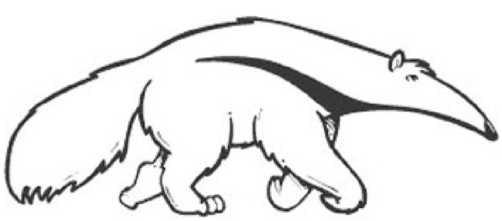
Advances in Combinatorial Optimization for Graphical Models

ICAPS Tutorial

Rina Dechter

University of California, Irvine

Radu Marinescu
Robert Mateescu
Lars Otten
Alex Ihler



BREN:ICS
INFORMATION AND COMPUTER SCIENCES

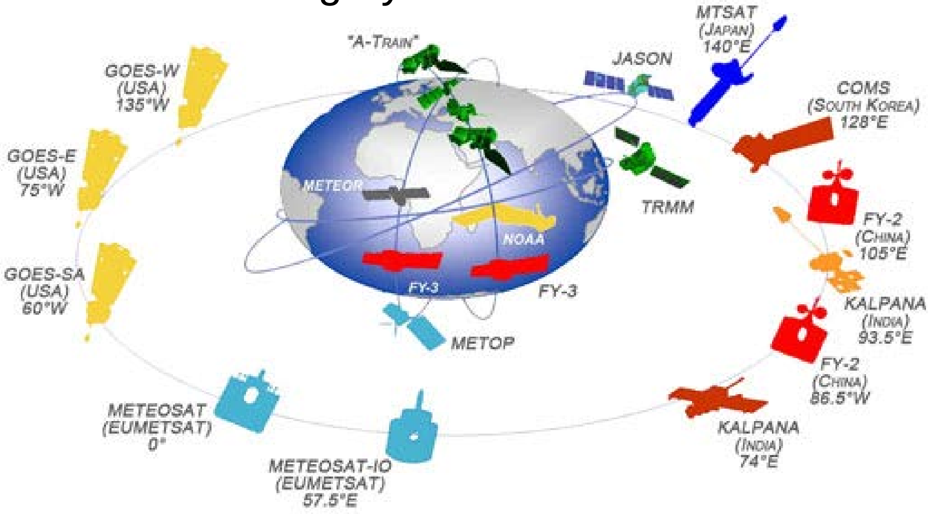
UNIVERSITY of CALIFORNIA IRVINE

Outline

- **Introduction**
  - Graphical models
  - Optimization tasks for graphical models
  - Birds-view of techniques
- **Inference**
  - Variable Elimination, Bucket Elimination
- **Search**
  - AND/OR search spaces
  - Depth-First Branch-and-Bound, Best-First Search
- **Lower-bounds and relaxations**
  - Bounded variable elimination
  - Iterative cost shifting and local consistency
- **Advanced tasks for optimization**
  - Marginal Map for Conformant planning
  - Influence diagrams
- **Software**

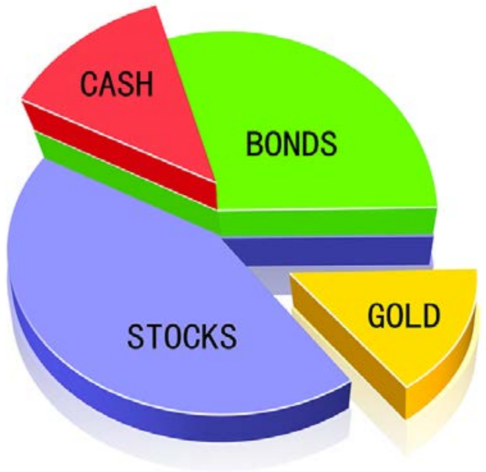
Combinatorial Optimization

Satellite imagery



Find an optimal schedule for the satellite that maximizes the number of photographs taken, subject to on-board recording capacity

Investments



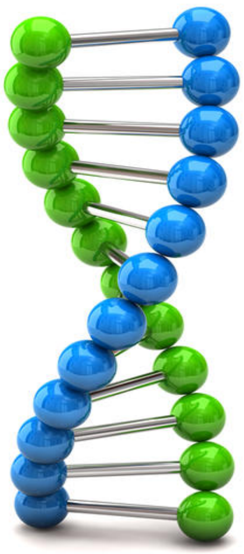
How much to invest in each asset to earn 8 cents per Invested dollar and the investment risk is minimized

Combinatorial Optimization

Communications

Assign frequencies to a set of radio links
such that interferencies are minimized

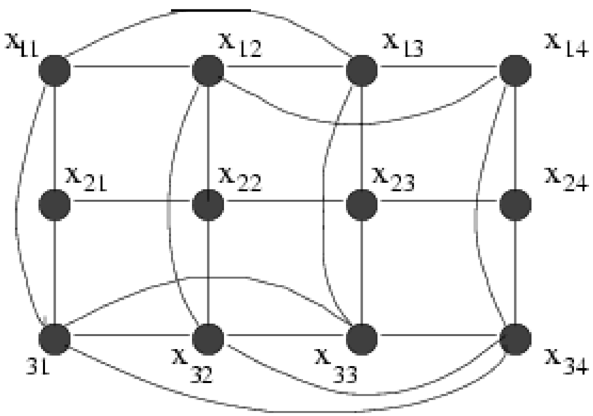
Bioinformatics



Find a joint haplotype configuration for
all members of the pedigree which maximizes the probability of data

Constrained Optimization

Example: power plant scheduling



Unit #	Min Up Time	Min Down Time
1	3	2
2	2	1
3	4	1

$$\text{Variables} = \{X_1,\ldots,X_n\},\ \text{domain} = \{ON, OFF\}.$$
$$\text{Constraints}: X_1 \vee X_2, \neg X_3 \vee X_4, \text{min-up and min-down time},$$
$$\text{power demand}: \sum \text{Power}(X_i) \geq Demand$$

$$Objective: \text{ minimize TotalFuelCost}(X_1,\ldots,X_N)$$

Graphical Models, Queries, Algorithms

Constraint Optimization Problems for Graphical Models

A *finite* *COP* is a triple $R = \langle X, D, F \rangle$ where:

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

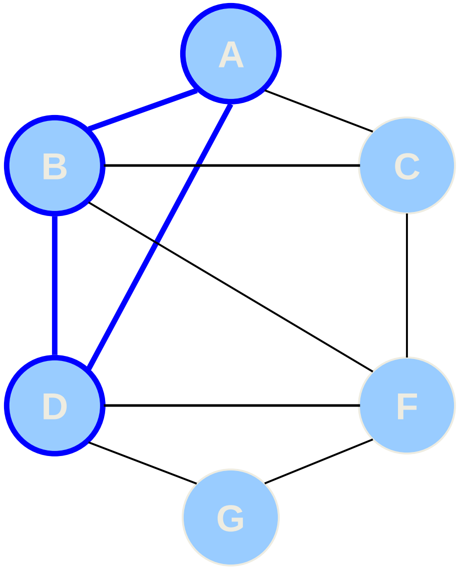
$D = \{D_1, ..., D_n\}$ - domains

$F = \{f_1, ..., f_m\}$ - cost functions

f(A,B,D) has scope {A,B,D}

A	B	D	Cost
1	2	3	3
1	3	2	2
2	1	3	0
2	3	1	0
3	1	2	5
3	2	1	0

Primal graph =
Variables --> nodes
Functions, Constraints -→ arcs



F(a,b,c,d,f,g)= f1(a,b,d)+f2(d,f,g)+f3(b,c,f)

Global Cost Function

$$F(X) = \sum_{i=1}^{m} f_i(X)$$

Constraint Networks

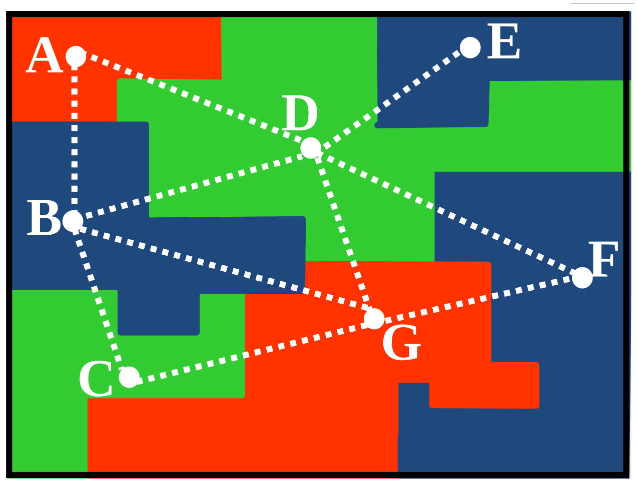
Map coloring

Variables: countries (A B C etc.)

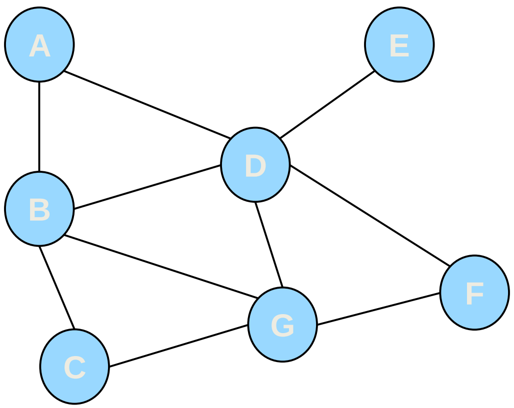
Values: colors (red green blue)

Constraints: **A ≠ B, A ≠ D, D ≠ E**, *etc.*

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

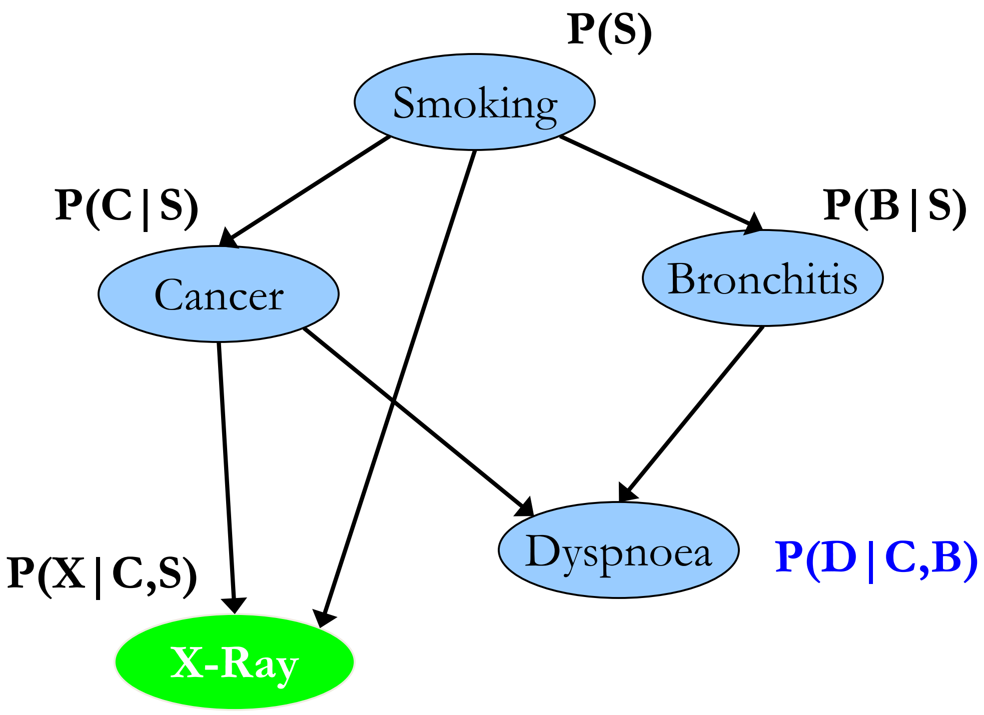


Constraint graph



Probabilistic Networks

BN = (X,D,G,P)



P(D|C,B)

C	B	D=0	D=1
0	0	0.1	0.9
0	1	0.7	0.3
1	0	0.8	0.2
1	1	0.9	0.1

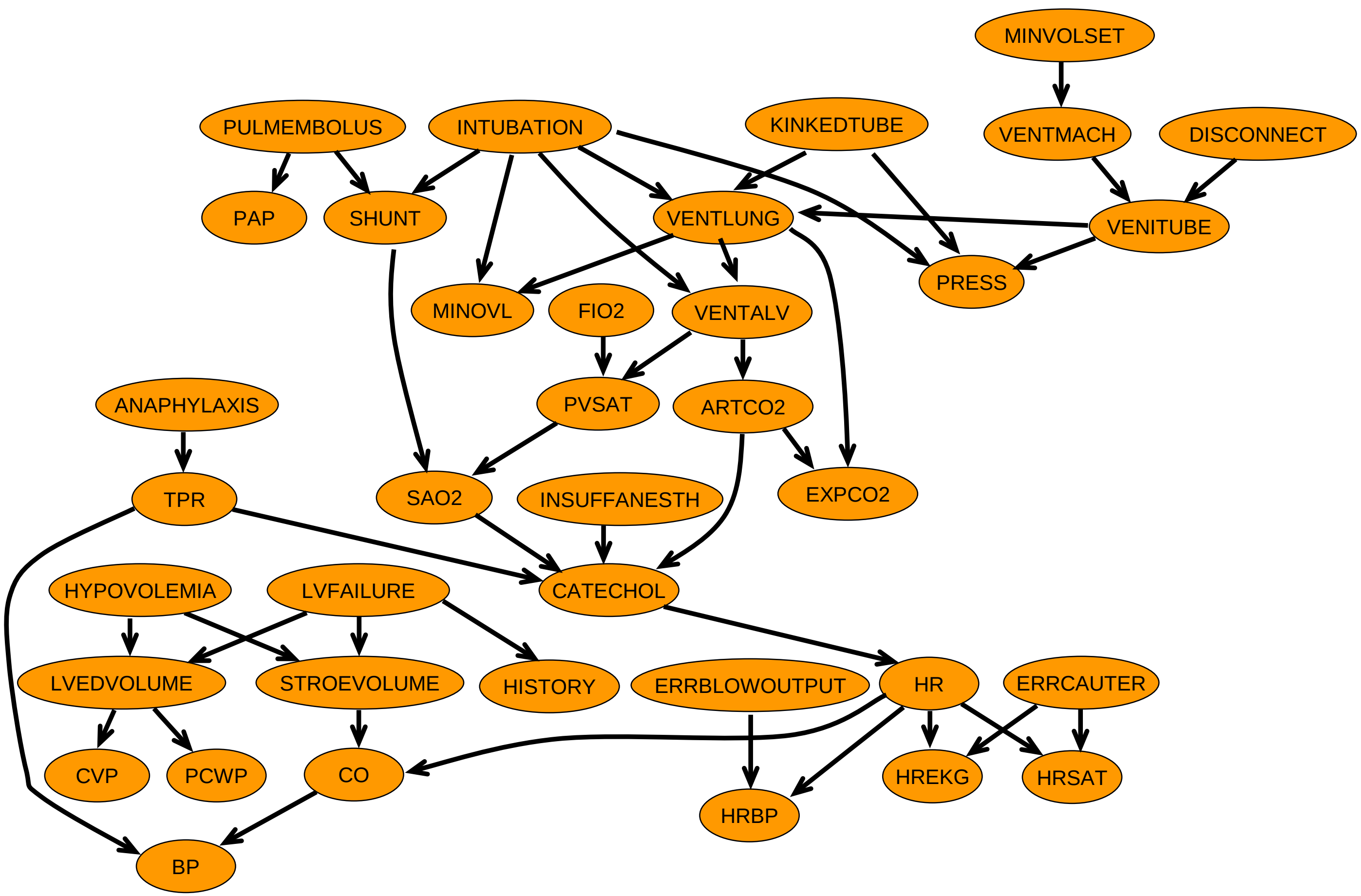
P(S,C,B,X,D) = P(S)· P(C|S)· P(B|S)· P(X|C,S)· P(D|C,B)

MPE= Find a maximum probability assignment, given evidence

MPE= find argmax P(S)· P(C|S)· P(B|S)· P(X|C,S)· P(D|C,B)

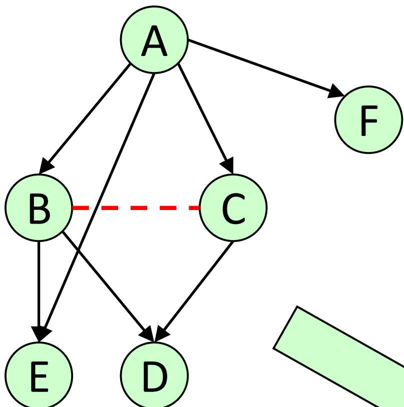
Monitoring Intensive-Care Patients

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



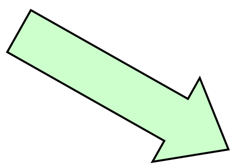
Mixed Networks (Mateescu and Dechter, 2004)

Belief Network

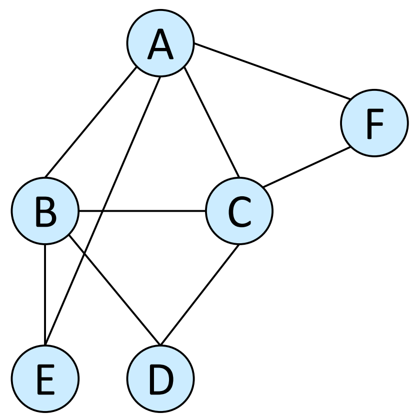


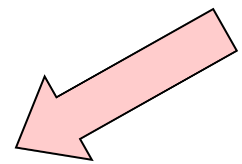
$P(D \mid B,C)$

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

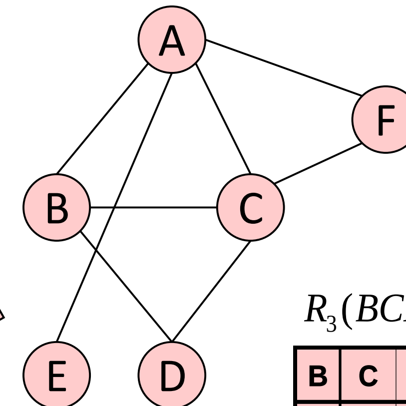


Moral mixed graph





Constraint Network



$R_3(BCD)$

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

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

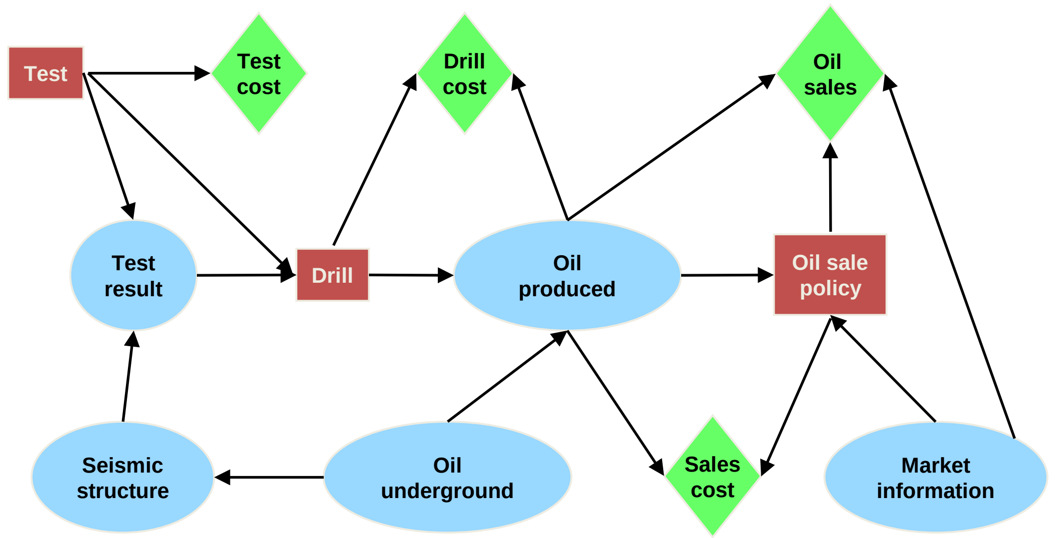
$$P_M(\bar{x}) = \begin{cases} P_B(\bar{x} \mid \bar{x} \in \rho) = \dfrac{P_B(\bar{x})}{P_B(\bar{x} \in \rho)}, \text{if } \bar{x} \in \rho \\ 0, \text{otherwise} \end{cases}$$

Influence Diagrams

Task: find optimal policy:

$$E = \max_{\Delta=(\delta_1,\ldots,\delta_m)} \sum_{x=(x_1,\ldots,x_n)} \prod_i P_i(x)u(x)$$

Influence diagram *ID* = (*X,D,P,R*).



Chance variables: $X = X_1,\ldots,X_n$ **over domains.**

Decision variables: $D = D_1,\ldots,D_m$

CPT's for chance variables: $P_i = P(X_i \mid pa_i), i = 1..n$

Reward components: $R = \{r_1,\ldots,r_j\}$

Utility function: $u = \sum_i r_i$

Graphical Models

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

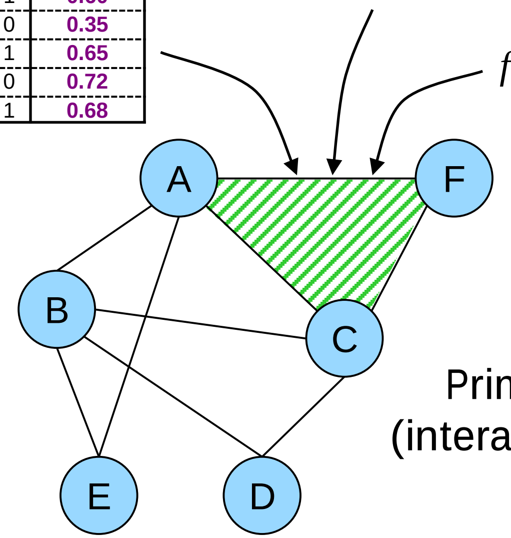
Conditional Probability Table (CPT)

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

Probabilistic Inference Tasks

- BU/ Partition function: Belief updating:

$$\mathbf{BEL(X_i) = P(X_i = x_i \mid evidence)}$$

- MPE: Finding most probable explanation (MPE)

$$\bar{x}^* = \arg\max_{\bar{x}} P(\bar{x}, e)$$

- Marginal Map: Finding maximum a-posteriory hypothesis

$$(a_1^*, \ldots, a_k^*) = \arg\max_{\bar{a}} \sum_{X/A} P(\bar{x}, e)$$

$A \subseteq X$: **hypothesis variables**

- MEU: Finding maximum-expected-utility (MEU) decision

$$(d_1^*, \ldots, d_k^*) = \arg\max_{\bar{d}} \sum_{X/D} P(\bar{x}, e) U(\bar{x})$$

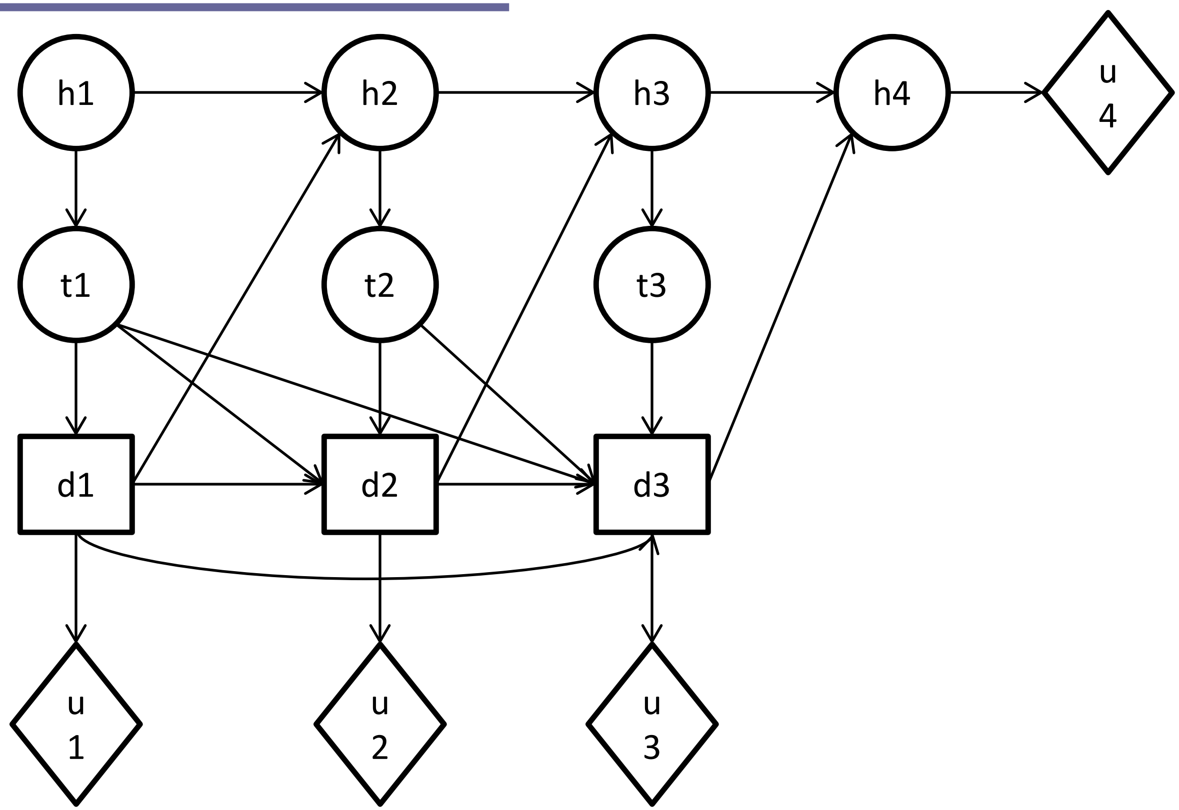
$D \subseteq X$: **decision variables**

$U(\bar{x})$: **utility function**

Dynamic Belief Networks/ Influence Diagrams

hidden state variable

observation

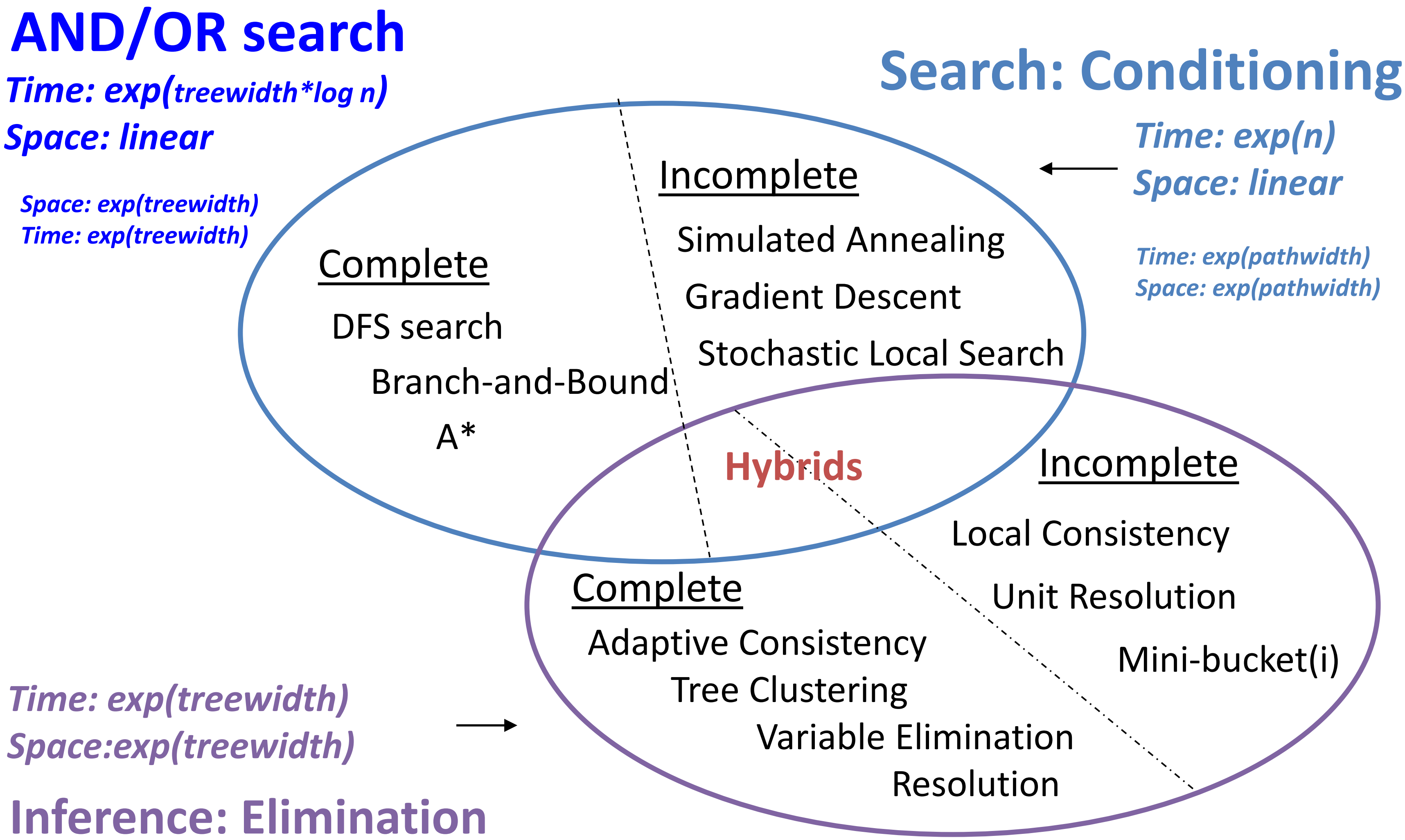


Partially observable states with no-forgetting.

Lauritzen, Nilsson, "Representing and Solving Decision Problems with Limited Information.", 2001

Birds-View of Algorithms: Inference and Search

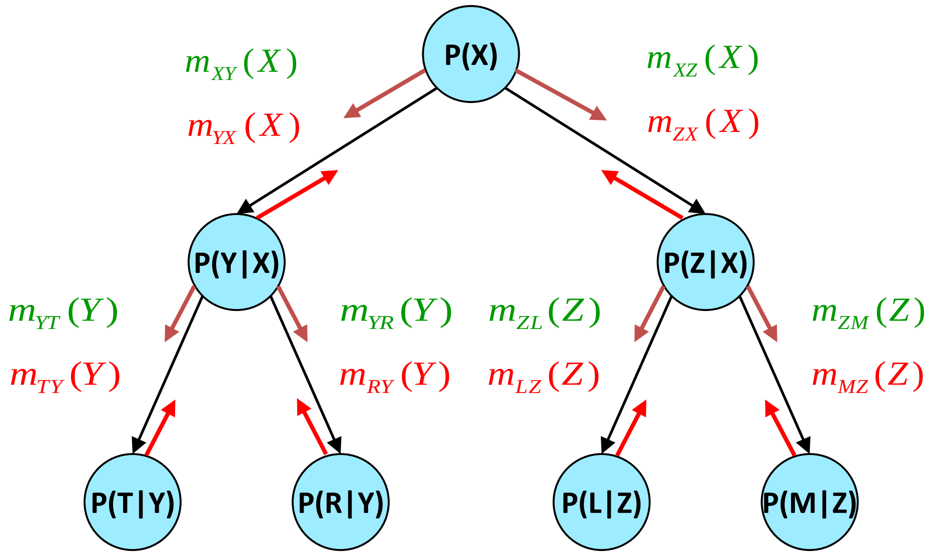
Solution Techniques



Tree-Solving is Easy

Belief updating (sum-prod)

CSP – consistency (projection-join)

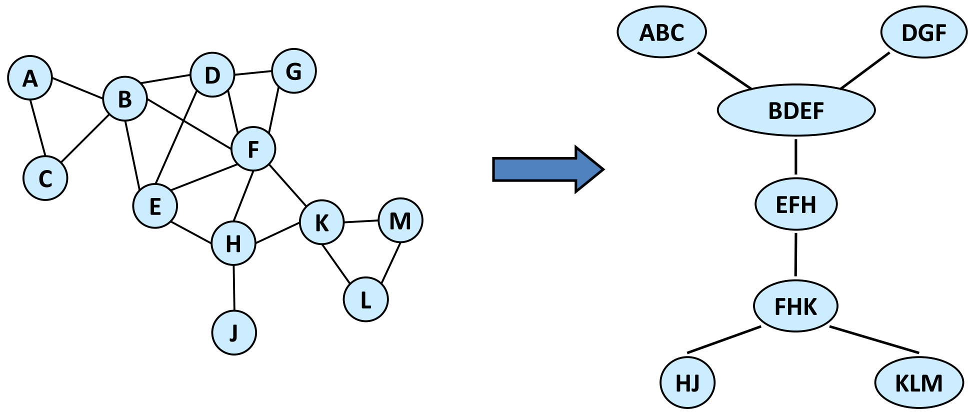


MPE (max-prod)

#CSP (sum-prod)

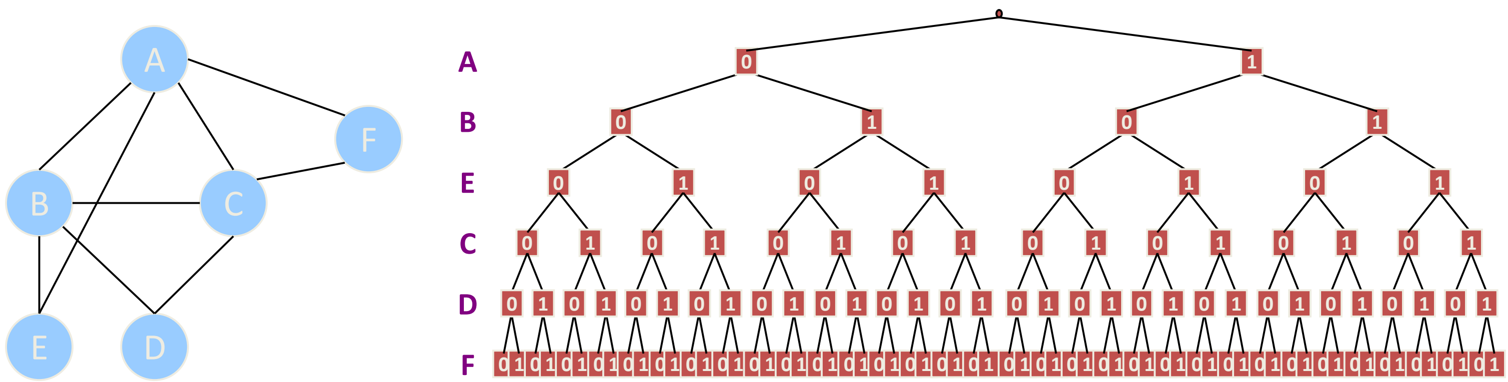
Trees are processed in linear time and memory

Inference vs Conditioning-Search



Inference

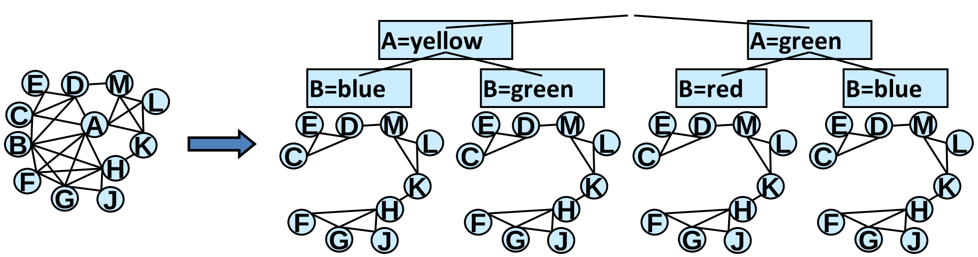
exp(w*) time/space



Search

Exp(n) time

O(n) space



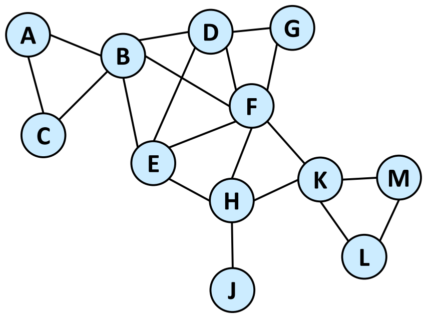
Search+inference:

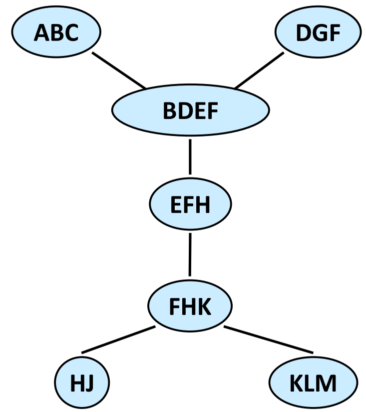
Space: exp(w)

Time: exp(w+c(w))

w: user controlled

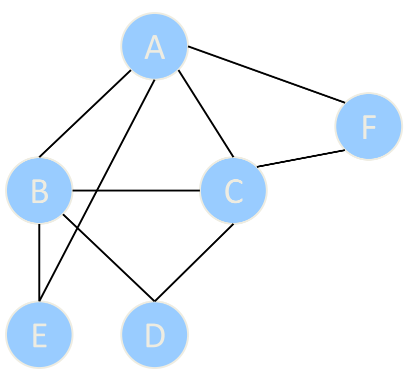
Inference vs Conditioning-Search

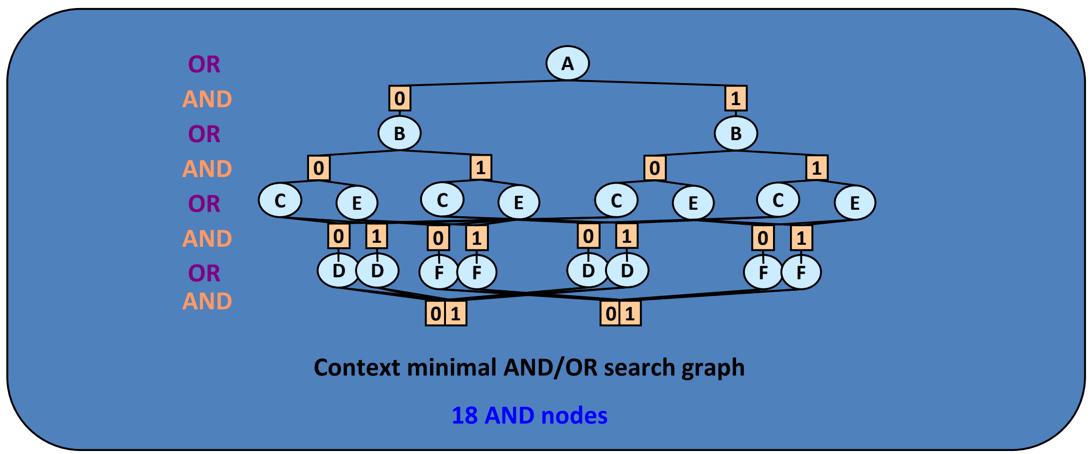




Inference

exp(w*) time/space

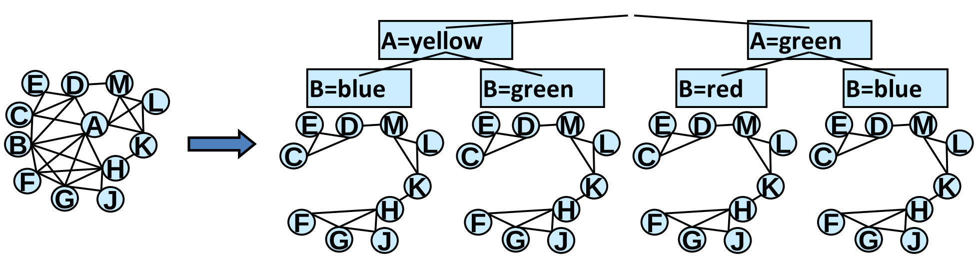




Search

Exp(w*) time

O(w*) space



Search+inference:

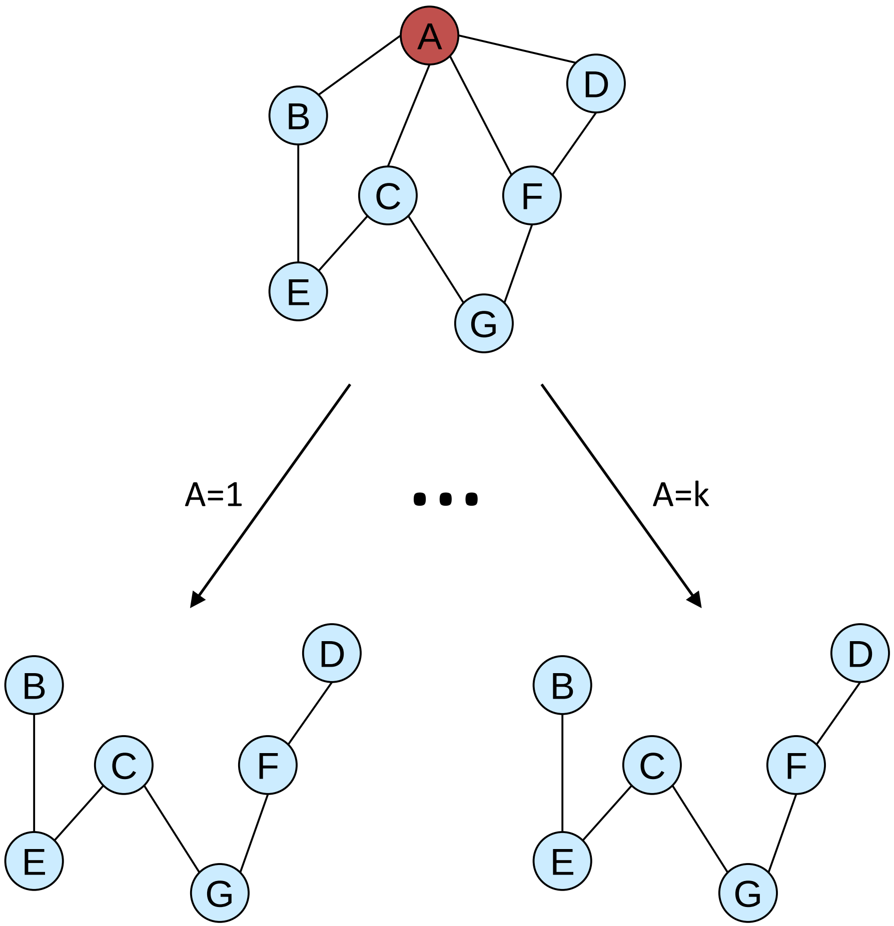
Space: exp(q)

Time: exp(q+c(q))

q: user controlled

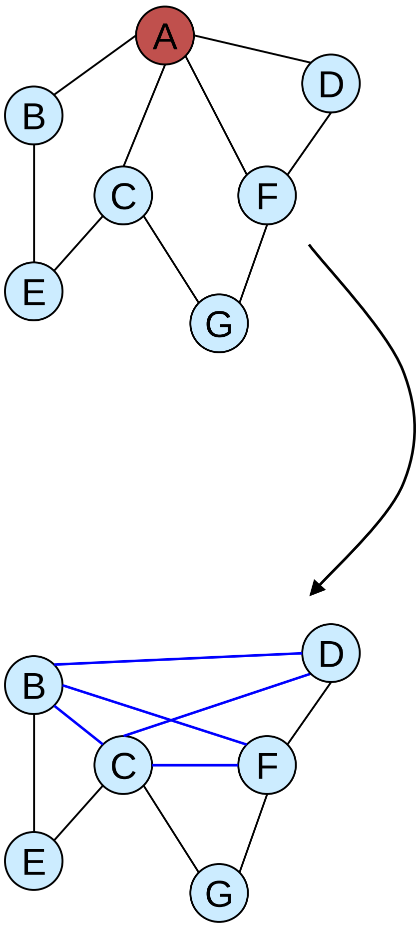
Conditioning vs. Elimination

Conditioning (search)



k “sparser” problems

Elimination (inference)



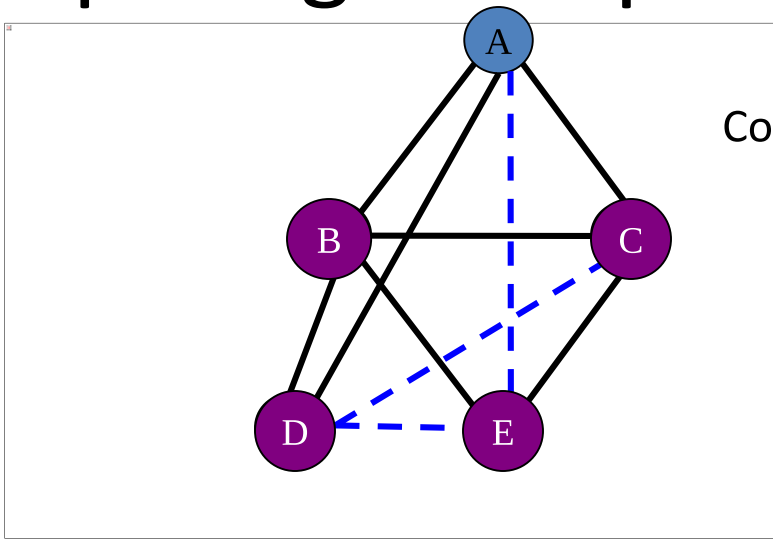
1 “denser” problem

Outline

- **Introduction**
  - Graphical models
  - Optimization tasks for graphical models
  - Birds view of techniques
- **Inference**
  - **Variable Elimination, Bucket Elimination**
- **Search**
  - AND/OR search spaces
  - Depth-First Branch-and-Bound and Best-First Search
- **Lower-bounds and relaxations**
  - Bounded variable elimination
  - Iterative cost shifting and local consistency
- **Advanced tasks for optimization**
  - Marginal Map for Conformant planning
  - M-best search
  - Influence diagrams
- **Software**

Inference: Bucket Elimination

Computing the Optimal Cost Solution



Constraint graph

$$\textbf{OPT} = \min_{e=0,d,c,b} \underbrace{f(a,b)}+f(a,c)+f(a,d)+\underbrace{f(b,c)+f(b,d)+f(b,e)+f(c,e)}$$

Combination

$$\min_{e=0} \min_{d} \; f(a,d) + \min_{c} \; f(a,c)+f(c,e) + \min_{b} \; \underbrace{f(a,b)+f(b,c)+f(b,d)+f(b,e)}_{h^B(a,d,c,e)}$$

Variable Elimination

Query 1: find $OPT = \min_{X_1,\ldots,X_n} \sum_{j=1}^{r} f_j(X)$

Algorithm **elim-opt** (Dechter, 1996)
Non-serial Dynamic Programming (Bertele & Briochi, 1973)

$$OPT = \min_{a,e,d,c,b} F(a,b) + F(a,c) + F(a,d) + F(b,c) + F(b,d) + F(b,e) + F(c,e)$$

$\min_{b} \sum$ ← Elimination operator

bucket B: f(a,b) f(b,c) f(b,d) f(b,e)

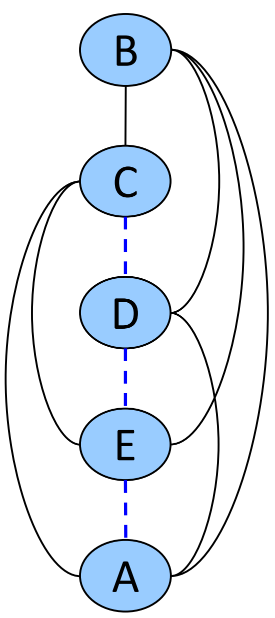
bucket C: f(c,a) f(c,e) $h^B(a,d,c,e)$

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

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

bucket A: $h^E(a)$

OPT



Generating the Optimal Assignment

5. $b' = \arg\min_b f(a',b) + f(b,c') + f(b,d') + f(b,e')$

4. $c' = \arg\min_c f(c,a') + f(c,e') + h^B(a',d',c,e')$

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

2. $e' = 0$

1. $a' = \arg\min_a h^E(a)$

B: f(a,b) f(b,c) f(b,d) f(b,e)

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

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

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

A: $h^E(a)$

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

Combination of Cost Functions

A	B	f(A,B)
b	b	6
b	g	0
g	b	0
g	g	6

+

B	C	f(B,C)
b	b	6
b	g	0
g	b	0
g	g	6

A	B	C	f(A,B,C)
b	b	b	12
b	b	g	6
b	g	b	0
b	g	g	6
g	b	b	6
g	b	g	0
g	g	b	6
g	g	g	12

= 0 + 6

Elimination in a Cost Function

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

Min

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

A	**g(A)**
b	1
g	2
r	1

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

$\mathbf{h}_{\varnothing}$
1

Complexity of Bucket Elimination

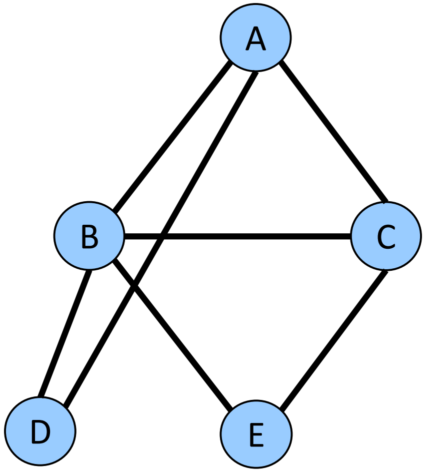
Bucket-Elimination is **time** and **space**

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

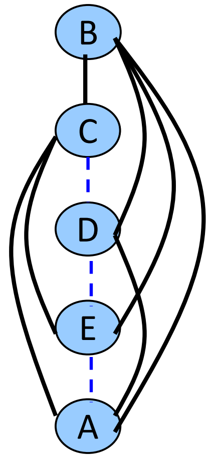
$w^*(d)$ – the induced width of the primal graph along ordering d

r = number of functions

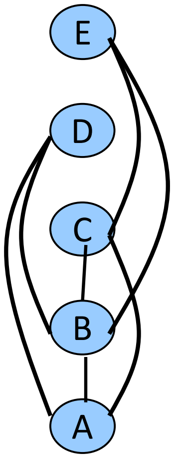
The effect of the ordering:



constraint graph



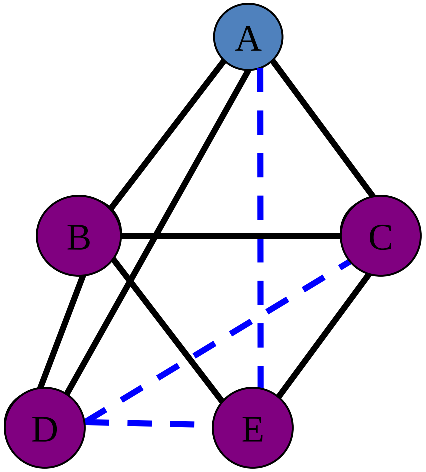
$w^*(d_1) = 4$



$w^*(d_2) = 2$

Finding smallest induced-width is hard!

Query 2: Belief updating: P(X|evidence)=?



"Moral" graph

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

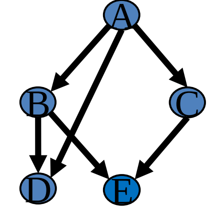
$$\sum_{e=0,d,c,b} P(a)\underbrace{P(b|a)}P(c|a)\underbrace{P(d|b,a)P(e|b,c)}=$$

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

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) $\lambda^B(a,d,c,e)$

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

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

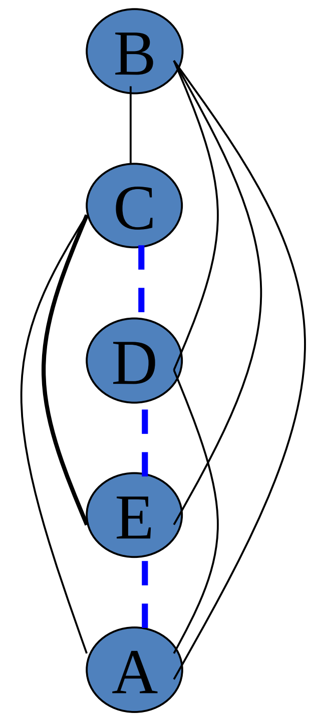
bucket A: P(a) $\lambda^E(a)$

P(e=0)

P(a|e=0)

W*=4

”induced width”
(max clique size)



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

Combination of Cost Functions

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

A sum-product algorithm

BE-BEL

Input: A belief network $\{P_1, ..., P_n\}$, d,e.
Output: belief of X_1 given e.

1. **Initialize:**
2. **Process buckets** from $p = n$ to 1
   for matrices $\lambda_1, \lambda_2, ..., \lambda_j$ in $bucket_p$ do
   - **If** (observed variable) $X_p = x_p$ assign $X_p = x_p$ to each λ_i.
   - **Else**, (multiply and sum)
     $\lambda_p = \sum_{X_p} \Pi_{i=1}^{j} \lambda_i$.
     Add λ_p to its bucket.
3. **Return** $Bel(x_1) = \alpha P(x_1) \cdot \Pi_i \lambda_i(x_1)$

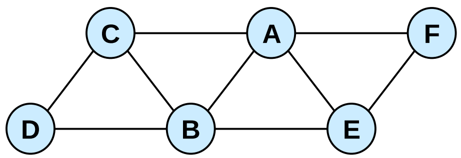
Marginal Map and ID to come…

Outline

- **Introduction**
  - Graphical models
  - Optimization tasks for graphical models
  - Birds view of techniques
- **Inference**
  - Variable Elimination, Bucket Elimination
- **Search**
  - AND/OR search spaces
  - Depth-First Branch-and-Bound and Best-First Search
- **Lower-bounds and relaxations**
  - Bounded variable elimination
  - Iterative cost shifting and local consistency
  - Using bounds as heuristic functions
- **Advanced tasks for optimization**
  - Marginal Map for Conformant planning
  - Influence diagrams
- **Software**

Search:
AND/OR search spaces

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**

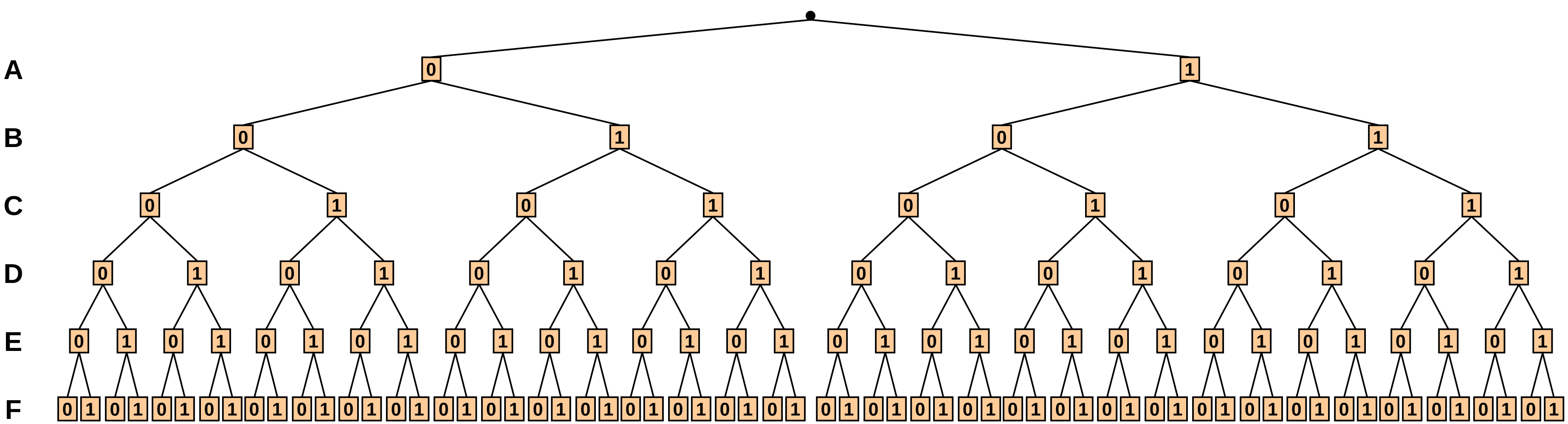
B	D	f_6
0	0	**4**
0	1	**2**
1	0	**1**
1	1	**0**

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

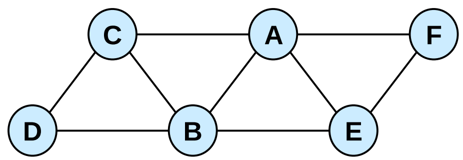
C	D	f_8
0	0	**1**
0	1	**4**
1	0	**0**
1	1	**0**

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

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



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**

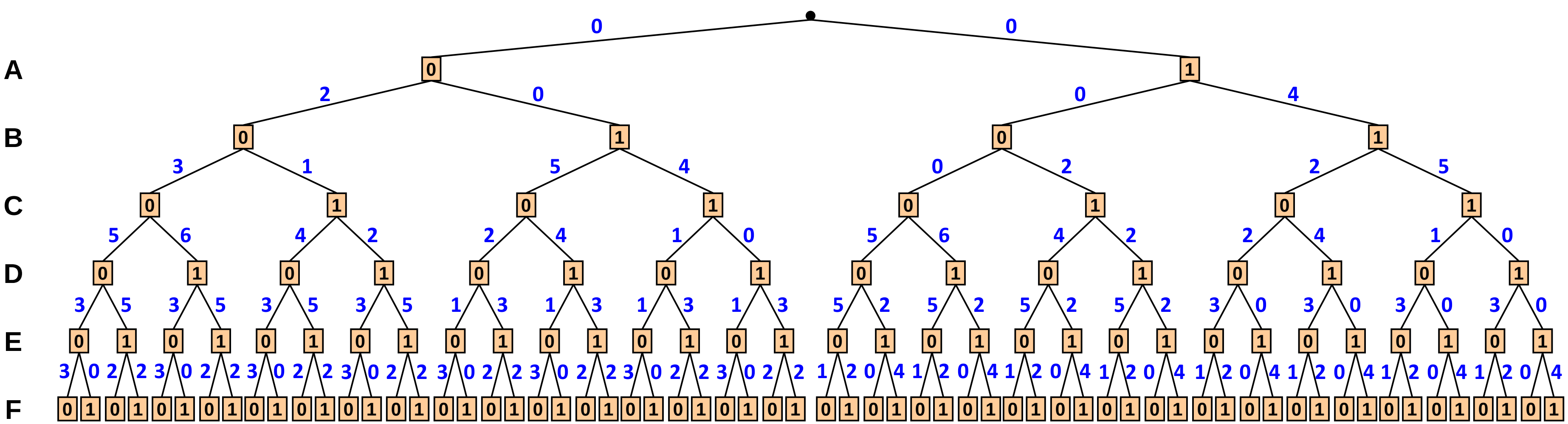
B	D	f_6
0	0	**4**
0	1	**2**
1	0	**1**
1	1	**0**

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**

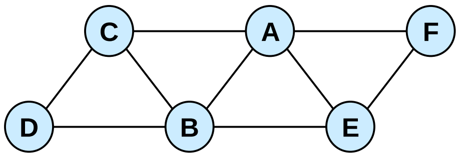
E	F	f_9
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**2**

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



Arc-cost is calculated based from cost functions with empty scope (conditioning)

The Value Function



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**

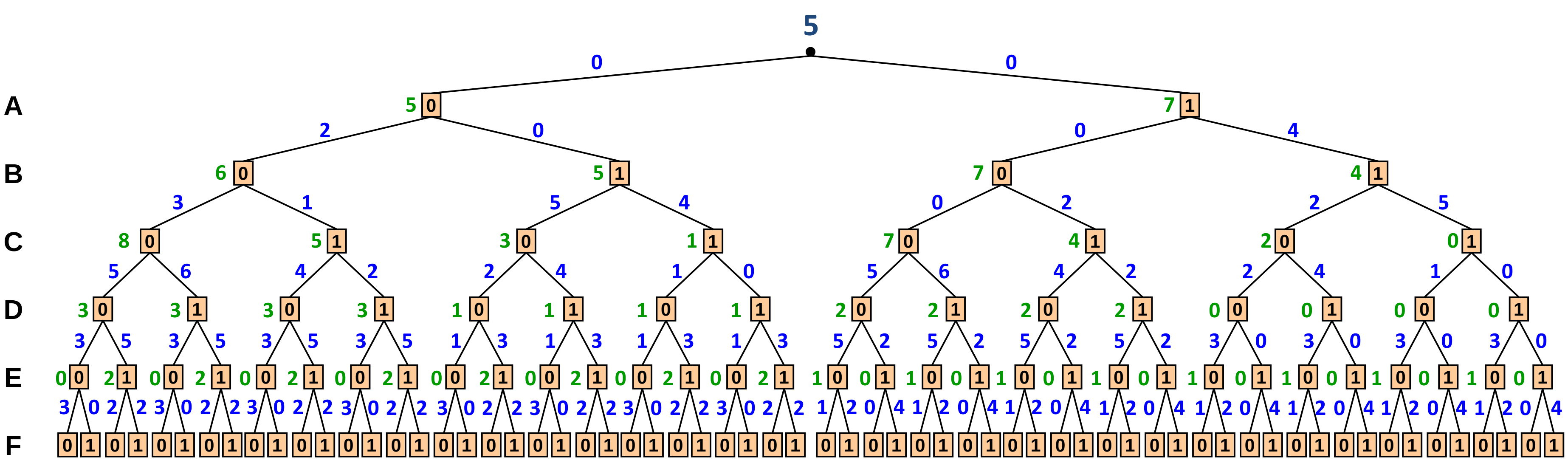
B	D	f_6
0	0	**4**
0	1	**2**
1	0	**1**
1	1	**0**

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**

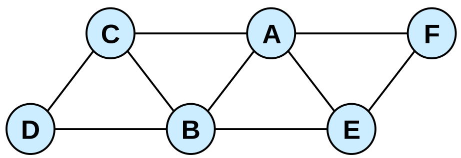
E	F	f_9
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**2**

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



Value of node = 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**

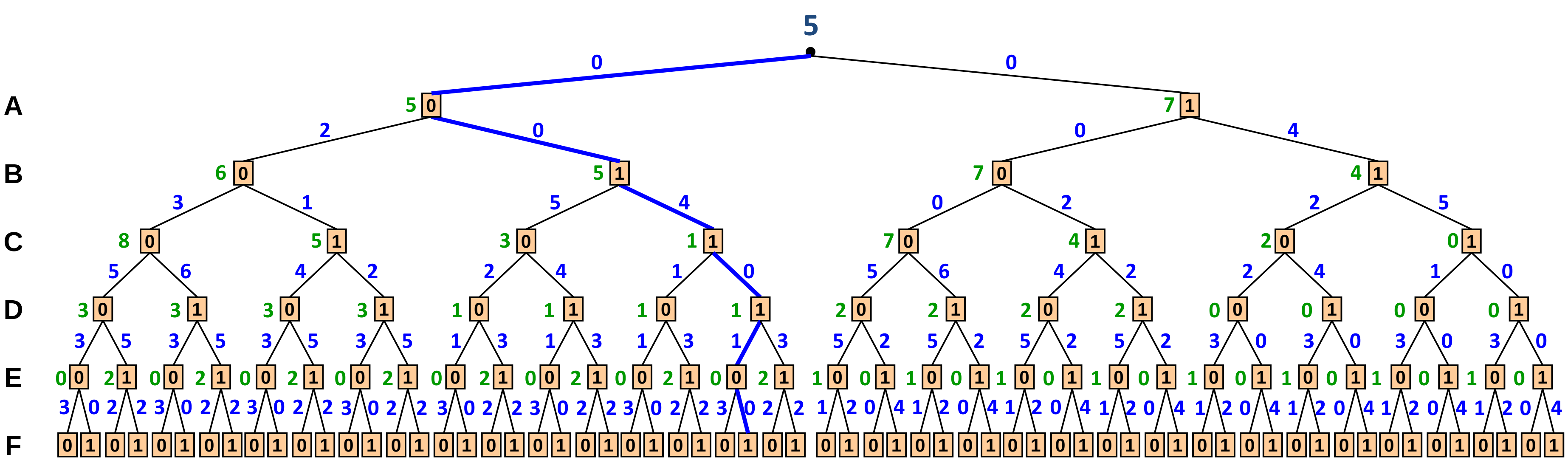
B	D	f_6
0	0	**4**
0	1	**2**
1	0	**1**
1	1	**0**

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**

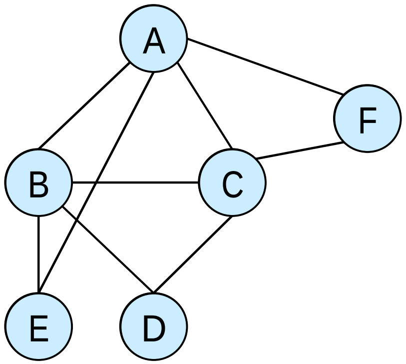
E	F	f_9
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**2**

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

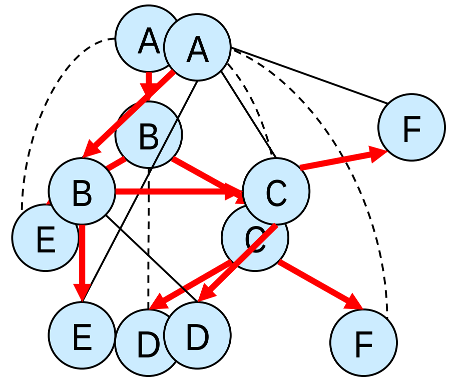


Value of node = minimal cost solution below it

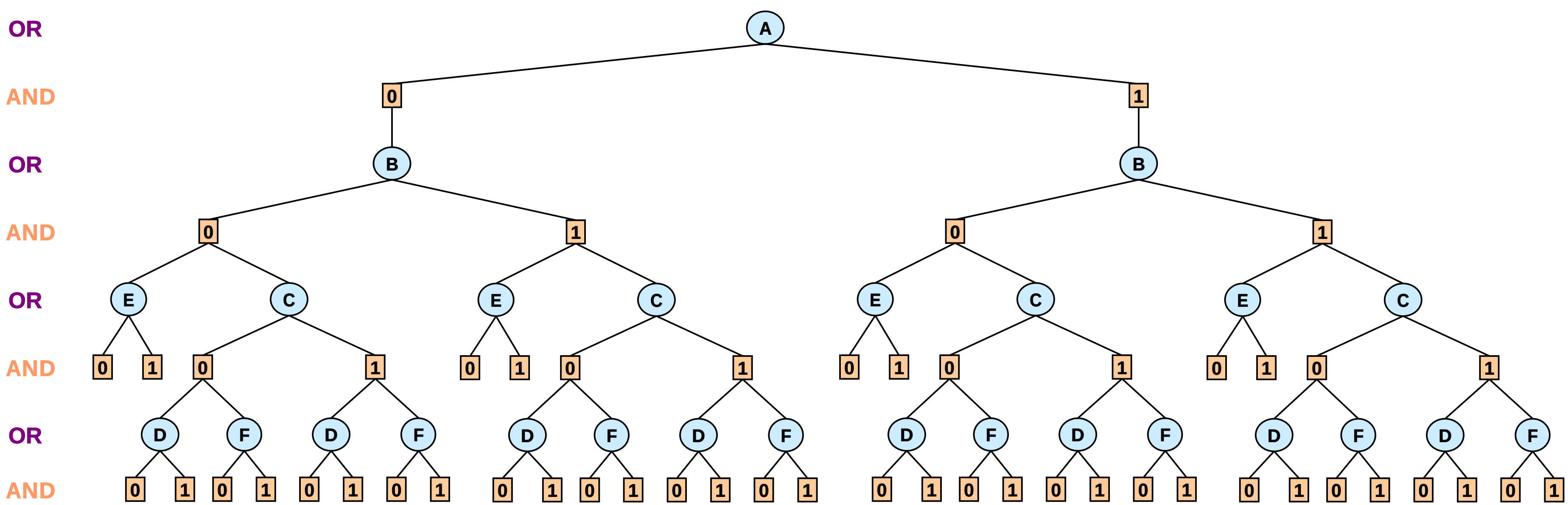
AND/OR Search Space



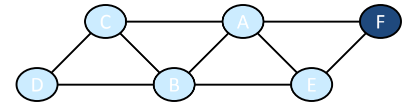
Primal graph

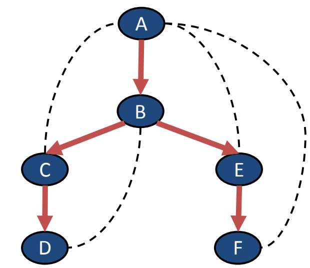


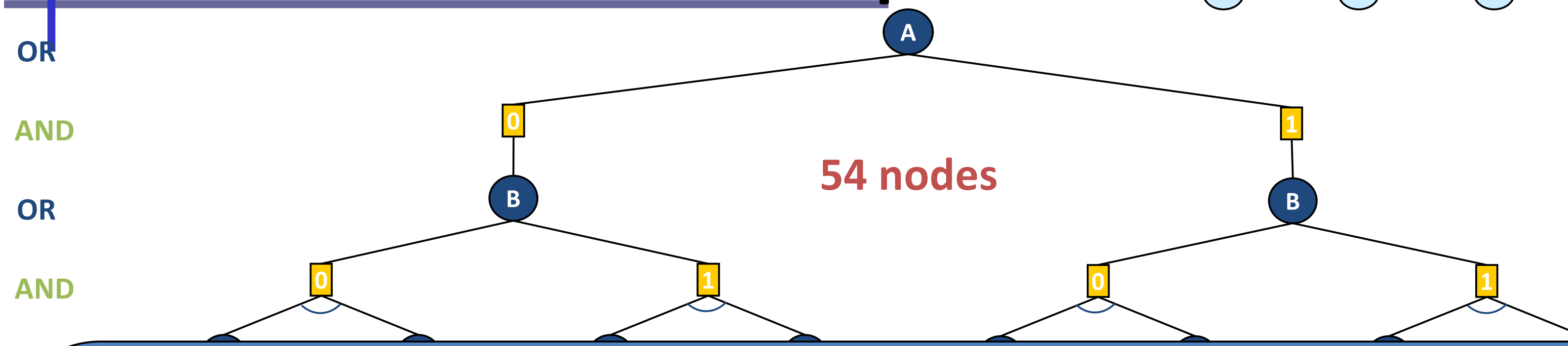
DFS tree



AND/OR vs. OR Spaces



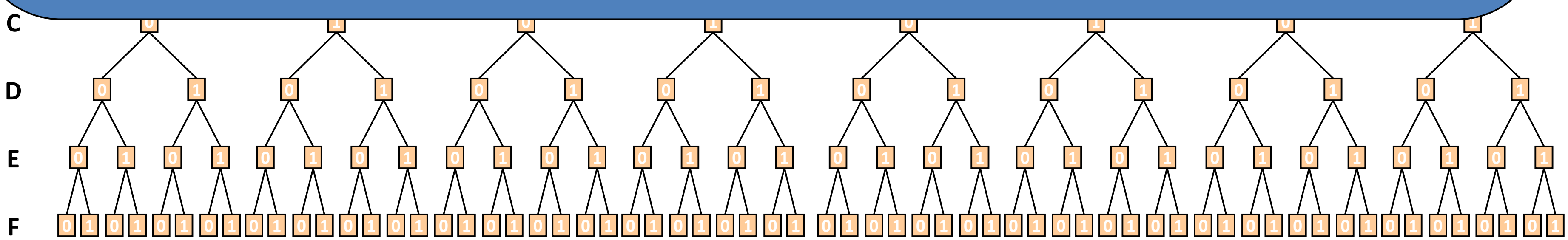




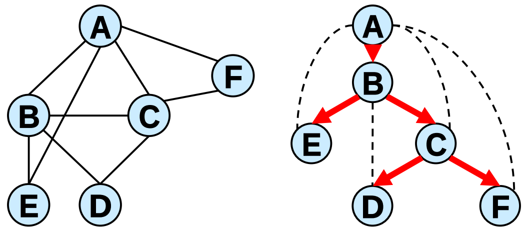
Time $O(nk^h)$

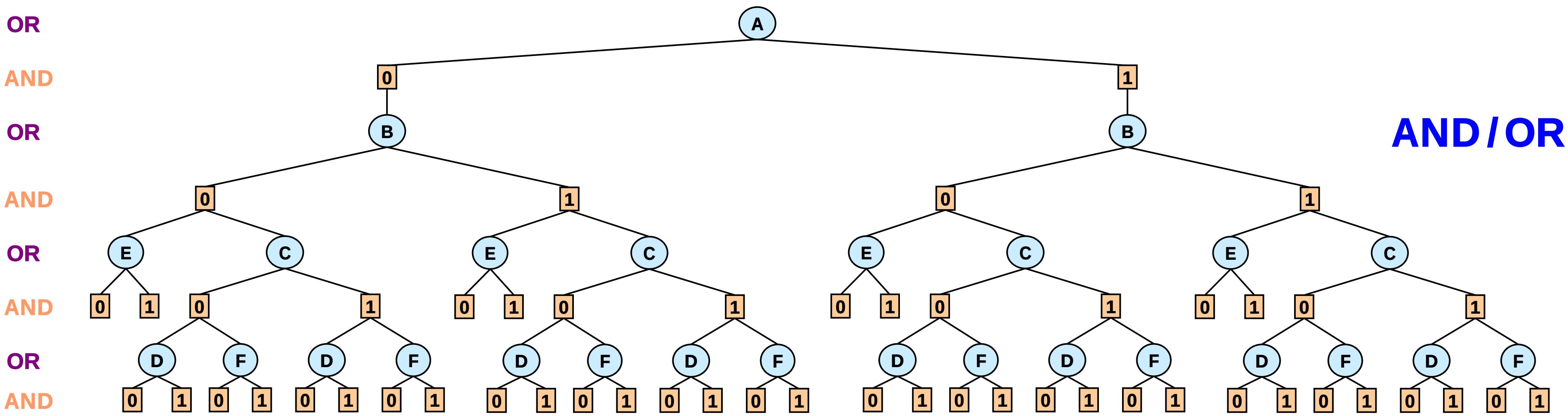
Space O(n)

height is bounded by (log n) w*

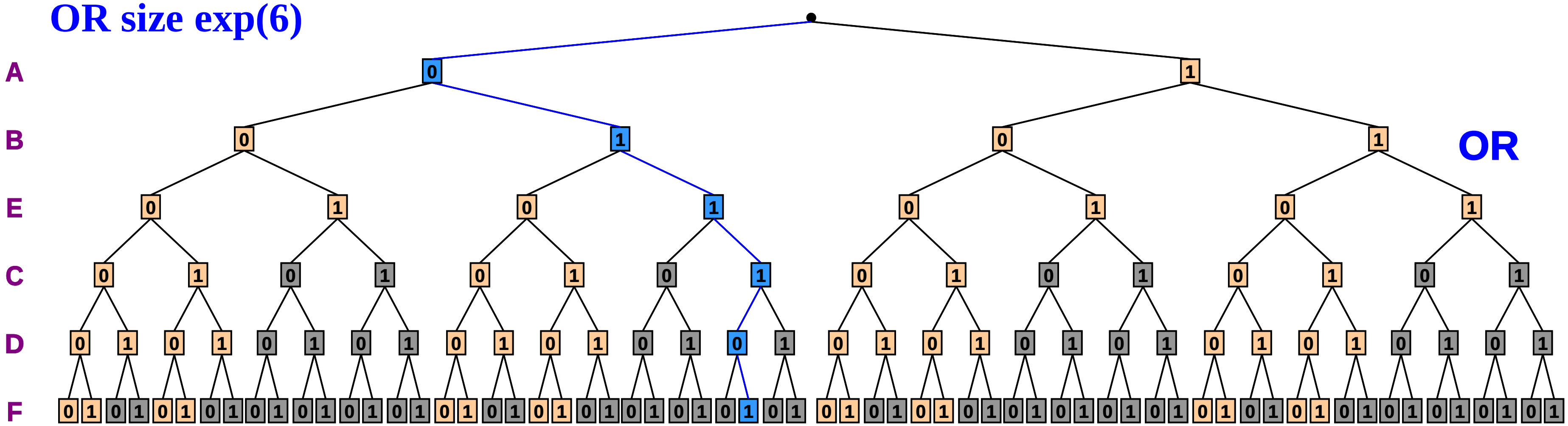


AND/OR vs. OR

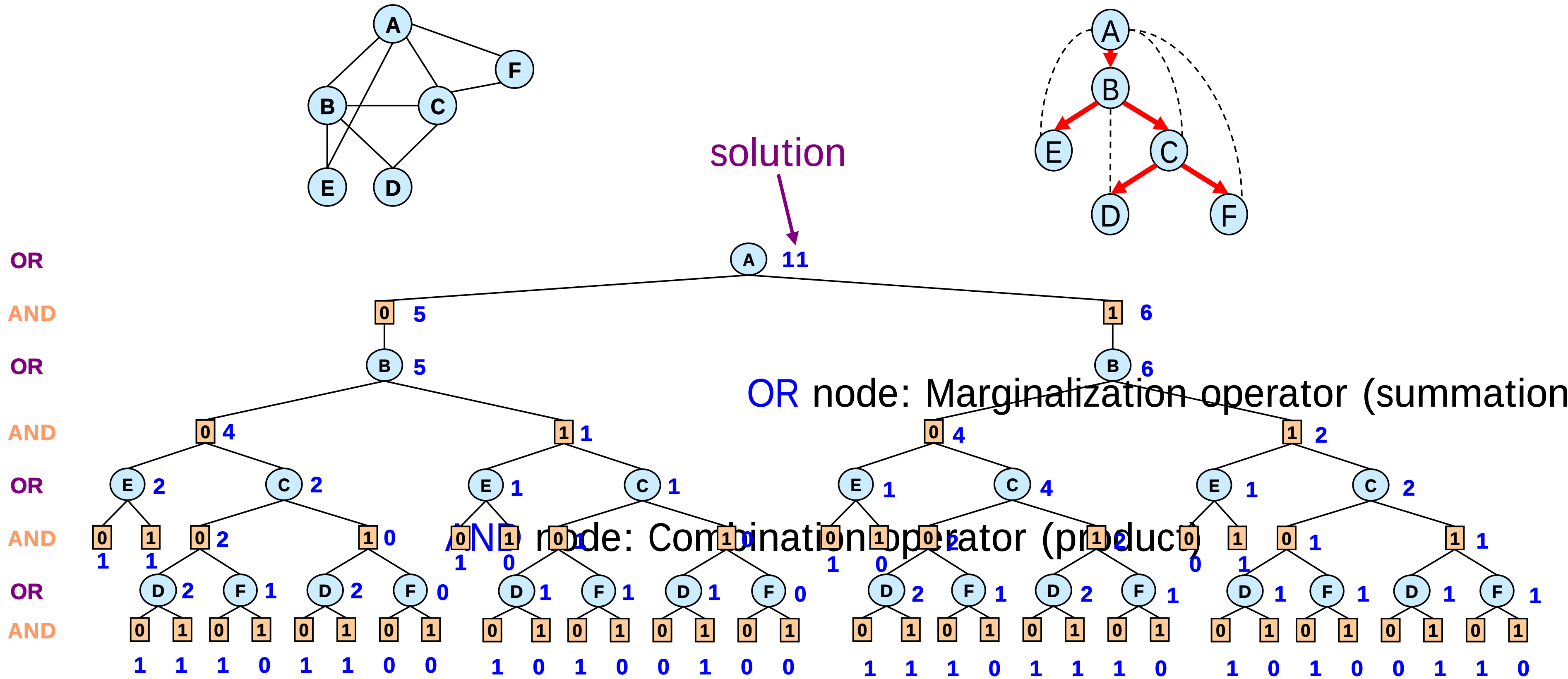




AND/OR size: exp(4),
OR size exp(6)

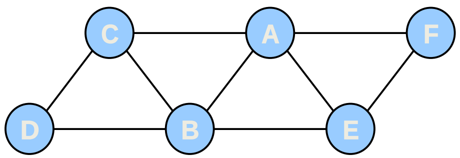


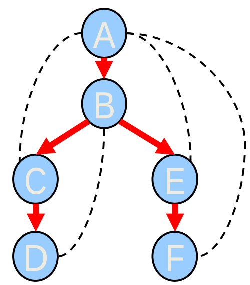
DFS algorithm (#CSP example)



Value of node = number of solutions below it

AND/OR Tree Search for COP





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

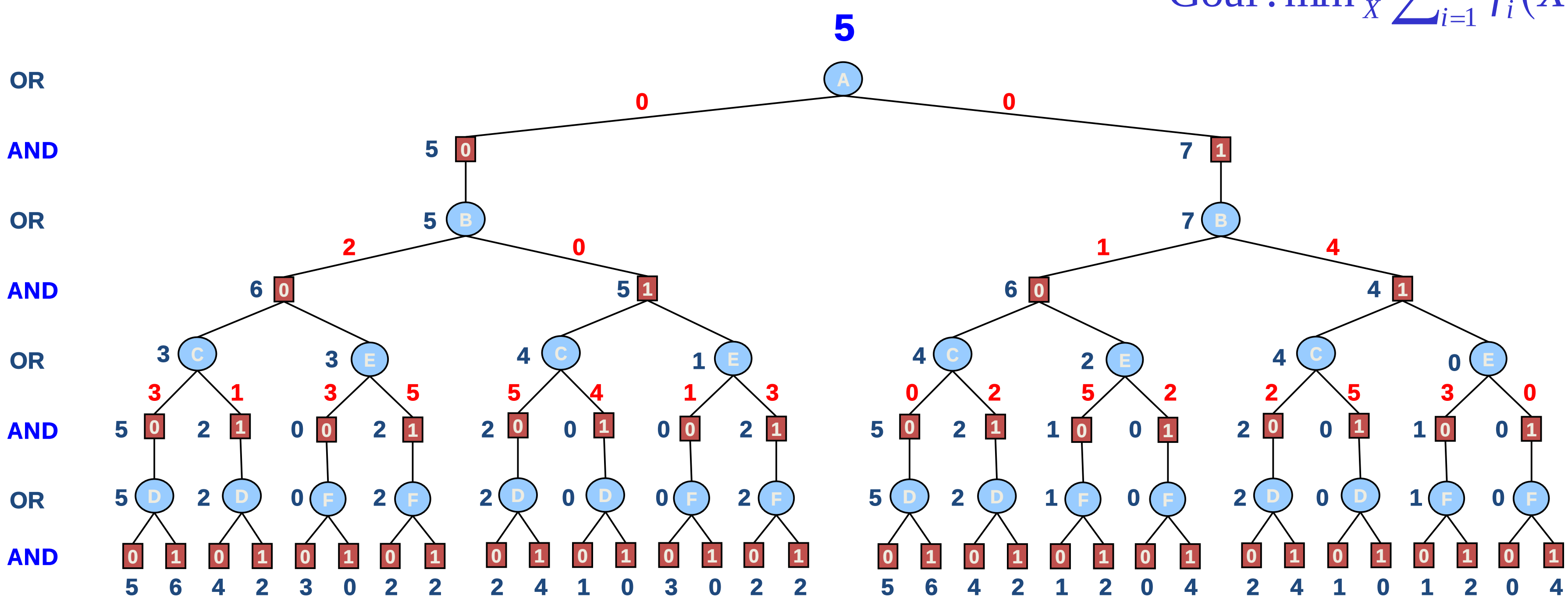
B	D	f_6
0	0	4
0	1	2
1	0	1
1	1	0

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

$$\text{Goal}: \min_X \sum_{i=1}^{9} f_i(X)$$

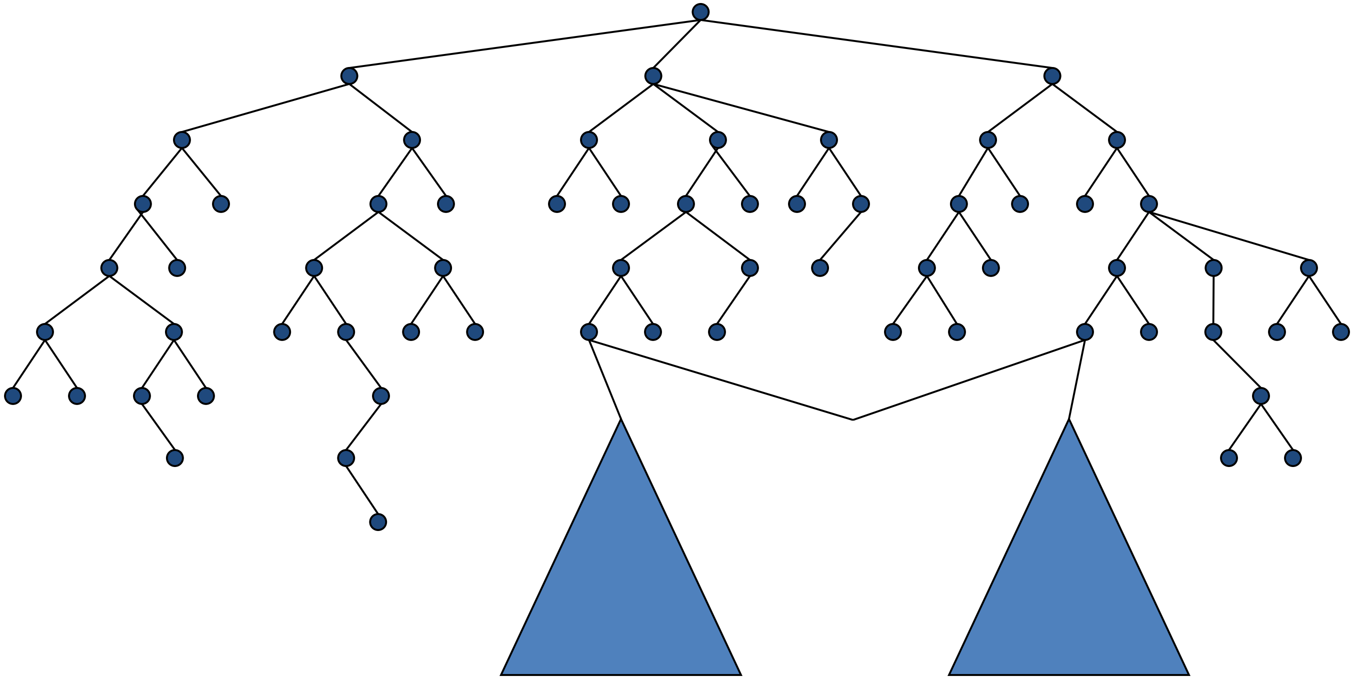


AND node = Combination operator (summation)

OR node = Marginalization operator (minimization)

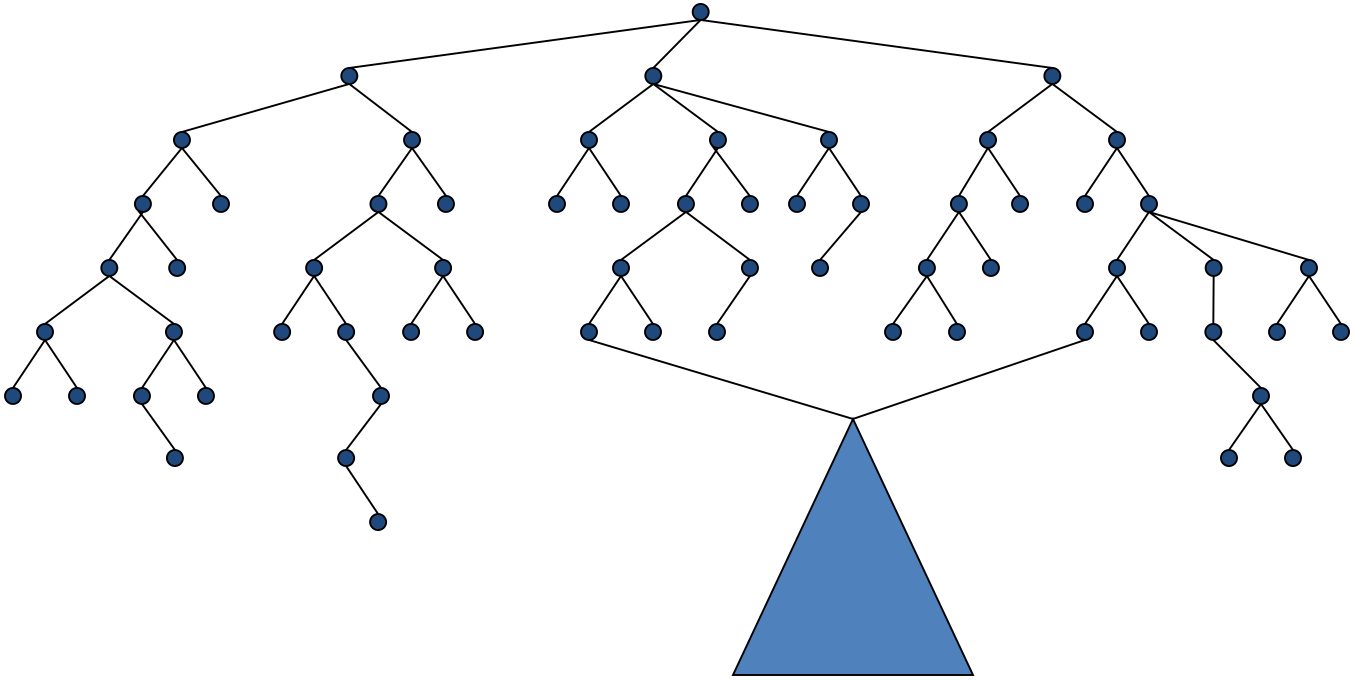
From Search Trees to Search Graphs

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

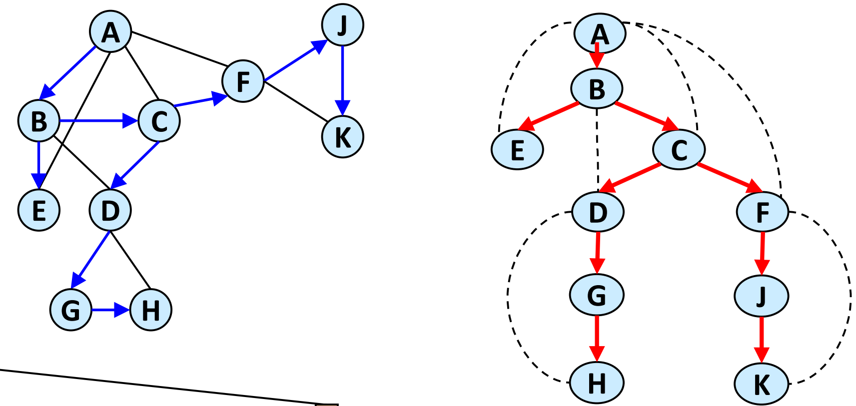


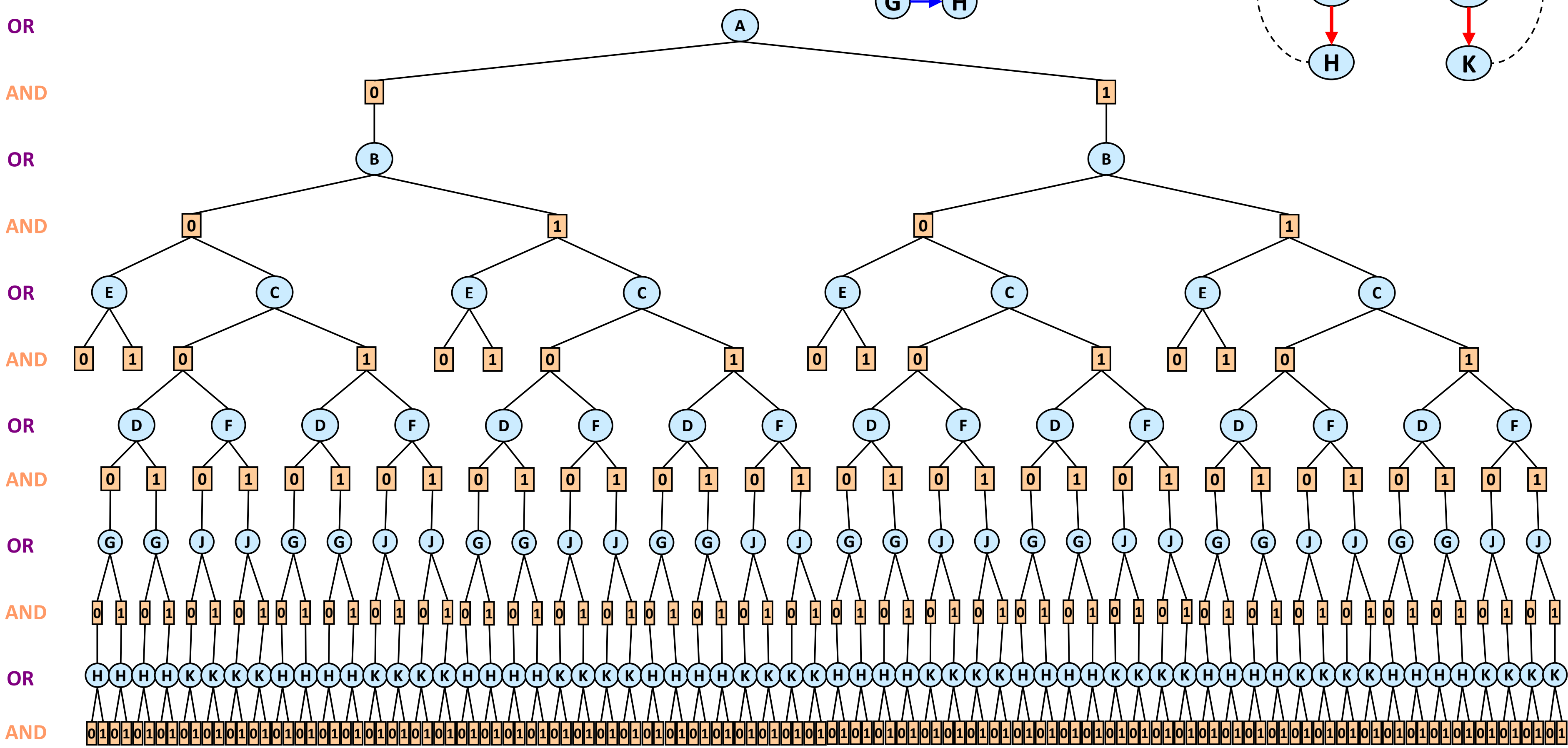
From Search Trees to Search Graphs

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

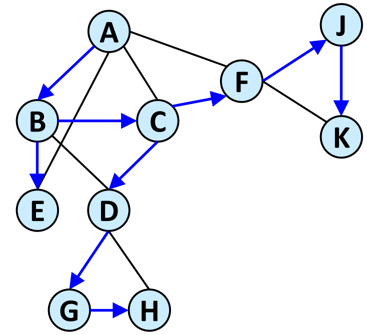


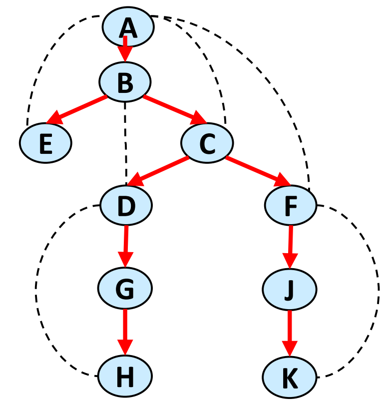
From AND/OR Tree

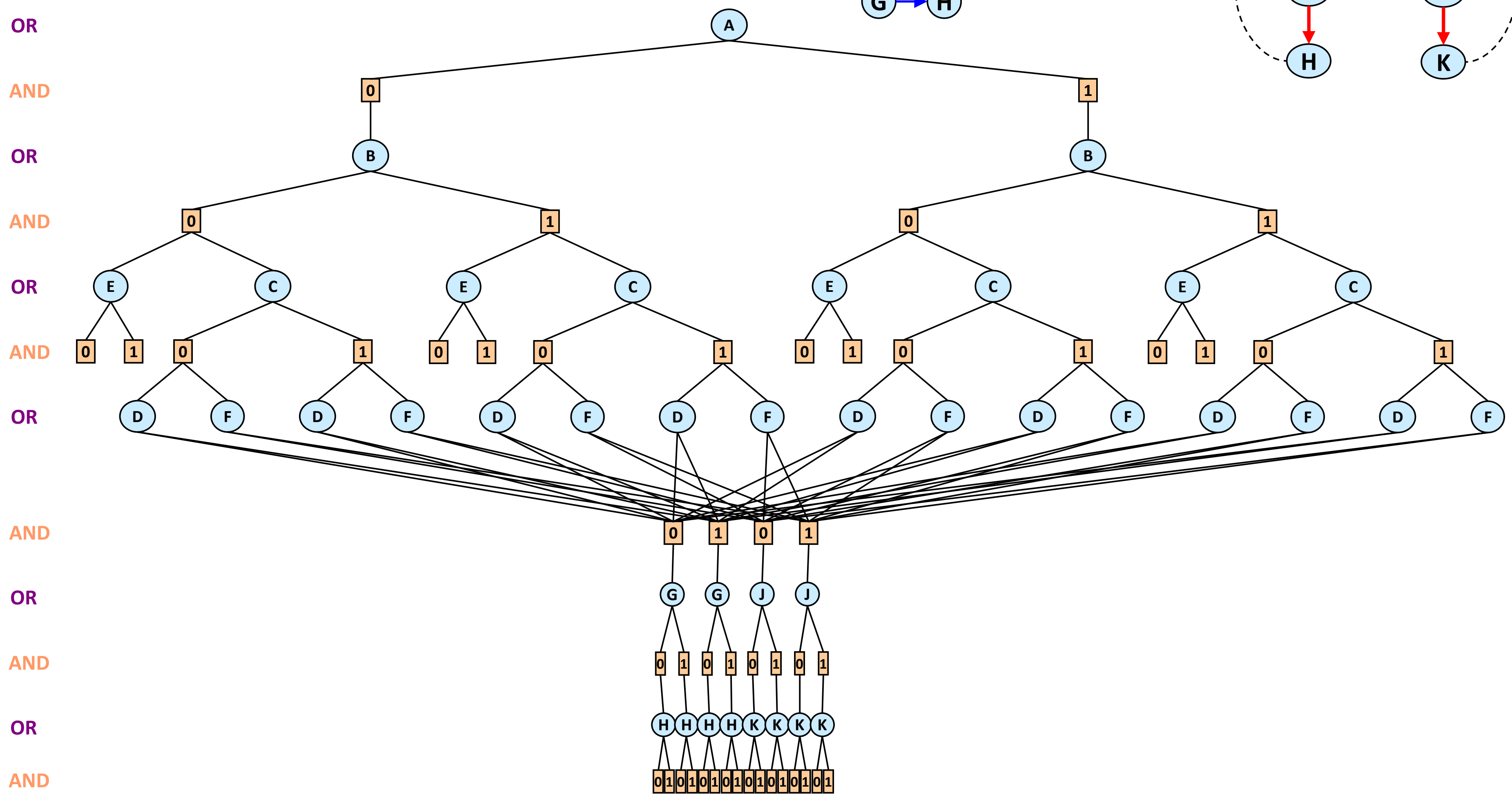




An AND/OR Graph



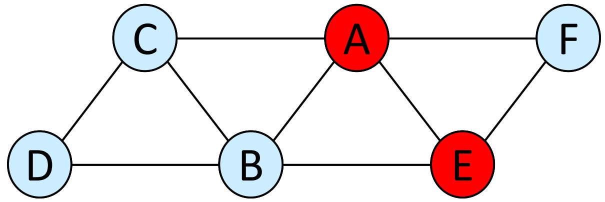


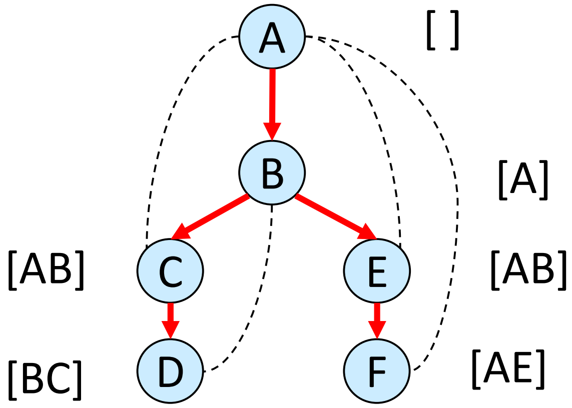


Merging Based on Context

- One way of recognizing nodes that can be merged (based on graph structure)

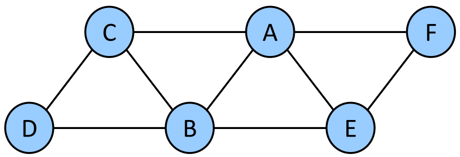
context(X) = ancestors of X in the pseudo tree
that are connected to X, or to
descendants of X

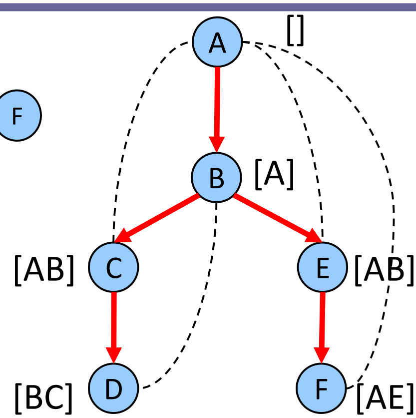




pseudo tree

AND/OR Search Graph





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

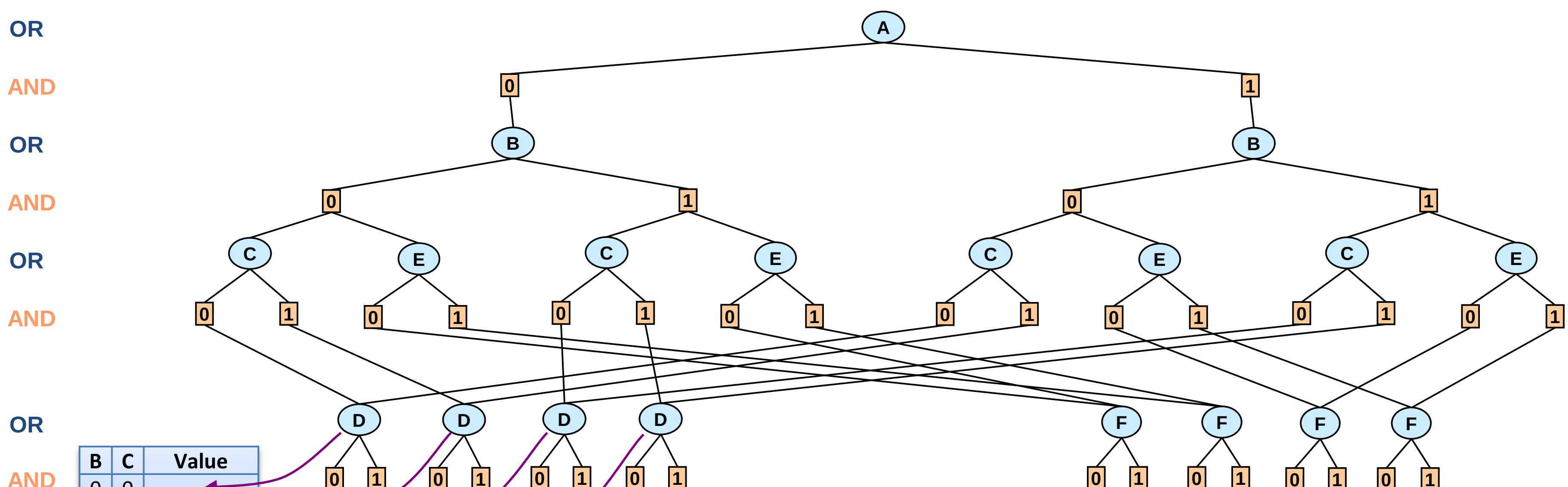
B	D	f_6
0	0	4
0	1	2
1	0	1
1	1	0

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

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



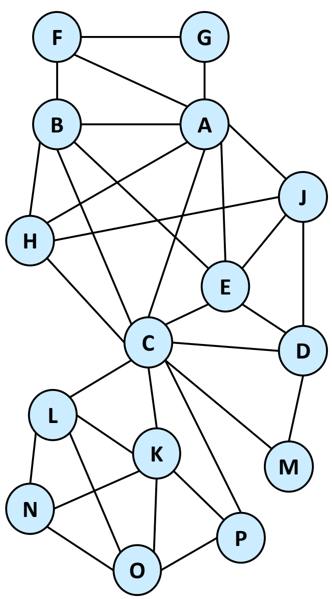
B	C	Value
0	0	
0	1	
1	0	
1	1	

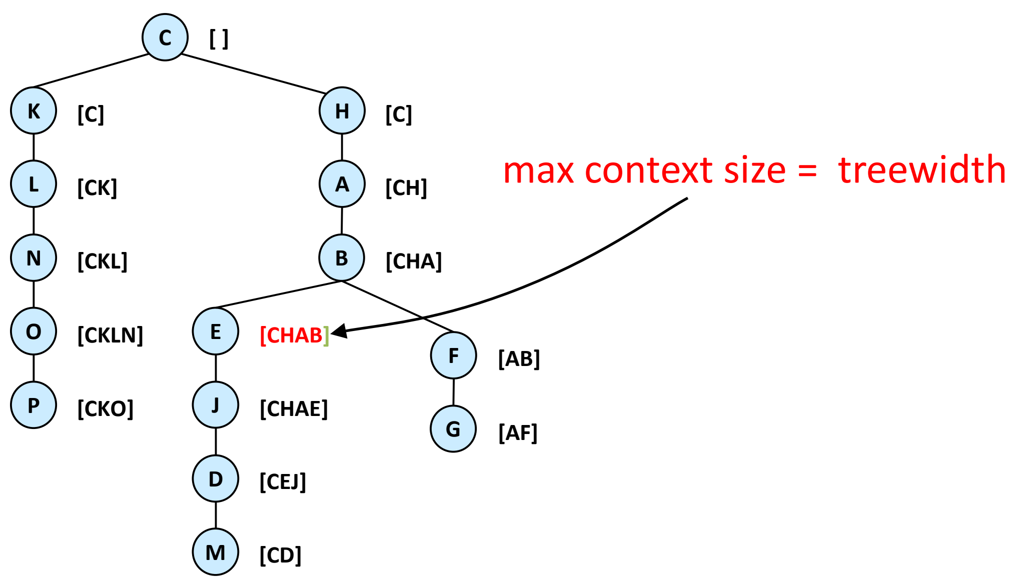
Cache table for D

context minimal graph

How Big Is The Context?

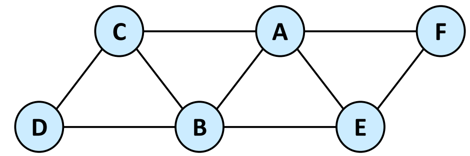
Theorem: The maximum context size for a pseudo tree is equal to the treewidth of the graph along the pseudo tree.

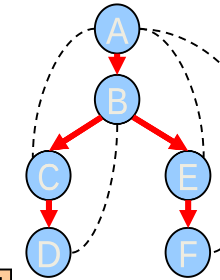




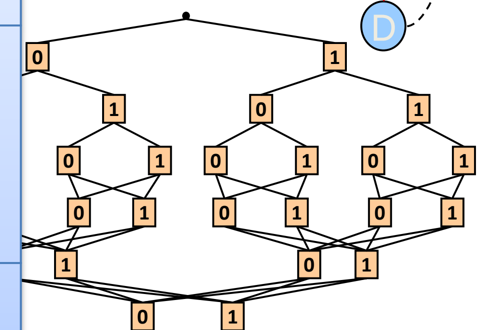
(C K H A B E J L N O D P M F G)

All Four Search Spaces



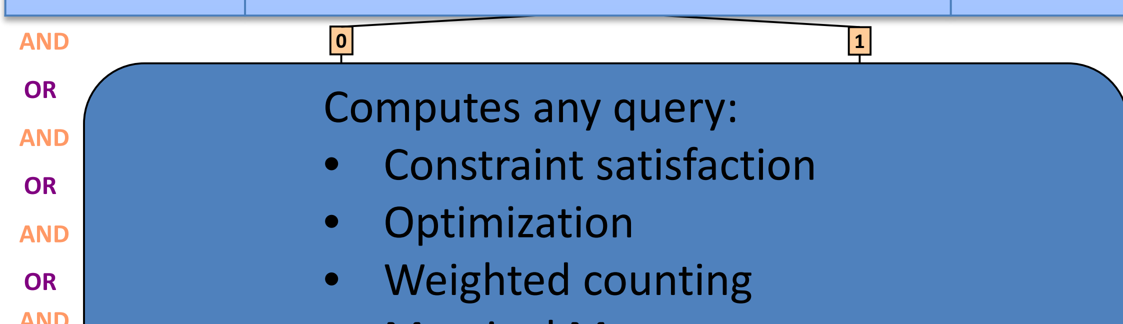


	AND/OR graph	OR graph
Space	$O(n\ k^{w^*})$	$O(n\ k^{pw^*})$
Time	$O(n\ k^{w^*})$	$O(n\ k^{pw^*})$



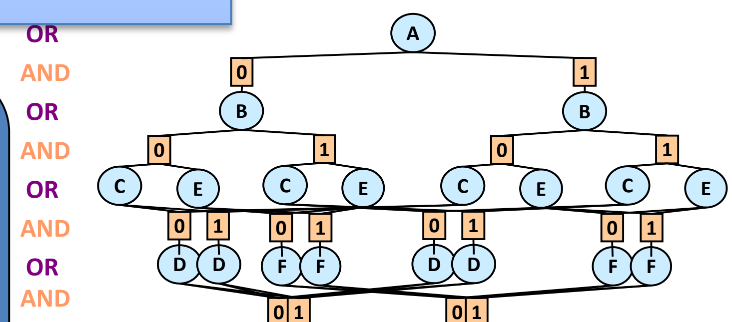
text minimal OR search graph

28 nodes



Computes any query:

- Constraint satisfaction
- Optimization
- Weighted counting
- Marginal Map
- Maximum expected utility



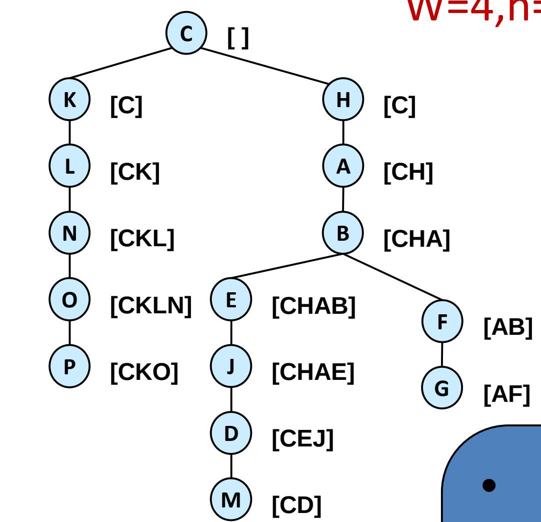
Context minimal AND/OR search graph

18 AND nodes

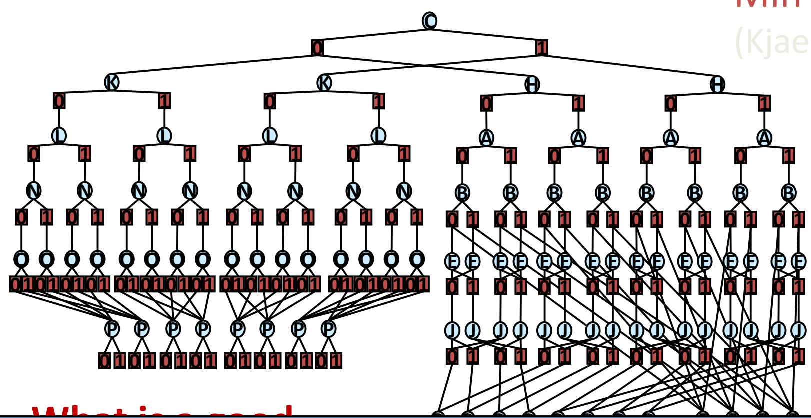
Any query is best computed
Over the c-minimal AO space

The impact of the pseudo-tree

W=4,h=8

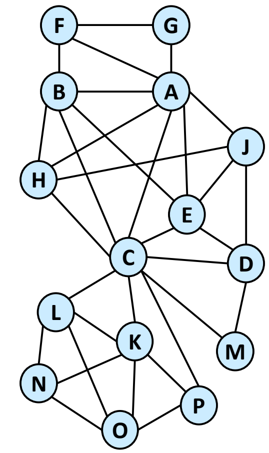


(C K H A B E J L N O D



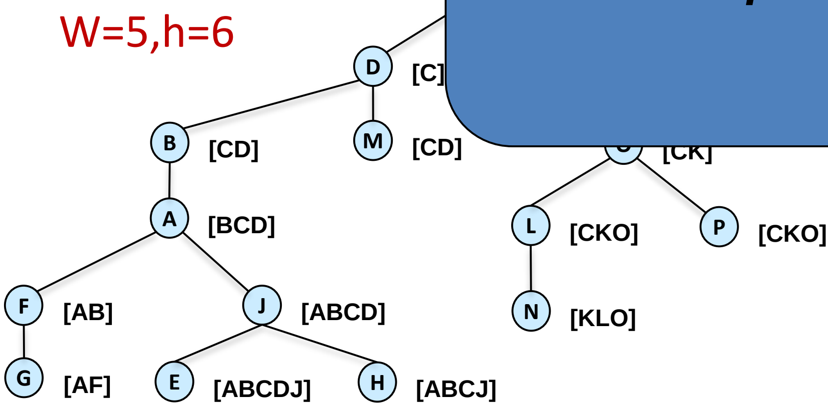
Min-Fill

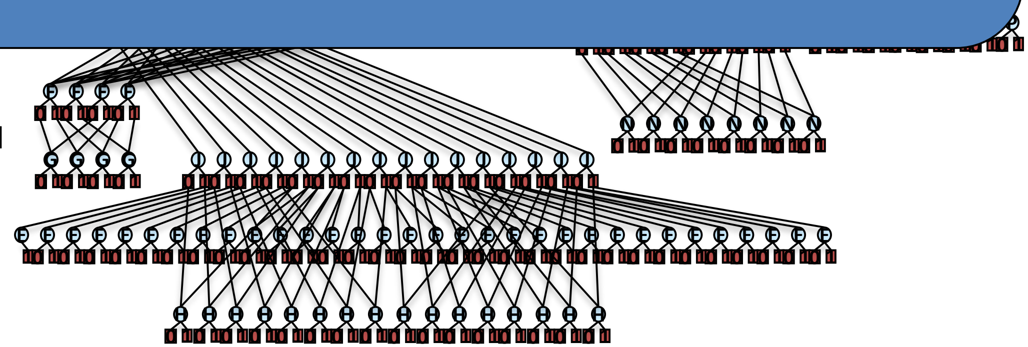
(Kjaerulff90)



- ***Optimization***
  - ***Choose pseudo-tree with a minimal search graph***
  - But determinism is unpredictable
  - ***And pruning by BnB is even more unpredictable***

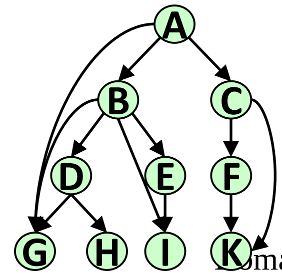
W=5,h=6

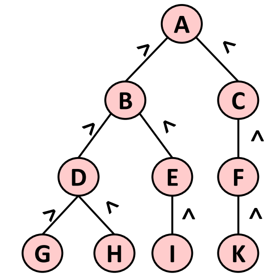




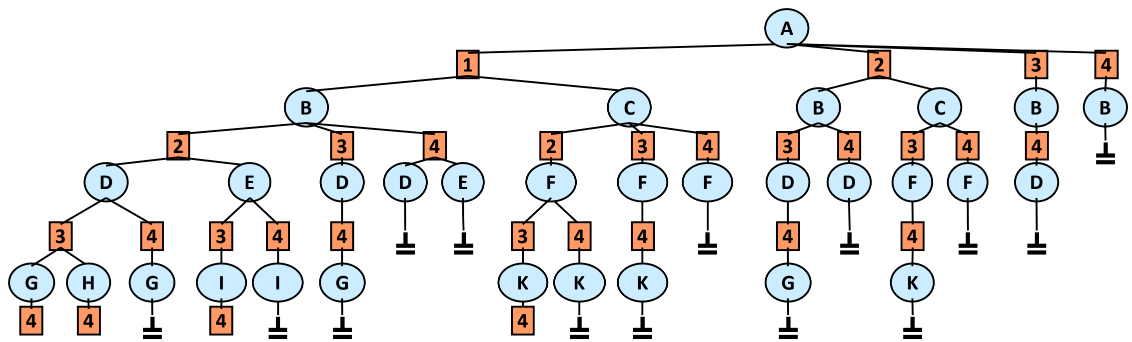
(C D K B A O M L N P J H E F G)

The Effect of Constraint Propagation

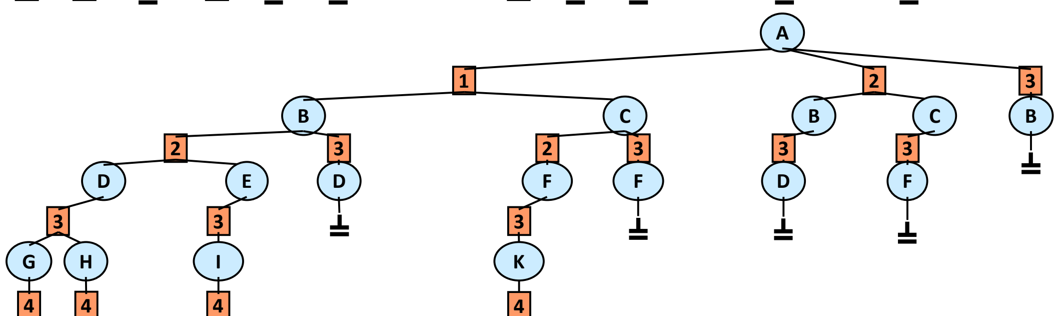




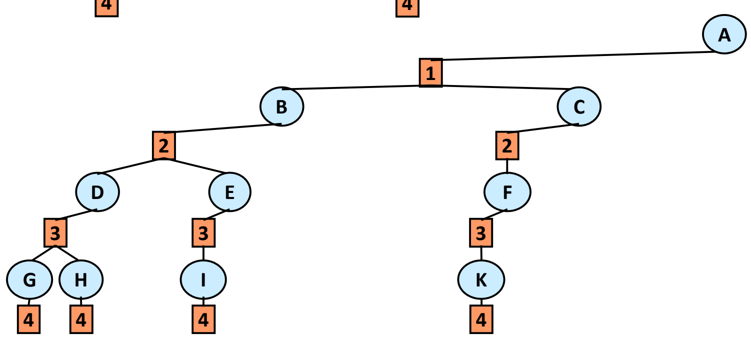
Domains are {1,2,3,4}



CONSTRAINTS ONLY



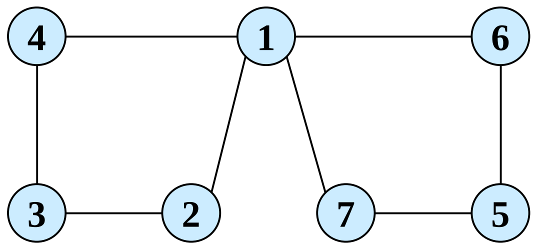
FORWARD CHECKING



MAINTAINING ARC CONSISTENCY

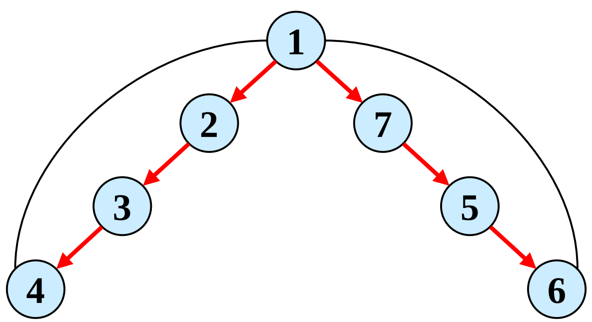
Pseudo-Trees

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

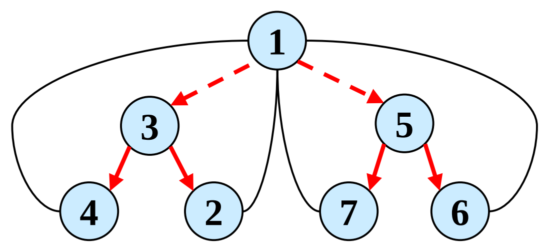


(a) Graph

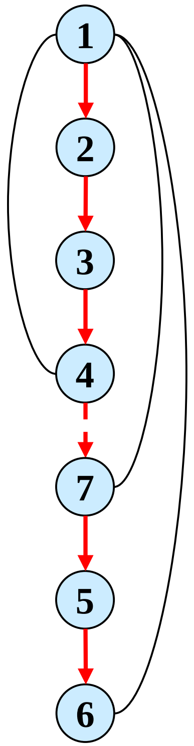
$m <= w^* \log n$



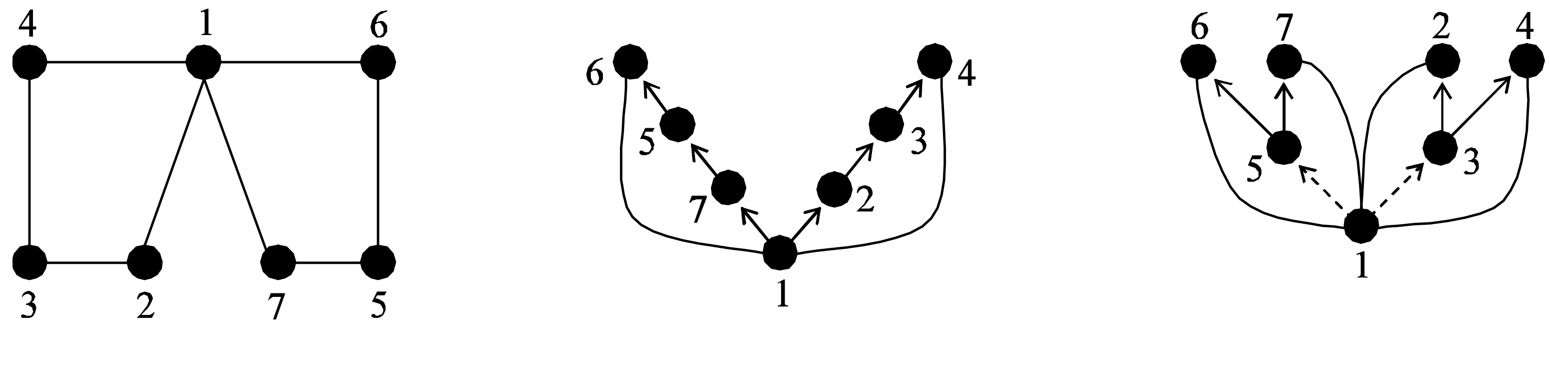
(b) DFS tree
depth=3



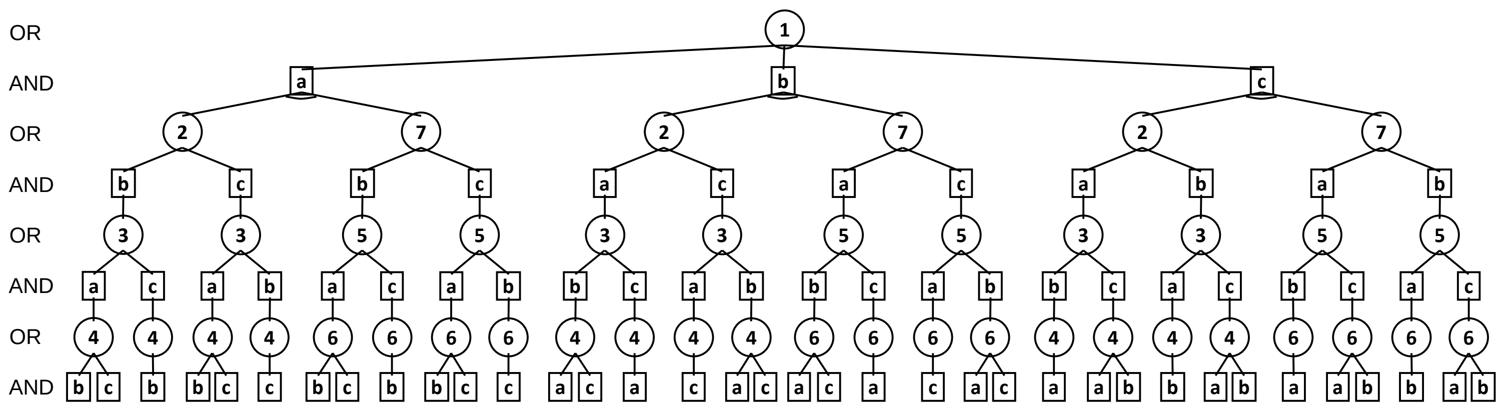
(c) pseudo- tree
depth=2

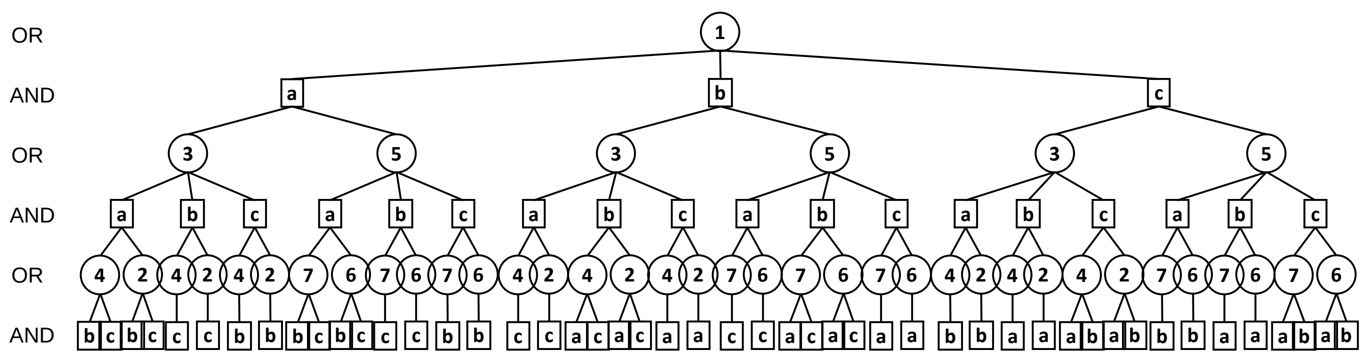


(d) Chain
depth=6



(a) (b) (c)





Search:
AND/OR Branch and Bound
Best-first search

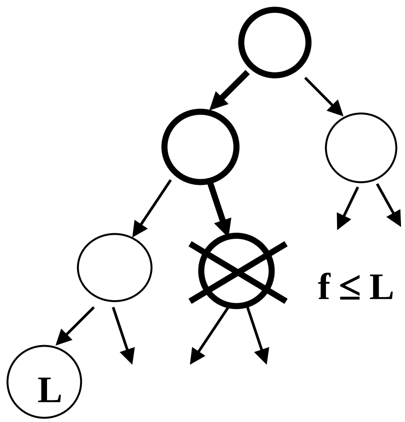
Basic Heuristic Search Schemes

Heuristic function **f(x^p)** computes a lower bound on the best extension of **x^p** and can be used to guide a heuristic search algorithm. We focus on:

1. Branch-and-Bound

Use heuristic function **f(x^p)** to prune the depth-first search tree

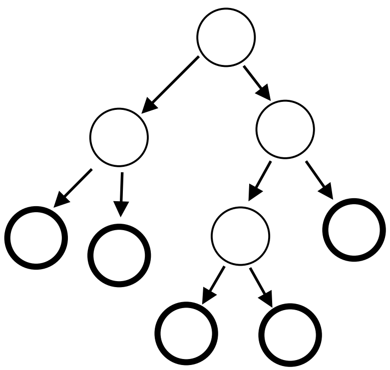
Linear space



2. Best-First Search

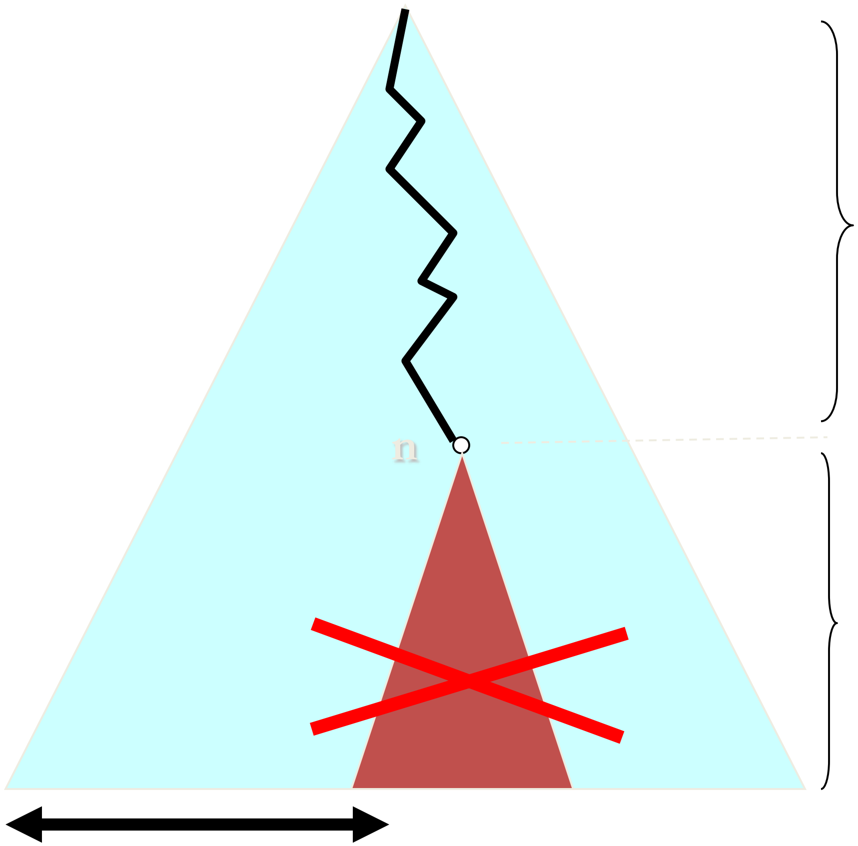
Always expand the node with the highest heuristic value **f(x^p)**

Needs lots of memory



Classic Branch-and-Bound

Each node is a COP subproblem
(defined by current conditioning)



g(n)

f(n) = g(n) + h(n)

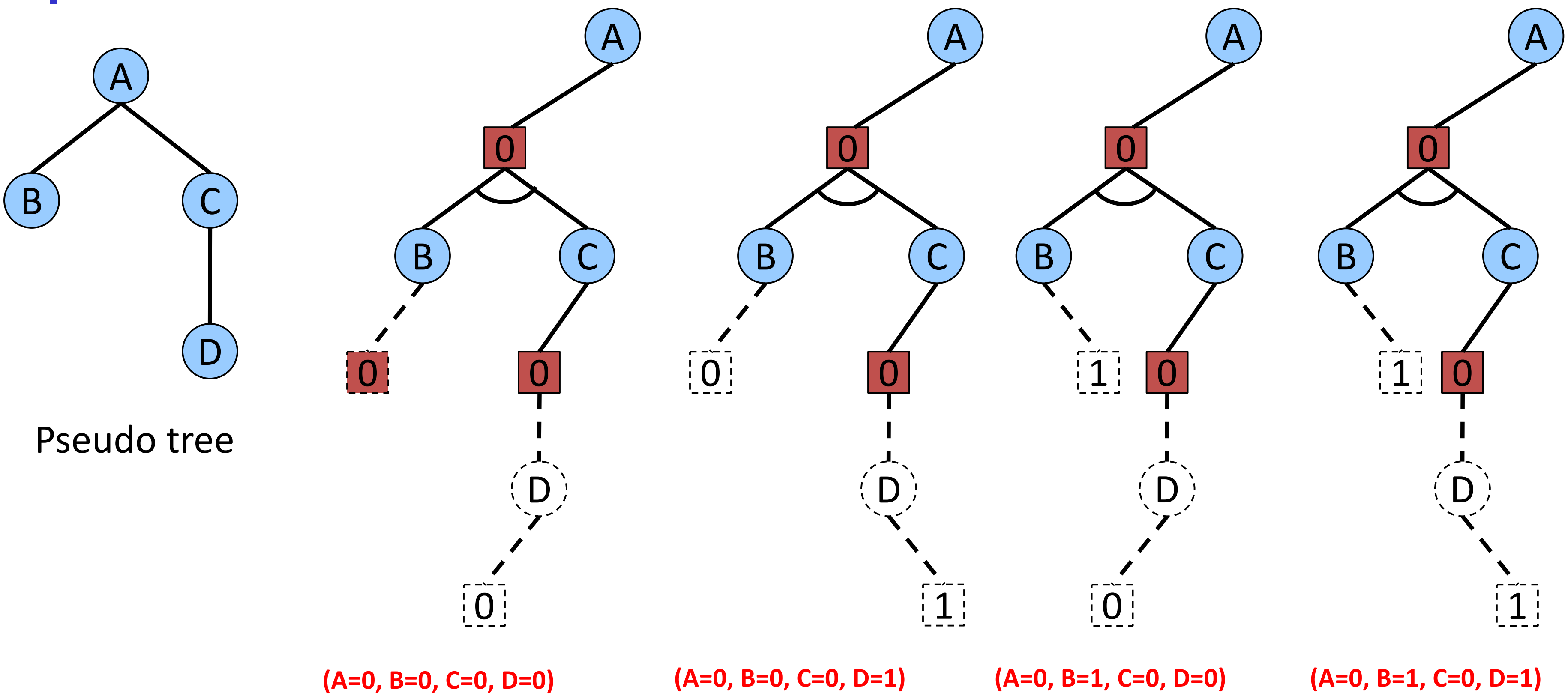
f(n) = lower bound

Prune if f(n) ≥ UB

h(n) - under-estimates
Optimal cost below n

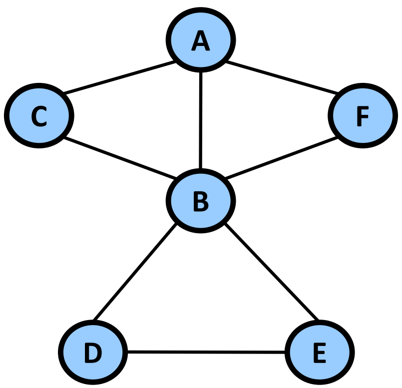
(UB) Upper Bound = best solution so far

Partial Solution Tree for AND/OR



Extension(T') – solution trees that extend T'

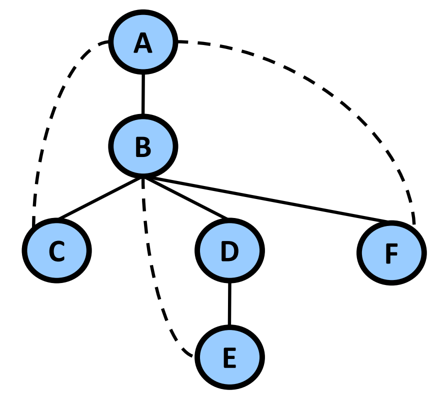
Exact Evaluation Function

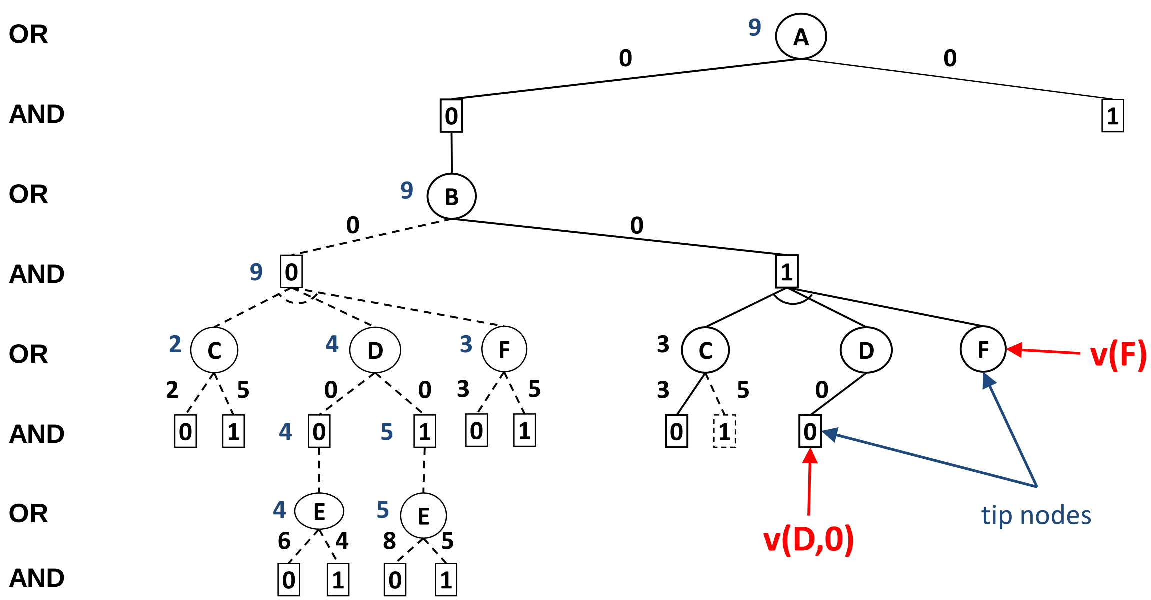


A	B	C	f_1(ABC)
0	0	0	2
0	0	1	5
0	1	0	3
0	1	1	5
1	0	0	9
1	0	1	3
1	1	0	7
1	1	1	2

A	B	F	f_2(ABF)
0	0	0	3
0	0	1	5
0	1	0	1
0	1	1	4
1	0	0	6
1	0	1	5
1	1	0	6
1	1	1	5

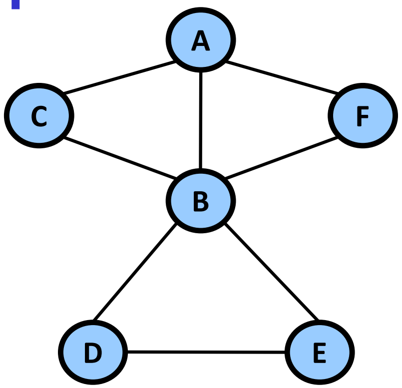
B	D	E	f_3(BDE)
0	0	0	6
0	0	1	4
0	1	0	8
0	1	1	5
1	0	0	9
1	0	1	3
1	1	0	7
1	1	1	4





f*(T') = w(A,0) + w(B,1) + w(C,0) + w(D,0) + v(D,0) + v(F)

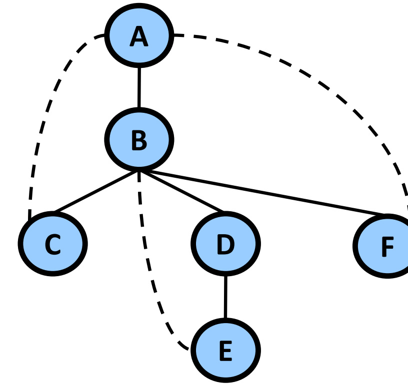
Heuristic Evaluation Function

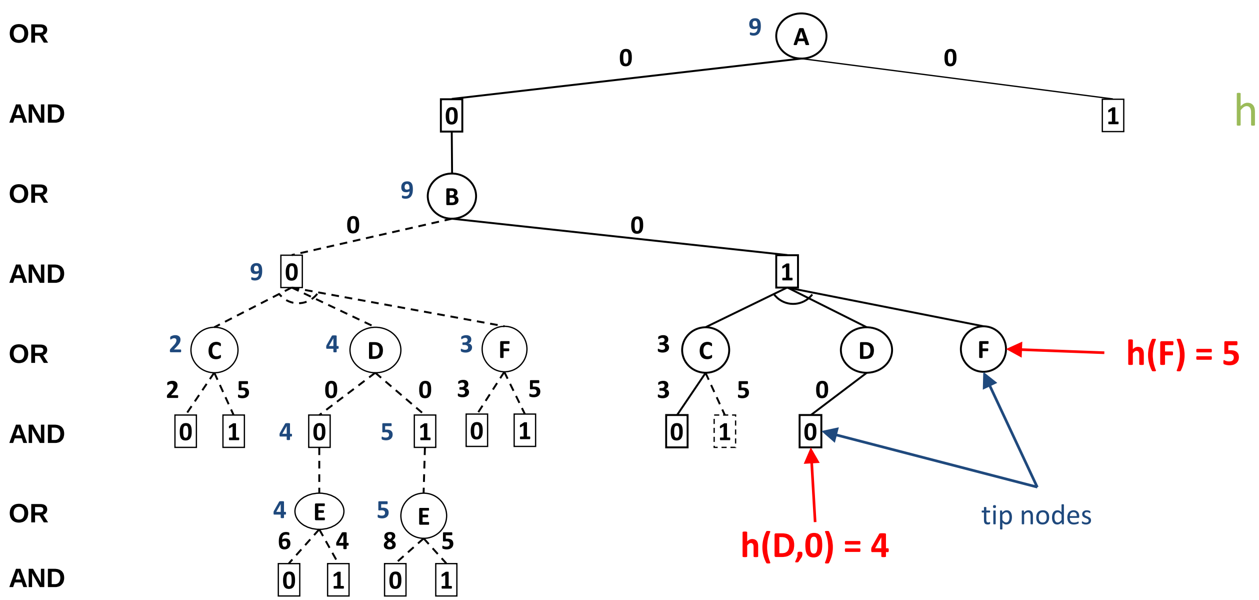


A	B	C	$f_1(ABC)$
0	0	0	2
0	0	1	5
0	1	0	3
0	1	1	5
1	0	0	9
1	0	1	3
1	1	0	7
1	1	1	2

A	B	F	$f_2(ABF)$
0	0	0	3
0	0	1	5
0	1	0	1
0	1	1	4
1	0	0	6
1	0	1	5
1	1	0	6
1	1	1	5

B	D	E	$f_3(BDE)$
0	0	0	6
0	0	1	4
0	1	0	8
0	1	1	5
1	0	0	9
1	0	1	3
1	1	0	7
1	1	1	4

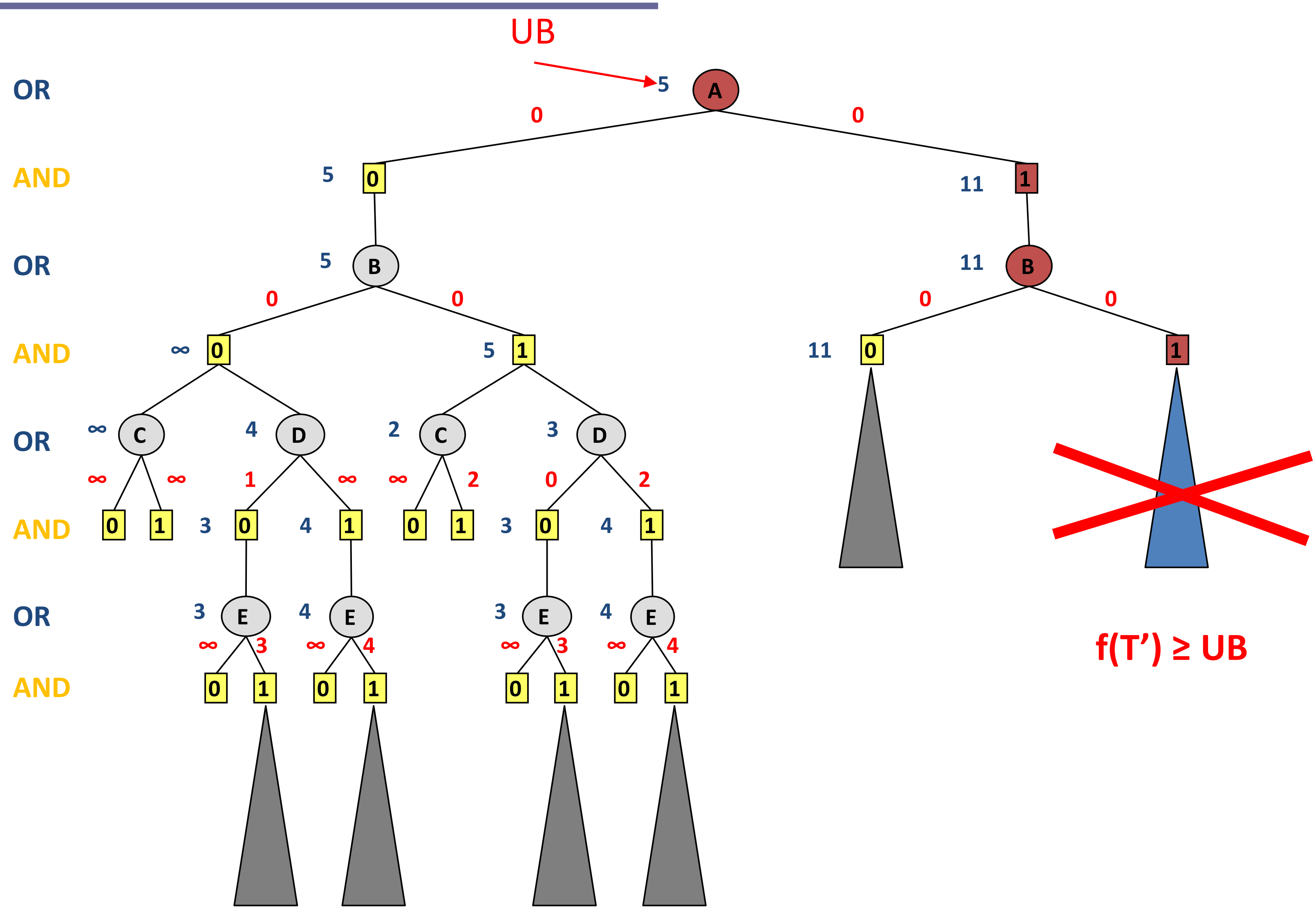




$h(n) \leq v(n)$

$f(T') = w(A,0) + w(B,1) + w(C,0) + w(D,0) + h(D,0) + h(F) = 12 \leq f^*(T')$

AND/OR Branch-and-Bound Search



AND/OR Branch-and-Bound Search (AOBB)

(Marinescu & Dechter, IJCAI'05, AIJ 2009)

- Associate each node n with a heuristic lower bound h(n) on v(n)

- EXPAND (top-down)
  - Evaluate f(T') and prune search if f(T') ≥ UB
  - Expand the tip node n

- PROPAGATE (bottom-up)
  - Update value of the parent p of n
    - OR nodes: minimization
    - AND nodes: summation

Best-First vs. Depth-first Branch-and-Bound

- **Best-First (A*): (optimal)**
  - Expand least number of nodes given h
  - Requires to store all search tree

- **Depth-first Branch-and-Bound:**
  - Can use only linear space
  - If find an optimal solution early will expand the same space as Best-First (if search space is a tree)
  - B&B can improve heuristic function dynamically

Best-First AND/OR Search (AOBF)

(Marinescu & Dechter, CPAIOR'07, AAAI'07, UAI'07)

- **Maintains the set of best partial solution trees**
- **Top-down Step (EXPAND)**
  - Traces down marked connectors from root
    - i.e., best partial solution tree
  - Expands a tip node **n** by generating its successors **n'**
  - Associate each successor with heuristic estimate **h(n')**
    - Initialize **v(n') = h(n')**

- **Bottom-up Step (REVISE)**
  - Updates node values **v(n)**
    - OR nodes: **minimization**
    - AND nodes: **summation**
  - Marks the most promising solution tree from the root
  - Label the nodes as SOLVED:
    - OR is SOLVED if marked child is SOLVED
    - AND is SOLVED if all children are SOLVED
- **Terminate when root node is SOLVED**

(specializes Nilsson's AO* to solving COP) (Nilsson, 1984)

AOBF versus AOBB

- **AOBF** with the same heuristic as **AOBB** is likely to expand the smallest search space

- **AOBB** improves its heuristic function dynamically, whereas **AOBF** uses only **h(n)**

- **AOBB** can use far less memory by avoiding for example dead-caches, whereas **AOBF** keeps in memory the explicated search graph

- **AOBB** is any-time, whereas **AOBF** is not

Second Part:

Bounding Schemes Approximations

Outline

- **Introduction**
  - Graphical models
  - Optimization tasks for graphical models
  - Birds view of techniques
- **Inference**
  - Variable Elimination, Bucket Elimination
- **Search**
  - AND/OR search spaces
  - Depth-First Branch-and-Bound and Best-First Search
- **Lower-bounds and relaxations**
  - Bounded variable elimination
  - Iterative cost shifting and local consistency
  - Using bounds as heuristic functions
- **Advanced tasks for optimization**
  - Marginal Map for Conformant planning
  - M-best search
  - Influence diagrams
- **Software**

The mini-bucket scheme

Mini-Bucket Approximation

Split a bucket into mini-buckets => bound complexity

bucket (X) =

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

$$h^X = \min_X \sum_{i=1}^{n} h_i$$

{ $h_1, \ldots, h_r$ }

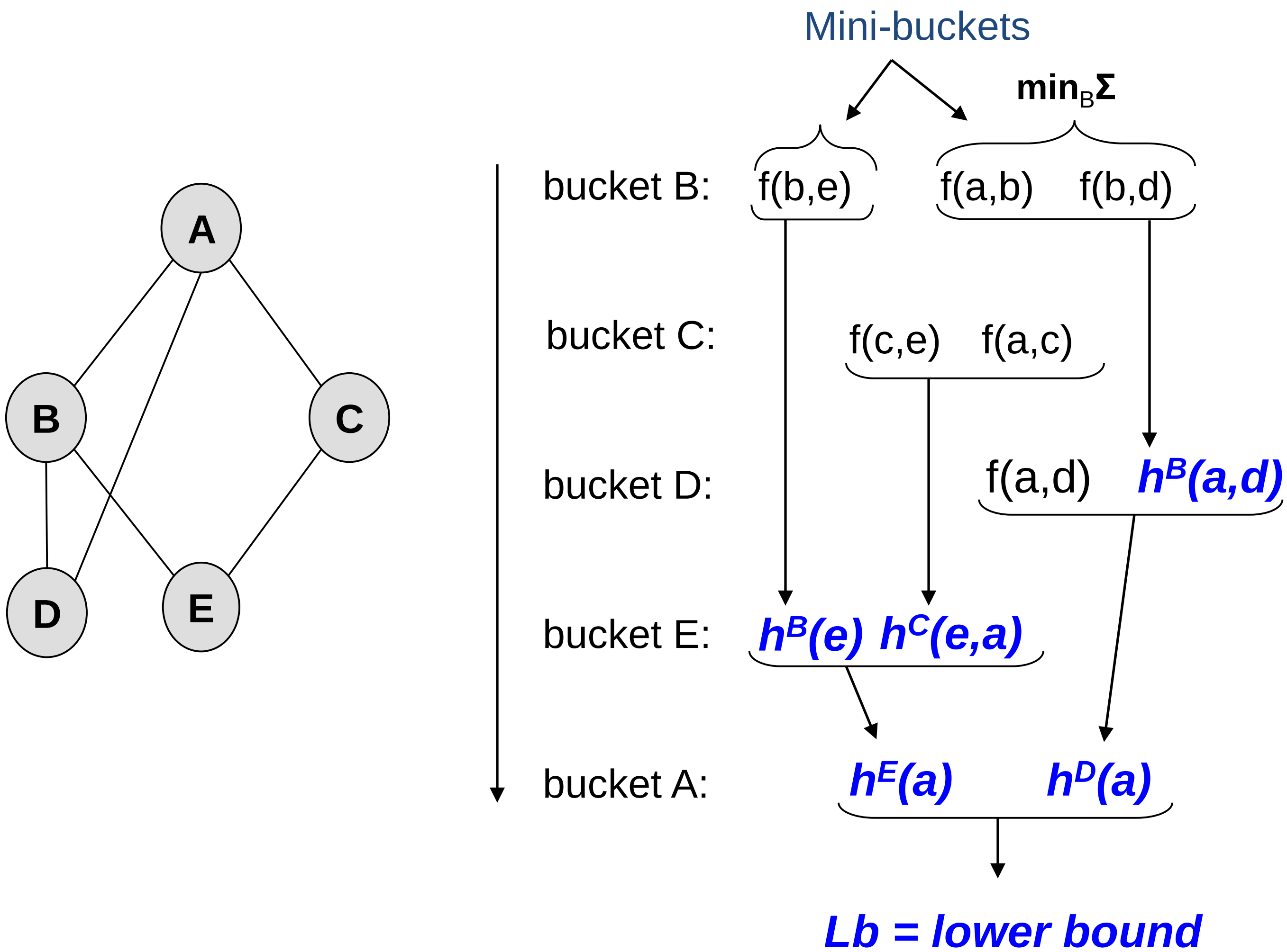
{ $h_{r+1}, \ldots, h_n$ }

$$g^X = \left(\min_X \sum_{i=1}^{r} h_i\right) + \left(\min_X \sum_{i=r+1}^{n} h_i\right)$$

$$g^X \le h^X$$

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

Mini-Bucket Elimination

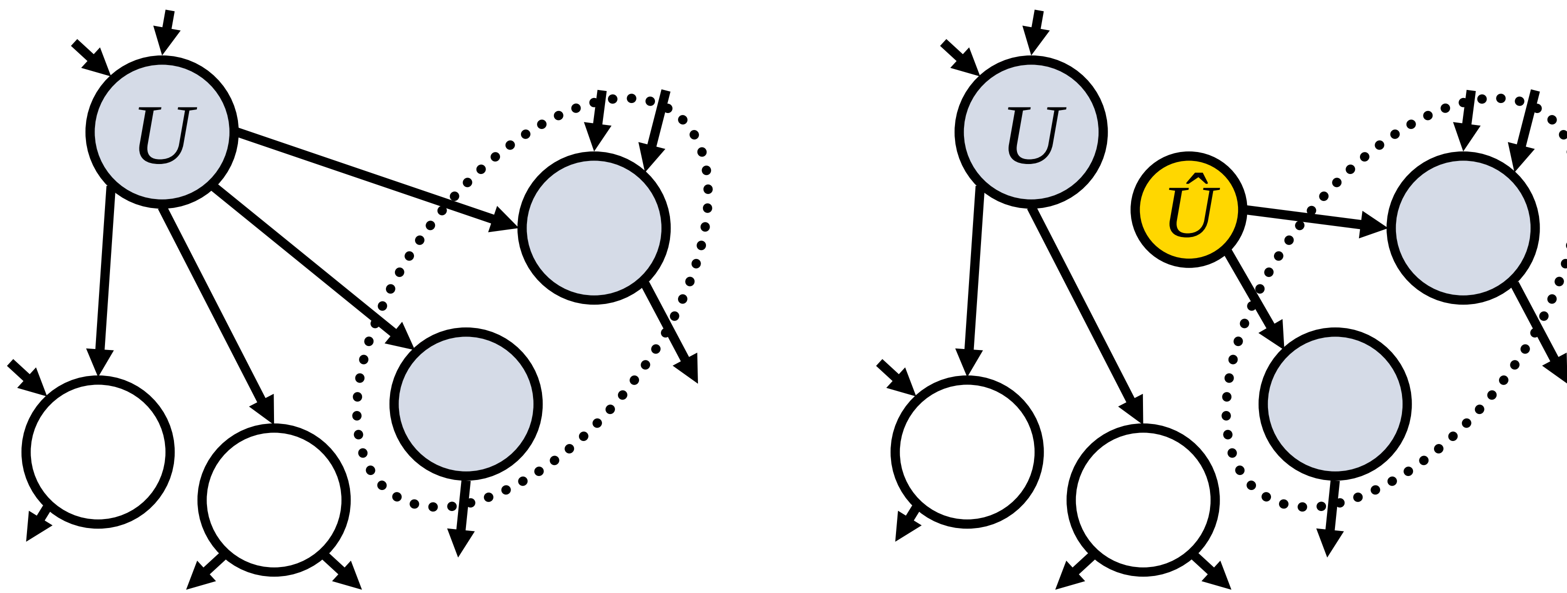


Semantics of Mini-Bucket: Splitting a Node

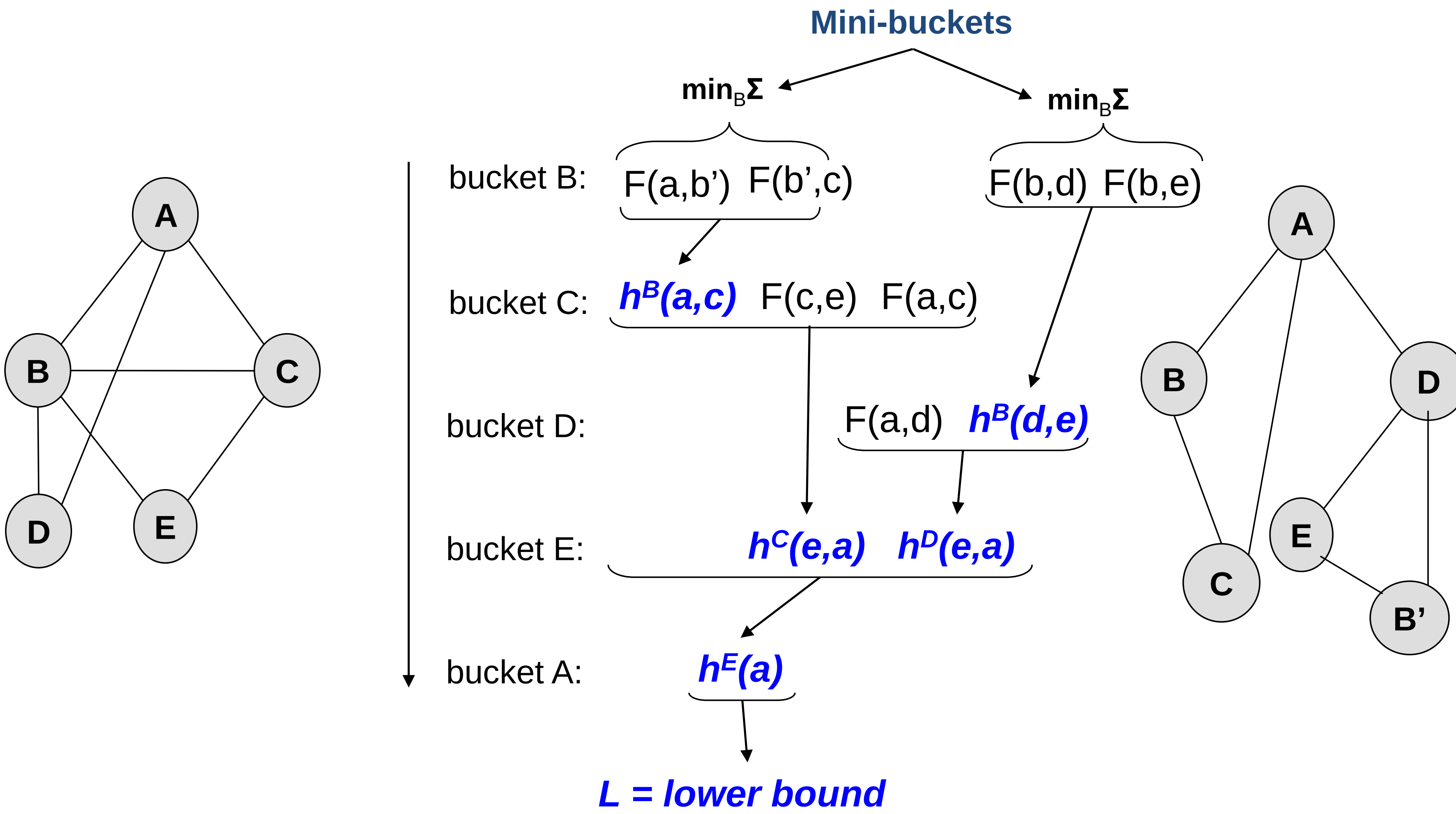
Variables in different buckets are renamed and duplicated
(Kask *et. al.*, 2001), (Geffner *et. al.*, 2007), (Choi, Chavira, Darwiche , 2007)

Before Splitting:
Network *N*

After Splitting:
Network *N'*



Mini-Bucket Elimination semantic



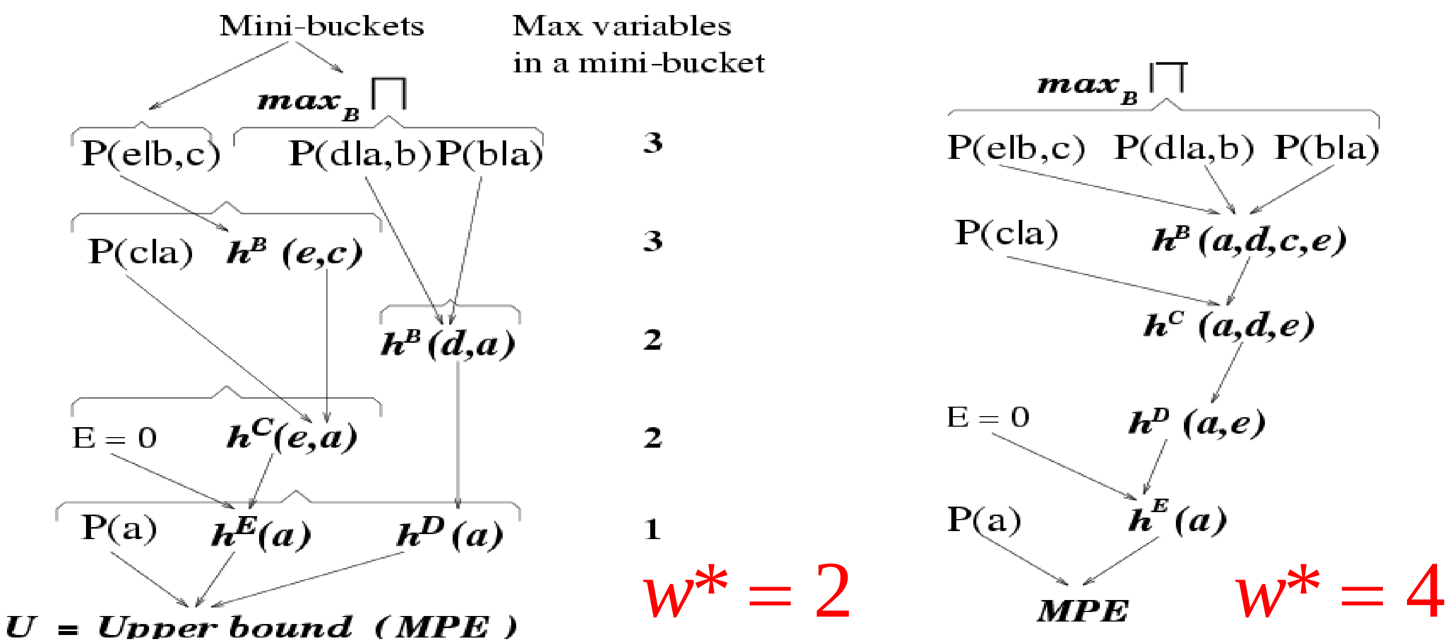
79

MBE-MPE(i)

Algorithm **Approx-MPE** (Dechter & Rish, 1997)

- **Input**: i – max number of variables allowed in a mini-bucket
- **Output**: [lower bound (P of a sub-optimal solution), upper bound]

Example: **approx-mpe(3)** versus **elim-mpe**



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 & Rish, 1997)

Anytime Approximation

anytime - mpe(ε)

Initialize : $i = i_0$

While time and space resources are available

$i \leftarrow i + i_{step}$

$U \leftarrow$ upper bound computed by *approx - mpe(i)*

$L \leftarrow$ lower bound computed by *approx - mpe(i)*

keep the best solution found so far

if $1 \le \frac{U}{L} \le 1 + \varepsilon$, return solution

end

return the largest L and the smallest U

Empirical Evaluation

(Rish & Dechter, 1999)

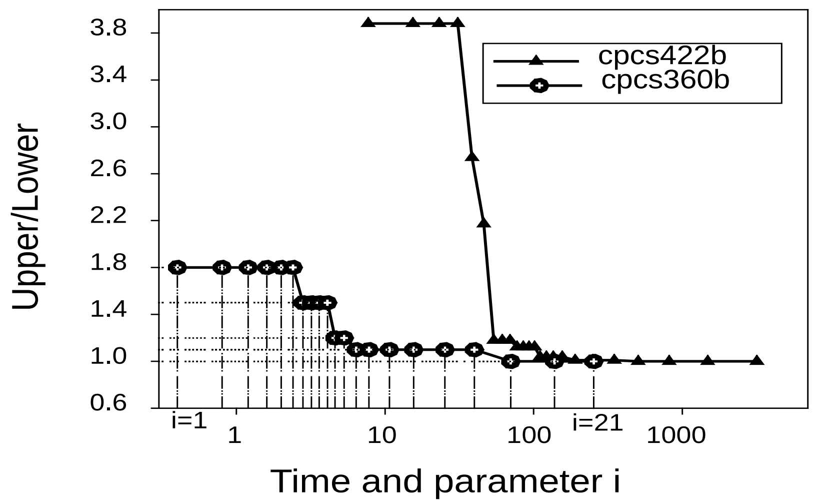
- **Benchmarks**
  - Randomly generated networks
  - CPCS networks
  - Probabilistic decoding

- **Task**
  - Comparing **approx-mpe** and **anytime-mpe** versus bucket-elimination (**elim-mpe**)

CPCS networks – medical diagnosis (noisy-OR model)

Test case: no evidence

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

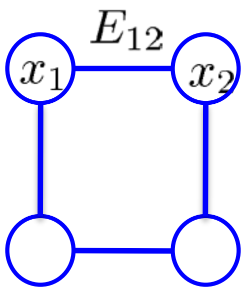


Time (sec)

Algorithm	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

Iterative cost-shifting and local consistency

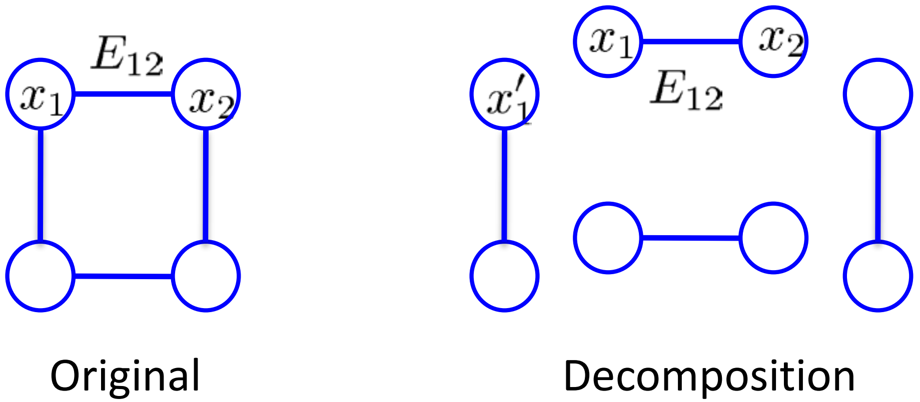
Tightening bounds via cost-shifting



Original

$$\max_{\underline{\mathrm{x}}} \sum_{ij} E_{ij}(x_i, x_j)$$

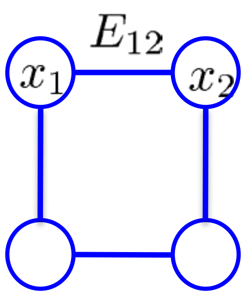
Tightening bounds via cost-shifting



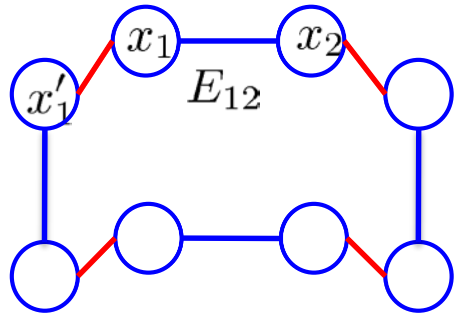
$$\max_{\underline{\mathrm{x}}} \sum_{ij} E_{ij}(x_i, x_j) \leq \sum_{ij} \max_{\underline{\mathrm{x}}} E_{ij}(x_i, x_j)$$

- Decompose graph into smaller subproblems
- Solve each independently; optimistic bound
- Exact if all copies agree

Decomposition view



Original



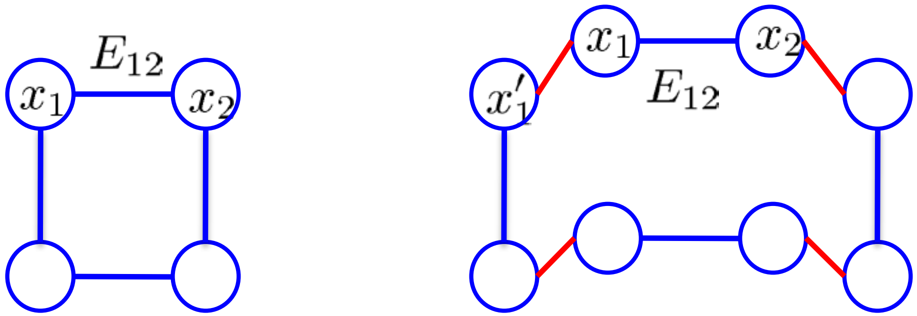
Decomposition

$$\forall i \sum_j \lambda_{ij}(x_i) = 0$$

$$\max_{\underline{X}} \sum_{ij} E_{ij}(x_i, x_j) \leq \min_{\lambda} \sum_{ij} \max_{\underline{X}} E_{ij}(x_i, x_j) + \lambda_{ij}(x_i) + \lambda_{ji}(x_j)$$

- Decompose graph into smaller subproblems
- Solve each independently; optimistic bound
- Exact if all copies agree
- Enforce lost equality constraints via Lagrange multipliers

Decomposition view



Original　　　　Decomposition

$$\forall i \sum_j \lambda_{ij}(x_i) = 0$$

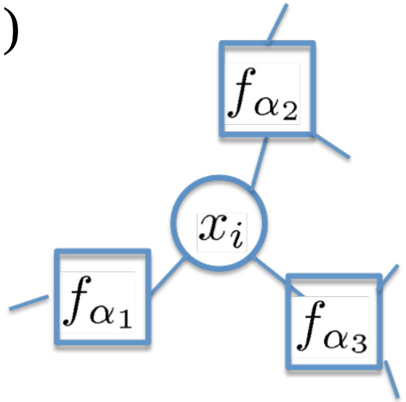
$$\max_{\underline{\mathrm{x}}} \sum_{ij} E_{ij}(x_i, x_j) \le \min_{\lambda} \sum_{ij} \max_{\underline{\mathrm{x}}} E_{ij}(x_i, x_j) + \lambda_{ij}(x_i) + \lambda_{ji}(x_j)$$

Same bound by different names

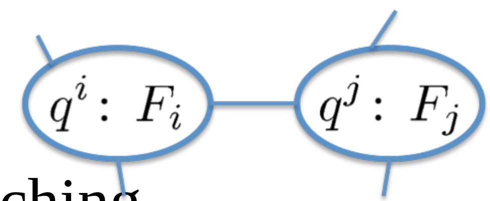
- Dual decomposition (Komodakis et al. 2007)
- TRW, MPLP (Wainwright et al. 2005; Globerson & Jaakkola 2007)
- Soft arc consistency (Cooper & Schiex 2004)

Various Update Schemes

- Can use any decomposition updates
  - (message passing, subgradient, augmented, etc.)
- FGLP: Update the original factors



- JGLP: Update clique function of the join graph

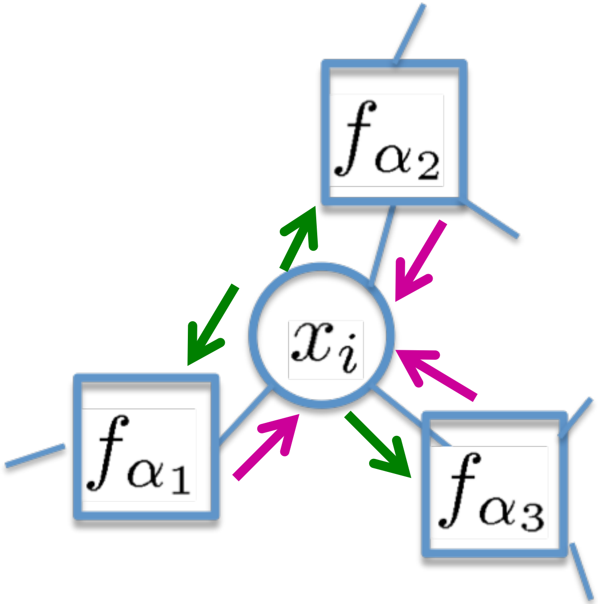


- **MBE-MM:** Mini-bucket with moment matching
  - Apply cost-shifting within each bucket only

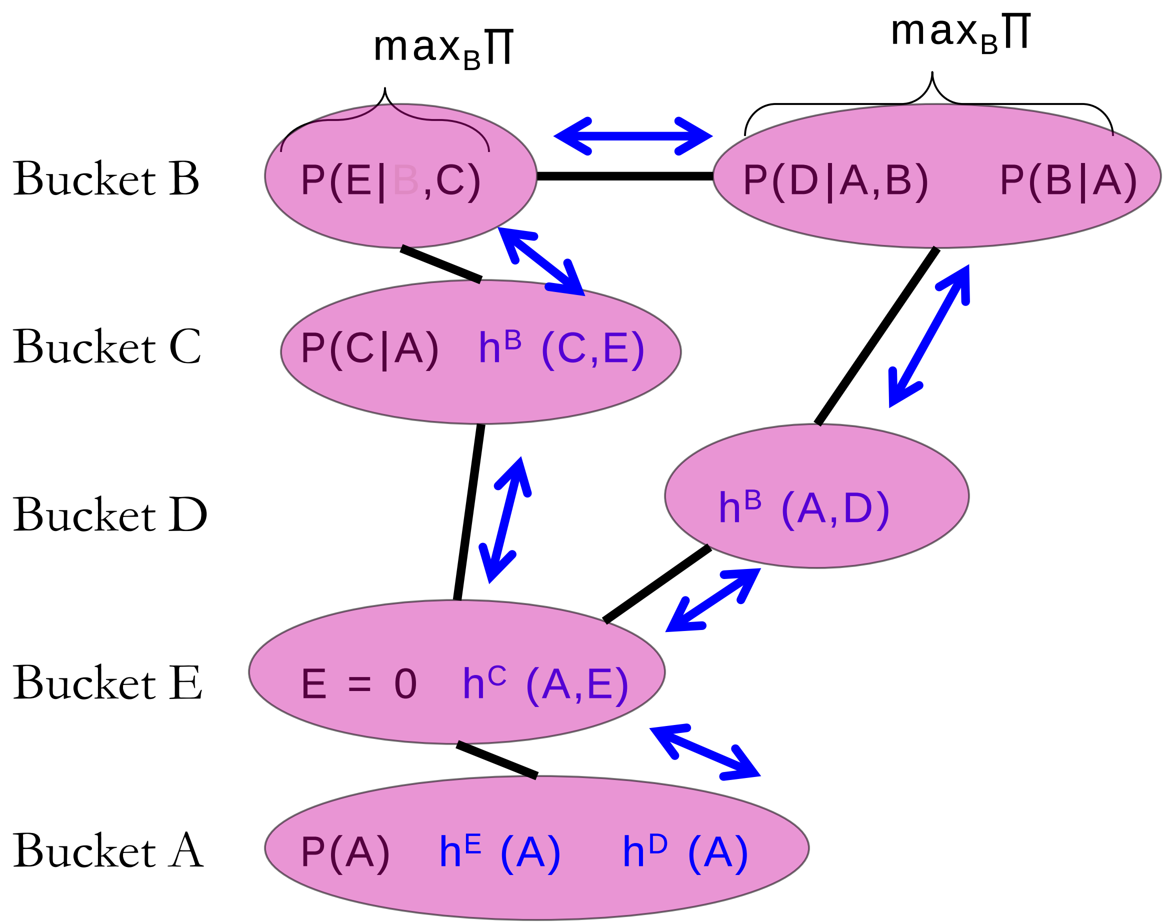
Factor graph Linear Programming

- Update the original factors (FGLP)
  - Tighten all factors over over x_i simultaneously
  - Compute max-marginals $\forall \alpha, \;\; \gamma_\alpha(x_i) = \max_{x_\alpha \setminus x_i} f_\alpha$
  - & update:

$$\forall \alpha, \;\; f_\alpha(x_\alpha) \leftarrow f_\alpha(x_\alpha) - \gamma_\alpha(x_i) + \frac{1}{|F_i|} \sum_\beta \gamma_\beta(x_i)$$



Join Graph Linear Programming (JGLP(i))



MB defines
A Join Graph

MBE-MM: MBE with moment matching

m_{11}, m_{12}- moment-matching messages

$\max_B \prod$ $\quad$ m_{11} $\quad$ $\max_B \prod$

Bucket B $\quad$ P(E|B,C) $\quad$ ⟷ $\quad$ P(B|A) $\quad$ P(D|A,B)

m_{12}

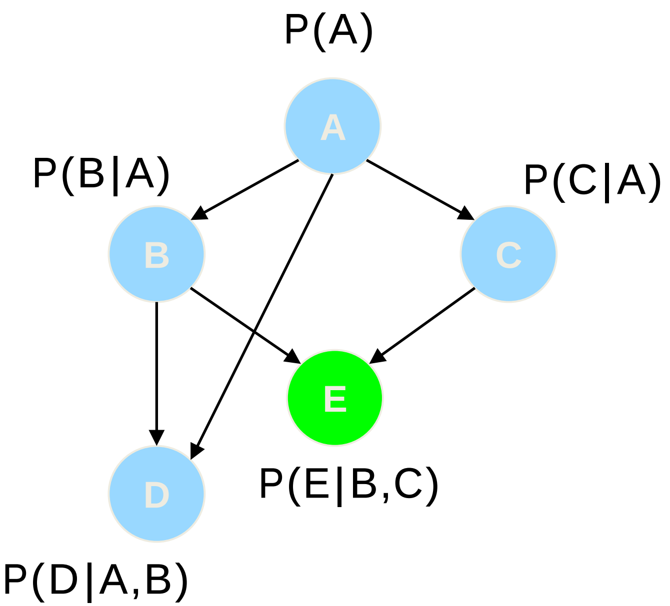
Bucket C $\quad$ P(C|A) $\quad$ h^B (C,E)

Bucket D $\quad$ h^B (A,D)

Bucket E $\quad$ E = 0 $\quad$ h^C (A,E)

Bucket A $\quad$ P(A) $\quad$ h^E (A) $\quad$ h^D (A)

W=2

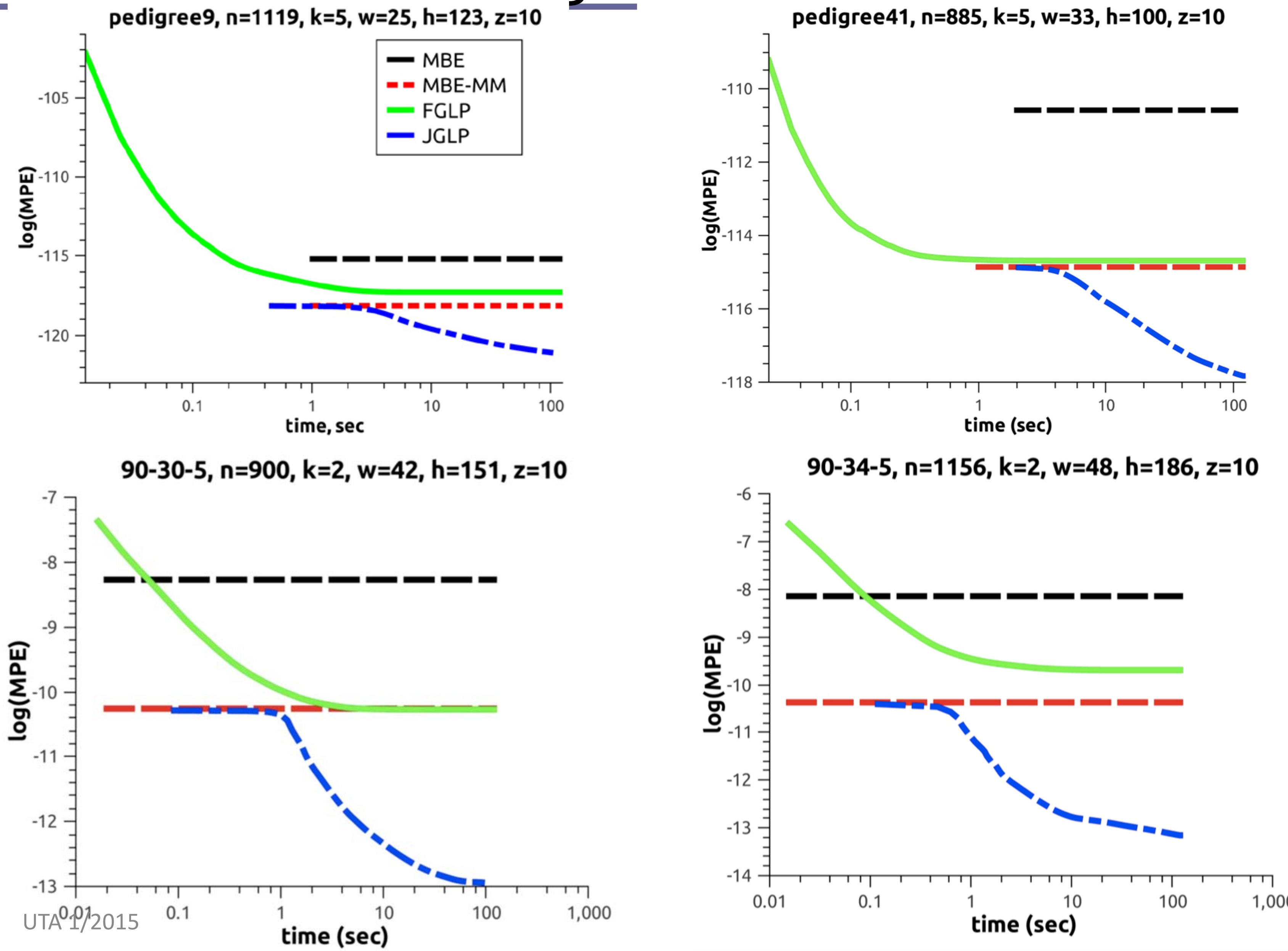


MPE* is an upper bound on MPE --U
Generating a solution yields a lower bound--L

Emprical evaluation/Benchmarks

Benchmark	# inst	n	k	w^*	h_T
Pedigrees	12	581-1006	3-7	16-39	52-104
Grids	32	144-2500	2	15-90	48-283
LargeFam	30	863-1400	17-58	33-111	
Type4	10	3907-8186	5-5	21-32	319-625
WCSP	56	25-1057	2-100	5-287	11-337

Iterative tightening as bounding schemes



Outline

- **Introduction**
  - Graphical models
  - Optimization tasks for graphical models
  - Birds view of techniques
- **Inference**
  - Variable Elimination, Bucket Elimination
- **Search**
  - AND/OR search spaces
  - Depth-First Branch-and-Bound and Best-First Search
- **Lower-bounds and relaxations**
  - Bounded variable elimination
  - Iterative cost shifting and local consistency
  - **Using the above as heuristics for search**
- **Advanced tasks for optimization**
  - Marginal Map for Conformant planning
  - M-best search
  - Influence diagrams
- **Software**

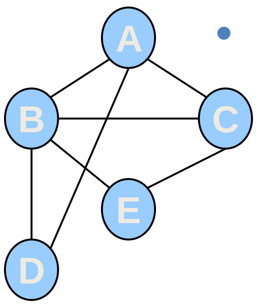
Converting Bounds to Heuristics

How to Generate Heuristics

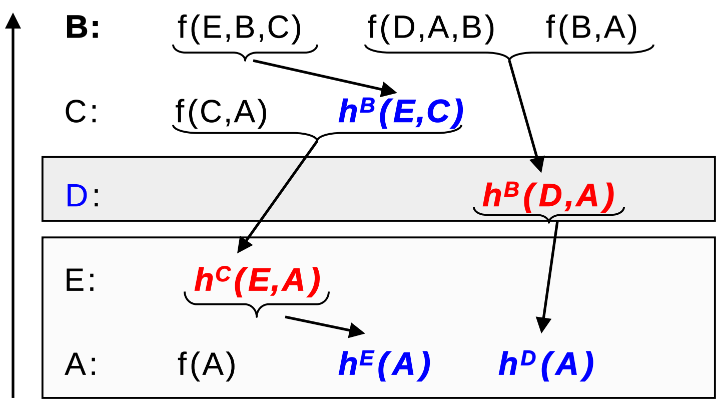
- The principle of relaxed models
  - Mini-Bucket Elimination
  - Re-paremetrization, moment matching
  - Bounded directional consistency ideas
  - Linear relaxation for integer programs

MBE Heuristics for BBsearch

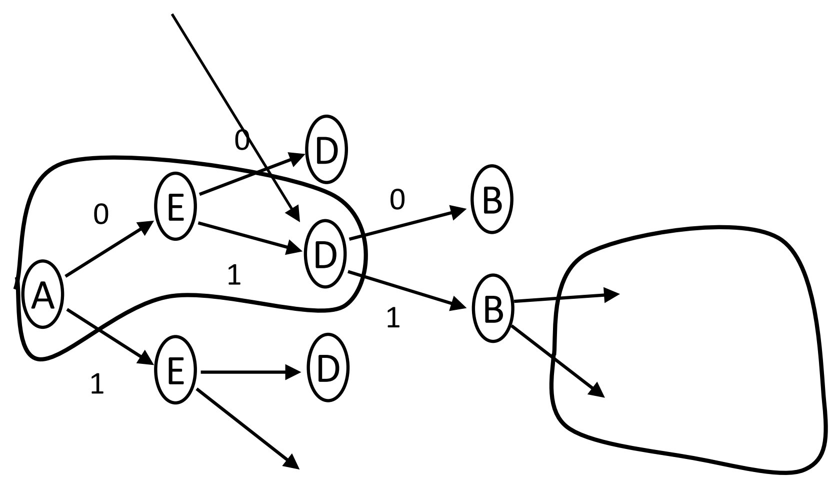
- Given a partial assignment $\mathbf{x^p}$, estimate the cost of the best extension to a full solution
- The evaluation function $\mathbf{f(x^p)}$ can be computed using function recorded by the Mini-Bucket scheme



Cost Network



f(a,e,D))=g(a,e) + H(a,e,D)



$$f(a,e,D) = \underbrace{f(a)}_{g} + \underbrace{h^B(D,a) + h^C(e,a)}_{h\text{ – is admissible}}$$

Heuristics Properties

- MBE Heuristic is monotone, admissible
- Computed in linear time

- **IMPORTANT**:
  - Heuristic strength can vary by MBE(i)
  - Higher i-bound $\Rightarrow$ more pre-processing $\Rightarrow$ stronger heuristic $\Rightarrow$ less search

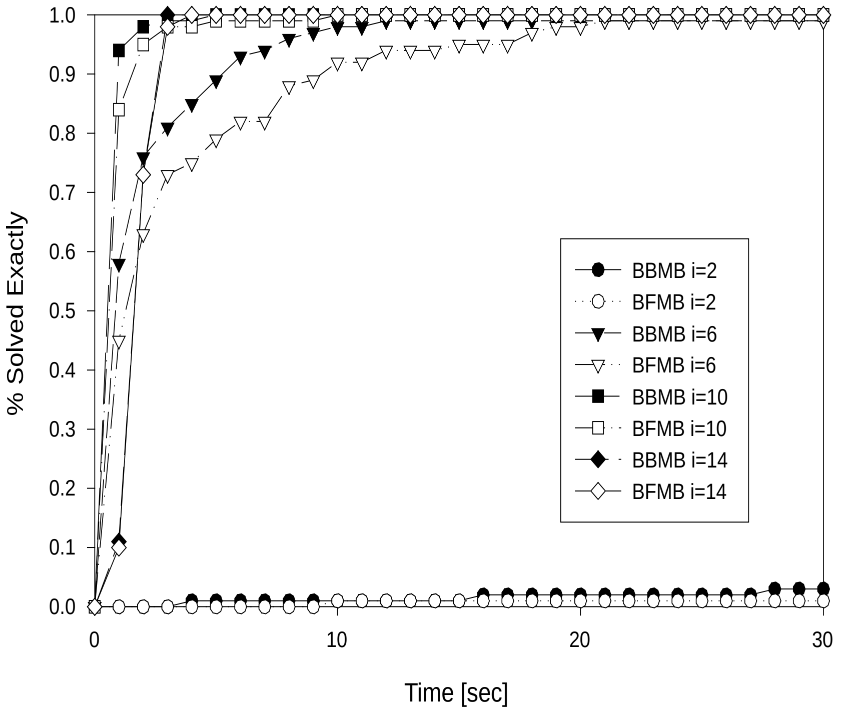
- Allows controlled trade-off between preprocessing and search

Experimental Methodology

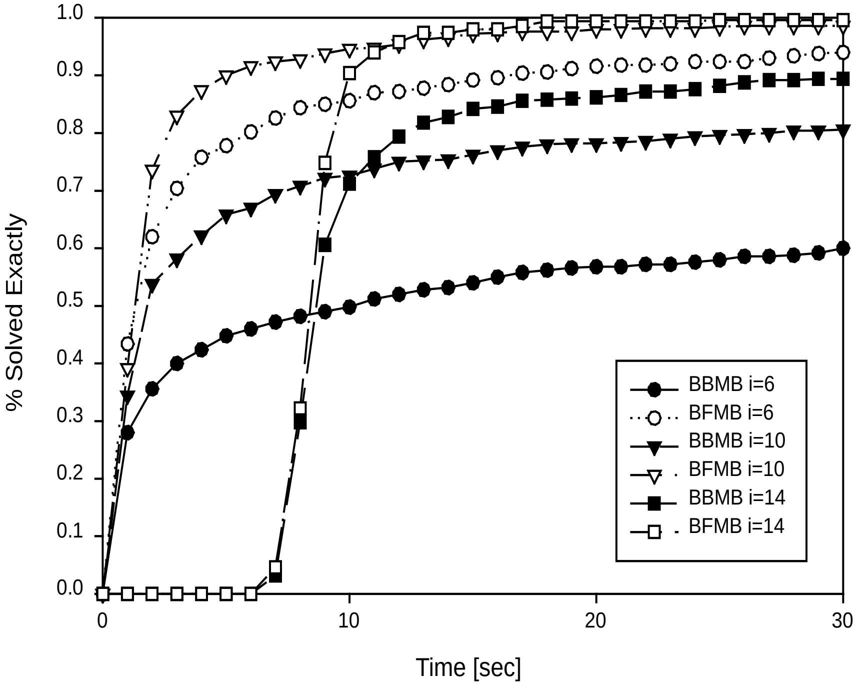
- **Algorithms**
  - BBMB(i) - Branch-and-Bound with MB(i)
  - BBFB(i) - Best-First with MB(i)
  - MBE(i) – Mini-Bucket Elimination
- **Benchmarks**
  - Random Coding (Bayesian)
  - CPCS (Bayesian)
  - Random (CSP)
- **Measures of performance**
  - Compare accuracy given a fixed amount of time
    - i.e., how close is the cost found to the optimal solution
  - Compare trade-off performance as a function of time

Empirical Evaluation of Mini-Bucket heuristics:
Random coding networks (Kask & Dechter, UAI'99, Aij 2000)

Random Coding, K=100, noise=0.28

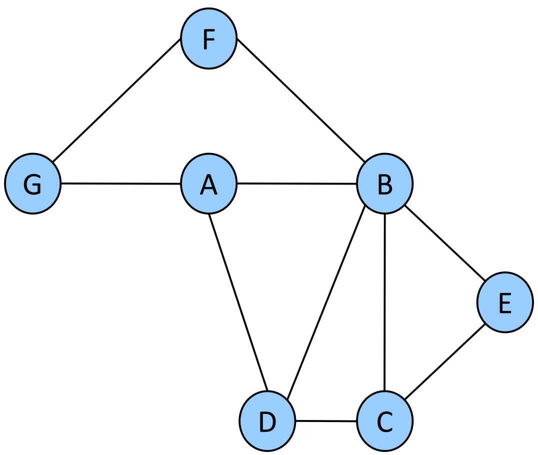


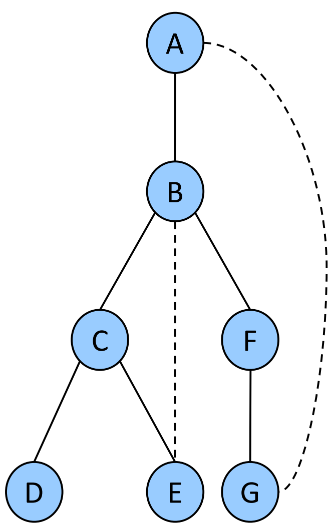
Random Coding, K=100, noise=0.32

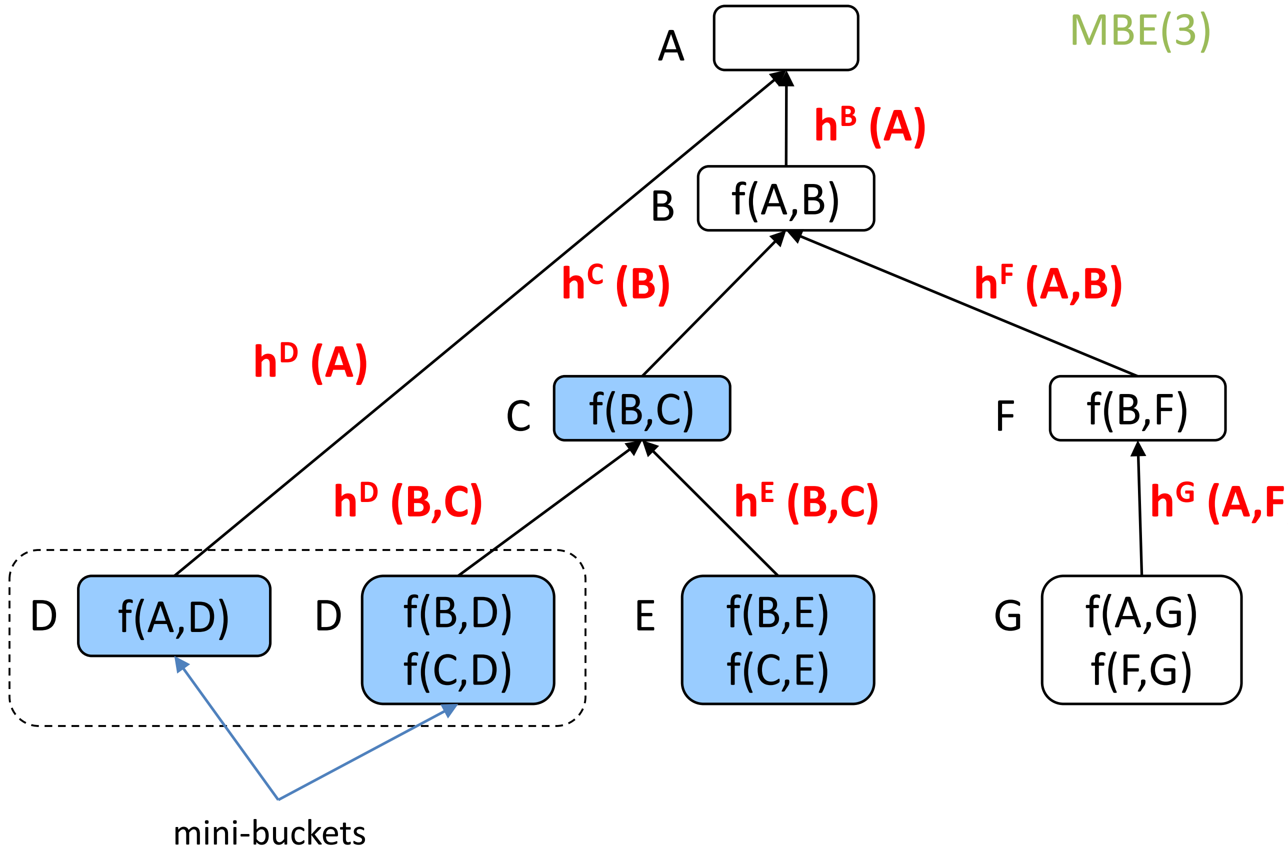


Each data point represents an average over 100 random instances

MBE Heuristics for AND/OR Search







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

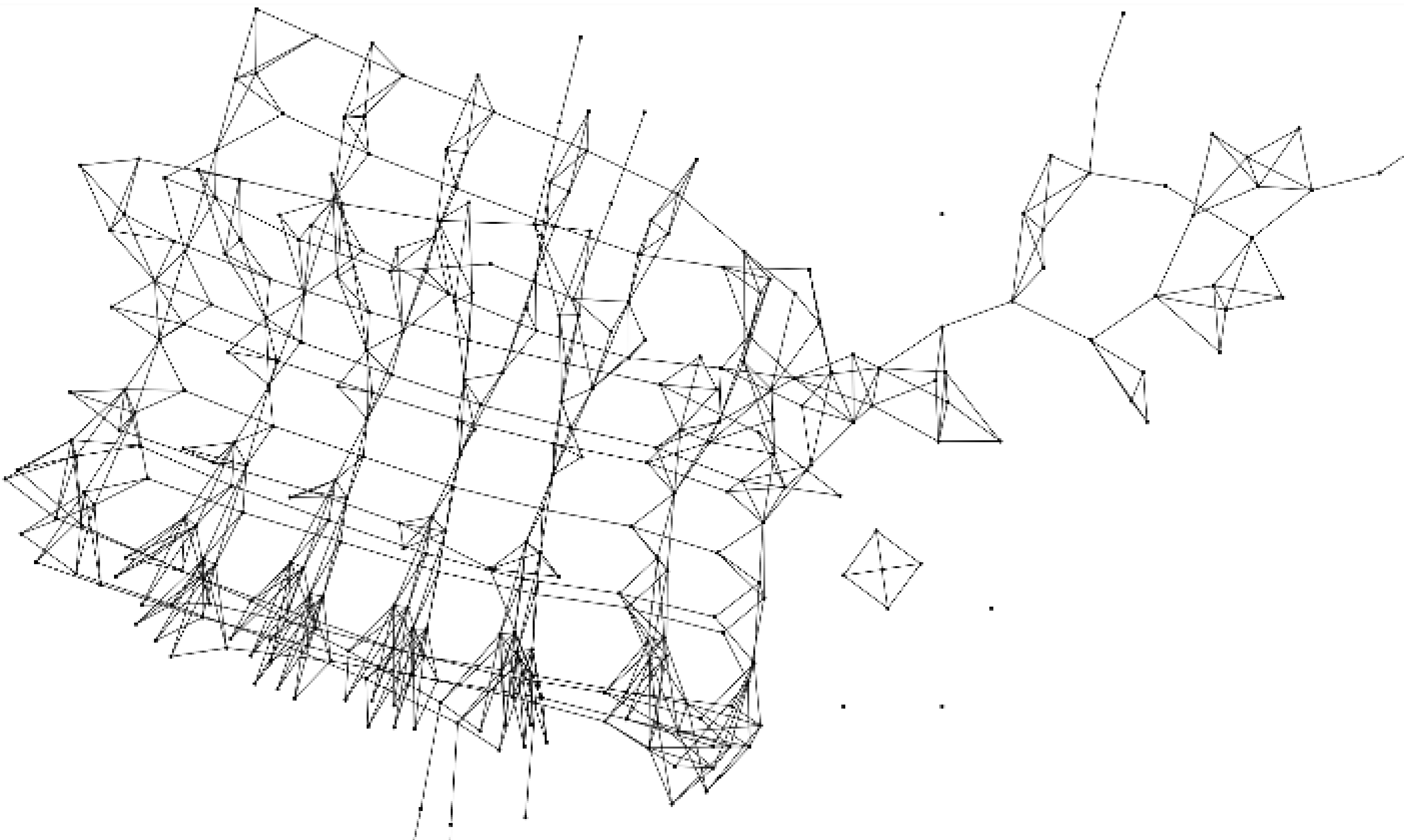
Ordering: (A, B, C, D, E, F, G)

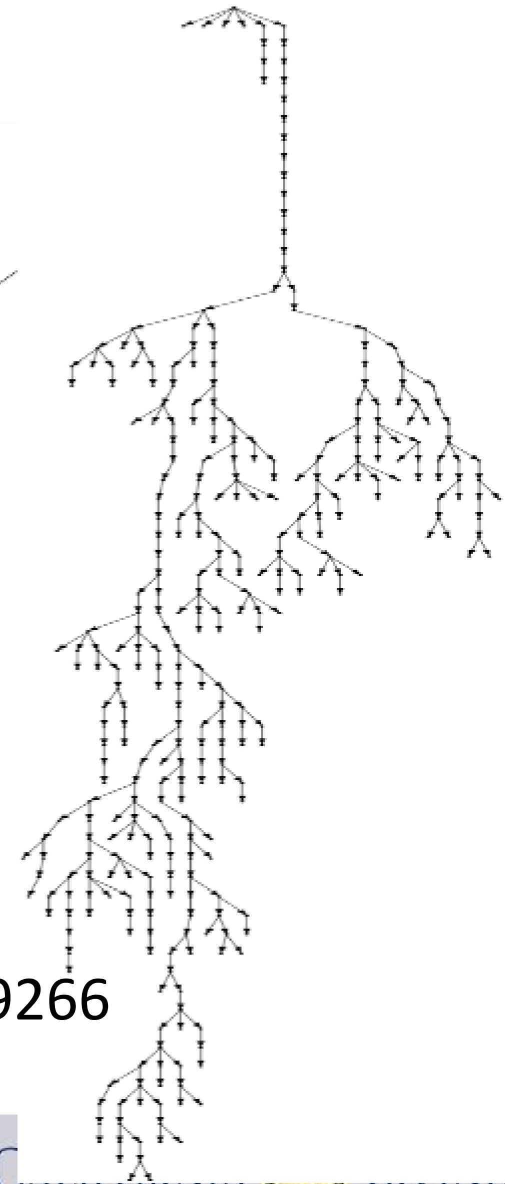
Empirical Evaluation:
In house
Pascal 2011 Competition

Emprical evaluation/Benchmarks

Benchmark	# inst	n	k	w^*	h_T
Pedigrees	12	581-1006	3-7	16-39	52-104
Grids	32	144-2500	2	15-90	48-283
LargeFam	30	863-1400	17-58	33-111	
Type4	10	3907-8186	5-5	21-32	319-625
WCSP	56	25-1057	2-100	5-287	11-337

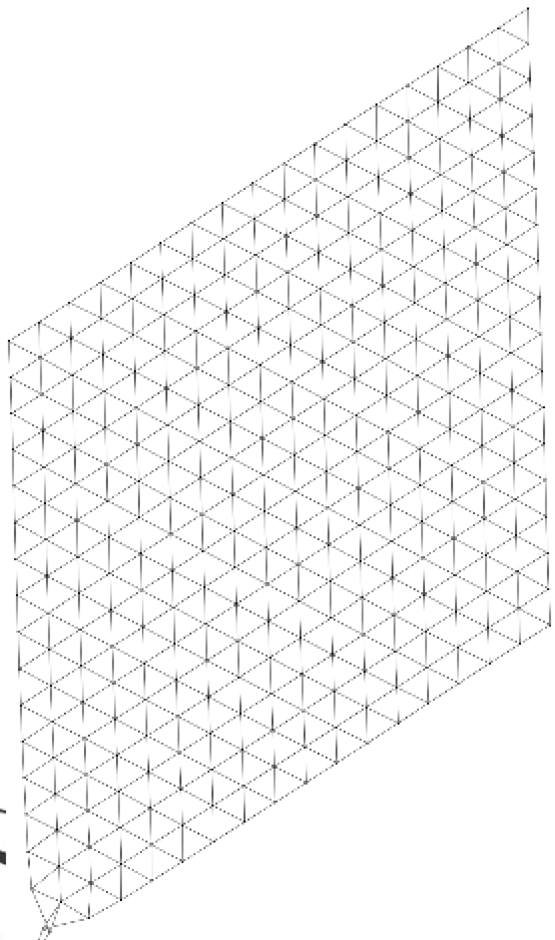
Pedigree1 (n=298, w=15, h=61, k=4)





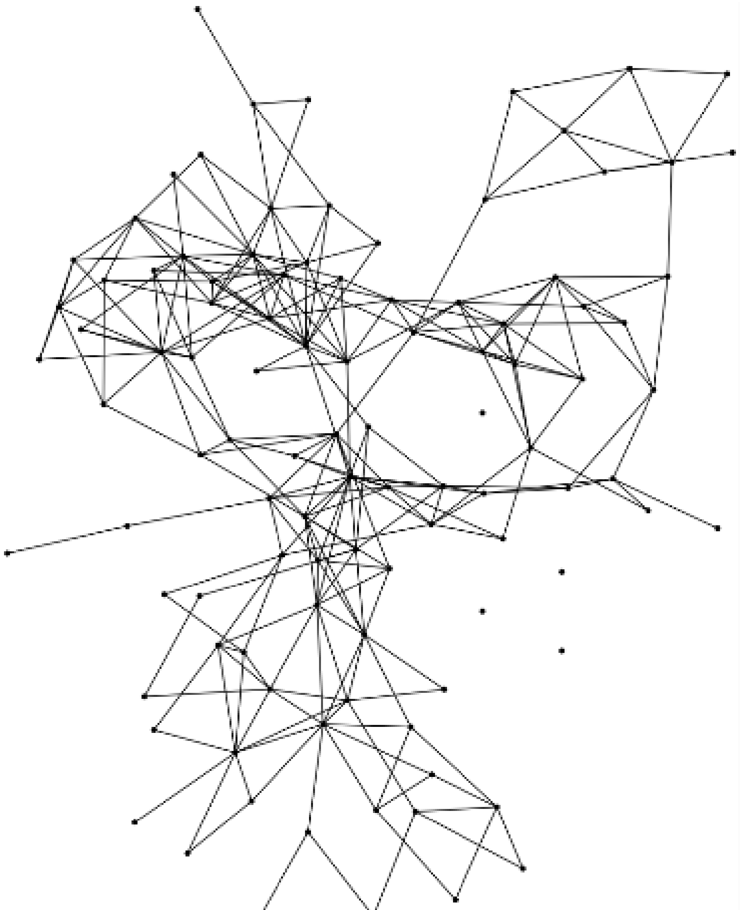
#OR/#AND : pedigree1: 3094892/8119266

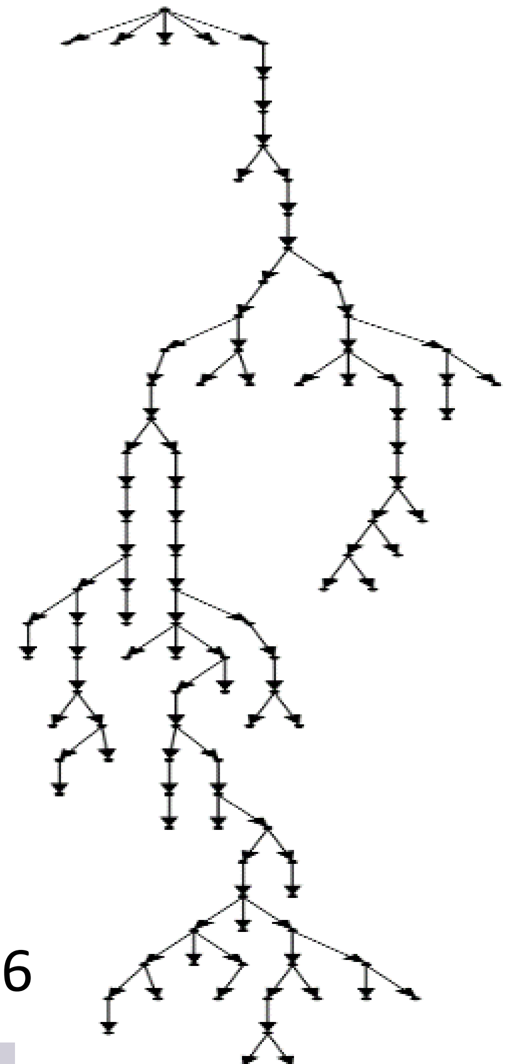
Grid 75-18-5 (n=324, w=24, h=85, k=2)



#OR/#AND : 99380327/198760654

pdb1dfx (n=103, w=8, h=30, k=81)





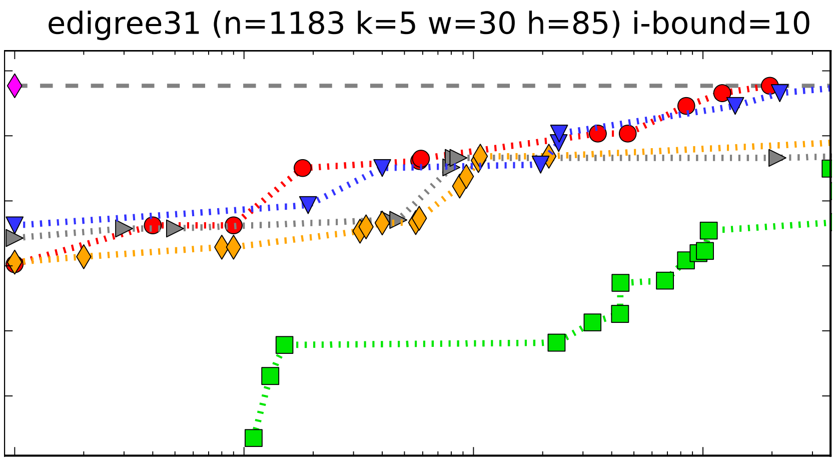
#OR/#AND: 50717208263/775835160296

Iterative tightening as Heuristic Generators

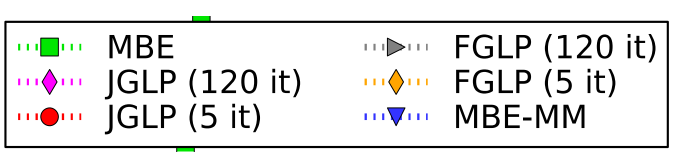
- 4 schemes used:
  - **AOBB-MBE**: AOBB guided by pure MBE heuristics
  - **AOBB-MBE+MM**: AOBB guided by MBE and max- marginal matching
  - **AOBB-FGLP+MBE**: AOBB with heuristics from FGLP followed by MBE
  - **AOBB-JGLP**: AOBB guided by JGLP-produced heuristics

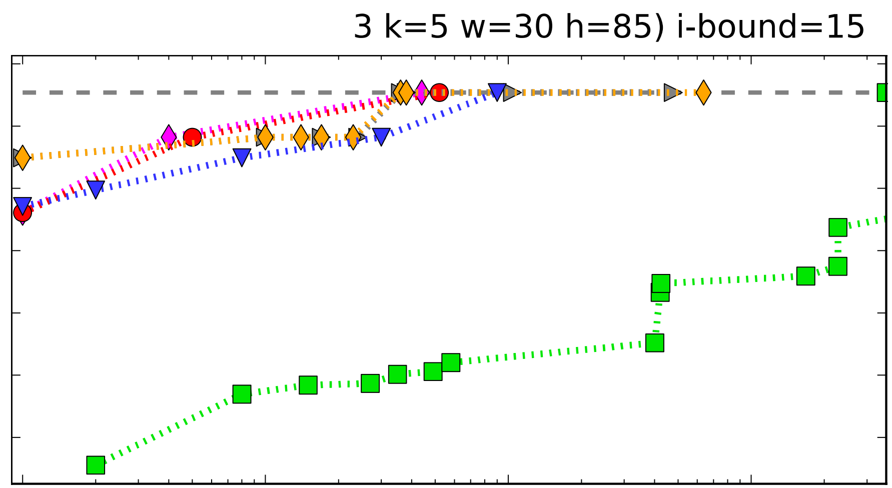
- FGLP, JGLP ran for 30 seconds
- Total search time bound 24 h
- Memory limit 3 Gb
- Mini-bucket z-bounds={10,15,20}

Empirical Evaluation: Haplotype Problems

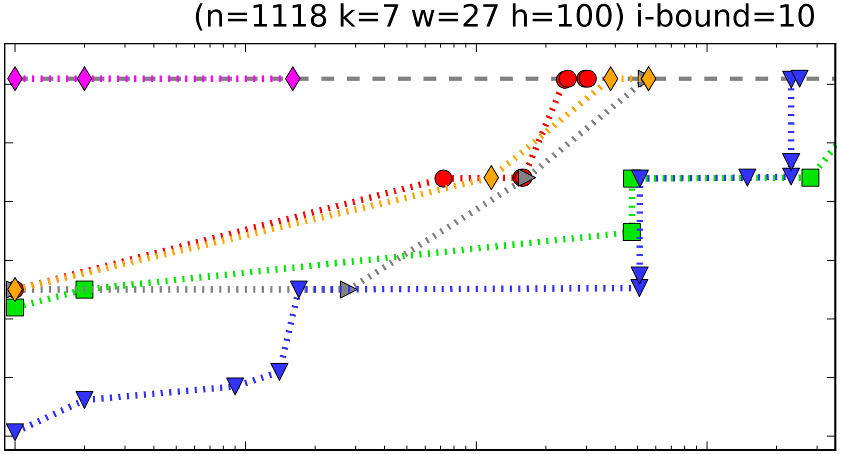


Time bound – 24 h

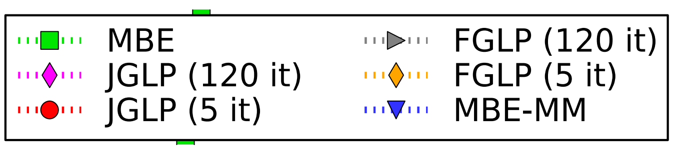


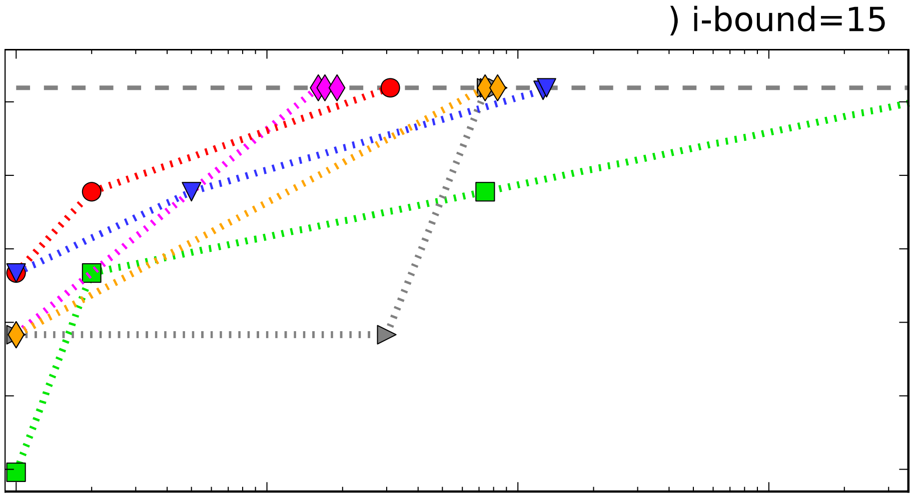


Empirical Evaluation: Haplotype Problems

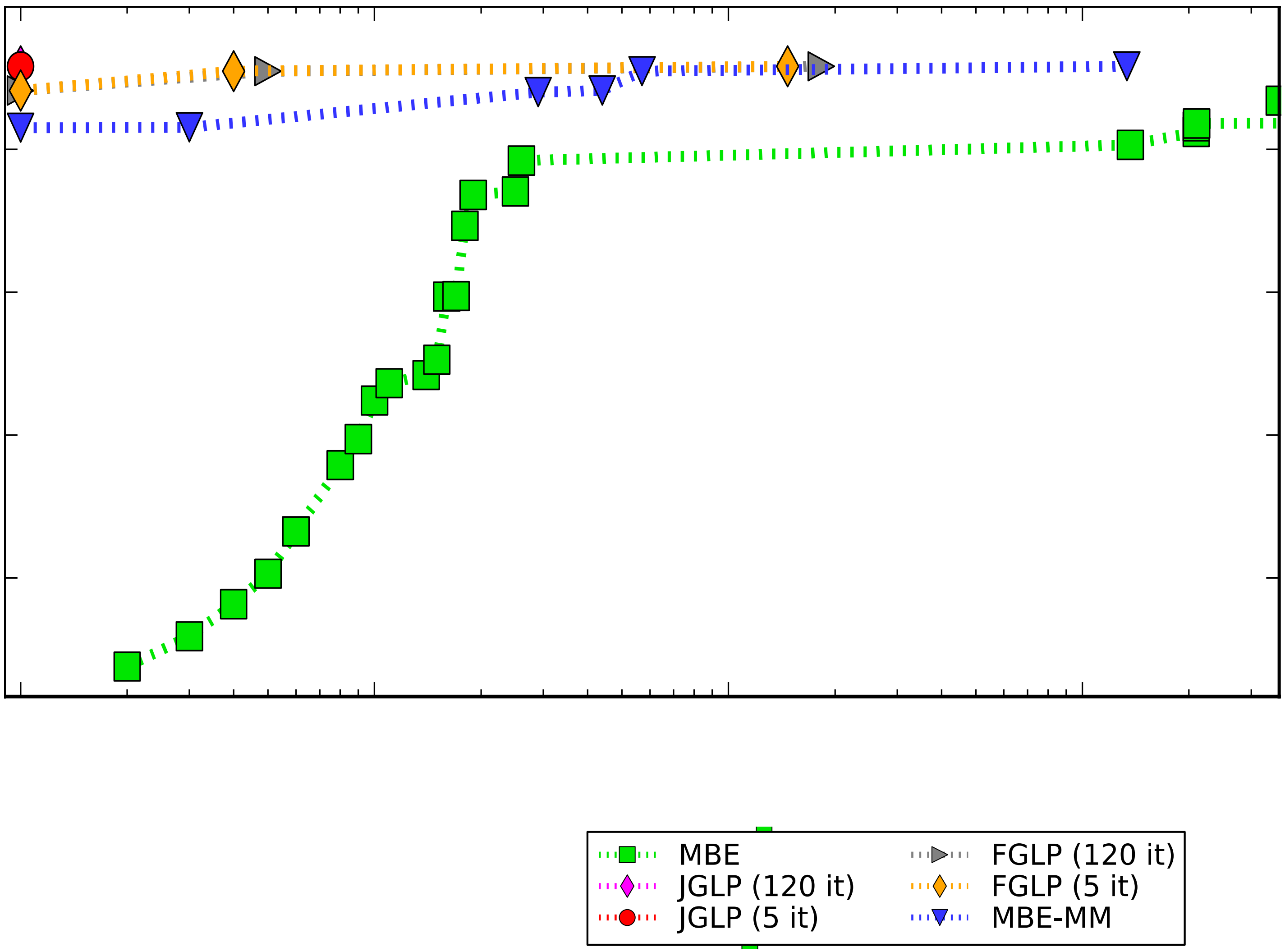


Time bound – 24 h





Empirical Evaluation; Grid Networks

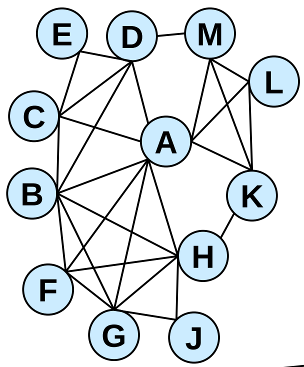


Search + Inference

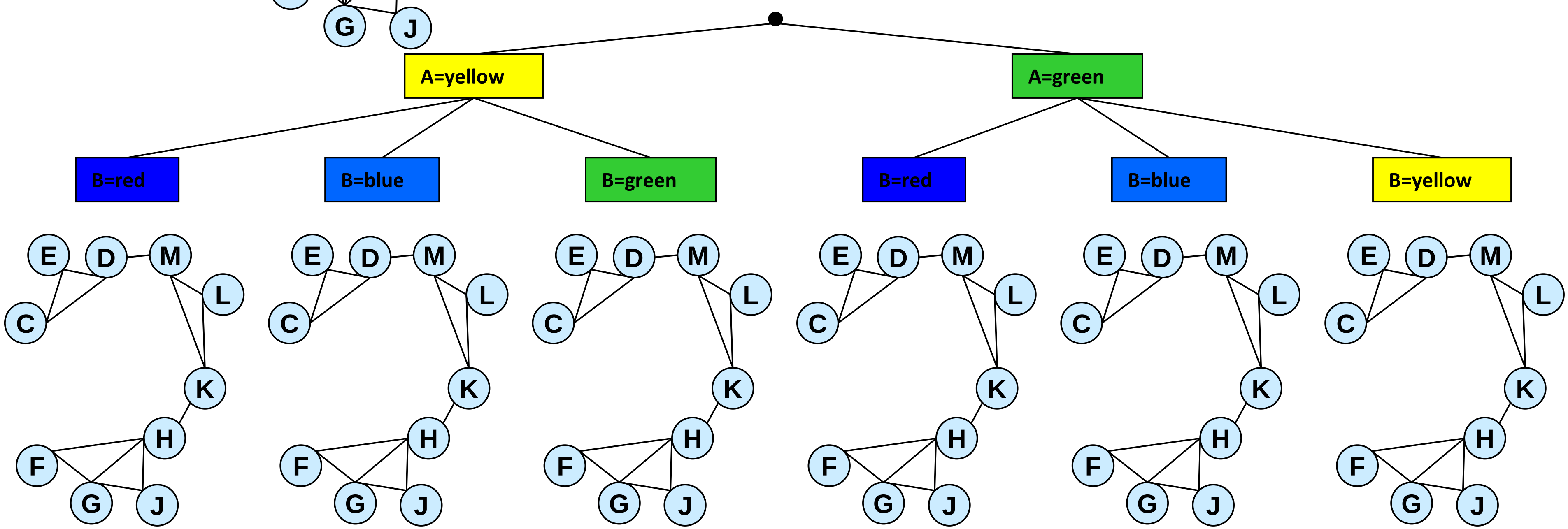
W-Cutset conditioning + inference.

Time exp in cycle-cutset
Memory-linear

Graph Coloring problem



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



Search+Inference :Trading Space for Time

- AO(j): searches depth-first, cache i-context
  - j = the max scope-size of a cache table.

i=0 | **i=j** | i=w*

Space: **O(n)**

Time: **O(n$k^{w* \log n}$)**

Space: **O(nk^{j})**

Time: **O(nk^{j+m_j})**

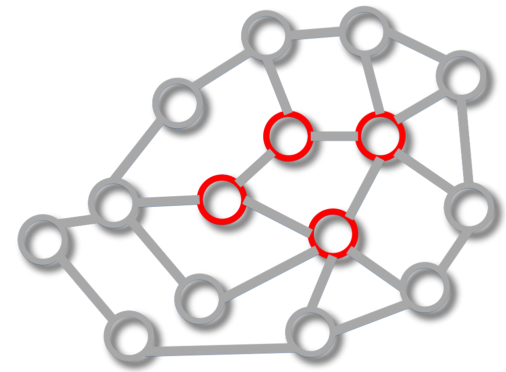
Space: **O(nk^{w*})**

Time: **O(nk^{w*+1})**

Outline

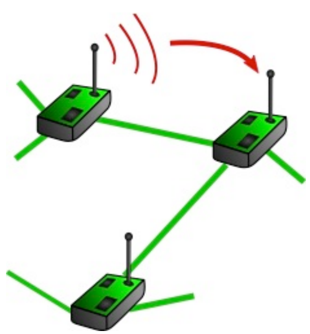
- **Introduction**
  - Graphical models
  - Optimization tasks for graphical models
  - Birds-view of techniques
- **Inference**
  - Variable Elimination, Bucket Elimination
- **Search**
  - AND/OR search spaces
  - Depth-First Branch-and-Bound, Best-First Search
- **Lower-bounds and relaxations**
  - Bounded variable elimination
  - Iterative cost shifting and local consistency
- **Advanced tasks for optimization**
  - Marginal Map for Conformant planning
  - Influence diagrams
- **Software**

Why marginal MAP?

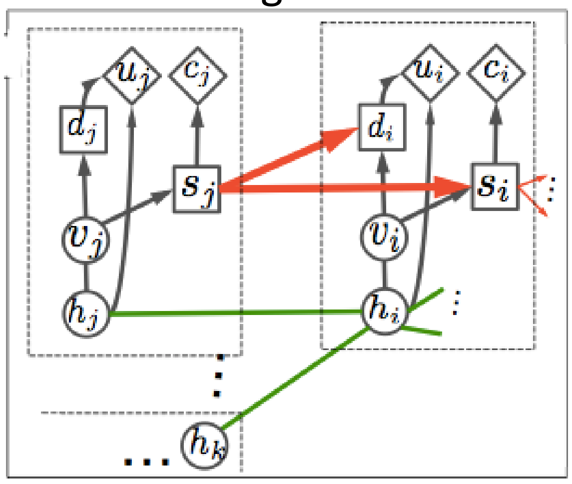


- Often, marginal MAP is the "right" task:
  - We have a model describing a large system
  - We care about predicting the state of some part
- Example: decision making
  - Sum over random variables (random effects, etc.)
  - Max over decision variables (specify action policies)

Sensor network



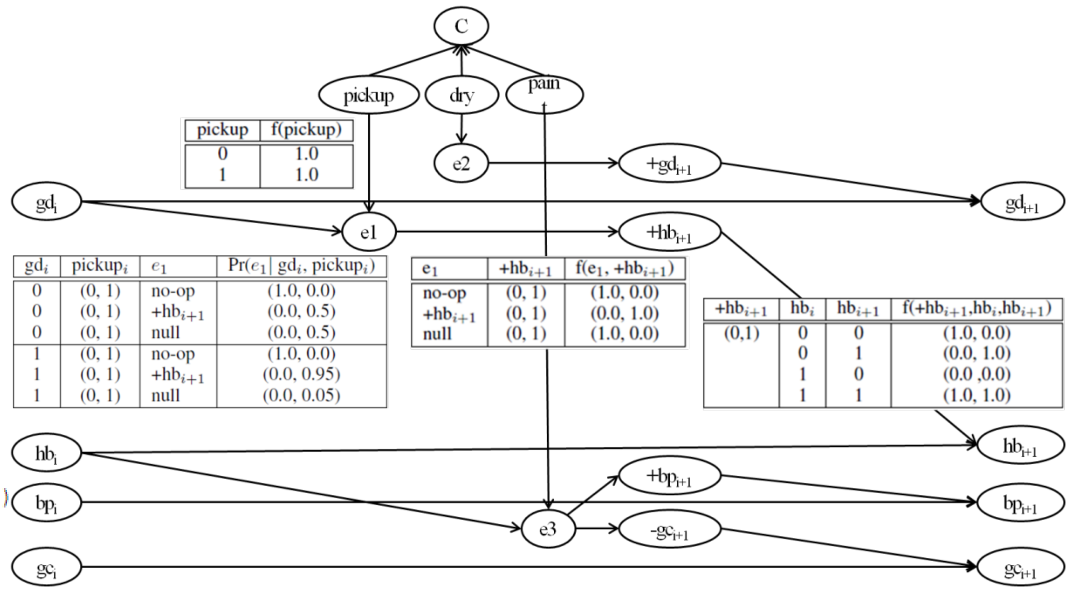
Influence diagram:



Marginal Map: Conformant Planning

Example: Slippery Gripper

```
(define (domain ext-slippery-gripper)
  (:requirements :negative-preconditions :conditional-effects
                 :probabilistic-effects)
  (:predicates (gripper-dry) (holding-block) (block-painted)
               (gripper-clean))
  (:action pickup
      :effect (and (when (gripper-dry)
                         (probabilistic 0.95 (holding-block)))
                   (when (not (gripper-dry))
                         (probabilistic 0.5 (holding-block)))))
  (:action dry
      :effect (probabilistic 0.8 (gripper-dry)))
  (:action paint
      :effect (and (block-painted)
                   (when (not (holding-block))
                         (probabilistic 0.1 (not (gripper-clean))))
                   (when (holding-block)
                         (not (gripper-clean))))))
(define (problem ext-slippery-gripper)
  (:domain ext-slippery-gripper)
  (:init (gripper-clean)
         (probabilistic 0.7 (gripper-dry)))
  (:goal (and (gripper-clean) (holding-block) (block-painted))))
```



pickup	f(pickup)
0	1.0
1	1.0

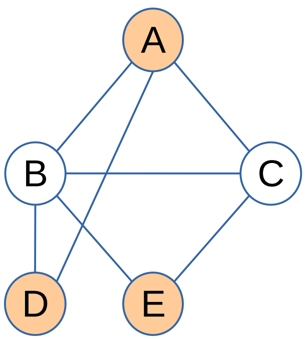
gd_i	$pickup_i$	e_1	$Pr(e_1 \mid gd_i, pickup_i)$
0	(0, 1)	no-op	(1.0, 0.0)
0	(0, 1)	$+hb_{i+1}$	(0.0, 0.5)
0	(0, 1)	null	(0.0, 0.5)
1	(0, 1)	no-op	(1.0, 0.0)
1	(0, 1)	$+hb_{i+1}$	(0.0, 0.95)
1	(0, 1)	null	(0.0, 0.05)

e_1	$+hb_{i+1}$	$f(e_1, +hb_{i+1})$
no-op	(0, 1)	(1.0, 0.0)
$+hb_{i+1}$	(0, 1)	(0.0, 1.0)
null	(0, 1)	(1.0, 0.0)

$+hb_{i+1}$	hb_i	hb_{i+1}	$f(+hb_{i+1}, hb_i, hb_{i+1})$
(0,1)	0	0	(1.0, 0.0)
	0	1	(0.0, 1.0)
	1	0	(0.0 ,0.0)
	1	1	(1.0, 1.0)

- Slippery Gripper Domain [5] : from PPDDL into 2TDBN

Bucket Elimination For MMAP



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

$\mathbf{X}_S = \{B, C\}$

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

[Dechter, 1999]

constrained elimination order

SUM

MAX

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

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

max_D

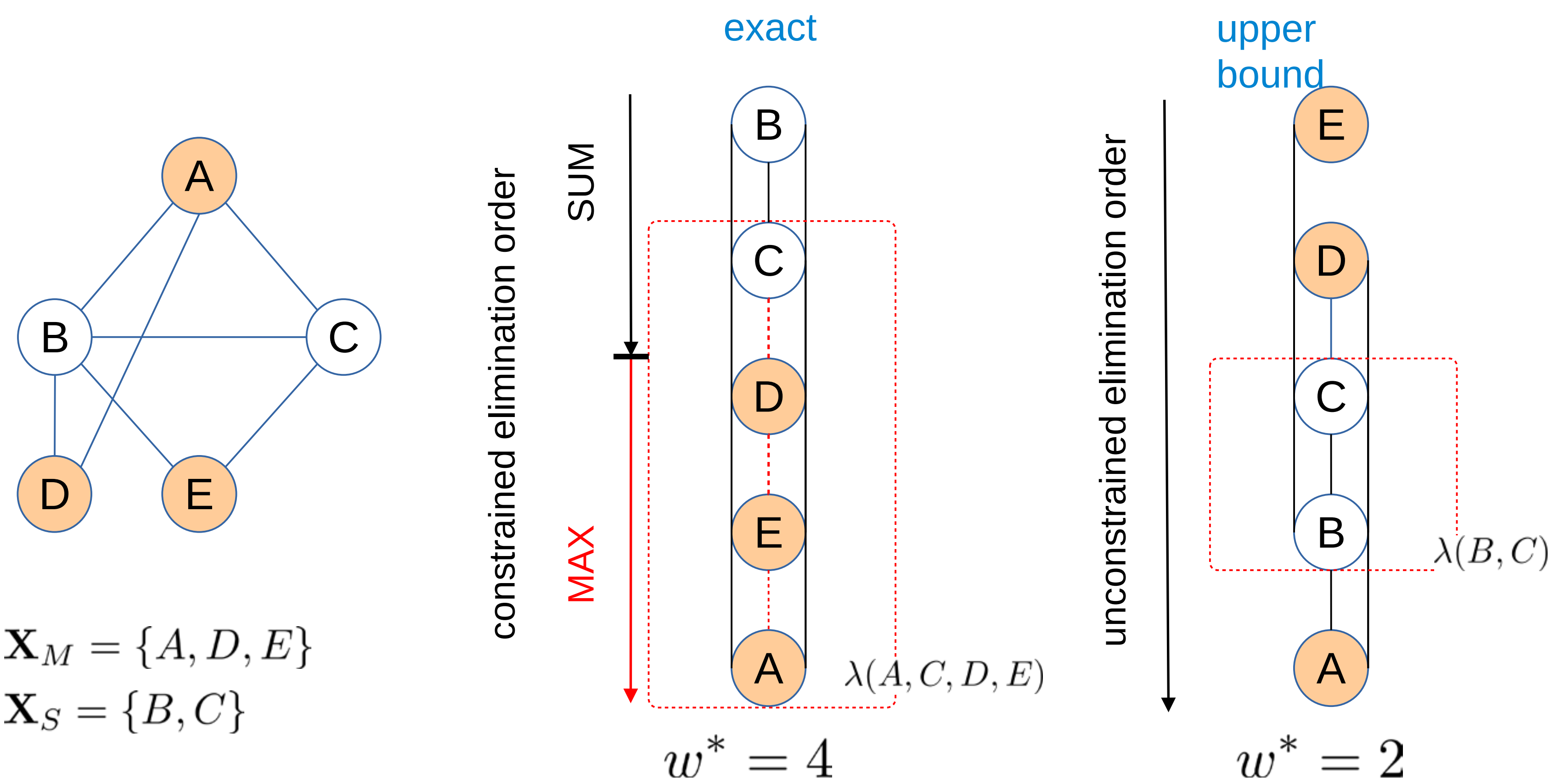
E: $\lambda^D(A,E)$

max_E

A: $\lambda^E(A)$

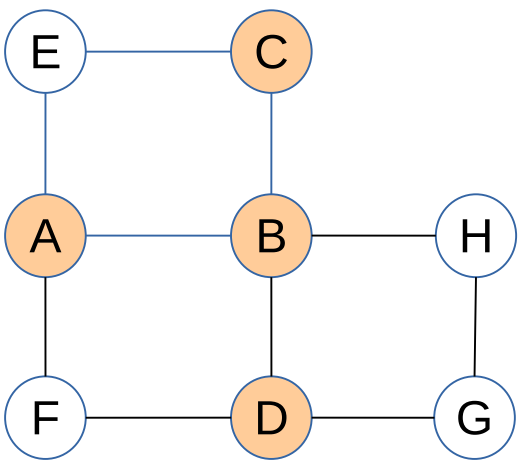
MAP* is the marginal MAP value

Impact of Orderings



In practice, constrained induced is much larger!

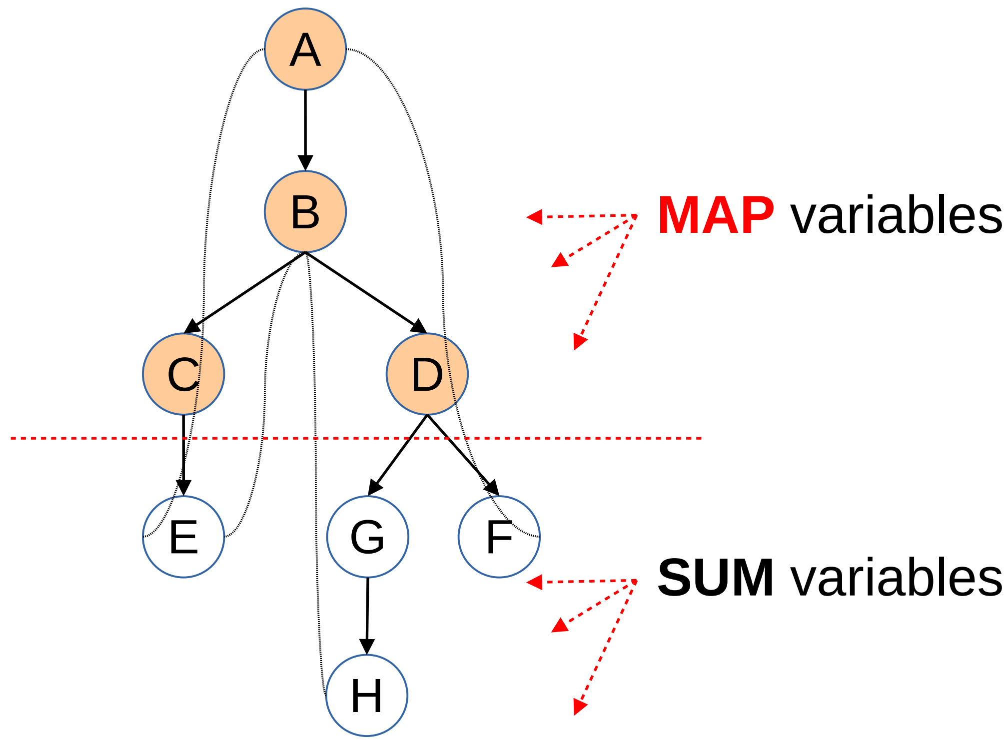
AND/OR Search Spaces for Marginal MAP



primal graph

$$X_M = \{A, B, C, D\}$$

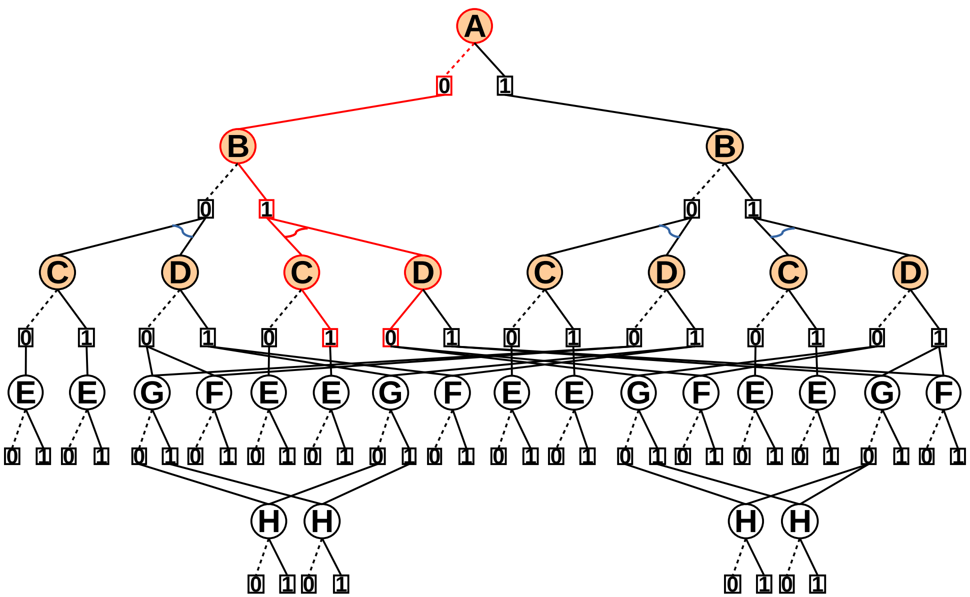
$$X_S = \{E, F, G, H\}$$



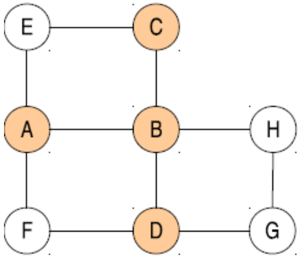
constrained pseudo tree

AND/OR Search Spaces for Marginal MAP (AOBBMM)

- Node types
  - OR (MAP): max
  - OR (SUM): sum
  - AND: multiplication

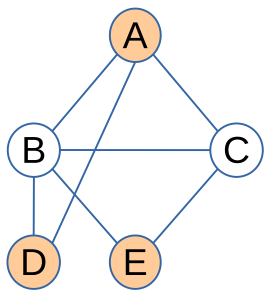


- Depth-first AND/OR search:
  - Maintain best solution cost **L** so far
  - Lower bound
- Heuristic evaluation function **f(n)**
  - Upper bound
- Prune only at OR nodes of t MAP variables
- Cost of MAP assignment obtained by searching the corresponding SUM sub-problem (with caching)



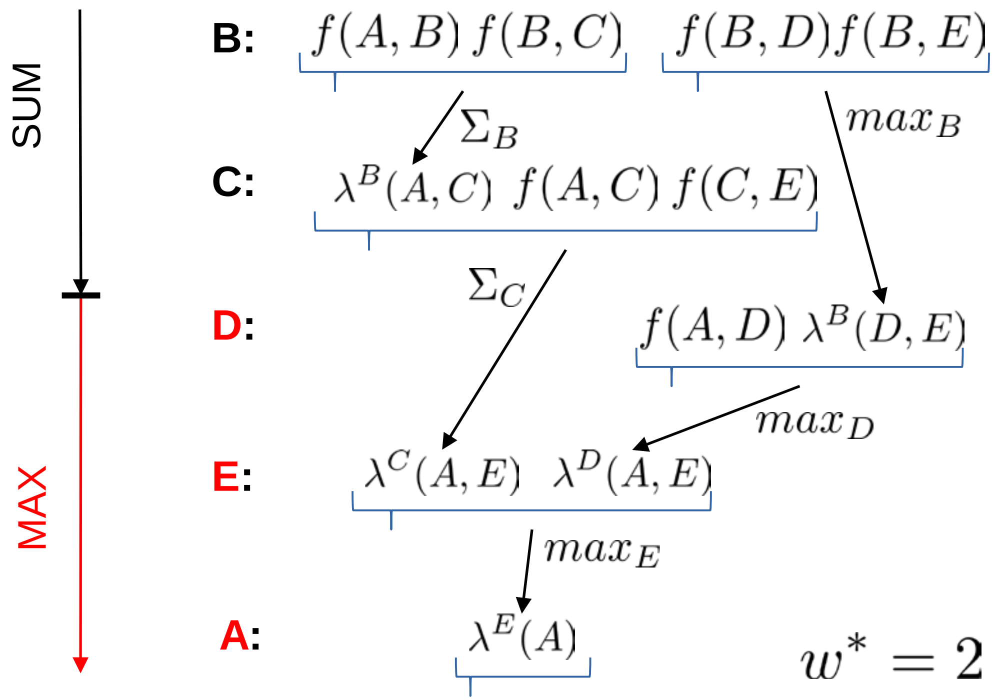
Important: unconstrained ordering (join tree) is not compatible with the pseudo tree!

Mini-Bucket for Marginal MAP



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

Partition a bucket into mini-buckets with i variables



[Dechter and Rish, 2001]

MAP* is an **upper bound** on the marginal MAP value

Weighted Mini-Bucket

- Replace the naive mini-bucket bound with Holder's inequality
  - –Mini-buckets are assigned non-negative weights that sum to 1
  - –Developed for likelihood (summation) tasks [Liu and Ihler, 2012]
  - –Related to variational bounds on likelihood (e.g. [Globerson and Jaakkola, 2007])

$$\sum_x f(x) \longleftrightarrow (\sum_x f(x)^{\frac{1}{w}})^w \triangleq \sum_x^w f(x)$$

- *Marginal MAP:*
  - –

$$X_k \in \mathbf{X}_S \text{ set } \Sigma_r w_{kr} = 1$$

$$X_k \in \mathbf{X}_M \text{ set } \Sigma_r w_{kr} = 0 \longrightarrow \sum_x^w f(x) \sim \max_x f(x)$$

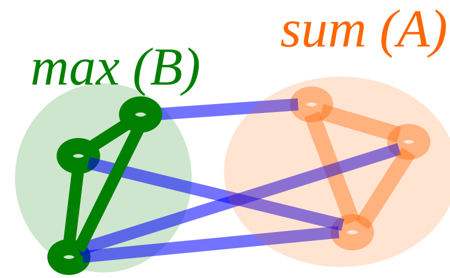
Upper bounds produced by WMB are relatively loose. We tighten them by **cost shifting**.

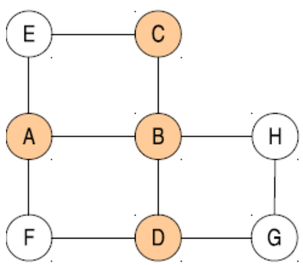
A New Algorithm for Marginal MAP

- Radu Marinescu, Rina Dechter, Alex Ihler (UAI 2014,IJCAI 2015).

- **Problem:** $$\boldsymbol{x}_B^* = \arg\max_{\boldsymbol{x}_B} \sum_{\boldsymbol{x}_A} \prod_{\alpha} \psi(\boldsymbol{x}_\alpha)$$

Marginalize away variables A, then and Find optimal configuration of variables B





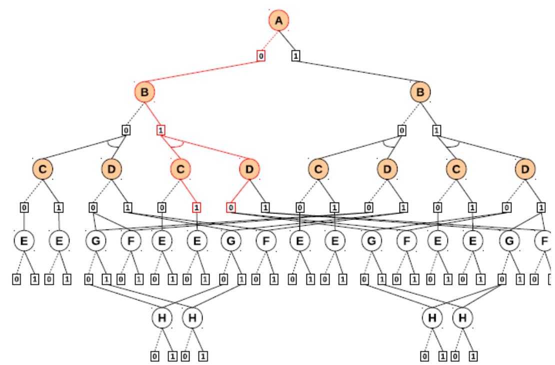


Figure 2: AND/OR search spaces for marginal MAI

Improving Marginal Map for Graphical
Heuristics generated by weighted mini-bucket and moment-matching heuristics.

- Branch and Bound Search of AND/OR search

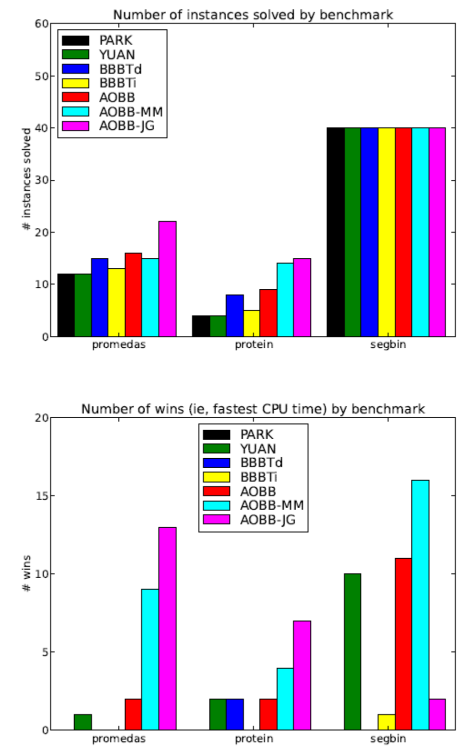
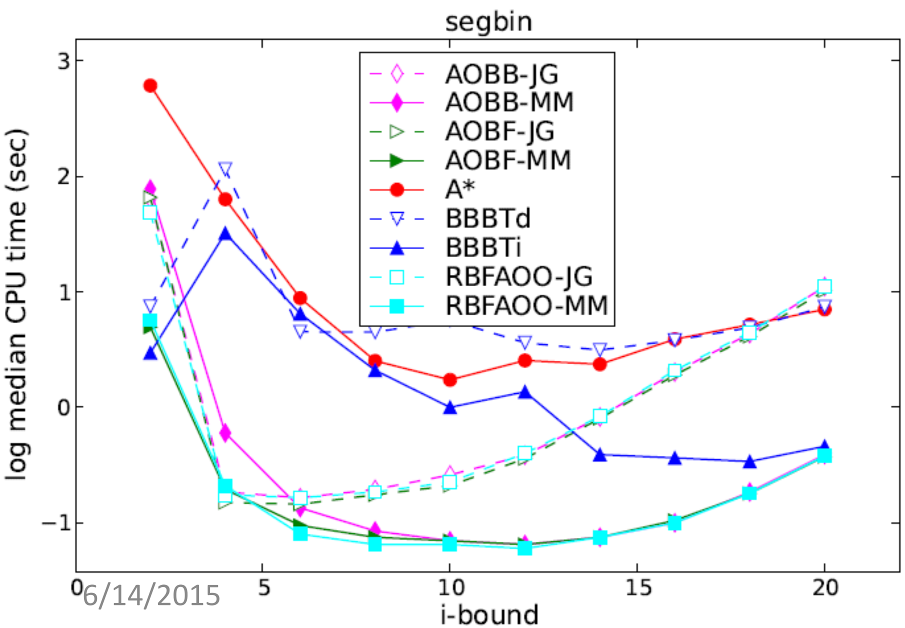
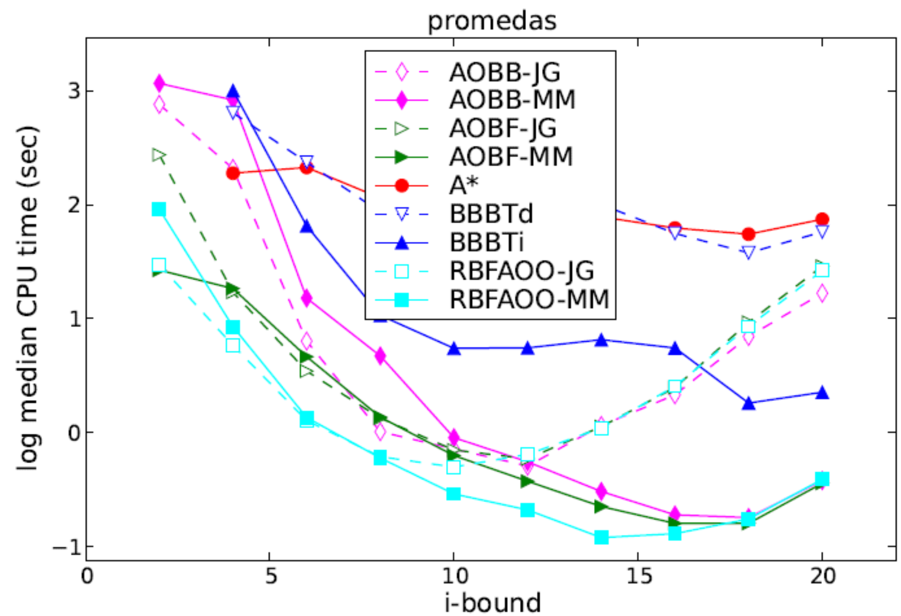


Figure 5: Number of instances solved (top) and number of wins (bottom) by benchmark.

Search for Marginal MAP

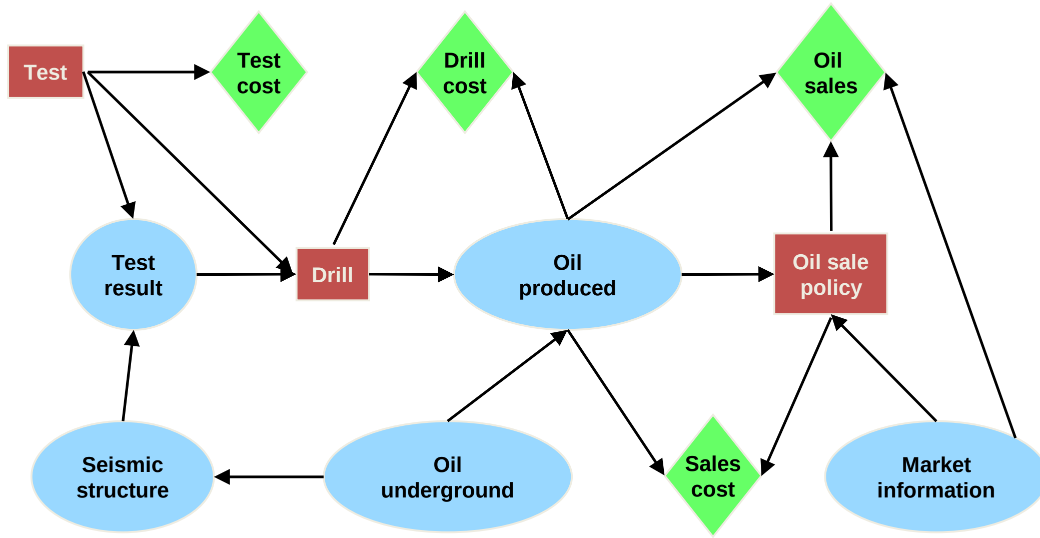
- Depth-first AND/OR search [Marinescu, Ihler, Dechter, UAI 2014]
- Best-first variants
  - Recursive best-first search [Marinescu, Ihler, Dechter, submitted]
  - Weighted best-first search [e.g., Flerova et al., PlanSOpt @ AAAI15]
- Current issues: improving any-time behavior
  - Better initialization & heuristics
  - Treatment of MAP configurations





Influence Diagrams

Query 4: Decision Making MDPs,POMDPs



$$M.E.U = \sum_{I_0} \max_{D_1} \cdots \max_{D_m} \sum_{I_m} (\prod_{P_i} P_i)(\sum_{r_i} r_i),$$

The Car Example (Howard 1976)

A car buyer needs to buy one of two used cars. The buyer can carry out tests with various costs, and then, decide which car to buy.

T: Test variable (t_0, t_1, t_2) (t_1 test car 1, t_2 test car 2)

D: the decision of which car to buy, $D \in \{buy1, buy2\}$

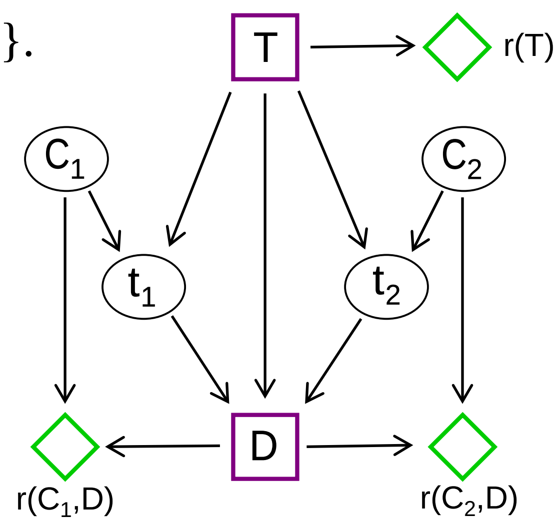
C_i: the quality of car i, $C_i \in \{q_1, q_2\}$

t_i: the outcome of the test on car i, $t_i \in \{pass, fail, null\}$.

$r(T)$: The cost of testing,

$r(C_1,D)$, $r(C_2,D)$: the reward in buying cars 1 and 2.

The utility is: $r(T) + r(C_1,D) + r(C_2,D)$.



Task: determine decision rules T and D such that:

$$E = \max_{T,D} \sum_{t_2,t_1,C_2,C_1} P(t_2,|C_2,T)P(C_2)P(t_1|C_1,T)\cdot$$

$$P(C_1)[r(T)+r(C_2,D)+r(C_1,D)]$$

Bucket Elimination for meu (Algorithm BE-meu-id)

Input: An Influence diagram $ID = \{P_1,\ldots,P_n,r_1,\ldots,r_j\}$
Output: Meu and optimizing policies.

1. **Order the variables and partition into buckets.**
2. **Process buckets from last to first:**

$o = T,t_2,t_2,D,C_2,C_1$

$bucket(C_1)$: $P(C_1), P(t_1|C_1,T), r(C_1,D)$

$bucket(C_2)$: $P(C_2), P(t_2|C_2,T), r(C_2,D)$

$bucket(D)$: $\theta_{C_1}(t_1,T,D), \theta_{C_2}(t_2,T,D)$

$bucket(t_1)$: $\lambda_{C_1}(t_1,T)$ $\theta_D(t_1,t_2,T), \delta(t_1,t_2,T)$

$bucket(t_2)$: $\lambda_{C_2}(t_2,T)$ $\theta_{t_1}(t_2,T)$

$bucket(T)$: $r(T)$ $\lambda_{t_1}(T)$ $\lambda_{t_2}(T)$ $\theta_{t_1}(T)$

θ_T, δ_T

3. **Forward:** Assign values in ordering d

Processing A Chance Bucket

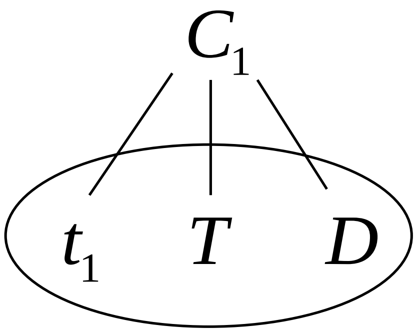
Chance bucket: $bucket_p$**:** $\lambda_1, \lambda_2, \ldots, \lambda_j, \qquad \theta_1, \theta_2 \ldots, \theta_l$

$$\lambda_p = \sum_{X_p} \prod_i \lambda_l, \quad , \theta_p = \frac{1}{\lambda p} \sum_{X_p} \prod_{i=1}^{j} \lambda_l \sum_{j=1}^{l} \theta_j$$

Bucket C_1 **in car example:** $P(C_1), P(t_1 \mid C_1, T), r(C_1, D)$

$$\lambda_{C_1}(t_1, T) = \sum_{C_1} P(C_1) P(t_1 \mid C_1, T),$$

$$\theta_{C_1}(t_1, T, D) = \frac{1}{\lambda_{C1}} \sum_{C_1} P(C_1) P(t_1 \mid C_1, T) r(C_1, D)$$



Processing A Decision Bucket

Decision variable: $bucket_p$**:** $\quad \underbrace{\lambda_1, \lambda_2, \ldots, \lambda_j, \qquad \theta_1, \theta_2 \ldots, \theta_l}$

$$\lambda_p = \max_{Xp} \prod_{i=1}^{j} \lambda_i, \sum_{j=1}^{l} \theta_j$$

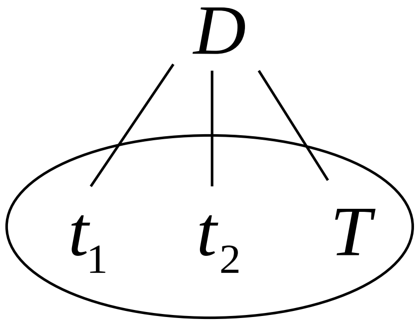
$$\underbrace{\theta_1, \ldots, \theta_l}$$

$$\theta_p = \max_{X_P} \sum_j \theta_j, \qquad \delta^o = argmax_{X_p} \theta_p$$

Processing D**, car example:**

$\theta(t_1, T, D) \qquad \theta(t_2, T, D)$

$$\theta_P(t_1, t_2, T) = \max_D (\theta(t_1, T) + \theta(t_2, T))$$



AND/OR Search for Influence Diagrams

A New Approach to Influence Diagrams Evaluation

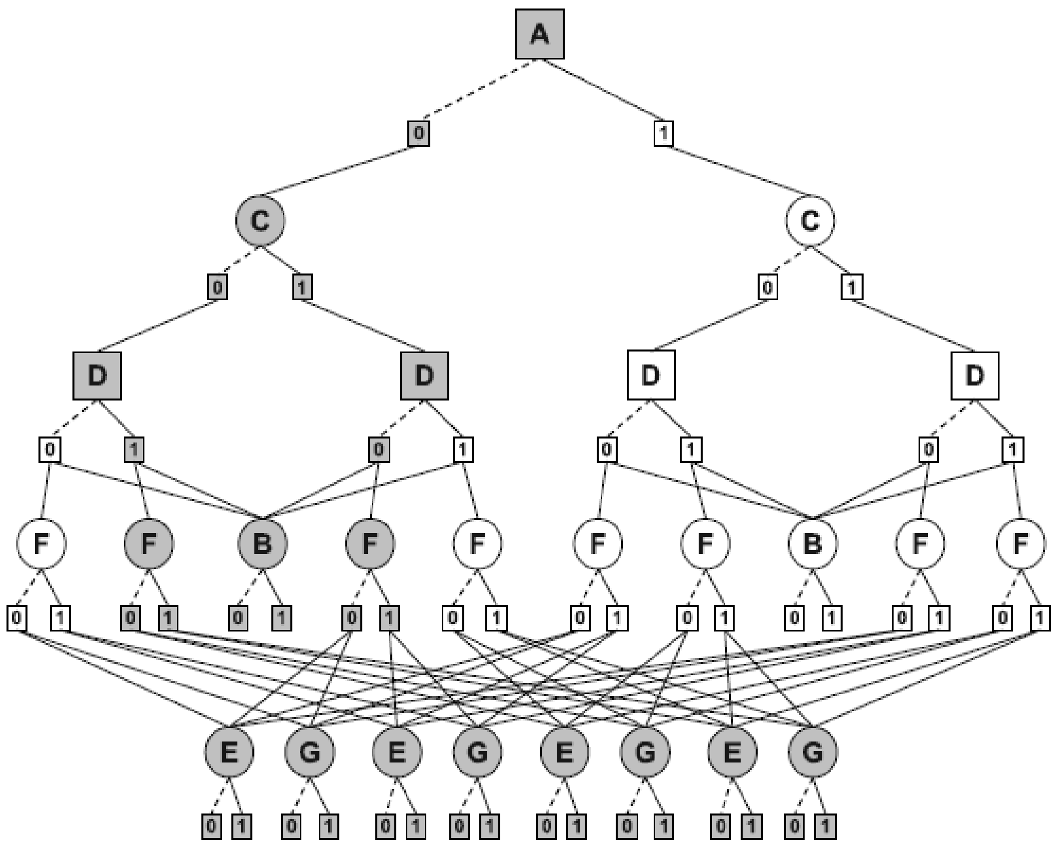


Fig. 3 Context-minimal AND/OR search graph for influence diagrams.

Radu Marinescu, A New Approach to Influence Diagrams Evaluation,2010

Outline

- **Introduction**
  - Graphical models
  - Optimization tasks for graphical models
  - Birds-view of techniques
- **Inference**
  - Variable Elimination, Bucket Elimination
- **Search**
  - AND/OR search spaces
  - Depth-First Branch-and-Bound, Best-First Search
- **Lower-bounds and relaxations**
  - Bounded variable elimination
  - Iterative cost shifting and local consistency
- **Advanced tasks for optimization**
  - Marginal Map for Conformant planning
  - M-best search
  - Influence diagrams
- **Software**

PASCAL 2012 Inference Challenge

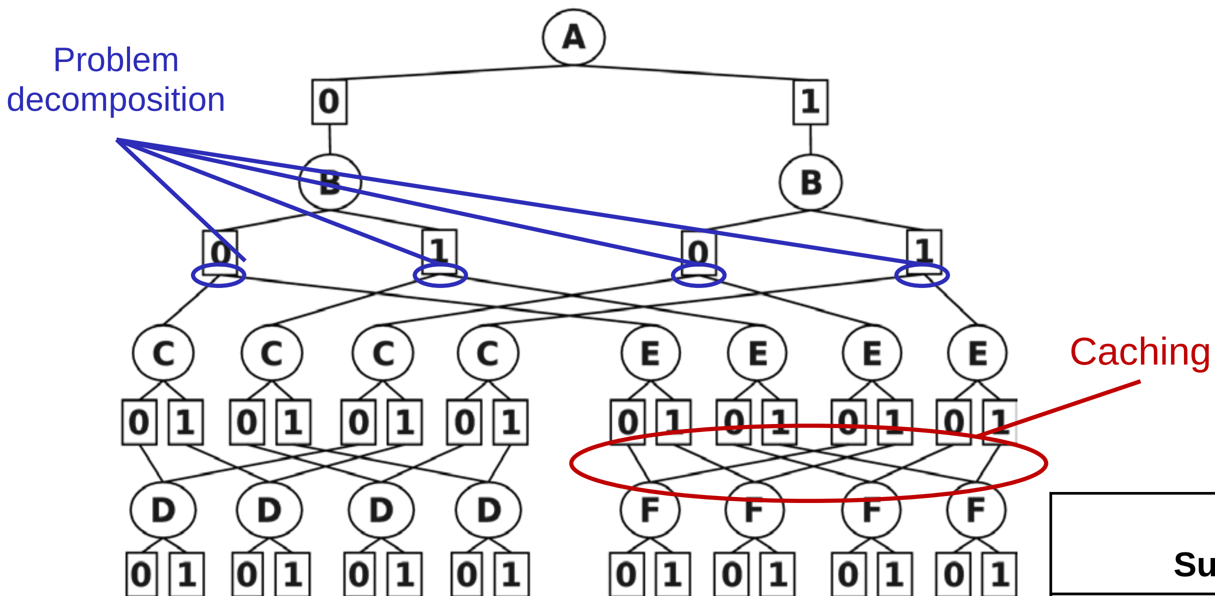
DAOOPT: Improving AND/OR Branch-and-Bound for Graphical Models

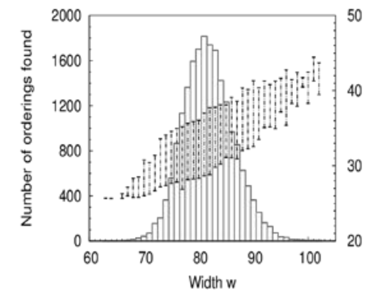
(also won 2nd place uai 2014 as is)

Lars Otten, Alexander Ihler,
Kalev Kask, Rina Dechter

Dept. of Computer Science
University of California, Irvine

State-of-the-art AOBB Search



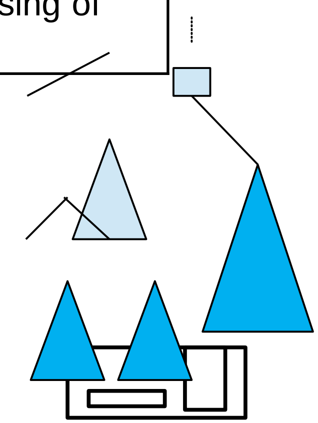


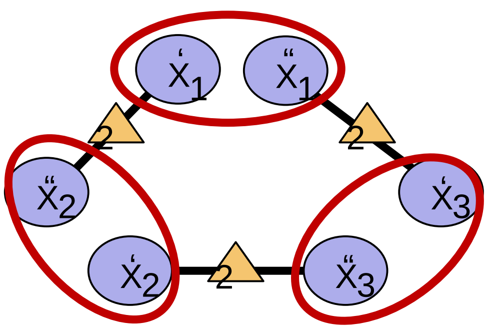
Enhanced Variable Ordering Schemes

Highly efficient, stochastic minfill / mindegree implementations for lower-width orderings.

Breadth-First Subproblem Rotation

Improved anytime perfor-mance through interleaved processing of independent subproblems.





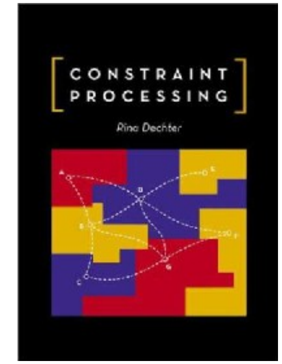
$\min_{\lambda} \sum_{(ij)} \max_{X} \left(f_{ij}(X_i, X_j) + \lambda_{ij}(X_i), \lambda_{ji}(X_j) \right)$

Mini-Buckets Cost-shifting (MPLP) Re-parametrization

Tighter bounds by iteratively solving linear programming relaxations and message passing on join graph.

UCI Library: Summary

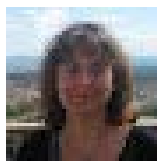
- Exact/anytime:
  - **Likelihood:** *BE, BEEM, VEC(w), AOlibPE(c-bound)*
  - ***MAP****: VE, BEEM (external memory/multi-core), AOBB(i), BRAOBB(i), DAOOPT(Distributed AOBB).*
  - Marginal Map (currently developed)
- *Approximation/anytime, for all queries:*
  - *BP, IJGP(i-bound)*
  - *IJGP-Importance Sampling(i-bound)*
  - *IJGP-SampleSearch(i-bound)*
  - *MBE (mini-bucket), Weighted-mini-bucket, reparameterized MB*
  - *STLS (currently developed, for MAP)*
- *Supporting schemes: Variable-ordering (IGVO)*



For publication see:

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



Thank you

Kalev Kask
Irina Rish
Bozhena Bidyuk
Robert Mateescu
Radu Marinescu
Vibhav Gogate
Emma Rollon
Lars Otten
Natalia Flerova
Andrew Gelfand
William Lam
Junkyu Lee