BREN:ICS INFORMATION AND COMPUTER SCIENCES

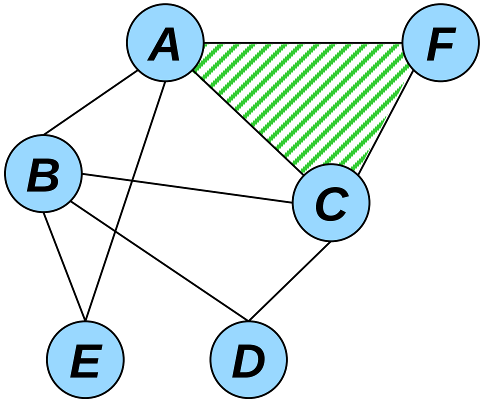
UNIVERSITY of CALIFORNIA IRVINE

Modern Exact and Approximate Combinatorial Optimization algorithms for Graphical Models

In the pursuit of a universal solver

Rina Dechter

Donald Bren school of ICS, University of California, Irvine

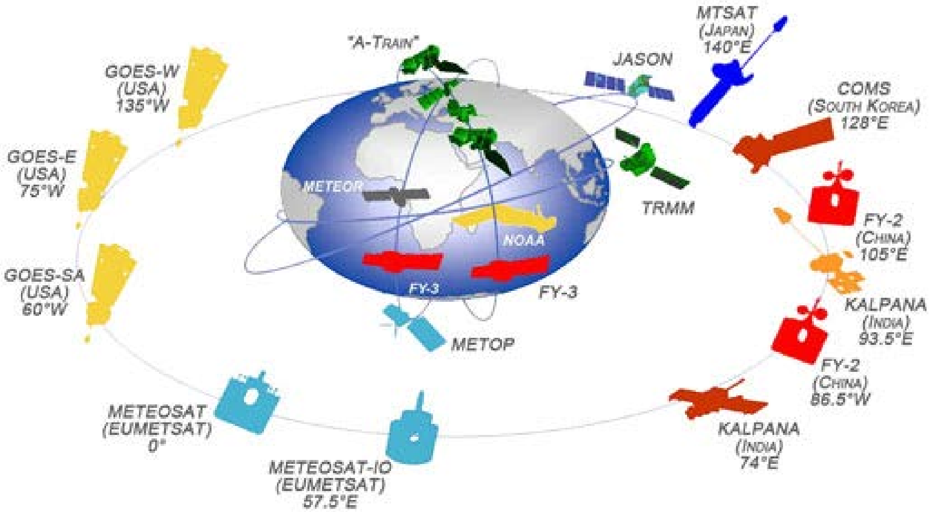


Main collaborator:
Radu Marinescu
Lars Otten
Alex Ihler
Kalev Kask
Robert Mateescu

CMU 2016

Combinatorial Optimization

Planning & Scheduling



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

Computer Vision

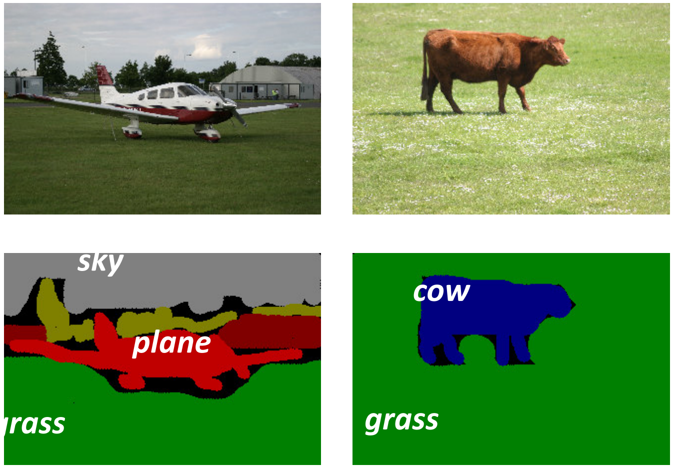


Image classification: label pixels in an image by their associated object class

[He et al. 2004; Winn et al. 2005]

Sample Applications for Graphical Models

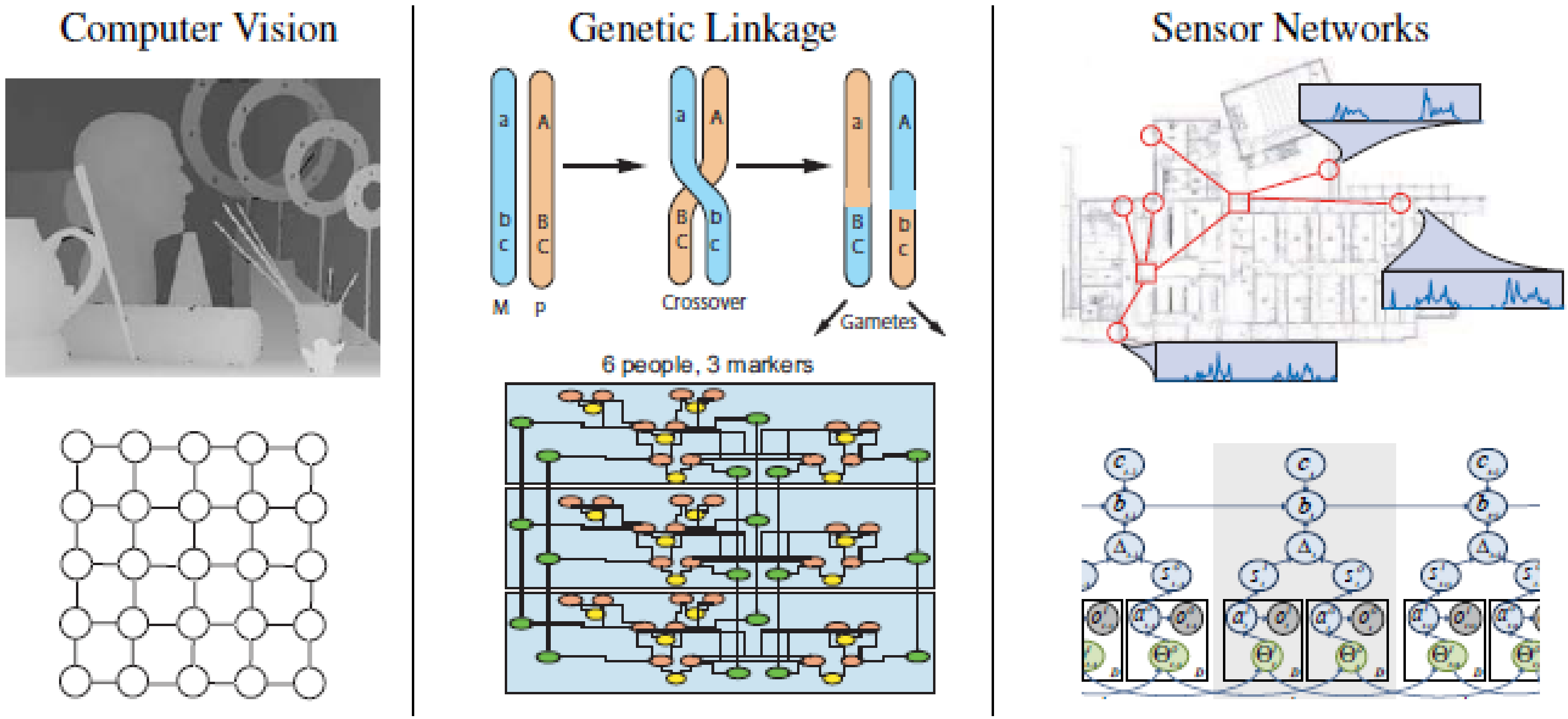
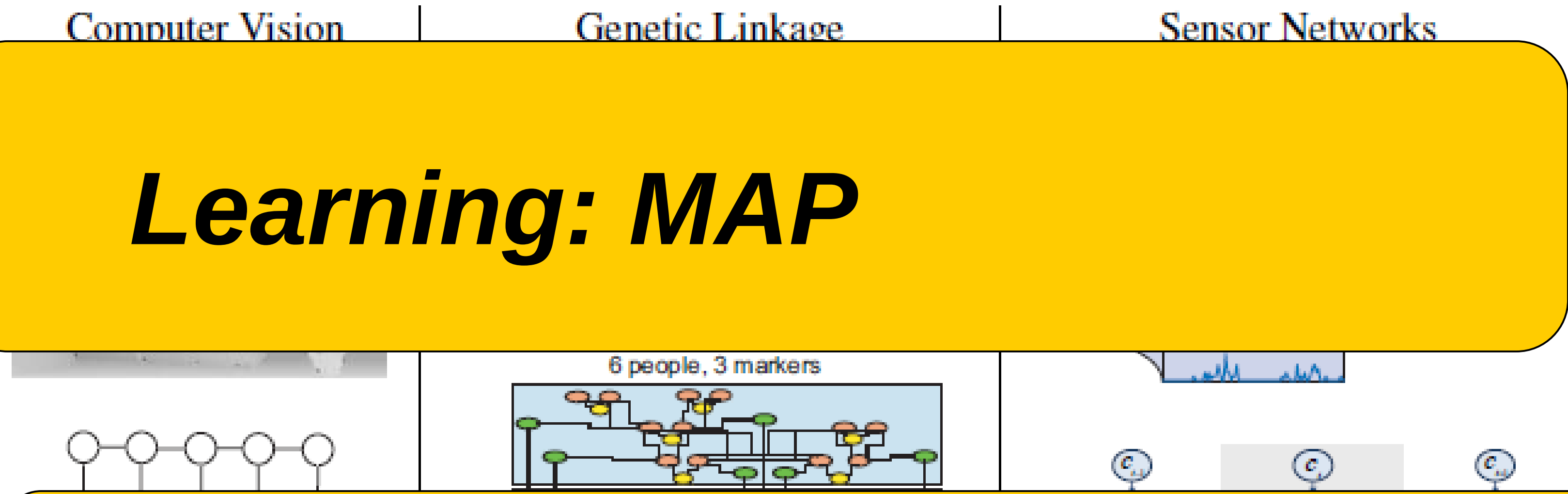


Figure 1: Application areas and graphical models used to represent their respective systems: (a) Finding correspondences between images, including depth estimation from stereo; (b) Genetic linkage analysis and pedigree data; (c) Understanding patterns of behavior in sensor measurements using spatio-temporal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

Sample Applications for Graphical Models



Learning: MAP

Reasoning: MAP

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.

How to Design a Good MAP Solver

- Heuristic Search
- The core of a good search algorithm
  - A compact search space
  - A good heuristic evaluation function
  - A good traversal strategy
- Anytime search yields a good approximation.

Outline

- Graphical models, Queries, Algorithms
- Inference Algorithms: bucket-elimination
- Bounded Inference: a) mini-bucket, b) cost-shifting
- AND/OR search spaces and AND/OR BnB
- Evaluation, Software
- Conclusion

Graphical Models, Queries, Algorithms

Any collection of local functions over a subset of variable Is a graphical model

Constraint Optimization Problems for Graphical Models

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

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

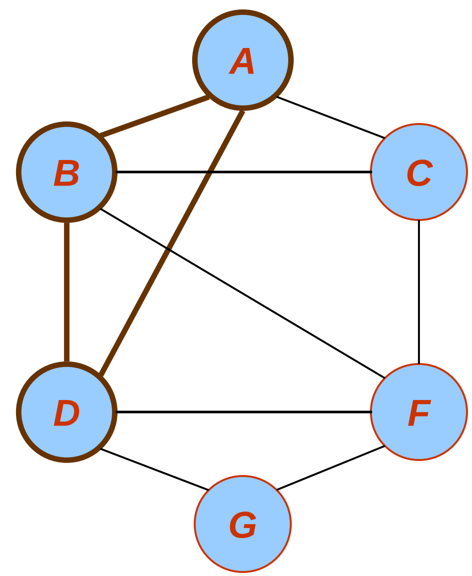
$D = \{D_1, \ldots, D_n\}$ - domains

$F = \{f_1, \ldots, f_m\}$ - 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

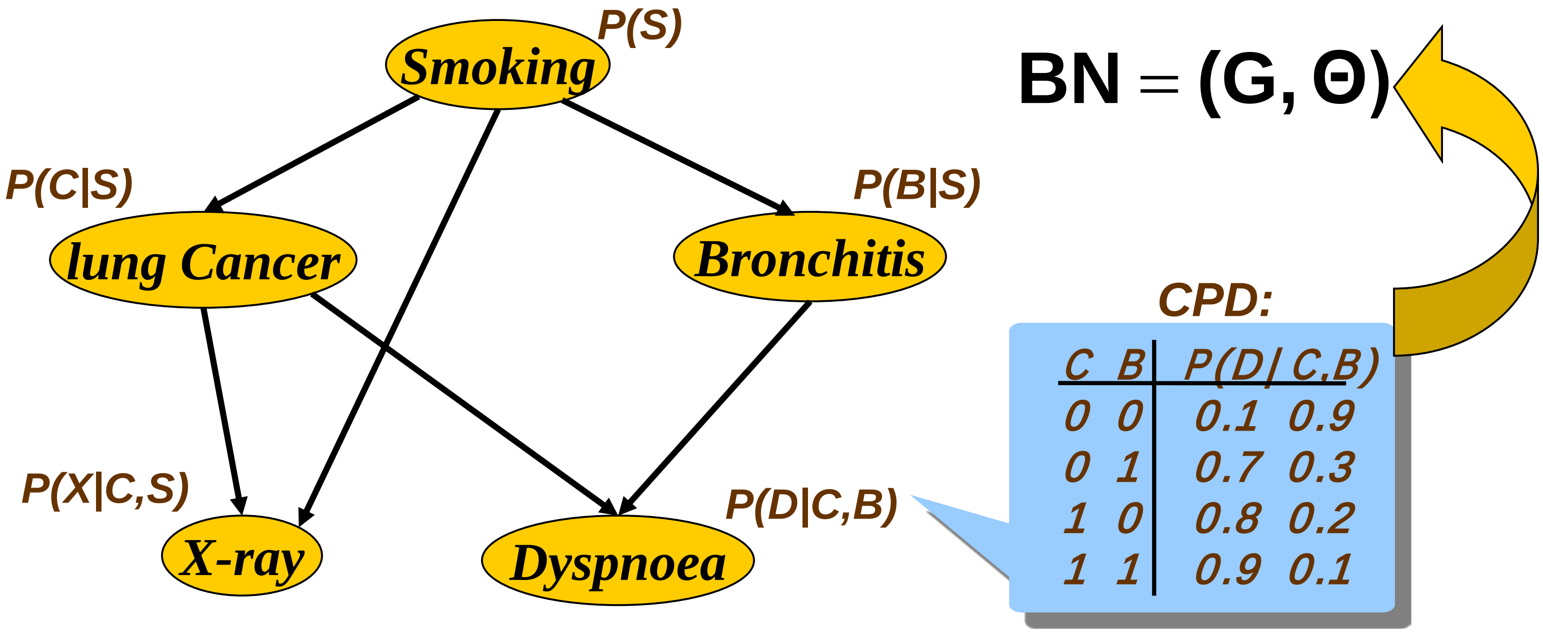


F(a,b,c,d,f,g)= f1(a,b,d)+f2(d,f,g)+f3(b,c,f)

Global Cost Function

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

Bayesian Networks (Pearl, 1988)



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)

Belief Updating:

P (lung cancer=yes | smoking=no, dyspnoea=yes) = ?

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

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 : Sum, product, join
  - Elimination: projection, sum, max/min

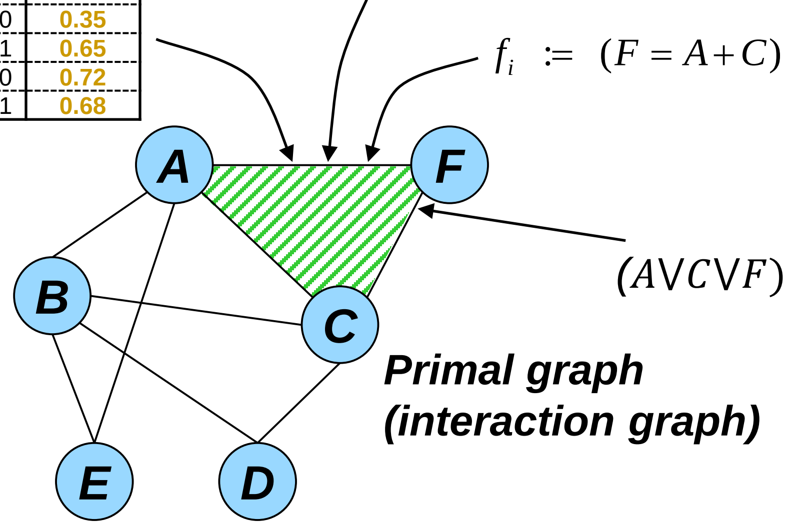
- Tasks:
  - **Belief updating**: $\Sigma_{X\backslash Y}\prod_j P_i$
  - **MPE\MAP**: $\max_X \prod_j P_j$
  - **Marginal MAP**: $\max_Y \Sigma_{X\backslash Y} \prod_j P_j$
  - **CSP**: $\prod_X \times_j C_j$
  - **Max-CSP**: $\min_X \Sigma_j F_j$

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

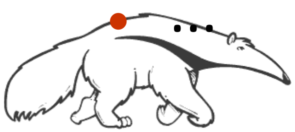


All these tasks are NP-hard

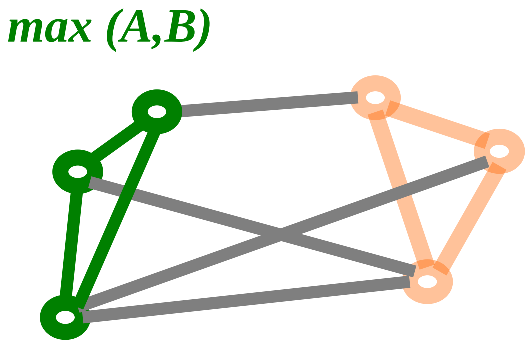
- *exploit problem structure*
- *identify special cases*
- *approximate*

Example Domains for Graphical Models

- Natural Language Processing
  - Information extraction, semantic parsing, translation, summarization, ...
- Computer Vision
  - Object recognition, scene analysis, segmentation, tracking, ...
- Computational Biology
  - Pedigree analysis, protein folding / binding / design, sequence matching, ...
- Networks
  - Webpage link analysis, social networks, communications, citations, ...
- Robotics
  - Planning & decision making, ...
- ...



Max-product = min-sum



$$x^*_{AB} = \arg\max_{x_A, x_B} \prod_{x_\alpha} \varphi_\alpha$$

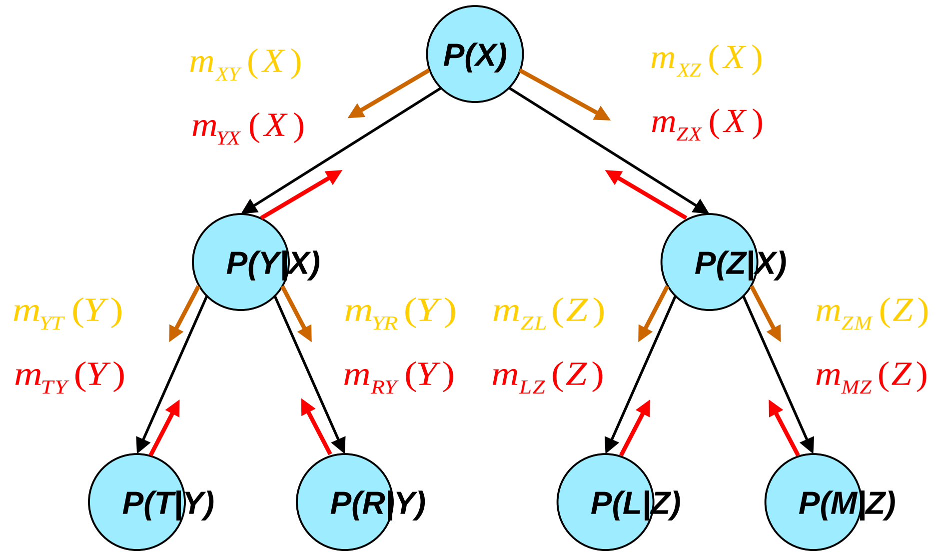
$$x^*_{AB} = \arg\min_{x_A, x_B} \sum_{x_\alpha} \varphi'_\alpha$$

$$\varphi'_\alpha = -\log \varphi_\alpha$$

Tree-solving is easy

Belief updating (sum-prod)

CSP – consistency (projection-join)

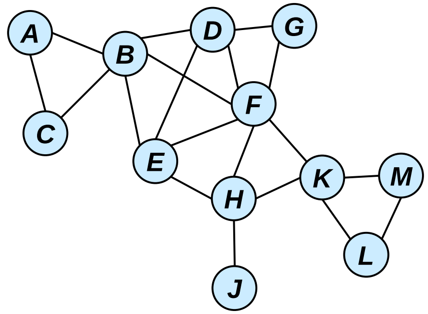


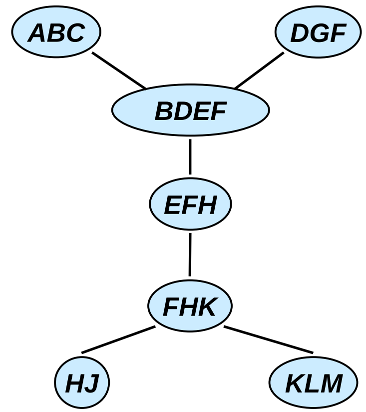
MPE (max-prod)

#CSP (sum-prod)

Trees are processed in linear time and memory

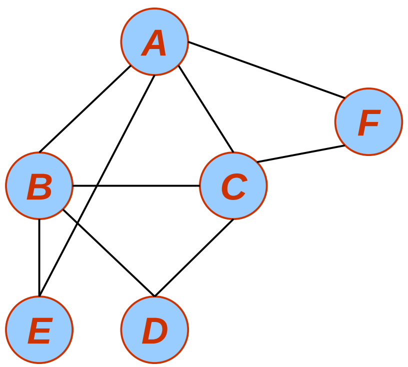
Inference vs conditioning-search

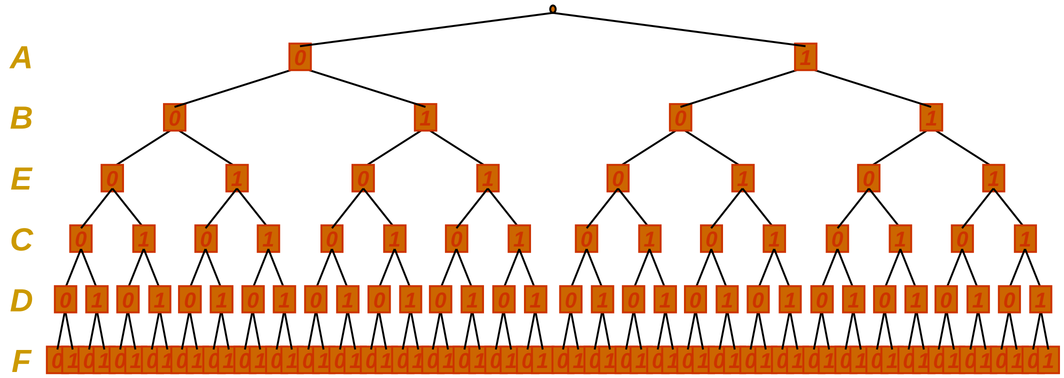




Inference

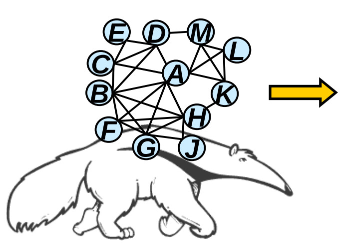
exp(w) time/space*

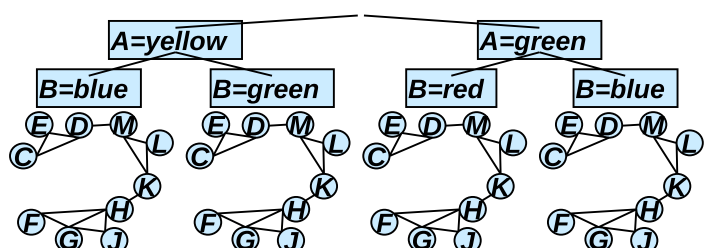




Search

Exp(n) time
O(n) space

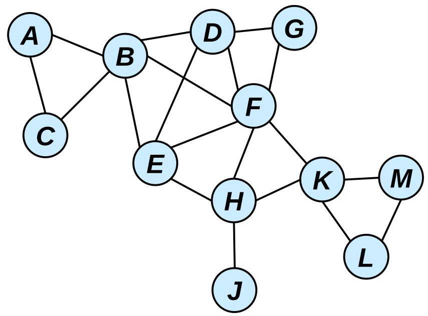


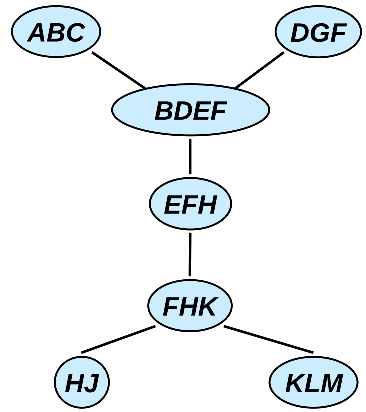


Search+inference:
Space: exp(w)
Time: exp(w+c(w))

w: user controlled

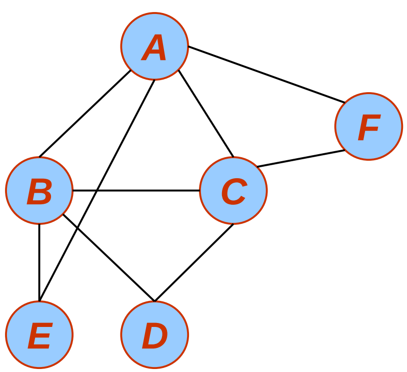
Inference vs conditioning-search

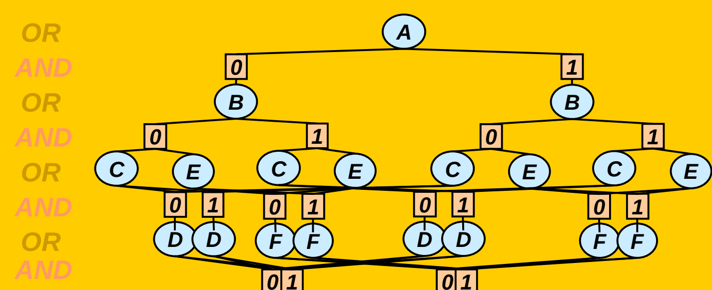




Inference

exp(w) time/space*



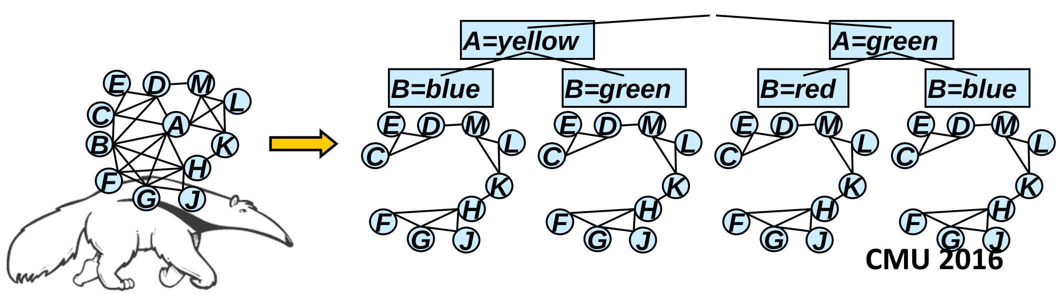


Context minimal AND/OR search graph

18 AND nodes

Search

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



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

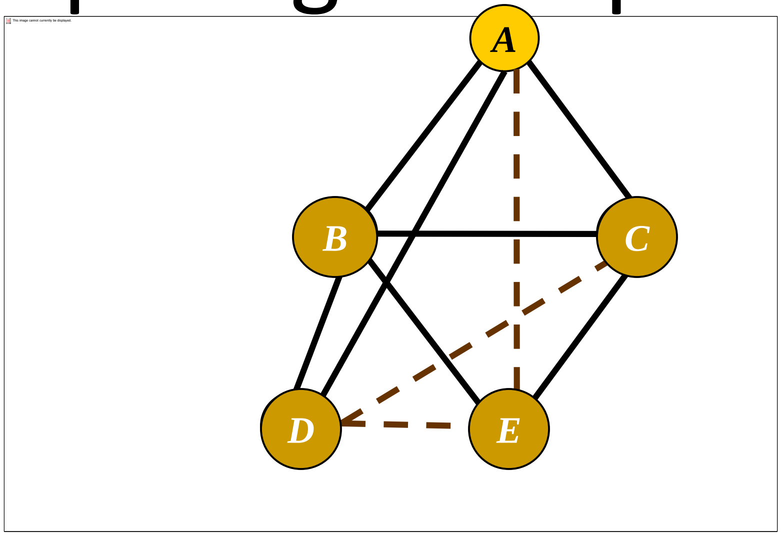
q: user controlled

Outline

- Graphical models, Queries, Algorithms
- **Inference Algorithms: bucket-elimination**
- Bounded Inference: a) mini-bucket, b) cost-shifting
- AND/OR search spaces and AND/OR BnB
- Evaluation, Software
- Conclusions

- ***Solving MAP by Inference:***
- ***Non-serial Dynamic programming***
- ***The induced-width/treewidth***

Computing the Optimal Cost Solution



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

Combination

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

$$\lambda_{B\rightarrow C}(a,d,c,e)$$

Variable Elimination

Bucket Elimination

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

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

$\min_b \sum$ ← ***Elimination/Combination operators***

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

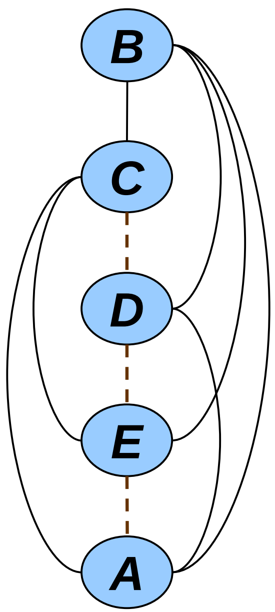
bucket C: $f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,d,c,e)$

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

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

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

F*=OPT



Complexity of Bucket Elimination

Algorithm elim-opt [Dechter, 1996]
Non-serial Dynamic Programming [Bertele & Briochi, 1973]

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

$\min_b \sum$ ← ***Elimination/Combination operators***

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

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

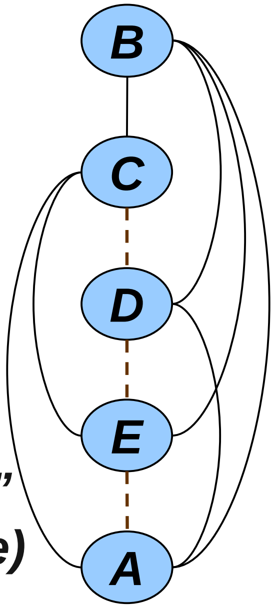
bucket D: $f(a,d)\ \lambda_{C\to D}(a,d,e)$

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

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

$F^*=OPT$

exp(w*=4)
"induced width"
(max clique size)



Generating the Optimal Assignment

$$\mathbf{b}^* = \arg\min_{\mathbf{b}} f(a^*, b) + f(b, c^*) + f(b, d^*) + f(b, e^*)$$

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

Time and space exponential in the induced-width / treewidth

$$O(nk^{w^*+1})$$

Return: $(\mathbf{a}^*, \mathbf{b}^*, \mathbf{c}^*, \mathbf{d}^*, \mathbf{e}^*)$

Complexity of Bucket Elimination

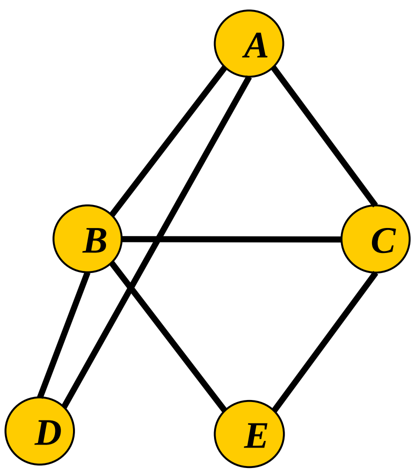
Bucket Elimination is time and space

$$O\left(r \exp(\, w^*(d)\,)\right)$$

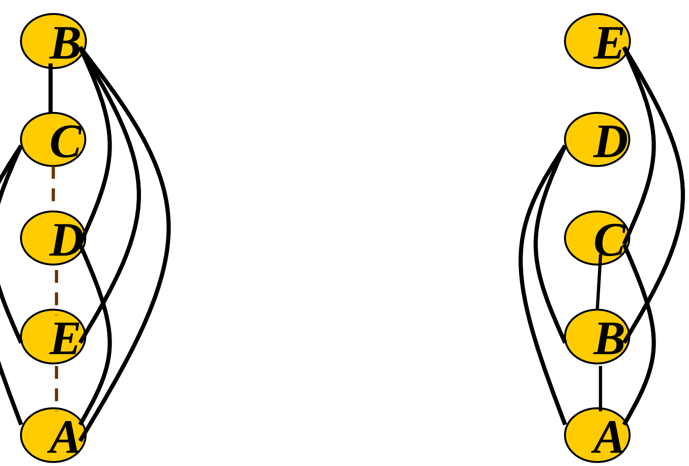
$$w^*(d) - \text{the induced width of graph along ordering } d$$

r = number of functions

The effect of the ordering:



"Moral" graph

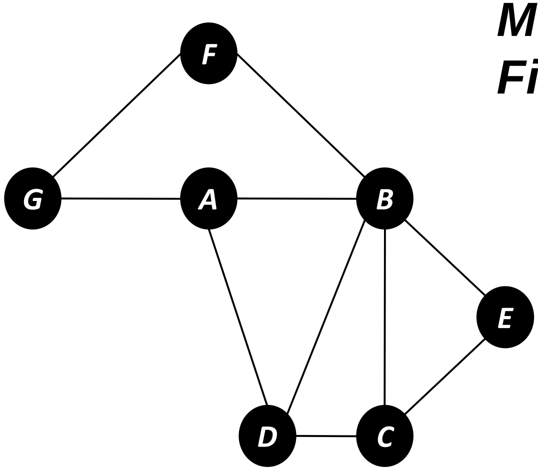


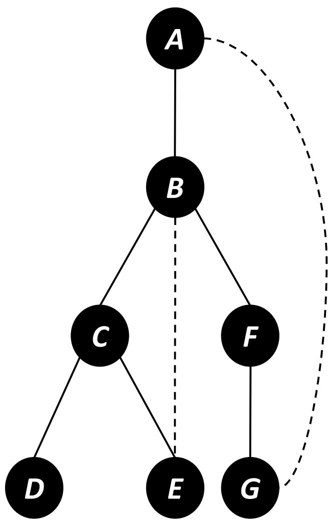
$$w^*(d_1) = 4 \qquad w^*(d_2) = 2$$

Finding the smallest induced width is hard!

Bucket Elimination

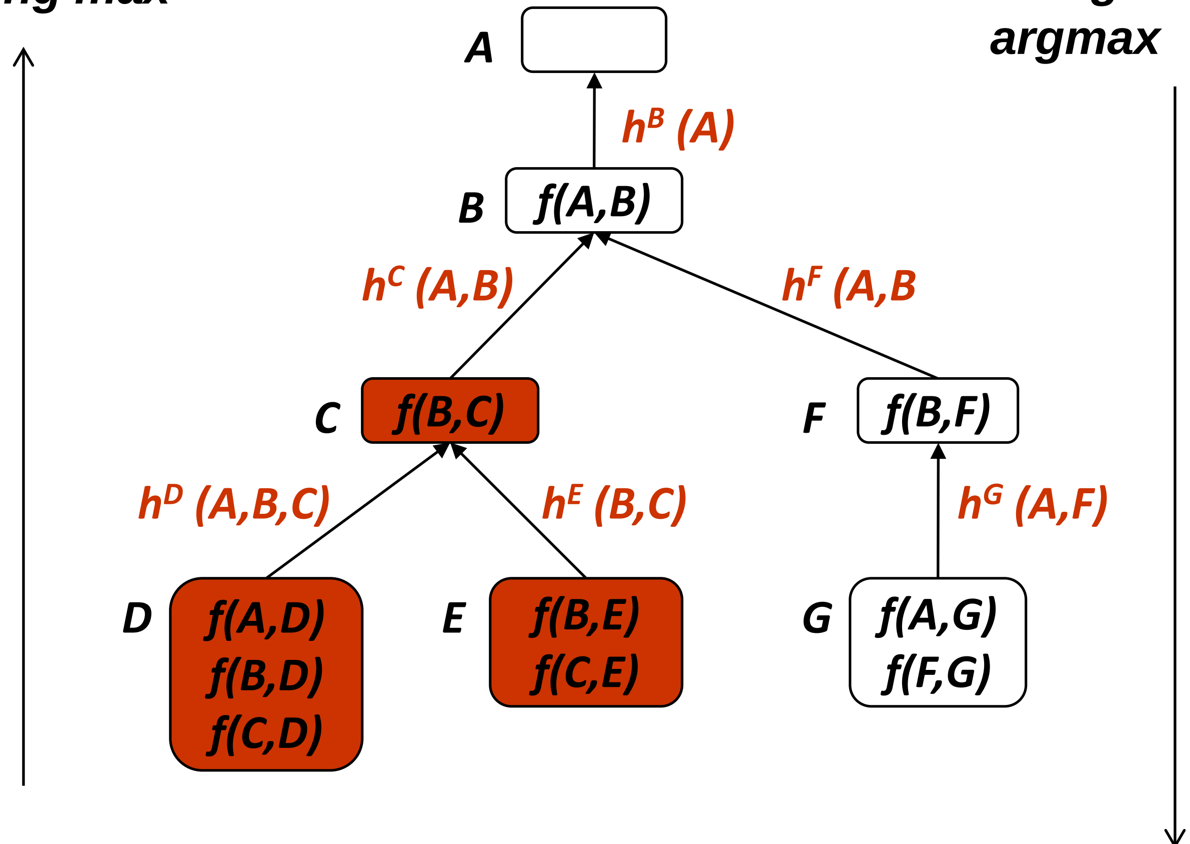
$$\max_{a,b,c,d,e,f,g} f(a,b)+f(a,d)+f(b,c)+f(a,d)+f(b,d)+f(c,d)+f(b,e)+f(c,e)+f(b,f)+f(a,g)+f(f,g)=$$





Messages
Finding max

Assignment
argmax



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

Outline

- Graphical models, Queries, Algorithms
- Inference Algorithms: bucket-elimination
- Bounded Inference: a) mini-bucket, b) cost-shifting
- AND/OR search spaces and AND/OR BnB
- Evaluation, Software
- Conclusions

Bounding approximations:
Generating a heuristic evaluation function

- ***The mini-bucket scheme***
- ***The cost-shifting or re-parameterization scheme***
- ***Combining the two***

Mini-Bucket Approximation

Split a bucket into mini-buckets => bound complexity

bucket (X) =

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

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

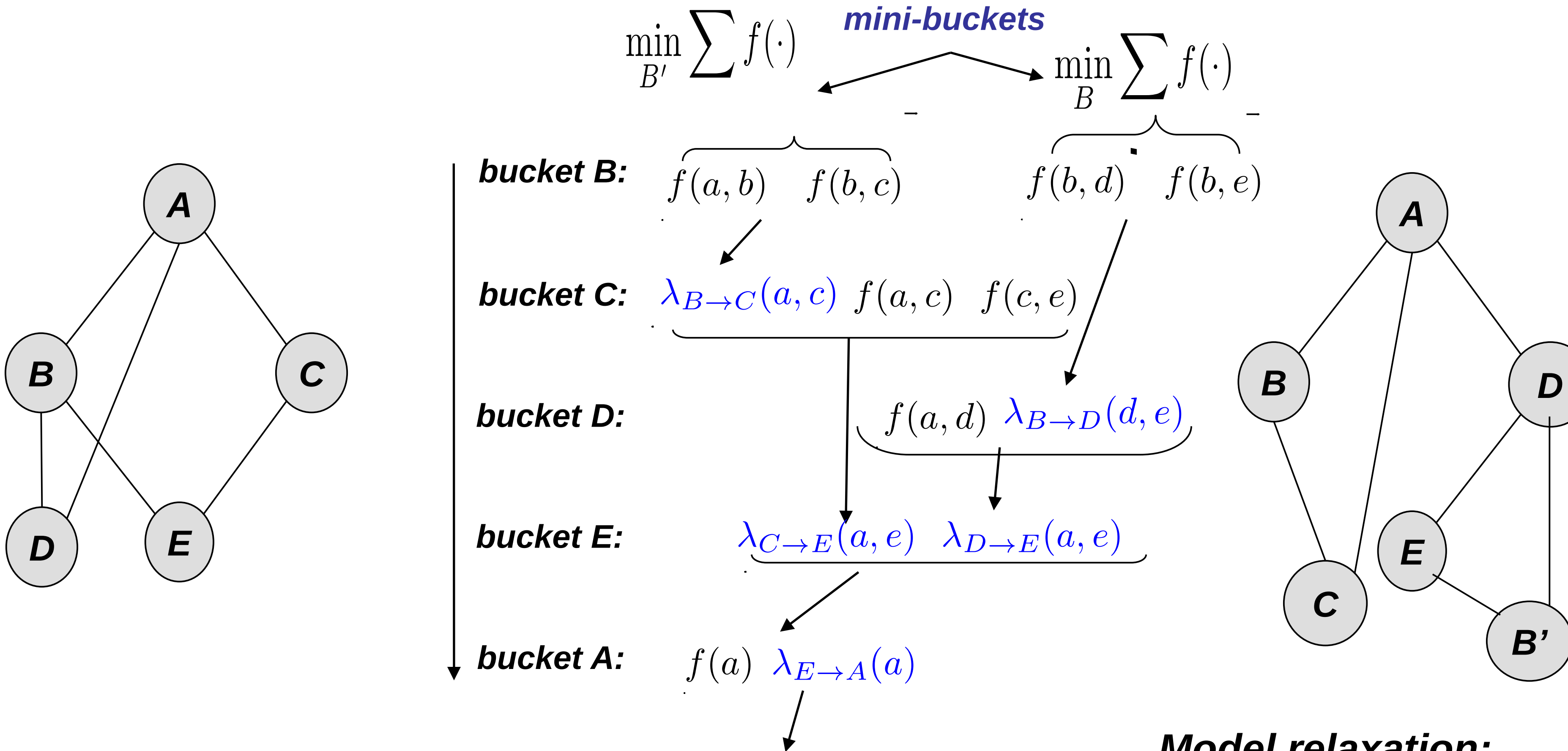
$$\{ f_1, \ldots, f_r \} \qquad \{ f_{r+1}, \ldots, f_n \}$$

$$\lambda'_X(\cdot) = \left(\min_x \sum_{i=1}^{r} f_i(\cdot)\right) + \left(\min_x \sum_{i=r+1}^{n} f_i(\cdot)\right)$$

$$\lambda'_X(\cdot) \;\le\; \lambda_X(\cdot)$$

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

Mini-Bucket Elimination



L = lower bound

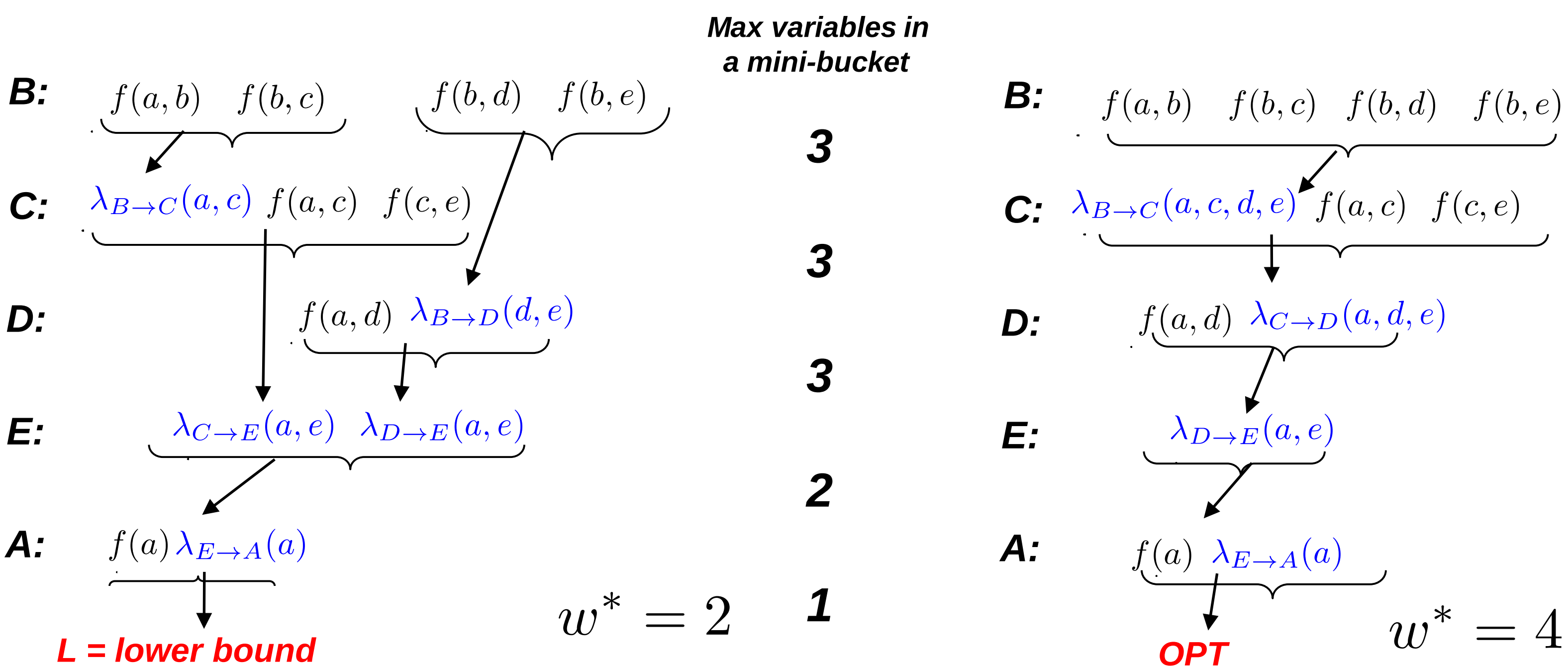
Model relaxation:
[Kask et al., 2001]
[Geffner et al., 2007]
[Choi et al., 2007]
[Johnson et al. 2007]

[Dechter and Rish, 1997; 2003]

MBE-MPE(i) vs BE-MPE

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

MBE-MPE(3) ***versus*** ***BE-MPE***



[Dechter and Rish, 1997; 2003]

Mini-Bucket Decoding

$$\mathbf{b}^* = \arg\min_{\mathbf{b}} f(a^*, b) + f(b, c^*) + f(b, d^*) + f(b, e^*)$$

$$\mathbf{c}^* = \arg\min_{\mathbf{c}} f(c, a^*) + f(c, e^*) + \lambda_{B\to C}(a^*, d^*, c, e^*)$$

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

$$\mathbf{e}^* = \arg\min_{\mathbf{e}} \lambda_{D\to E}(a^*, e)$$

$$\mathbf{a}^* = \arg\min_{\mathbf{a}} f(a) + \lambda_{E\to A}(a)$$

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

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

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

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

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

Return: $(\ \mathbf{a}^*, \mathbf{b}^*, \mathbf{c}^*, \mathbf{d}^*, \mathbf{e}^*\ )$

Greedy configuration = upper bound

L = lower bound

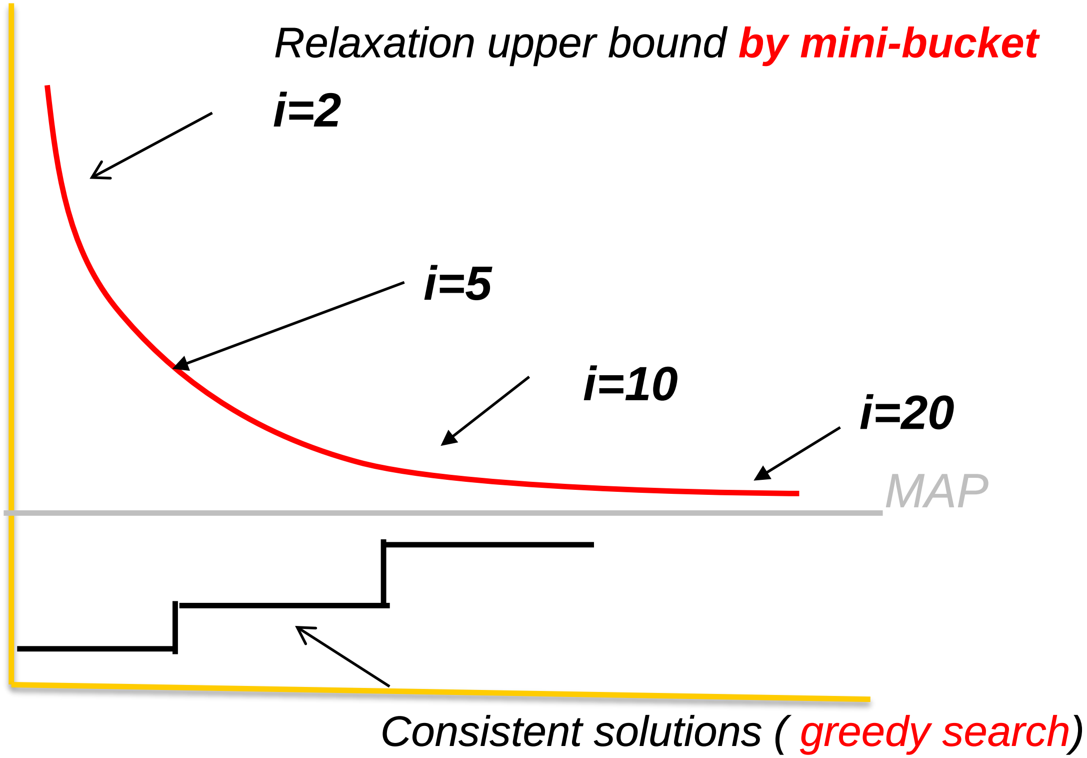
[Dechter and Rish, 1997, 2003]

Properties of MBE(i)

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

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

Bounding from above and below



How to partition given an i-bound?

- Scope-based greedy
- Content-based greedy
- (Rollon & Dechter 2010)

Relaxation provides upper bound
Any configuration: lower bound

Outline

- Graphical models, Queries, Algorithms
- Inference Algorithms: bucket-elimination
- Bounded Inference: a) mini-bucket, b) cost-shifting
- AND/OR search spaces and AND/OR BnB
- Evaluation, Software
- Conclusions

Cost-Shifting

(Reparameterization)

$+\lambda(B)$

A	B	f(A,B)
b	b	6 + 3
b	g	0 - 1
g	b	0 + 3
g	g	6 - 1

+

$-\lambda(B)$

B	C	f(B,C)
b	b	6 - 3
b	g	0 - 3
g	b	0 + 1
g	g	6 + 1

B	λ(B)
b	3
g	-1

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

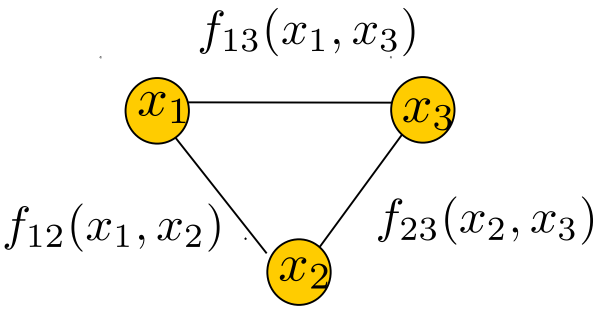
= 0 + 6

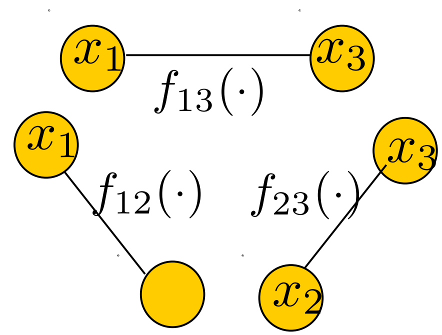
Modify the individual functions

- but -

keep the sum of functions unchanged

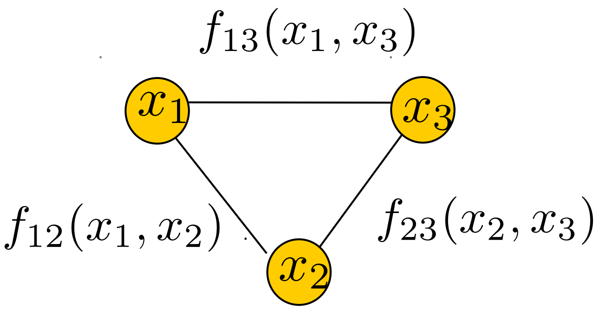
Dual Decomposition

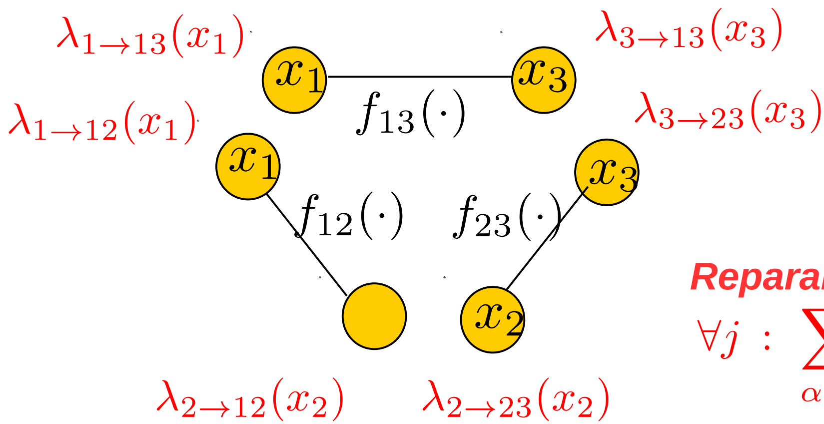




$$F^* = \min_x \sum_\alpha f_\alpha(x) \quad \geq \quad \sum_\alpha \min_x \; f_\alpha(x)$$

Dual Decomposition





Reparameterization:

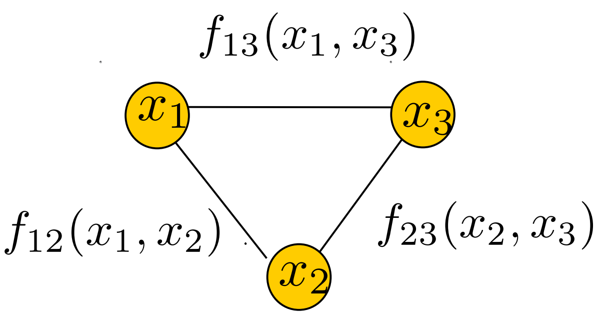
$$\forall j \; : \; \sum_{\alpha \ni j} \lambda_{j\to\alpha}(x_j) = 0$$

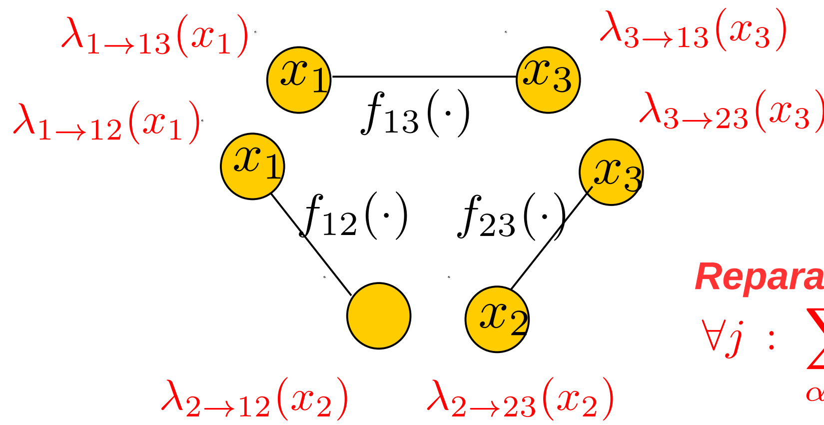
$$F^* = \min_x \sum_\alpha f_\alpha(x) \qquad \geq \max_{\lambda_{i\to\alpha}} \sum_\alpha \min_x \left[f_\alpha(x) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \right]$$

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

(Convex dual: linear programming relaxation)

Dual Decomposition





Reparameterization:

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

$$F^* = \min_x \sum_\alpha f_\alpha(x) \qquad \geq \max_{\lambda_{i\to\alpha}} \sum_\alpha \min_x \Big[f_\alpha(x) + \sum_{i\in\alpha} \lambda_{i\to\alpha}(x_i) \Big]$$

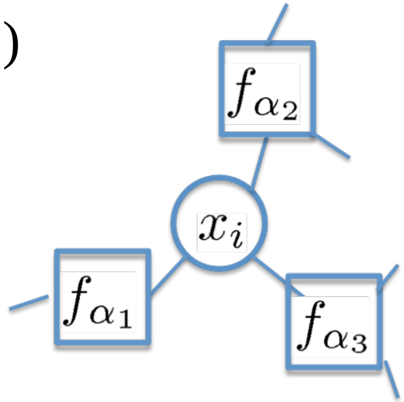
Many names for the same class of bounds:

- Dual decomposition [Komodakis et al. 2007]
- TRW, MPLP [Wainwright et al. 2005, Globerson & Jaakkola 2007]
- Soft arc consistency [Cooper & Schiex 2004]
- Max-sum diffusion [Warner 2007]

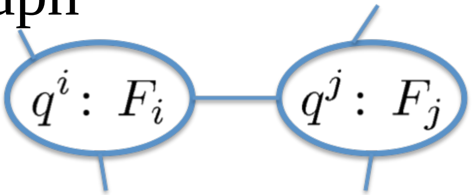
(Convex dual: linear programming relaxation)

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

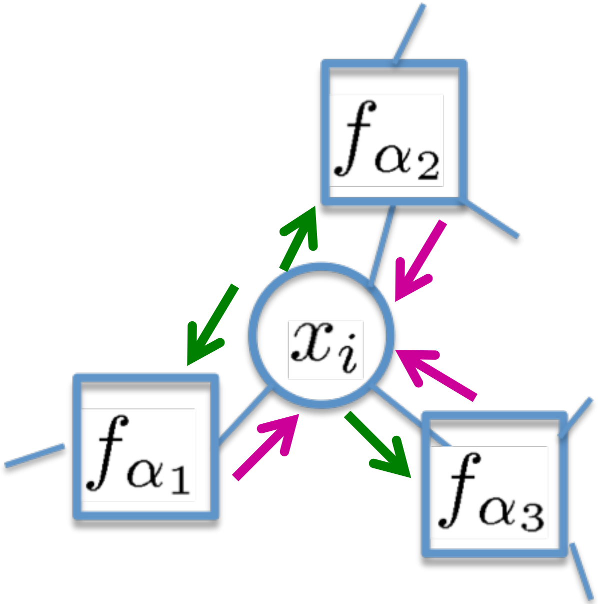


- MBE-MM Update within each bucket only
  - Apply cost-shifting within each bucket only

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:

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



MBE-MM: MBE with moment matching

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

$max_B \Pi$ m_{11} $max_B \Pi$

Bucket B   P(E|B,C) ⟷ P(B|A) P(D|A,B)   (m_{11} →, ← m_{12})

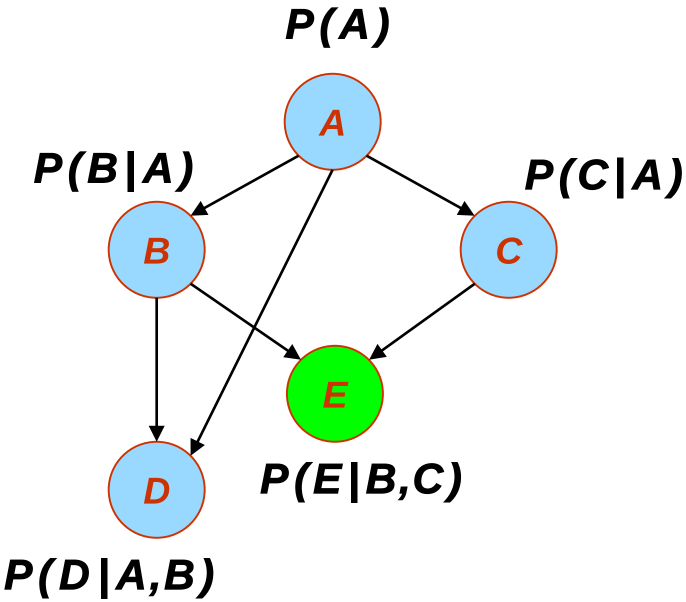
Bucket C   P(C|A) h^B (C,E)

Bucket D h^B (A,D)

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

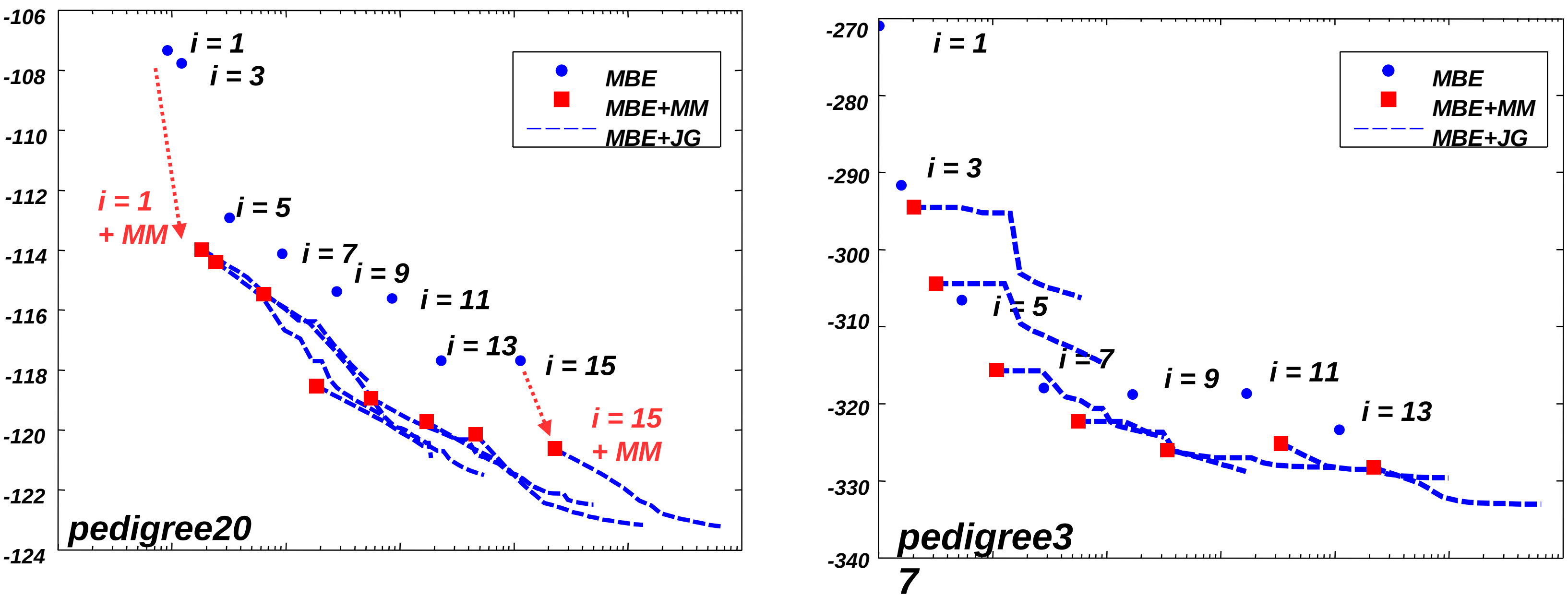
Bucket A   P(A) h^E (A) h^D (A)

W=2

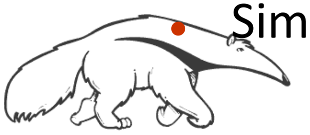


MPE is an upper bound on MPE --U*
Generating a solution yields a lower bound--L

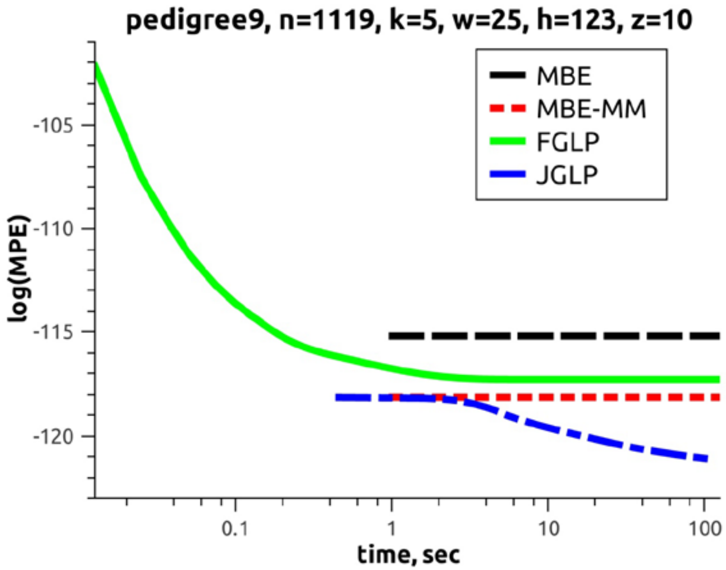
Anytime Approximation

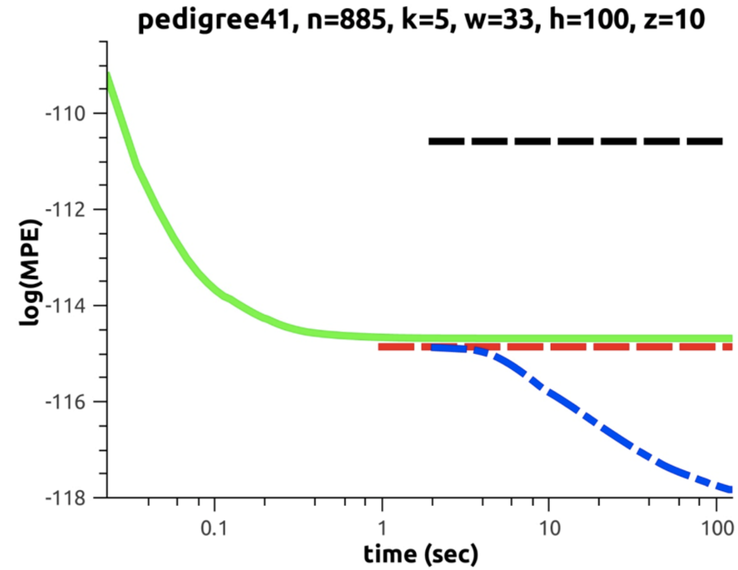


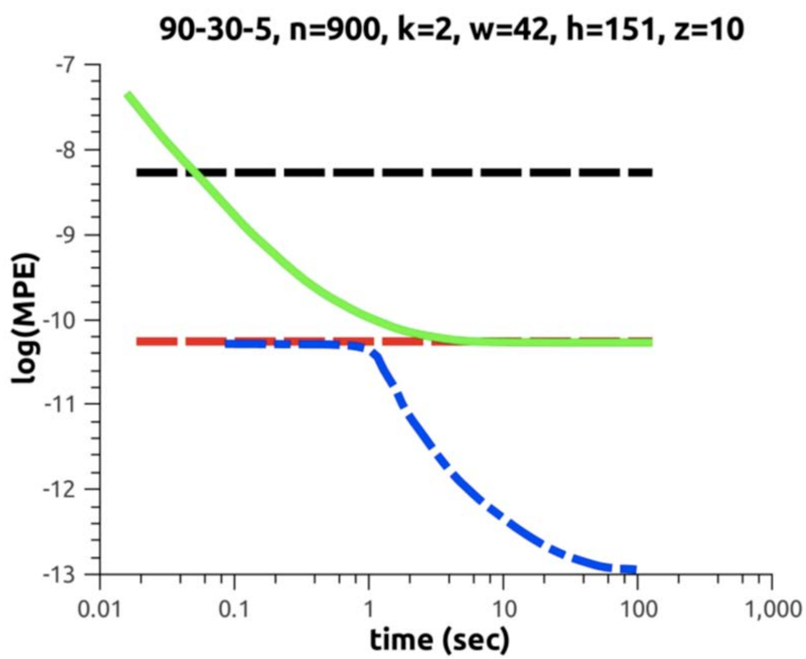
- Can tighten the bound in various ways
  - Cost-shifting (improve consistency between cliques)
  - Increase i-bound (higher order consistency)
- Simple moment-matching step improves bound significantly

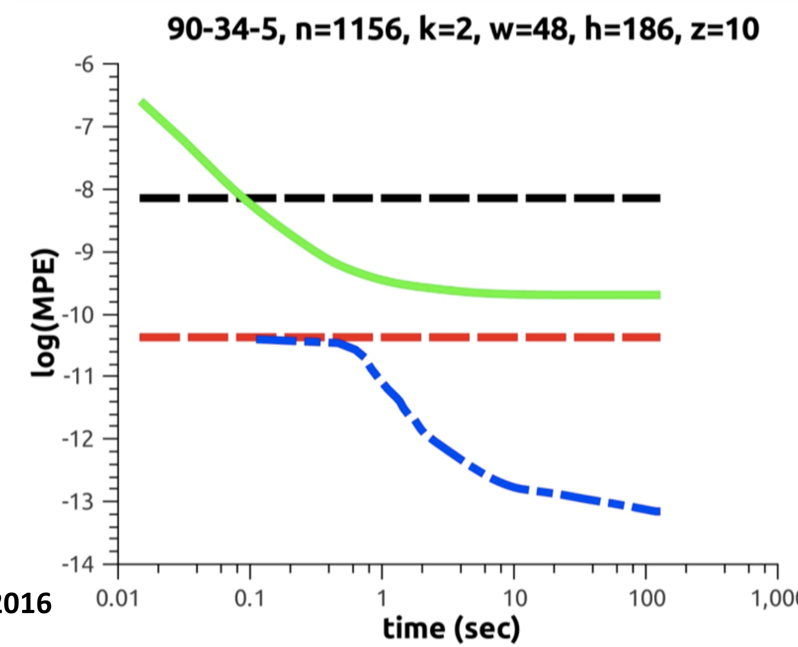


Iterative tightening as bounding schemes









Outline

- Graphical models, Queries, Algorithms
- Inference Algorithms: bucket-elimination
- Bounded Inference: a) mini-bucket, b) cost-shifting
- Generating heuristics using mini-bucket elimination
- AND/OR search spaces and AND/OR BnB
- Evaluation, Software
- Conclusions

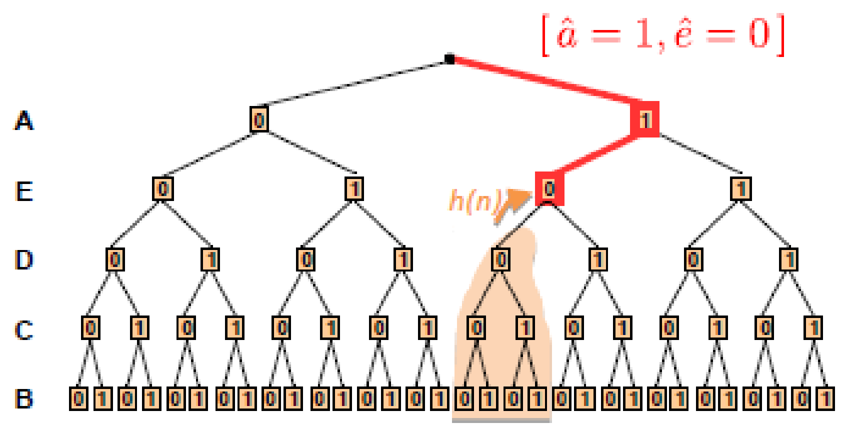
Generating Heuristics for Graphical Models

Given a cost function:

$$f(a,\ldots,e) = f(a) + f(a,b) + f(a,c) + f(a,d) + f(b,c) + f(b,d) + f(b,e) + f(c,e)$$

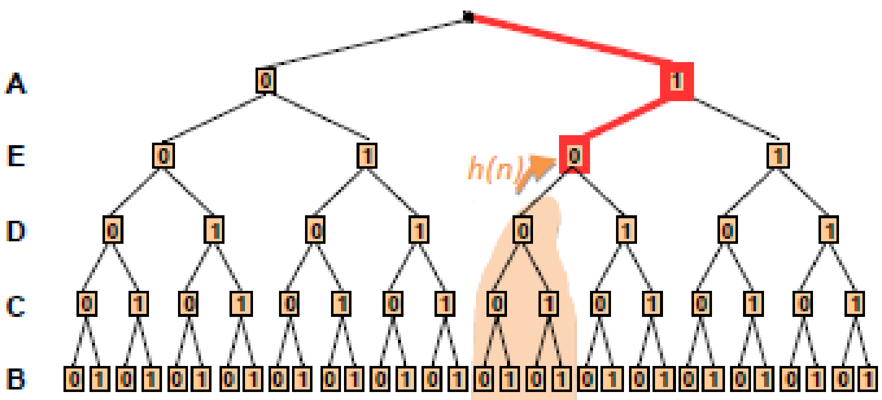
define an evaluation function over a partial assignment as the cost of its best extension:

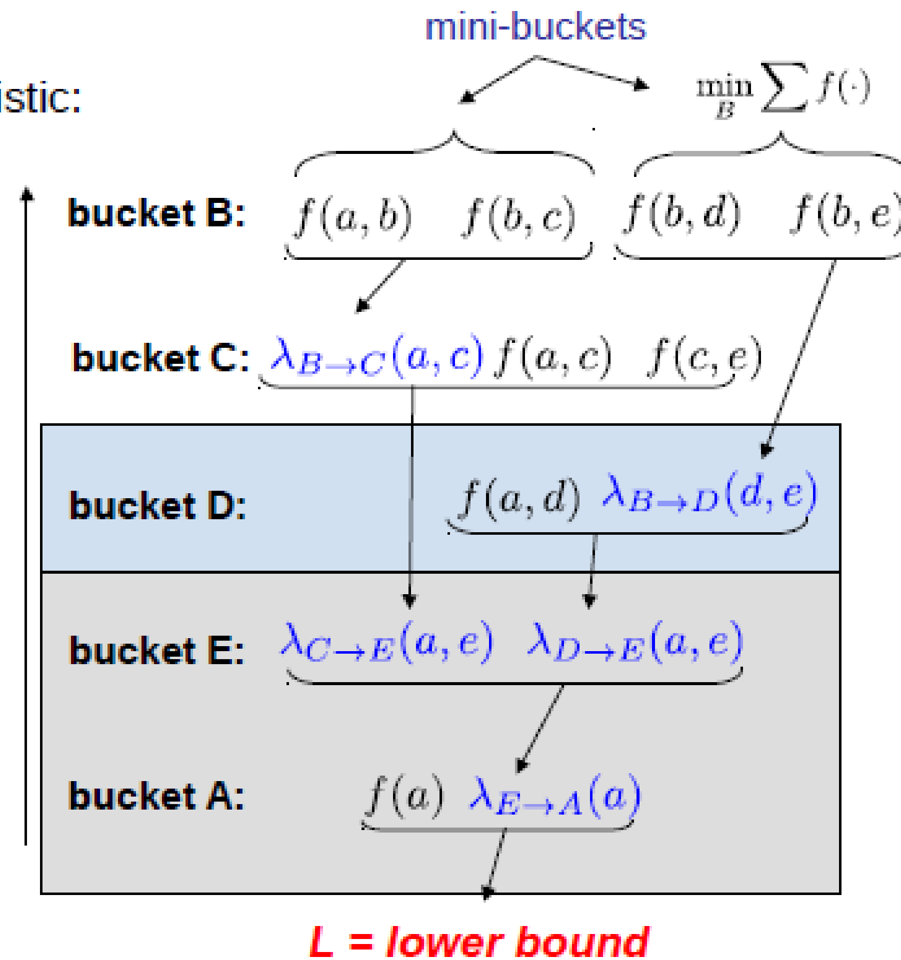
$$\begin{aligned} f^*(\hat{a},\hat{e},D) &= \min_{b,c} F(\hat{a},b,c,D,\hat{e}) \\ &= \underbrace{f(\hat{a})}_{} + \underbrace{\min_{b,c} f(\hat{a},b) + f(\hat{a},c) + \ldots}_{} \\ &= g(\hat{a},\hat{e},D) + h^*(\hat{a},\hat{e},D) \end{aligned}$$



Static Mini-Bucket Heuristics

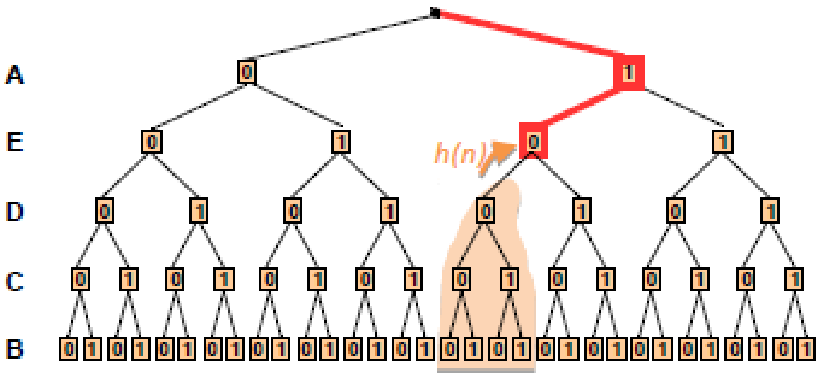
Given a partial assignment, $[\hat{a} = 1, \hat{e} = 0]$
(weighted) mini-bucket gives an admissible heuristic:

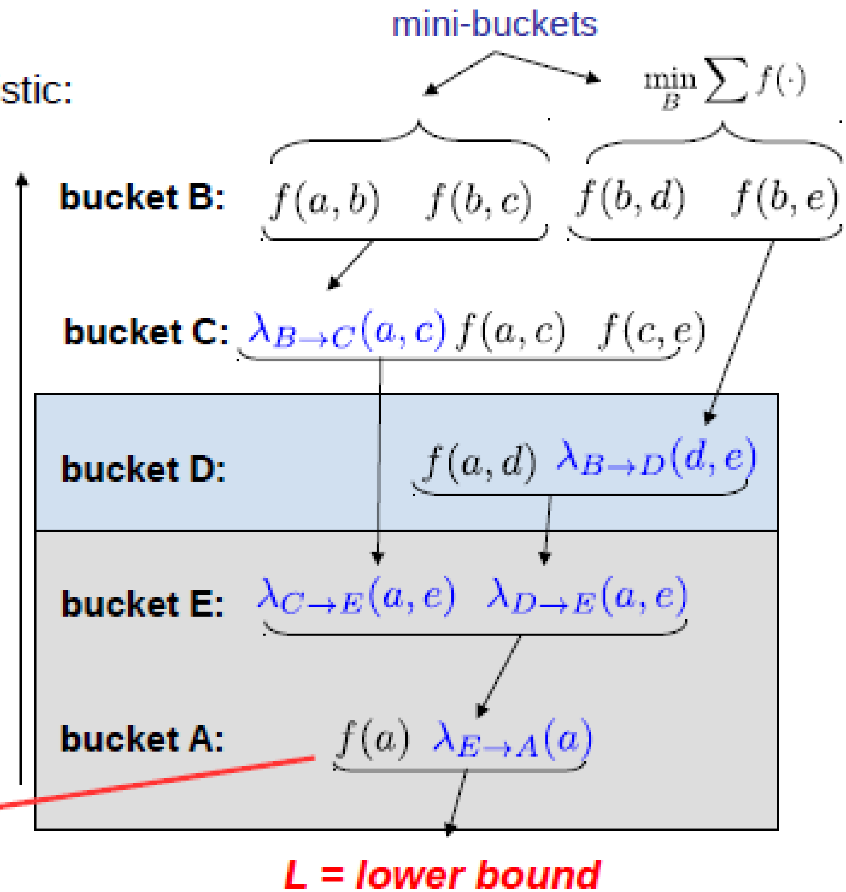




Static Mini-Bucket Heuristics

Given a partial assignment, $[\hat{a}=1, \hat{e}=0]$
(weighted) mini-bucket gives an admissible heuristic:

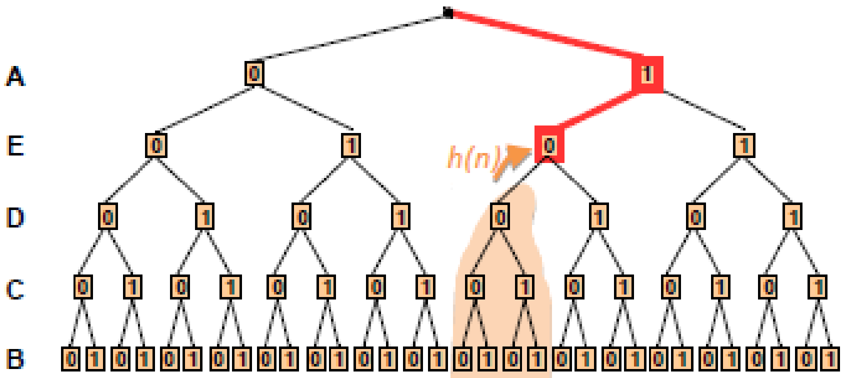


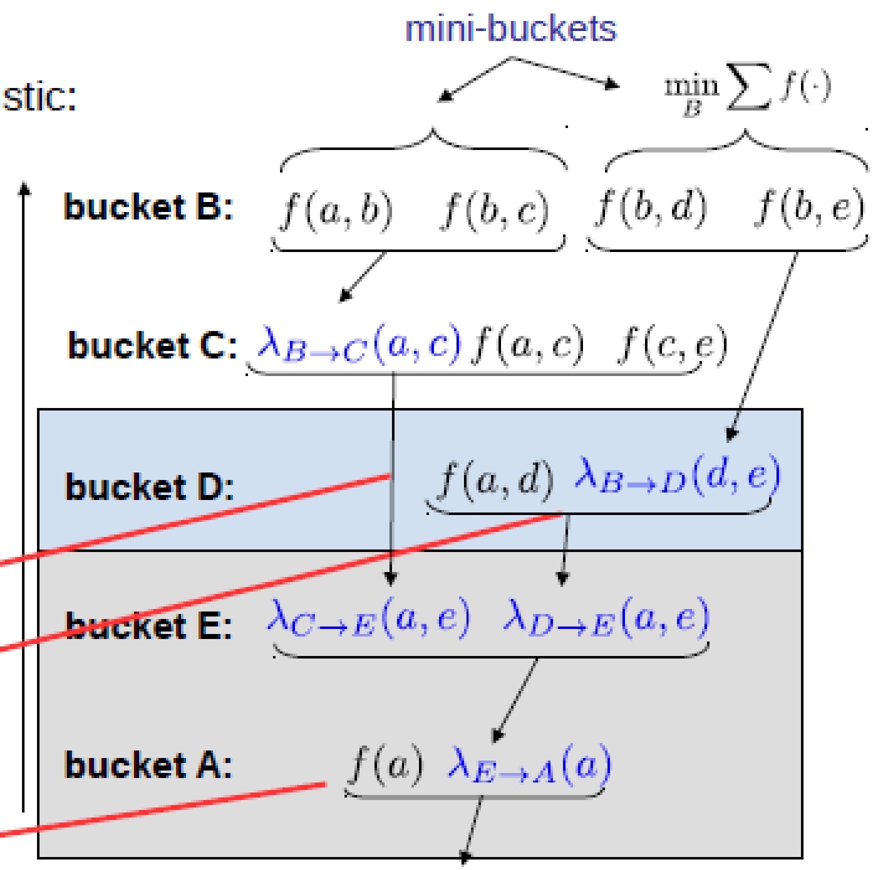


cost so far:
$g(\hat{a},\hat{e}) = f(A=\hat{a})$

Static Mini-Bucket Heuristics

Given a partial assignment, $[\hat{a}=1, \hat{e}=0]$
(weighted) mini-bucket gives an admissible heuristic:





L = lower bound

cost to go:

$$\tilde{h}(\hat{a},\hat{e},D) = \lambda_{C\to E}(\hat{a},\hat{e}) + f(\hat{a},D) + \lambda_{B\to D}(D,\hat{e})$$

(admissible: $\tilde{h}(\hat{a},\hat{e},D) \le h^*(\hat{a},\hat{e},D)$)

cost so far:

$$g(\hat{a},\hat{e}) = f(A=\hat{a})$$

Properties of the Heuristic

- MB heuristic is monotone, admissible
- Computed in linear time

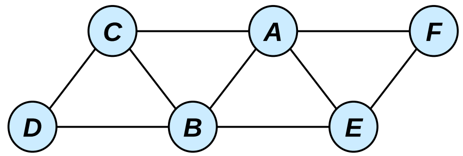
- **IMPORTANT**
  - Heuristic strength can vary by MB(i) & message passing
  - Higher i-bound → more pre-processing
    → more accurate heuristic
    → less search

- Allows controlled trade-off between pre-processing and search

Outline

- Graphical models, Queries , Algorithms
- Inference Algorithms
- Bounded Inference: mini-bucket, cost-shifting
- AND/OR search spaces and AND/OR BnB
- Evaluation, Software
- Conclusions

Classic OR 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**

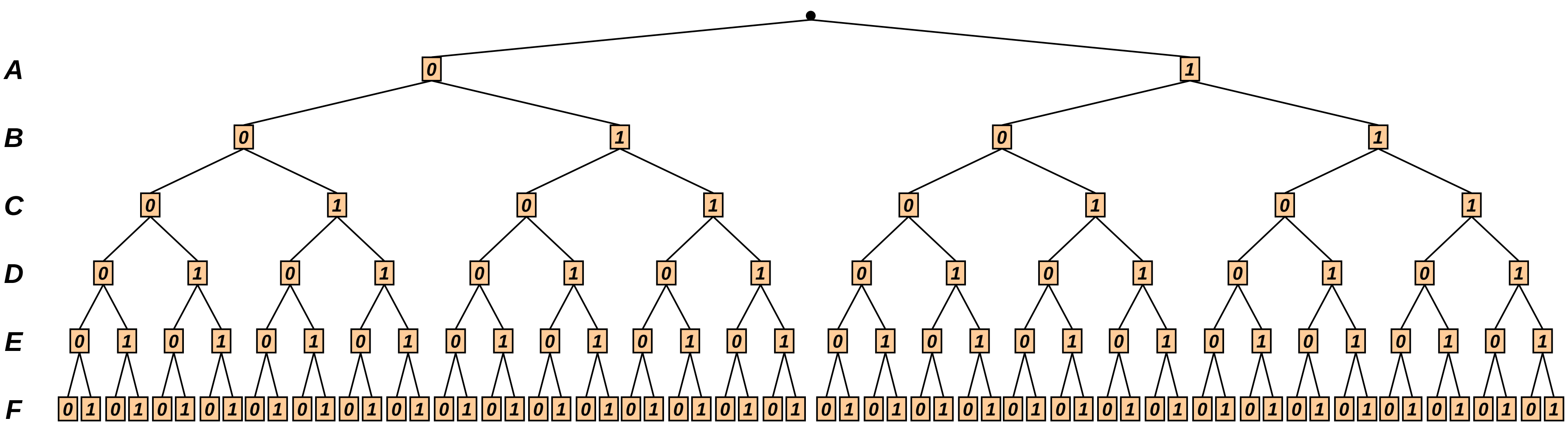
B	D	f_6
0	0	**4**
0	1	**2**
1	0	**1**
1	1	**0**

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

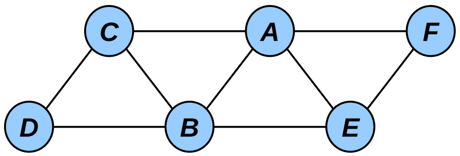
C	D	f_8
0	0	**1**
0	1	**4**
1	0	**0**
1	1	**0**

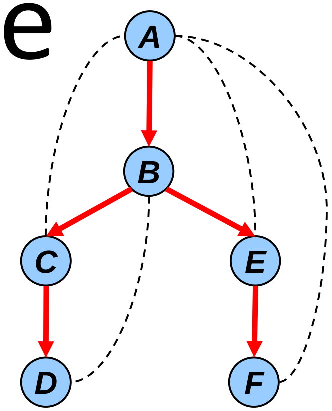
E	F	f_9
0	0	**1**
0	1	**0**
1	0	**0**
1	1	**2**

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

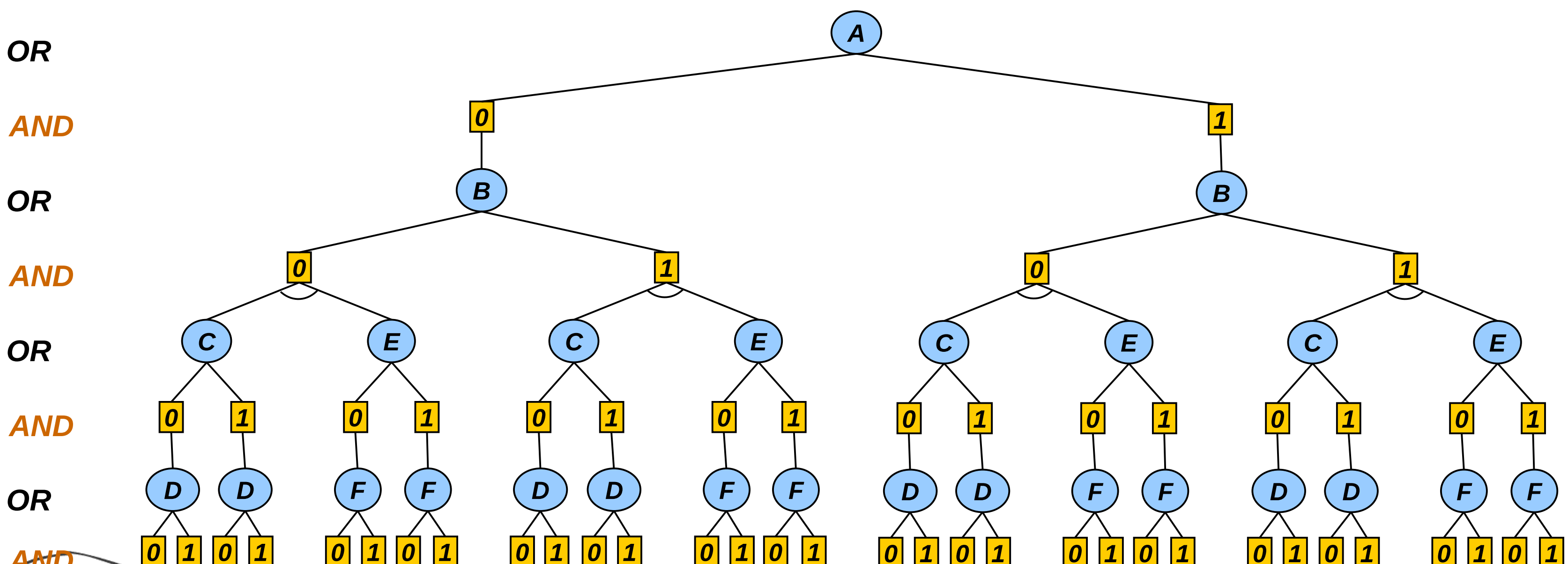


The AND/OR Search Tree

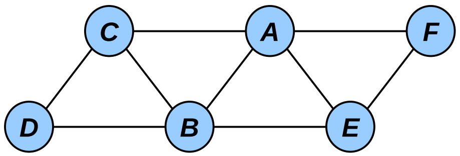


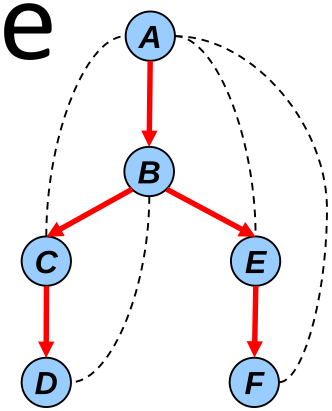


Pseudo tree ***(Freuder & Quinn85)***

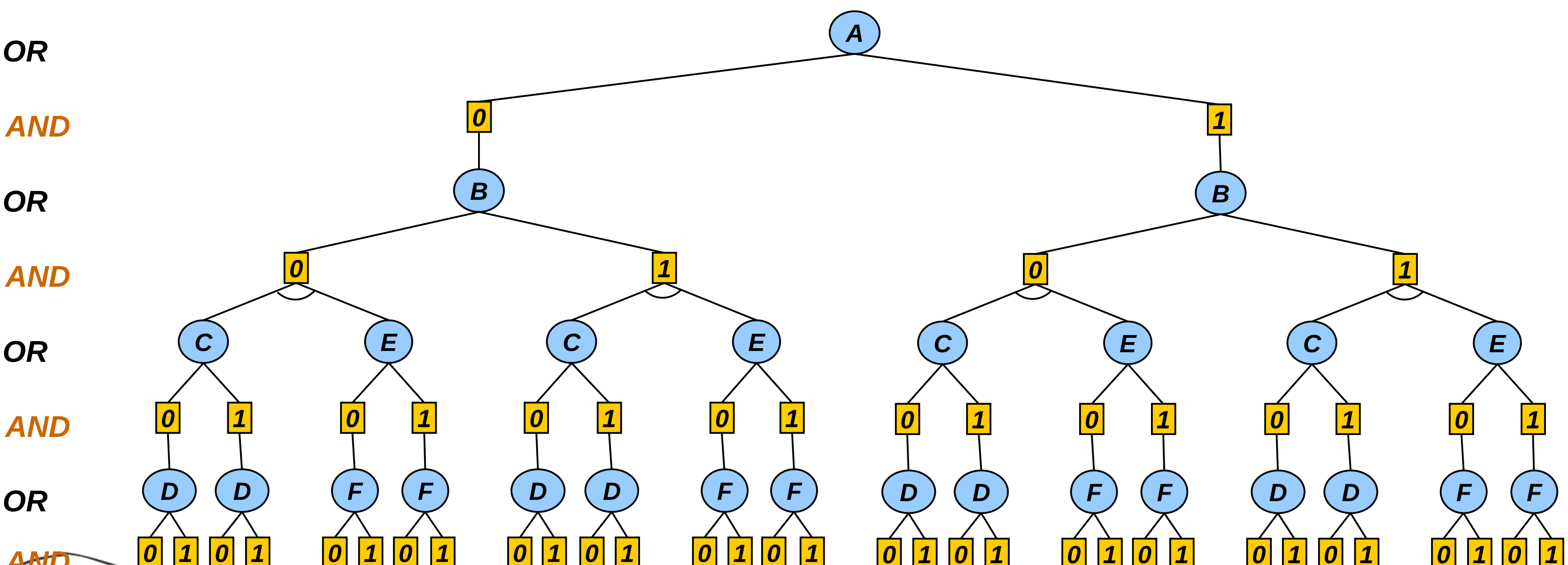


The AND/OR Search Tree

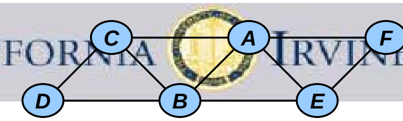




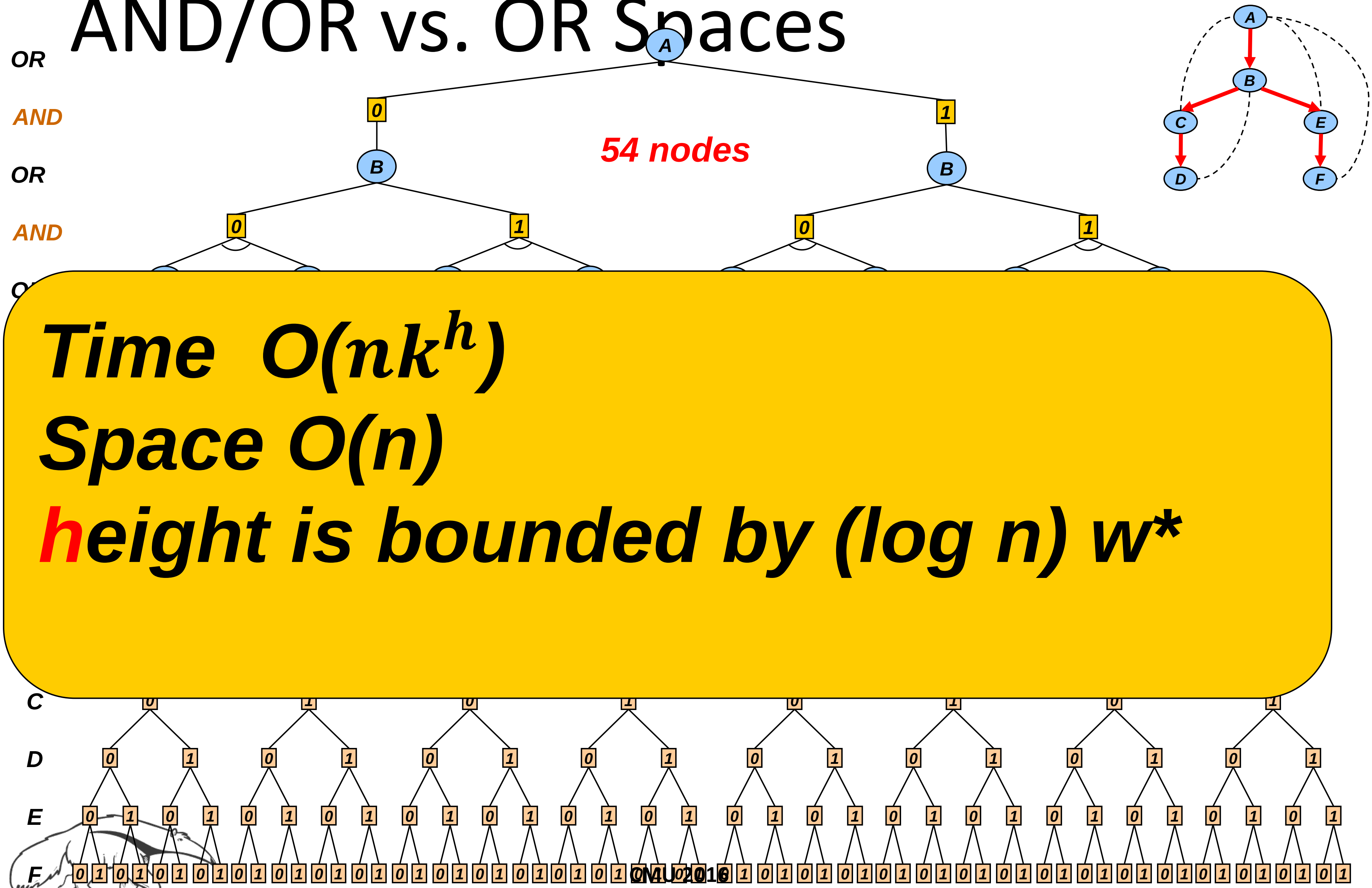
Pseudo tree



A solution subtree is (A=0, B=1, C=0, D=0, E=1, F=1)



AND/OR vs. OR Spaces

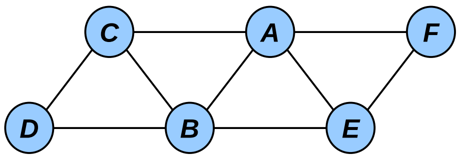


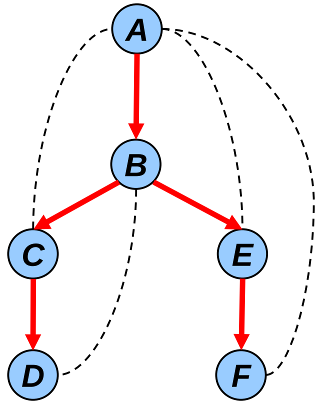
Time O(nk^h)

Space O(n)

height is bounded by (log n) w*

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

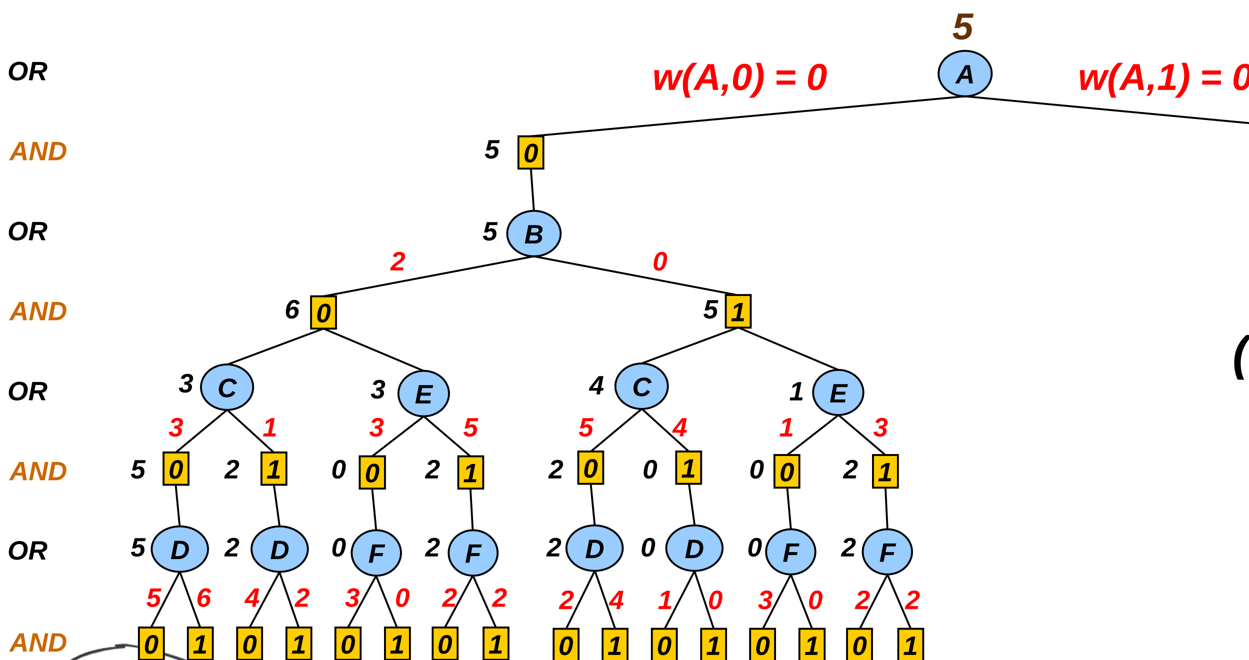
B	D	f_6
0	0	4
0	1	2
1	0	1
1	1	0

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

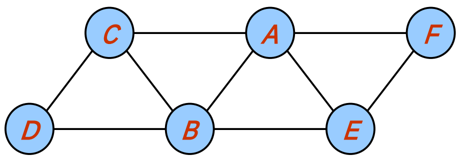
$$f(X) = \min_X \sum_{i=1}^{9} f_i(X)$$

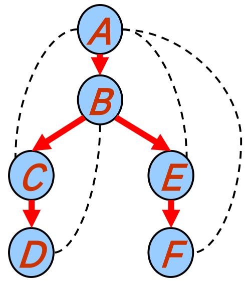


Node Value
(bottom-up evaluation)

OR – minimization
AND – summation

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**

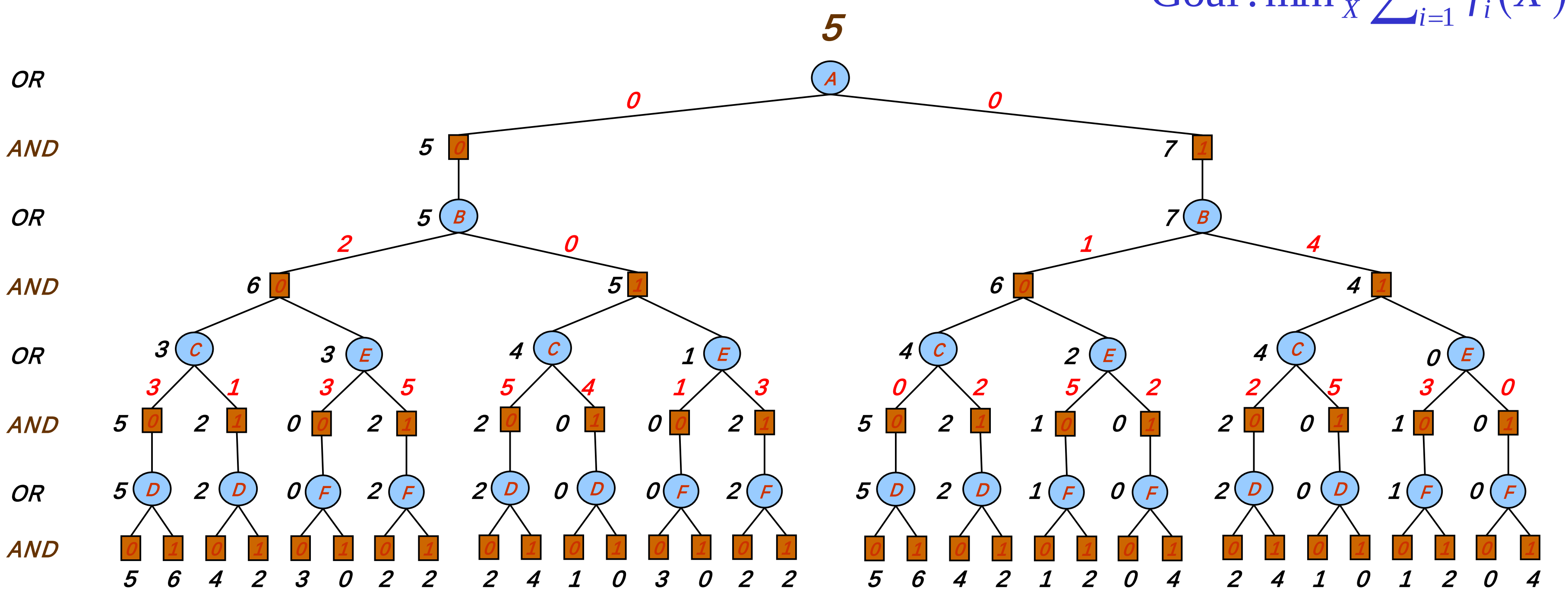
B	D	f_6
0	0	**4**
0	1	**2**
1	0	**1**
1	1	**0**

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**

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

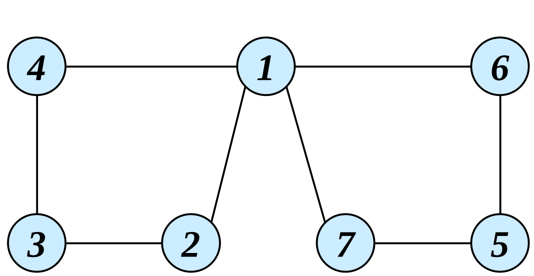


AND node = Combination operator (summation)

OR node = Marginalization operator (minimization)

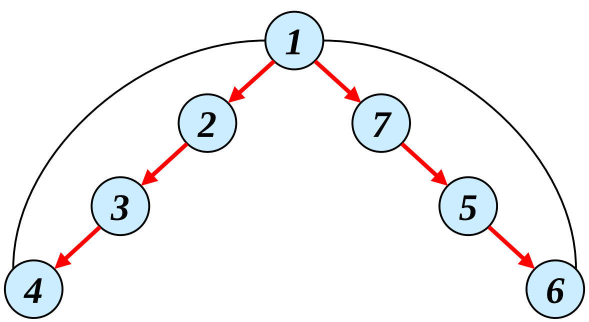
Pseudo-Trees

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

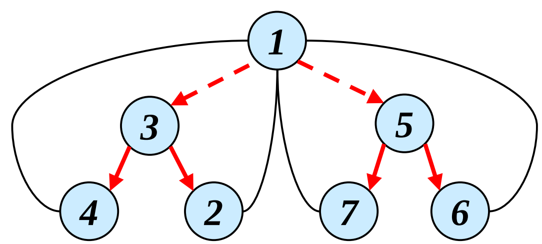


(a) Graph

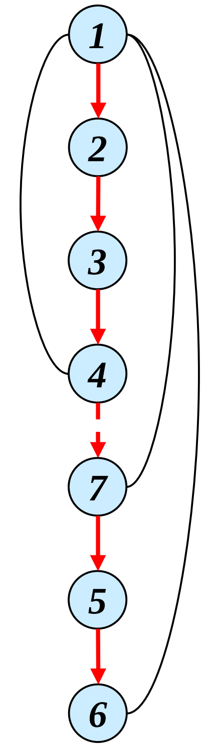
m <= w* log n



(b) DFS tree
depth=3



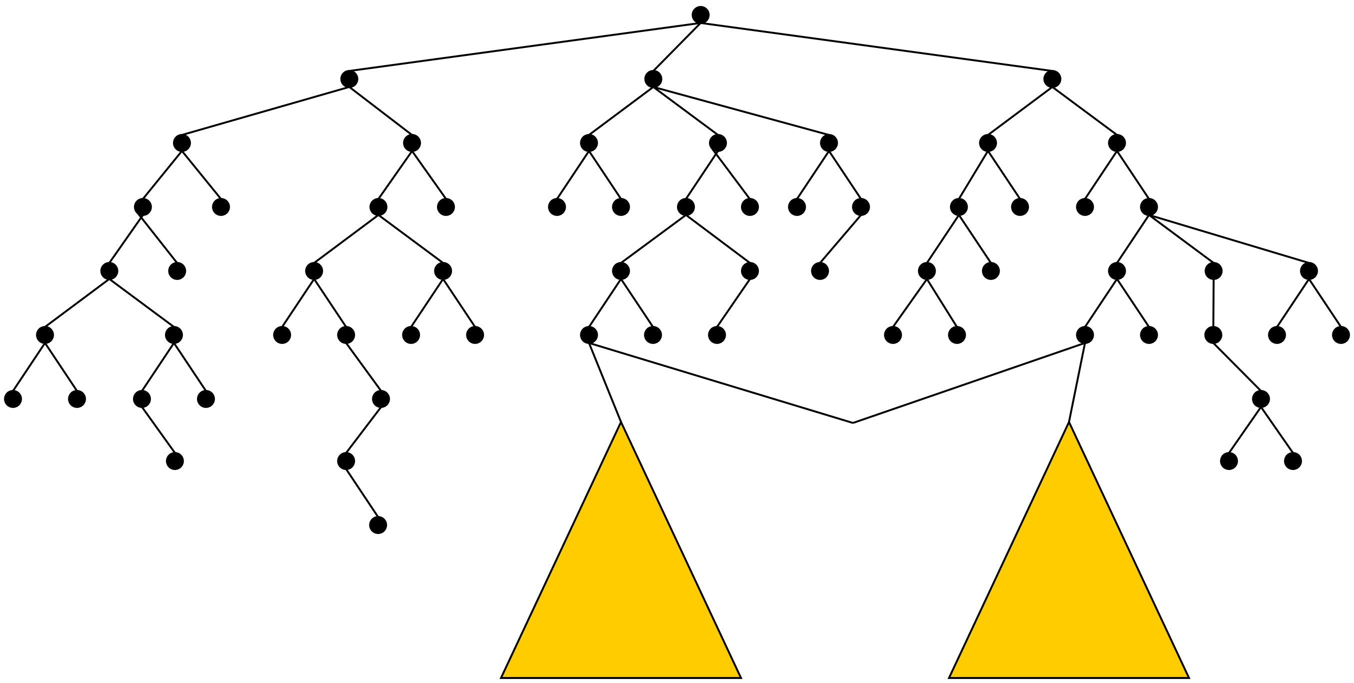
(c) pseudo- tree
depth=2



(d) Chain
depth=6

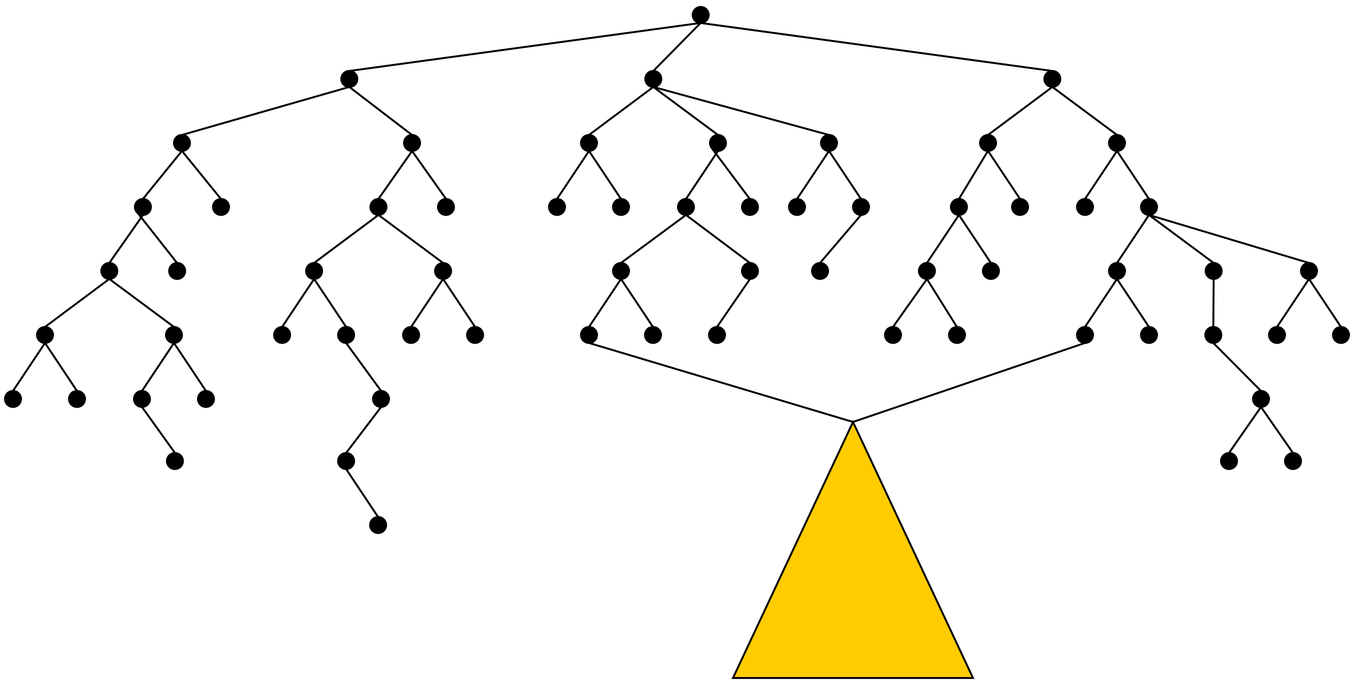
From Search Trees to Search Graphs

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

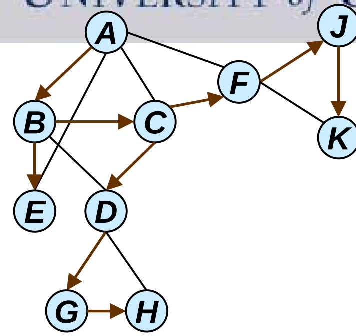


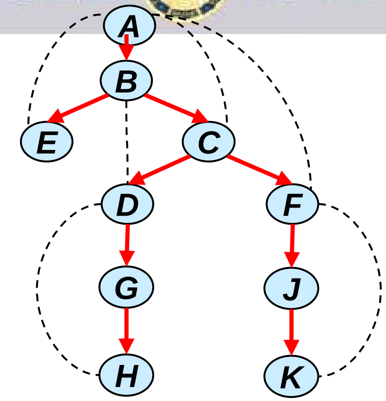
From Search Trees to Search Graphs

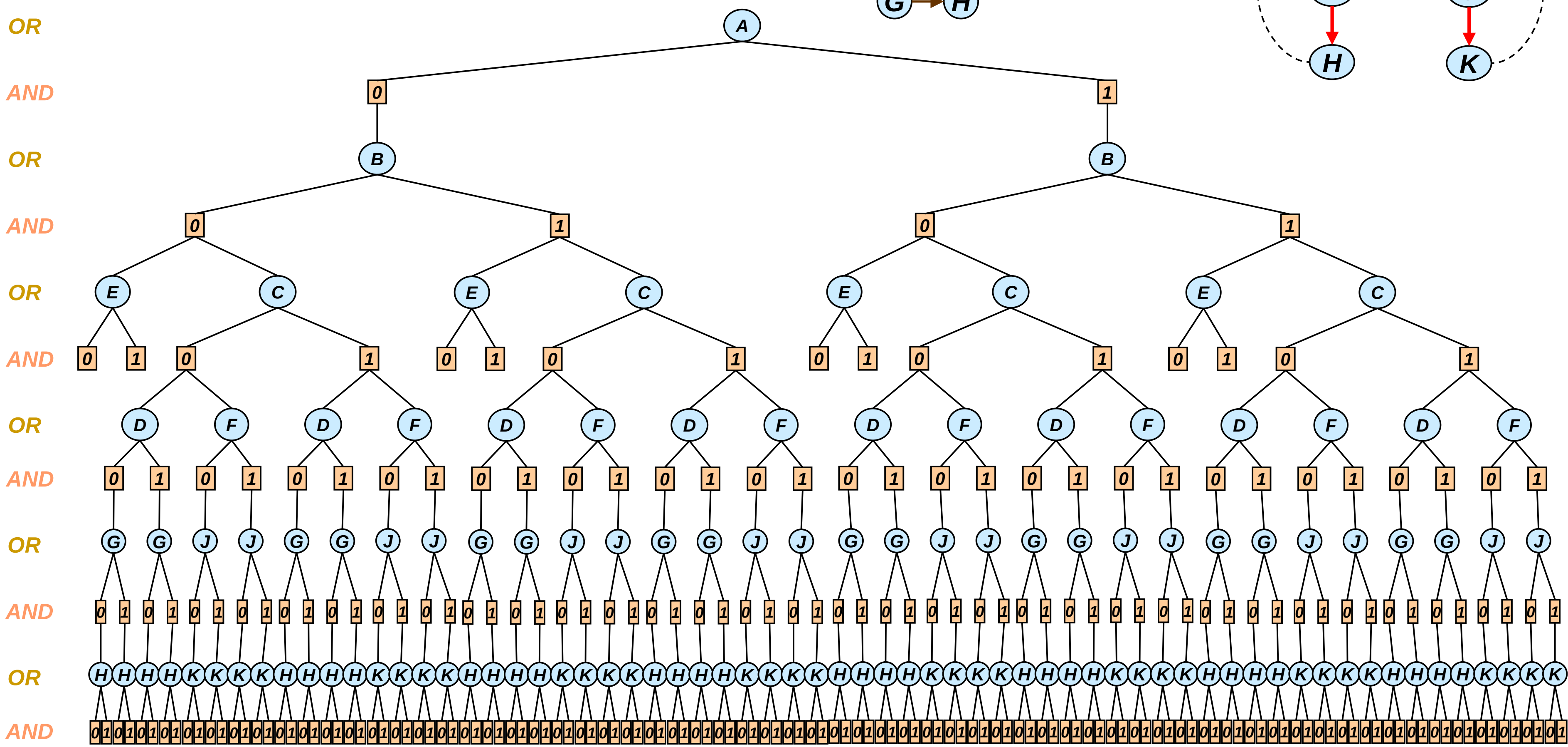
- Any two nodes that root identical sub-trees or sub-graphs can be merged



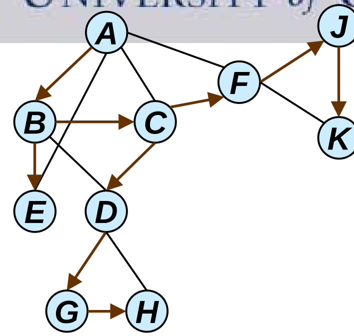
From AND/OR Tree

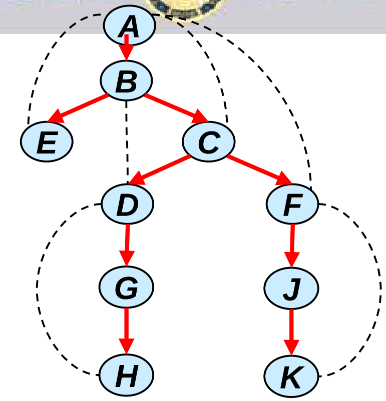


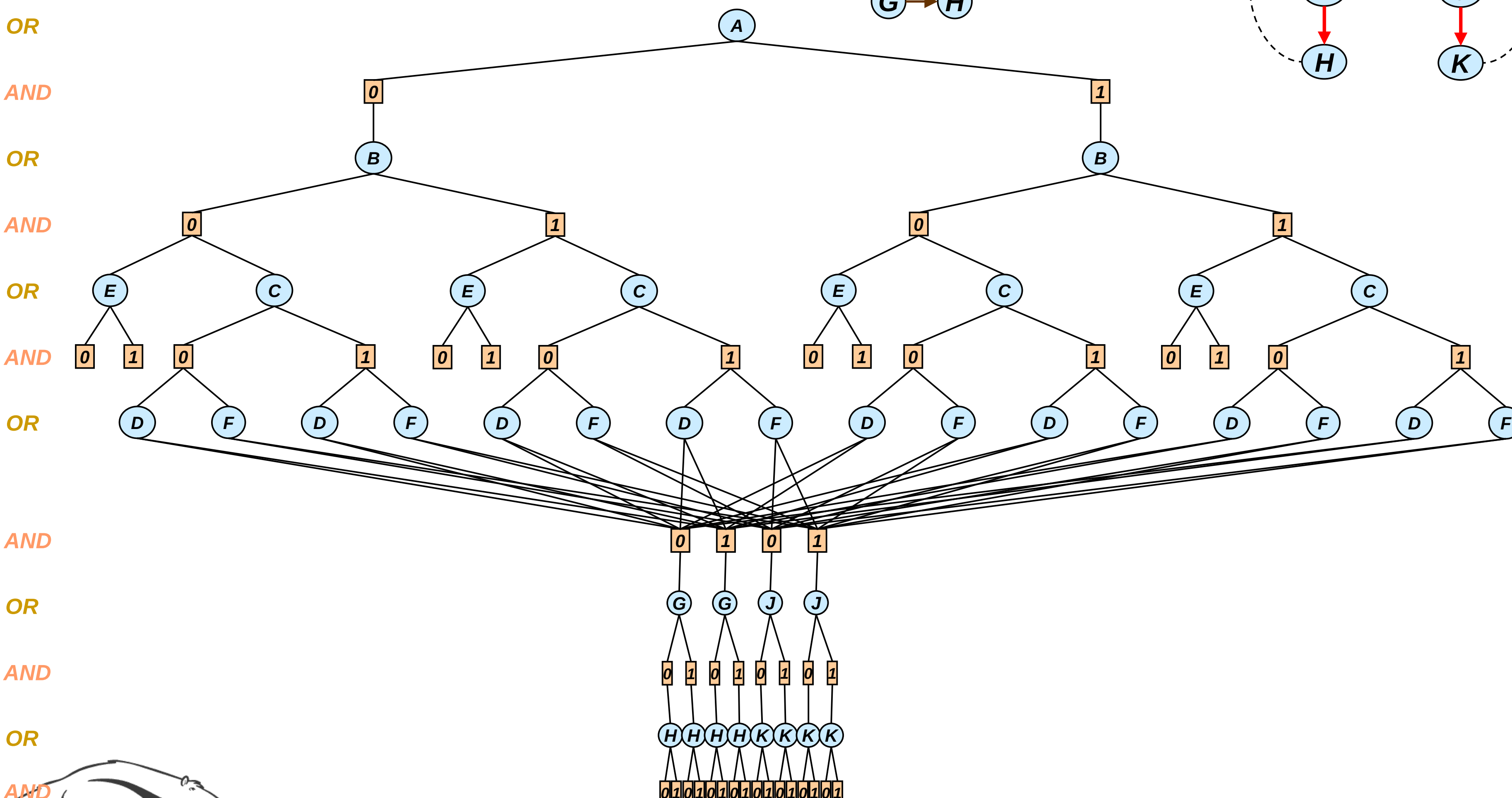




An AND/OR Graph



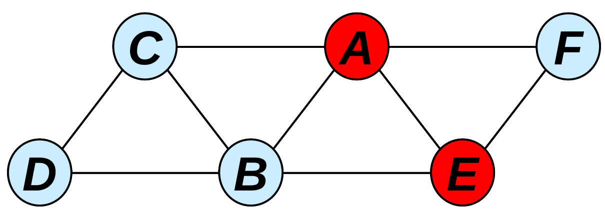


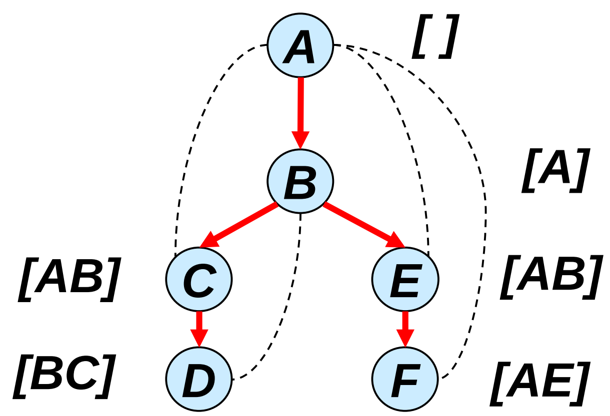


Merging Based on Context

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

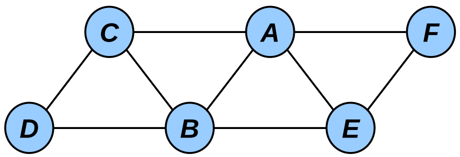
  context(X) = ancestors of X in the pseudo tree that are connected to X, or to descendants of X

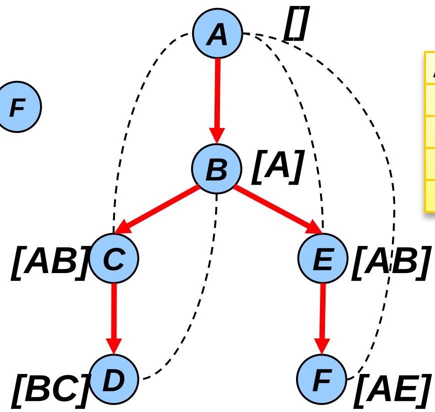




pseudo tree

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

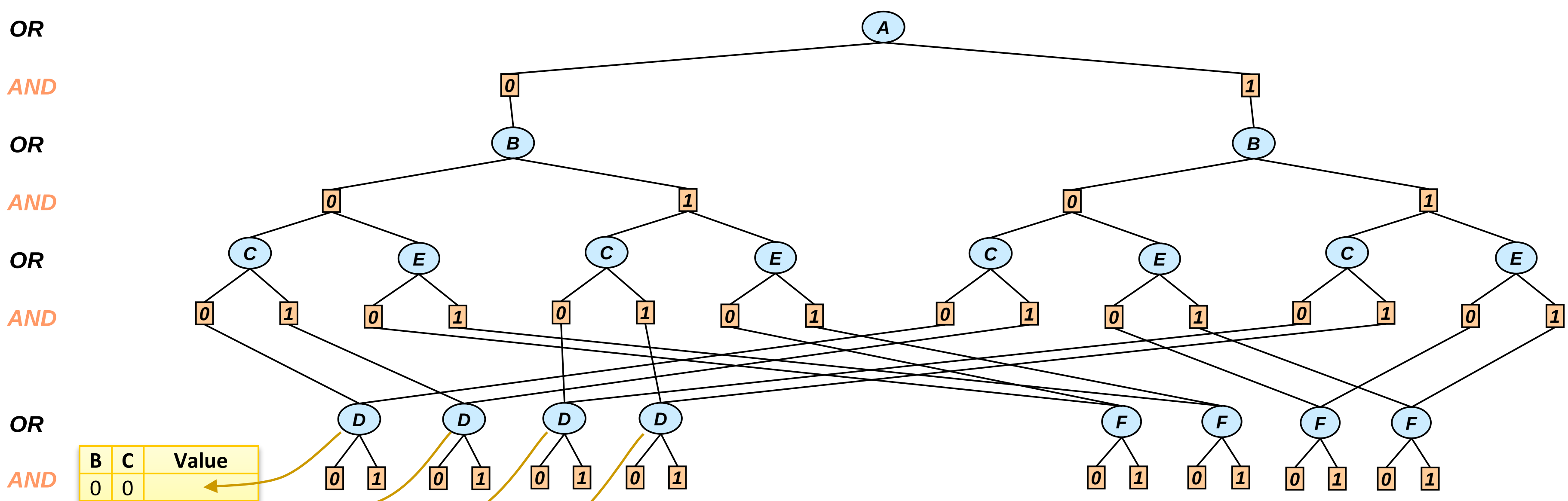
B	D	f_6
0	0	4
0	1	2
1	0	1
1	1	0

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

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



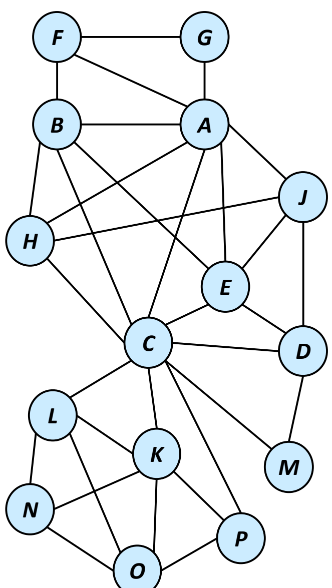
B	C	Value
0	0	
0	1	
1	0	
1	1	

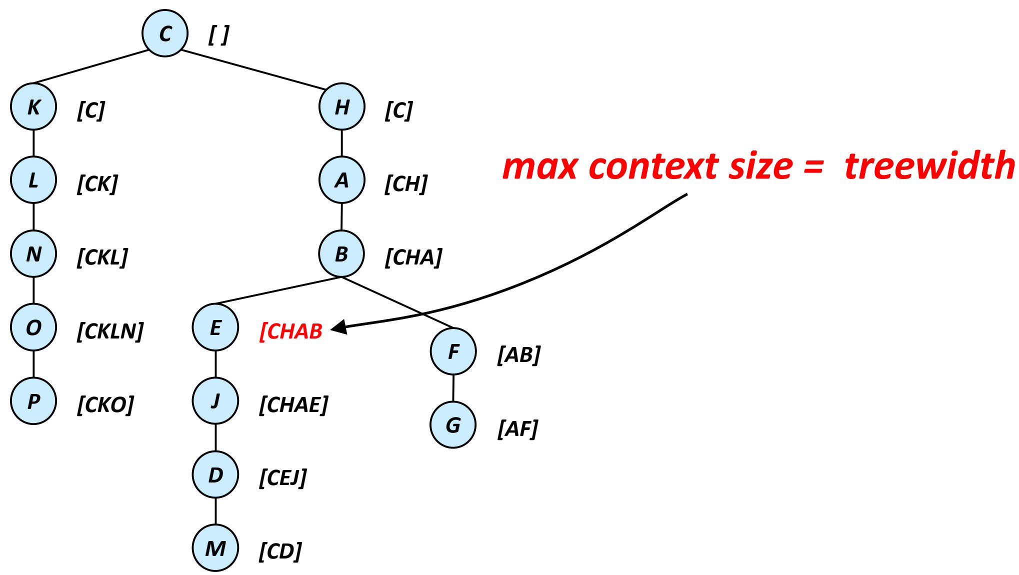
context minimal graph

Cache table for D

How Big Is The Context?

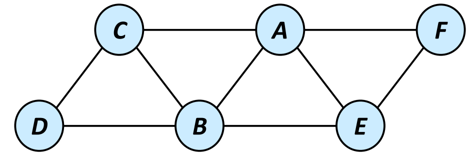
Theorem: The maximum context size for a pseudo tree is equal to the treewidth of the graph along the pseudo tree.

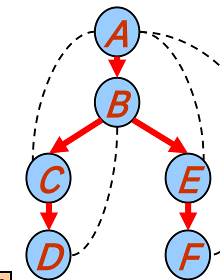




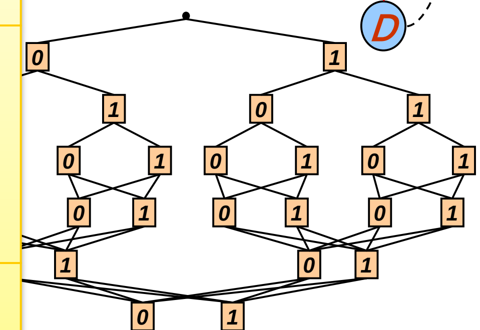
(C K H A B E J L N O D P M F G)

All Four Search Spaces





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

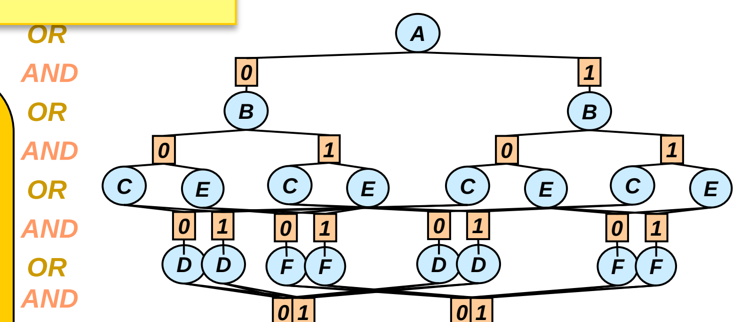


...xt minimal OR search graph

28 nodes

Computes any query:

- *Constraint satisfaction*
- *Optimization*
- *Weighted counting*



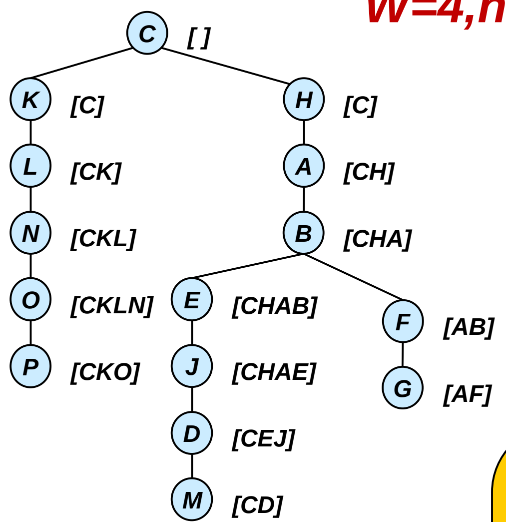
Context minimal AND/OR search graph

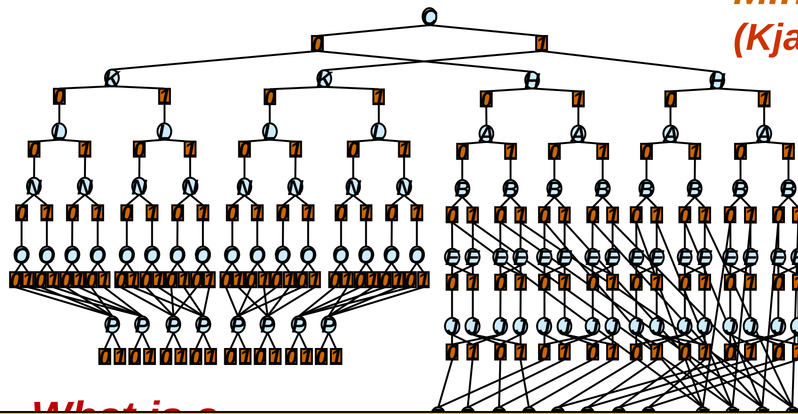
18 AND nodes

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

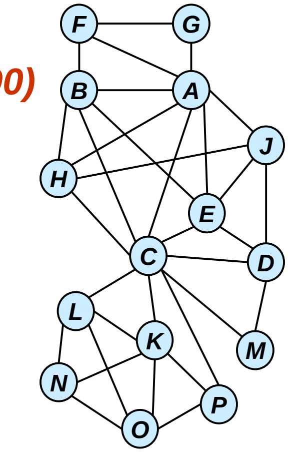
The impact of the pseudo-tree

W=4,h=8





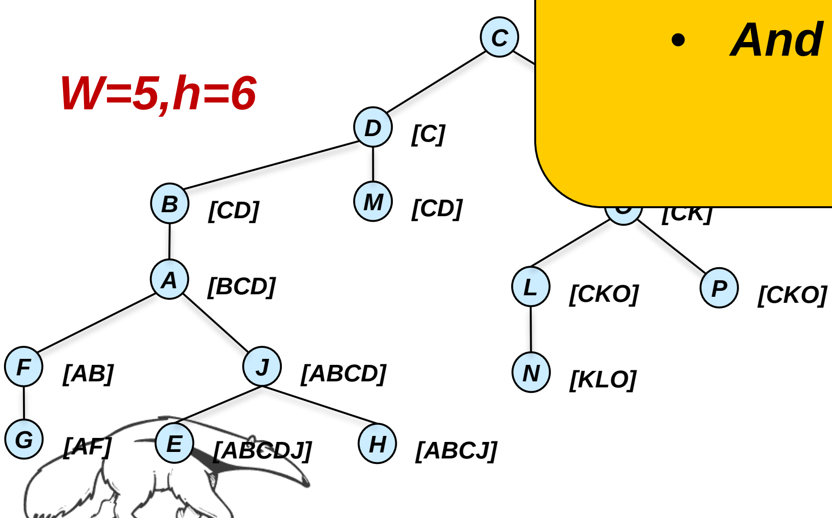
Min-Fill (Kjaerulff90)

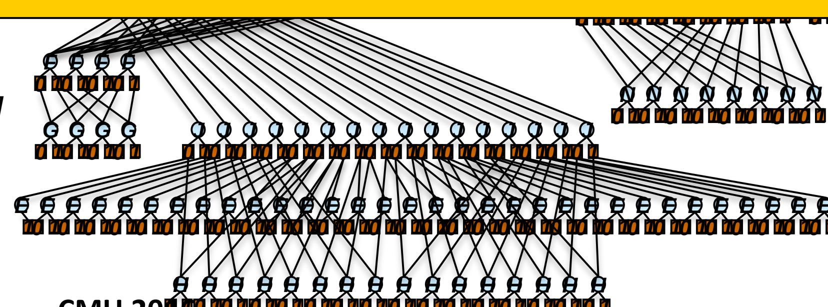


(C K H A B E J L N O D P M F G)

- *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)

Outline

- Graphical models, Queries, Algorithms
- Inference Algorithms: bucket-elimination
- Bounded Inference: a) mini-bucket, b) cost-shifting
- AND/OR search spaces and **AND/OR Branch & Bound**
- Evaluation, Software
- Summary and future work

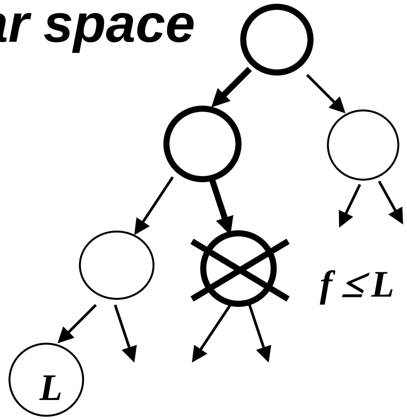
Basic Heuristic Search Schemes

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

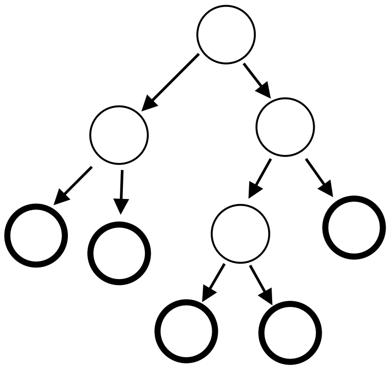
Use heuristic function $f(x^p)$ to prune the depth-first search tree

Linear space



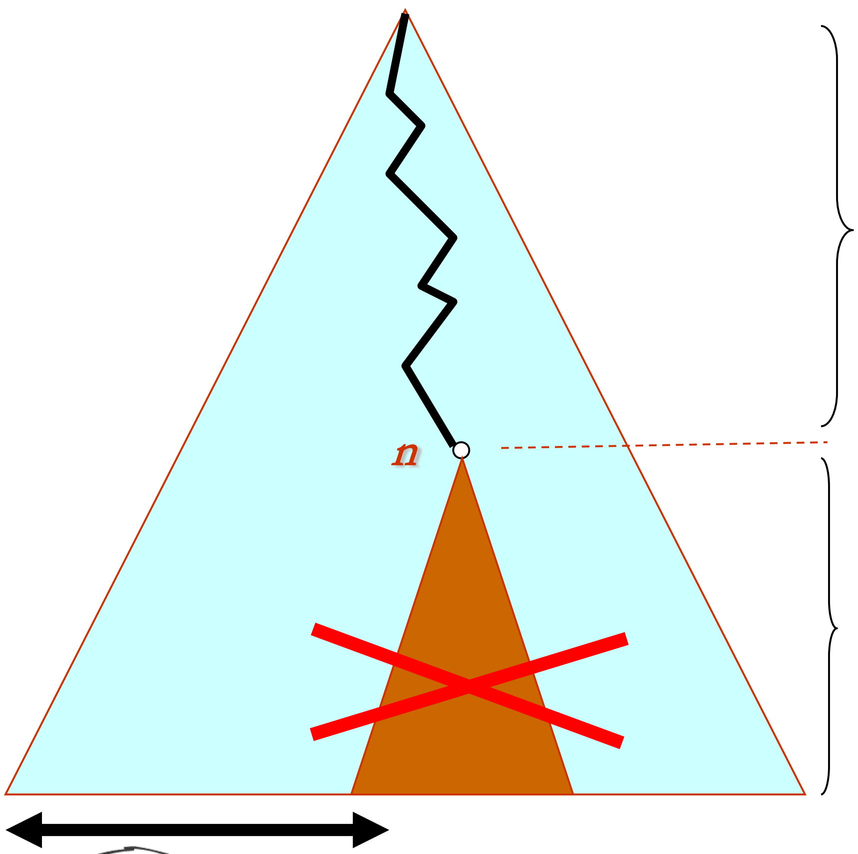
2. Best-First Search

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

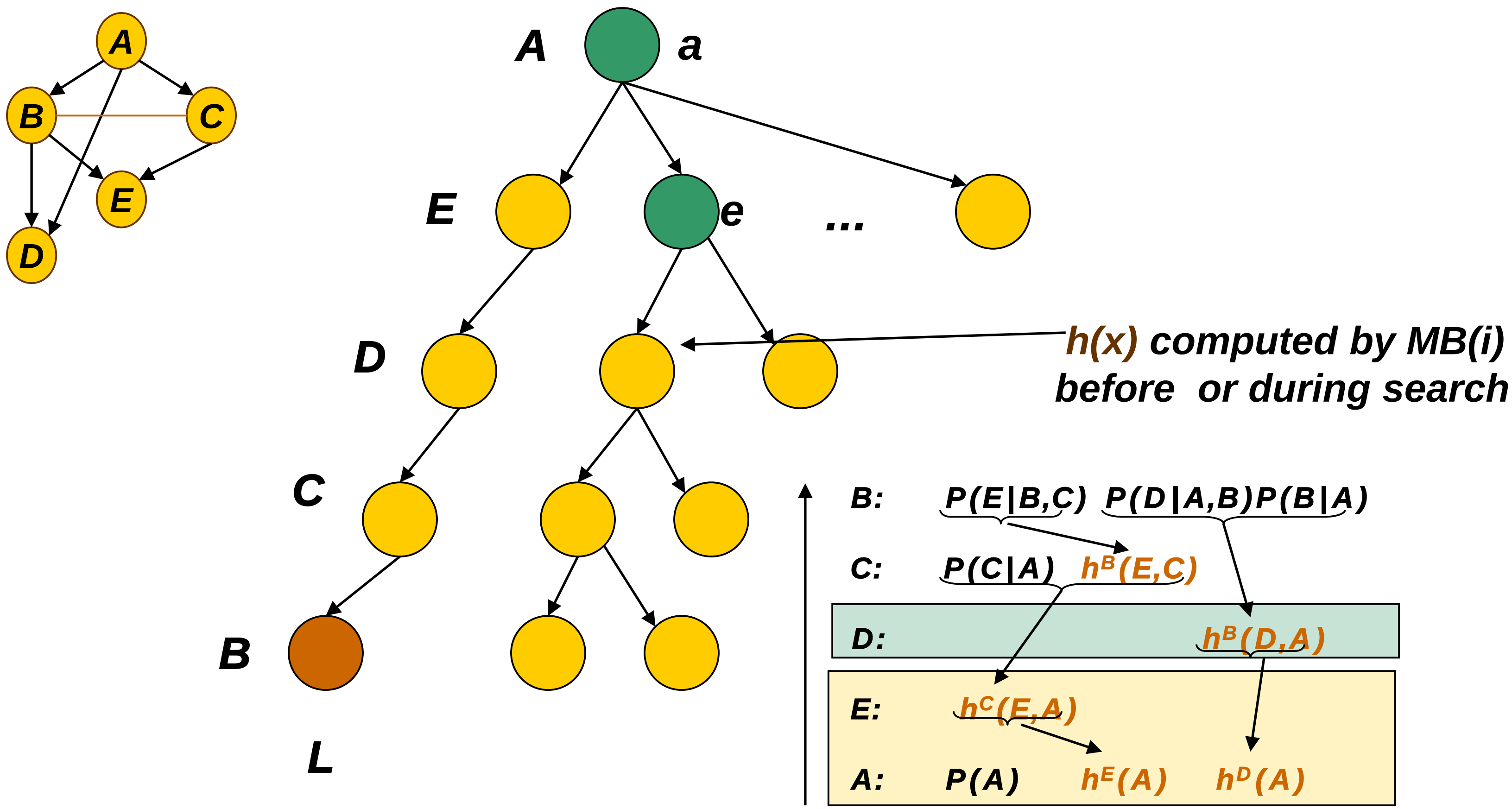
Prune if f(n) ≥ UB

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

(UB) Upper Bound = best solution so far

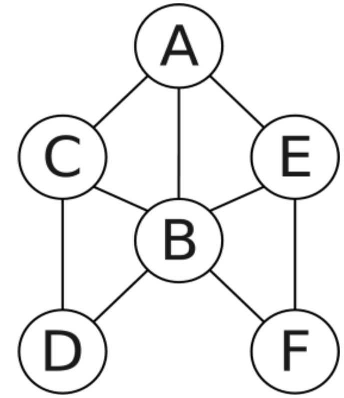
Mini-bucket Heuristics for BB search

(Kask and dechterAIJ, 2001, Kask, Dechter and Marinescu 2004, 2005, 2009, Otten 2012)

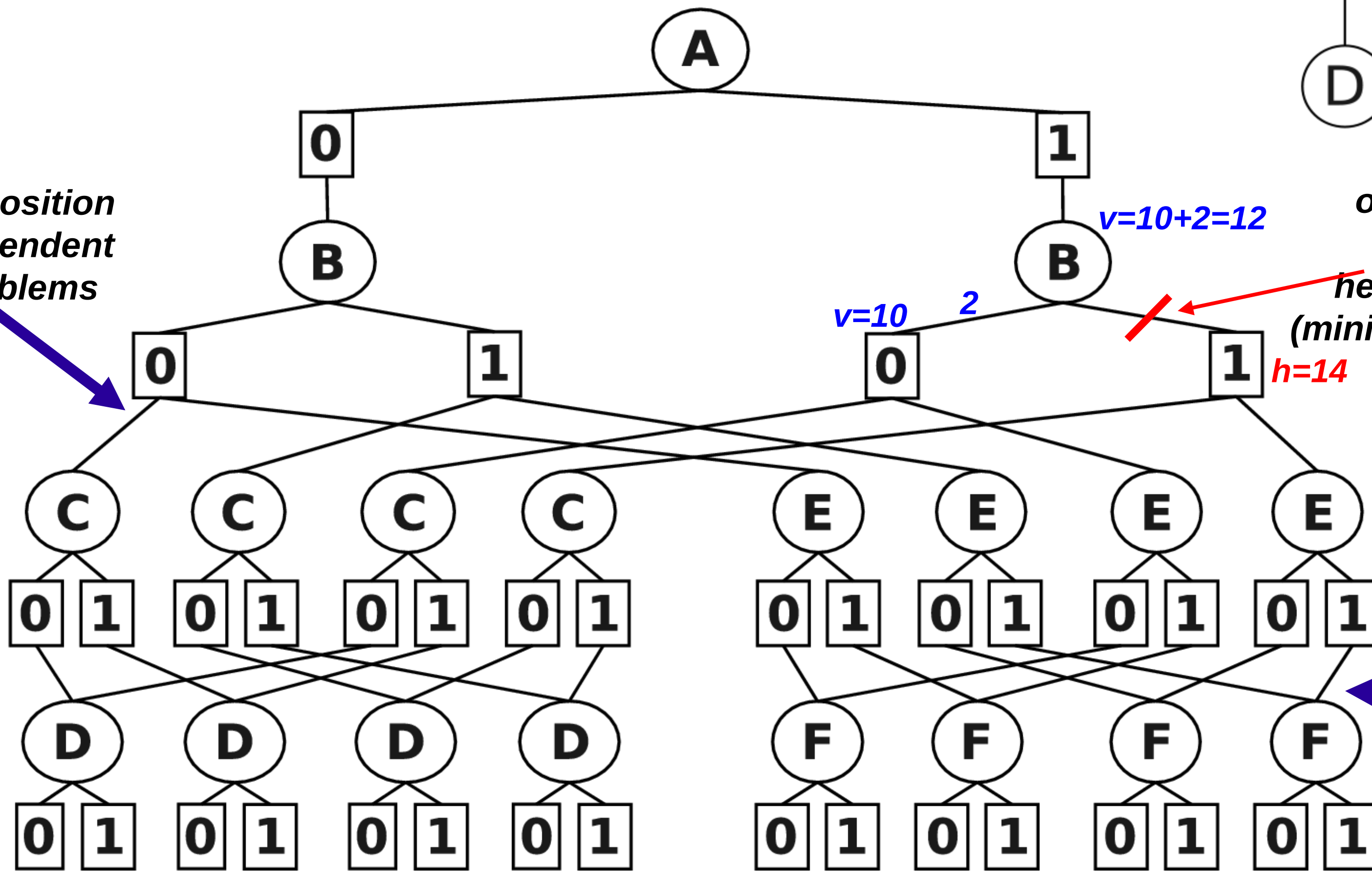


$$f(a,e,D) = P(a) \cdot h^B(D,a) \cdot h^C(e,a)$$

AND/OR Branch-and-Bound



Decomposition of independent subproblems



Prune based on current best solution and heuristic estimate (mini-bucket heuristic).

Cache table for F (independent of A)

B	*E*	*cost*
0	*0*	*10*
0	*1*	*6*
1	*0*	*...*
1	*1*	*...*

[Marinescu & Dechter, AIJ'09]

MAP: Anytime, Branch & Bound

- Best-First, Recursive Best-First
- Anytime:
  - Breadth-Rotate AND/OR BnB (2011)
  - Weighted heuristic AND/OR search (2014)
- Finding m-best solutions
- Marginal map

Empirical Evaluation (exact)

Grid and Pedigree benchmarks; Time limit 1 hour.

Outline

- Graphical models, Queries, Algorithms
- Inference Algorithms: bucket-elimination
- Bounded Inference: a) mini-bucket, b) cost-shifting
- AND/OR search spaces and AND/OR BnB
- Evaluation, Software
- Conclusion

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

Our Solvers are being used:

- Superlink online, software for linkage analysis (Geiger et. Al)
- Figaro, probabilistic language (Avi Pfeffer)

Software

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

  (standalone AOBB, AOBF solvers)
- **daoopt**
  - https://github.com/lotten/daoopt

  (distributed and standalone AOBB solver)

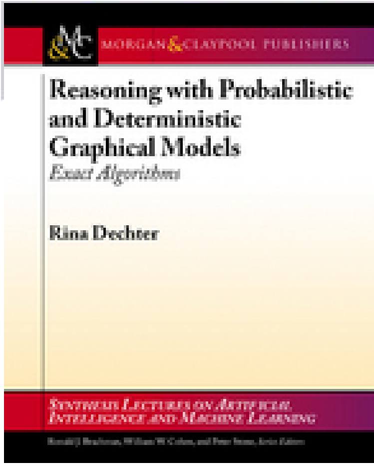
UAI Probabilistic Inference Competitions

- **2006**  ***(aolib)***
- **2008**  ***(aolib)***
- **2011**  ***(daoopt)***
- **2014**  ***(daoopt)*** |  ***(daoopt)***  ***(merlin)***

MPE/MAP | ***MMAP***

Conclusion

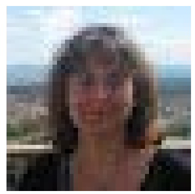
- Combining two “universal” lower-bounds scheme (mpe/map)
  - The mini-bucket scheme
  - cost-shifting or re-parameterization, schemes
- Exploiting bounds as heuristics in AND/OR search
- Yields BRAOBB that wins first place Pascal competition 2011, over many benchmarks
- Future work: Dynamic heuristics and search spaces



For publication see:

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




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