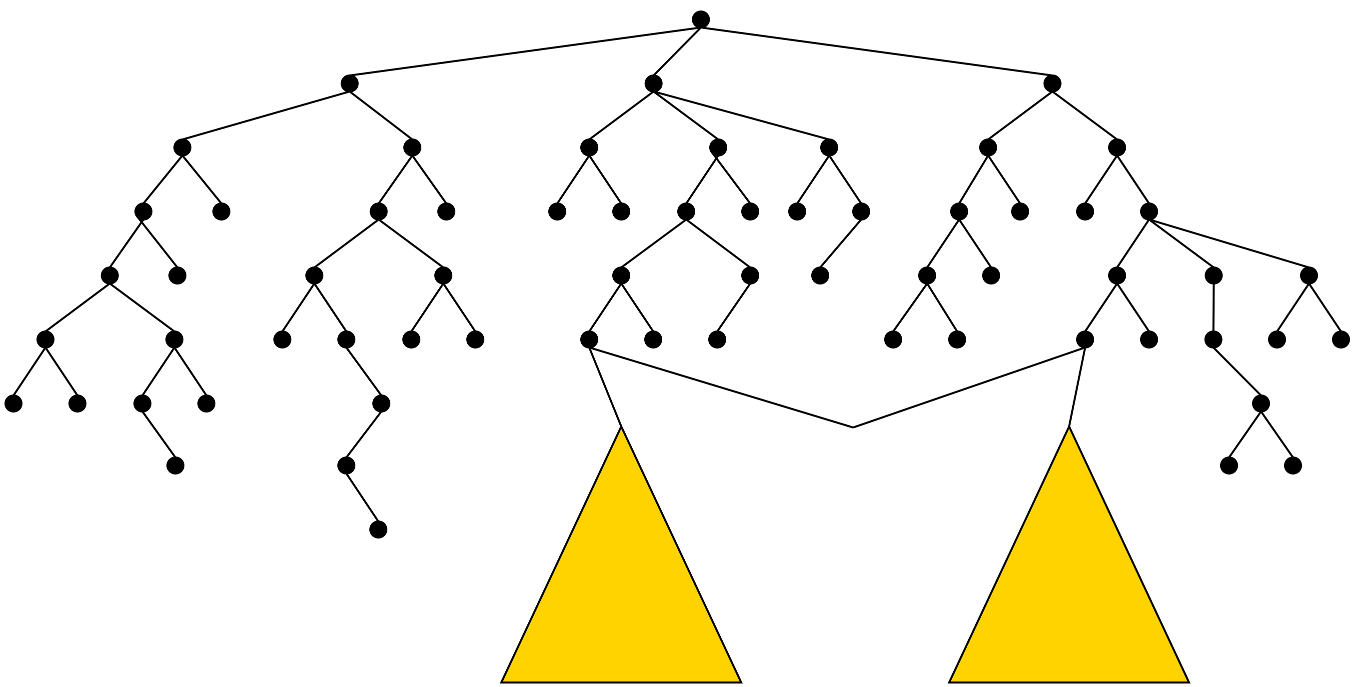
Complexity of AND/OR Tree Search

	AND/OR tree	**OR tree**
Space	$O(n)$	$O(n)$
Time	$O(n\ k^{m})$ $O(n\ k^{w^* \log n})$ (Freuder & Quinn85), (Collin, Dechter & Katz91), (Bayardo & Miranker95), (Darwiche01)	$O(k^{n})$

k = *domain size*
m = *depth of pseudo-tree*
n = *number of variables*
*w**= *treewidth*

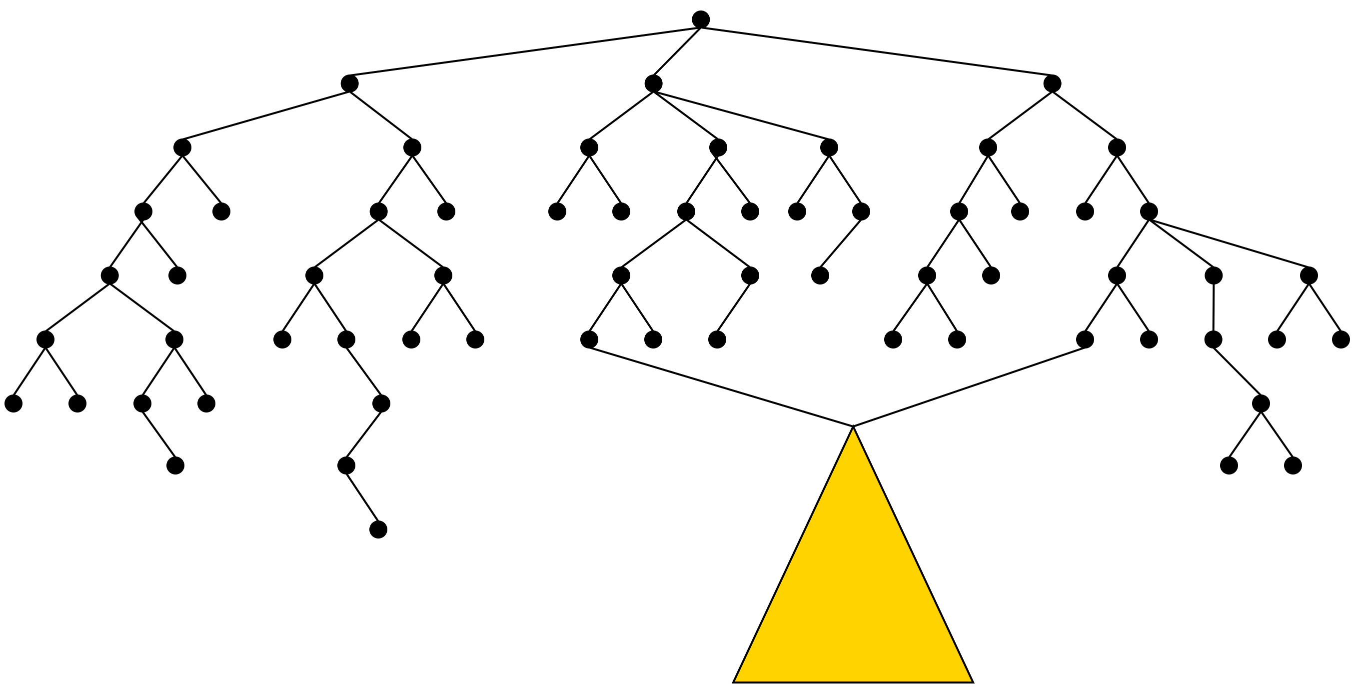
From Search Trees to Search Graphs

- Any two nodes that root identical sub-trees or sub-graphs can be merged

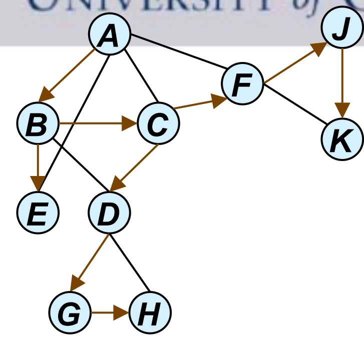


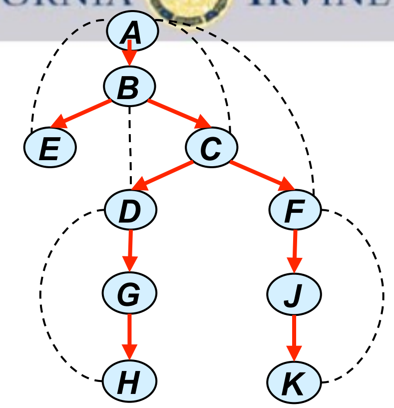
From Search Trees to Search Graphs

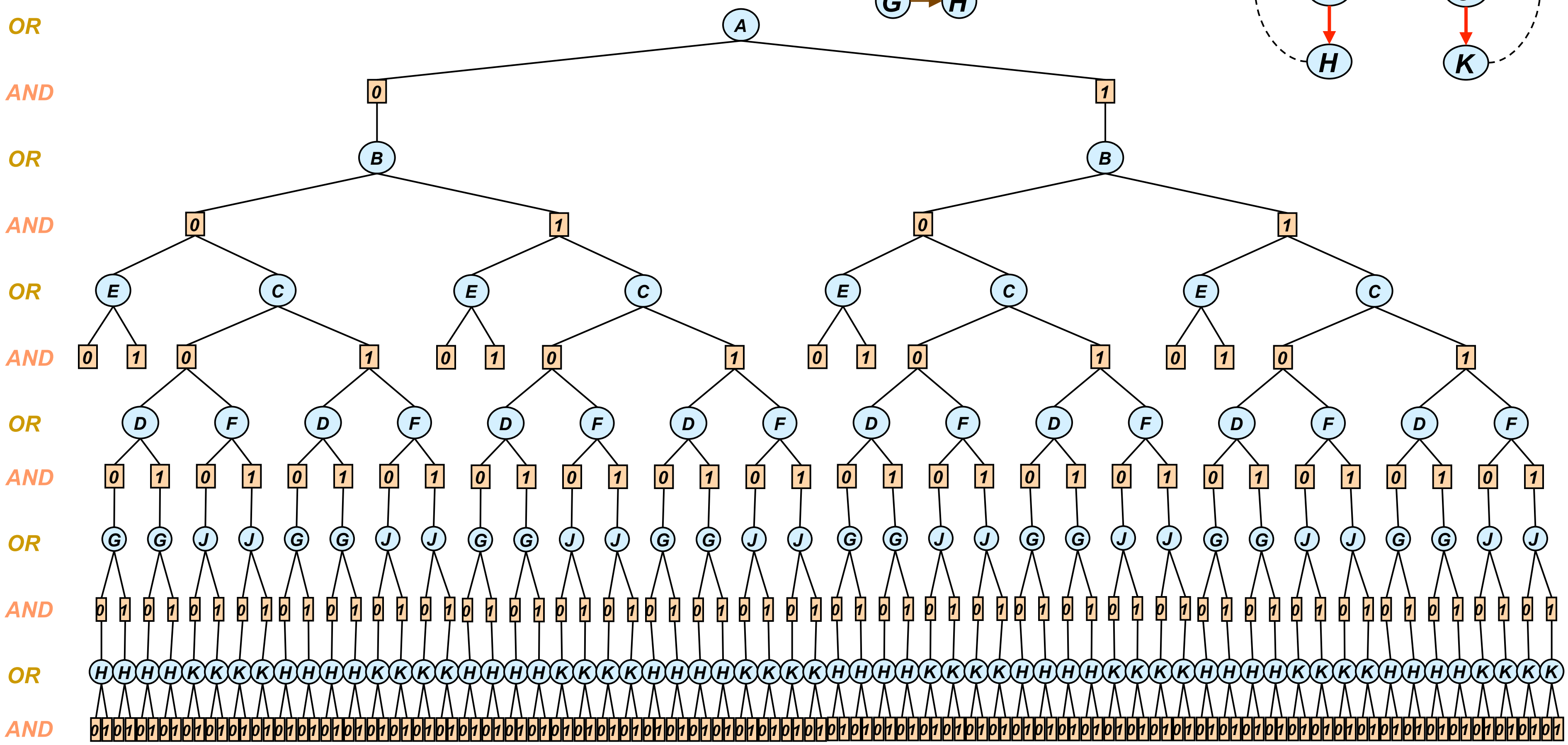
- Any two nodes that root identical sub-trees or sub-graphs can be merged



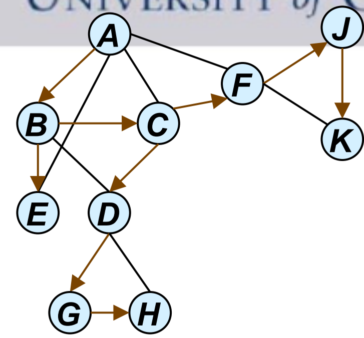
From AND/OR Tree

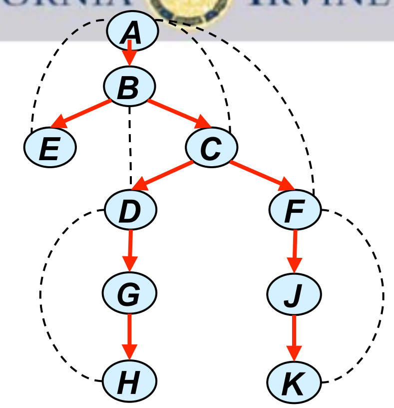


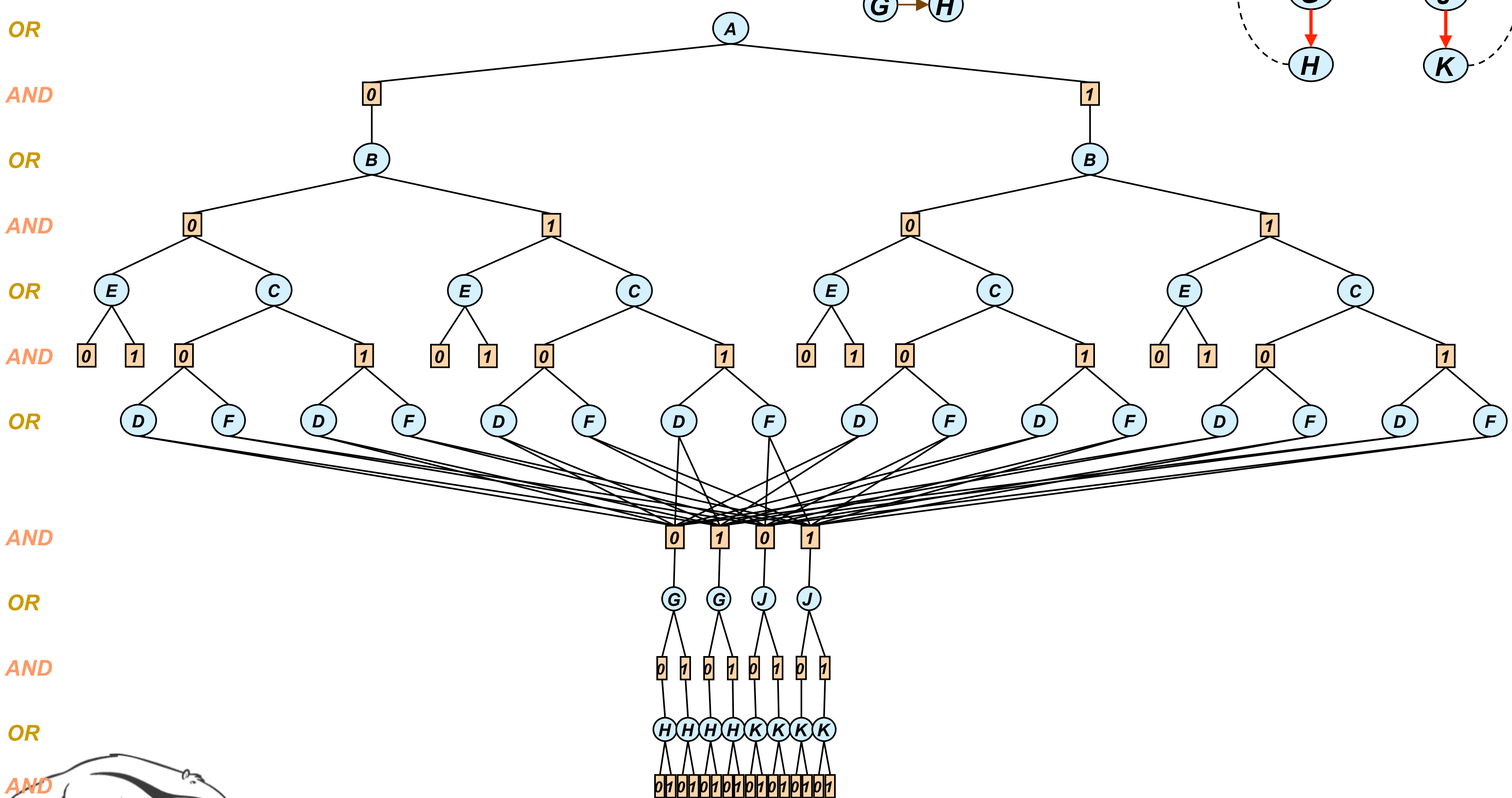




An AND/OR Graph



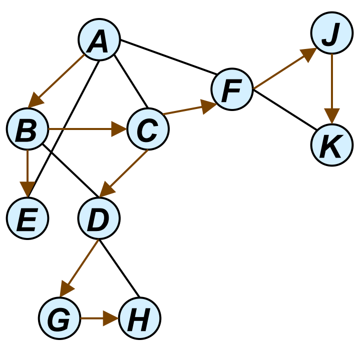


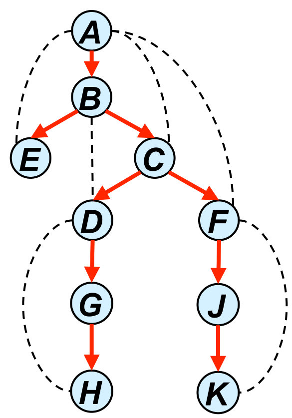


Context-based Caching

- Caching is possible when context is the same

- context = parent-separator set in induced pseudo-graph
  = current variable +
  parents connected to subtree below





context(B) = {A, B}

context(c) = {A,B,C}

context(D) = {D}

context(F) = {F}

AND/OR Tree DFS Algorithm (Belief Updating)

$P(E \mid A,B)$

A	B	E=0	E=1
0	0	.4	.6
0	1	.5	.5
1	0	.7	.3
1	1	.2	.8

Evidence: E=0

$P(B \mid A)$

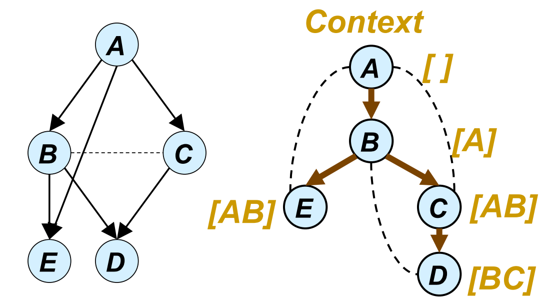
A	B=0	B=1
0	.4	.6
1	.1	.9

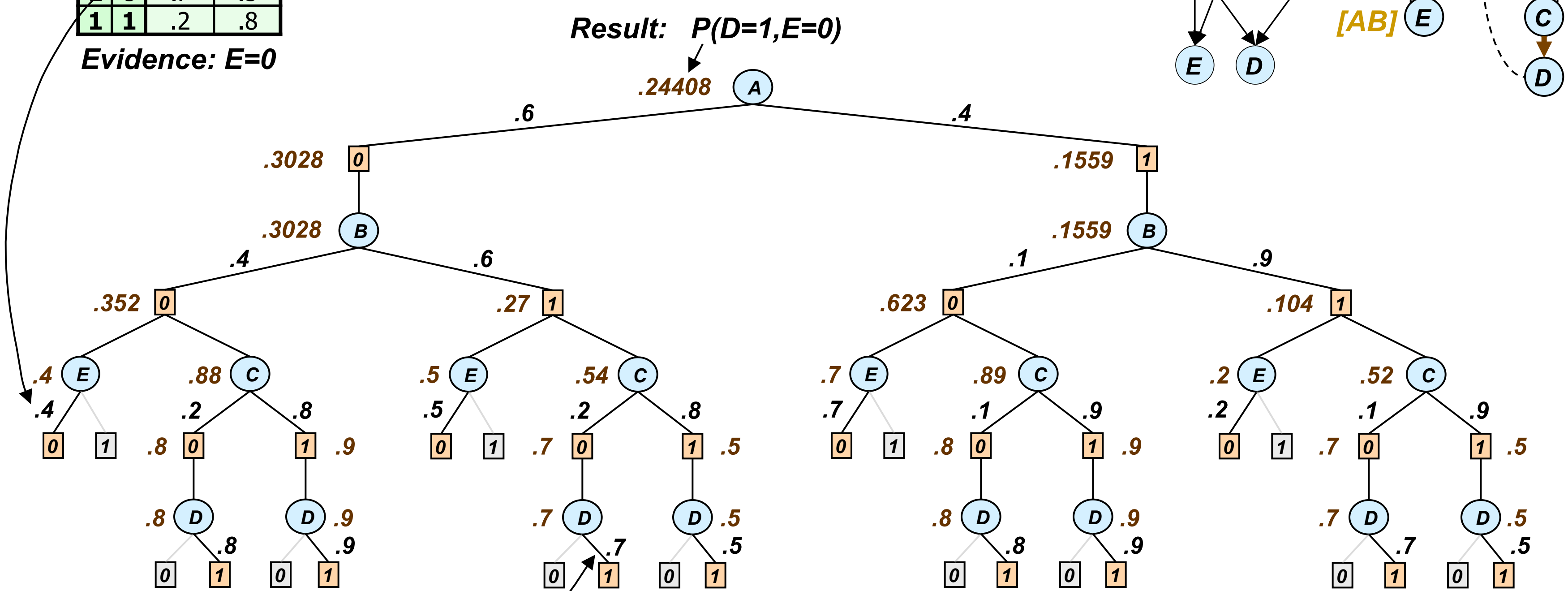
$P(C \mid A)$

A	C=0	C=1
0	.2	.8
1	.7	.3

$P(A)$

A	P(A)
0	.6
1	.4





$P(D \mid B,C)$

B	C	D=0	D=1
0	0	.2	.8
0	1	.1	.9
1	0	.3	.7
1	1	.5	.5

Evidence: D=1

OR node: Marginalization operator (summation)

AND node: Combination operator (product)

Value of node = updated belief for sub-problem below

AND/OR Graph DFS Algorithm (Belief Updating)

$P(E \mid A, B)$

A	B	E=0	E=1
0	0	.4	.6
0	1	.5	.5
1	0	.7	.3
1	1	.2	.8

Evidence: E=0

$P(B \mid A)$

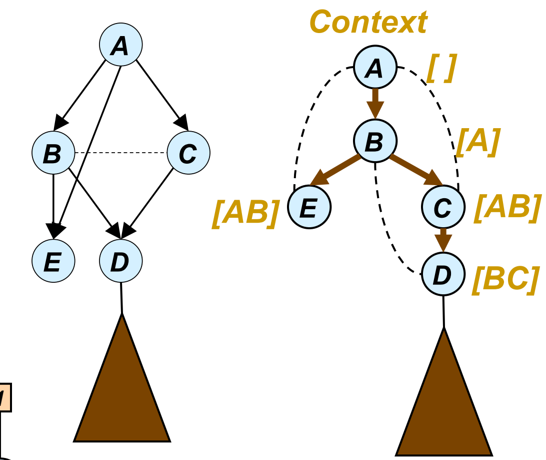
A	B=0	B=1
0	.4	.6
1	.1	.9

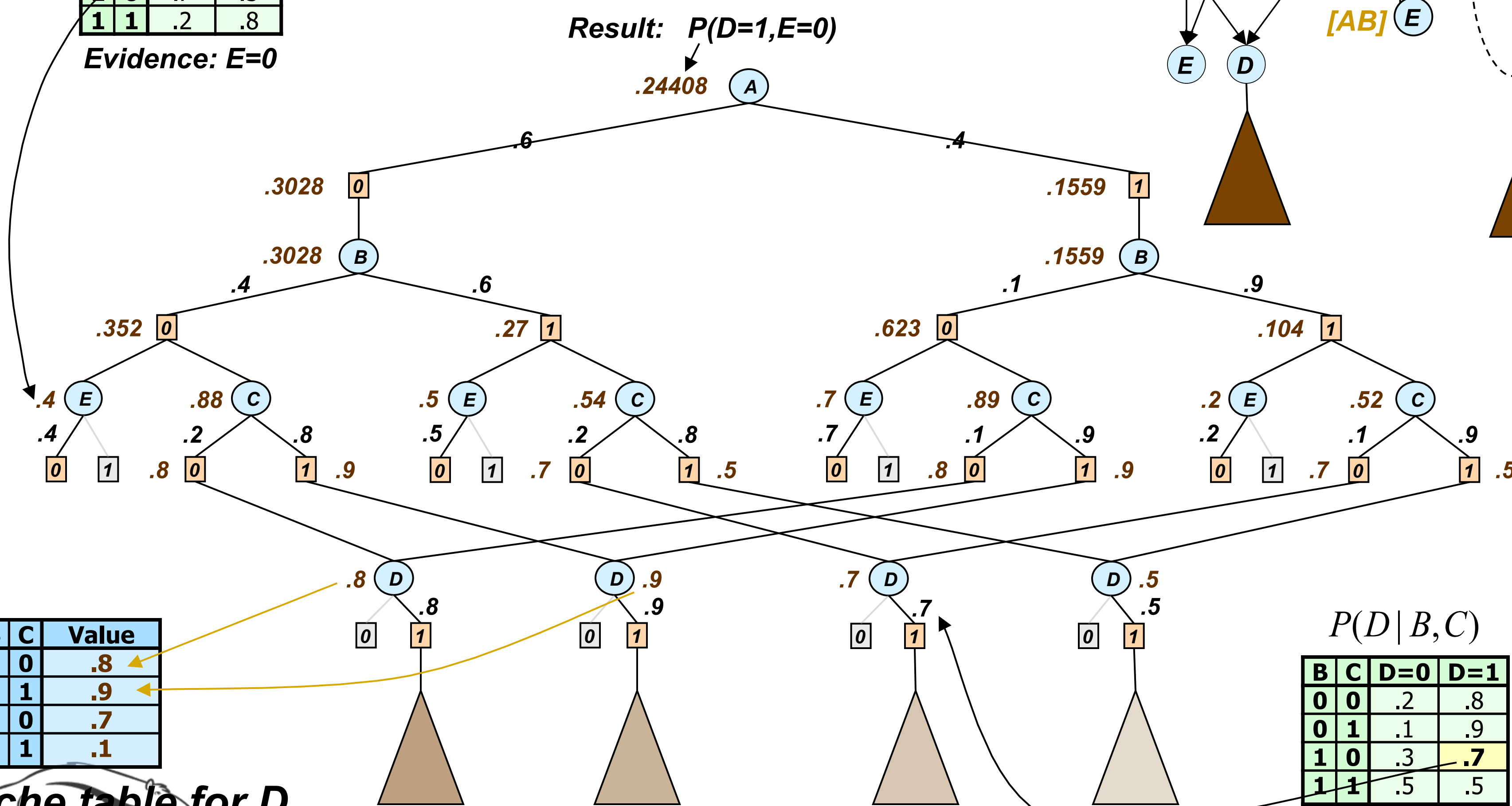
$P(C \mid A)$

A	C=0	C=1
0	.2	.8
1	.7	.3

$P(A)$

A	P(A)
0	.6
1	.4





B	C	Value
0	0	.8
0	1	.9
1	0	.7
1	1	.1

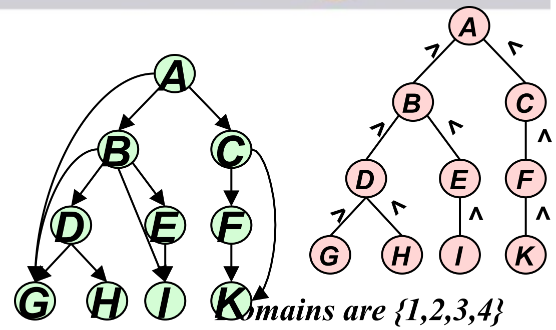
Cache table for D

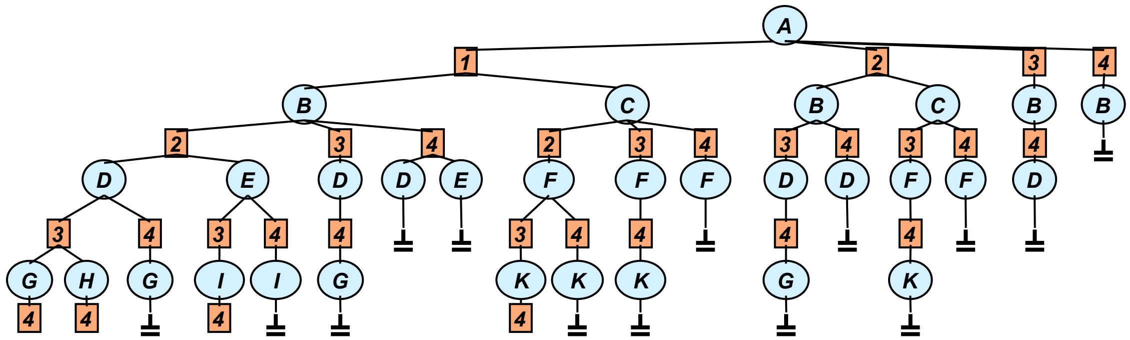
$P(D \mid B, C)$

B	C	D=0	D=1
0	0	.2	.8
0	1	.1	.9
1	0	.3	.7
1	1	.5	.5

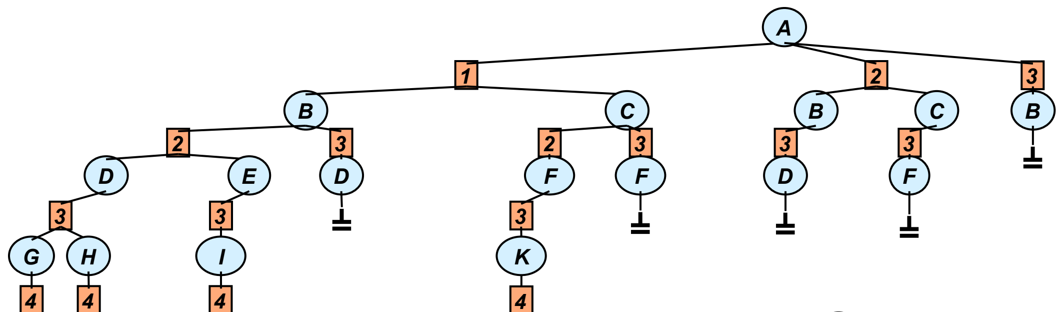
Evidence: D=1

The Effect of Constraint Propagation

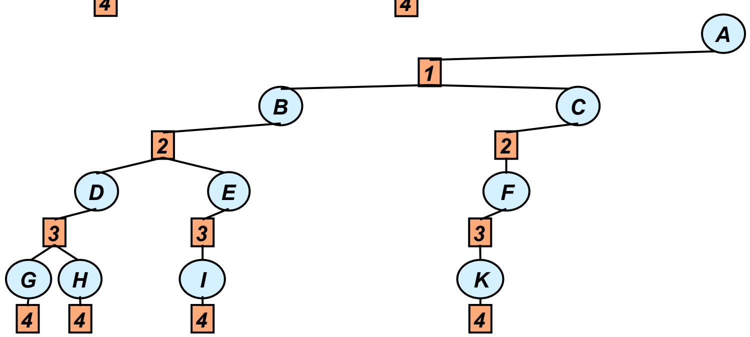




CONSTRAINTS ONLY

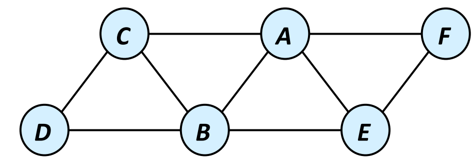


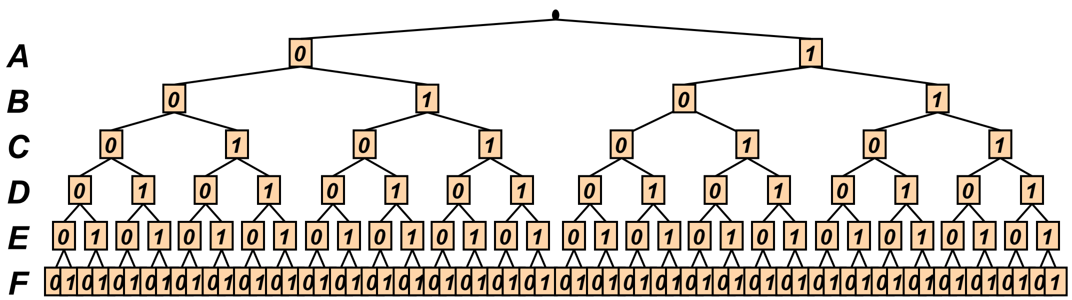
FORWARD CHECKING



MAINTAINING ARC CONSISTENCY

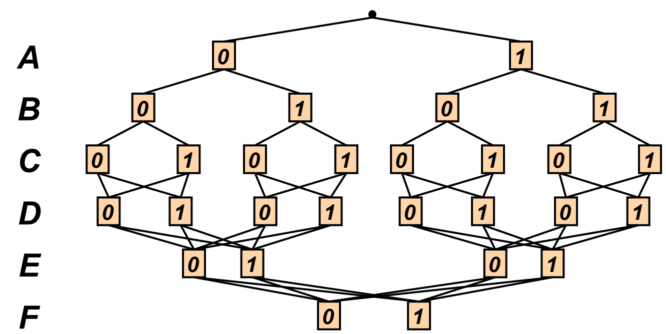
All Four Search Spaces





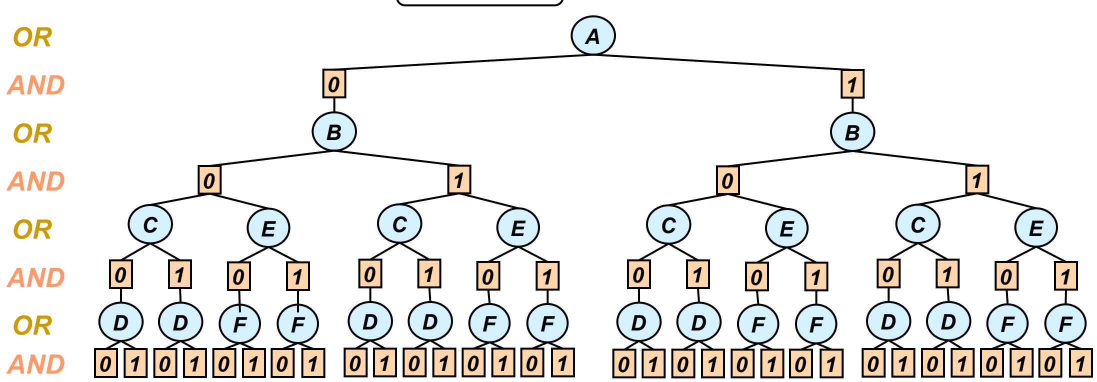
Full OR search tree

126 nodes



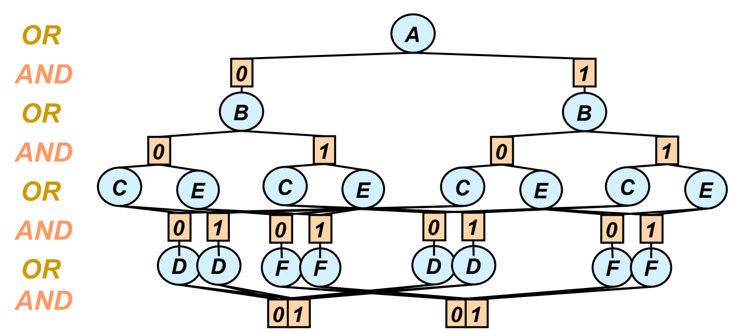
Context minimal OR search graph

28 nodes



Full AND/OR search tree

54 AND nodes



Context minimal AND/OR search graph

18 AND nodes

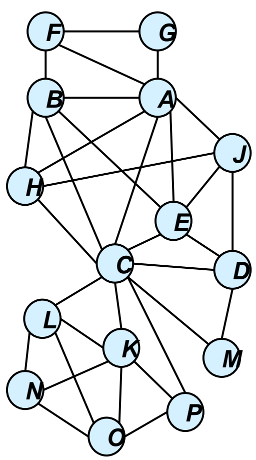
Any query is best computed
Over the c-minimal AO space

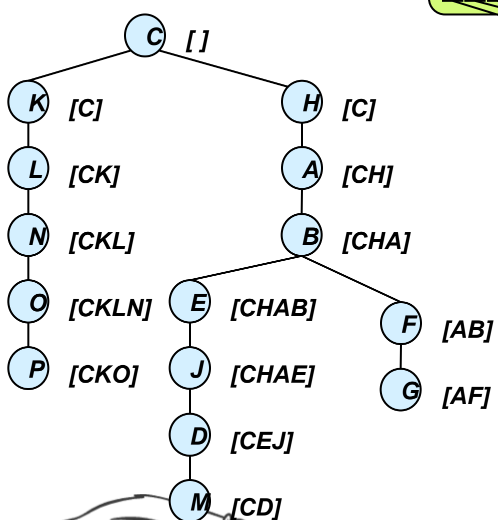
Complexity of AND/OR Graph Search

	AND/OR graph	**OR graph**
Space	$O(n\ k^{w^*})$	$O(n\ k^{pw^*})$
Time	$O(n\ k^{w^*})$	$O(n\ k^{pw^*})$

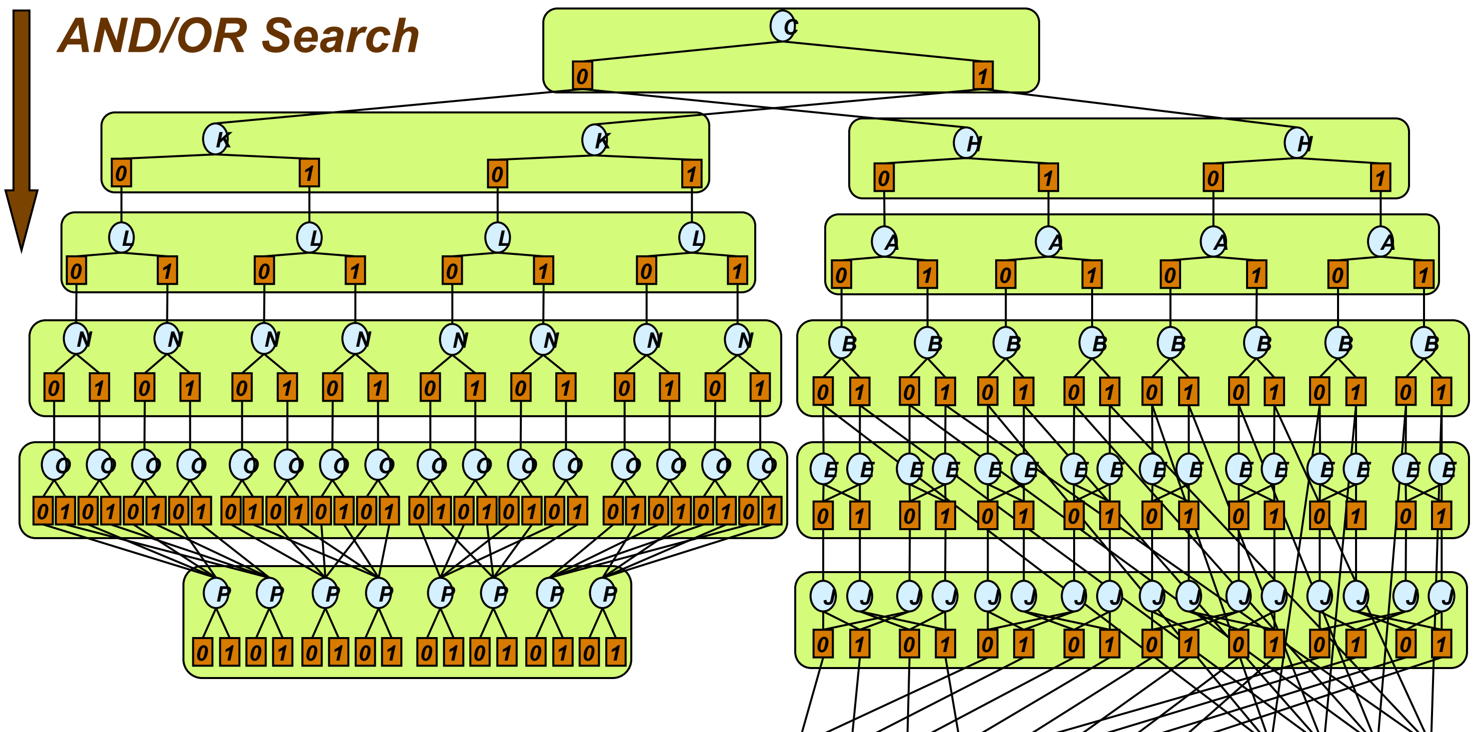
***k* = domain size**
***n* = number of variables**
w*= treewidth
pw*= pathwidth

AND/OR Context Minimal Graph



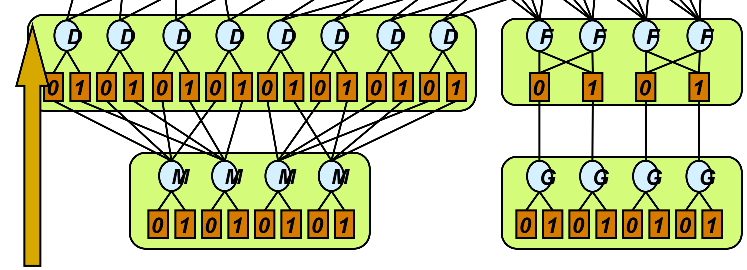


(C K H A B E J L N O D P M F G)



***THEOREM** (Mateescu and Dechter, 2005) :*

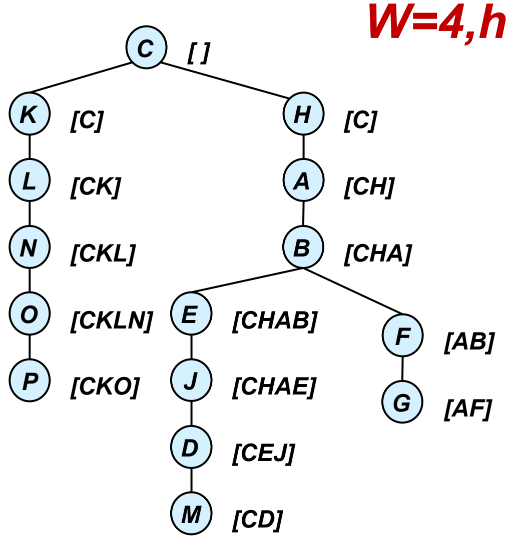
For positive models and given a pseudo tree, AND/OR search and Bucket-elimination have the performance



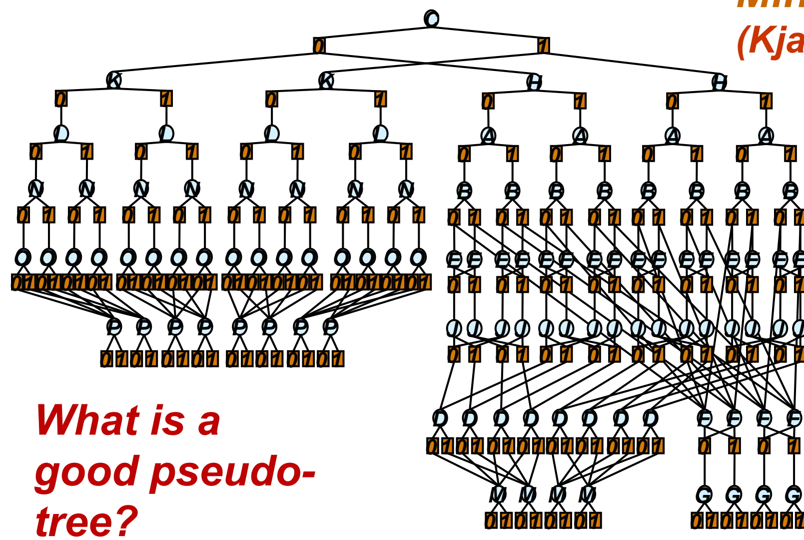
Variable Elimination

The impact of the pseudo-tree

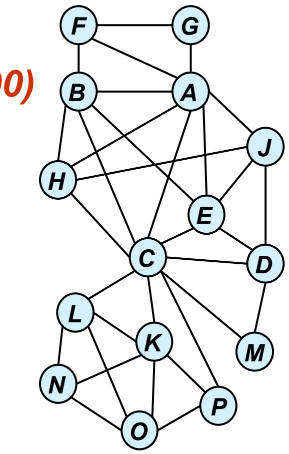
W=4,h=8



(C K H A B E J L N O D P M F G)



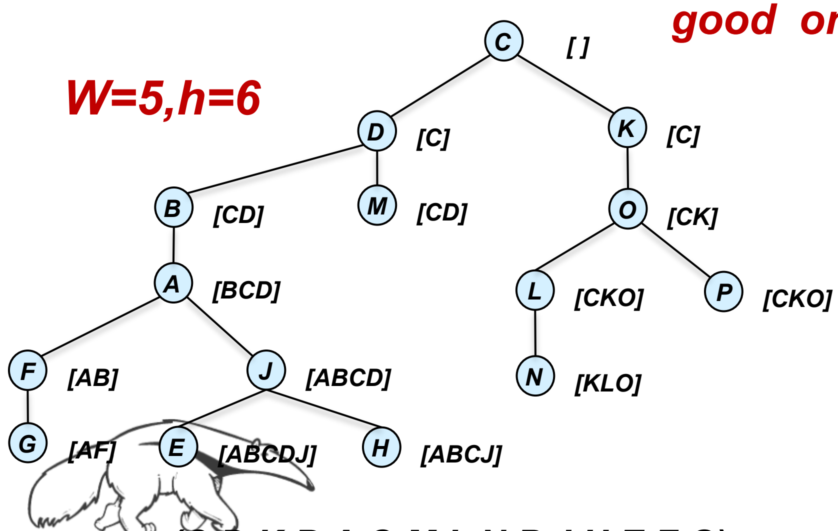
Min-Fill (Kjaerulff90)



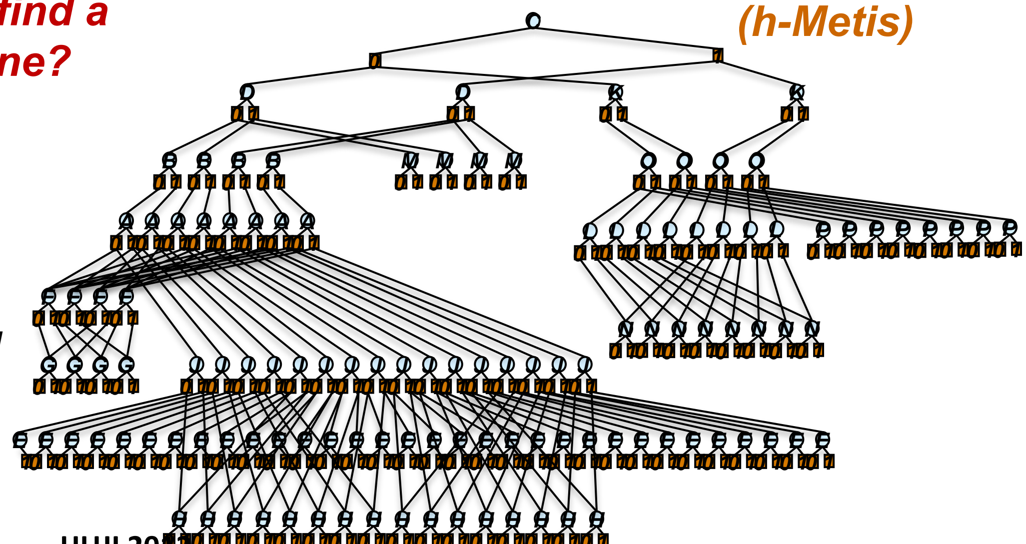
What is a good pseudo-tree?
How to find a good one?

Hypergraph Partitioning (h-Metis)

W=5,h=6



(C D K B A O M L N P J H E F G)



Outline

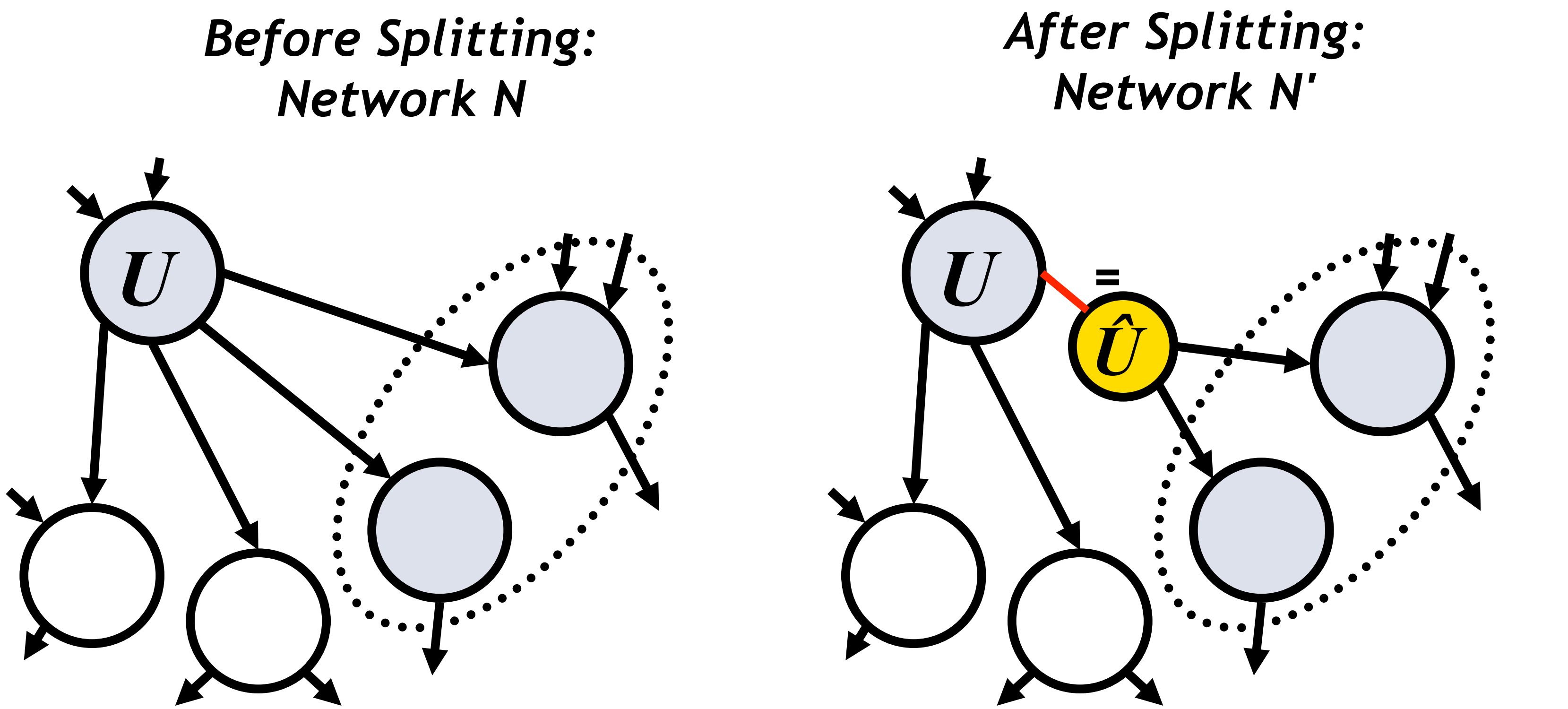
- Graphical models: the primary reasoning principles
- Inference
- AND/OR Search Trees and Graphs
- Lower Bounding schemes
- AND/OR Branch-and-Bound Search
- Experiments

Primary bounding schemes

- Goal: bound $\min_x \sum_i f_i(x)$ or $\max_x \prod_i f_i(x)$
- Two primary relaxation ideas :
- **Node duplication control:**
  - Mini-bucket scheme: (Dechter and Rish 1997,2003, Kask and Dechter, 1999, Rollon and Dechter 2010)
- **Reparameterization schemes:**
  - Soft arc-consistency (Bistareli, 2000, Sciex 2000)
  - Linear relaxation/ Dual-decomposition: (Globerson and Jaakkola 2007, Sontag, Globerson and Jaakkola, 2010, Kovalevsky et al. 1975)

Mini-Bucket: duplicating/Splitting a Node

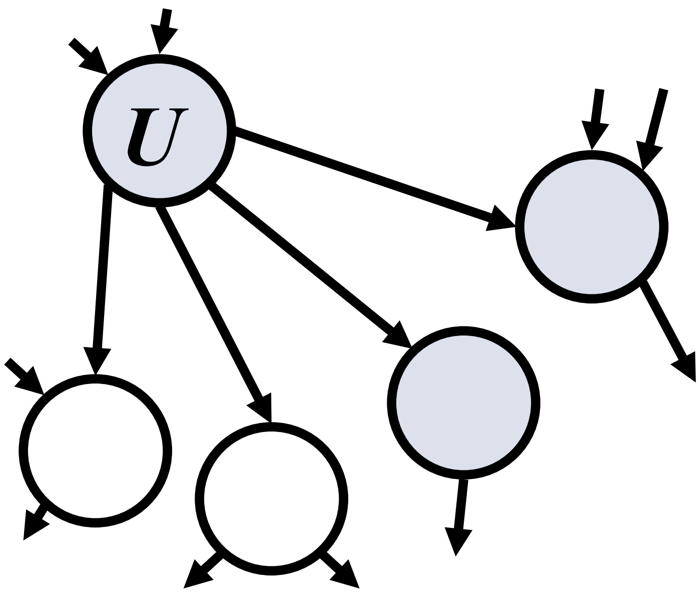
Variables in different buckets are renamed and duplicated
(Kask et. al., 2001), (Geffner et. al., 2007), (Choi, Chavira, Darwiche , 2007)



Semantics of dual decomposition: Splitting each functions + reparameterization

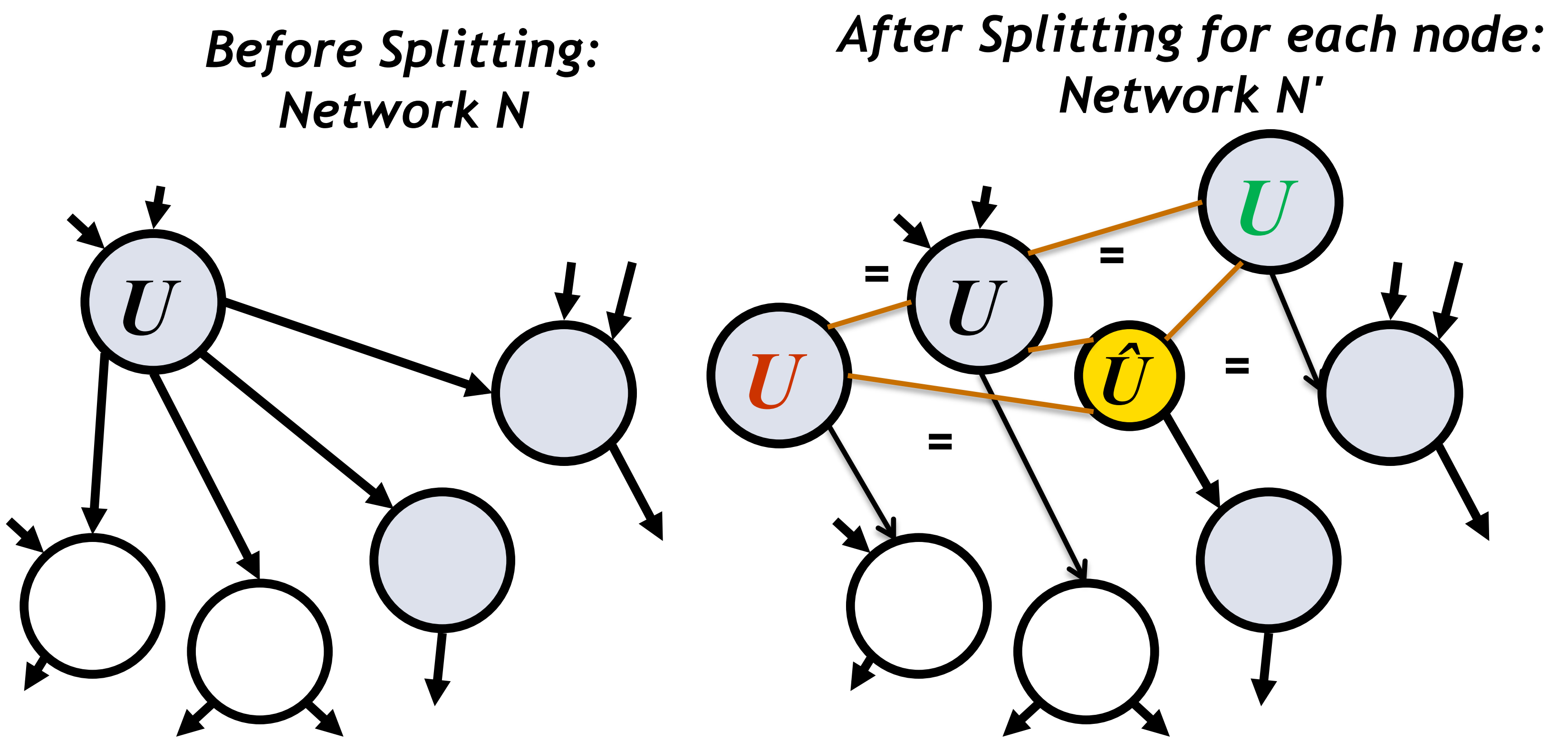
Variables in different buckets are renamed and duplicated (Globerson and Jakkola, 2008),

Before Splitting:
Network N



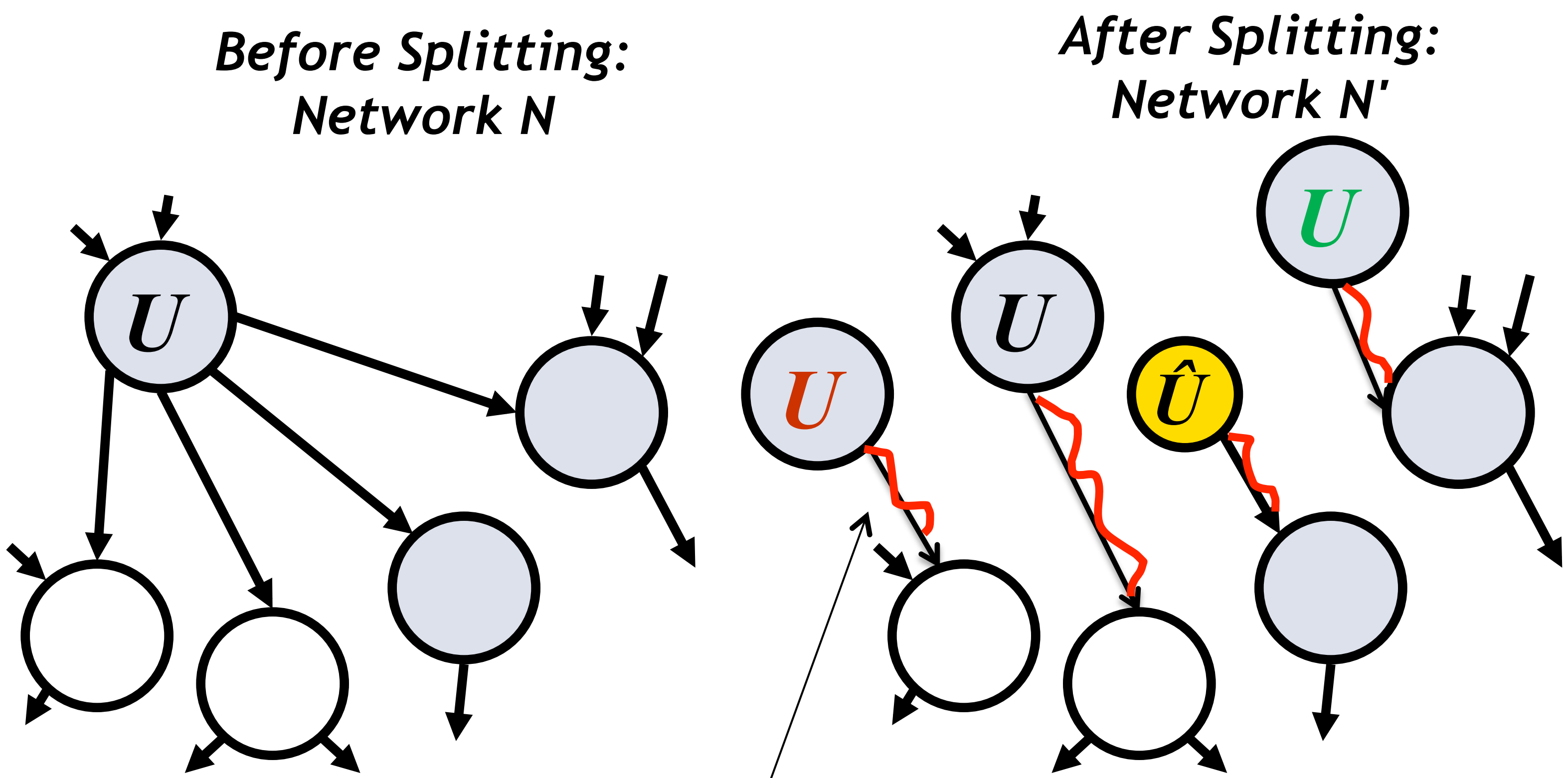
Reparameterization: duplicating a Node for each arc/function

Variables in different buckets are renamed and duplicated (Globerson and Jakkola, 2008),



Reparameterization: duplicating a Node for each arc/function

Variables in different buckets are renamed and duplicated
(Globerson and Jakkola, 2008)



Reparameterize by cost shifting, optimally by linear programming

Deriving upper-lower bounds

- Mini-Bucket Elimination for graphical models
- Cost-shifting schemes
- Hybrids of cost-shifting and mini-bucket
- The impact of these bounds as heuristics for search

Mini-bucket approximation:MPE task

(Dechter and Rish, 1997, 2003)

Split a bucket into mini-buckets =>bound complexity

$$\text{bucket}(X) = \underbrace{\{ h_1, \ldots, h_r, h_{r+1}, \ldots, h_n \}}_{h^X = \max_X \prod_{i=1}^{n} h_i}$$

$$\underbrace{\{ h_1, \ldots, h_r \}} \qquad \underbrace{\{ h_{r+1}, \ldots, h_n \}}$$

$$g^X = \left(\max_X \prod_{i=1}^{r} h_i\right) \cdot \left(\max_X \prod_{i=r+1}^{n} h_i\right)$$

$$\boxed{h^X \leq g^X}$$

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

Mini-Bucket Elimination

$\max_B \prod$ $\max_B \prod$

Node duplication, renaming

Bucket B **P(E|B,C)** **P(B|A) P(D|A,B)**

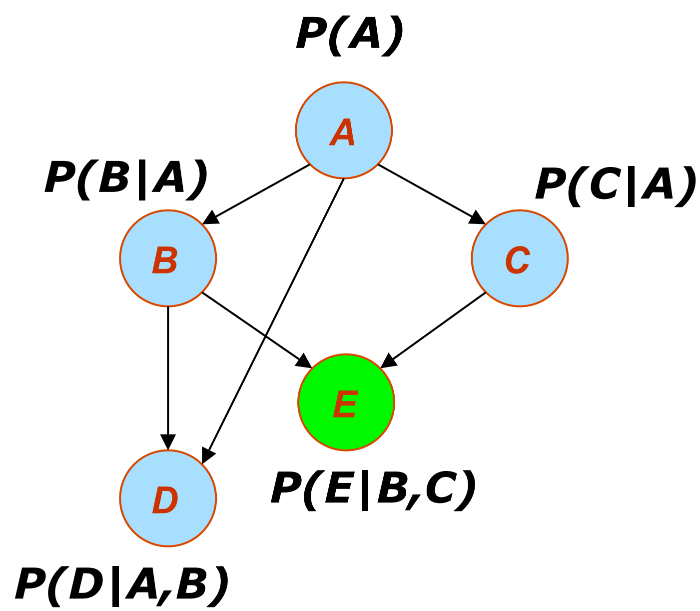
Bucket C **P(C|A)** **h^B (C,E)**

Bucket D **h^B (A,D)**

Bucket E **E = 0** **h^C (A,E)**

Bucket A **P(A)** **h^E (A)** **h^D (A)**

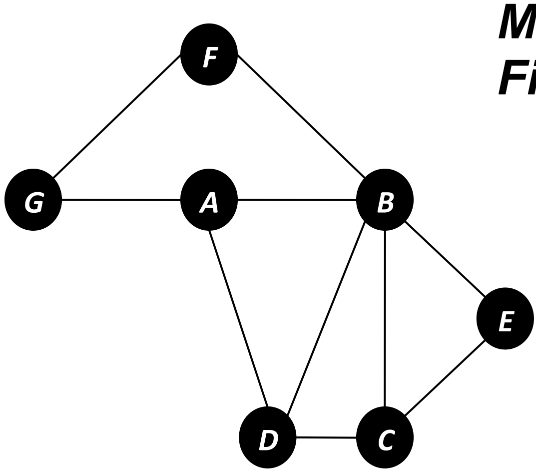
W=2

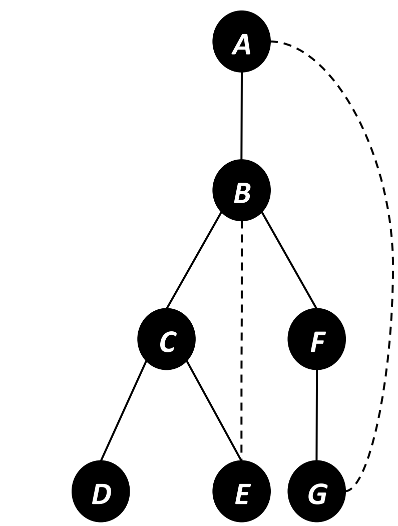


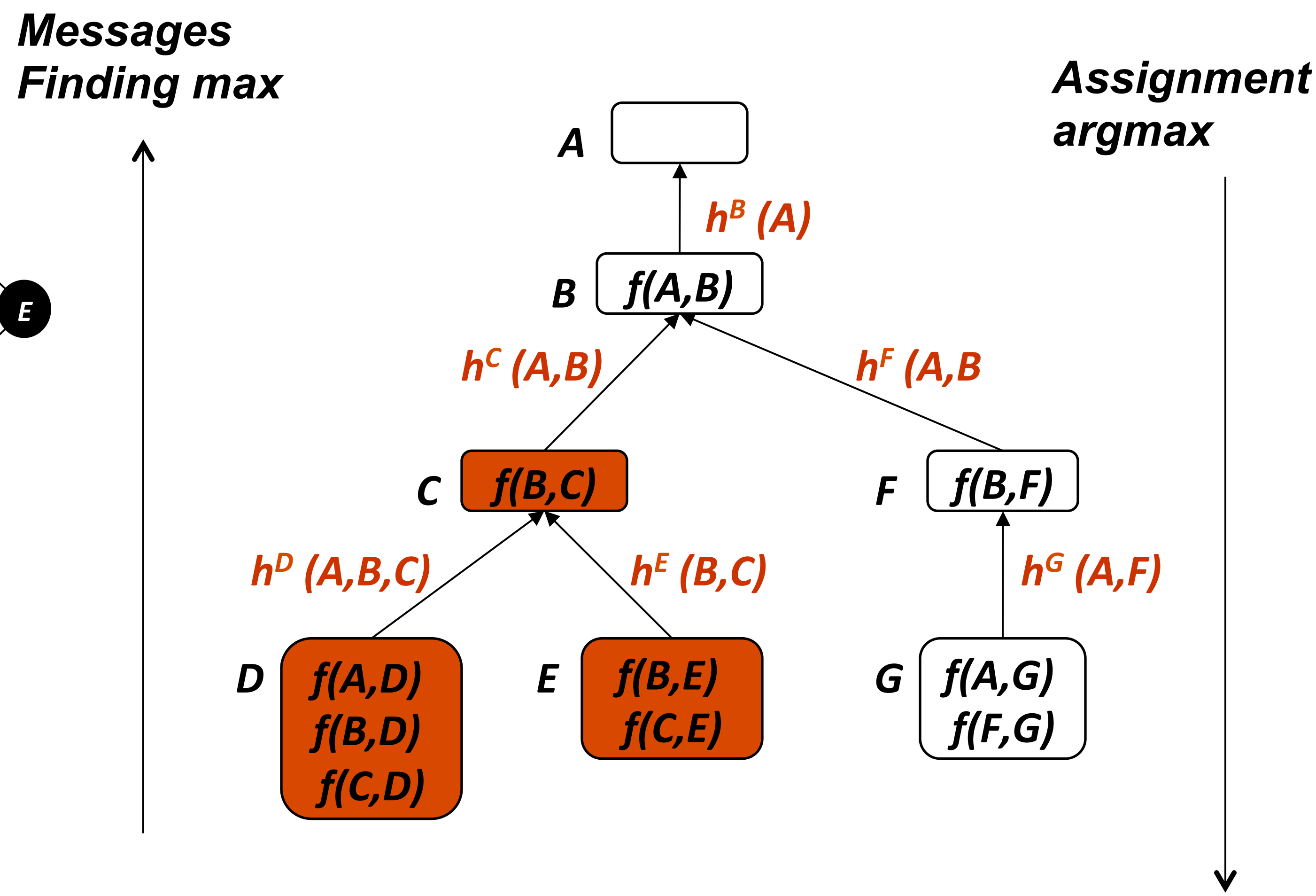
MPE* is an upper bound on MPE --U
Generating a solution yields a lower bound--L

Bucket Elimination

$$\min_{a,b,c,d,e,f,g} f(a,b)+f(a,d)+f(b,c)+f(a,d)+f(b,d)+f(c,d)+f(b,e)+f(c,e)+f(b,f)+f(a,g)+f(f,g)=$$

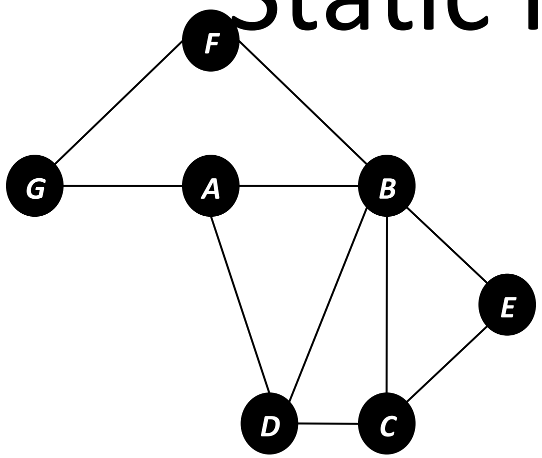


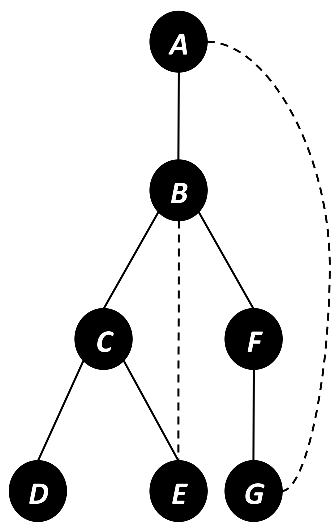


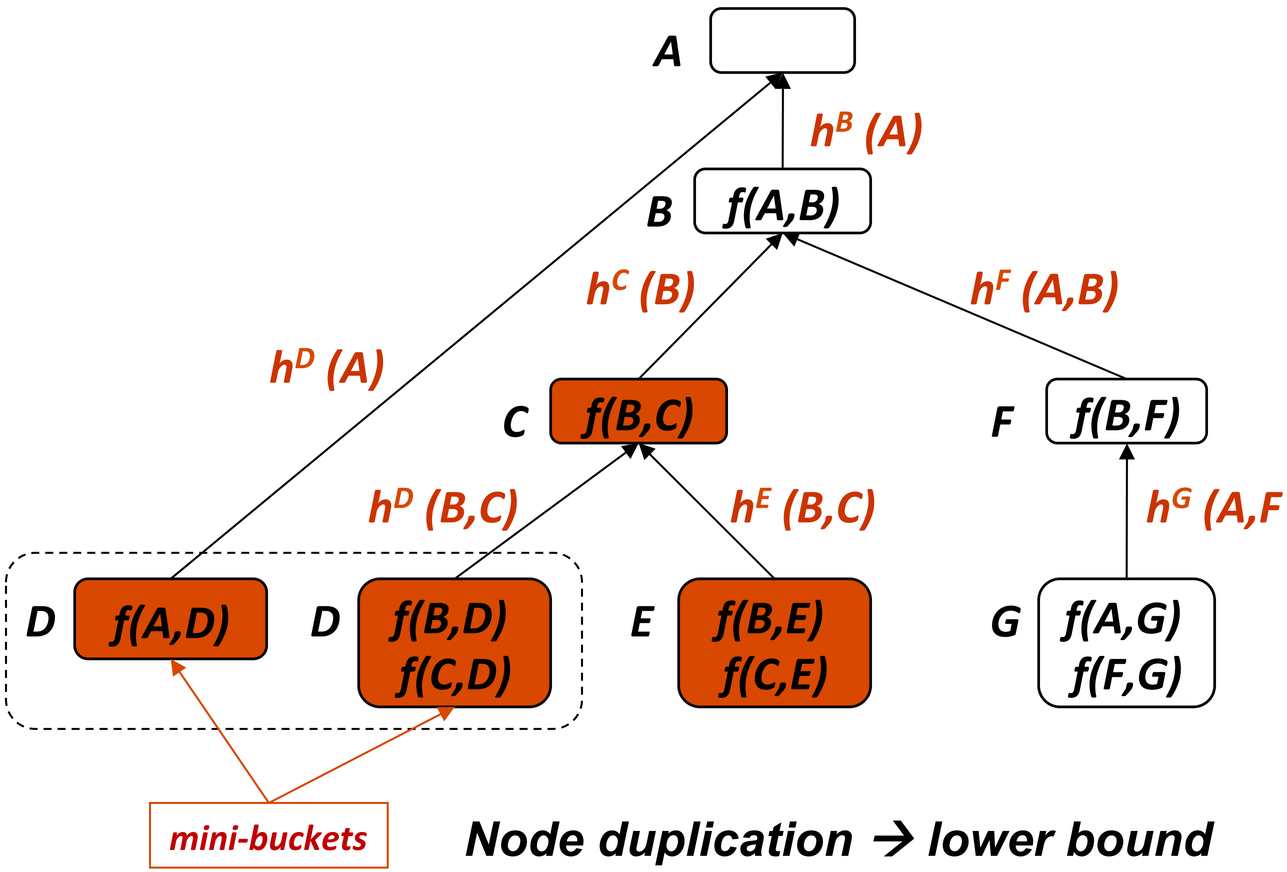


Ordering: (A, B, C, D, E, F, G)

Static Mini-Bucket Heuristics







Node duplication → lower bound

Ordering: (A, B, C, D, E, F, G)

Properties of MBE(i)

- **Complexity:** *O(r exp(i))* time and *O(exp(i)) space*.
- Yields an upper-bound and a lower-bound.

- **Accuracy:** determined by upper/lower (U/L) bound.

- As ***i*** increases, both accuracy and complexity increase.

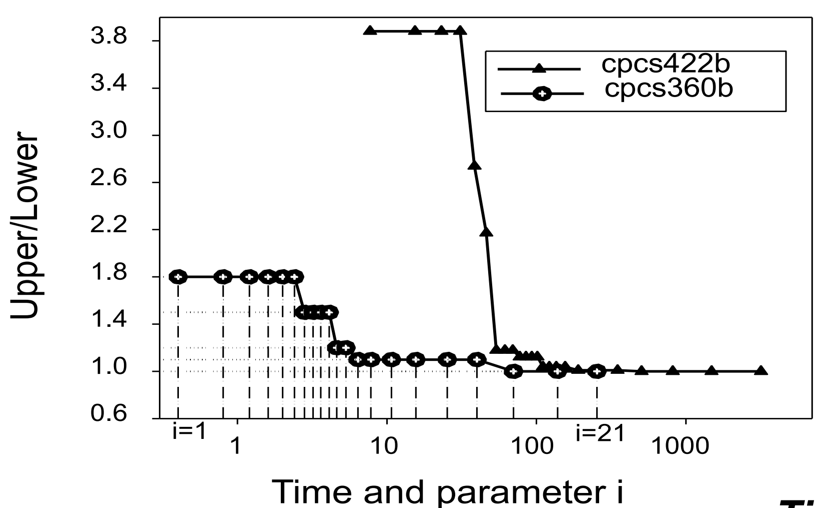
- Possible use of mini-bucket approximations:
  - As anytime algorithms
  - As heuristics in search

- Other tasks: similar mini-bucket approximations for: belief updating, MAP and MEU (Dechter and Rish, 1997)

CPCS networks – medical diagnosis
(noisy-OR CPD's)

Test case: no evidence

Anytime-mpe(0.0001)
U/L error vs time



Algorithm	***Time (sec)*** ***cpcs360***	***cpcs422***
elim-mpe	***115.8***	***1697.6***
anytime-mpe(ε), $\varepsilon = 10^{-4}$	***70.3***	***505.2***
anytime-mpe(ε), $\varepsilon = 10^{-1}$	***70.3***	***110.5***

Mini-Clustering (for sum-product)

Split a cluster into mini-clusters => bound complexity

$$\underbrace{\{h_1,\ldots,h_r,h_{r+1},\ldots,h_n\}}$$

APPROXIMATE algorithm

$$\color{red}{\underbrace{\{h_1,\ldots,h_r\}}} \qquad \color{blue}{\underbrace{\{h_{r+1},\ldots,h_n\}}}$$

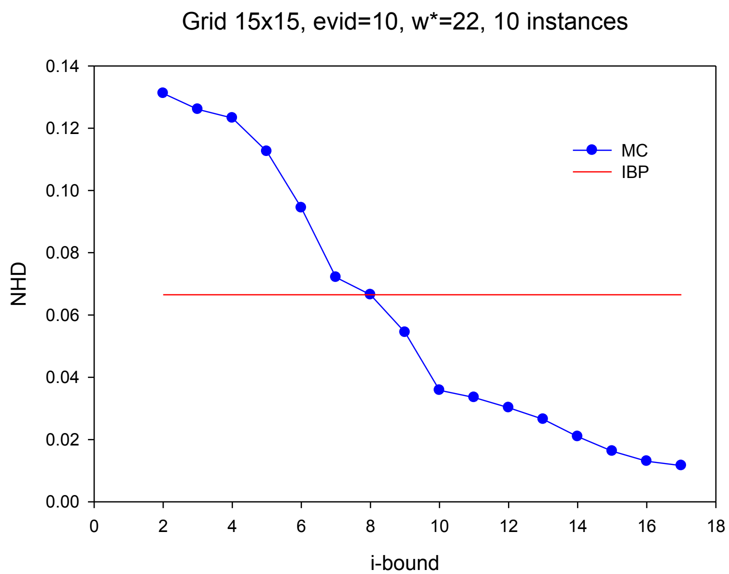
$$\sum_{elim}\prod_{i=1}^{n} h_i \quad \leq \quad \left(\sum_{elim}\prod_{i=1}^{r} h_i\right)\cdot\left(\sum_{elim}\prod_{i=r+1}^{n} h_i\right)$$

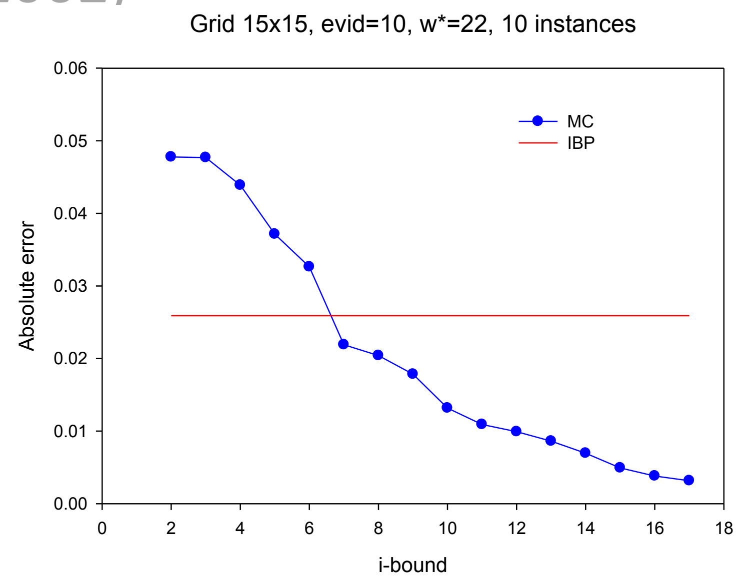
Exponential complexity decrease

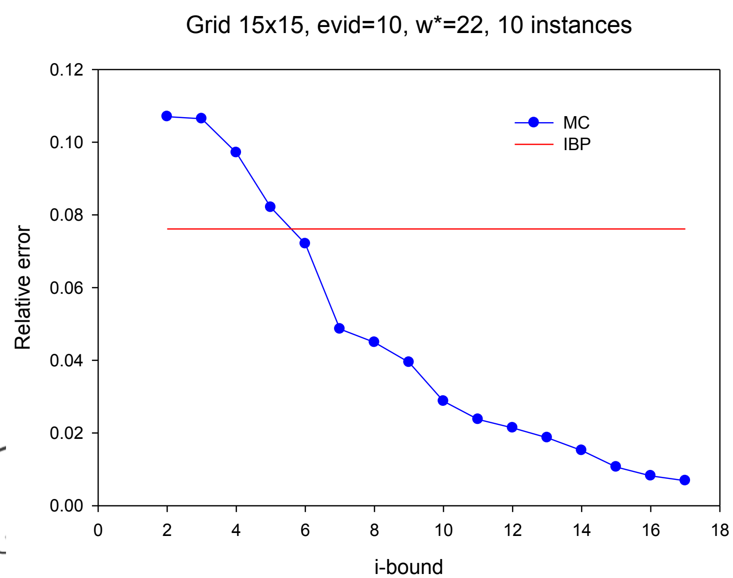
$$O(e^n) \rightarrow O(e^{\mathrm{var}(r)}) + O(e^{\mathrm{var}(n-r)})$$

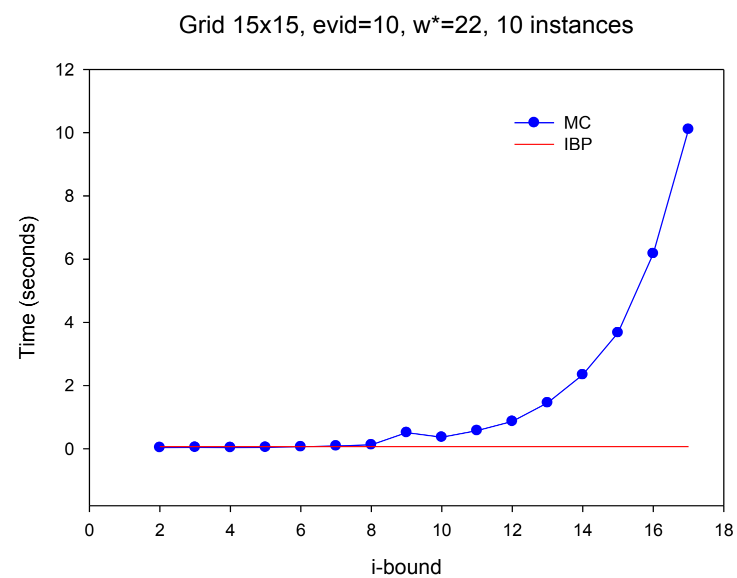
Grid 15x15 - 10 evidence

(Mateescu, Kask and Dechter, 2002)









Finding optimal i-partition;

(Rollon and Dechter 2010)

- Given an i-bound, and a distance d() , choose partition Q* s.t.

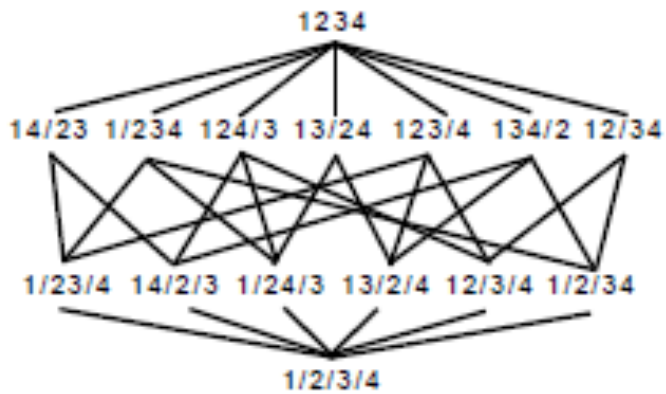
$$Q^* = \arg\min_Q \left\{ d(g^Q, g) \right\} \quad \text{where} \quad Q \quad \text{is an i-partition}$$

- We considered as distance measure d():
  - Log relative error (RE)
  - Maximum log relative error (MRE)
  - KL divergence (KL)
  - Absolute error (AE)

Optimizing the partitioning

(Rollon and Dechter 2010)

Scope-based Partitioning scheme (SCP) minimizes the number of mini-buckets as respecting the i bound i (Rish, Kask 2000)



Partitioning lattice of bucket $\{f_1, f_2, f_3, f_4\}$.

- *Log relative error*:

$$RE(f,h) = \sum_t (\log(f(t)) - \log(h(t)))$$

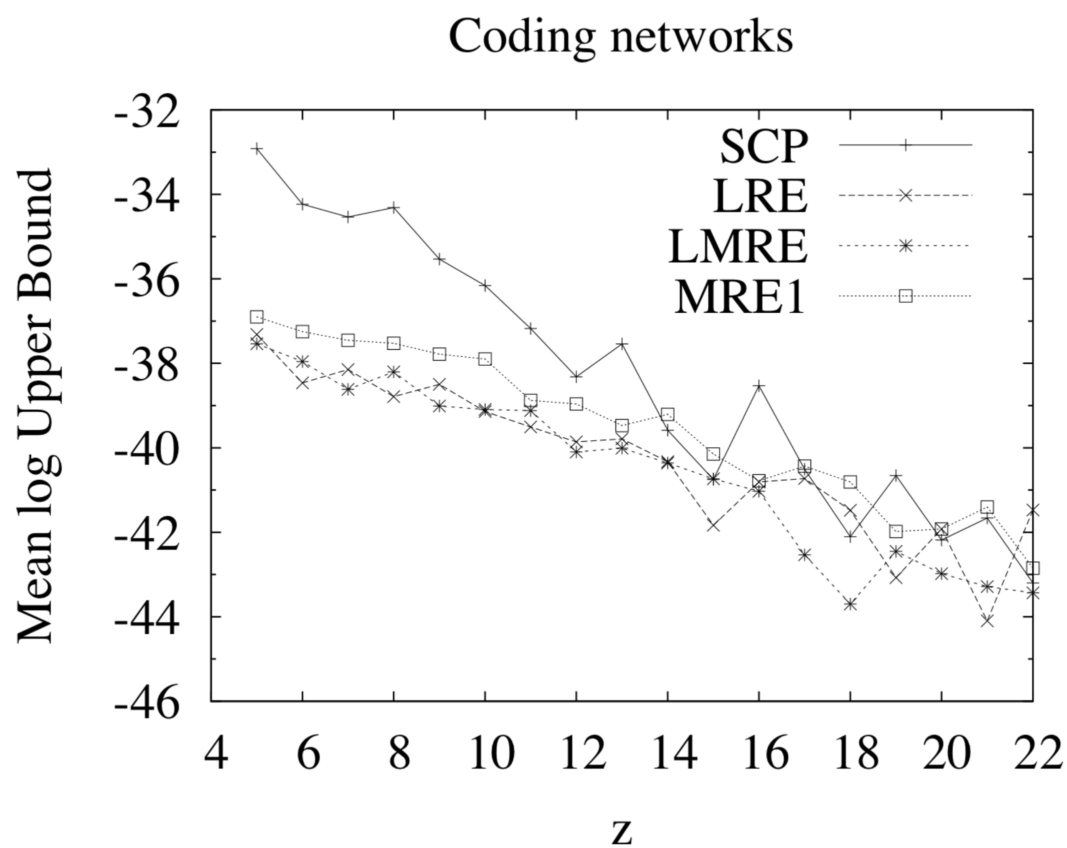
- *Max log relative error*:

$$MRE(f,h) = \max_t \{\log(f(t)) - \log(h(t))\}$$

Use greedy heuristic derived from a distance function to decide which functions go into a single mini-bucket

Experiments: Coding networks

- ***Performance of the partition heuristics as a function of the i-bound.***



Reparameterization Schemes:

How to find best cost-shifting bound

$$MAP = \min_x \sum_{f_i \in F} f_i(x,y)$$

Task: given a fixed partition Q=$\{\{f_1\},\ldots,\{f\}_r\}$ find f's that minimize distance over classes of cost-shifting.

$$distance_{f'\ cost-shifts}\{\mathrm{Map}, \sum_{f'_i \in F} \min f'_i(x,y)\}$$

Soft arc-consistency uses heuristic idea.

Optimal schemes: Optimal soft arc-consistency (OSAC), and dual-decompositions (MPLP).

Reparameterization using Linear relaxation-based schemes (MPLP class) Globerson and Jaakkola, Nips 2008)

Find:

$\boldsymbol{x} = (x_1, \ldots, x_n)$ to all the variables which maximizes the sum of the factors:

$$\mathrm{MAP}(\boldsymbol{\theta}) = \max_{\boldsymbol{x}} \sum_{i \in V} \theta_i(x_i) + \sum_{f \in F} \theta_f(\boldsymbol{x}_f). \tag{1.1}$$

Best upper bound by Equivalence preserving transformations:

$$\min_{\boldsymbol{\delta}} L(\boldsymbol{\delta}), \tag{1.2}$$

$$L(\boldsymbol{\delta}) = \sum_{i \in V} \max_{x_i} \Big(\theta_i(x_i) + \sum_{f : i \in f} \delta_{fi}(x_i) \Big) + \sum_{f \in F} \max_{\boldsymbol{x}_f} \Big(\theta_f(\boldsymbol{x}_f) - \sum_{i \in f} \delta_{fi}(x_i) \Big).$$

$\delta_{fi}(x_i)$ ***Is the cost shifted from f to value x_i of X_i.***

There are several variations of scheme computing the optimizing shifts based on partial gradient descent, which differ by what is being kept constant.
The 1.2 task is the Dual of a linear relaxation of the original problem.

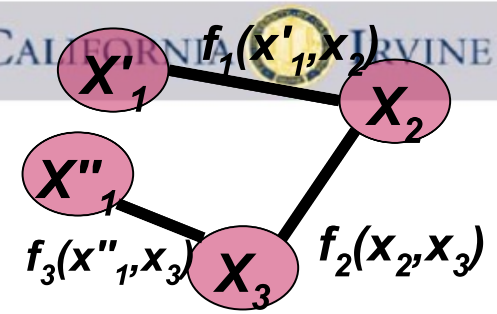
Mini-Bucket with moment-matching

(Ihler, Flerova, Dechter, Otten, 2011)

- ***MBE: non-iterative message-passing schemes***
- ***iterative schemes using re-parametrization***
- ***MPLP*** *[Globerson , Jakkola, Sontag et al. 2008],*
- ***Max-sum diffusion*** *[Kovalevsky et al. 1975]*
- ***Soft arc-consistency*** *[Schiex 2000, Bistarelli et al. 2000]*

Mini-Bucket with moment-matching

(Ihler, Flerova, Dechter, Otten, 2011)



$$C_1(x_2,x_3) = \max_{x_1} f_1(x_1,x_2)\cdot f_3(x_1,x_3)$$

$$C_1(x_2,x_3) = \max_{x_1} f_1(x_1,x_2)\boxed{g(x_1)}\cdot f_3(x_1,x_3)\boxed{/g(x_1)}$$

$$\boxed{C(x_2,x_3) \le \hat{C}_1(x_2,x_3) = \max_{x_1'} f_1(x_1',x_2)g(x_1') \cdot \max_{x_1''} f_3(x_1'',x_3)/g(x_1'')}$$

$$\max_{x_2} f_1(x_1,x_2)g(x_1) = \max_{x_3} f_3(x_1,x_3)/g(x_1)$$

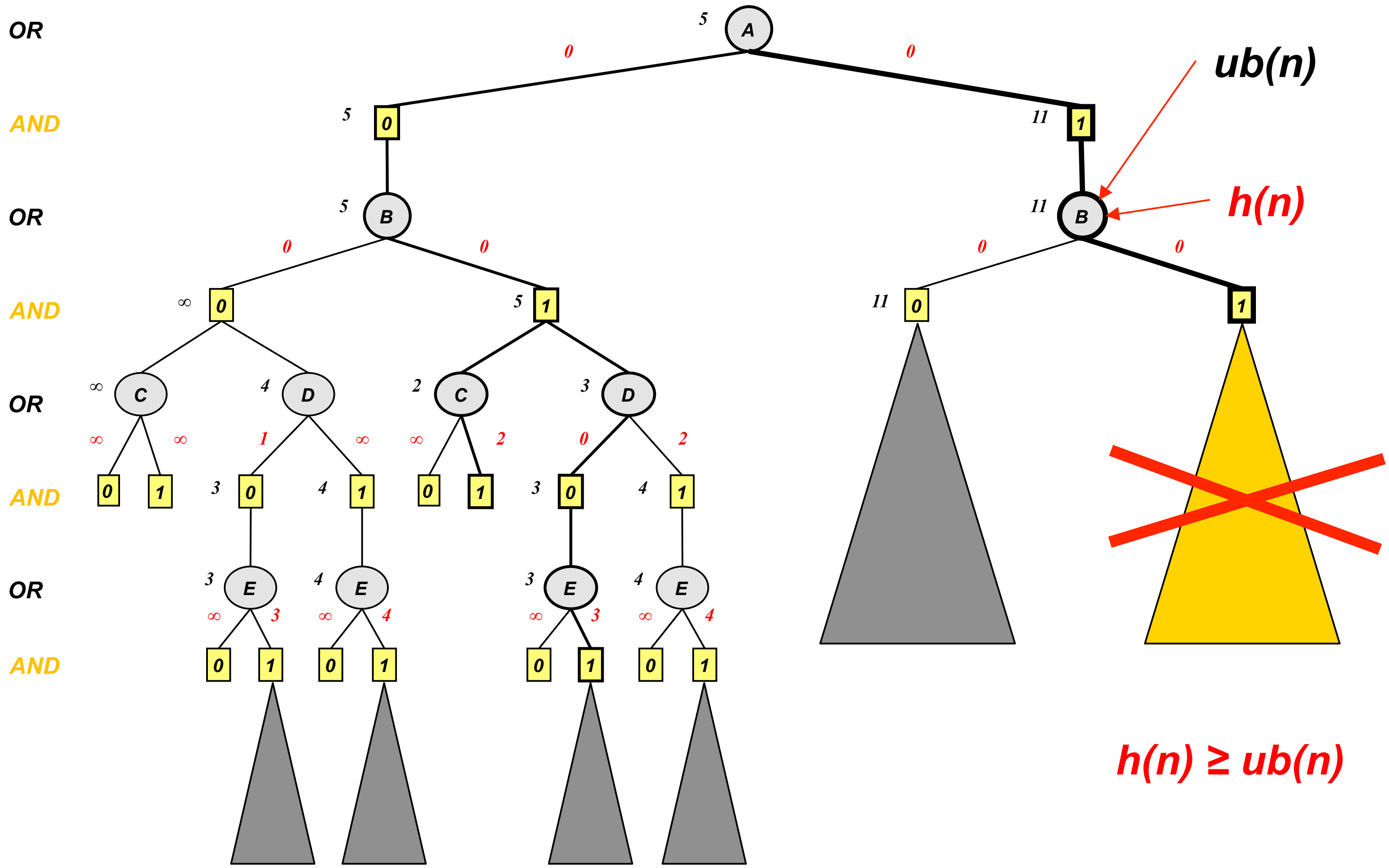
$$g(x_1) = \sqrt{\frac{\max_{x_3} f_3(x_1,x_3)}{\max_{x_2} f_1(x_1,x_2)}}$$

$$\boxed{\hat{C}_1(x_2,x_3) = \max_{x_1} f_1(x_1,x_2)\sqrt{\frac{\max_{x_3} f_3(x_1,x_3)}{\max_{x_2} f_1(x_1,x_2)}} \cdot \max_{x_1} f_3(x_1,x_3)\sqrt{\frac{\max_{x_2} f_1(x_1,x_2)}{\max_{x_3} f_3(x_1,x_3)}}}$$

Outline

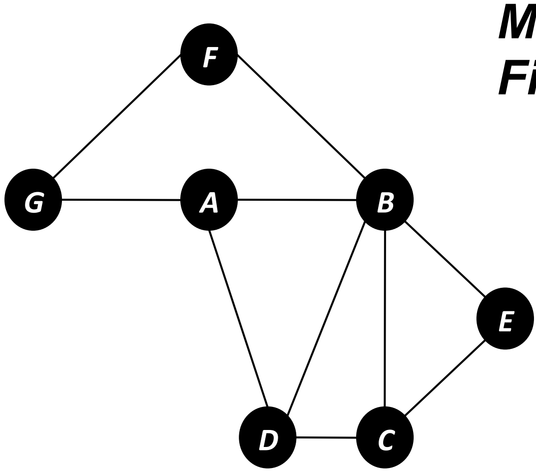
- Graphical models: reasoning principles
- Inference
- Search; via AND/OR Search
- Lower Bounding schemes for inference
- Lower-bounding heuristics for AND/OR search
- Experiments

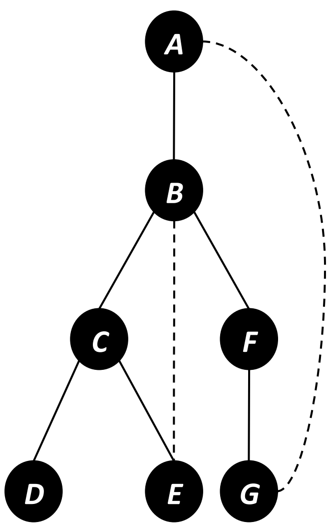
AND/OR Branch-and-Bound



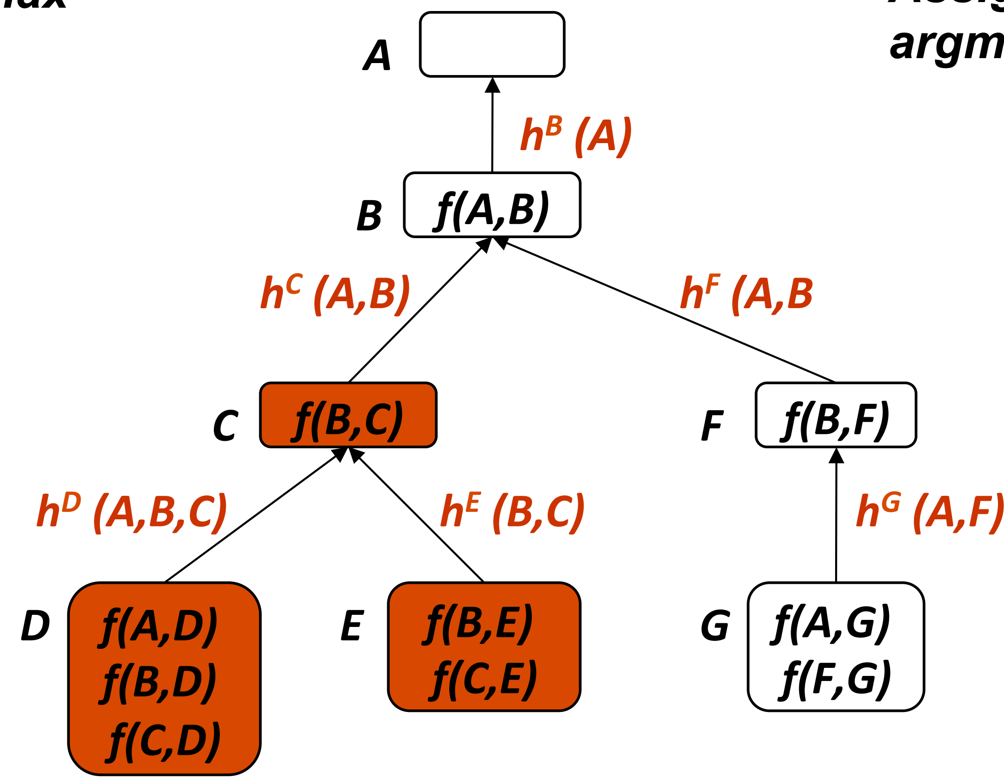
Bucket Elimination

$$\min_{a,b,c,d,e,f,g} f(a,b) + f(a,d) + a(b,c) + f(a,d) + f(b,d) + f(c,d) + f(b,e) + f(c,e) + f(b,f) + f(a,g) + f(f,g) =$$





Messages Finding max

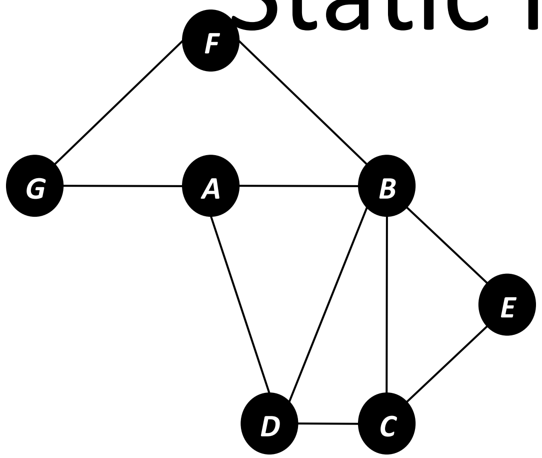


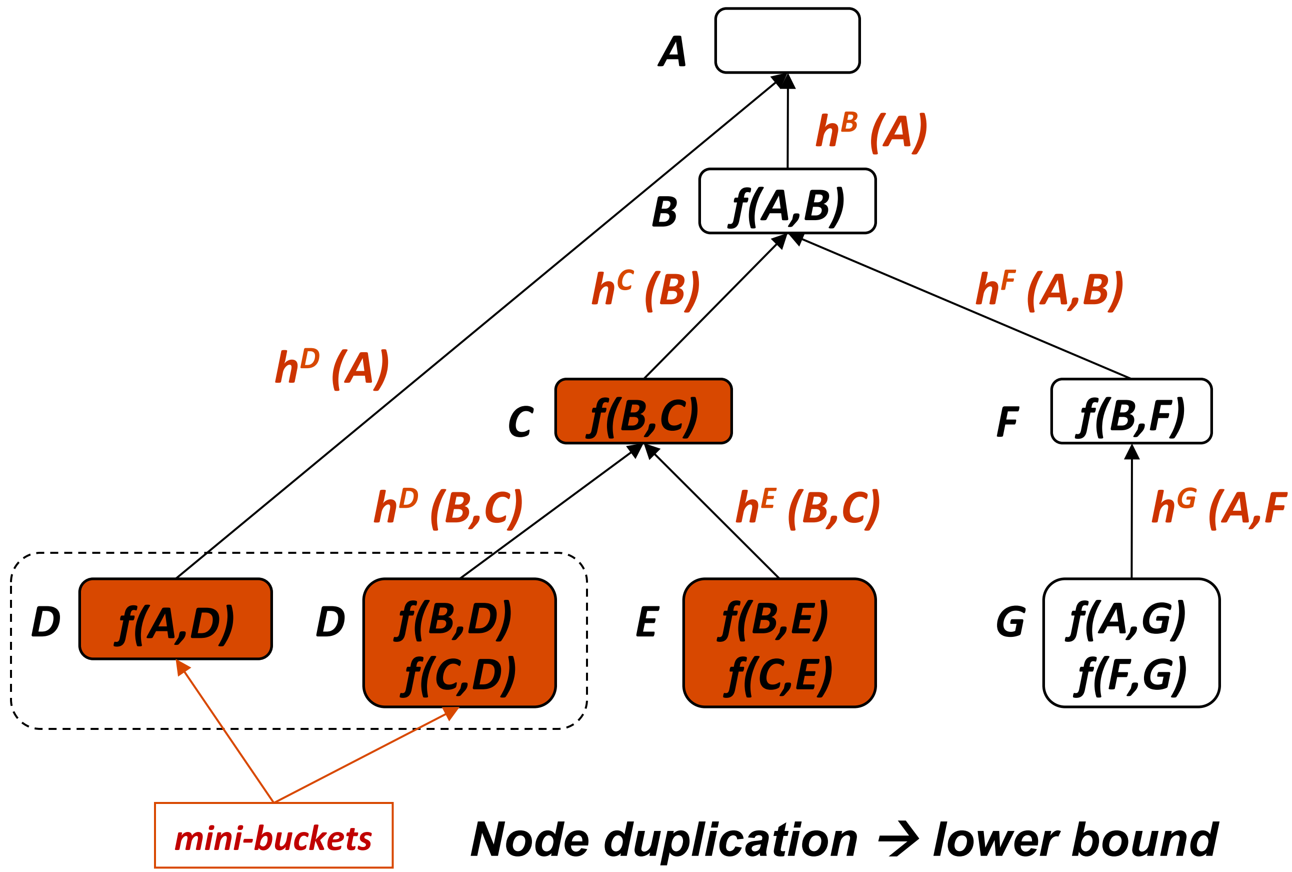
Assignment argmax

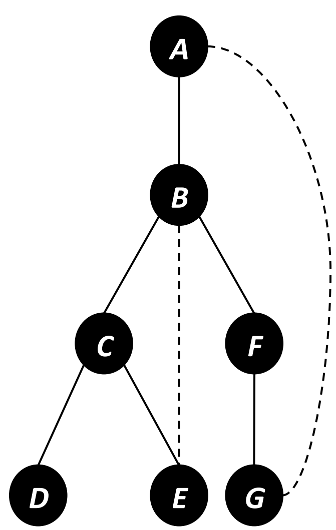
$$h^*(a, b, c) = h^D(a, b, c) + h^E(b, c)$$

Ordering: (A, B, C, D, E, F, G)

Static Mini-Bucket Heuristics







Node duplication → lower bound

$$h(a, b, c) = h^D(a) + h^D(b, c) + h^E(b, c)$$
$$\leq h^*(a, b, c)$$

Ordering: (A, B, C, D, E, F, G)

Outline

- Graphical models: the primary reasoning principles
- Inference
- AND/OR Search Trees and Graphs
- Lower Bounding heuristics for search
- AND/OR Branch-and-Bound Search
- Experiments

Grid Networks (BN)

(Sang et al.05)

grid (w*, h) (n, e)	Samlam	MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=12 time	nodes	MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=14 time	nodes	MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=16 time	nodes	MBE(i) BB-C+SMB(i) AOBB+SMB(i) AOBB-C+SMB(i) AOBF-C+SMB(i) i=18 time	nodes
90-24-1 (33, 111) (576, 20)	out	0.28 - - - out	 - - -	0.64 - 2338.67 1273.09 21.94	 - 24,117,151 9,047,518 75,637	1.69 - 1548.09 596.27 10.59	 - 18,238,983 4,923,760 33,770	4.60 - 138.67 70.42 **6.06**	 - 1,413,764 473,675 5,144
90-34-1 (45, 153) (1154, 80)	out	0.63 - - - out	 - - -	1.25 - - - out	 - - -	3.72 - - - **243.63**	 - - - 596,978	11.66 - - - 270.88	 - - - 667,013
90-38-1 (47, 163) (1444, 120)	out	0.78 - 2032.33 969.02 101.69	 - 6,835,745 2,623,971 174,786	1.67 - - 1753.10 103.80	 - - 3,794,053 146,237	4.20 - 807.38 203.67 54.00	 - 2,850,393 614,868 95,511	12.36 - 568.69 165.45 **53.44**	 - 2,079,146 488,873 78,431

Min-fill pseudo tree. Time limit 1 hour.

Genetic Linkage Analysis

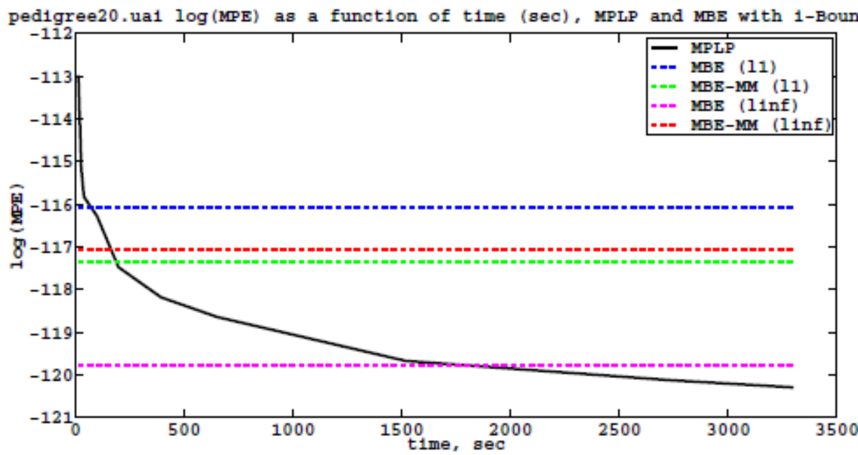
(Fishelson & Geiger02)

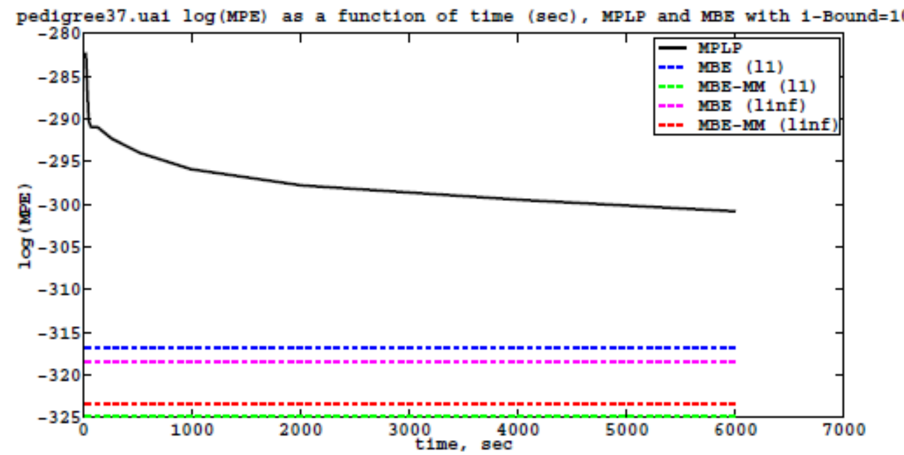
pedigree (w*, h) (n, d)	**Samlam** **Superlink**	**MBE(i)** **BB-C+SMB(i)** **AOBB+SMB(i)** **AOBB-C+SMB(i)** **AOBF-C+SMB(i)** i=12		**MBE(i)** **BB-C+SMB(i)** **AOBB+SMB(i)** **AOBB-C+SMB(i)** **AOBF-C+SMB(i)** i=14		**MBE(i)** **BB-C+SMB(i)** **AOBB+SMB(i)** **AOBB-C+SMB(i)** **AOBF-C+SMB(i)** i=16		**MBE(i)** **BB-C+SMB(i)** **AOBB+SMB(i)** **AOBB-C+SMB(i)** **AOBF-C+SMB(i)** i=18	
		time	nodes	time	nodes	time	nodes	time	nodes
ped30 (23, 118) (1016, 5)		0.42		0.83		1.78		5.75	
		-	-	-	-	-	-	-	-
	out	-	-	-	-	-	-	214.10	1,379,131
	13095.83	10212.70	93,233,570	8858.22	82,552,957	-	-	34.19	193,436
		out		out		out		**30.39**	72,798
ped33 (37, 165) (581, 5)		0.58		2.31		7.84		33.44	
		-	-	-	-	-	-	-	-
	out	2804.61	34,229,495	737.96	9,114,411	3896.98	50,072,988	159.50	1,647,488
	-	1426.99	11,349,475	307.39	2,504,020	1823.43	14,925,943	86.17	453,987
		out		140.61	407,387	out		**74.86**	134,068
ped42 (25, 76) (448, 5)		4.20		31.33		96.28		out	
		-	-	-	-	-	-		
	out	-	-	-	-	-	-		
	561.31	-	-	-	-	2364.67	22,595,247		
		out		out		**133.19**	93,831		

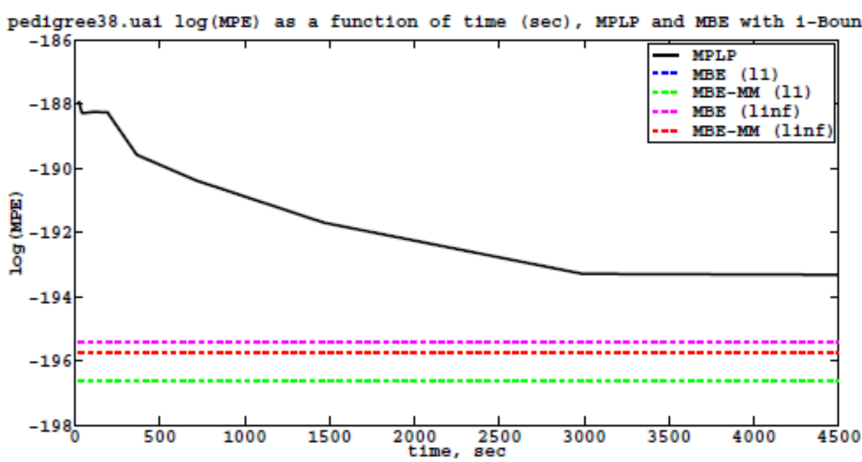
Min-fill pseudo tree. Time limit 3 hours.

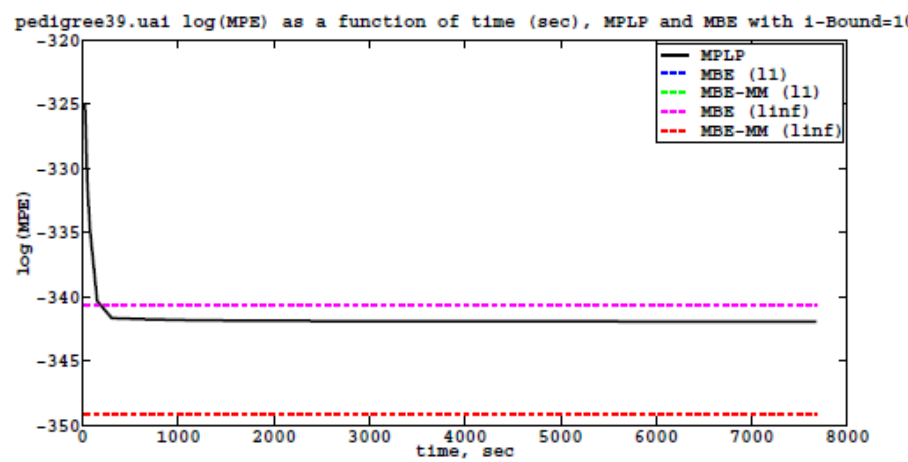
MBE-MM vs. MPLP (pedigrees)

MBE, MBE-MM z-bound = 10, time cutoff = 3600 sec
MPLP on original factors, cutoff = 1500 iterations









Runtime (sec) (pedigrees) by AOBB with MBE, MBE-MM or MPLP as a heuristic generator.

Time cutoff 24 h
Memory limit 2G

Instances (n,k,w,h)	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=4	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=6	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=8	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=10
pedigree1 298,4,15,48	0 0 42	0 0 11	1 0 7	0 1 4
pedigree7 867,4,32,90	— — —	— 46261 54000	— 5993 13211	— 1987 4975
pedigree9 935,7,27,100	— — —	— — —	— 7086 9397	46434 1206 2161
pedigree20 387,5,22,60	4460 10805 11203	167 378 582	137 112 262	44 25 87
pedigree23 309,5,25,51	45 31 89	22 3 201	13 2 24	4 0 9
pedigree25 993,5,25,69	— 13321 4670	1303 36 48	145 4 3	58 0 1
pedigree30 1015,5,21,108	13198 — —	1690 442 508	246 36 99	109 21 34
pedigree33 581,4,28,98	24142 1958 2076	201 260 287	177 7 8	89 5 8
pedigree37 726,5,21,56	298 174 145	33 8 6	13 0 1	6 1 0
pedigree39 953,5,21,76	30724 8315 9100	2871 294 377	732 29 26	136 15 17
pedigree50 478,6,17,47	25440 — —	39 47 12146	16 11 886	6 17 46

Runtime (sec) expanded (grids) by AOBB with MBE, MBE-MM or MPLP as a heuristic generator.

Time cutoff 24 h, memory limi 2 Gb

Instances (n,k,w,h)	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=3	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=5	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=10	AOBB-MBE(z) AOBB-MBE-MM(z) AOBB-MPLP(z) z-bound=15
50-16-5 256,2,21,79	14047 11918 7243	6759 257 209	97 1 1	3 0 1
50-20-5 400,2,27,97	— — —	— 28985 6529	3589 11 26	385 1 6
75-16-5 256,2,21,73	2457 511 516	245 32 37	7 0 1	0 0 0
75-20-5 400,2,27,99	— — 47557 1	— 25258 8458	1912 6 13	7 1 1
90-20-5 400,2,27,99	7281 5575 3162	1199 585 389	14 1 0	0 0 0
90-21-5 441,2,28,106	7722 9064 4709	1585 861 593	15 0 1	1 0 0
90-22-5 484,2,30,109	27283 17130 10279	2327 1172 604	50 6 1	3 0 1
90-26-5 676,2,36,136	— 70798 58797	36469 7077 4000	386 21 16	52 2 2

UAI 2010 evaluation, 2008, 2006

We are first in Pascal 2012, so far… Please join

- Toulbar2: INRA
  Summary: Toulbar2 is an open source exact anytime Weighted CSP solver using Branch and Bound and soft local consistency
  Team members: S. de Givry, D. Allouche, A. Favier, T. Schiex
  Additional contributors: M. Sanchez, S. Bouveret, H. Fargier, F. Heras, P. Jegou, J. Larrosa, K. L. Leung, S. N'diaye, E. Rollon, C. Terrioux, G. Verfaillie, M. Zytnicki
  Contact person: Thomas Schiex, Thomas.Schiex@toulouse.inra.fr
  Detailed description
- Daoopt: UCI Irvine
  Summary: "daoopt" and "daoopt.anytime" are based on AND/OR branch and bound graph search, with mini bucket heuristics and LDS (Limited Discrepancy Search) initialization.
  Team members: Lars Otten, Rina Dechter
  Additional Contributor: Radu Marinescu
  Contact person: Lars Otten, lotten@ics.uci.edu
  Detailed description
  Web-site: http://graphmod.ics.uci.edu

3rd in all 3 categories After Toolbar, Joris

Software

- AND/OR search algorithms
- Bucket-tree elimination
- Generalized belief propagation
- Samplesearch sampling

are available at:

- http://graphmod.ics.uci.edu/group/Software

Kalev Kask
Bozhena Bidyuk

Irina Rish

Thank you!

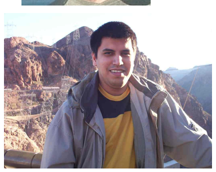
Radu Marinescu

For publication see:

http://www.ics.uci.edu/~dechter/publications.html

Robert Mateescu

We are first in Pascal challenge 2012 (Globerson, Elidan), so far…Please join



Vibhav Gogate

Lars Otten HUJI 2012

Emma Rollon
Natalia Flerova