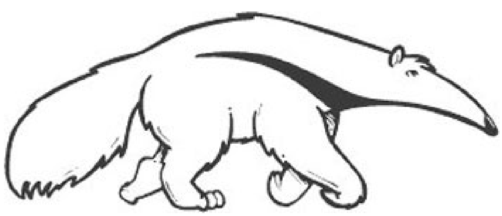
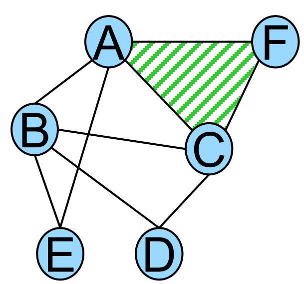
AND/OR Search Spaces: for Anytime Probabilistic Reasoning

Rina Dechter

UCI





Main Collaborators:
Alexander Ihler
Kalev Kask
Radu Marinescu
Bobak Pezeshki
Junkyu Lee

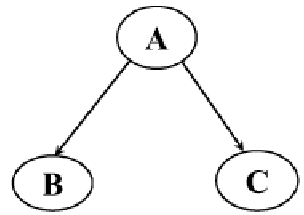
Outline

- AND/OR search spaces vs. Probabilistic circuits
- Review AND/OR search spaces for PGM
- AND/OR Multi-valued Decision Diagrams (AOMDD)
- Anytime algorithms over AND/OR search spaces
- AND/OR Abstraction sampling.
- Moving forward: Neurosymbolic, causality

Outline

- AND/OR search spaces vs. Probabilistic circuits
- Review of AND/OR search spaces for PGM
- AND/OR Abstraction sampling, balancing exact vs approximate, time vs memory vs accuracy.
- Moving forward: Reasoning under partial models and data.

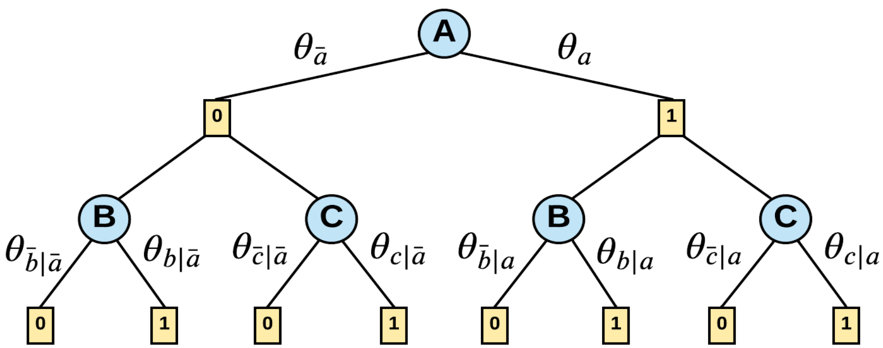
AND/OR vs Arithmetic Circuit Example

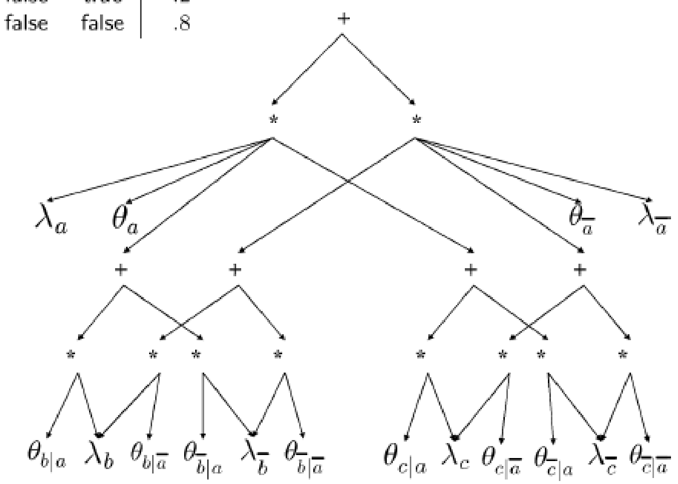


A	θ_A
true	.5
false	.5

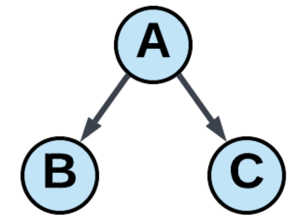
A	B	$\theta_{B\|A}$
true	true	1
true	false	0
false	true	0
false	false	1

A	C	$\theta_{C\|A}$
true	true	.8
true	false	.2
false	true	.2
false	false	.8





AND/OR Spaces and Circuits

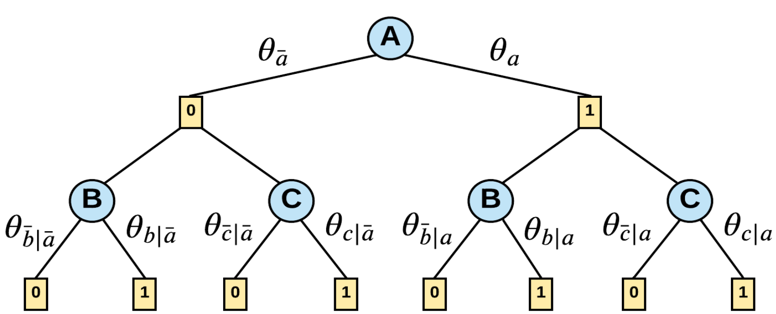


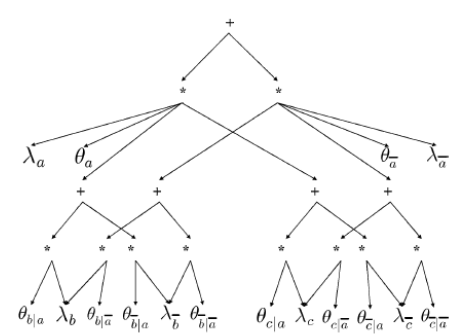
AND/OR space

- Isomorphic in practice
- Pseudo trees
- ***Used, anytime algorithms***
- Input: a full graphical model
- Input is a graph + data
- Can exploit local structure
- Multi-valued variables and tabular representation

Probabilistic Circuits

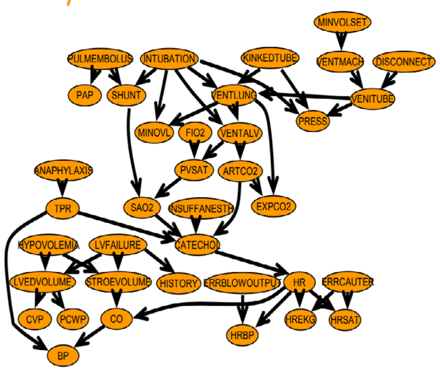
- Can be more expressive
- Dtrees
- **Used for compilation**
- Input: a full graphical model
- Input is a graph/circuit + data.
- Exploit logical structure.
- Bi-valued variables, logical functions.



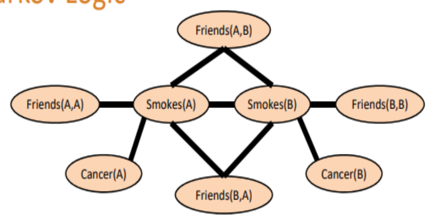


Graphical Models – Overview

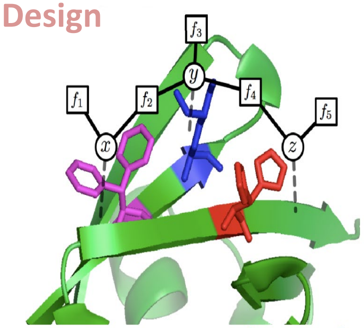
Bayesian Networks



Markov Logic

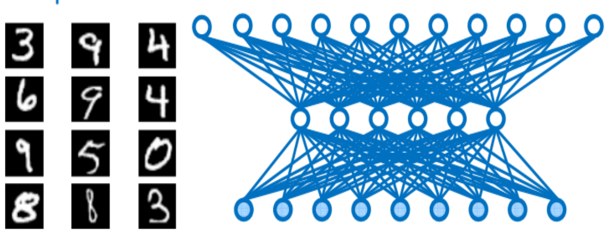


Protein Folding and Design

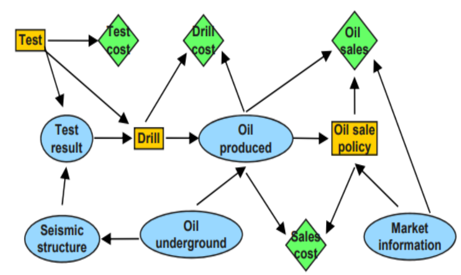


[Yanover & Weiss 2002]

Deep Boltzmann Machines

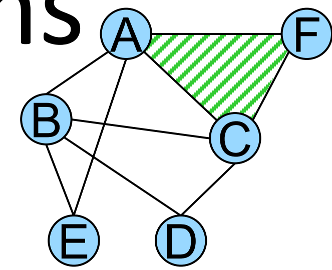


Influence Diagrams



Probabilistic Reasoning Problems

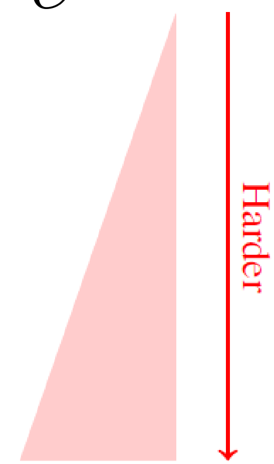
$$X = \{X_1, \ldots, X_n\}$$
$$D = \{D_1, \ldots, D_n\}$$
$$F = \{f_{\alpha_1}, \ldots, f_{\alpha_m}\}$$



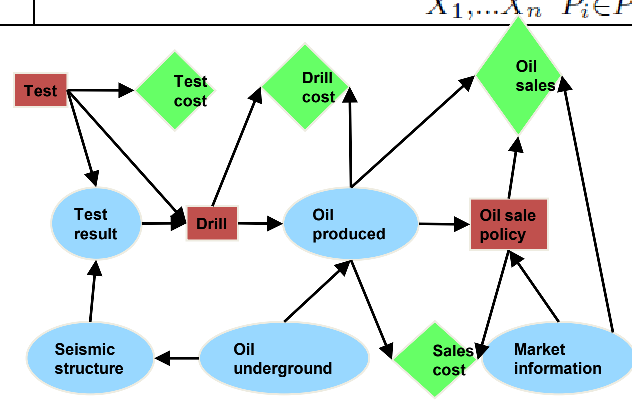
- Exact Algorithm by BE or AND/OR search, Complexity

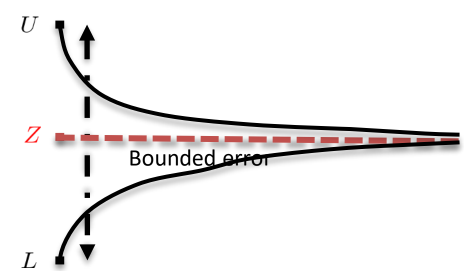
$$e^{\text{tree-width}}$$

Max-Inference:	$f(x^*) = \max_x \prod_\alpha f_\alpha(x_\alpha)$
Sum-Inference:	$Z = \sum_x \prod_\alpha f_\alpha(x_\alpha)$
Mixed-Inference (MMAP):	$f_M(x_M^*) = \max_{x_M} \sum_{x_S} \prod_\alpha f_\alpha(x_\alpha)$
Mixed-Inference (MEU):	$\text{MEU} = \max_{D_1,\ldots,D_m} \sum_{X_1,\ldots X_n} (\prod_{P_i \in P} P_i) \times (\sum_{r_i \in R} r_i)$



Influence diagrams
For planning





Anytime vs Compilation Methodology

- We want a unifying methodology that is anytime and provide bounds that improve with time regardless of memory
- Winning frameworks: search, or sampling guided by heuristics generated via **compilation**.

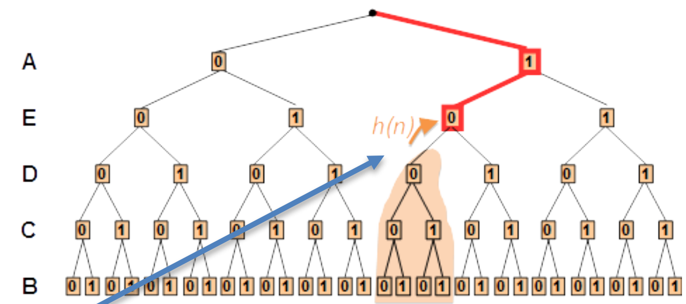
bucket B: $f(a,b)\ f(b,c)\quad f(b,d)\ f(b,e)$

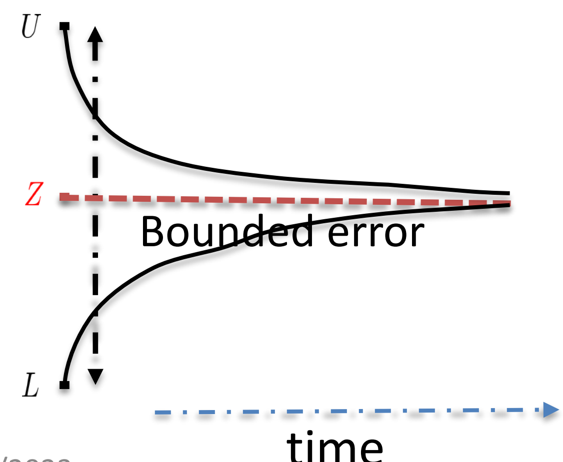
bucket C: $\lambda_{B\to C}(a,c)\ f(c,a)\ f(c,e)$

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

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

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

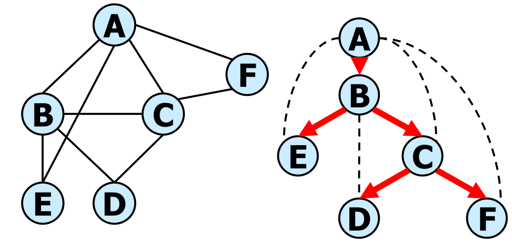


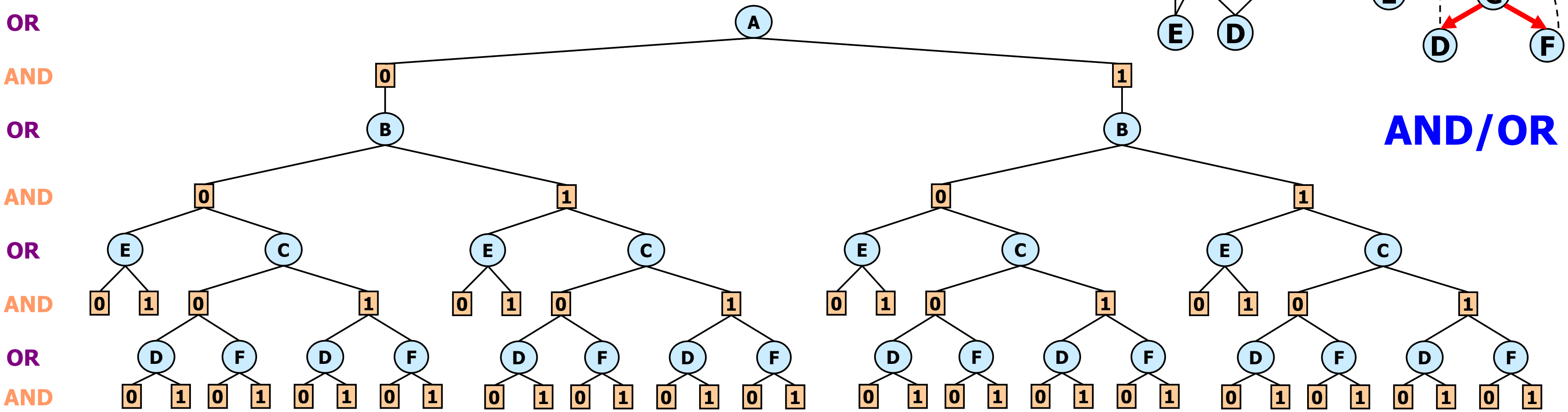


Outline

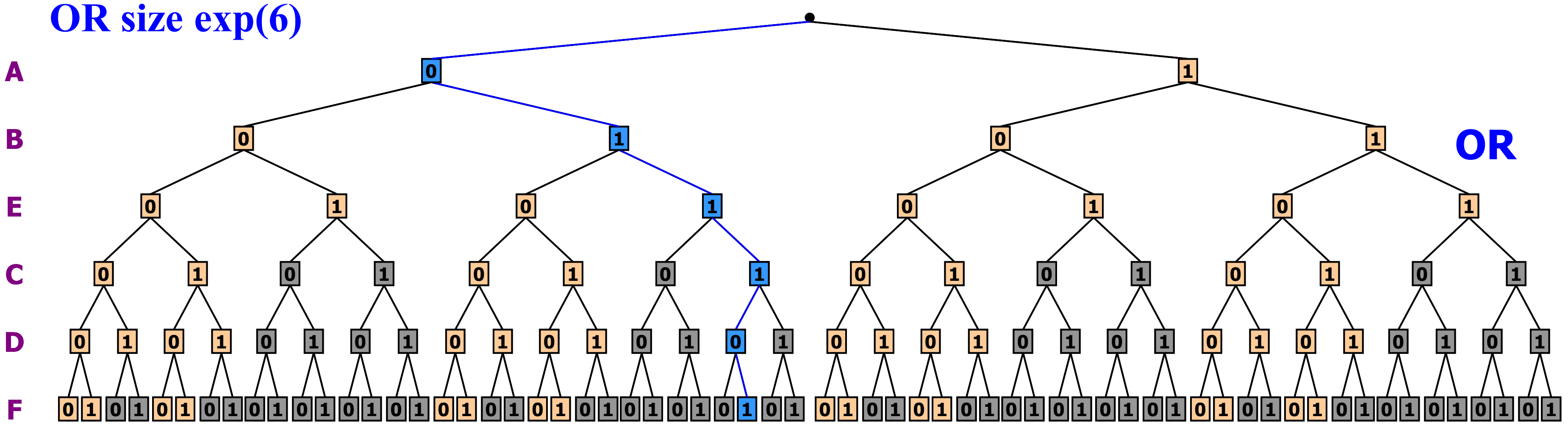
- AND/OR search spaces and Probabilistic circuits
- AND/OR search spaces for PGM
- Anytime algorithms over AND/OR space
- AND/OR Abstraction sampling, balancing exact vs approximate, time vs memory vs accuracy.
- Moving forward: Reasoning under partial models and data.

AND/OR vs. OR

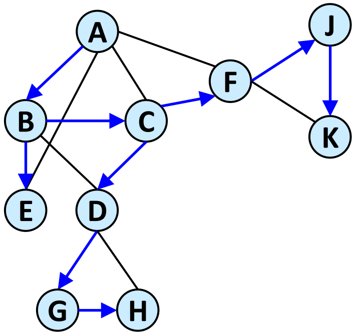


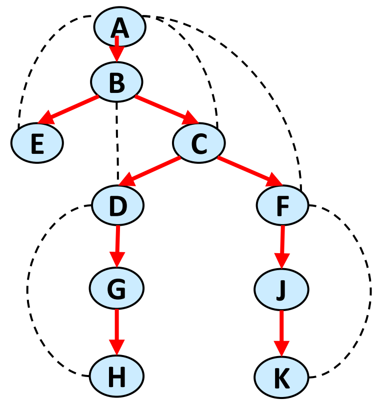


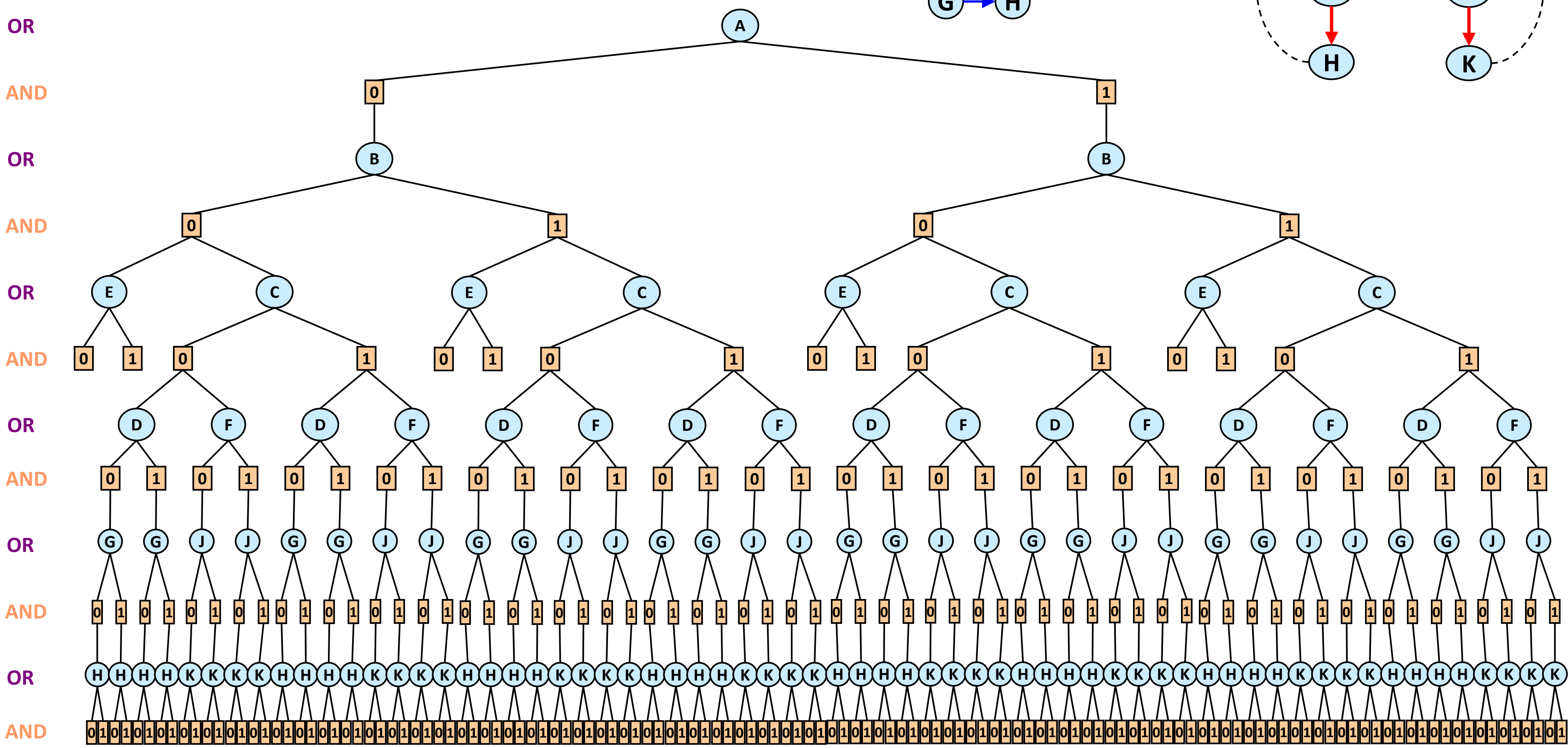
AND/OR size: exp(4),
OR size exp(6)



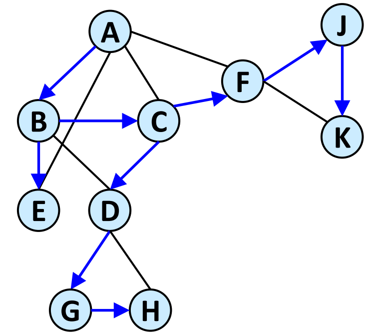
From AND/OR Tree

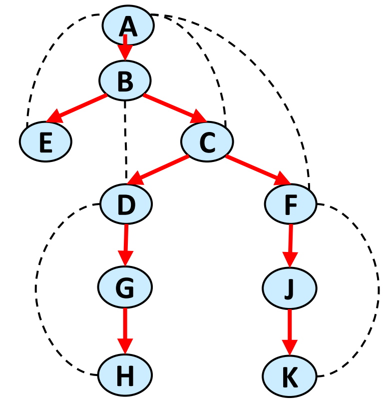


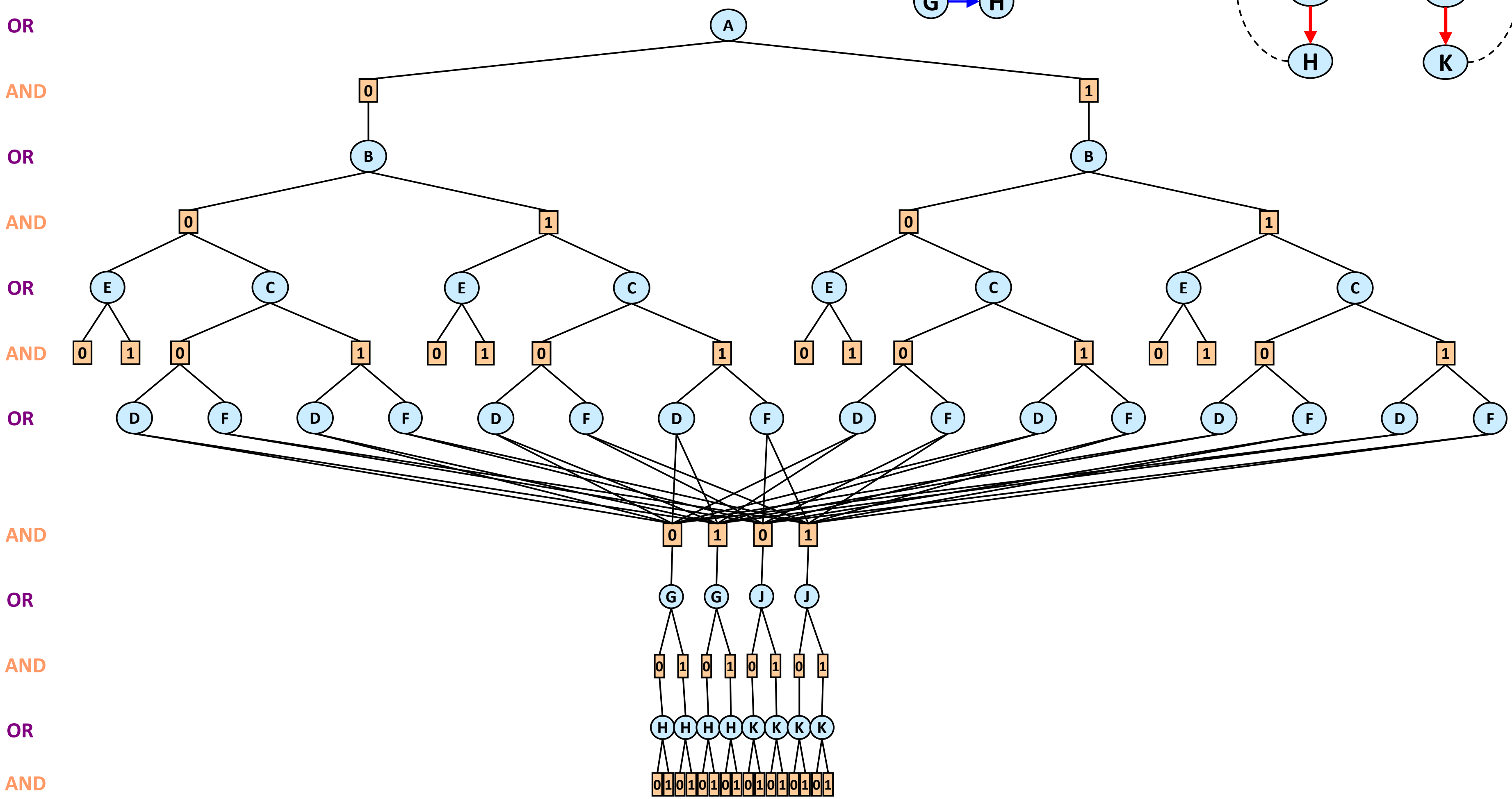




To an AND/OR Graph







Potential Search Spaces

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

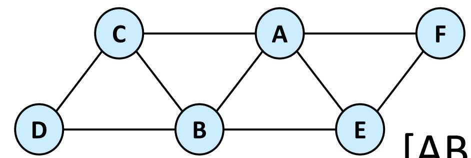
B	C	f_5
0	0	0
0	1	1
1	0	2
1	1	4

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

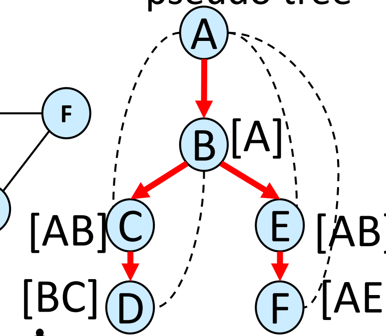
B	E	f_7
0	0	3
0	1	2
1	0	1
1	1	0

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

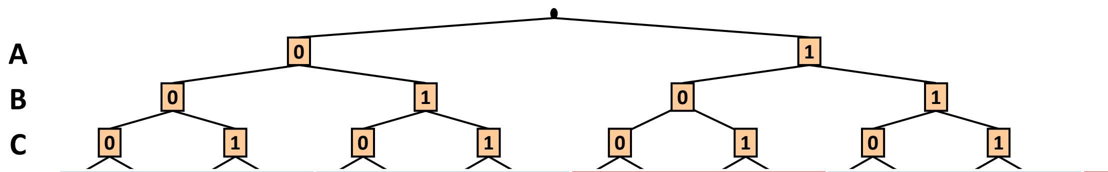
E	F	f_9
0	0	1
0	1	0
1	0	0
1	1	2

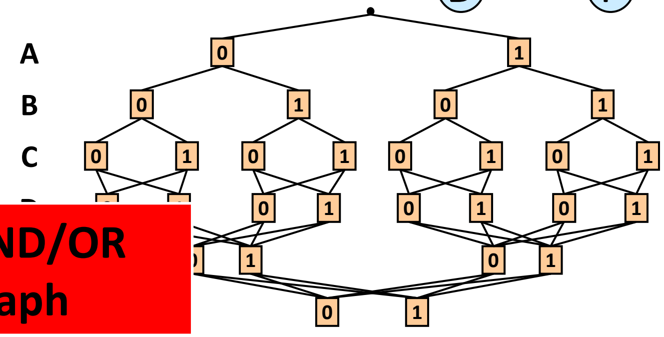


pseudo tree



[Dechter & Mateescu, 2007]

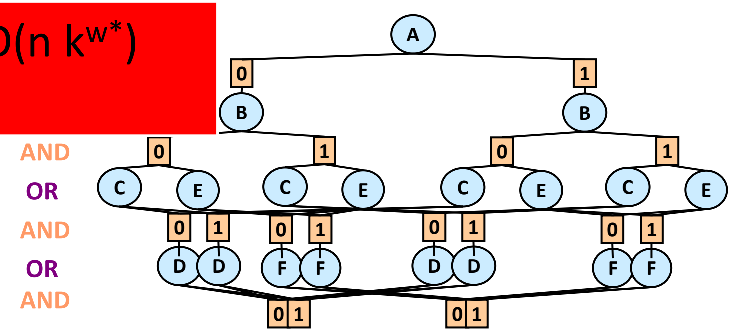




Context minimal OR search graph

28 nodes

	OR tree	AND/OR tree	OR graph	AND/OR graph
time	$O(k^n)$	$O(nk^h)$	$O(n\,k^{pw^*})$	$O(n\,k^{w^*})$
memory	O(n)	O(n)	$O(n\,k^{pw^*})$	$O(n\,k^{w^*})$



Context minimal AND/OR search graph

18 AND nodes

Computes any query:

- Constraint satisfaction
- Optimization (MAP)
- Marginal (P(e))
- Marginal map

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

Cost of a Solution Tree

$P(E \mid A, B)$

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

Evidence: E=0

$P(B \mid A)$

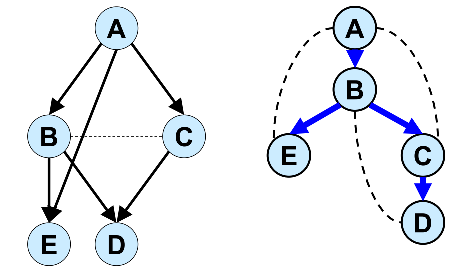
A	B=0	B=1
0	.4	.6
1	.1	.9

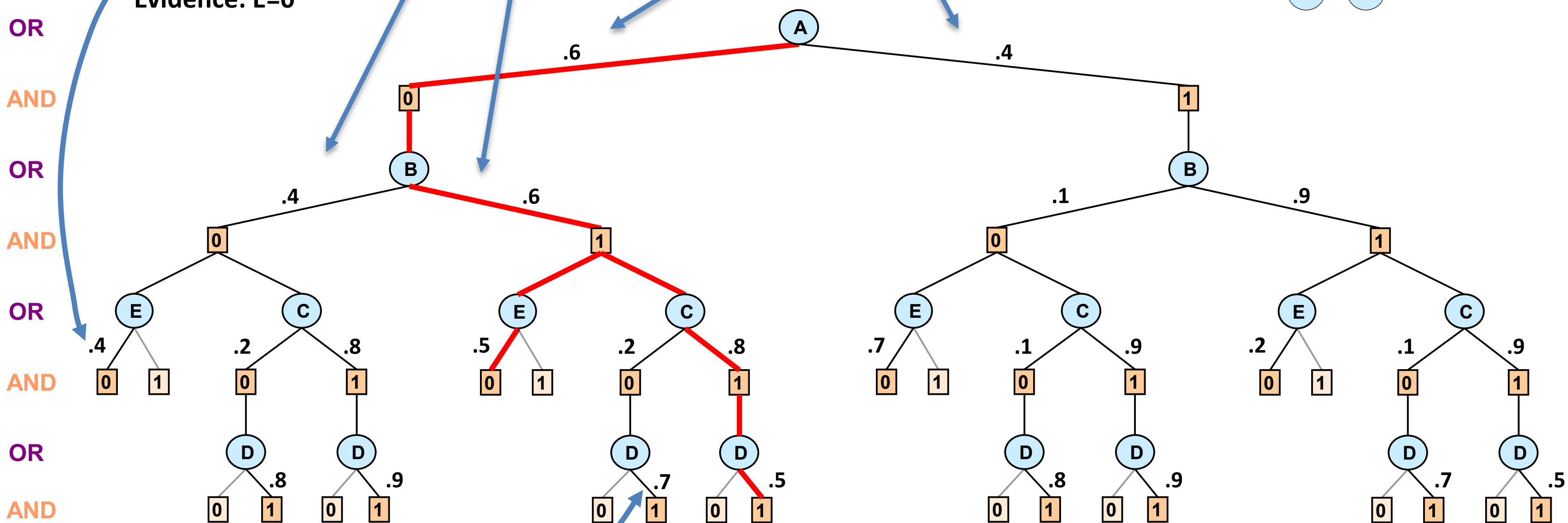
$P(C \mid A)$

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

$P(A)$

A	P(A)
0	.6
1	.4





$P(D \mid B, C)$

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

Evidence: D=1

Cost of the solution tree: the product of weights on its arcs

Cost of (A=0,B=1,C=1,D=1,E=0) = 0.6 · 0.6 · 0.5 · 0.8 · 0.5 = 0.0720

Value of a Node (e.g., Probability of Evidence)

$P(E \mid A,B)$

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

Evidence: E=0

$P(B \mid A)$

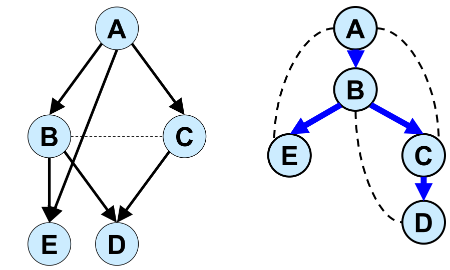
A	B=0	B=1
0	.4	.6
1	.1	.9

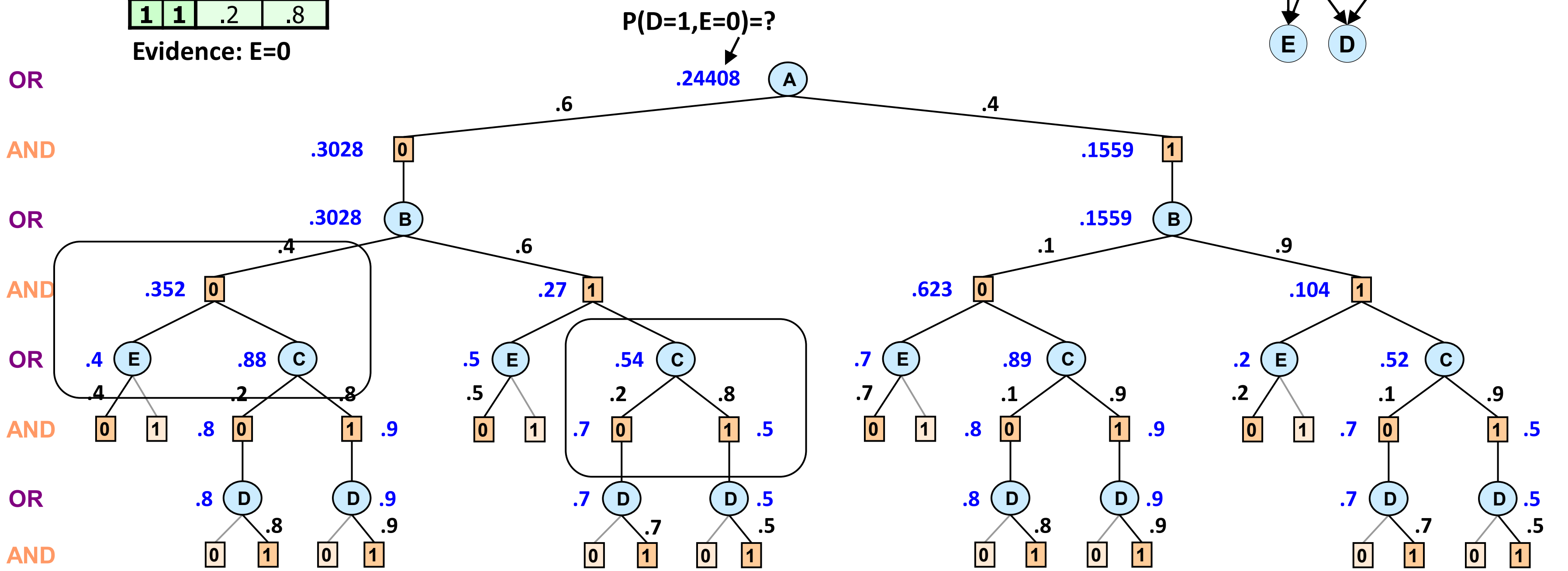
$P(C \mid A)$

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

$P(A)$

A	P(A)
0	.6
1	.4





$P(D \mid B,C)$

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

Evidence: D=1

Value of node = updated belief for sub-problem below

AND node: product $\prod_{n' \in children(n)} v(n')$

OR node: Marginalization by summation $\sum_{n' \in children(n)} w(n,n')\, v(n')$

Answering Queries: Sum-Product (Belief Updating)

$P(E \mid A,B)$

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

Evidence: E=0

$P(B \mid A)$

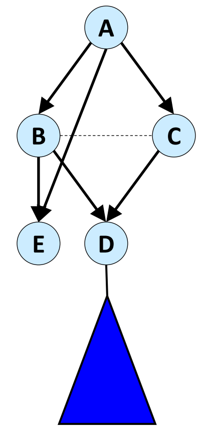
A	B=0	B=1
0	.4	.6
1	.1	.9

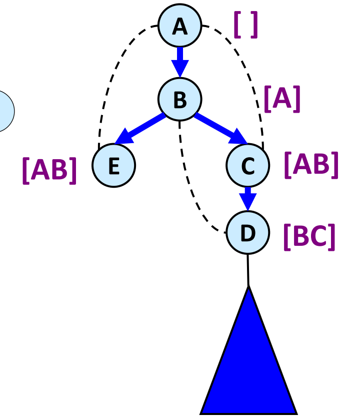
$P(C \mid A)$

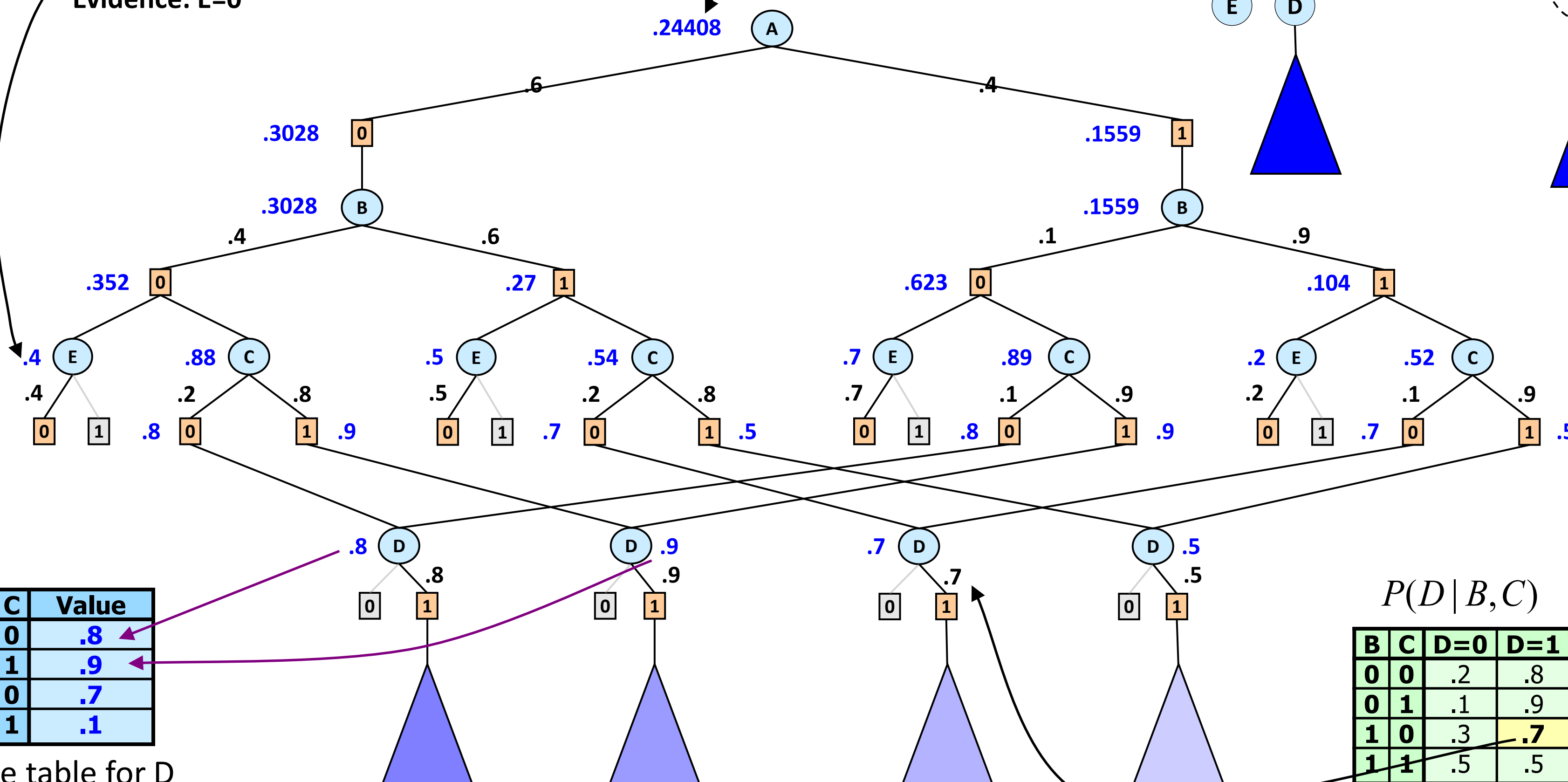
A	C=0	C=1
0	.2	.8
1	.7	.3

$P(A)$

A	P(A)
0	.6
1	.4







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

Cache table for D

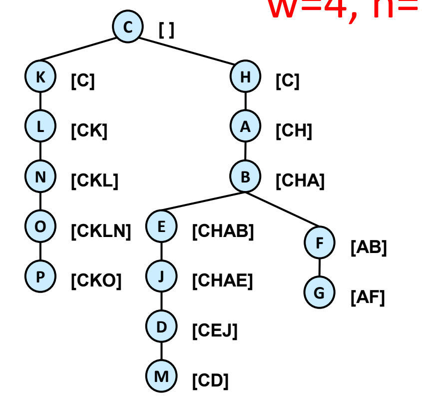
$P(D \mid B,C)$

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

Evidence: D=1

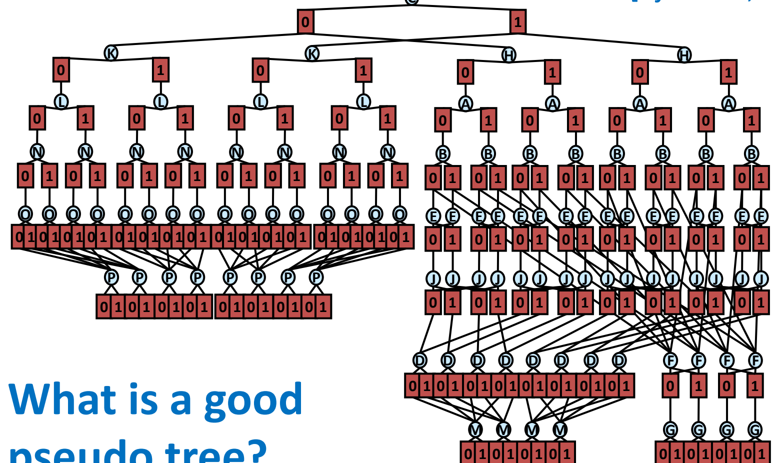
The Impact of the Pseudo Tree

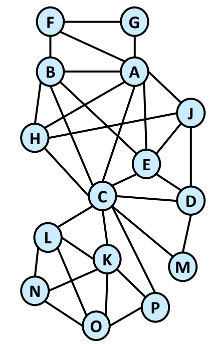
w=4, h=8



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

Min-fill [Kjaerulff, 1990]

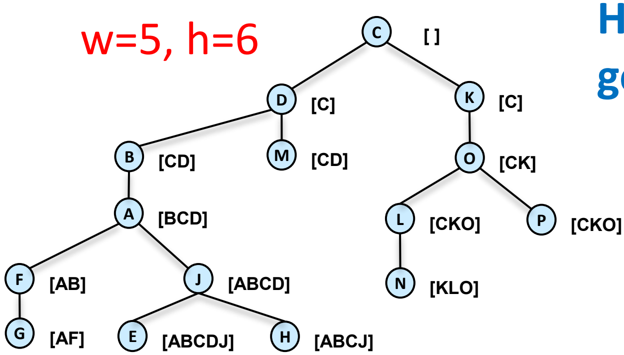




What is a good pseudo tree?

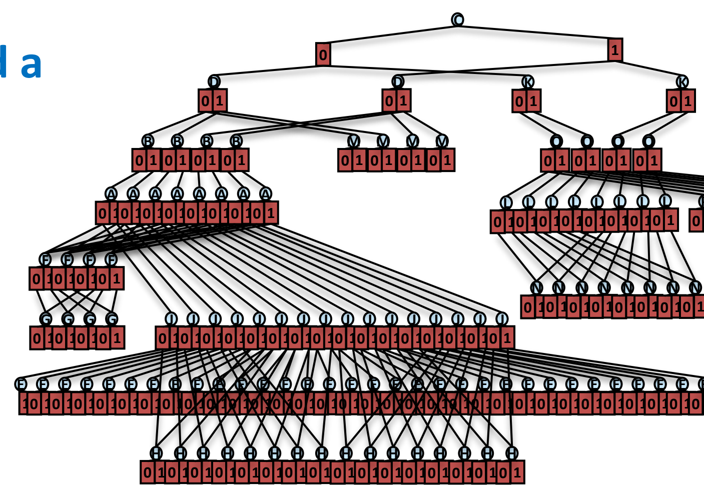
How to find a good one?

w=5, h=6



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

Hypergraph Partitioning (hMetis)

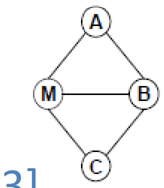


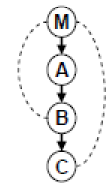
Outline

- AND/OR search spaces vs. Probabilistic circuits
- Review AND/OR search spaces for PGM
- AND/OR Multi-valued Decision Diagrams (AOMDD)
- Anytime algorithms over AND/OR search spaces
- AND/OR Abstraction sampling, balancing exact vs approximate, time vs memory vs accuracy.
- Moving forward: Neurosymbolic, causality

From Context Minimal AND/OR Graphs to AND/OR MDDs

[Mateescu, Marinescu, Lam, Dechter, 2007, 2013]





M	A	B	f(M,A,B)
0	0	0	12
0	0	1	5
0	1	0	18
0	1	1	2
1	0	0	4
1	0	1	10
1	1	0	6
1	1	1	4

M	B	C	g(M,B,C)
0	0	0	3
0	0	1	5
0	1	0	14
0	1	1	12
1	0	0	9
1	0	1	15
1	1	0	7
1	1	1	6

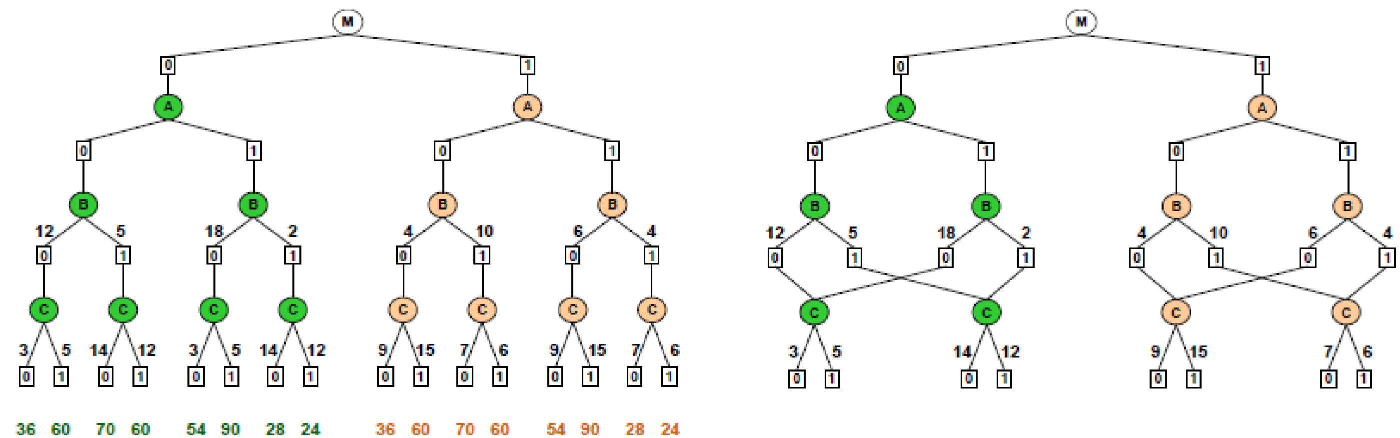


Figure 20: AND/OR search tree and context minimal graph

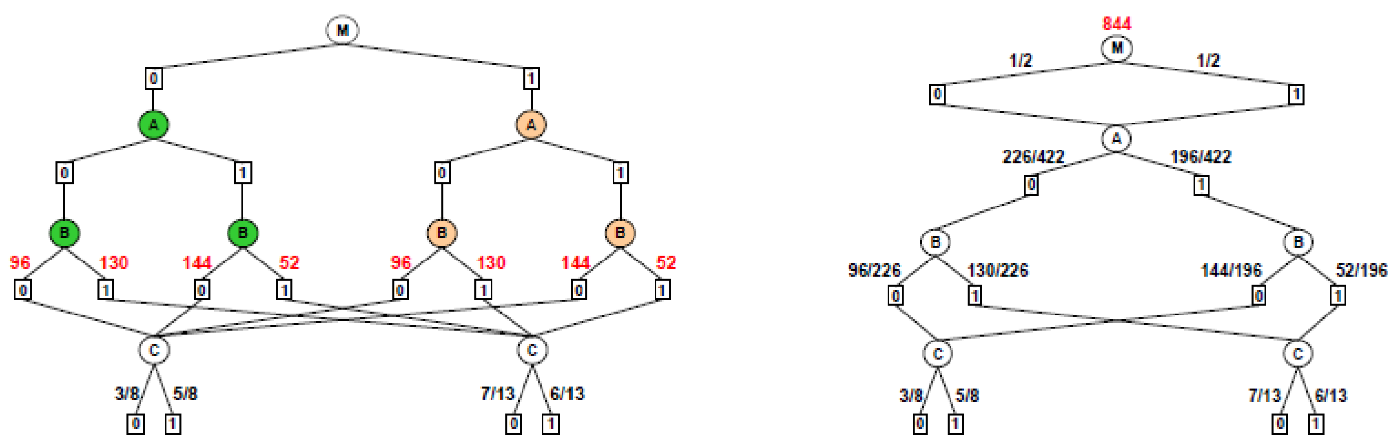


Figure 22: AOMDD for the weighted graph

Outline

- AND/OR search spaces vs. Probabilistic circuits
- Review AND/OR search spaces for PGM
- AND/OR Multi-valued Decision Diagrams (AOMDD)
- Anytime algorithms via AND/OR heuristic search
- AND/OR Abstraction sampling, balancing exact vs approximate, time vs memory vs accuracy.
- Moving forward: Neurosymbolic, causality

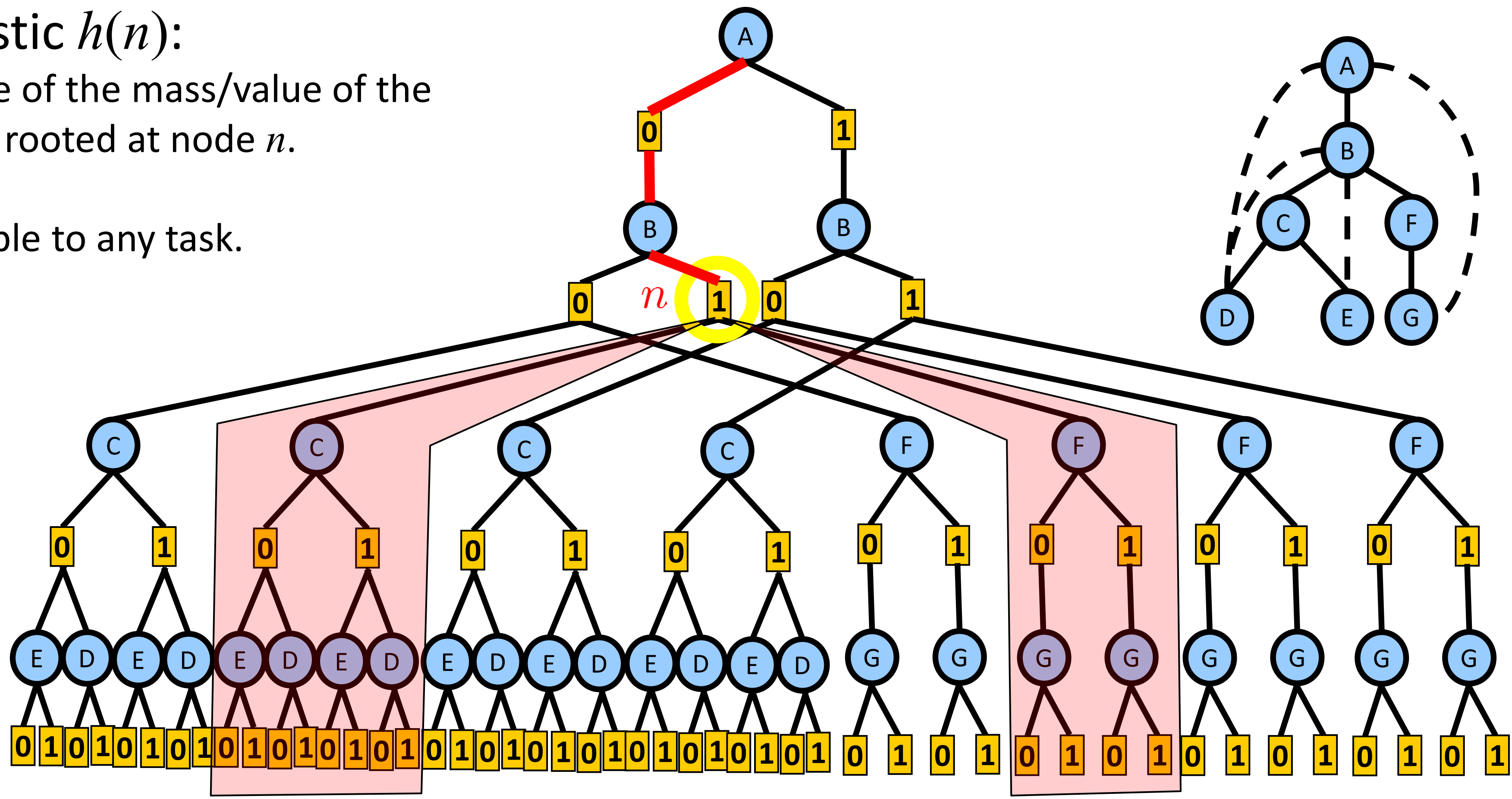
Anytime Algorithms via Heuristic Search

Heuristic $h(n)$:

Estimate of the mass/value of the subtree rooted at node n.

Applicable to any task.

Pseudo-tree

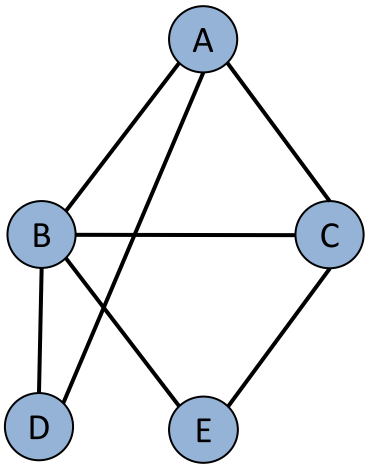


Mini-Bucket Elimination

For optimization

[Dechter & Rish 2003]

Tighten by cost-shifting



mini-buckets

m_{11}

m_{12}

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

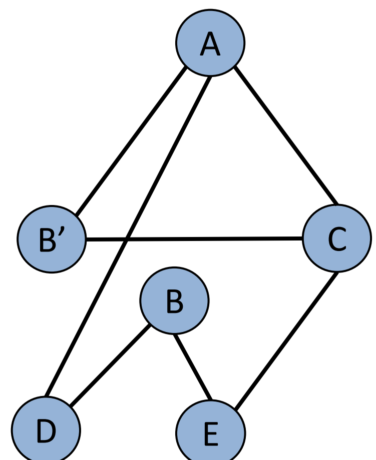
bucket C: $\underbrace{f(c,a)\ f(c,e)\ \lambda_{B\to C}(a,c)}$

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

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

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

U = upper bound



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

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

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

Can tighten heuristics using cost-shifting, Power summation and increased i-bound

Weighted Mini-Bucket

(for summation bounds)

Exact bucket elimination:

$$\lambda_B(a,c,d,e) = \sum_b \big[\, f(a,b)\cdot f(b,c)\cdot f(b,d)\cdot f(b,e) \,\big]$$

$$\le \Big[\sum_b^{w_1} f(a,b)f(b,c)\Big] \cdot \Big[\sum_b^{w_2} f(b,d)f(b,e)\Big]$$

$$= \lambda_{B\to C}(a,c) \qquad \cdot\ \lambda_{B\to D}(d,e)$$

(mini-buckets)

where $\displaystyle\sum_x^{w} f(x) = \Big[\sum_x f(x)^{1/w}\Big]^w$

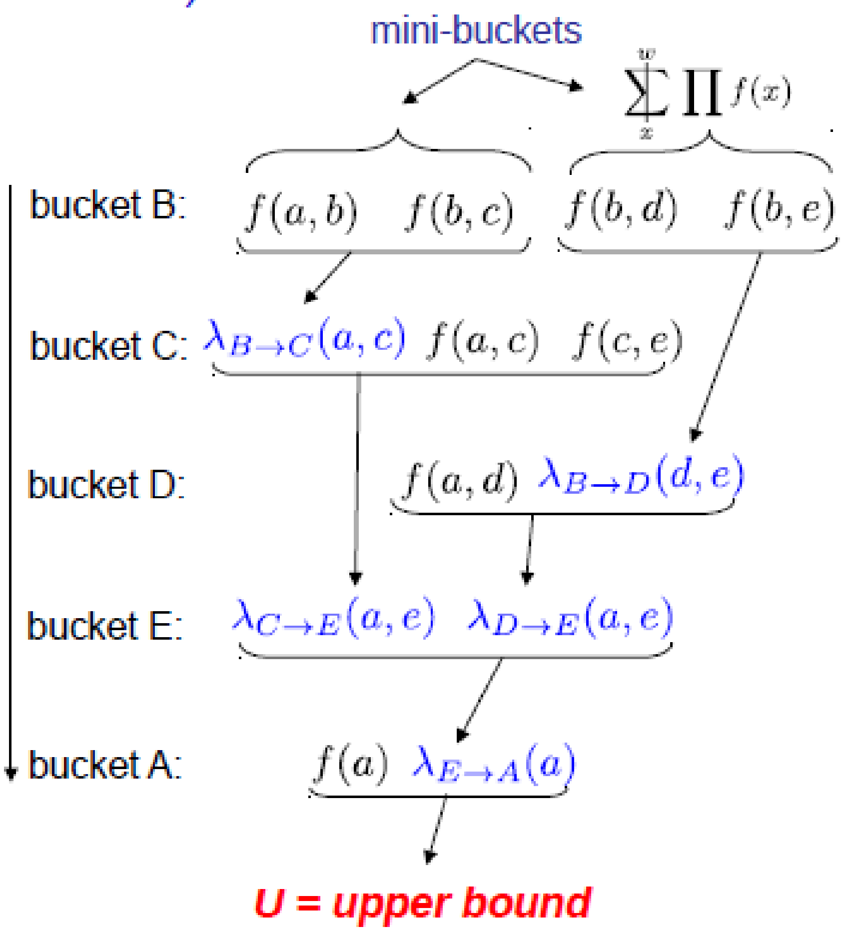
is the weighted or “power” sum operator

By Holder's inequality,

$$\sum_x^{w} f_1(x)f_2(x) \le \Big[\sum_x^{w_1} f_1(x)\Big]\Big[\sum_x^{w_2} f_2(x)\Big]$$

where $w_1 + w_2 = w$ and $w_1 > 0,\ w_2 > 0$

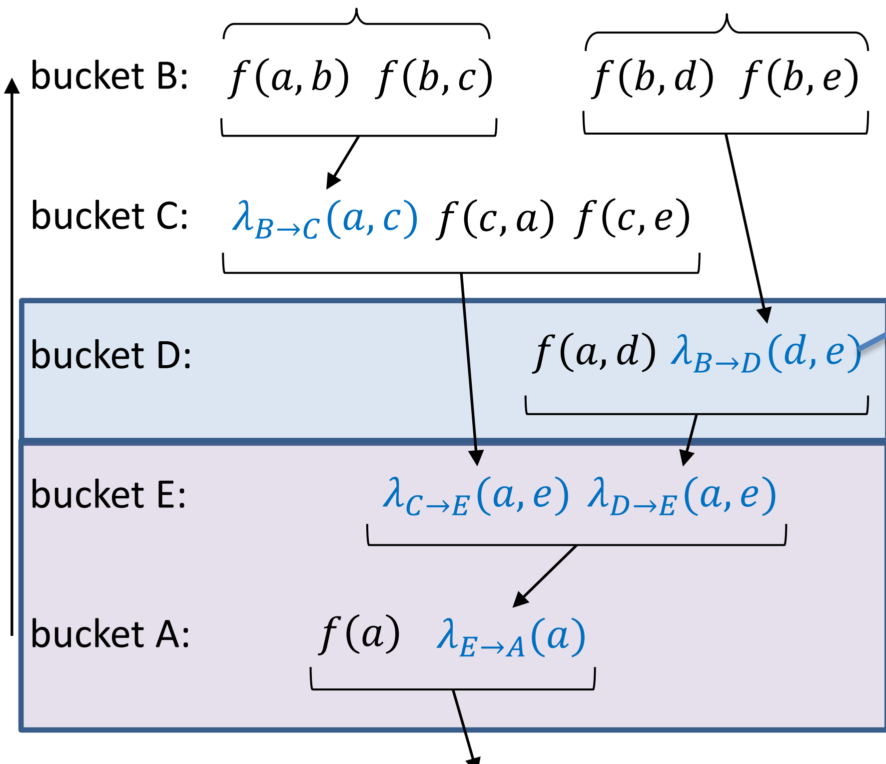
(lower bound if $w_1 > 0,\ w_2 < 0$)

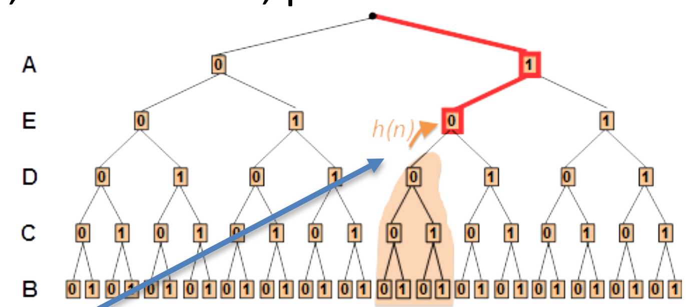


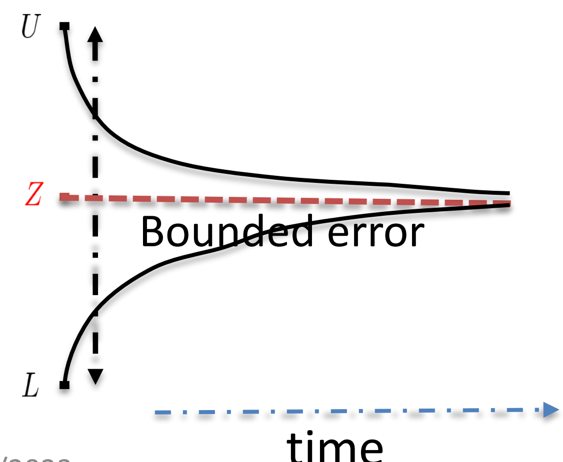
[Liu & Ihler 2011]

Anytime Algorithms via Heuristic Search

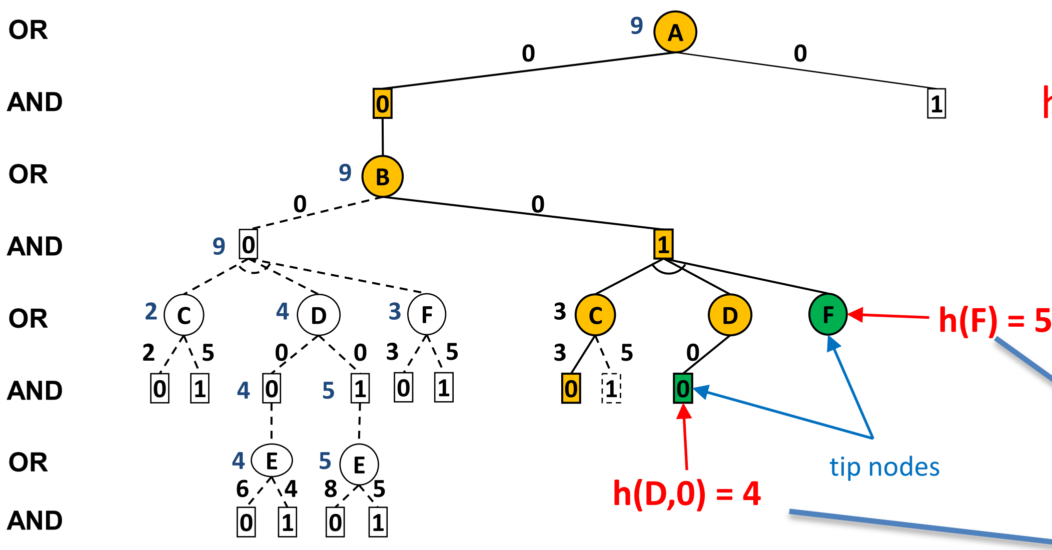
- We used a wide spectrum of heuristic search ideas to yield anytime algorithms with anytime bounds.
- **Tasks**: MAP, m-best, Partition function, Summation, Marginal Maps, Influence diagrams
- **Search methods**: Best-first, BnB, recursive BFs, Breadth-rotating for anytime AND/OR, Weighted heuristic, Dynamic vs static heuristic, look-ahead, parallel and distributed processing



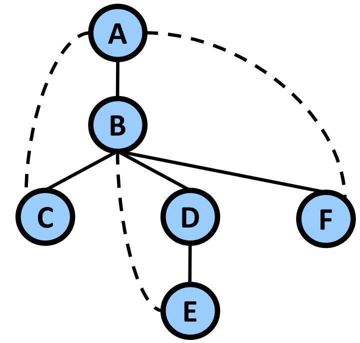




MBE Heuristic for AO Search (MAP)



h(n) ≤ v(n)



f(T') = w(A,0) + w(B,1) + w(C,0) + w(D,0) + h(D,0) + h(F) = 12 ≤ f*(T')

$\min_B \sum f(\cdot)$

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

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

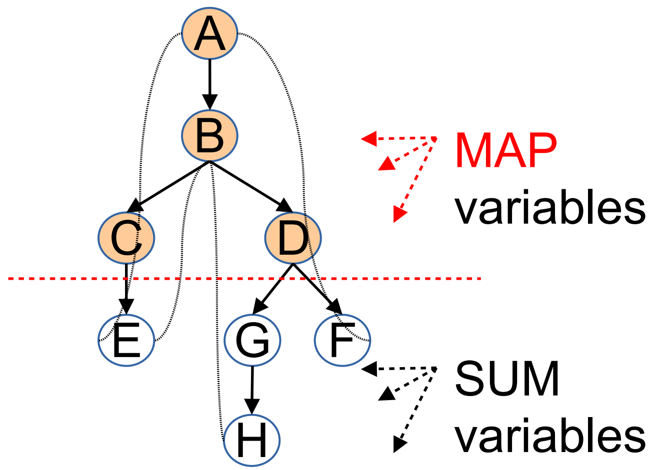
bucket D: $f(a,d)\,\lambda_{B\to D}(d,e)$

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

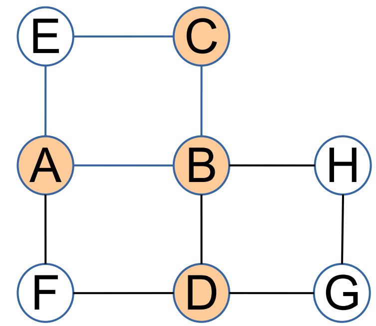
bucket A: $f(a)\ \ \lambda_{E\to A}(a)$

L = lower bound

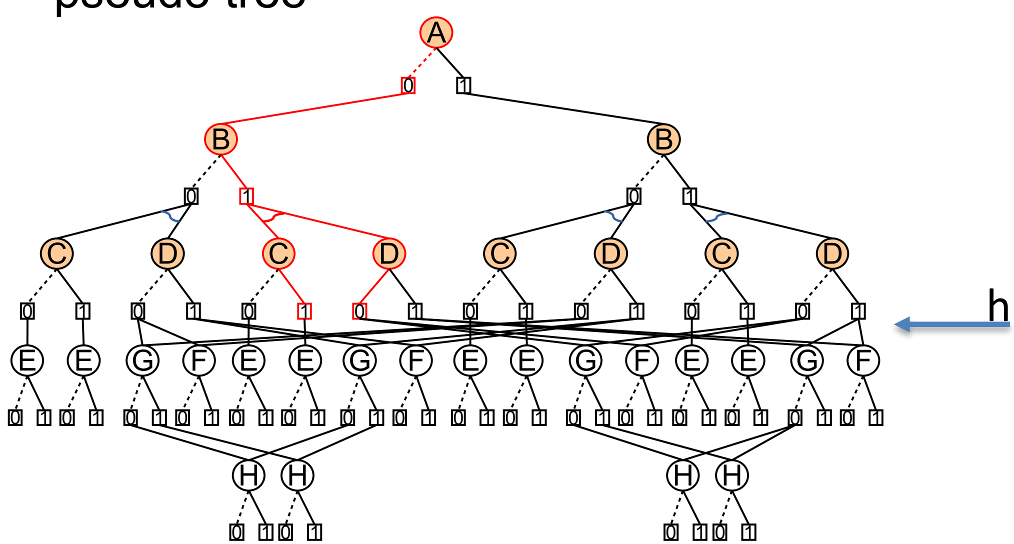
AND/OR Search for Marginal MAP

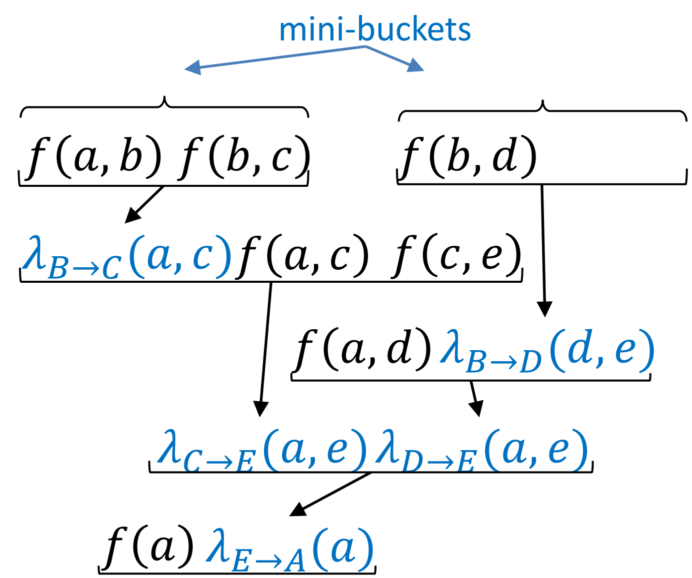


constrained pseudo tree



primal graph





[Marinescu, Dechter and Ihler, 2014]

Anytime Solvers for Marginal MAP

[Marinsecu, Lee, Dechter, Ihler, AAAI-2017, JAIR 2019]

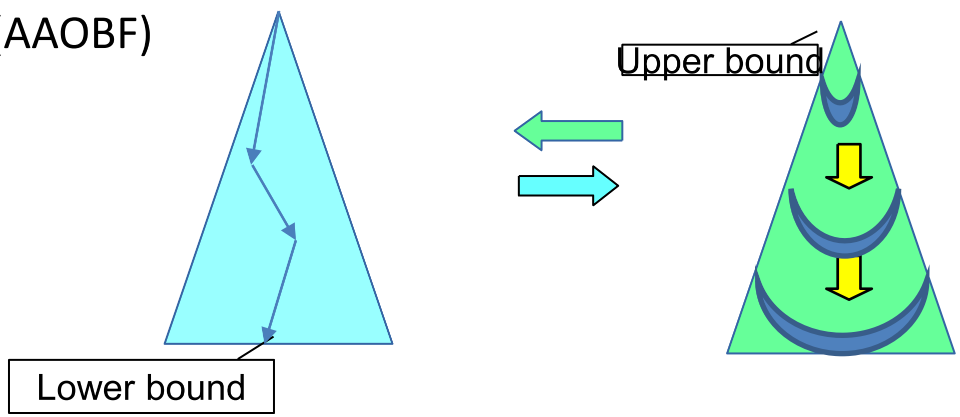
- **Weighted Best-First search:**
  - Weighted Restarting AOBF (WAOBF)
  - Weighted Restarting RBFAOO (WRBFAOO)
  - Weighted Repairing AOBF (WRAOBF)

Weighted A* search [Pohl 1970]

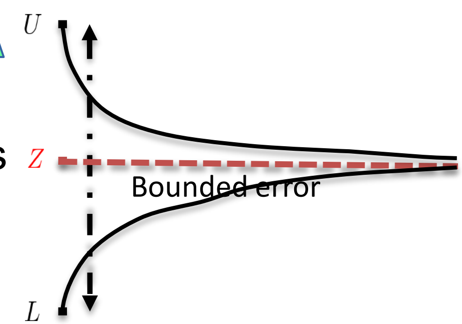
- non-admissible heuristic
- Evaluation function:

  $f(n)=g(n)+w \cdot h(n)$
- **Guaranteed w-optimal solution, cost $C \leq w \cdot C^*$**

- **Interleaving Best-first and depth-first search**:
  - Look-ahead (LAOBF),
  - alternating (AAOBF)



- Better guidance for depth-first dives using improved heuristics
- Memory robust best-first search
- using improved lower bounds



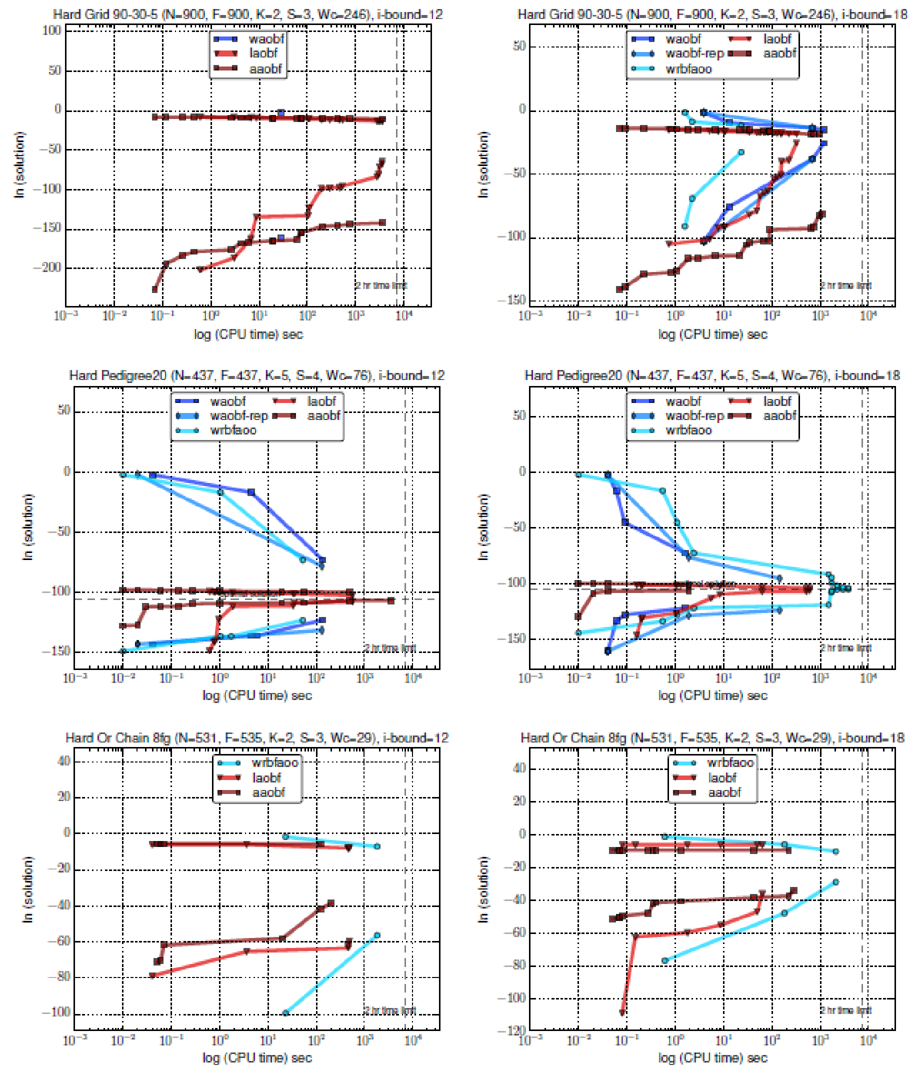
Anytime Bounding of Marginal MAP

(UAI'14, IJCAI'15, AAAI'16, AAAI'17, (Marinescu, Lee, Ihler, Dechter)

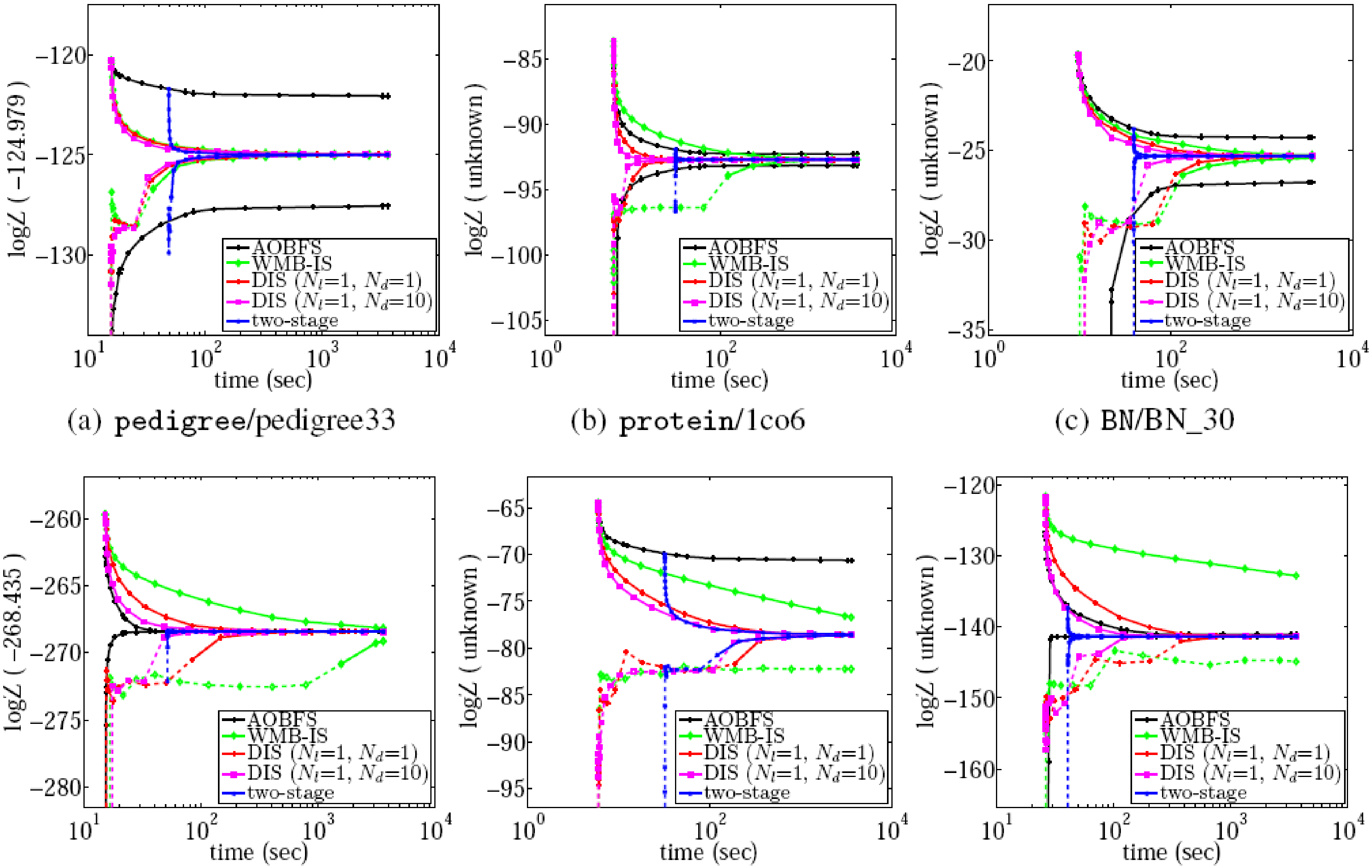
- Search: LAOBF, AAOBF, BRAOBB, WAOBF,WAOBF-rep
- heuristic: WMB-MM (20)
- memory: 24 GB

- Anytime lower and upper bounds from hard problem instances with i-bound 12 (left) and 18 (right).

- The horizontal axis is the CPU time in log scale and the vertical axis is the
- value of marginal MAP in log scale.



Partition function (Lou thesis)



(a) pedigree/pedigree33
(b) protein/1co6
(c) BN/BN_30
(d) pedigree/pedigree37
(e) protein/1bgc
(f) BN/BN_129

Students' Theses

- **Bozhena Bidyuk.** "Exploiting Graph Cutsets for Sampling-Based Approximations in Bayesian Networks“, ***2006***
- **Robert Mateescu.** "AND/OR Search Spaces for Graphical Models"*, **2007**.*
- **Radu Marinescu.** "AND/OR Search Strategies for Combinatorial Optimization in Graphical Models."*, **2008***
- **Vibhav Gogate.** "Sampling Algorithms for Probabilistic Graphical Models with Determinism." *, **2009.***
- **Andrew Gelfand.** "Bottom-Up Approaches to Approximate Inference and Learning in Discrete Graphical Models." *, **2014***.
- **Natalia Flerova.** "Methods for advancing combinatorial optimization over graphical models" *, **2015**.*
- **William Lam.** "Advancing Heuristics for Search over Graphical Models" **2017**.
- **Qi Lou.** "Anytime Approximate Inference in Graphical Models" *Ph. D Thesis **2018**.*
- **Junkyu Lee.** "Decomposition Bounds for Influence Diagrams" *Ph.D Thesis, **2020**.*

AO search for MAP winning UAI Probabilistic Inference Competitions

- **2006** 

(aolib)
- **2008** 

(aolib)
- **2011** 

(daoopt)
- **2014** 

(daoopt)

(daoopt)


(merlin)

MPE/MAP

MMAP

Software

- My software page
- **daoopt**
  - https://github.com/lotten/daoopt

  (distributed and standalone AOBB solver)
- **merlin**
  - https://developer.ibm.com/open/merlin

  (standalone WMB, AOBB, AOBF, RBFAOO solvers)

  open source, BSD license

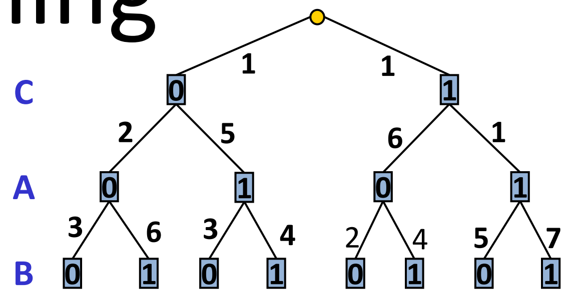
pyGMs : Python Toolbox for Graphical Models by Alexander Ihler.

Outline

- AND/OR search spaces vs. Probabilistic circuits
- Review AND/OR search spaces for PGM
- AND/OR Multi-valued Decision Diagrams (AOMDD)
- Anytime algorithms via AND/OR heuristic search
- AND/OR Abstraction sampling
- Moving forward: Neurosymbolic, causality

Between Sampling to Searching

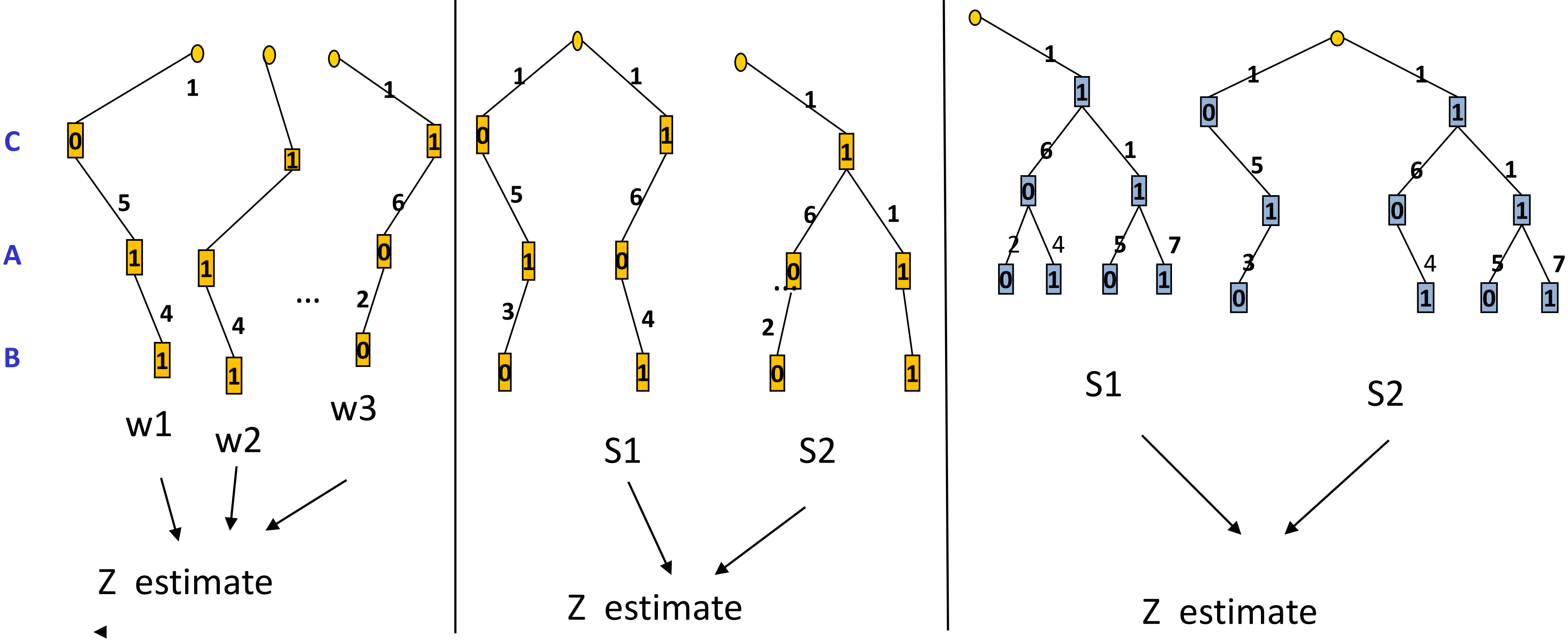
Summation queries, partition function



Importance sampling

2-config-subtree sampling

4-config-subtree sampling



More searching less sampling

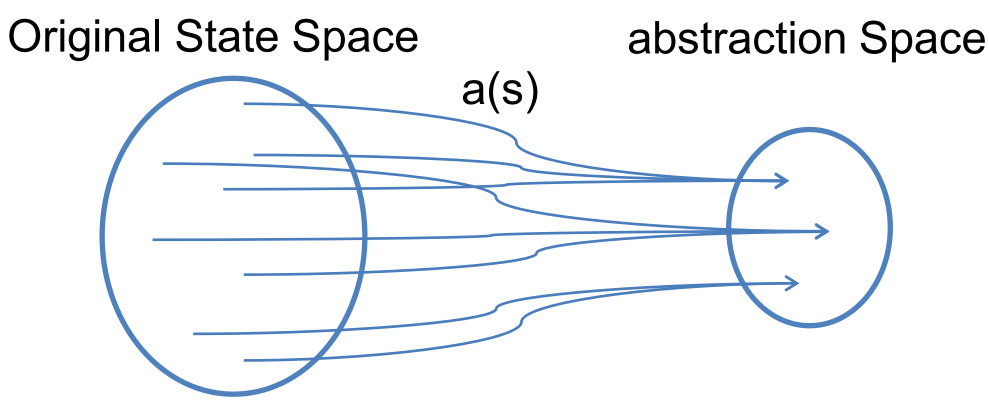
Stratified sampling

- Knuth 1975, Chen 1992 estimate search space size
- Partially enumerate, partially sample
  - Subdivide space into parts
  - Enumerate over parts, sample within parts
  - "Probe": random draw corresponding to multiple states
  - Theorem (Rizzo 2007): The variance reduction moving from Importance Sampling (IS) to Stratified IS with k strata's (under some conditions) is

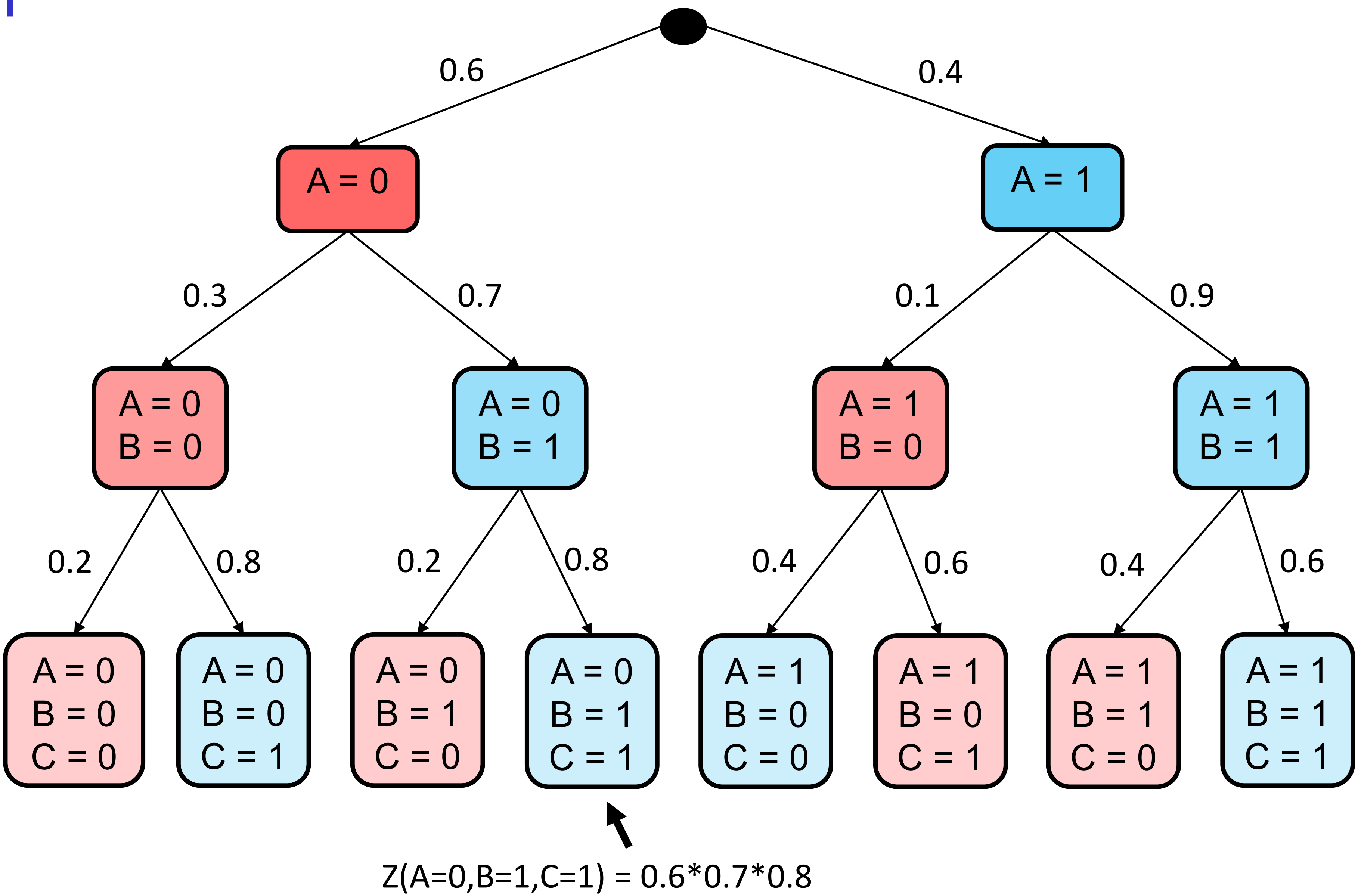
$$k \cdot var(Z_J)$$

Abstraction Function for States

- An abstraction function, $a : T \rightarrow I^+$ partitions the nodes in T.
- It is layer-based: Only nodes at the same level have the same abstract state.
- Examples: a heuristic function, Context-based abstraction

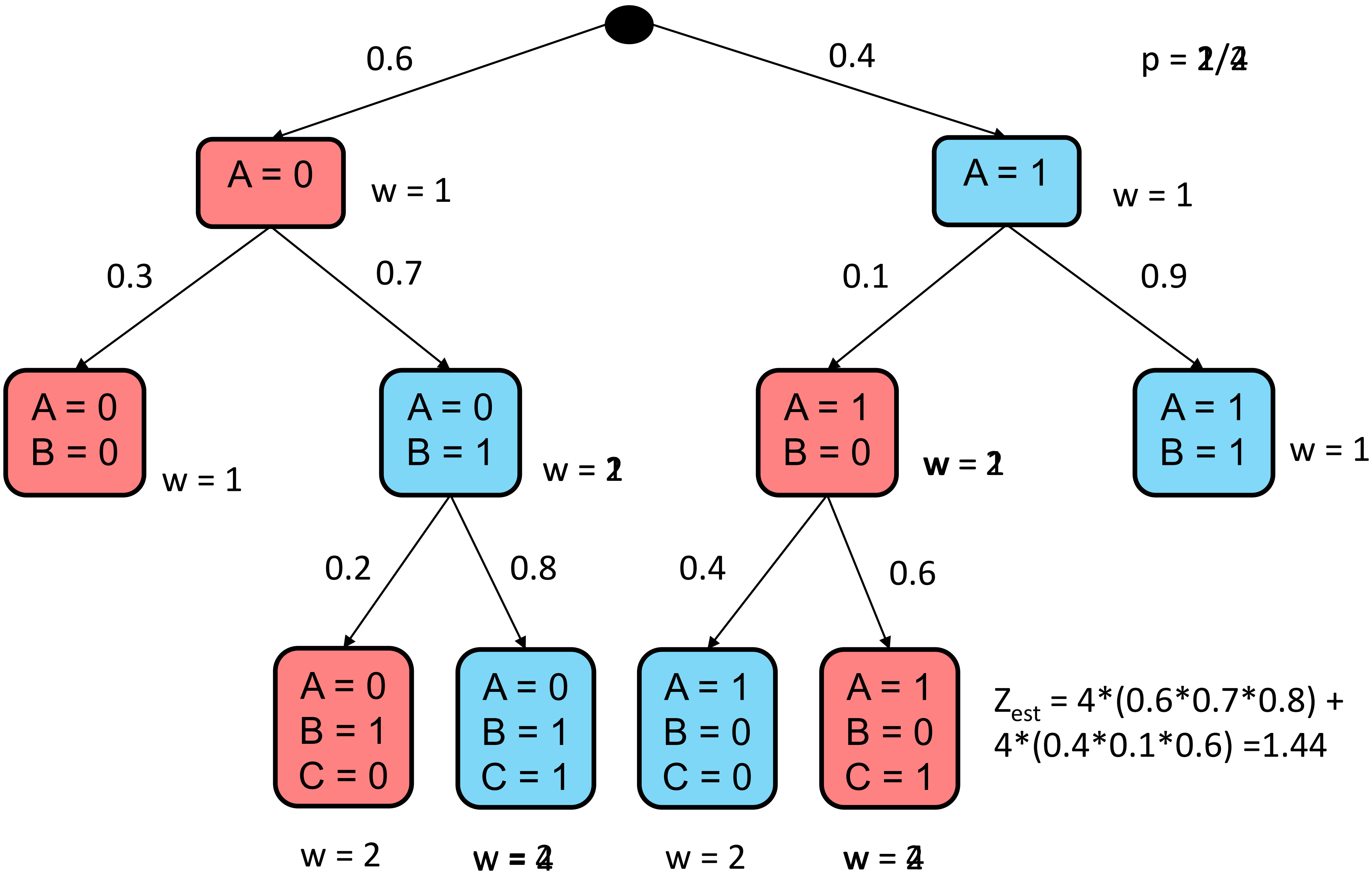


Full OR Tree



0.6
0.4
A = 0
A = 1
0.3
0.7
0.1
0.9
A = 0
B = 0
A = 0
B = 1
A = 1
B = 0
A = 1
B = 1
0.2
0.8
0.2
0.8
0.4
0.6
0.4
0.6
A = 0
B = 0
C = 0
A = 0
B = 0
C = 1
A = 0
B = 1
C = 0
A = 0
B = 1
C = 1
A = 1
B = 0
C = 0
A = 1
B = 0
C = 1
A = 1
B = 1
C = 0
A = 1
B = 1
C = 1
Z(A=0,B=1,C=1) = 0.6*0.7*0.8

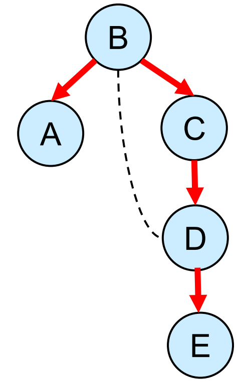
Method 1 – OR Tree

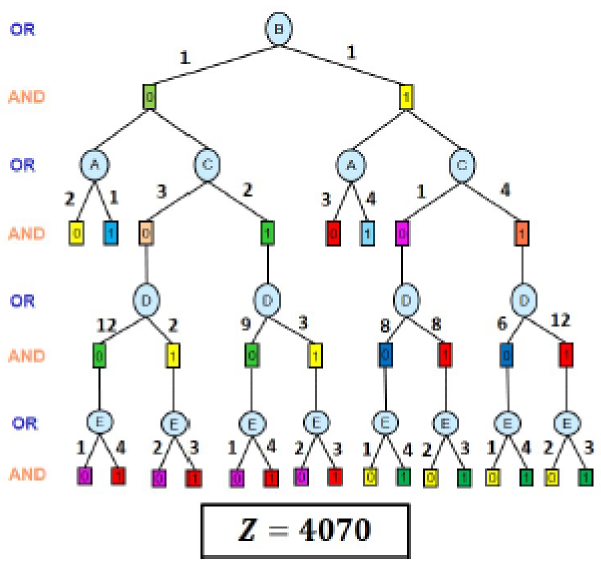


Z_{est} = 4*(0.6*0.7*0.8) + 4*(0.4*0.1*0.6) =1.44

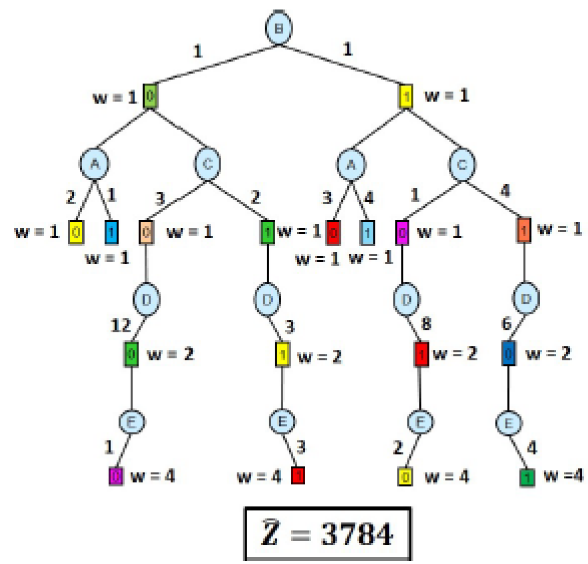
Abstraction Sampling - AND/OR

- Input: Abstraction function a, (partition the states at each level), a sampling proposal p.
- Traverse AND/OR search tree breadth-first
- Compute estimate $\hat{Z}$



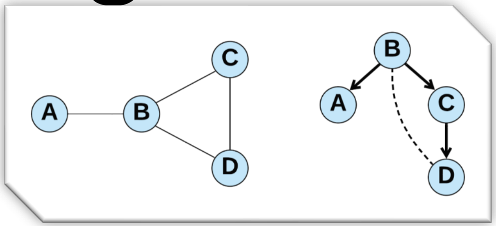


(d) Full AND/OR search tree



(e) Probe - AND/OR

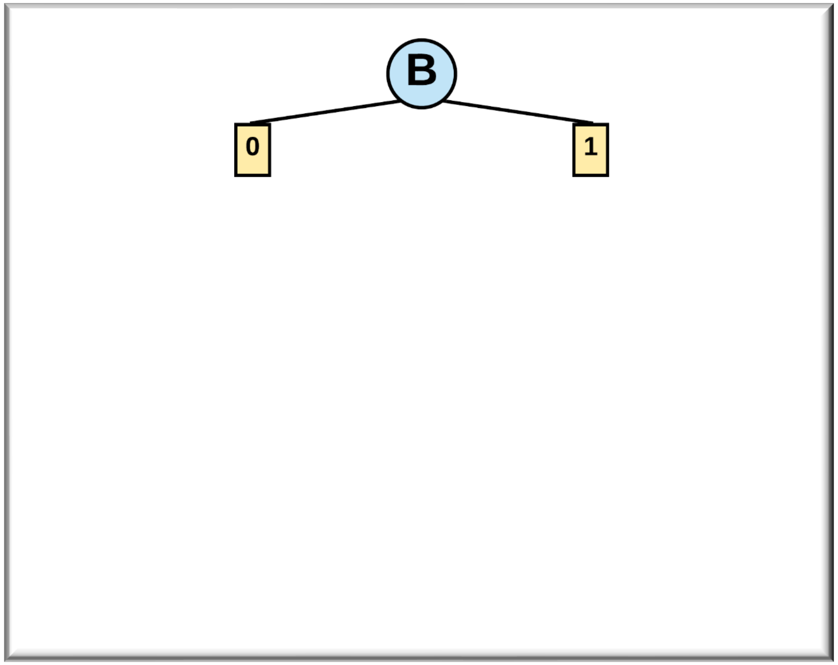
AND/OR Abstraction Sampling



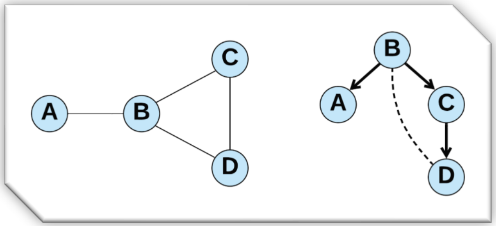
Input: Abstraction function a, (partition the states at each level). Sampling proposal p, *pseudo-tree*

Key Points:

- Expands along a depth first traversal of the guiding pseudo tree
- Perform abstraction at each level
- Immediately performs recursive pruning of branches that cannot be part of valid configurations



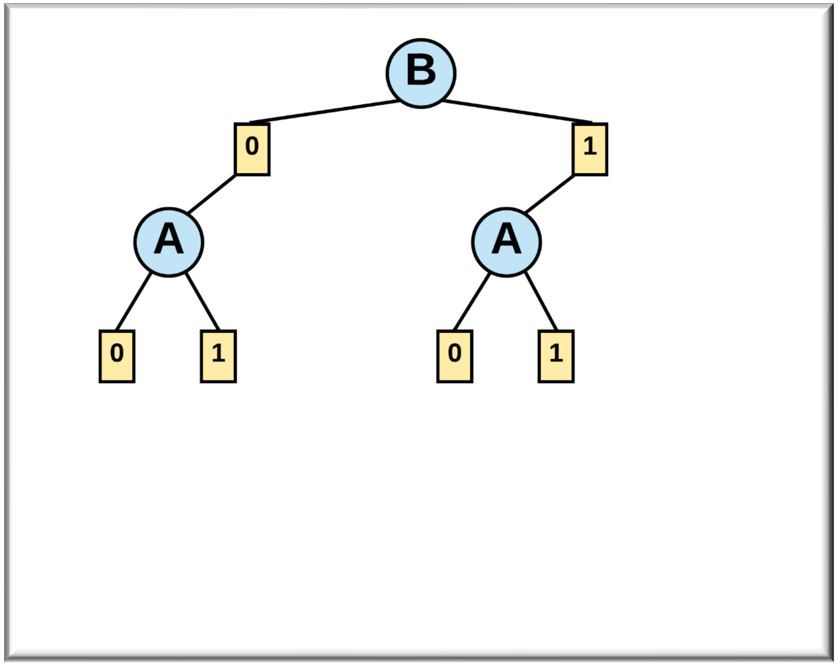
New Scalable AOAS



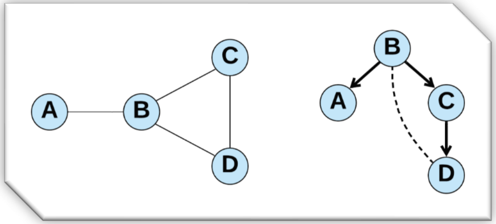
New AND/OR abstraction sampling scheme that allows for non-proper abstractions while still ensuring formation of valid probes.

Key Points:

- Performs non-proper abstractions
- Expands along a depth first traversal of the guiding pseudo tree
- Immediately performs recursive pruning of branches that cannot be part of valid configurations



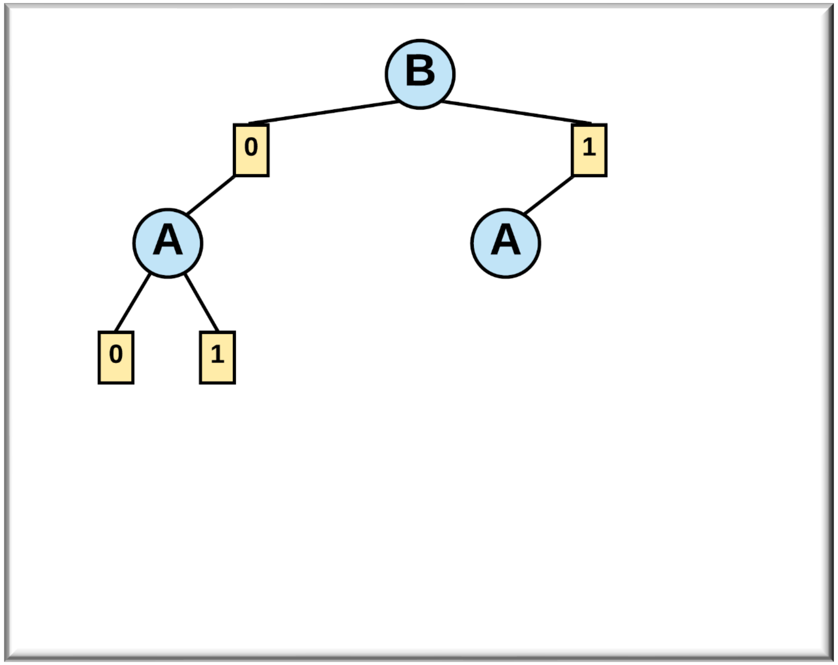
New Scalable AOAS



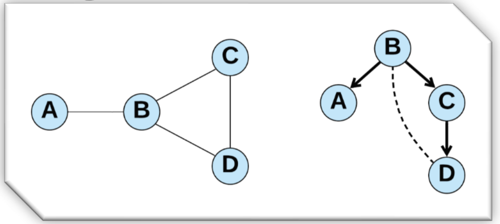
New AND/OR abstraction sampling scheme that allows for non-proper abstractions while still ensuring formation of valid probes.

Key Points:

- ❑ Performs non-proper abstractions
- ❑ Expands along a depth first traversal of the guiding pseudo tree
- ❑ Immediately performs recursive pruning of branches that cannot be part of valid configurations



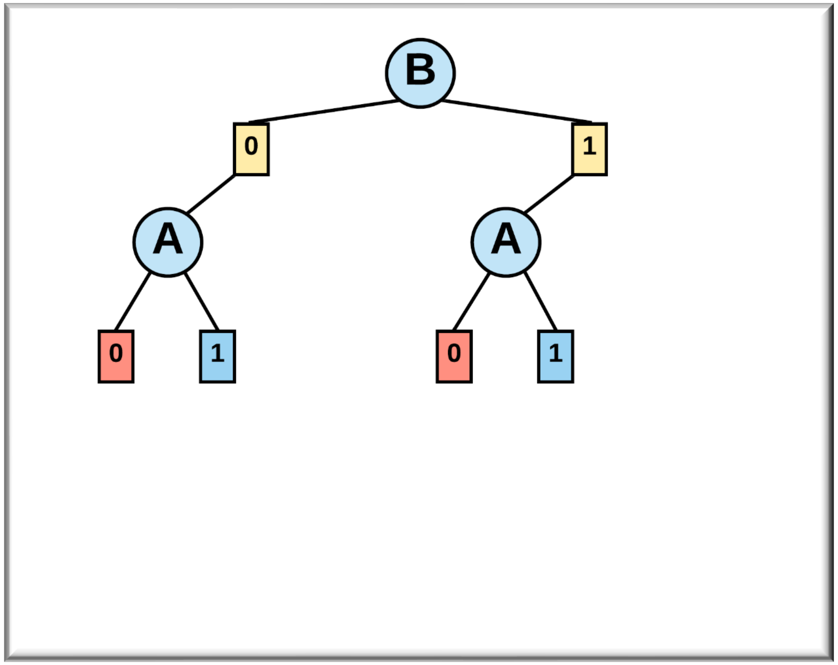
AND/OR Abstraction Sampling



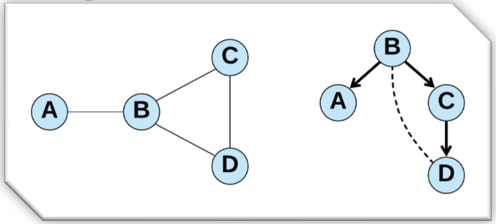
Input: Abstraction function a, (partition the states at each level). Sampling proposal p

Key Points:

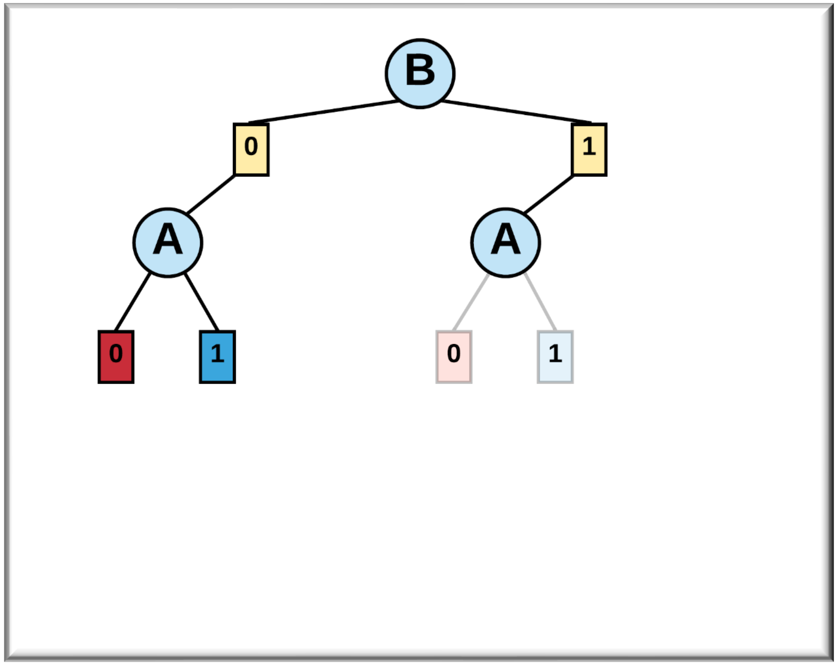
- Expands along a depth first traversal of the guiding pseudo tree
- Perform abstraction at each level
- Immediately performs recursive pruning of branches that cannot be part of valid configurations



AND/OR Abstraction Sampling



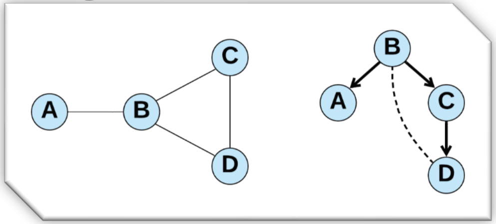
Input: Abstraction function a, (partition the states at each level). Sampling proposal p



Key Points:

- ❑ Expands along a depth first traversal of the guiding pseudo tree
- ❑ Perform abstraction at each level
- ❑ Immediately performs recursive pruning of branches that cannot be part of valid configurations

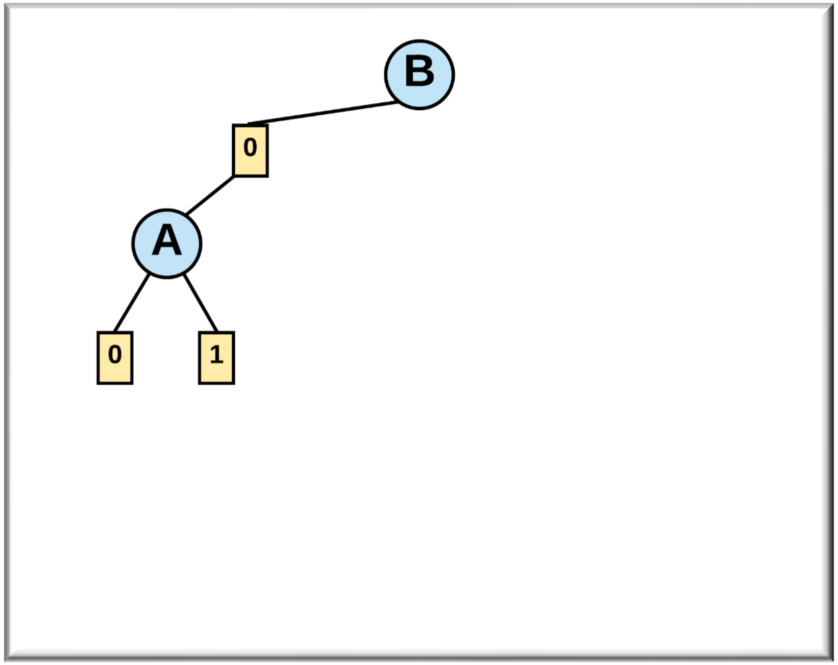
AND/OR Abstraction Sampling



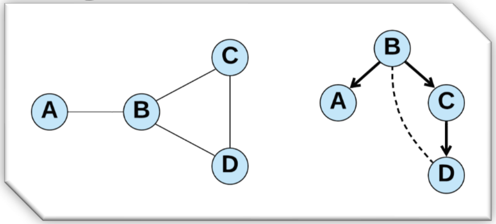
Input: Abstraction function a, (partition the states at each level). Sampling proposal p

Key Points:

- Expands along a depth first traversal of the guiding pseudo tree
- Perform abstraction at each level
- Immediately performs recursive pruning of branches that cannot be part of valid configurations



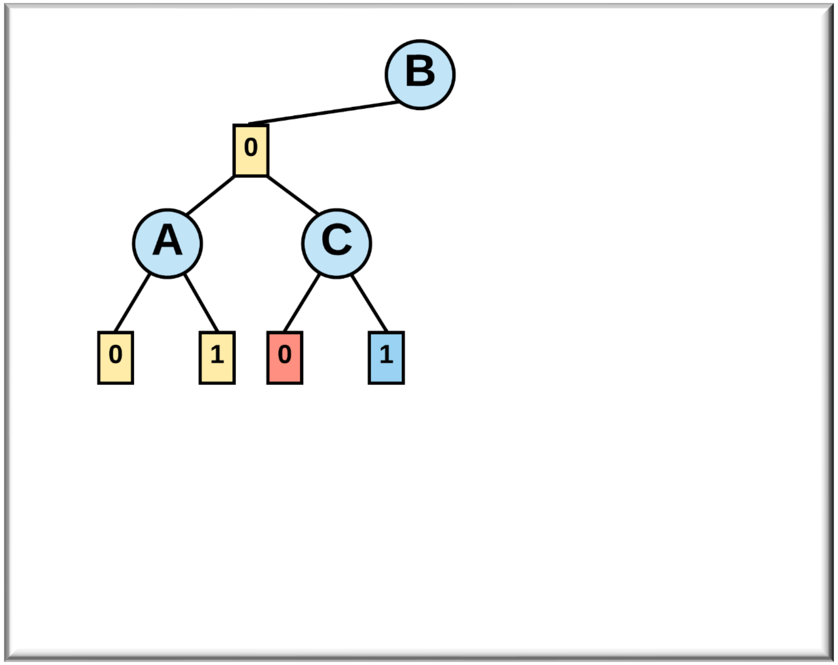
AND/OR Abstraction Sampling



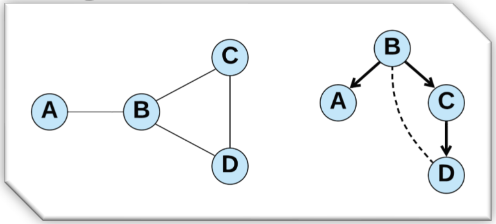
Input: Abstraction function a, (partition the states at each level). Sampling proposal p

Key Points:

- Expands along a depth first traversal of the guiding pseudo tree
- Perform abstraction at each level
- Immediately performs recursive pruning of branches that cannot be part of valid configurations



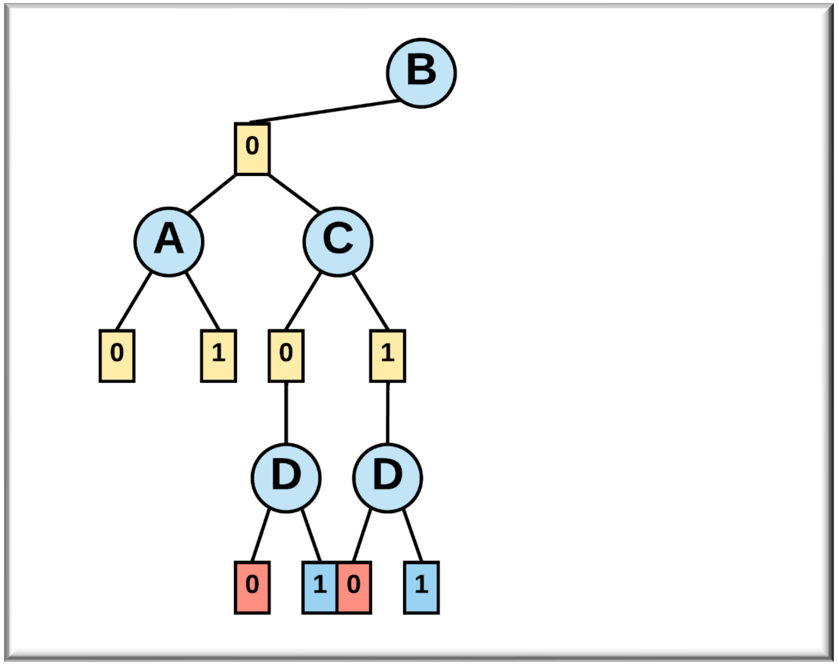
AND/OR Abstraction Sampling



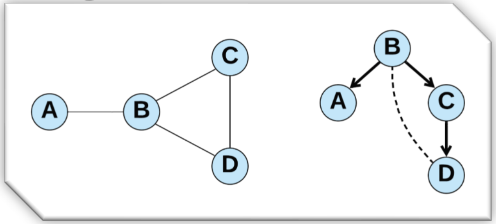
Input: Abstraction function a, (partition the states at each level). Sampling proposal p

Key Points:

- Expands along a depth first traversal of the guiding pseudo tree
- Perform abstraction at each level
- Immediately performs recursive pruning of branches that cannot be part of valid configurations



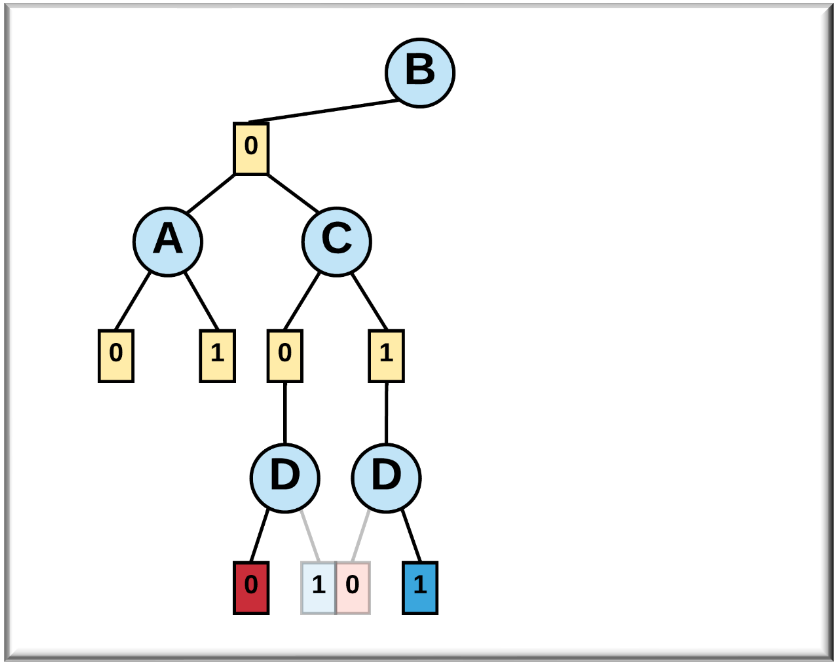
AND/OR Abstraction Sampling



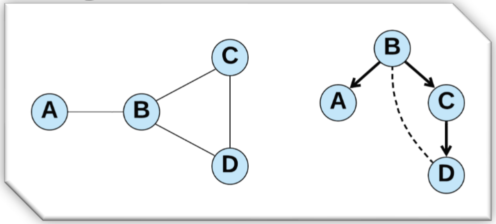
Input: Abstraction function a, (partition the states at each level). Sampling proposal p

Key Points:

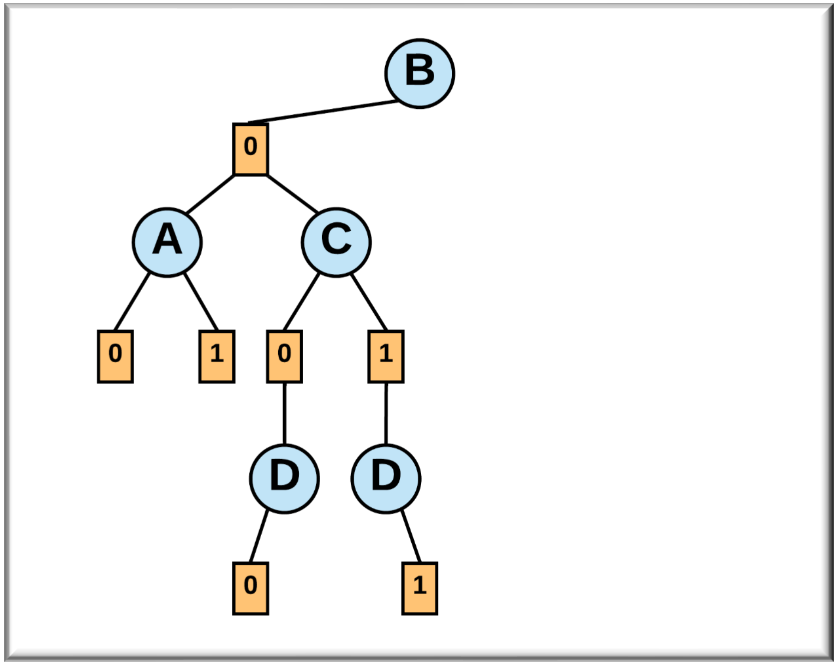
- Expands along a depth first traversal of the guiding pseudo tree
- Perform abstraction at each level
- Immediately performs recursive pruning of branches that cannot be part of valid configurations



AND/OR Abstraction Sampling



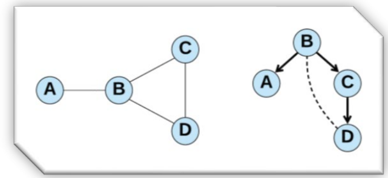
Input: Abstraction function a, (partition the states at each level). Sampling proposal p



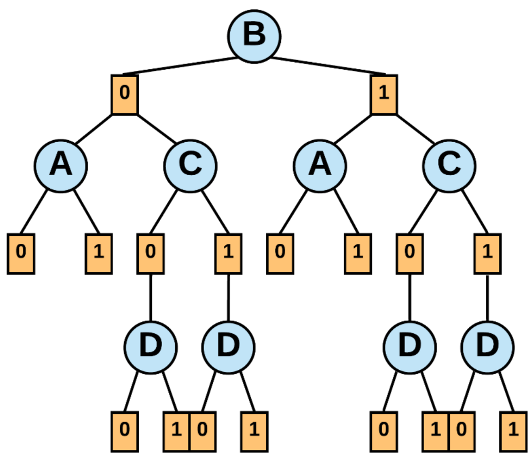
Key Points:

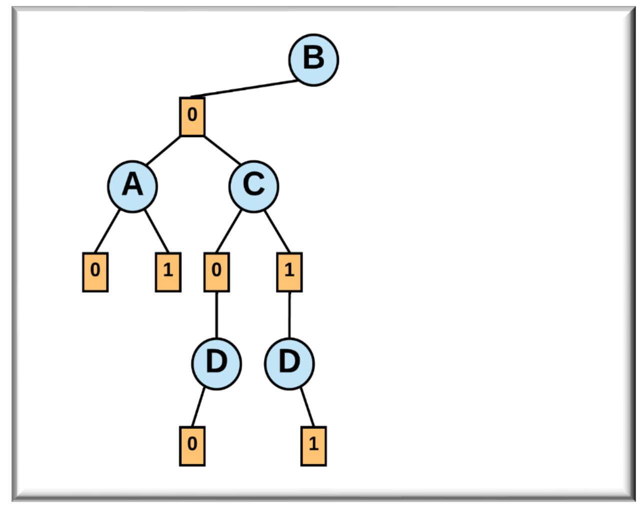
- Expands along a depth first traversal of the guiding pseudo tree
- Perform abstraction at each level
- Immediately performs recursive pruning of branches that cannot be part of valid configurations

AND/OR Abstraction Sampling



Input: Abstraction function a, (partition the states at each level). Sampling proposal p





$$\hat{Z} = \frac{1}{K}\sum_{k=1}^{K} Z'_k$$

$$p(n) \leftarrow \frac{w(n)\cdot g(n)\cdot h(n)\cdot r(n)}{\sum_{m\in A_i} w(m)\cdot g(m)\cdot h(m)\cdot r(m)}$$

Properties

Complexity

$\boxed{O(n \cdot m)}$ where n is the number of variables, and m is the number of abstract states per variable

AOAS is and Unbiased Estimator of the Partition Function

THEOREM **2** (unbiasedness). *Given a graphical model* $\mathcal{M} = (\mathbf{X}, \mathbf{D}, \mathbf{\Phi})$, *algorithm* AOAS *provides an unbiased estimate for the partition function of* $\mathcal{M}$.

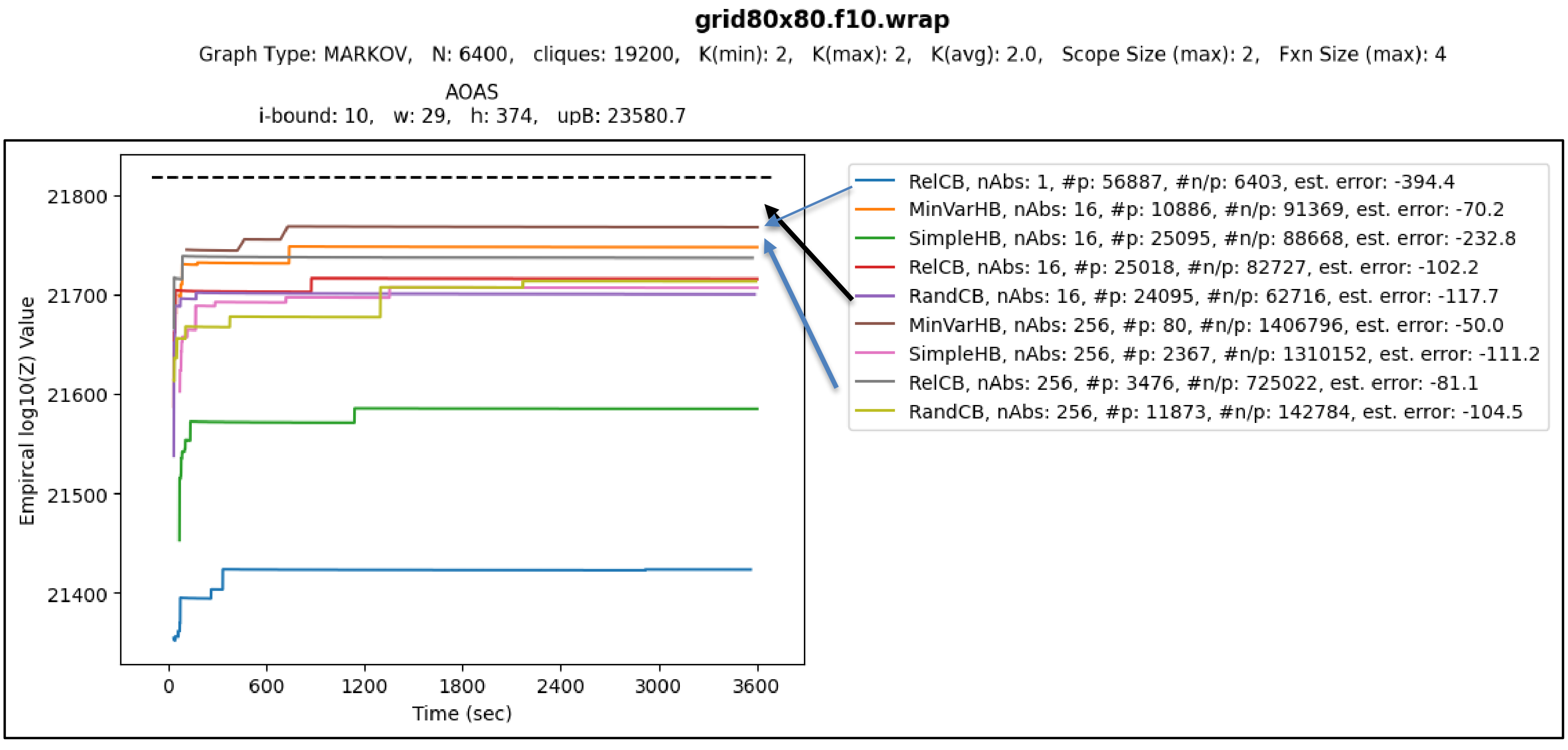
Accuracy/Variance reduction: Stratified Importance Sampling reduce the variance linearly in number of abstract states and the variance between abstract states.

Abstraction Function Comparison

Abstraction Function	Description	Randomized	Refinement Control
randCB	nodes partitioned into abstract states based on assignments to a random subset of their context variables	yes	number of abstract states
relCB	nodes partitioned into abstract states based on equivalent assignments to their most recent context variables	no	number of immediate context variables to consider
simpleHB	nodes partitioned into equal cardinality abstract states after being ordered by their sub-problem heuristic estimates	no	number of abstract states
minVarHB	nodes partitioned into abstract states to minimize the total internal variance of each abstract state w.r.t. node sub-problem heuristic estimates	no	number of abstract states

How can we determine which abstraction and what granularity to use?

Results


grid80x80.f10.wrap
Graph Type: MARKOV, N: 6400, cliques: 19200, K(min): 2, K(max): 2, K(avg): 2.0, Scope Size (max): 2, Fxn Size (max): 4
AOAS
i-bound: 10, w: 29, h: 374, upB: 23580.7
Empircal log10(Z) Value
21800
21700
21600
21500
21400
0
600
1200
1800
2400
3000
3600
Time (sec)
RelCB, nAbs: 1, #p: 56887, #n/p: 6403, est. error: -394.4
MinVarHB, nAbs: 16, #p: 10886, #n/p: 91369, est. error: -70.2
SimpleHB, nAbs: 16, #p: 25095, #n/p: 88668, est. error: -232.8
RelCB, nAbs: 16, #p: 25018, #n/p: 82727, est. error: -102.2
RandCB, nAbs: 16, #p: 24095, #n/p: 62716, est. error: -117.7
MinVarHB, nAbs: 256, #p: 80, #n/p: 1406796, est. error: -50.0
SimpleHB, nAbs: 256, #p: 2367, #n/p: 1310152, est. error: -111.2
RelCB, nAbs: 256, #p: 3476, #n/p: 725022, est. error: -81.1
RandCB, nAbs: 256, #p: 11873, #n/p: 142784, est. error: -104.5

Current Status of AOAS

- AOAS is highly promissing
- Trading off sampling and searching is better over AND/OR space
- Using abstractions yield often superior performance
- A lot more to explore (what abstraction function and what granularity, can we learn the abstraction function)
- But no bounds. Only unbiasedness.

New UAI Competition

- UAI Competition 2022

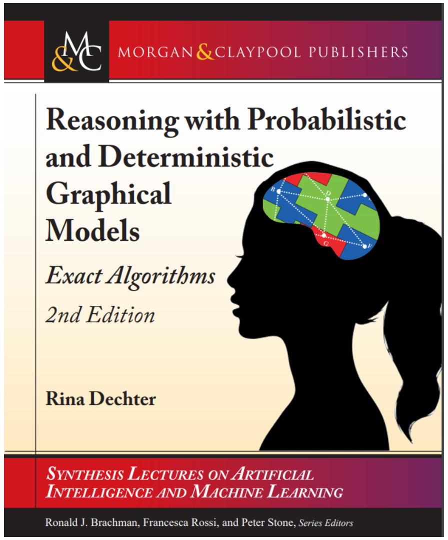
Solver	20sec	1200sec	3600sec
uai14-pr	61.7	96.8	96.7
ibia-pr	53.6	96.6	97.1
AbstractionSampling	78.9	91.7	93.9
lbp-pr	90.3	89.9	90.2

Summary

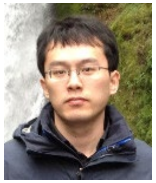
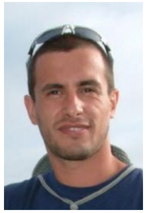
- AND/OR search spaces vs. Probabilistic circuits
- Review AND/OR search spaces for PGM
- AND/OR Multi-valued Decision Diagrams (AOMDD)
- Anytime algorithms over AND/OR search spaces
- **AND/OR Abstraction sampling.**
- **Moving forward: Neuro-symbolic, causality**

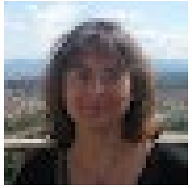
Thank You !

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

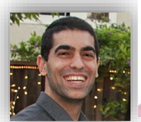


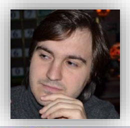



Alex Ihler
Kalev Kask
Irina Rish
Bozhena Bidyuk
Robert Mateescu
Radu Marinescu
Vibhav Gogate
Emma Rollon
Lars Otten
Natalia Flerova
Andrew Gelfand
William Lam
Filjor Broka
Junkyu LeeQi Lou
Bobak Pezeshki



UNIVERSITY of CALIFORNIA IRVINE