Part 1

Advancing AND/OR Abstraction Sampling

Part 2

K*-Based Computational Protein Design using AND/OR Search

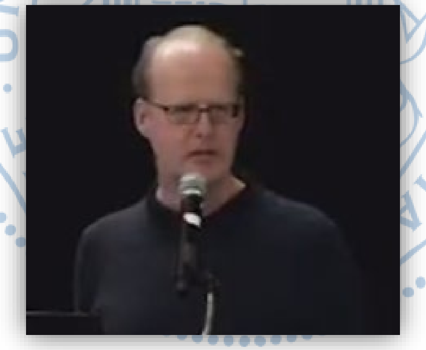
Bobak Pezeshki's PhD Final Defense

(Advised by Prof. Rina Dechter and Prof. Alexander Ihler)

Advancing AND/OR Abstraction Sampling

Kalev Kask, **Bobak Pezeshki**, Filjor Broka, Alex Ihler, and Rina Dechter. "Scaling Up AND/OR Abstraction Sampling". *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI-20).*

Bobak Pezeshki, Kalev Kask, Alex Ihler, and Rina Dechter. "Value-Based Abstraction Functions for Abstraction Sampling". *Proceedings of the 40th Conference on Uncertainty in Artificial Intelligence (UAI 2024).*



KALEV KASK,

BOBAK PEZESHKI,

FILJOR BROKA,

ALEX IHLER,

RINA DECHTER

Outline

- General Background
- Abstraction Sampling
  - General Scheme
  - AND/OR Abstraction Sampling
    - AOAS Algorithm
    - Analysis of its Properties
  - Abstraction Function Schemes
    - Context-based Abstraction Functions
    - Value-based Abstraction Functions
    - Completely Random Abstractions
- Empirical Evaluation
- Conclusion

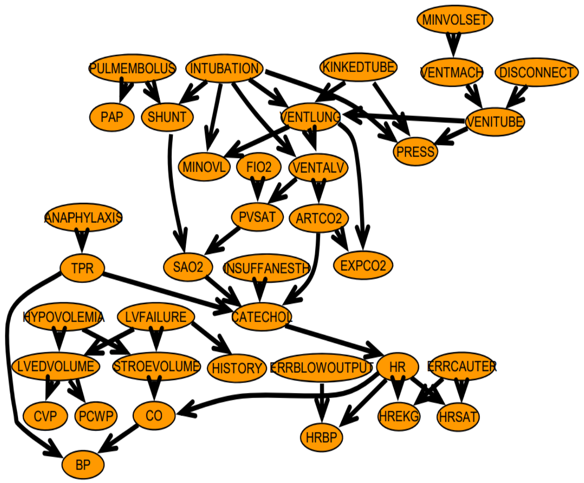
Main Contributions

- General Background
- Abstraction Sampling
    - General Scheme
    - AND/OR Abstraction Sampling
        - AOAS Algorithm
        - Analysis of its Properties
    - Abstraction Function Schemes
        - Context-based Abstraction Functions
        - Value-based Abstraction Functions
        - Completely Random Abstractions
- Empirical Evaluation
- Conclusion and Future Work

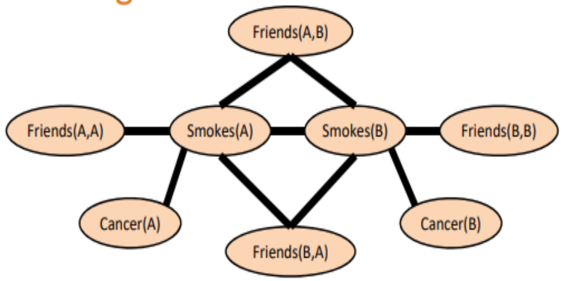
Background

Graphical Models – Overview

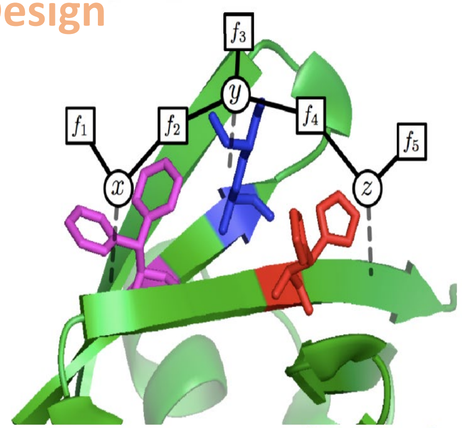
Bayesian Networks



Markov Logic

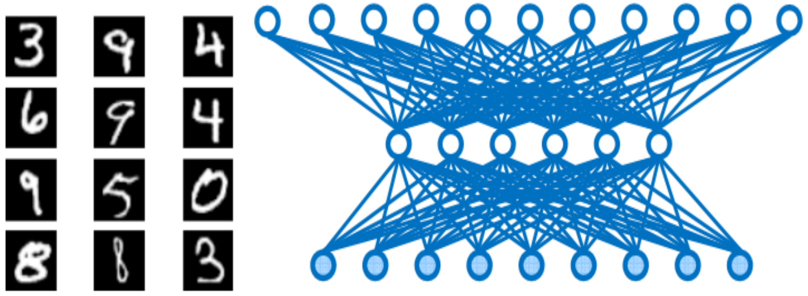


Protein Folding and Design

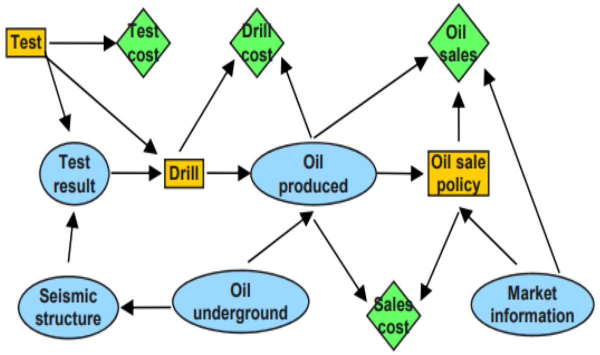


[Yanover & Weiss 2002]

Deep Boltzmann Machines



Influence Diagrams



Graphical Models – Formal Definition

$\mathcal{M} = \{$

$\mathbf{X} = \{ X_1, X_2, \ldots, X_N \}$ ← **Variables**

$\mathbf{D} = \{ D_{X_1}, D_{X_2}, \ldots, D_{X_N} \}$ ← **Domains**

$\mathbf{F} = \{ f_{\alpha_1}, f_{\alpha_2}, \ldots, f_{\alpha_M} \}$ ← **Factors**

$\}$

Example:

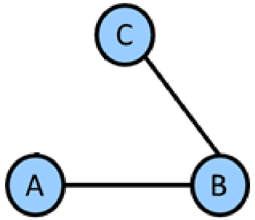
$A \in \{0,1\}$

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

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

A	B	f(A,B)
0	0	2
0	1	4
1	0	3
1	1	1

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



Primal graph

A combination operator ⊗ defines a global function.

$$p(A,B,C) \propto f_{AB}(A,B) \times f_{BC}(B,C)$$

ex. ⊗ = multiplication

Tasks

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

$$\underbrace{Z = \sum_{\boldsymbol{X}} \prod_{\alpha} f_\alpha(X_\alpha)}$$

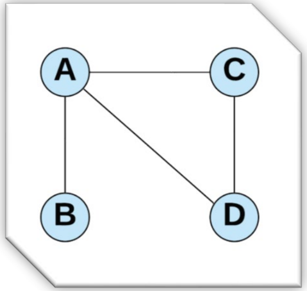
partition function
(#P-complete)
[Cooper, 1990]

Harder

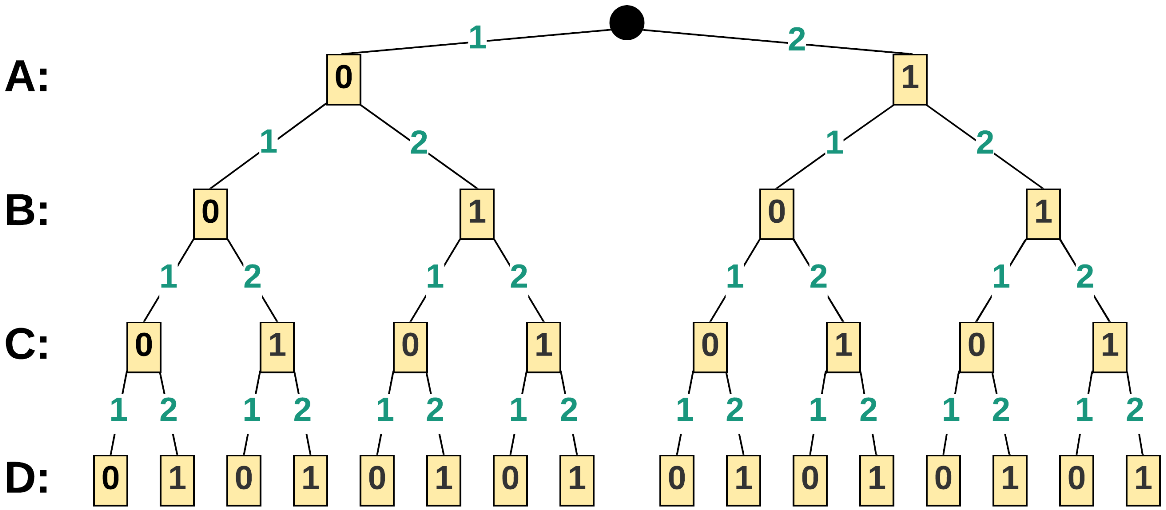
Computing marginals: $P(X_i) = \frac{1}{Z} \sum_{\boldsymbol{X}/X_i} \prod_{\alpha} f_\alpha(X_\alpha)$

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

Systematic Search vs Sampling

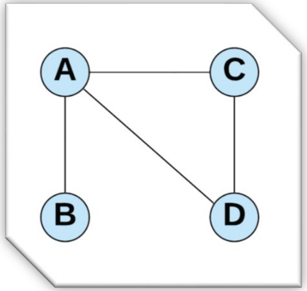


Systematic Search

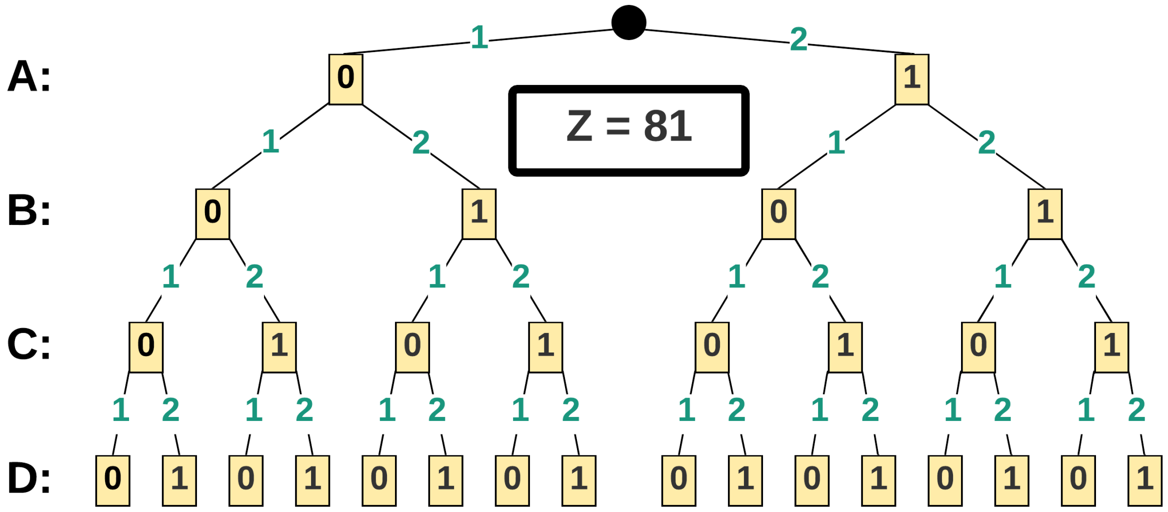


- Enumerate states
- Every stone turned
- No stone turned more than once

Systematic Search vs Sampling

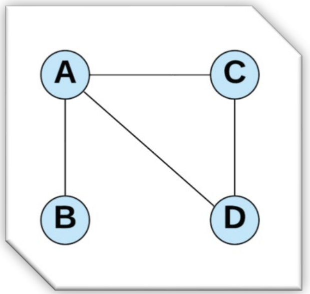


Systematic Search

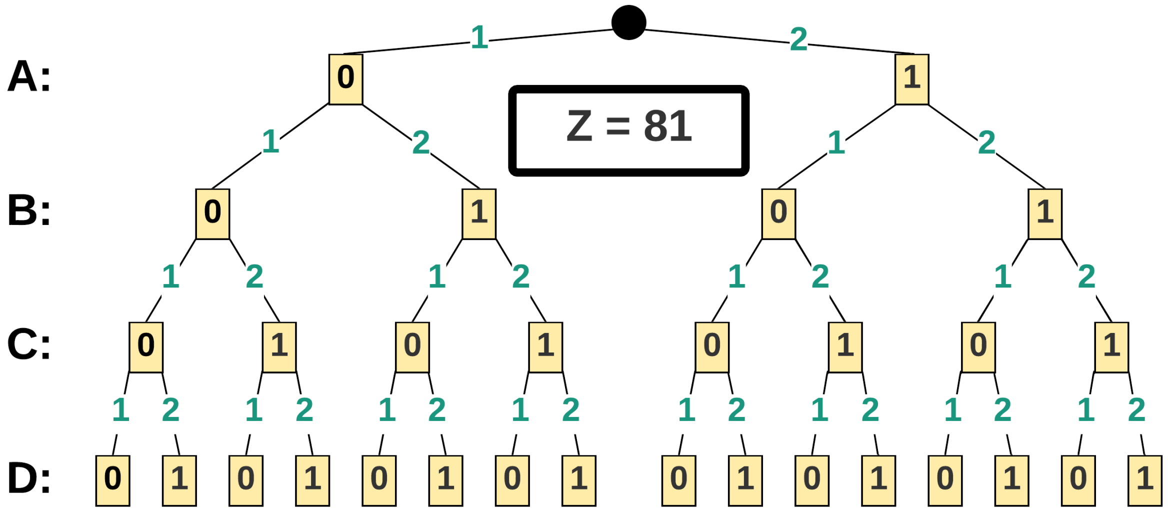


- Enumerate states
- Every stone turned
- No stone turned more than once

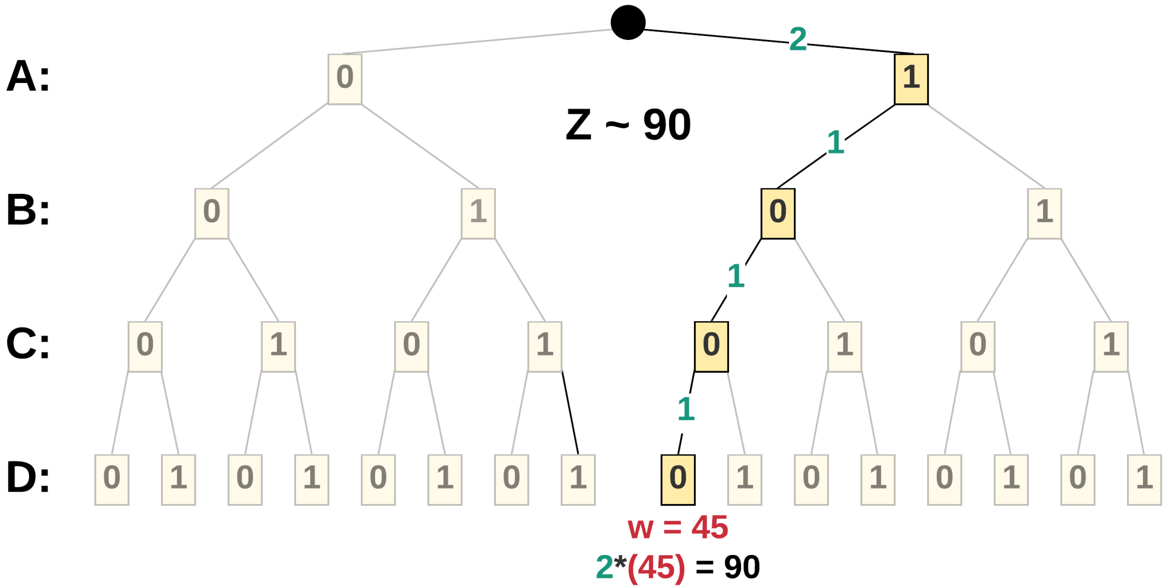
Systematic Search vs Sampling



Systematic Search



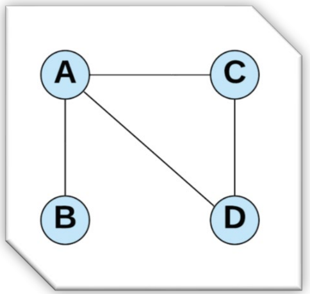
Importance Sampling [Liu, 2001]



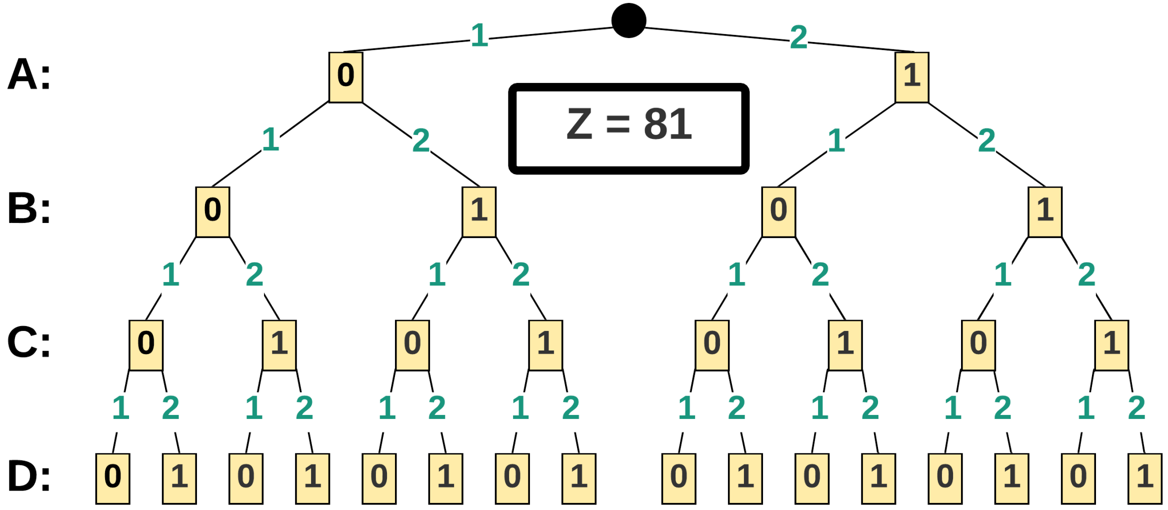
- Enumerate states
- Every stone turned
- No stone turned more than once

- Monte Carlo sampling method

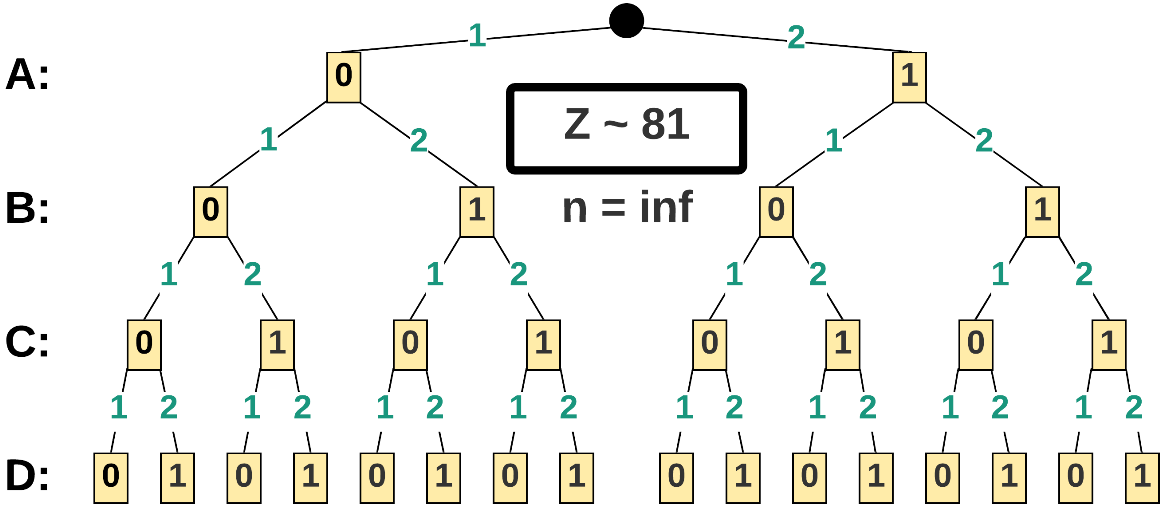
Systematic Search vs Sampling



Systematic Search



Importance Sampling [Liu, 2001]

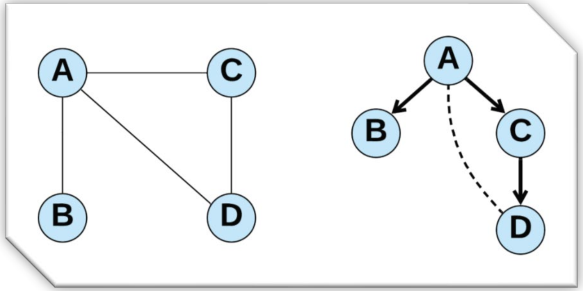


- Enumerate states
- Every stone turned
- No stone turned more than once

- Monte Carlo sampling method

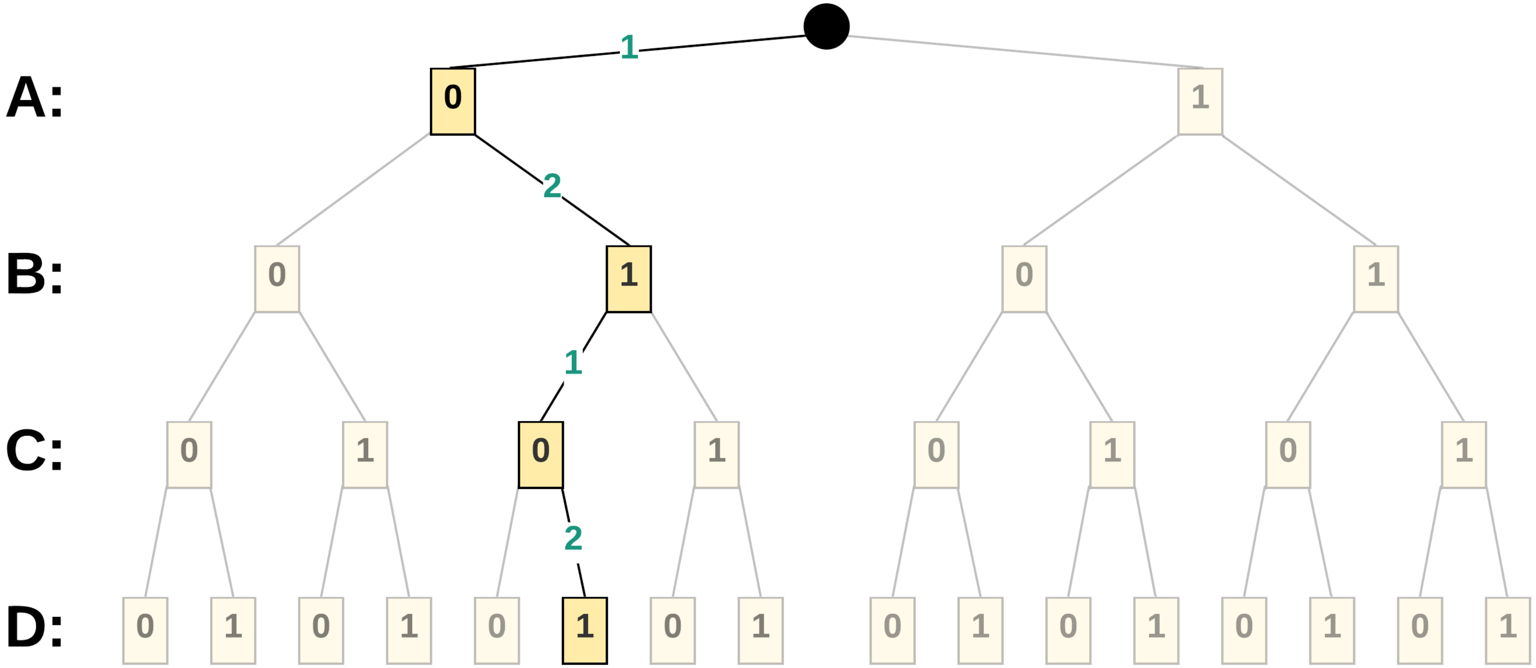
[J. Liu, Monte-Carlo strategies in scientific computing, Springer-Verlag, New York, 2001]

AND/OR Search Space



Compact search space taking advantage of conditional independencies

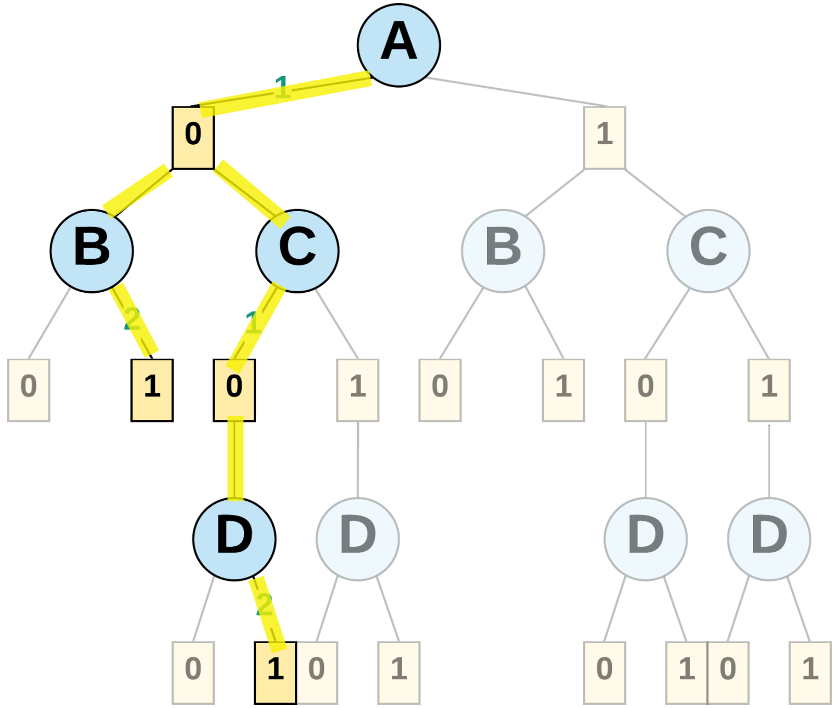
<u>Classical OR Search Space</u>



$g(A=0, B=1, C=2, D=1) = 1 \times 2 \times 1 \times 2 = 4$

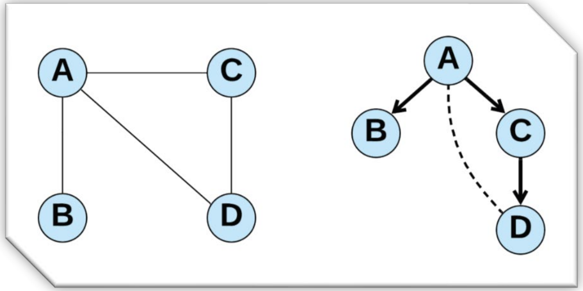
<u>Compact AND/OR Search Space</u>

[Dechter and Mateescu, 2004]

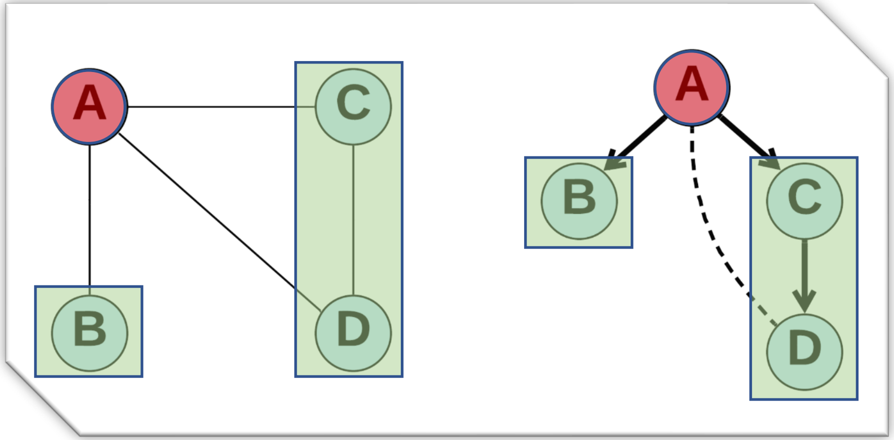


$g(A=0, B=1, C=2, D=1) = 1 \times ((2) \times (1 \times 2)) = 4$

Guiding Pseudo-Tree



Pseudo-Trees capture conditional independencies and guide the construction of the search space.

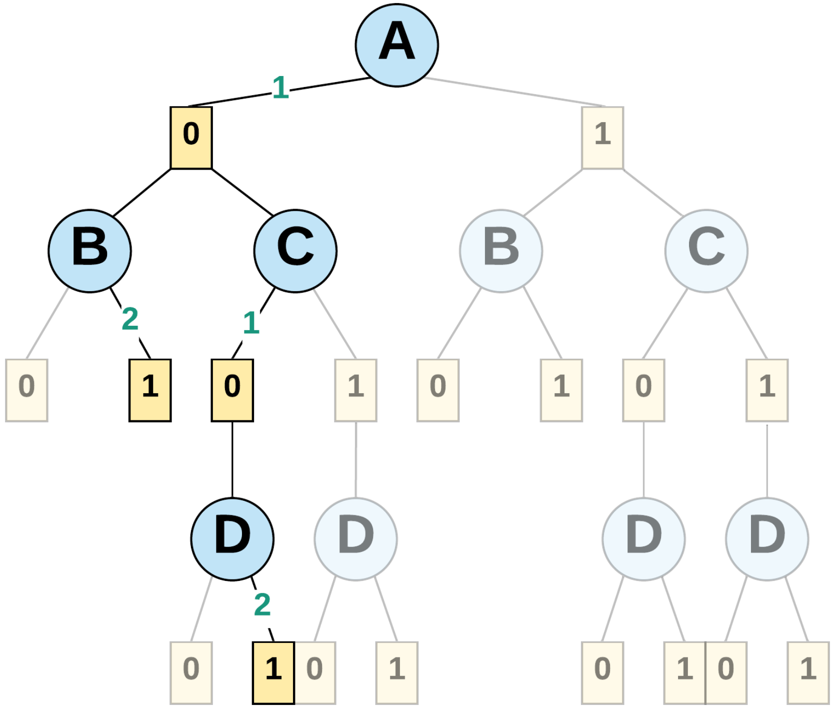


Primal Graph

Pseudo-Tree

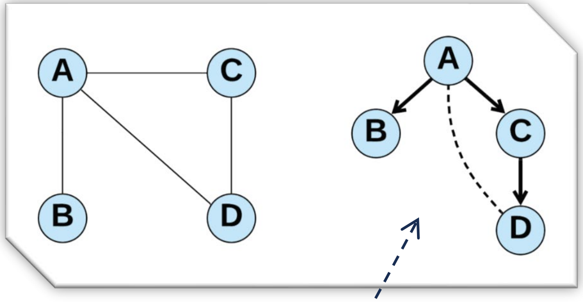
Compact AND/OR Search Space

[Dechter and Mateescu, 2004]

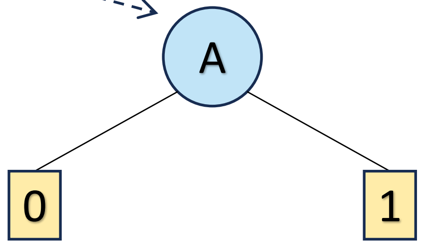


$g(A=0, B=1, C=2, D=1) = 1\times((2)\times(1\times2)) = 4$

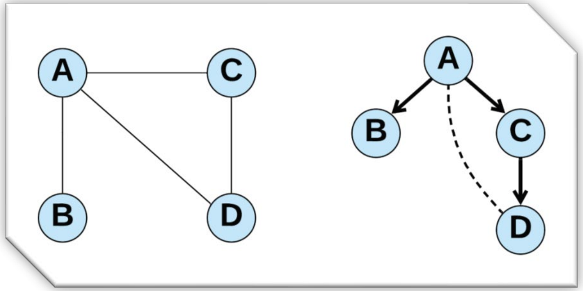
IS in AND/OR Trees



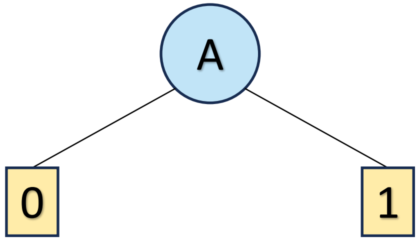
Progress variable-by-variable according to the guiding pseudo tree



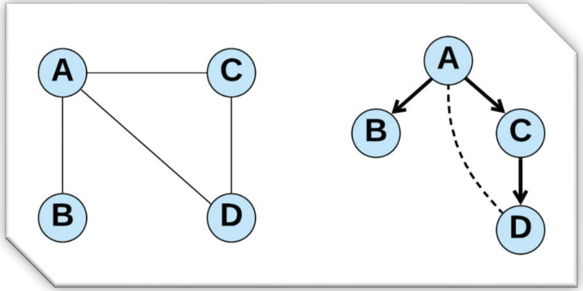
IS in AND/OR Trees



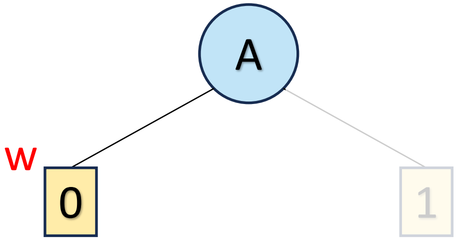
Stochastically select a value to assign the variable according to a proposal distribution, p.



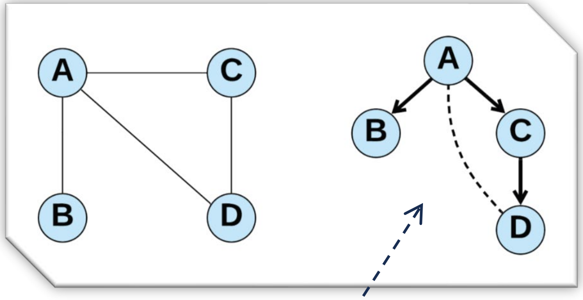
IS in AND/OR Trees



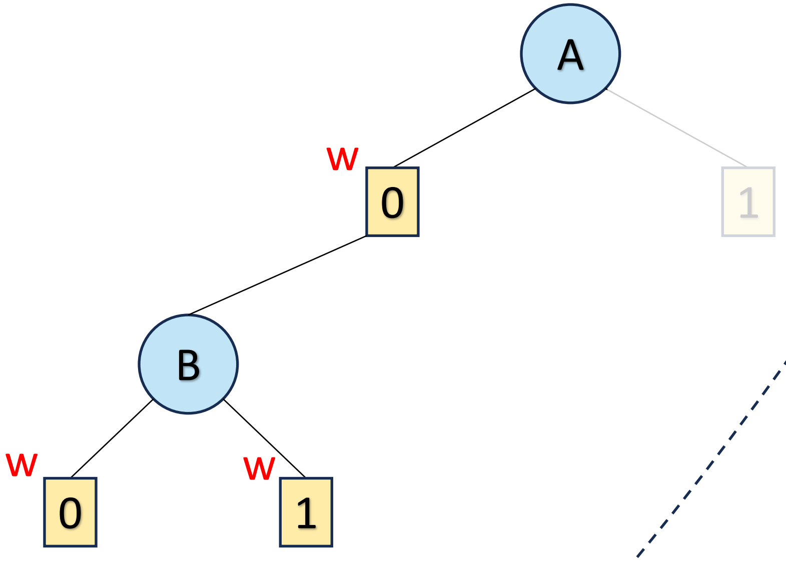
Update importance weight according to $w(n) = w/p(n)$



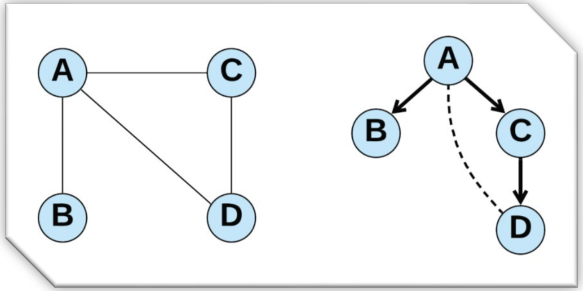
IS in AND/OR Trees



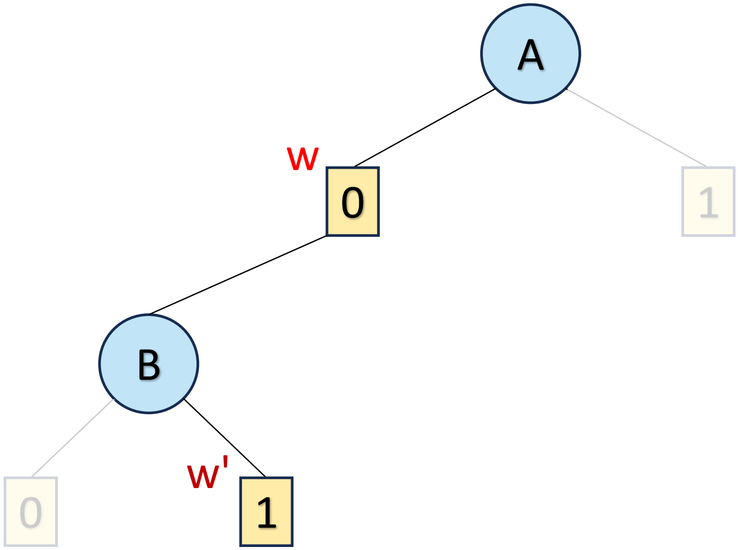
Expand to a next variable in the pseudo tree ordering



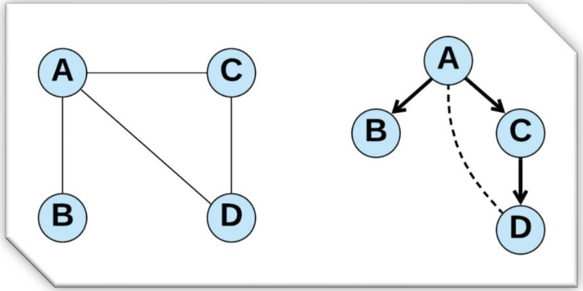
IS in AND/OR Trees



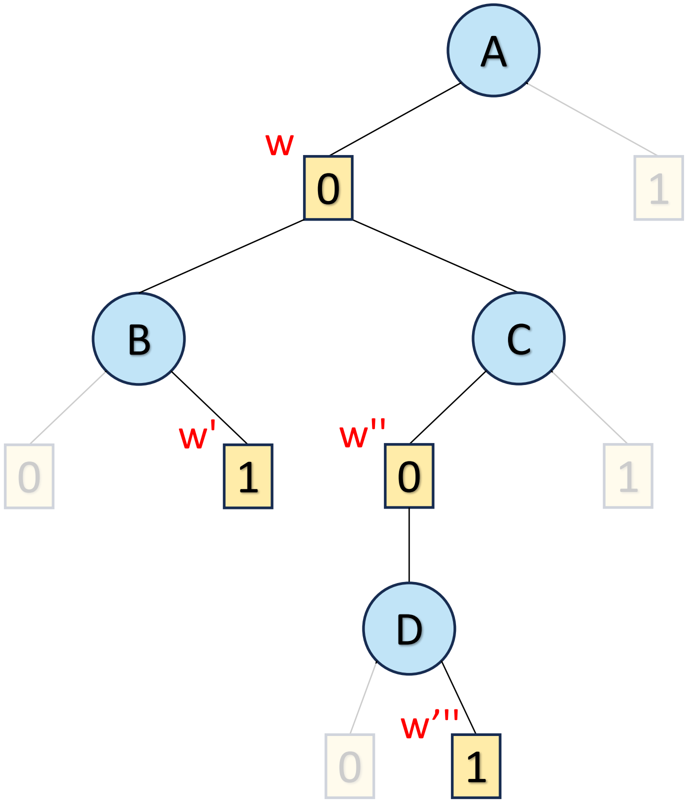
Stochastically assign value to variable according to proposal and update weights accordingly



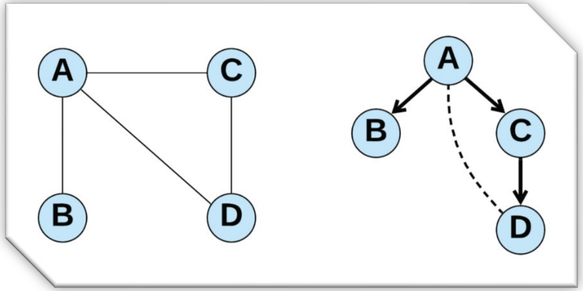
IS in AND/OR Trees



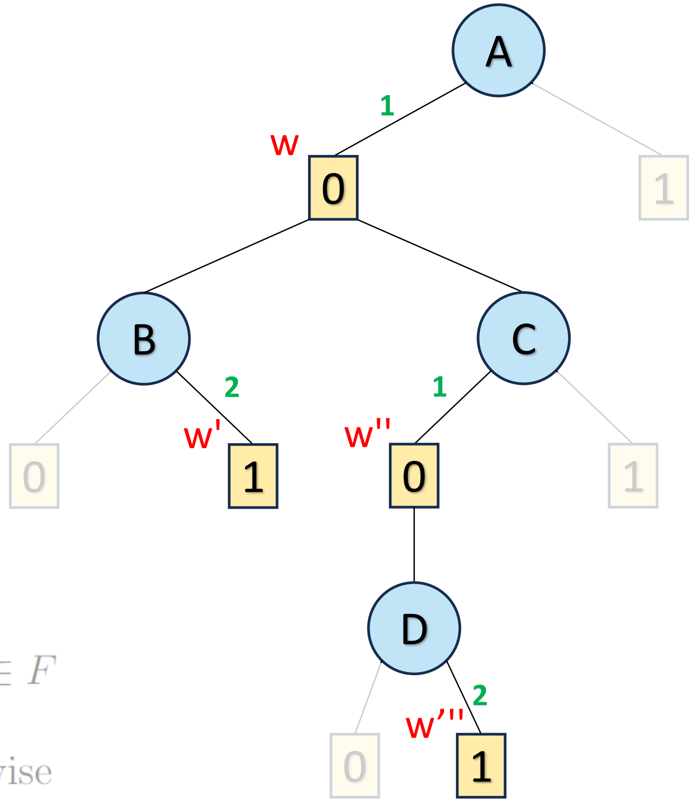
Repeat until every variable is assigned a value (a *solution tree* is sampled)



IS in AND/OR Trees



An estimate can be produced considering the cost associated with the sampled solution tree upweighted by the assigned importance weights.



$$\hat{V}^{(t)}(n_X) = \begin{cases} 1, & \text{if } n_X \in F \\ \prod_{Y \in ch_{\mathcal{T}}(X) \setminus \mathcal{T}'} V(Y_{n_X}) \cdot \prod_{Y \in ch_{\mathcal{T}}(X) \cap \mathcal{T}} \sum_{n_Y \in ch_Y(n_X)} \frac{w^{(t)}(n_Y)}{w^{(t)}(n_X)} c(n_Y)\, \hat{V}^{(t)}(n_Y), & \text{otherwise} \end{cases}$$

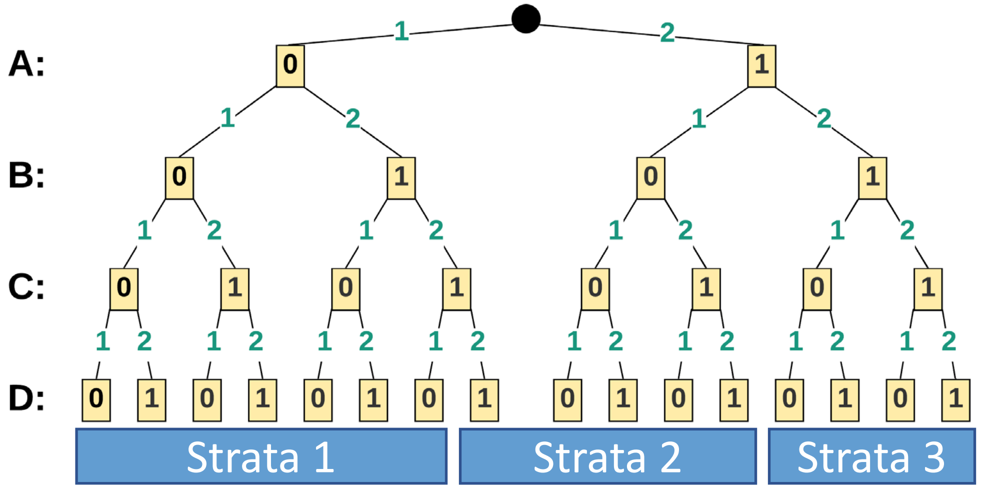
Stratified [Importance] Sampling

[Knuth, 1975], [Chen, 1992], [Rizzo, 2007]

Main idea: partially enumerate and partially sample search space

Steps:

- ❑ Subdivide space into set strata
- ❑ Enumerate strata choosing reweighted samples from each to form a probe
- ❑ Average estimates from sampled probes



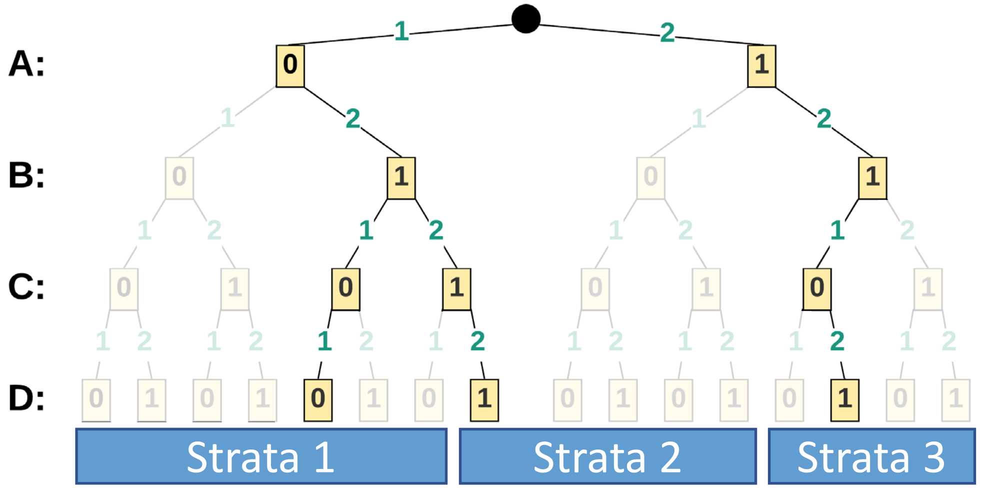
Stratified [Importance] Sampling

[Knuth, 1975], [Chen, 1992], [Rizzo, 2007]

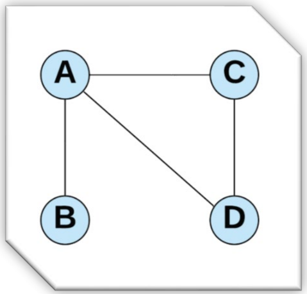
Main idea: partially enumerate and partially sample search space

Steps:

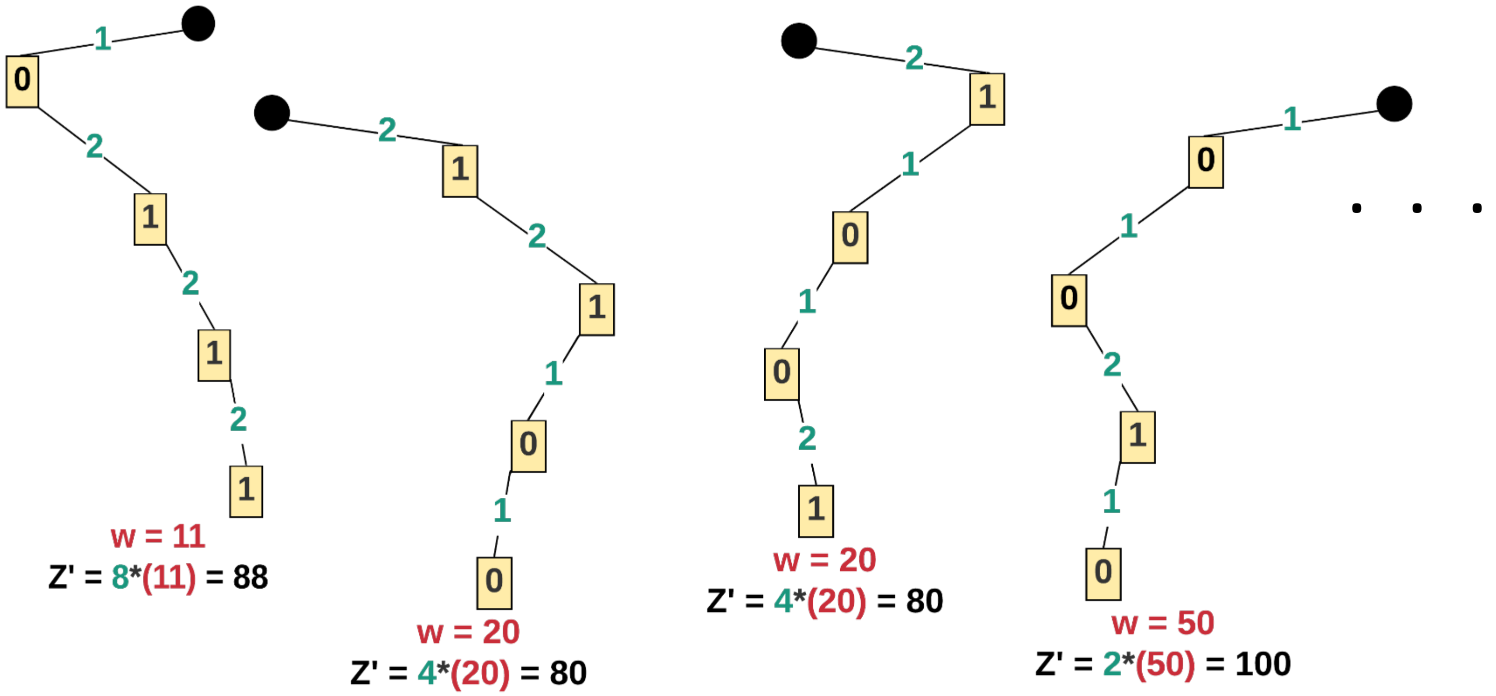
- ❑ Subdivide space into set strata
- ❑ Enumerate strata choosing reweighted samples from each to form a probe
- ❑ Average estimates from sampled probes



Interpolating Between Sampling and Search



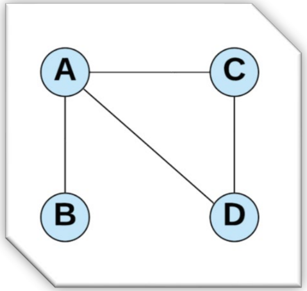
Importance Sampling



K independent samples

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

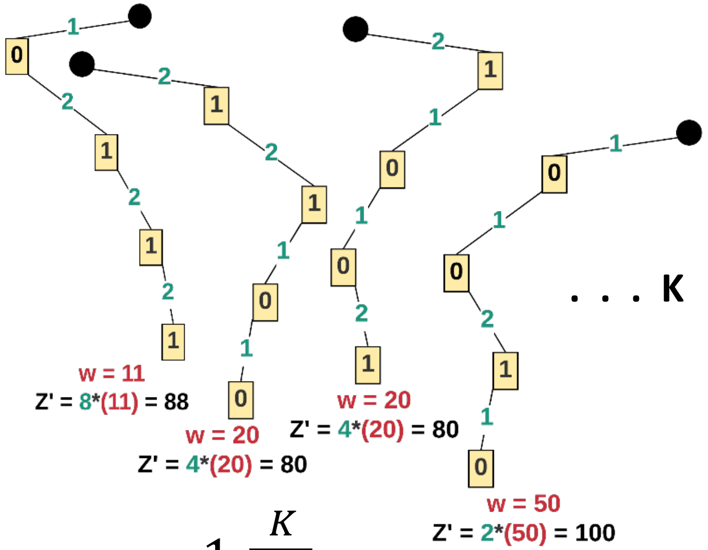
Interpolating Between Sampling and Search



We can draw samples of multiple configurations to more closely resemble search.

Importance Sampling

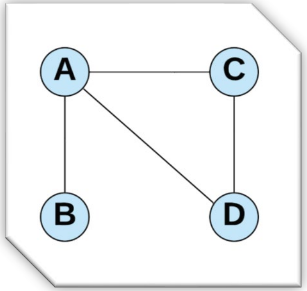
K independent samples



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

Sampling **Search**

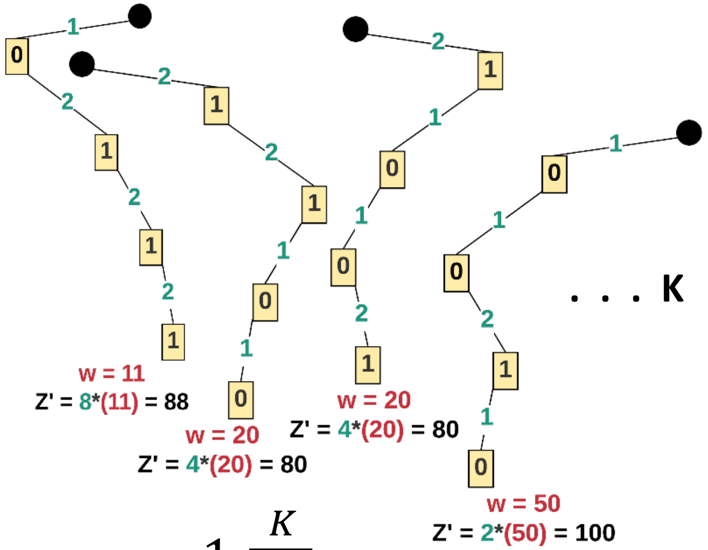
Interpolating Between Sampling and Search



We can draw samples of multiple configurations to more closely resemble search.

Importance Sampling

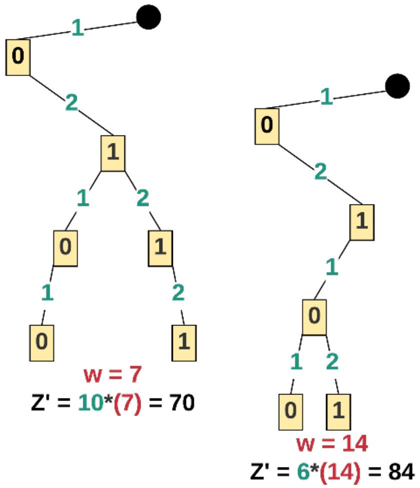
K independent samples



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

2-Config Sampling

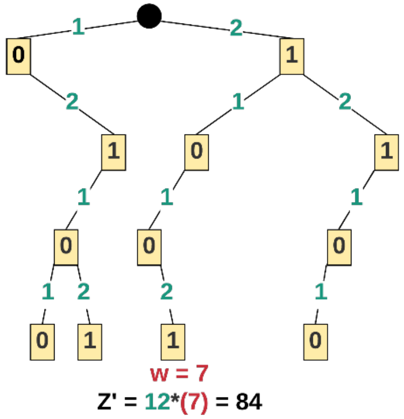
$\frac{K}{2}$ independent samples



$$\hat{Z} = \frac{2}{K}\sum_{k=1}^{K/2} Z'_k$$

4-Config Sampling

$\frac{K}{4}$ independent samples



$$\hat{Z} = \frac{4}{K}\sum_{k=1}^{K/4} Z'_k$$

Is it worth it?

Variance reduction of

$$\boxed{k^2\, Var(\hat{Z}_J)}$$

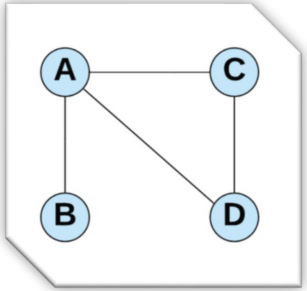
under certain conditions

[Rizzo, 2007]

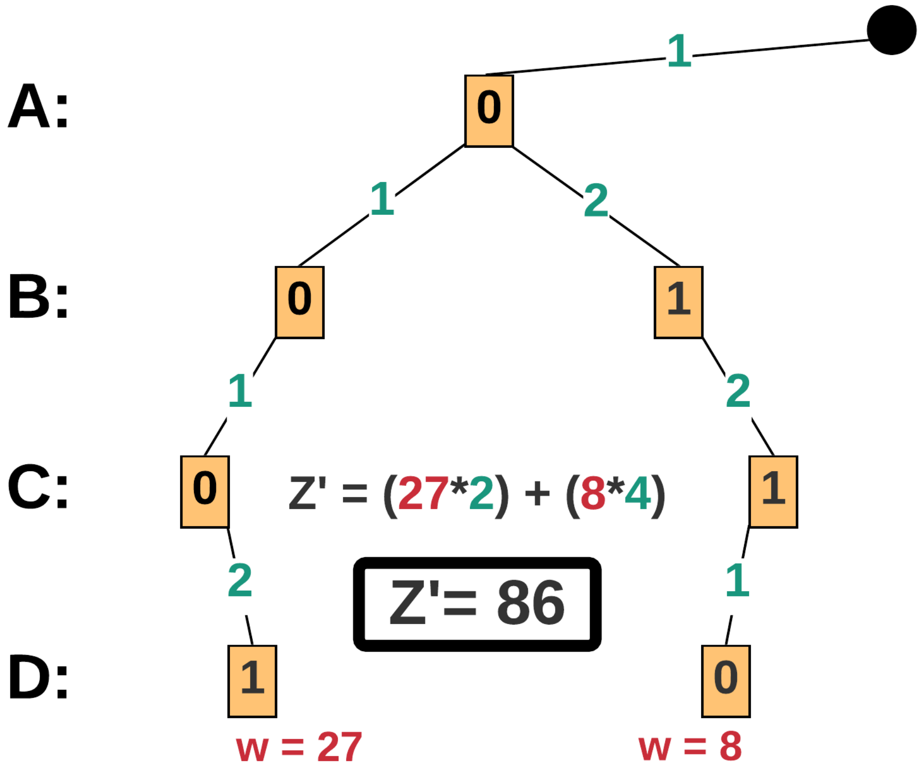
Sampling → **Search**

Abstraction Sampling

General AS Scheme *[Broka, Dechter, Ihler, and Kask, 2018]*



A sampling scheme that enables the interpolating between sampling and search by performing abstractions level-by-level.



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

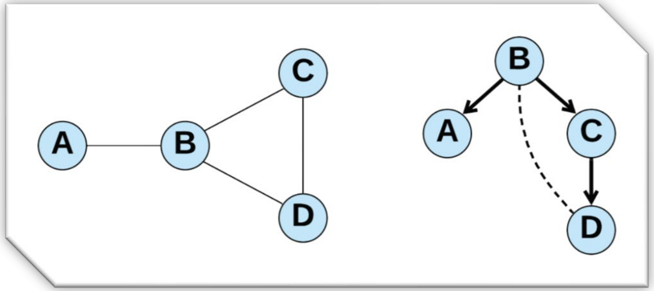
Original Work Set A Foundation…

Main Questions 1:

- How to adapt to the more compact AND/OR spaces?
  - Should valid samples consist of only solution subtrees?
  - How to abstract across different branches of the AND/OR tree?

Scalable AND/OR Abstraction Sampling (algorithm: AOAS)

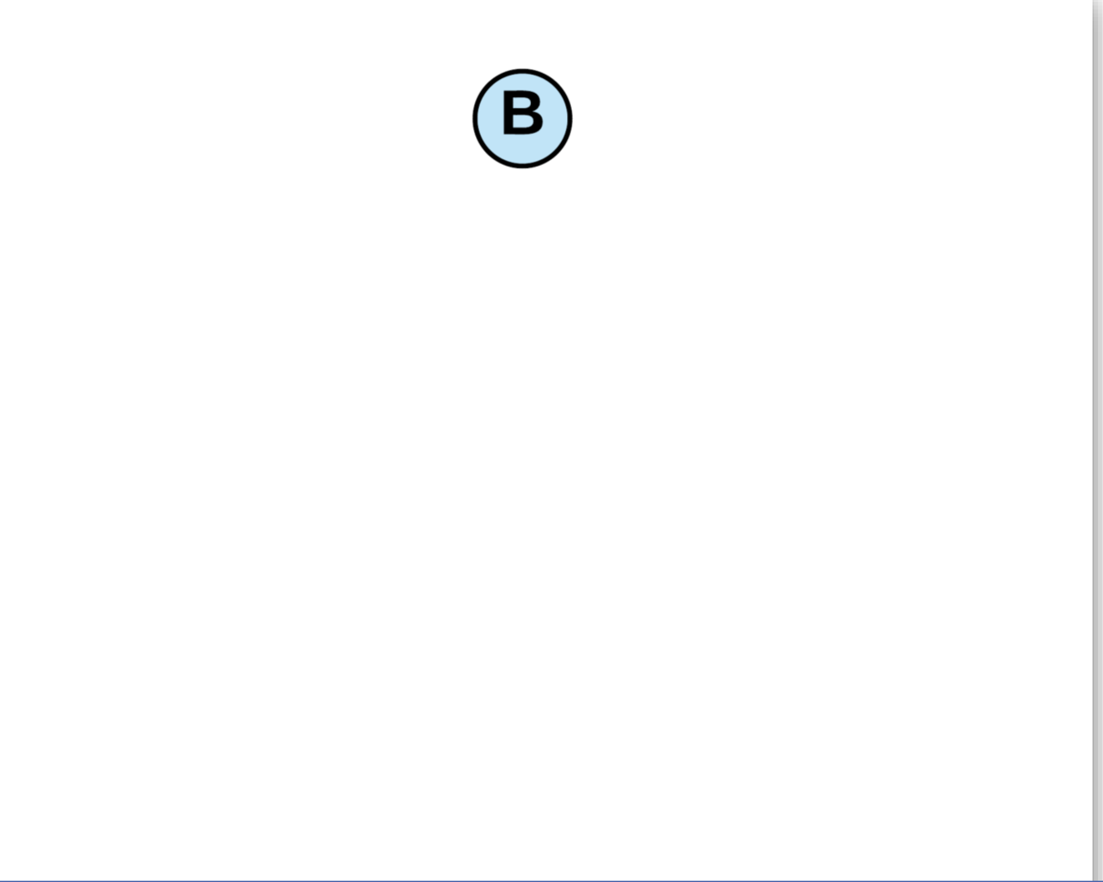
AOAS



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

Key Points:

- ❑ Allows for flexible 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

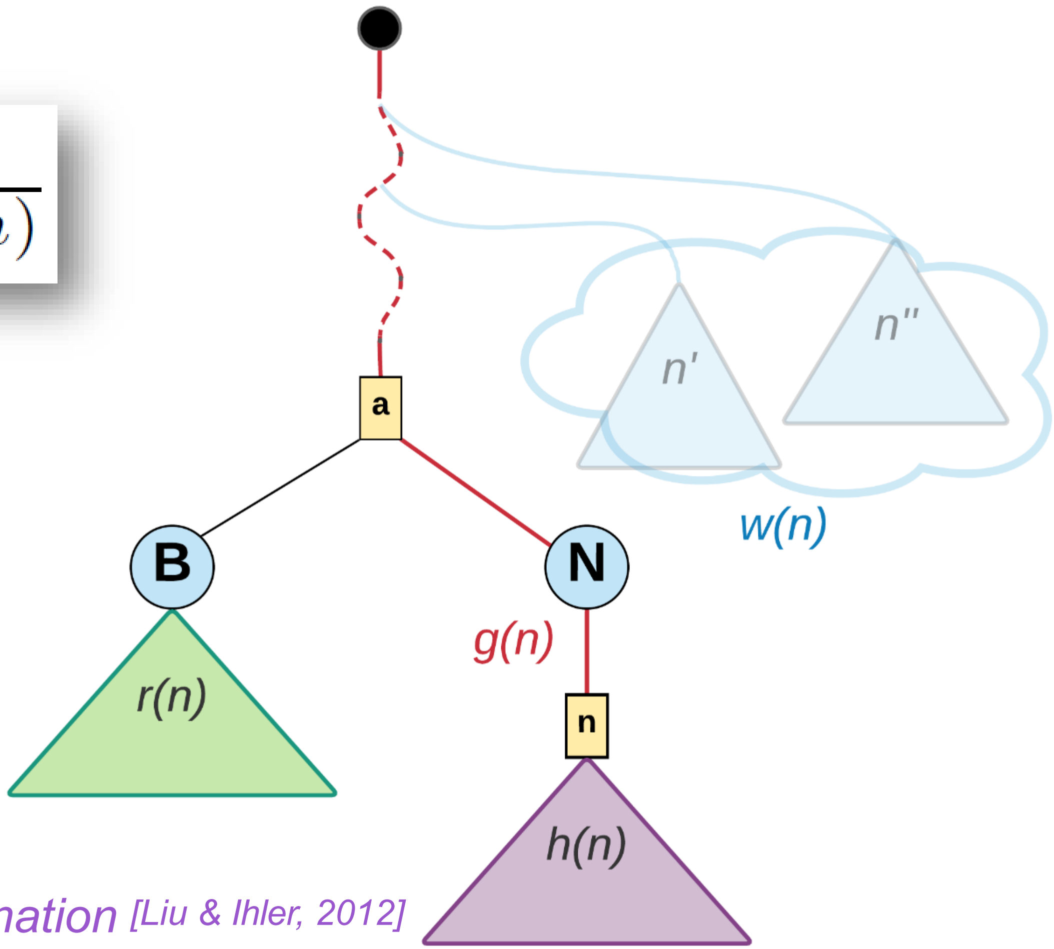


Proposal Distribution

A heuristic function h is used to estimate the value of unexplored subtrees.

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

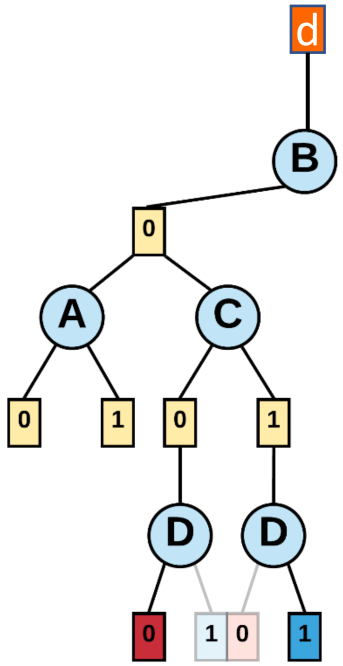
- *w(n)* captures the estimated weight of subtrees absorbed by the *n*'s ancestors during abstraction
- *g(n)* is the path cost from the root to *n*
- *h(n)* is the estimated mass of the subtree *n* roots
- *r(n)* is the estimated ancestor branching mass of *n*



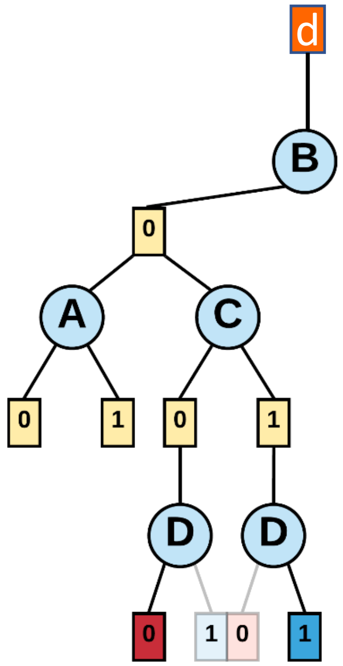
Estimated via Weighted Mini-bucket Elimination [Liu & Ihler, 2012]

Properties (Unbiasedness)

- Proof-Strategy
  - Key observation: at each step in the algorithm, either
    - The probe is expanded
    - An abstraction occurs
    - Pruning occurs



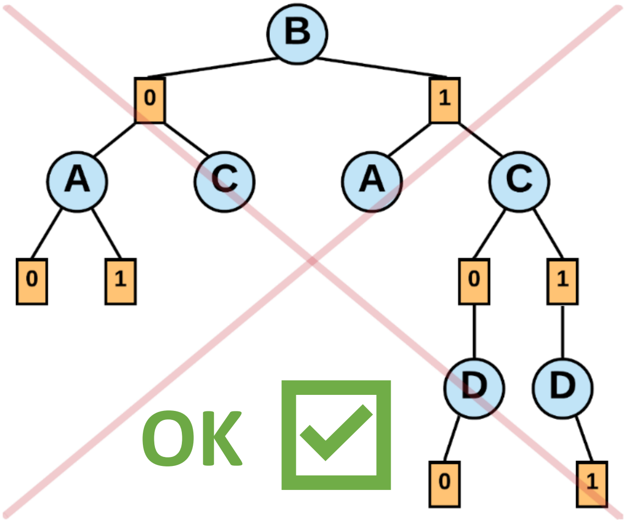
Properties (Unbiasedness)



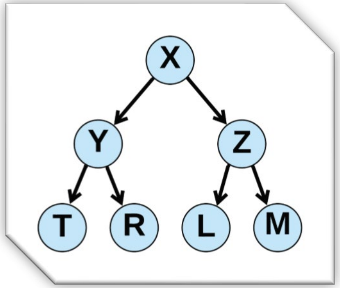
- Proof-Strategy
  - Key observation: at each step in the algorithm, either
    - The probe is expanded
    - An abstraction occurs
    - Pruning occurs
  - Main Idea:
    - Construct an estimator that equals
      - the exact Z value for the unexpanded probe ***(base case)***
      - the value of AOAS's estimator for the final probe
      - needs to include consideration of different branchings in the tree
      - can be computed by analyzing the frontier nodes of a single variable
    - Show that, at each step, the expectation of the estimator remains unchanged

Properties (Unbiasedness)

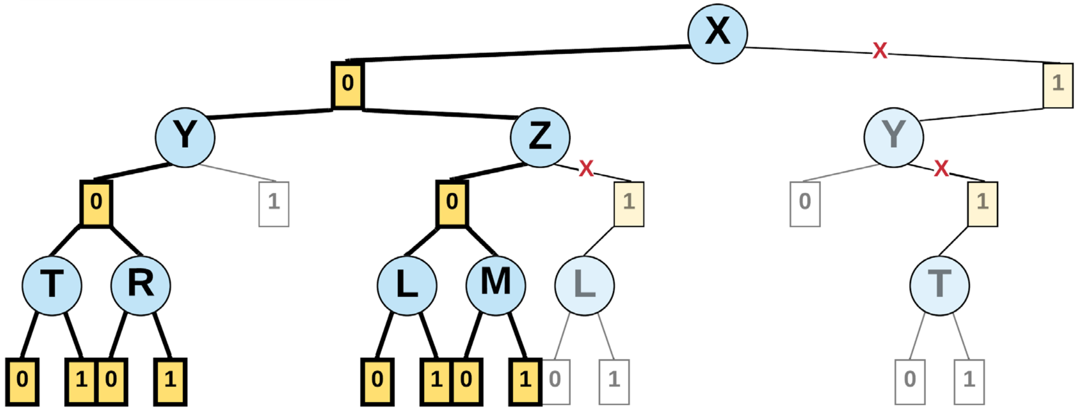
- Illuminating characteristics
  - Works for any valid importance sampling proposal distribution
  - Generalizes to BF expansion of the pseudo-tree
  - Generalizes to algorithms that allow non-solution trees as samples



Properties (Complexity)



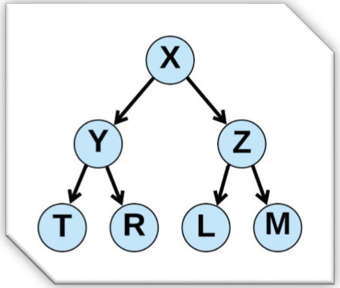
$O(n \cdot m)$ where n is the number of variables, and m is the number of abstract states per variable



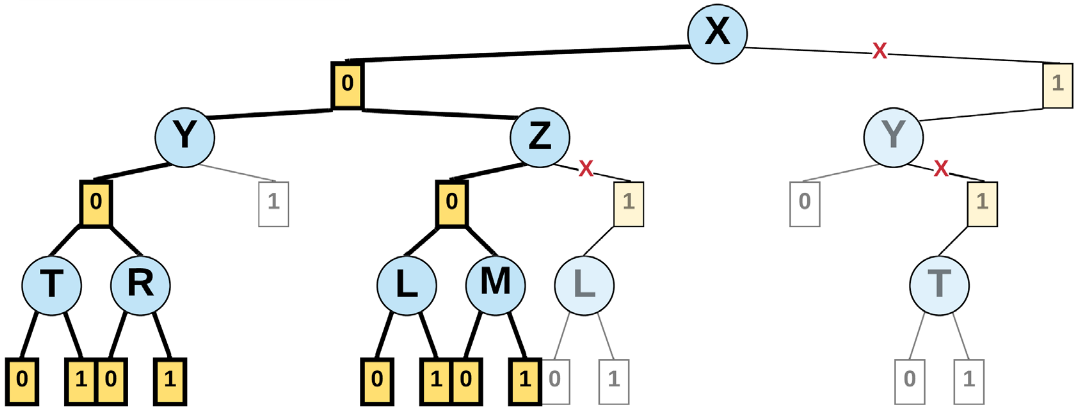
← **AOAS**

11 nodes; 16 solutions

Properties (Complexity)

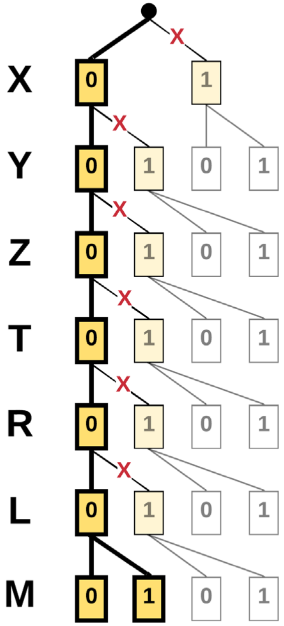


$\boxed{O(n \cdot m)}$ where n is the number of variables, and m is the number of abstract states per variable

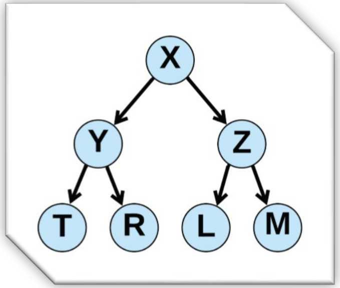


← **AOAS**
11 nodes; 16 solutions

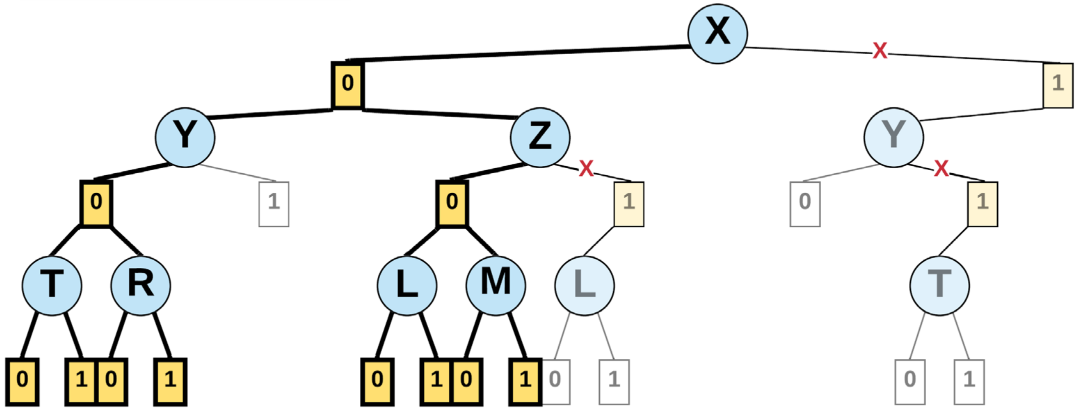
OR Abstraction Sampling →
8 nodes; 2 solutions



Properties (Complexity)



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

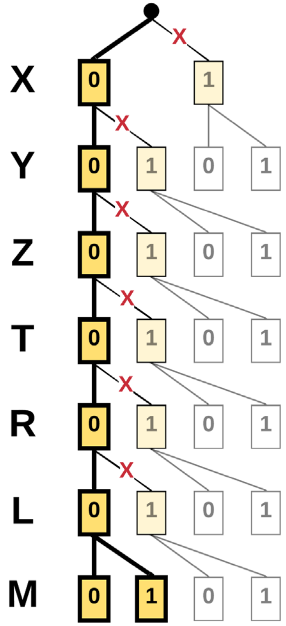


← **AOAS**

11 nodes; 16 solutions

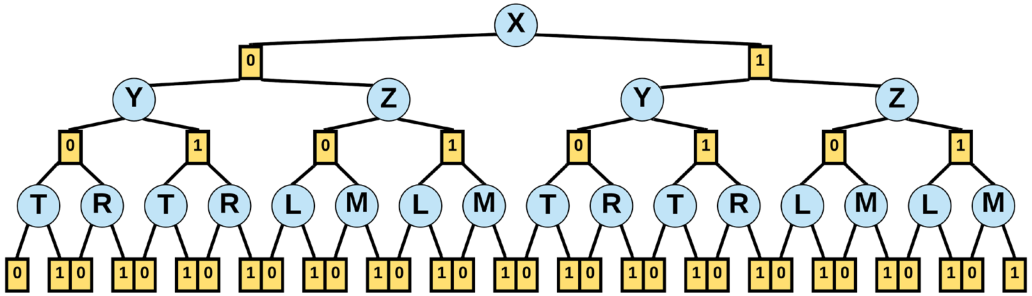
OR Abstraction Sampling →

8 nodes; 2 solutions



Previous AND/OR Scheme→

42 nodes; 128 solutions



Other Properties (see Thesis)

- Conditions for exact AOAS estimates
  - Proposal-based conditions
  - Abstraction-based conditions

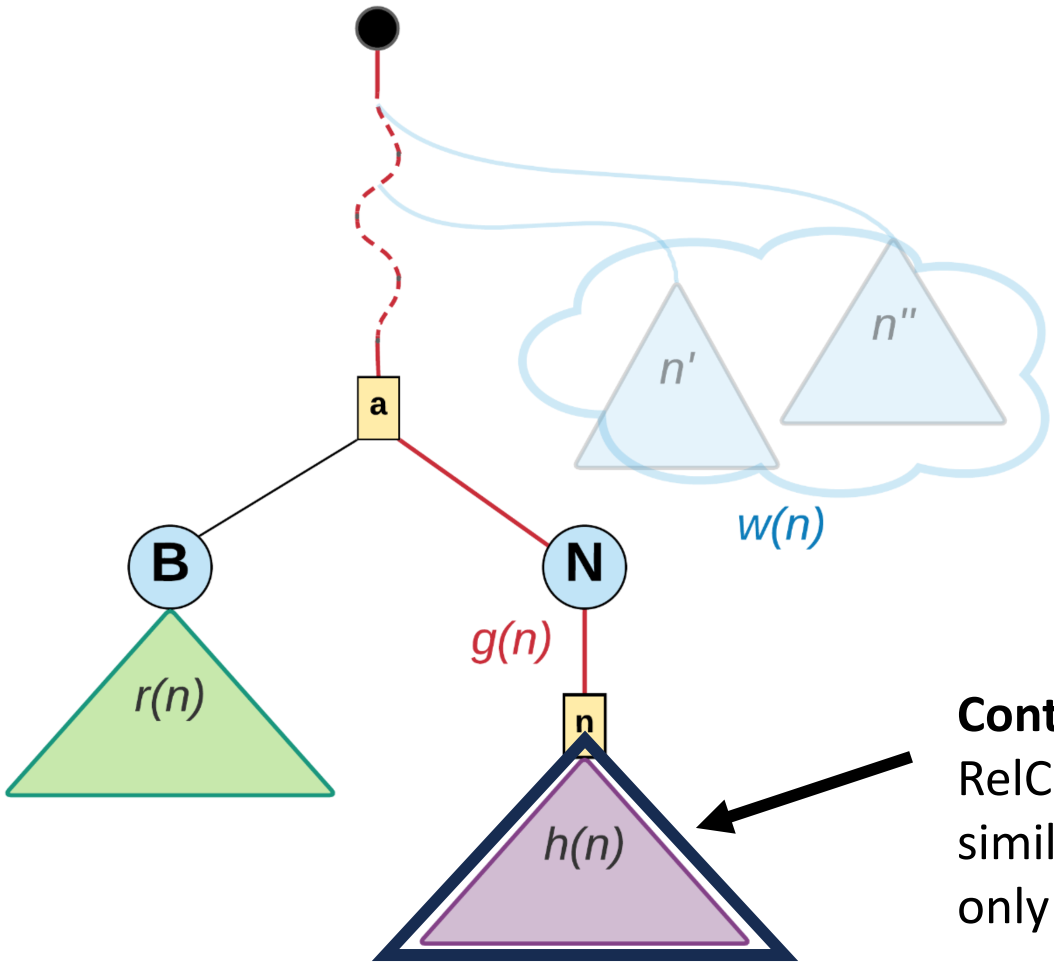
Original Work Set A Foundation…

Main Questions 2:

- ❑ How to construct powerful abstraction functions?

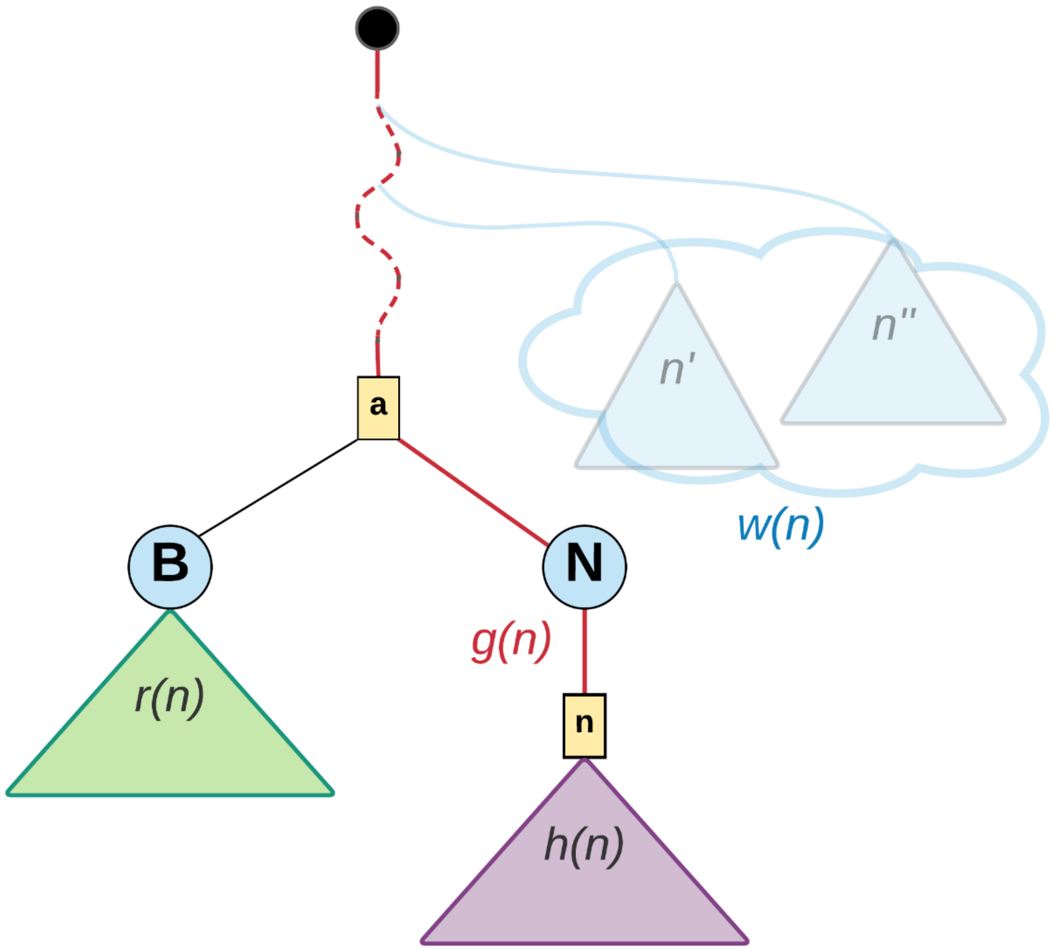
Abstraction Function Schemes

What did the previous abstraction schemes capture?



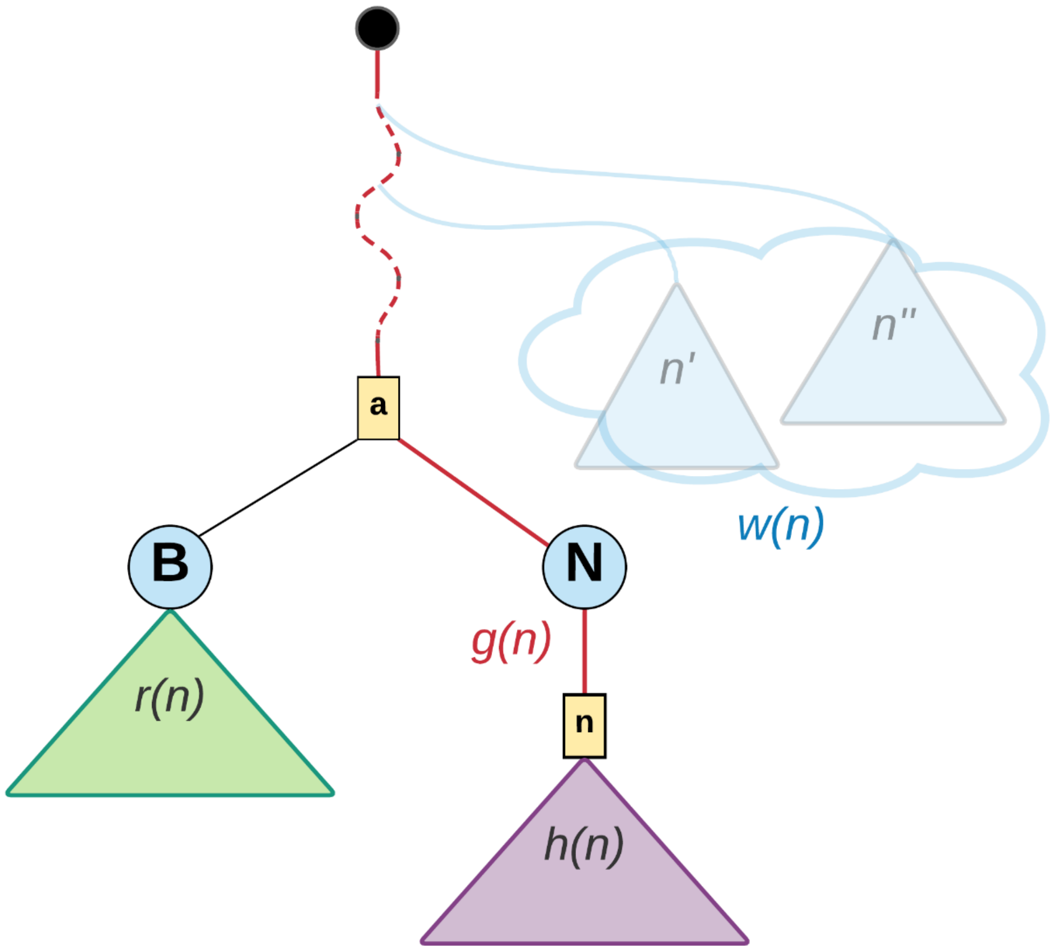
Context-Based Schemes:
RelCB and RandCB only estimate similarity of this piece and based only on graph structure

Value-Based Abstraction Functions - Intuition



- Use relevant quantities to assign a values to nodes
- Use those values to guide abstractions

Value-Based Abstraction Functions - Classes



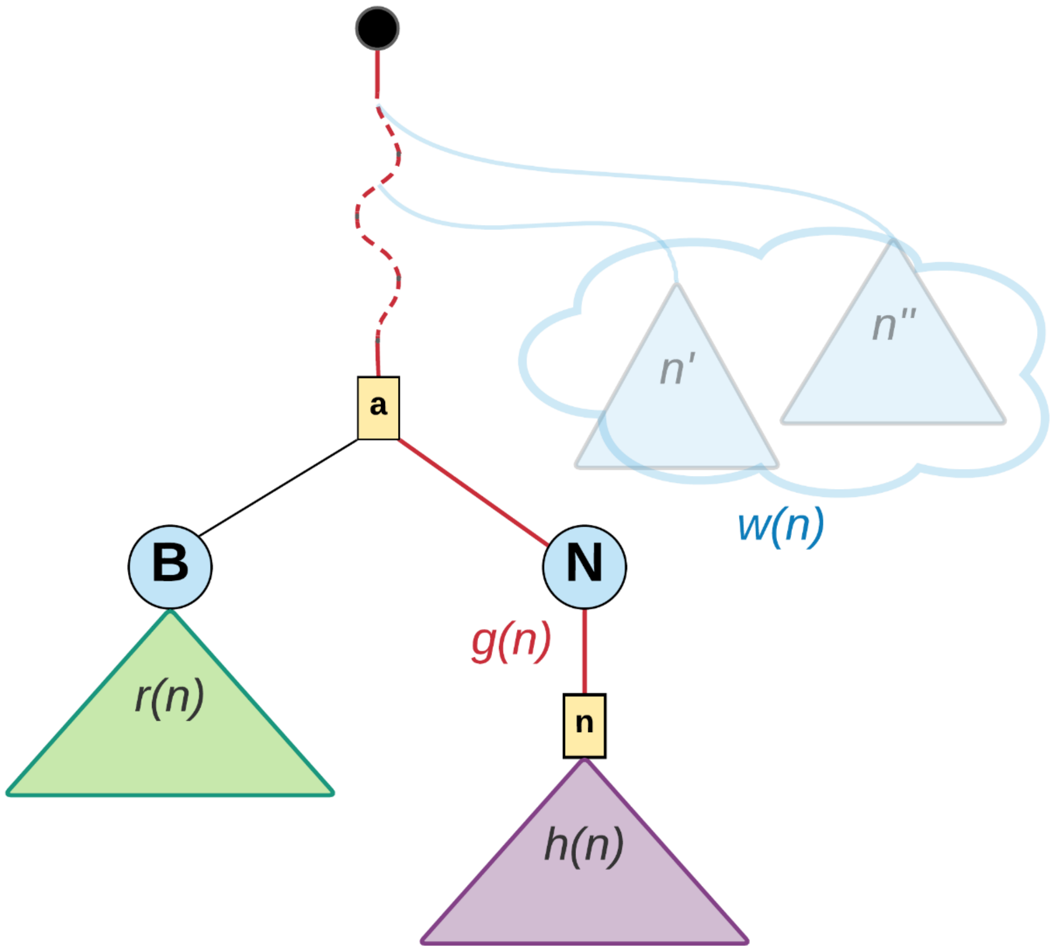
Potential Candidates:

HB: $\mu(n) = h(n)$

HRB: $\mu(n) = h(n)\ r(n)$

QB: $\mu(n) = w(n)\ g(n)\ h(n)\ r(n)$

Value-Based Abstraction Functions - Classes



Potential Candidates:

HB: $\mu(n) = h(n)$

HRB: $\mu(n) = h(n)\ r(n)$

QB: $\mu(n) = w(n)\ g(n)\ h(n)\ r(n)$

(best performing)

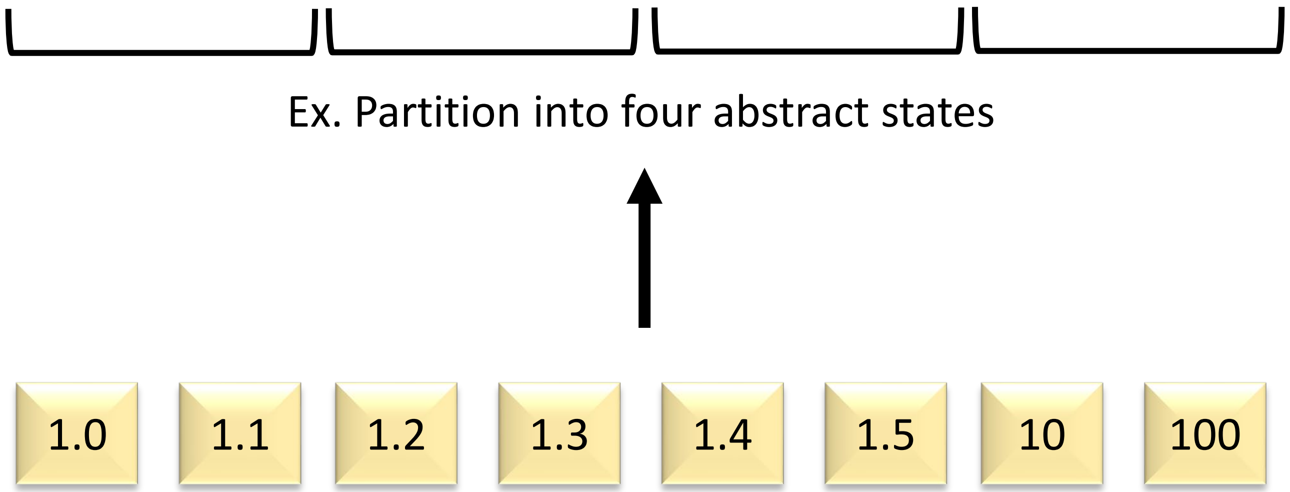
Value-Based Abstraction Functions - Partitioning Intuition

- Simple and fast
- Group similar nodes together
  - Minimize with-in variance of abstract states
- Form abstract states of roughly equal mass
  - *[Rizzo, 2007]*

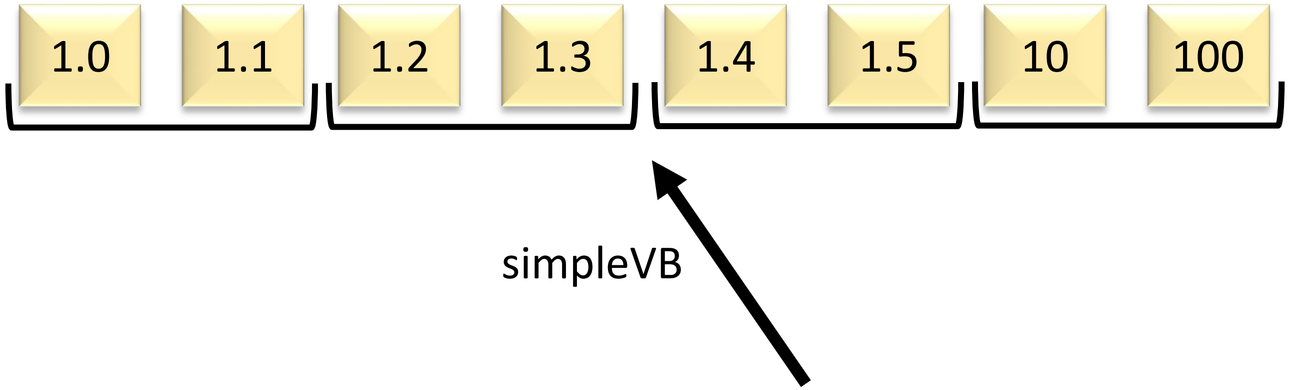
Value-Based Abstraction Functions - Partitioning Schemes

1.0	1.1	1.2	1.3	1.4	1.5	10	100

Value-Based Abstraction Functions - Partitioning Schemes

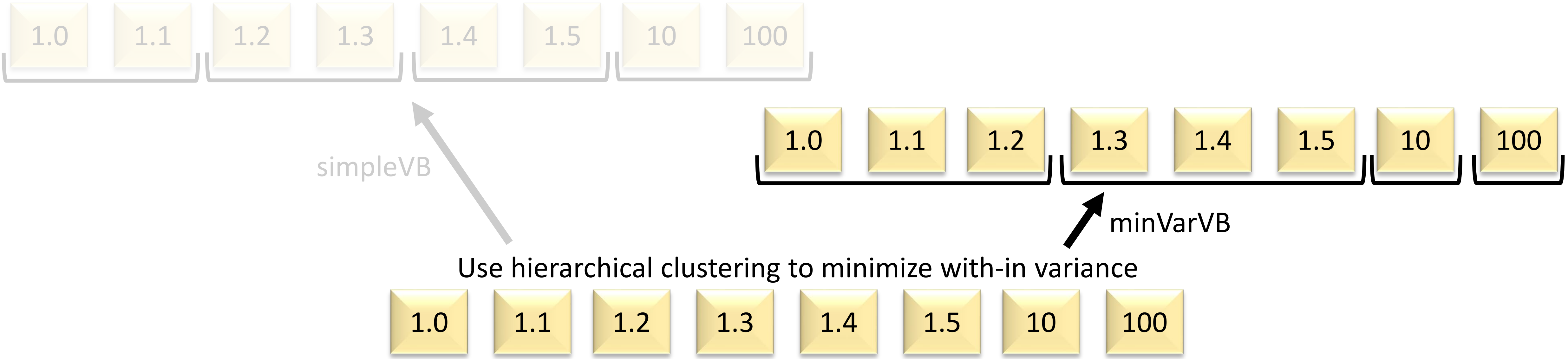


Value-Based Abstraction Functions - Partitioning Schemes

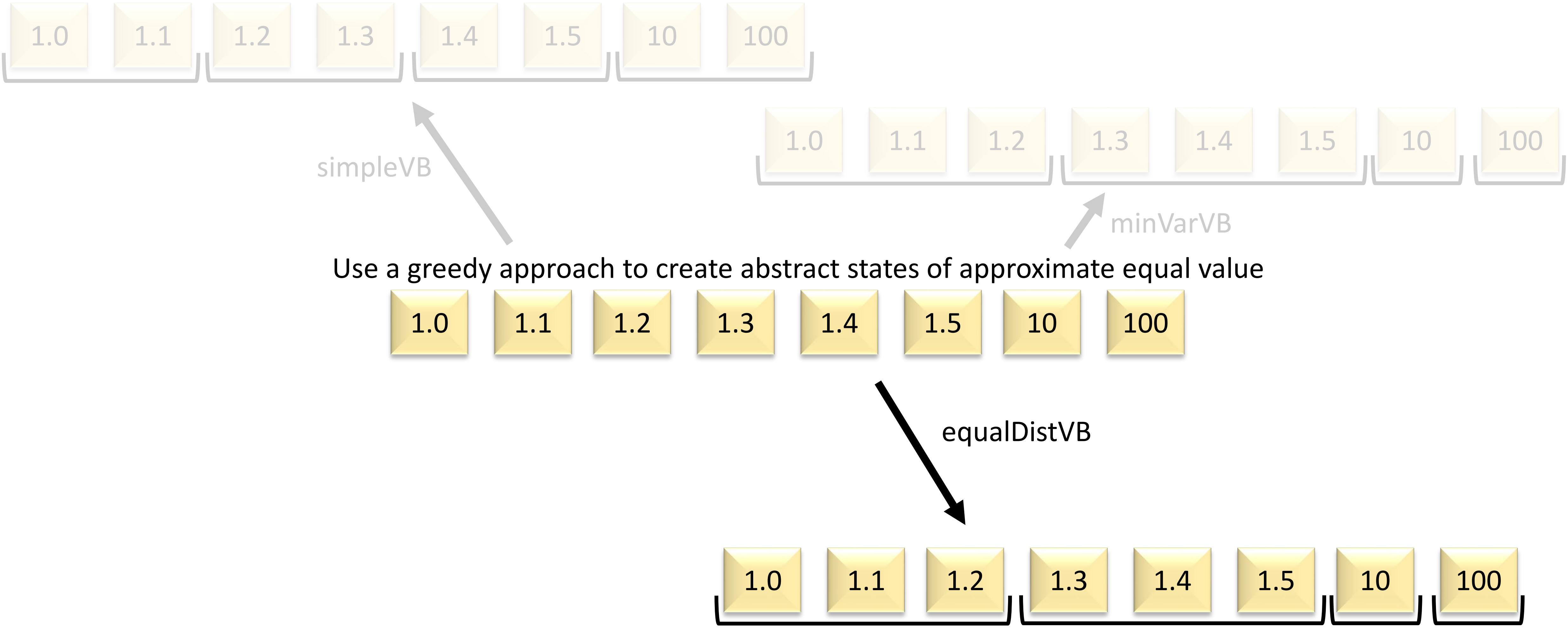


Simply partition into equal cardinality abstract states!

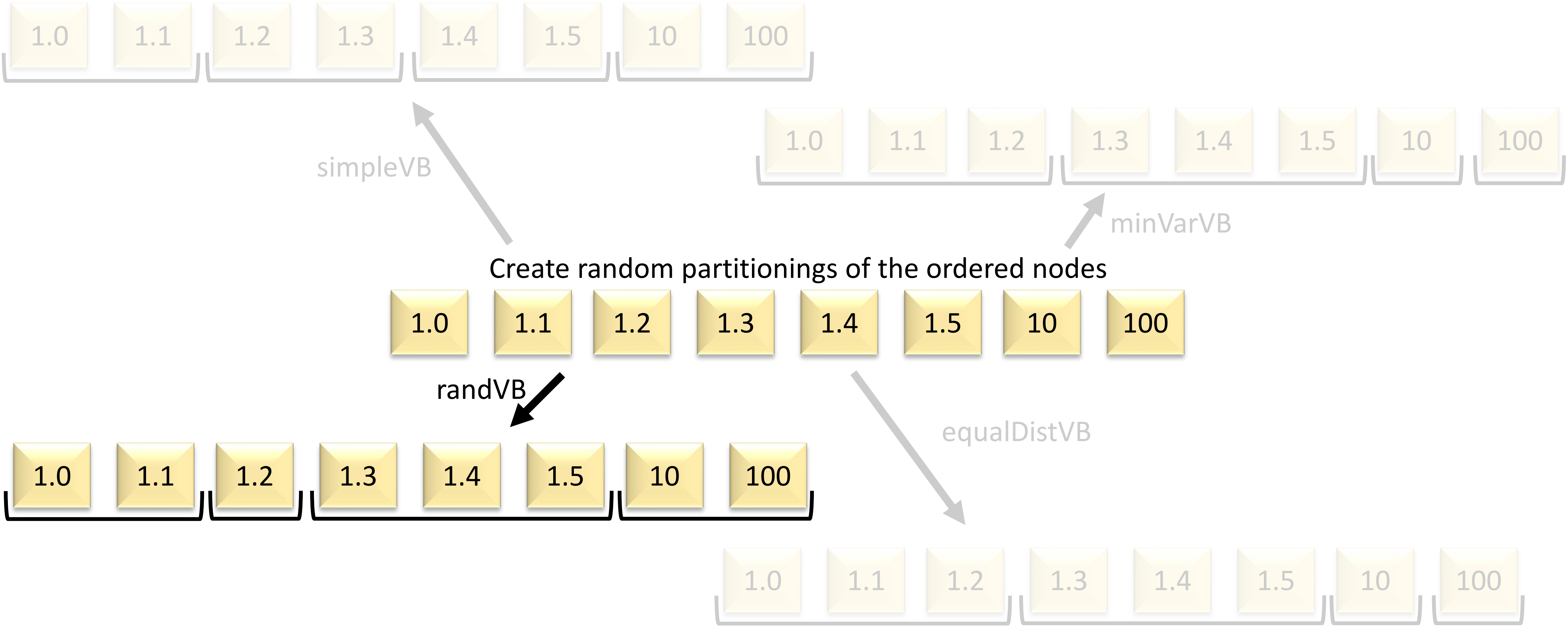
Value-Based Abstraction Functions - Partitioning Schemes



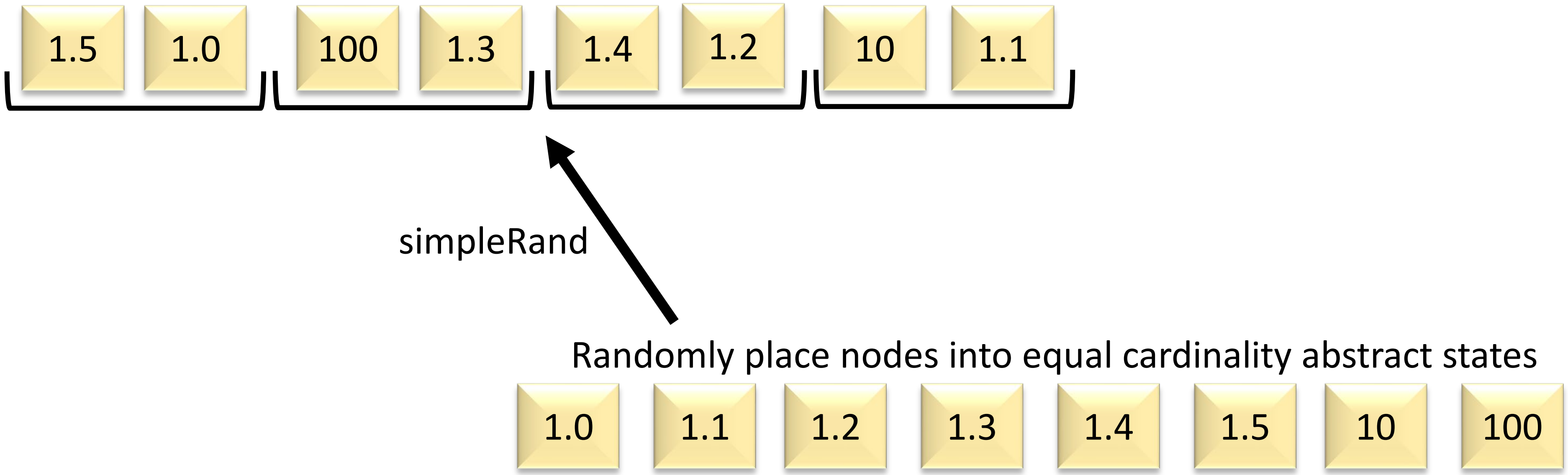
Value-Based Abstraction Functions - Partitioning Schemes



Value-Based Abstraction Functions - Partitioning Schemes



Purely Random Abstractions



Empirical Evaluation

- ❑ Performance of Abstraction Sampling comparing against existing schemes?
- ❑ Does sampling over the AND/OR space provide benefits?
- ❑ What abstraction functions empower Abstraction Sampling most?

Setup

- Problems (480+)
  - DBN, Grids, Linkage-Type4, Pedigree, Promedas
- Abstraction Sampling Algorithms
  - Sampling Schemes:
    - ORAS, proper-restricted-AOAS (pAOAS), **AOAS**
  - Abstraction Functions:
    - Context-Based, Value-Based, and Purely Random abstractions
    - Varying granularities
  - Heuristic:
    - Weighted Mini-Bucket Elimination (wMBE)
- Competing Algorithms
  - IS, DIS [WMB-IS, IJGP-SS]
- Questions
  - Quality of estimates, Scalability of Abstraction Functions

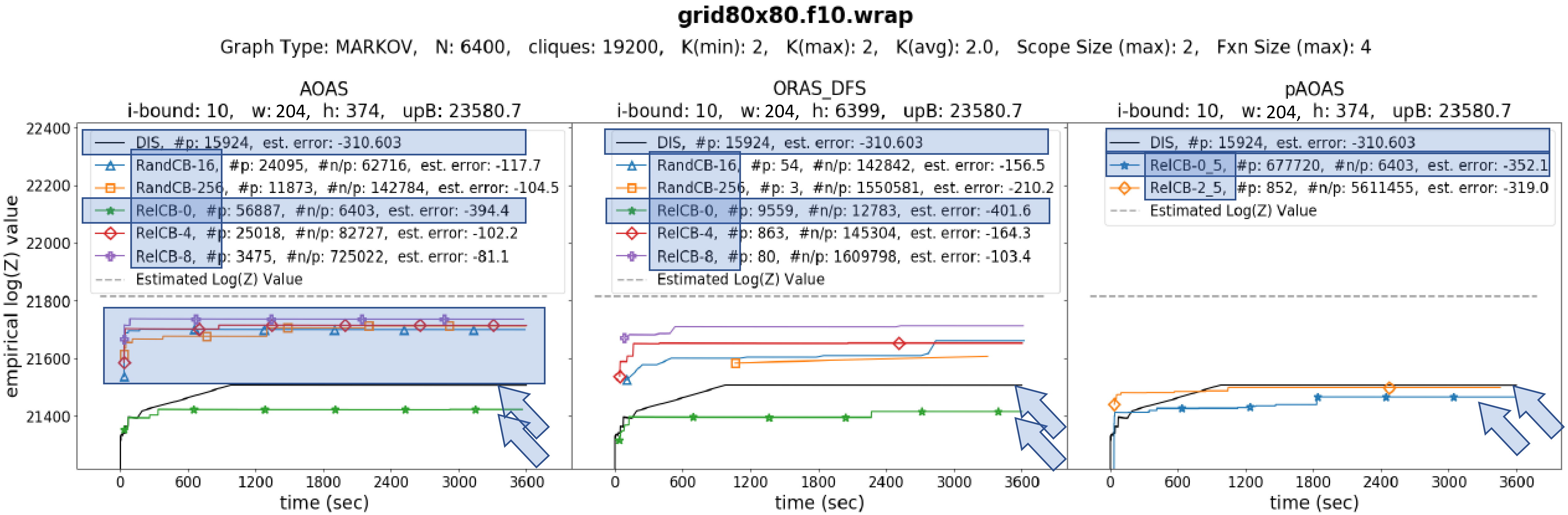
[Liu, Ihler, 2012]

[Liu, Fisher III, Ihler, 2015]

[Gogate and Dechter, 2011]

[Lou, Dechter, Ihler, 2019]

Plots



- - - : reference $log_{10}Z$ value

#p: number of probes

#n/p: number of nodes per probe

est. error: $log_{10}Z$ error w.r.t. the reference value

Aggregation Tables

Bmk: benchmark name

Sz: difficulty of subset of problems {small, LARGE}

Graph Scheme: Abstraction Sampling search scheme

Abs: granularity of abstraction function

n*: number of problems solved

log(err): average $\log_{10} Z$ error

error distr.: count of problems solved within an error threshold

#probes: average number of probes

#nodes/probe: average number of nodes per probe

[Lou et al., 2019]

i-Bound = 10					DIS Unable to Solve	(3600 sec)					
Bmk	Sz	Graph Scheme	Context Scheme	Abs	n*	log(err)	error distr. 0.5	error distr. 2	error distr. 10	#probes	#nodes/probe
Linkage-Type4	(LARGE, n:10822, d:1.8, w:51, h:581/5745)	AOAS	RelCB	0	7	-32.487	0	0	0	5.52E+05	829
				4	26	-14.174	1	1	10	1.92E+05	16472
				8	**41**	**-11.090**	**3**	**10**	**26**	**1.20E+04**	**463468**
			RandCB	16	26	-18.582	0	0	9	2.50E+05	10532
				256	27	-18.221	0	1	9	1.74E+05	18523
		pAOAS	RelCB (k=5)	0	8	-27.742	0	0	0	1.14E+07	759
				2	13	-27.201	0	0	1	6.96E+04	280176
				4	11	-25.427	0	0	0	8.90E+02	34177733
			RandCB	16	12	-24.204	0	0	1	4.24E+03	8820722
				256	14	-30.930	0	0	0	2.58E+03	24532116
		ORAS -DFS	RelCB	0	7	-29.820	0	0	0	1.42E+06	2520
				4	15	-20.802	0	0	2	5.14E+04	26863
				8	16	-18.971	0	0	5	2.44E+03	355404
			RandCB	16	13	-22.843	0	0	1	1.05E+04	23723
				256	12	-27.151	0	0	0	1.18E+02	288516

Comparison of Abstraction Functions

iB-10, t-1200sec, LARGE		DBN			Grids			Linkage-Type4			Promedas		
Class	Scheme	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error
QB	simple	1	0	6.540	16	0	197.931	2048	13	48.681	4	34	11.919
	minVar	2048	0	1.837	1024	0	28.423	256	31	93.058	16	13	5.403
	equalDist	512	0	5.423	2048	0	118.547	2048	22	46.196	512	15	5.960
	equalDist2	2048	0	3.813	2048	0	91.994	1024	21	40.310	2048	12	4.982
	equalDist3	**2048**	**0**	**1.645**	**2048**	**0**	**19.277**	**1024**	**20**	**37.490**	**256**	**5**	**2.560**
	equalDist4	**2048**	**0**	**1.643**	**2048**	**0**	**18.866**	**2048**	**16**	**30.512**	**512**	**5**	**2.476**
	rand	4	0	6.292	16	0	163.973	256	17	156.992	4	28	11.532
CTX	rand	64	0	5.710	512	0	111.104	2048	53	194.741	256	0	3.222
	rel	1	0	6.267	1024	0	80.633	1024	37	129.189	16	34	11.247
RAND	**rand**	**2048**	**0**	**2.123**	**2048**	**0**	**19.053**	**1024**	**19**	**33.804**	**1024**	**10**	**3.936**

Comparison of Abstraction Functions

iB-10, t-1200sec, LARGE		DBN			Grids			Linkage-Type4			Promedas		
Class	**Scheme**	**nAbs**	**Fail**	**Avg. Error**	**nAbs**	**Fail**	**Avg. Error**	**nAbs**	**Fail**	**Avg. Error**	**nAbs**	**Fail**	**Avg. Error**
QB	simple	1	0	6.540	16	0	197.931	2048	13	48.681	4	34	11.919
	minVar	2048	0	1.837	1024	0	28.423	256	31	93.058	16	13	5.403
	equalDist	512	0	5.423	2048	0	118.547	2048	22	46.196	512	15	5.960
	equalDist2	2048	0	3.813	2048	0	91.994	1024	21	40.310	2048	12	4.982
	equalDist3	**2048**	**0**	**1.645**	**2048**	**0**	**19.277**	**1024**	**20**	**37.490**	**256**	**5**	**2.560**
	equalDist4	**2048**	**0**	**1.643**	**2048**	**0**	**18.866**	**2048**	**16**	**30.512**	**512**	**5**	**2.476**
	rand	4	0	6.292	16	0	163.973	256	17	156.992	4	28	11.532
CTX	rand	64	0	5.710	512	0	111.104	2048	53	194.741	256	0	3.222
	rel	1	0	6.267	1024	0	80.633	1024	37	129.189	16	34	11.247
RAND	**rand**	**2048**	**0**	**2.123**	**2048**	**0**	**19.053**	**1024**	**19**	**33.804**	**1024**	**10**	**3.936**

Comparison of Abstraction Functions

iB-10, t-1200sec, LARGE		DBN			Grids			Linkage-Type4			Promedas		
Class	Scheme	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error
QB	simple	1	0	6.540	16	0	197.931	2048	13	48.681	4	34	11.919
	minVar	2048	0	1.837	1024	0	28.423	256	31	93.058	16	13	5.403
	equalDist	512	0	5.423	2048	0	118.547	2048	22	46.196	512	15	5.960
	equalDist2	2048	0	3.813	2048	0	91.994	1024	21	40.310	2048	12	4.982
	equalDist3	**2048**	**0**	**1.645**	**2048**	**0**	**19.277**	**1024**	**20**	**37.490**	**256**	**5**	**2.560**
	equalDist4	**2048**	**0**	**1.643**	**2048**	**0**	**18.866**	**2048**	**16**	**30.512**	**512**	**5**	**2.476**
	rand	4	0	6.292	16	0	163.973	256	17	156.992	4	28	11.532
CTX	rand	64	0	5.710	512	0	111.104	2048	53	194.741	256	0	3.222
	rel	1	0	6.267	1024	0	80.633	1024	37	129.189	16	34	11.247
RAND	**rand**	**2048**	**0**	**2.123**	**2048**	**0**	**19.053**	**1024**	**19**	**33.804**	**1024**	**10**	**3.936**

Comparison of Abstraction Functions

iB-10, t-1200sec, LARGE		DBN			Grids			Linkage-Type4			Promedas		
Class	Scheme	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error
QB	simple	1	0	6.540	16	0	197.931	2048	13	48.681	4	34	11.919
	minVar	2048	0	1.837	1024	0	28.423	256	31	93.058	16	13	5.403
	equalDist	512	0	5.423	2048	0	118.547	2048	22	46.196	512	15	5.960
	equalDist2	2048	0	3.813	2048	0	91.994	1024	21	40.310	2048	12	4.982
	equalDist3	**2048**	**0**	**1.645**	**2048**	**0**	**19.277**	**1024**	**20**	**37.490**	**256**	**5**	**2.560**
	equalDist4	**2048**	**0**	**1.643**	**2048**	**0**	**18.866**	**2048**	**16**	**30.512**	**512**	**5**	**2.476**
	rand	4	0	6.292	16	0	163.973	256	17	156.992	4	28	11.532
CTX	rand	64	0	5.710	512	0	111.104	2048	53	194.741	256	0	3.222
	rel	1	0	6.267	1024	0	80.633	1024	37	129.189	16	34	11.247
RAND	**rand**	**2048**	**0**	**2.123**	**2048**	**0**	**19.053**	**1024**	**19**	**33.804**	**1024**	**10**	**3.936**

Comparison of Abstraction Functions

iB-10, t-1200sec, LARGE		DBN			Grids			Linkage-Type4			Promedas		
Class	Scheme	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error
QB	simple	1	0	6.540	16	0	197.931	2048	13	48.681	4	34	11.919
	minVar	2048	0	1.837	1024	0	28.423	256	31	93.058	16	13	5.403
	equalDist	512	0	5.423	2048	0	118.547	2048	22	46.196	512	15	5.960
	equalDist2	2048	0	3.813	2048	0	91.994	1024	21	40.310	2048	12	4.982
	equalDist3	**2048**	**0**	**1.645**	**2048**	**0**	**19.277**	**1024**	**20**	**37.490**	**256**	**5**	**2.560**
	equalDist4	**2048**	**0**	**1.643**	**2048**	**0**	**18.866**	**2048**	**16**	**30.512**	**512**	**5**	**2.476**
	rand	4	0	6.292	16	0	163.973	256	17	156.992	4	28	11.532
CTX	rand	64	0	5.710	512	0	111.104	2048	53	194.741	256	0	3.222
	rel	1	0	6.267	1024	0	80.633	1024	37	129.189	16	34	11.247
RAND	**rand**	**2048**	**0**	**2.123**	**2048**	**0**	**19.053**	**1024**	**19**	**33.804**	**1024**	**10**	**3.936**

Comparison of Abstraction Functions

iB-10, t-1200sec, LARGE		DBN			Grids			Linkage-Type4			Promedas		
Class	Scheme	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error
QB	simple	1	0	6.540	16	0	197.931	2048	13	48.681	4	34	11.919
	minVar	2048	0	1.837	1024	0	28.423	256	31	93.058	16	13	5.403
	equalDist	512	0	5.423	2048	0	118.547	2048	22	46.196	512	15	5.960
	equalDist2	2048	0	3.813	2048	0	91.994	1024	21	40.310	2048	12	4.982
	equalDist3	**2048**	**0**	**1.645**	**2048**	**0**	**19.277**	**1024**	**20**	**37.490**	**256**	**5**	**2.560**
	equalDist4	**2048**	**0**	**1.643**	**2048**	**0**	**18.866**	**2048**	**16**	**30.512**	**512**	**5**	**2.476**
	rand	4	0	6.292	16	0	163.973	256	17	156.992	4	28	11.532
CTX	rand	64	0	5.710	512	0	111.104	2048	53	194.741	256	0	3.222
	rel	1	0	6.267	1024	0	80.633	1024	37	129.189	16	34	11.247
RAND	**rand**	**2048**	**0**	**2.123**	**2048**	**0**	**19.053**	**1024**	**19**	**33.804**	**1024**	**10**	**3.936**

Comparison of Abstraction Functions

iB-10, t-1200sec, LARGE		DBN			Grids			Linkage-Type4			Promedas		
Class	Scheme	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error	nAbs	Fail	Avg. Error
QB	simple	1	0	6.540	16	0	197.931	2048	13	48.681	4	34	11.919
	minVar	2048	0	1.837	1024	0	28.423	256	31	93.058	16	13	5.403
	equalDist	512	0	5.423	2048	0	118.547	2048	22	46.196	512	15	5.960
	equalDist2	2048	0	3.813	2048	0	91.994	1024	21	40.310	2048	12	4.982
	equalDist3	**2048**	**0**	**1.645**	**2048**	**0**	**19.277**	**1024**	**20**	**37.490**	**256**	**5**	**2.560**
	equalDist4	**2048**	**0**	**1.643**	**2048**	**0**	**18.866**	**2048**	**16**	**30.512**	**512**	**5**	**2.476**
	rand	4	0	6.292	16	0	163.973	256	17	156.992	4	28	11.532
CTX	rand	64	0	5.710	512	0	111.104	2048	53	194.741	256	0	3.222
	rel	1	0	6.267	1024	0	80.633	1024	37	129.189	16	34	11.247
RAND	**rand**	**2048**	**0**	**2.123**	**2048**	**0**	**19.053**	**1024**	**19**	**33.804**	**1024**	**10**	**3.936**

Comparison of Abstraction Granularity

iB-5, t-300sec, Exact			DBN		Grids		Pedigree		Promedas	
Class	Scheme	nAbs	Fail	Avg. Error	Fail	Avg. Error	Fail	Avg. Error	Fail	Avg. Error
QB	minVar	4	0	1.684	0	3.622	0	1.434	2	2.518
		64	0	0.180	0	1.897	0	0.210	1	1.062
		1024	0	0.060	0	1.566	0	0.479	2	1.837
	equalDist3	4	0	1.594	0	5.861	0	1.668	1	1.804
		64	0	0.236	0	2.570	0	0.221	0	0.570
		1024	0	0.051	0	1.844	0	0.155	0	0.462
	equalDist4	4	0	1.371	0	5.988	0	1.648	1	1.678
		64	0	0.215	0	2.438	0	0.231	0	0.596
		1024	0	0.150	0	1.891	0	0.150	0	0.455
CTX	rand	4	0	1.381	0	5.030	0	1.852	7	4.643
		64	0	1.763	0	5.950	0	0.598	1	2.659
		1024	0	2.007	0	5.513	0	1.114	1	2.442
	rel	4	0	1.850	0	5.933	0	1.332	10	5.729
		64	0	3.510	0	4.021	0	0.424	6	4.349
		1024	0	5.086	0	5.136	0	1.041	15	6.688
RAND	rand	4	0	1.018	0	4.329	0	1.705	2	2.947
		64	0	0.418	0	2.094	0	0.212	0	0.757
		1024	0	0.120	0	1.501	0	0.143	0	0.513

Comparison of Abstraction Granularity

iB-5, t-300sec, Exact			DBN		Grids		Pedigree		Promedas	
Class	Scheme	nAbs	Fail	Avg. Error	Fail	Avg. Error	Fail	Avg. Error	Fail	Avg. Error
QB	minVar	4	0	1.684	0	3.622	0	1.434	2	2.518
		64	0	0.180	0	1.897	0	0.210	1	1.062
		1024	0	0.060	0	1.566	0	0.479	2	1.837
	equalDist3	4	0	1.594	0	5.861	0	1.668	1	1.804
		64	0	0.236	0	2.570	0	0.221	0	0.570
		1024	0	0.051	0	1.844	0	0.155	0	0.462
	equalDist4	4	0	1.371	0	5.988	0	1.648	1	1.678
		64	0	0.215	0	2.438	0	0.231	0	0.596
		1024	0	0.150	0	1.891	0	0.150	0	0.455
CTX	rand	4	0	1.381	0	5.030	0	1.852	7	4.643
		64	0	1.763	0	5.950	0	0.598	1	2.659
		1024	0	2.007	0	5.513	0	1.114	1	2.442
	rel	4	0	1.850	0	5.933	0	1.332	10	5.729
		64	0	3.510	0	4.021	0	0.424	6	4.349
		1024	0	5.086	0	5.136	0	1.041	15	6.688
RAND	rand	4	0	1.018	0	4.329	0	1.705	2	2.947
		64	0	0.418	0	2.094	0	0.212	0	0.757
		1024	0	0.120	0	1.501	0	0.143	0	0.513

Comparison of Abstraction Granularity

iB-5, t-300sec, Exact			DBN		Grids		Pedigree		Promedas	
Class	Scheme	nAbs	Fail	Avg. Error	Fail	Avg. Error	Fail	Avg. Error	Fail	Avg. Error
QB	minVar	4	0	1.684	0	3.622	0	1.434	2	2.518
		64	0	0.180	0	1.897	0	0.210	1	1.062
		1024	0	0.060	0	1.566	0	0.479	2	1.837
	equalDist3	4	0	1.594	0	5.861	0	1.668	1	1.804
		64	0	0.236	0	2.570	0	0.221	0	0.570
		1024	0	0.051	0	1.844	0	0.155	0	0.462
	equalDist4	4	0	1.371	0	5.988	0	1.648	1	1.678
		64	0	0.215	0	2.438	0	0.231	0	0.596
		1024	0	0.150	0	1.891	0	0.150	0	0.455
CTX	rand	4	0	1.381	0	5.030	0	1.852	7	4.643
		64	0	1.763	0	5.950	0	0.598	1	2.659
		1024	0	2.007	0	5.513	0	1.114	1	2.442
	rel	4	0	1.850	0	5.933	0	1.332	10	5.729
		64	0	3.510	0	4.021	0	0.424	6	4.349
		1024	0	5.086	0	5.136	0	1.041	15	6.688
RAND	rand	4	0	1.018	0	4.329	0	1.705	2	2.947
		64	0	0.418	0	2.094	0	0.212	0	0.757
		1024	0	0.120	0	1.501	0	0.143	0	0.513

AS Comparison Chart

Algorithm	Compact Search Space	Scalable Abstractions
OR Abstraction Sampling	No	Yes
“Proper” AO Abstraction Sampling	Yes	No
AOAS	Yes	Yes

Value-Based Abstraction Functions - Best Scheme

Score > 1.0 ⇒ better than context-based

	HB	HRB	QB
simple	2.75	1.12	0.72
minVar	1.05	1.13	2.95
equalDist	0.75	0.59	1.16
equalDist2	0.84	0.75	1.82
equalDist3	1.20	1.01	4.05
equalDist4	0.87	1.14	3.90
rand	2.41	0.93	0.60

Many of the new schemes performed better than the context based schemes.

equalDistQB3 and equalDistQB4 were best performing!

Conclusion

AND/OR Abstraction Sampling via AOAS is an efficient effective stratified sampling method for solving summation tasks and can be empowered by use of several of the newly proposed abstraction functions.

End Part 1

K*-Based Computational Protein Design using AND/OR Search

TPM 2022 Best Paper Award

Bobak Pezeshki, Radu Marinescu, Alex Ihler, and Rina Dechter. “AND/OR Branch-and-Bound for Computational Protein Design Optimizing K*”. *Proceedings of the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022).*

Bobak Pezeshki, Radu Marinescu, Alex Ihler, and Rina Dechter. “Boosting AND/OR-Based Computational Protein Design: Dynamic Heuristics and Generalizable UFO”. *Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).*

BOBAK PEZESHKI,

RADU MARINESCU

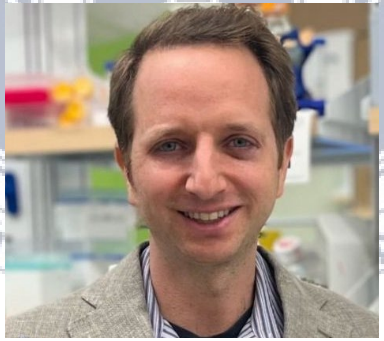
ALEX IHLER,

RINA DECHTER

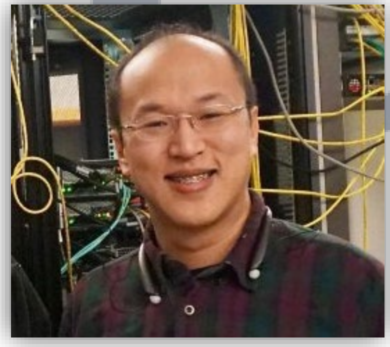
Special thanks to…

THOMAS SCHIEX

Special thanks to…



NATE GUERIN



JONATHAN JOU

GRAHAM HOLT

BRUCE DONALD

Outline

- ❑ Background: Computational Protein Design (CPD)
- ❑ K*MAP using AND/OR Search
  - ❑ Problem Formulation
  - ❑ AOBB-K* (using wMBE-K*)
  - ❑ Scalability Improvements
- ❑ Empirical Evaluation
- ❑ Conclusion and Future Work

Contributions

- ❑ Background: Computational Protein Design (CPD)
- ❑ K*MAP using AND/OR Search
  - ❑ Problem Formulation
  - ❑ AOBB-K* (using wMBE-K*)
  - ❑ Scalability Improvements
- ❑ Empirical Evaluation
- ❑ Conclusion and Future Work

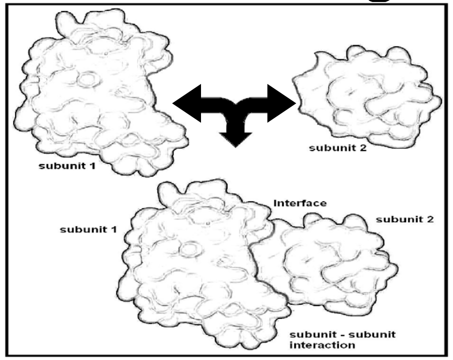
Background

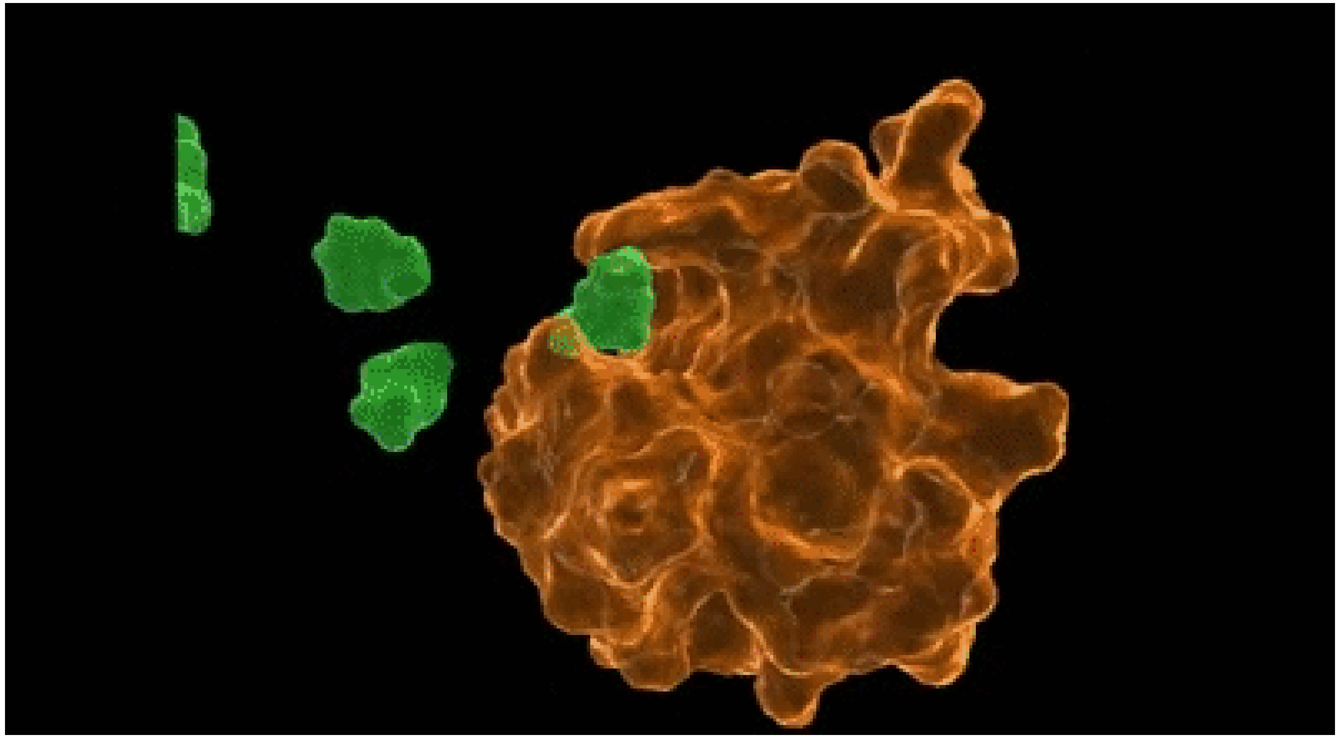
Computational Protein Design (CPD)

[Re]design proteins to perform desired biological functions.

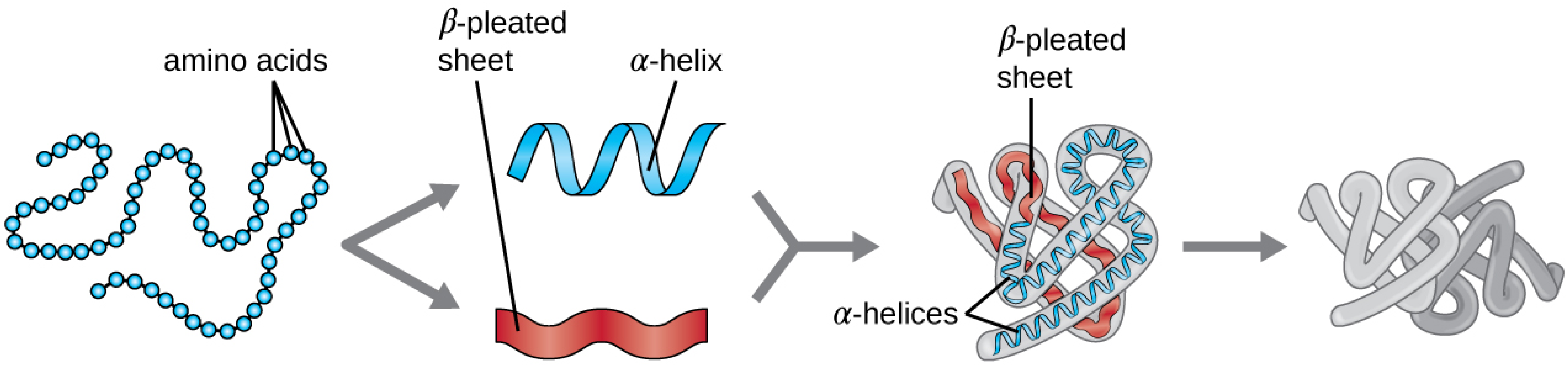
CPD often manifests as an optimization problem:

Ex. find the optimal composition that maximizes binding between subunits.





Computational Protein Design



Primary Protein Structure
Sequence of a chain of amino acids

Secondary Protein Structure
Local folding of the polypeptide chain into helices or sheets

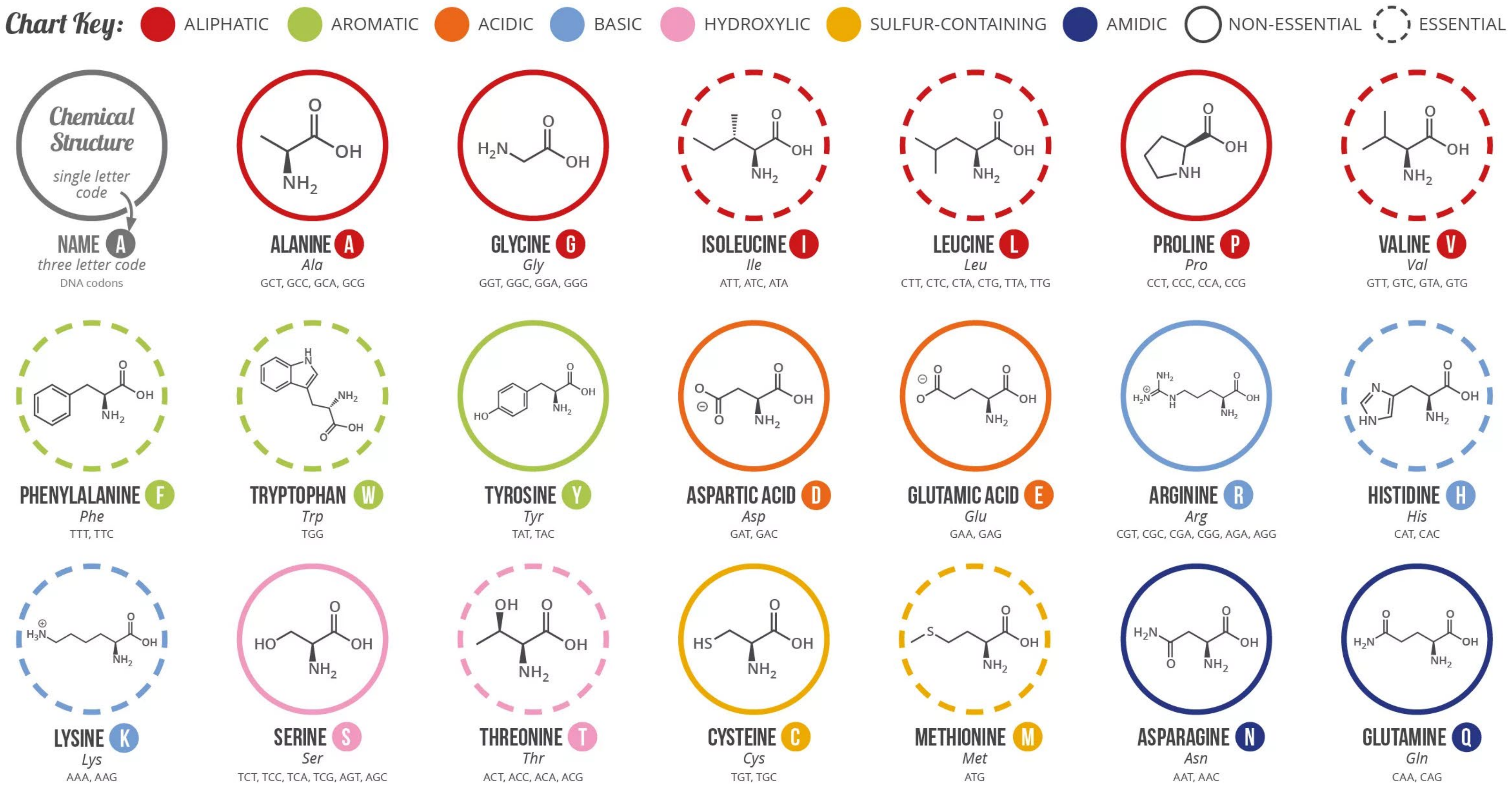
Tertiary Protein Structure
three-dimensional folding pattern of a protein due to side chain interactions

Quaternary Protein Structure
protein consisting of more than one amino acid chain

Computational Protein Design

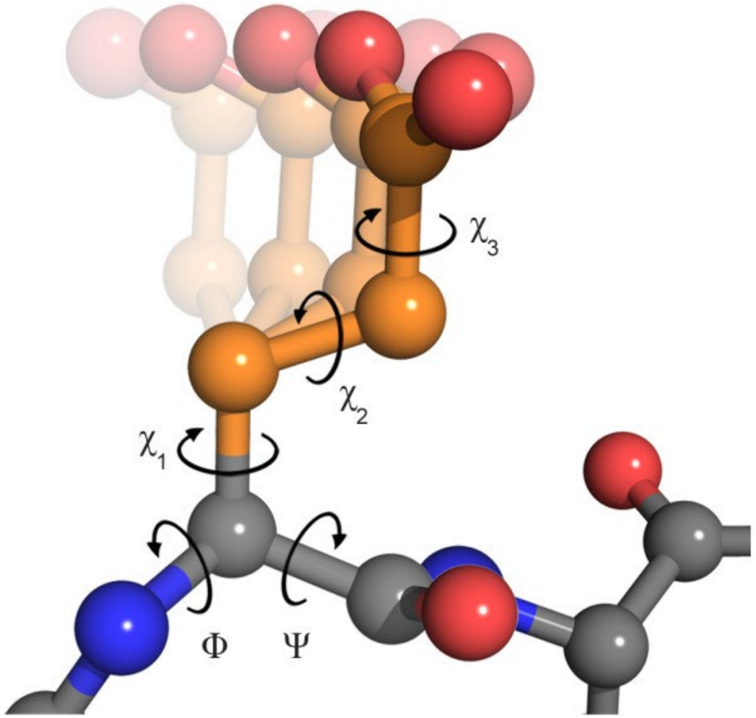
Cannot be made
by the human body

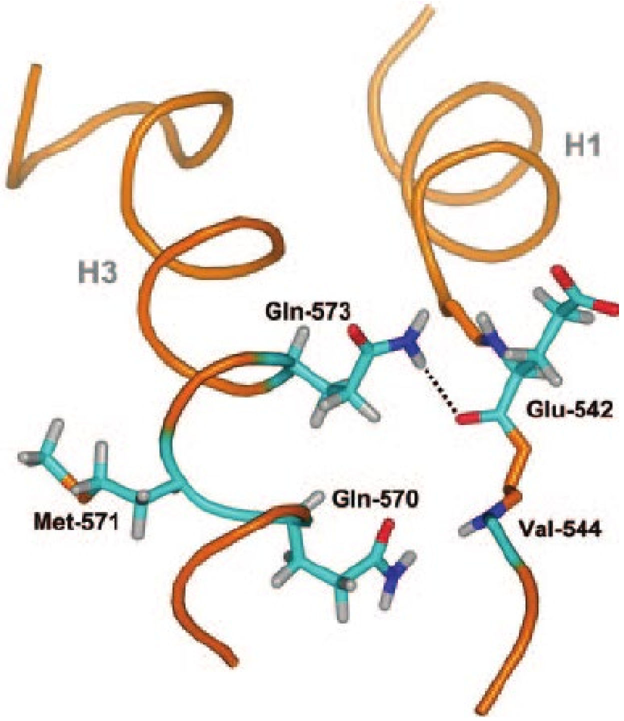
There Are ~20 Naturally Occurring Amino Acids



Computational Protein Design

Amino Acid Rotamers: Select conformational isomers of an amino acid



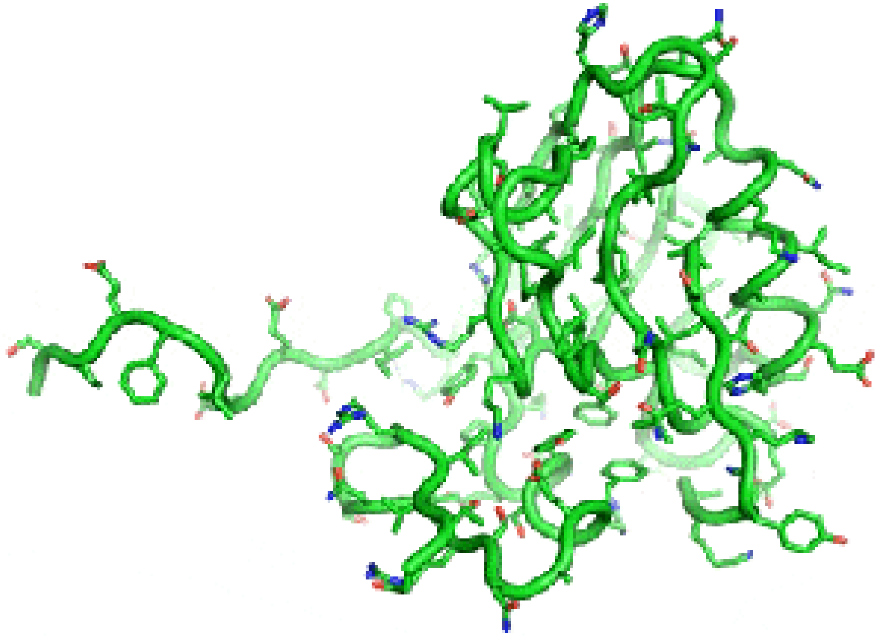


Peter Carlsson, Konrad F. Koehler, and Lennart Nilsson
Molecular Endocrinology 19(8):1960–1977. https://doi.org/10.1210/me.2004-0203

Proteins are Dynamic Structures

A protein’s structural state is probabilistic

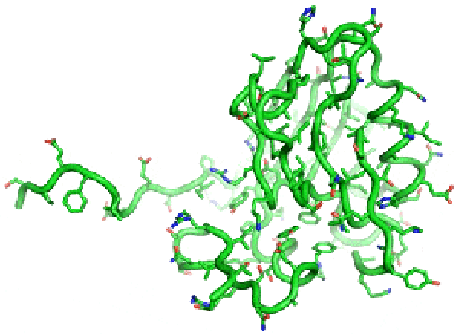
Proteins continuously transition between various energetically favorable conformation.



Partition Function

Partition Function (Z) Normalizes the Likelihood of the Protein In A Particular Conformational State

$$Z(r) = \sum_{c \in C(r)} \exp\{-E(c)/RT\}$$

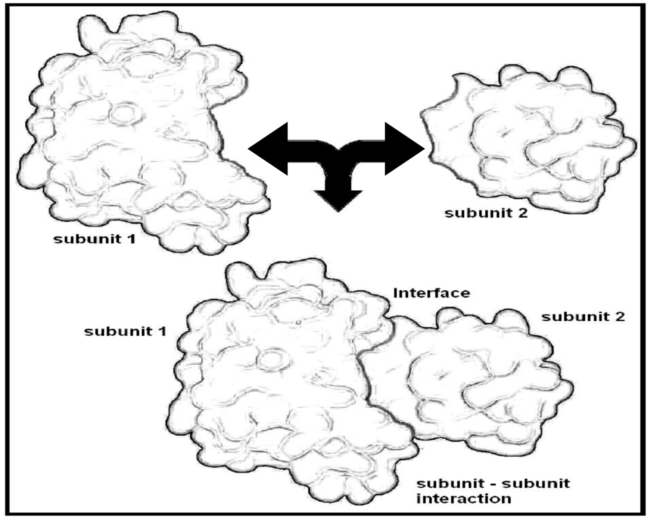


r = amino acid assignments to the residues
C(r) = possible rotamer conformations given a.a. sequence r
E(c) = energy given conformation c
R = universal gas constant (for unit conversion between kJ and K)
T = absolute temperature (Kelvin)

K* Objective *[Ojewole et al., 2018, Hill, 1987, Mc-Quarrie, 2000]*

K* approximate Ka, the affinity equilibrium constant

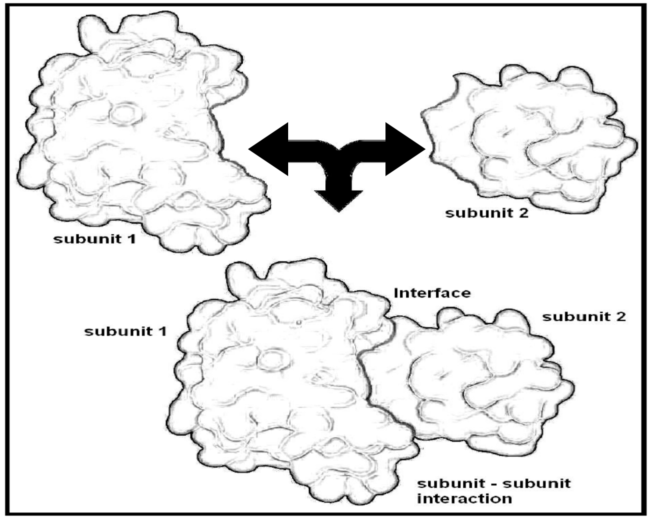
$$K^*(r) = \frac{Z_{complex}(r)}{Z_{subunit\ 1}(r)\ Z_{subunit\ 2}(r)}$$



Note that K* not only considers the “goodness” of the bonded state (PL),
but also weighs it relative to the “goodness” of the unbound (dissociate) states

K* Objective

$$K^*MAP = \max_{R} K^*(r)$$



ie. Find the sequence with the greatest K* ~ Ka

BBK* *[Ojewole, Jou, Fowler, Donald, 2018]*

- A*-like algorithm for designing proteins to improve binding
- Our objective: solve the same problem with algorithms that offer something more
  - New heuristic
  - Capture independences
  - Sampling

Task Difficulty

$$K^*(r) = \frac{Z_{complex}(r)}{Z_{subunit\ 1}(r)\, Z_{subunit\ 2}(r)}$$

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

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

$$K^*\,MAP = \max_{R} K^*(r)$$

Marginal MAP (MMAP)

- State-of-the-art search and sampling algorithms

<u>State-of-the-art Marginal MAP (MMAP) algorithms</u> [Marinescu, Lee, Dechter, Ihler, 2018]

Learning Depth-First AND/OR Search [Marinescu, Dechter, Ihler, 2018]

Stochastic Best-First AND/OR Search [Marinescu, Dechter, Ihler, 2018]

Recursive Best-First AND/OR Search [Marinescu, Dechter, Ihler, Kishimoto, Botea, 2018]

<u>State-of-the-art sampling algorithms</u>

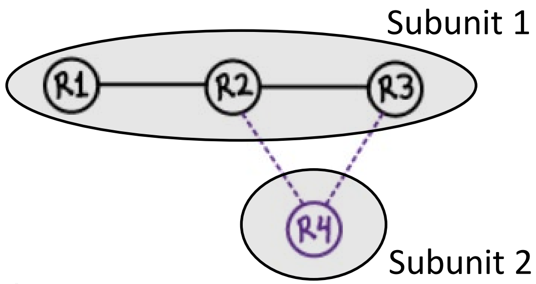
Dynamic Importance Sampling [Liu, Dechter, Ihler, 2017]

Abstraction Sampling [Kask, Pezeshki, Broka, Ihler, Dechter, 2020]

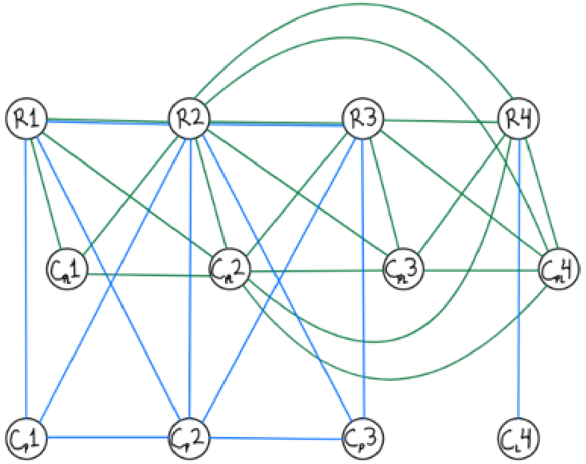
K*MAP using AND/OR Search

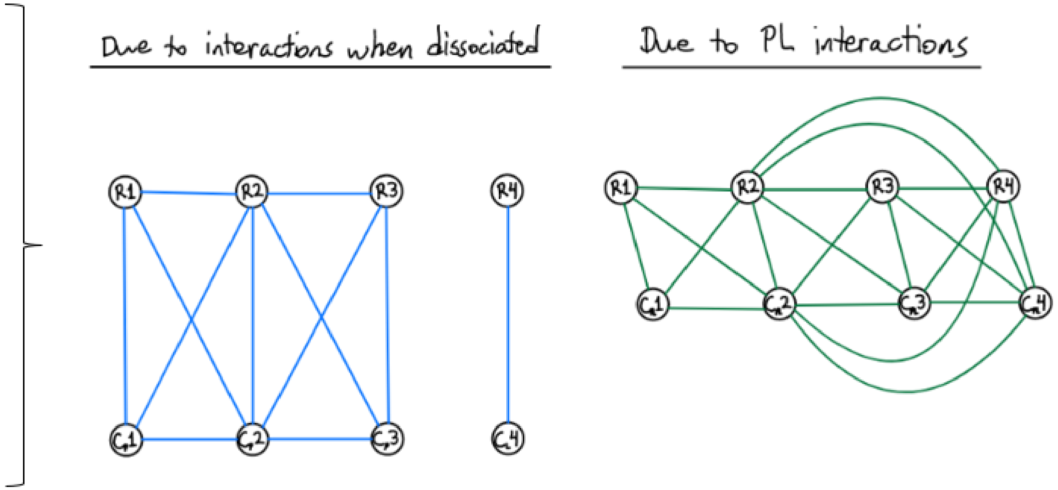
Problem Formulation

Two Formulations

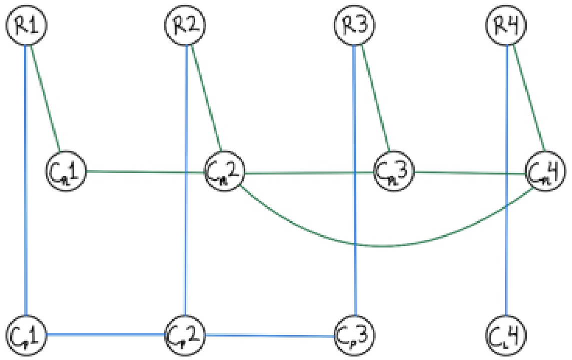


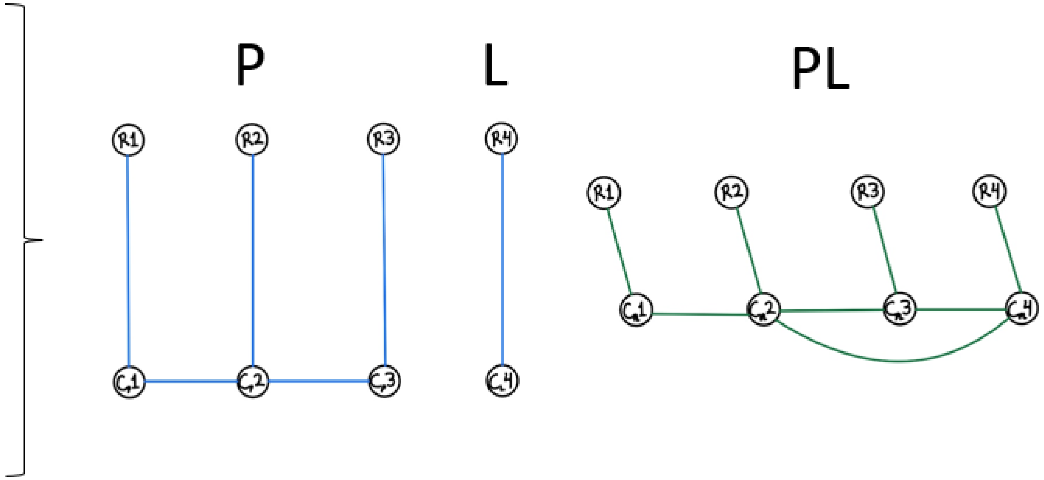
F1



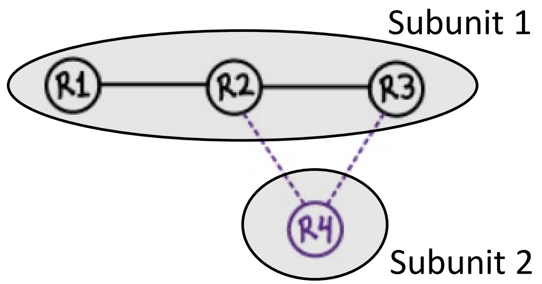


F2



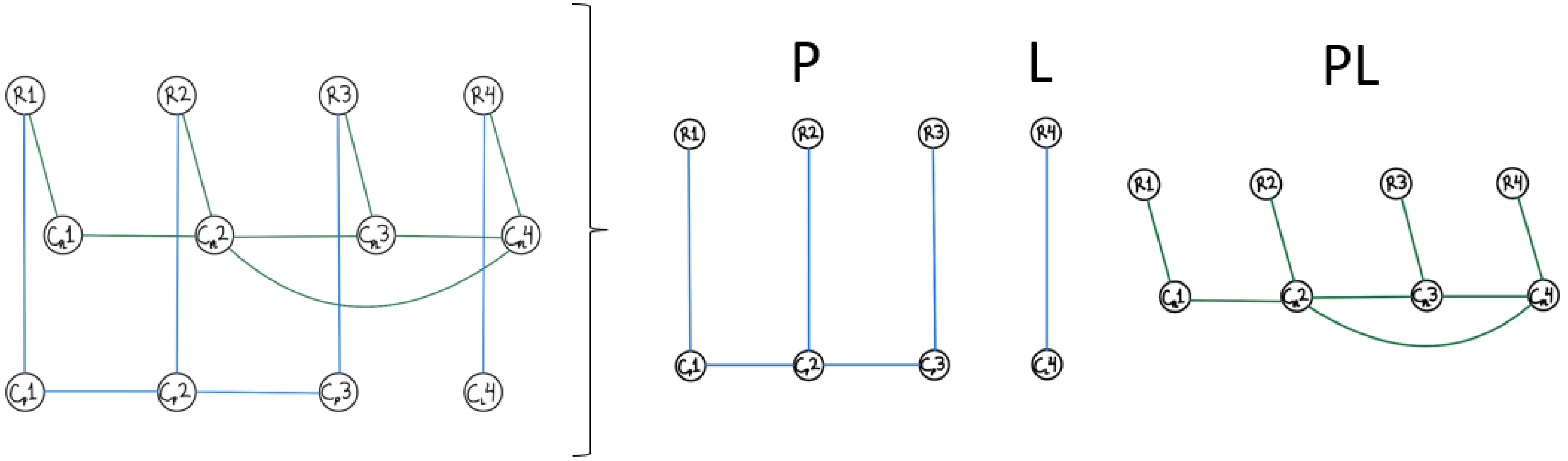


Two Formulations



F1

F2



Problem Formulation: Assumptions

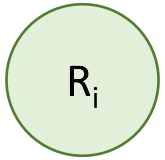
- **Select Residues:** Model using only a subset of the residues.

- **Discrete Rotamers:** Use discrete side-chain conformations.

- **Fixed Backbone:** Fix the position of the residues in space.

Problem Formulation: Variables and Domains

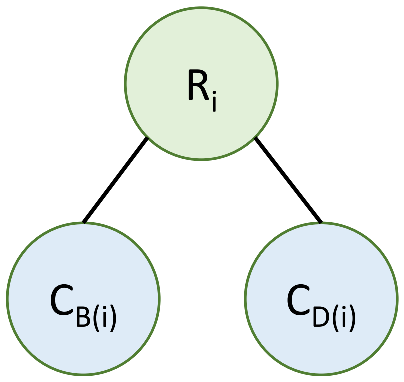
$$\boldsymbol{R} = \{\, \boldsymbol{R_i} \mid \boldsymbol{i} \in \{\mathbf{1}, \mathbf{2}, \dots, \boldsymbol{N}\} \,\}$$



- Residues considered for mutation
  - ie. variables we maximize over
- Domain = possible amino acids

Problem Formulation: Variables and Domains

$$\boldsymbol{C_X} = \{\, C_{X(i)} \mid i \in \{1, 2, \dots, N\} \,\}$$

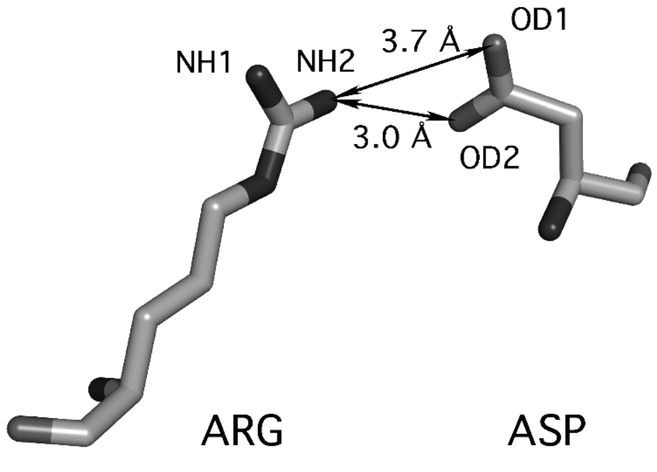


- Side-chain rotamers of the residues
  - Two for each R_i, one capturing the rotamers of the bound and the other for the unbound states
- Domain = discretized amino acid rotamers

$X \in \{Bound, Dissociate\}$

Problem Formulation: Functions

- **Interaction energies** between amino acid side chain rotamers
- **Constraints enforcing consistent assignments** between corresponding residue and conformation variables



K*MAP

$\gamma \in \{Bound, Dissociate\}$

let…

$$Z_\gamma(\boldsymbol{r}) = \sum_{\boldsymbol{C}_\gamma} \prod_{\mathscr{C}_\gamma} \mathscr{C}_{\gamma(i)}(r_i, c_{\gamma(i)}) \cdot \prod_{\boldsymbol{E}_\gamma} e^{-\frac{E_{\gamma(ij)}(c_{\gamma(i)}, c_{\gamma(j)})}{\mathscr{R}T}}$$

Objective:

$$K^*(R) = \frac{Z_B(R)}{Z_U(R)}$$

Task:

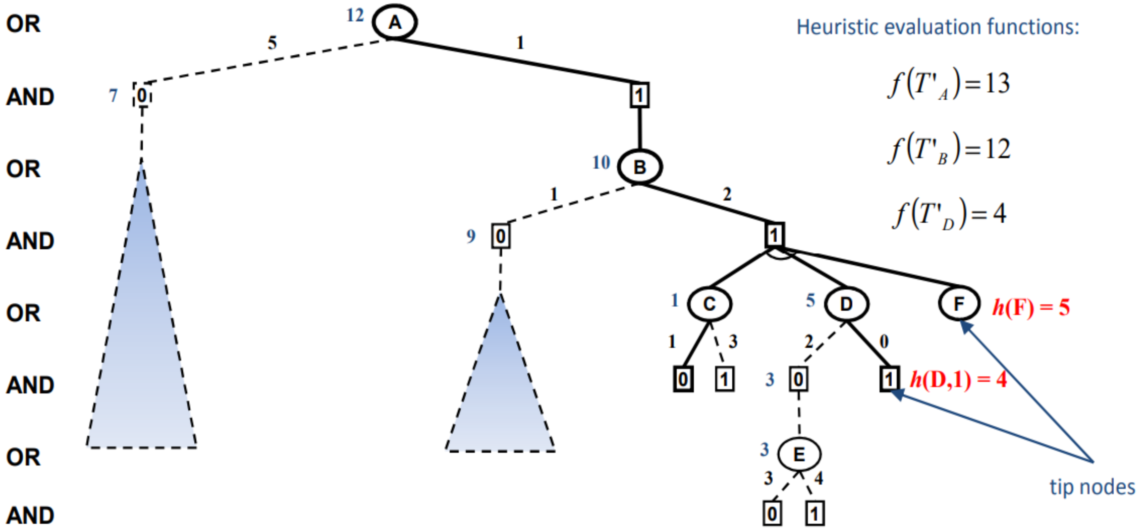
$$K^*\mathrm{MAP} = \max_{\boldsymbol{R}} K^*(\boldsymbol{r})$$

AOBB-K*

Based on AOBB-MMAP [Marinescu, Dechter, Ihler, 2014]

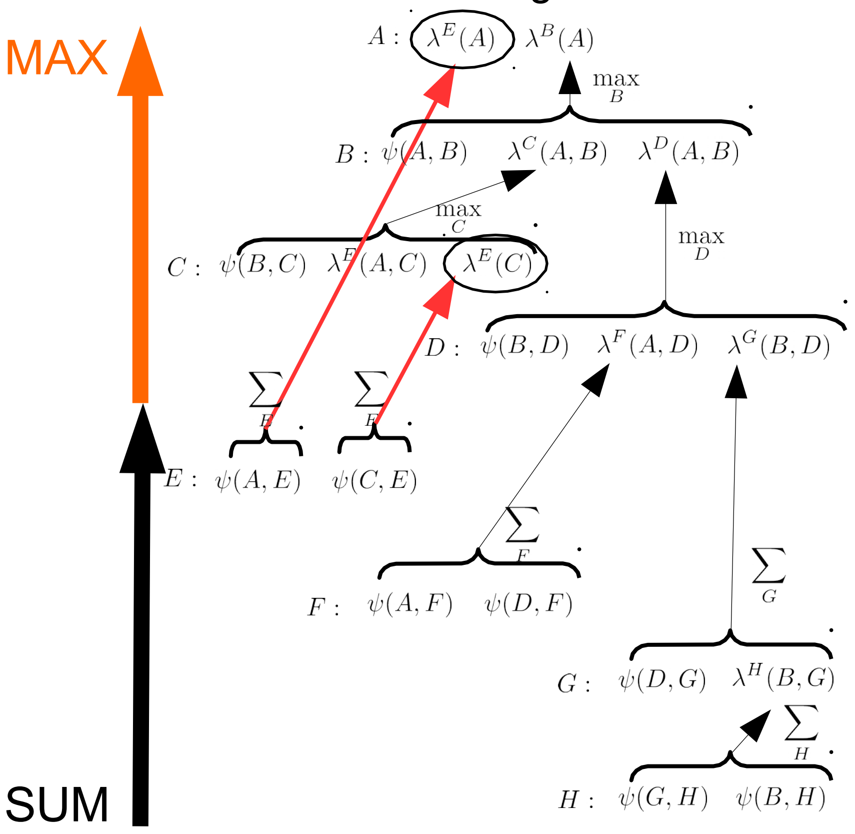
AOBB-K*

- **Branch-and-bound** over **AND/OR** search space
- Uses **wMBE-based heuristics** to guide search and prune suboptimal paths
- Uses **encodes determinism** and uses Mini SAT to **prune inconsistent paths**
- Enforces **biologically-relevant stability constraints**
- **Exact**



wMBE Heuristic for MMAP

- Mini-bucket elimination [Dechter & Rish 2001]
  - "i-bound", limit on the number of variables in a single mini-bucket



- Weighted Mini-bucket [Liu & Ihler, 2012]
  - Holder's inequality

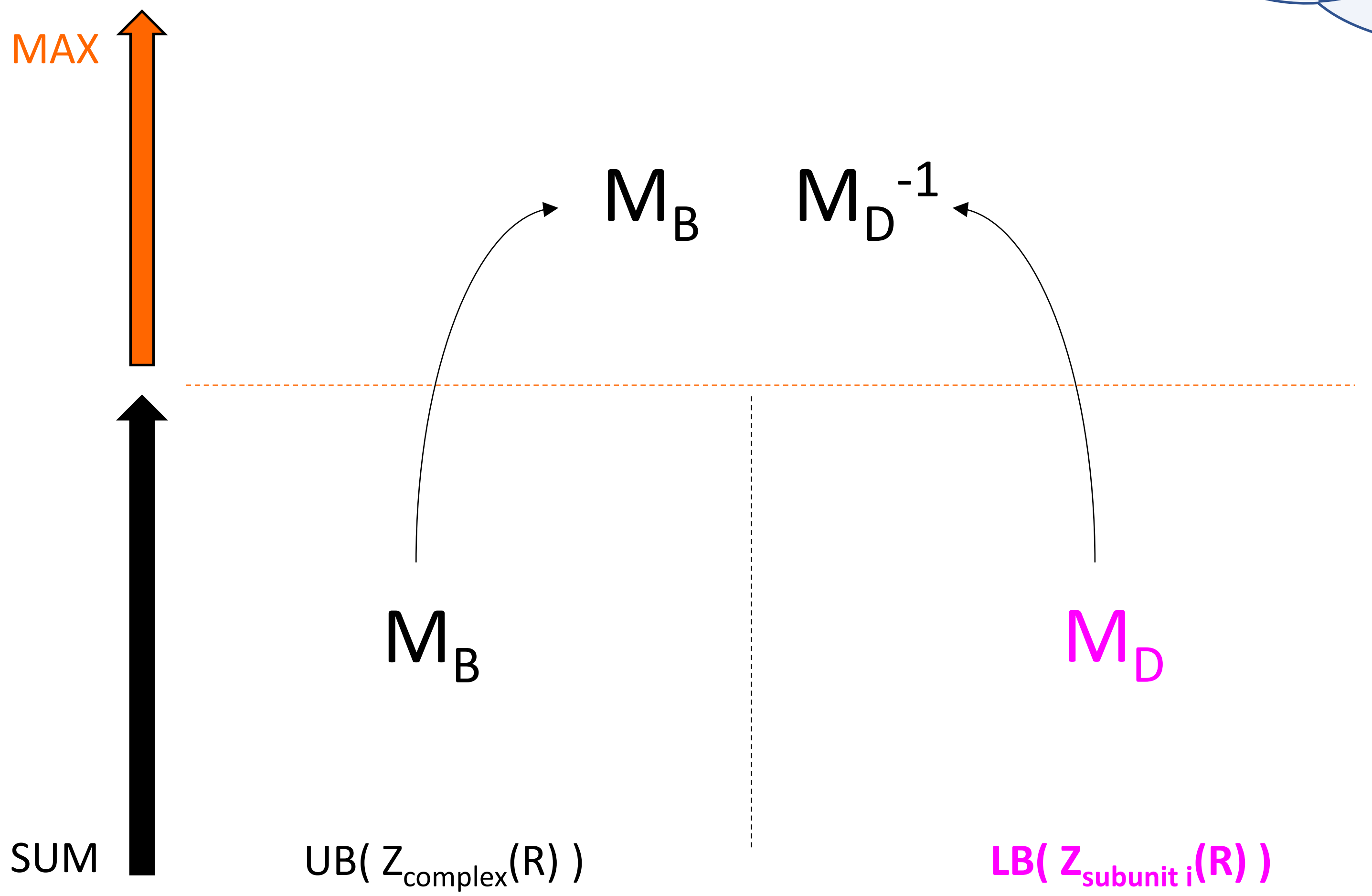
$$\sum^{w}[\prod_r(\psi_r)] \le \prod_r[\sum_x^{w_r}]$$

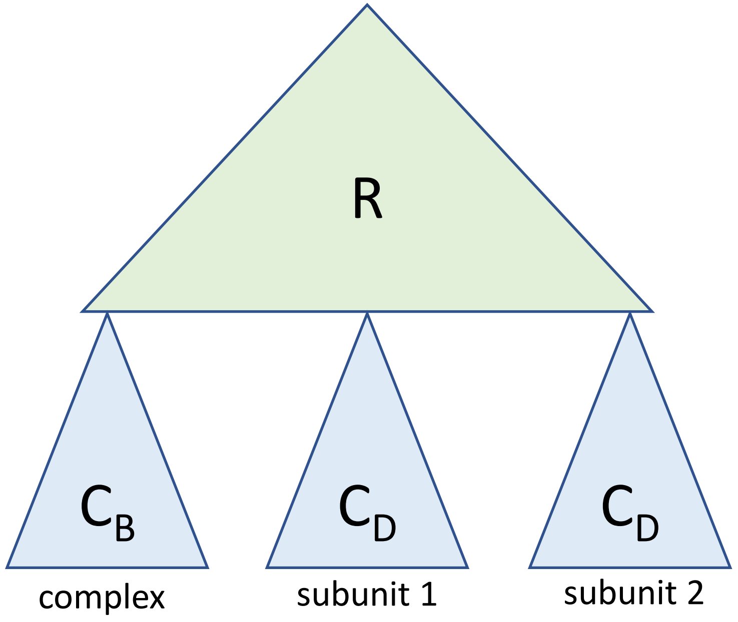
$$\sum_x^{w} f(x) \triangleq [\sum_x f(x)^{\frac{1}{w}}]^{w} \qquad w = \sum_r w_r$$

$$\sum_E[\psi(A,E)\psi(C,E)] \le [\sum_E \psi(A,E)][\sum_E \psi(C,E)]$$

wMBE Heuristic for K*

$$K^*(r) = \frac{Z_{complex}(r)}{Z_{subunit\ 1}(r)\, Z_{subunit\ 2}(r)}$$

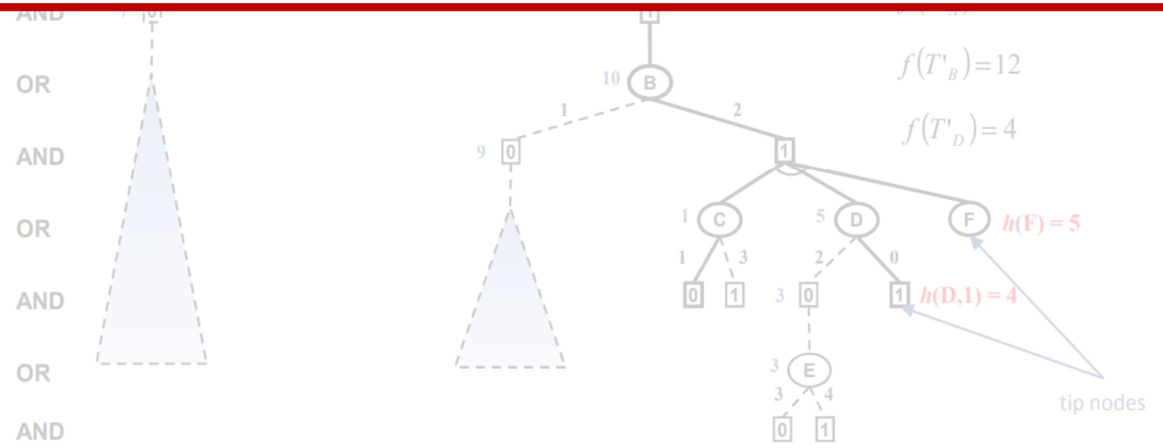




AOBB-K*

- **Branch-and-bound** over **AND/OR** search space
- Uses **wMBE-based heuristics** to guide search and prune suboptimal paths

Performed well on small problems,
but did not scale well



Scalability Improvements

List of improvements tested…

- ❑ Numerical stability fixes (part of the boosted **AOBB-K*-b** variant)
- ❑ Search the wild-type sequence first (part of the boosted **AOBB-K*-b** variant)
- ❑ Improve heuristic lower-estimates (part of the boosted **AOBB-K*-b** variant)
- ❑ Weighted heuristic search
- ❑ Dynamic heuristic recomputation
- ❑ Infuse artificial determinism to leverage CP

Underflow-Threshold Optimization (UFO)

General idea:

- During search we can use constraint processing schemes to identify inconsistent paths early on
- Problems may have "near-constraints" (i.e., very small function values) that prevent solutions that contain them in practice
- Treat "near-constraints" as constraints by underflowing their value to zero

Underflow-Threshold Optimization (UFO)

Algorithm Sketch:

- Set a time limit
- Use binary search to find the greatest constant $\tau \in [0, v_{max})$ such that
  - If we replace all function values $v < \tau$ with *0, t*here still exists a consistent path (ie. path with non-zero cost)
  - ***CPD: wild-type remains consistent***
- Relax threshold: $\tau := \tau \cdot \delta, \quad \delta \in (0, 1]$
- Replace any function value $v < \tau$ *with 0.*

Empirical Evaluation

- Does formulating the K*MAP task as a graphical model show potential?
- Which AOBB-K* scheme is best performing?
- How does performance compare to state-of-the-art BBK*?

Setup

- Real protein benchmarks obtained by the Donald Lab at Duke University
  - Contained instances for redesigning 1-3 residues
  - These were expanded to also consider redesign of 4-5 residues
- Algorithms tested
  - AOBB-K*
  - AOBB-K*-ω
  - AOBB-K*-b
  - AOBB-K*-b-DH
  - AOBB-K*-b-UFO
  - BBK*

    [Ojewole et al., 2018]

Setup

- Real protein benchmarks obtained by the Donald Lab at Duke University
  - Contained instances for redesigning 1-3 residues
  - These were expanded to also consider redesign of 4-5 residues
- Algorithms tested
  - AOBB-K*
  - AOBB-K*-ω
  - AOBB-K*-b
  - AOBB-K*-b-DH
  - AOBB-K*-b-UFO
  - BBK*
    [Ojewole et al., 2018]

Redesign of 5 Residues

		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
5	d7-5-1	AOBB-K*-b-UFO	3	**15.17**	1570.30	timeout	14.08	14.73	401.09
		AOBB-K*-b-DH	3	14.73	**57.91**	timeout	14.08	14.73	401.09
		AOBB-K*-b	3	14.73	62.53	timeout	14.08	14.73	401.09
	d7-5-3	AOBB-K*-b-UFO	3	**14.84**	891.90	timeout	14.08	15.60	205.56
		AOBB-K*-b	3	14.73	**67.53**	timeout	14.08	15.60	205.56
		AOBB-K*-b-DH	3	14.73	156.68	timeout	14.08	15.60	205.56
	d27-5-1	AOBB-K*-b	3	15.55	**274.30**	timeout	15.48	15.55	1270.65
		AOBB-K*-b-UFO	3	15.55	**276.91**	timeout	15.48	15.55	1270.65
		AOBB-K*-b-DH	3	15.55	321.02	timeout	15.48	15.55	1270.65
	d31-5-1	AOBB-K*-b-UFO	3	7.88	**22.35**	128.75	7.63	7.88	130.04
		AOBB-K*-b	3	7.88	**129.43**	timeout	7.63	7.88	130.04
		AOBB-K*-b-DH	3	7.88	145.63	timeout	7.63	7.88	130.04
	d47-5-1	AOBB-K*-b-UFO	3	**23.08**	2068.22	timeout	22.70	23.05	timeout
		AOBB-K*-b	3	22.74	**222.66**	timeout	22.70	23.05	timeout
		AOBB-K*-b-DH	3	22.74	241.88	timeout	22.70	23.05	timeout

Redesign of 5 Residues

		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
5	d7-5-1	AOBB-K*-b-UFO	3	15.17	1570.30	timeout	14.08	14.73	401.09
		AOBB-K*-b-DH	3	14.73	**57.91**	timeout	14.08	14.73	401.09
		AOBB-K*-b	3	14.73	62.53	timeout	14.08	14.73	401.09
	d7-5-3	AOBB-K*-b-UFO	3	**14.84**	891.90	timeout	14.08	15.60	205.56
		AOBB-K*-b	3	14.73	**67.53**	timeout	14.08	15.60	205.56
		AOBB-K*-b-DH	3	14.73	156.68	timeout	14.08	15.60	205.56
	d27-5-1	AOBB-K*-b	3	15.55	**274.30**	timeout	15.48	15.55	1270.65
		AOBB-K*-b-UFO	3	15.55	**276.91**	timeout	15.48	15.55	1270.65
		AOBB-K*-b-DH	3	15.55	321.02	timeout	15.48	15.55	1270.65
	d31-5-1	AOBB-K*-b-UFO	3	7.88	**22.35**	128.75	7.63	7.88	130.04
		AOBB-K*-b	3	7.88	**129.43**	timeout	7.63	7.88	130.04
		AOBB-K*-b-DH	3	7.88	145.63	timeout	7.63	7.88	130.04
	d47-5-1	AOBB-K*-b-UFO	3	23.08	2068.22	timeout	22.70	23.05	timeout
		AOBB-K*-b	3	22.74	**222.66**	timeout	22.70	23.05	timeout
		AOBB-K*-b-DH	3	22.74	241.88	timeout	22.70	23.05	timeout

Conclusion

*This simplified K*MAP task formulated as a graphical model allows existing graphical model algorithms to be adapted to the task and shows potential against current state of the art algorithms.*

End Part 2

Special Thank You’s

Professor Rina Dechter

Thank you for always kindly pushing me to grow. Your dedication, heart, and wisdom have left a permanent mark on me…

Professor Alexander Ihler

Thank you for investing in me and always helping me to understand complex concepts, both technically and intuitively…

Dear Mentors

Thank you for your guidance and support at all hours, and always taking the initiative and putting more on your shoulders to help the rest of us.

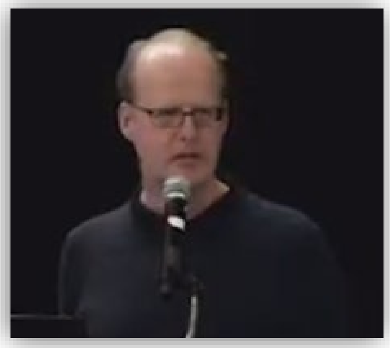
JUNKYU LEE

FILJOR BROKA

Thank you for believing that I can transfer my personality and passion into my work and the hours you stayed late to explain concepts to me.

Thank you for your positivity and all the meetings from across the world. Not to mention tag teaming those conference deadlines…

RADU MARINESCU



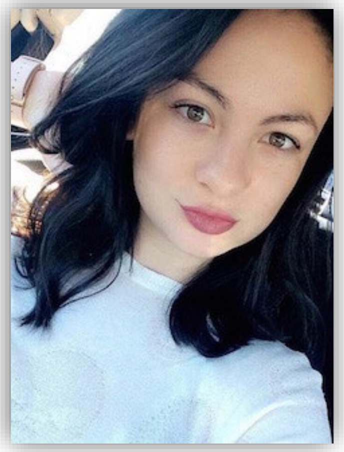
PROF. KALEV KASK

Thank you for helping me transition into my first research project with Abstraction Sampling and mentoring me as an instructor.

Partners in the lab…

Thank you for paving the way for our lunches, walks, and talks (research, philosophical, and otherwise). I look forward to many more to come.

NICHOLAS COHEN



ANNIE RAICHEV

Thank you for being the burst of energy in our group. You give the best presentations, and your cats are the cutest. #Barcelona 2024

Also special thanks to…

SHUFENG KONG,
YASAMAN RAZEGHI,
JIAPENG ZHAO,
SAKSHI AGARWAL

My Family

My Family

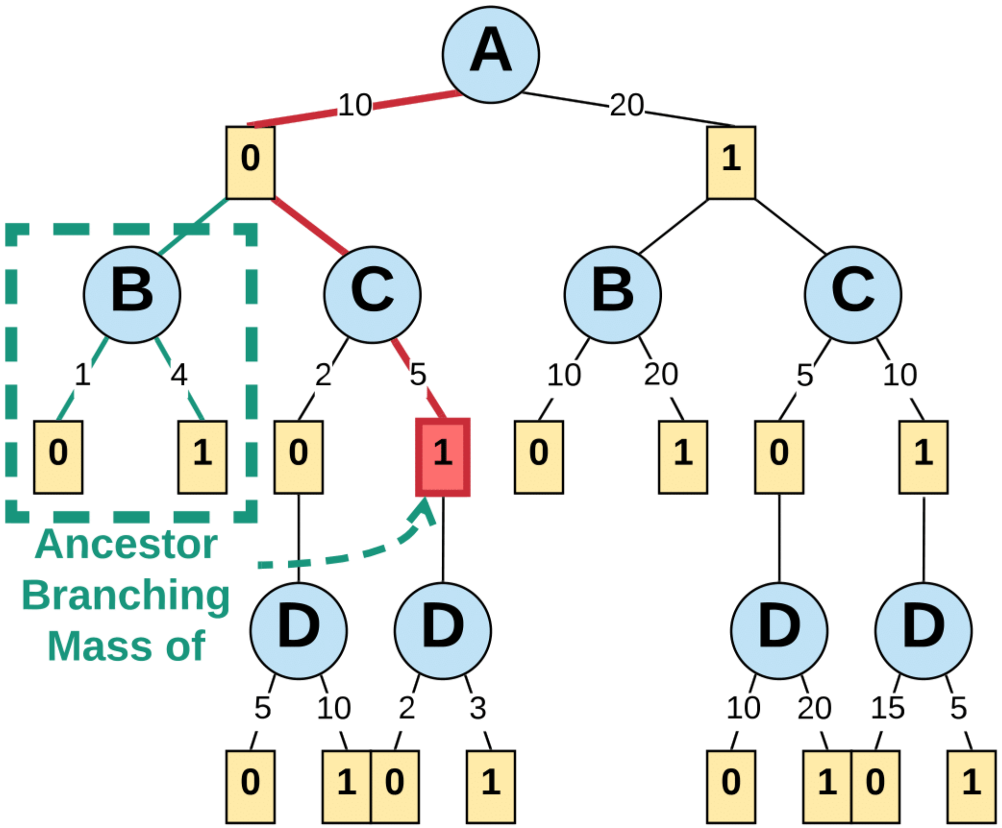
...and friends

And so many others…

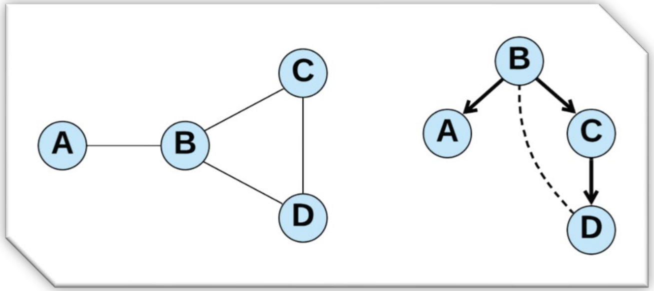
- *Professors from our department who always have their door open willing to help students, and constantly working to improve our department*
  Prof. Padhraic Smyth, Prof. Erik Sudderth, Prof. Sameer Singh, …
- *Professors who have offer enriching courses from which to grow upon*
  Prof. Stephen Mandt, Prof. Weining Shen, Prof. Michael Dillencourt…
- *Professors whom I've TA'ed for and/or are helping me evolve instructionally*
  Prof. Jennifer Wong-Ma, Raymond Klefstad, Prof. Bob Pelayo
- *My Cohort:* Claudio, Kyu-Seon, Wonnie, Madina, Gabe, John, Akshay, AK, Siwei
- *OIT Help Desk for their never-ending support of our technical issues*
- *ICS/CS Department Staff constantly helping take care of logistics and making sure our departments are running smoothly*
- *Counseling Staff that helped guide me through this process*

END

Ancestor Branching Mass

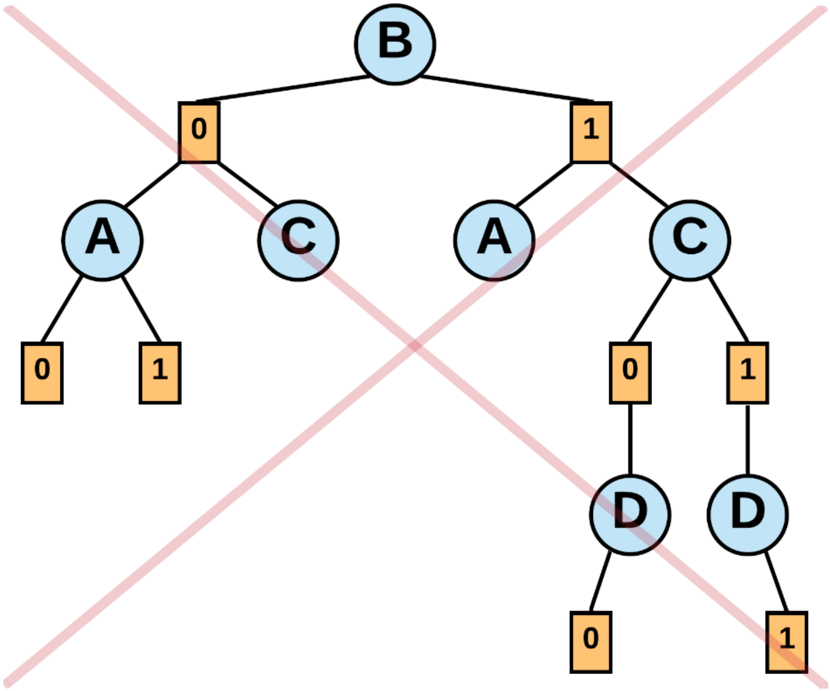


AND/OR Schemes

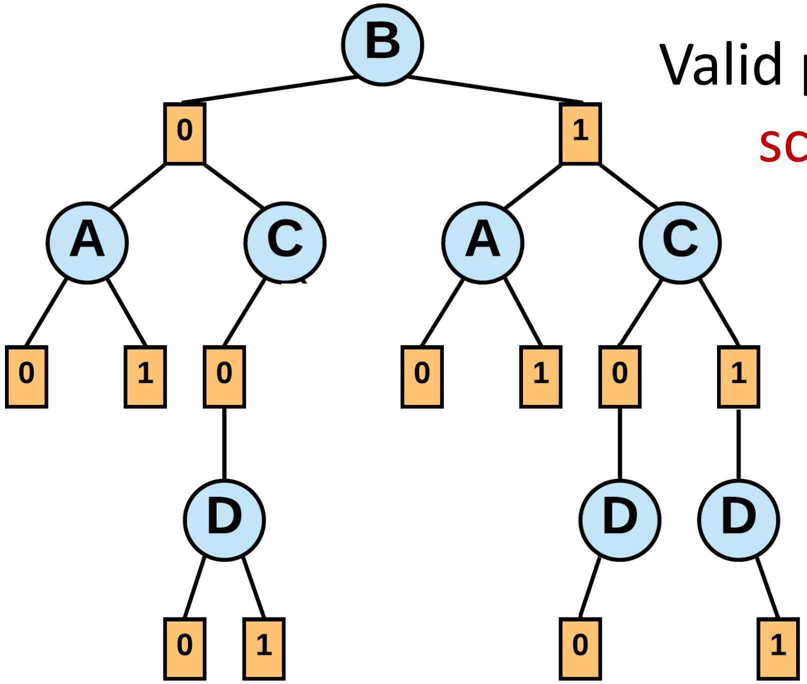


"proper" abstractions ensure that every AND/OR probe includes a valid configuration.

Naïve Non-proper Abstractions

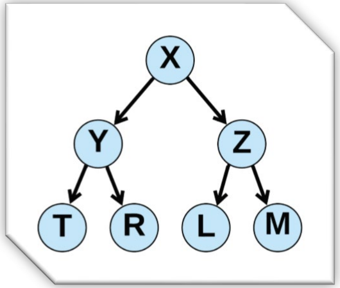


Proper Abstractions

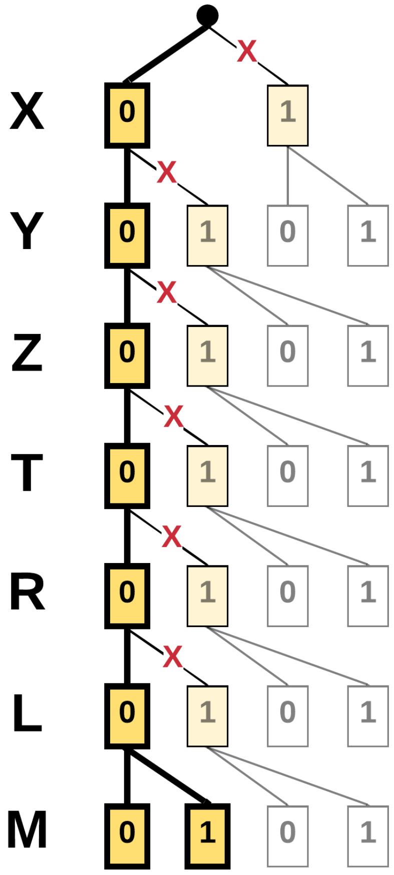


Valid probes, but severe scalability issues

Scalability Issues



Properness restricts the scope of abstractions leading to serious scalability issues.

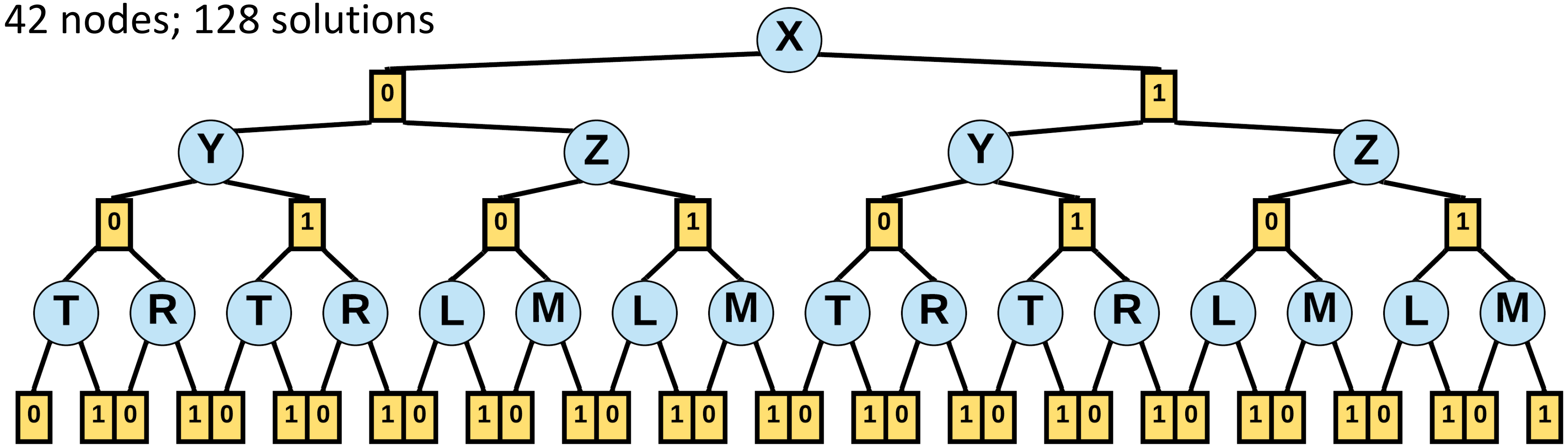


← OR Abstraction Sampling

8 nodes; 2 solutions

Proper AND/OR Abstraction Sampling

42 nodes; 128 solutions



Properties

Complexity

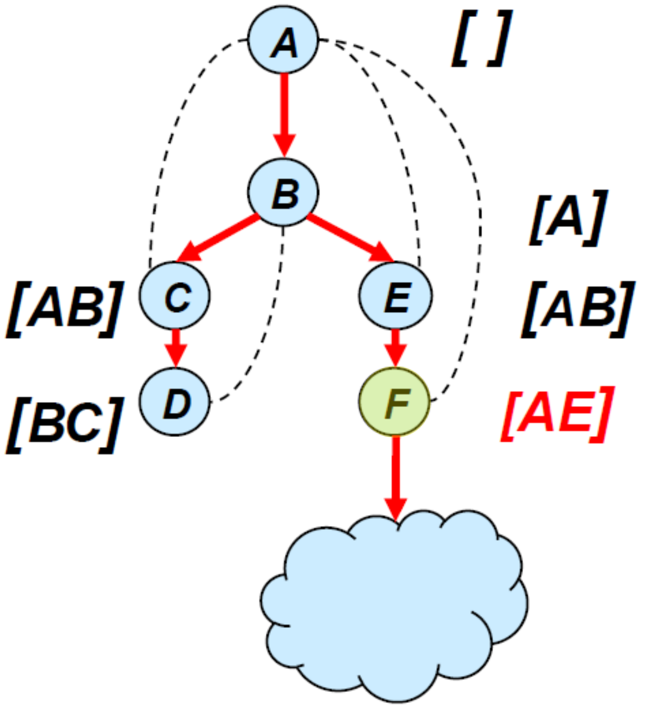
$\boxed{O(n \cdot m)}$ where n is the number of variables, and m is the number of abstract states per variable

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* $\mathcal{M}$.

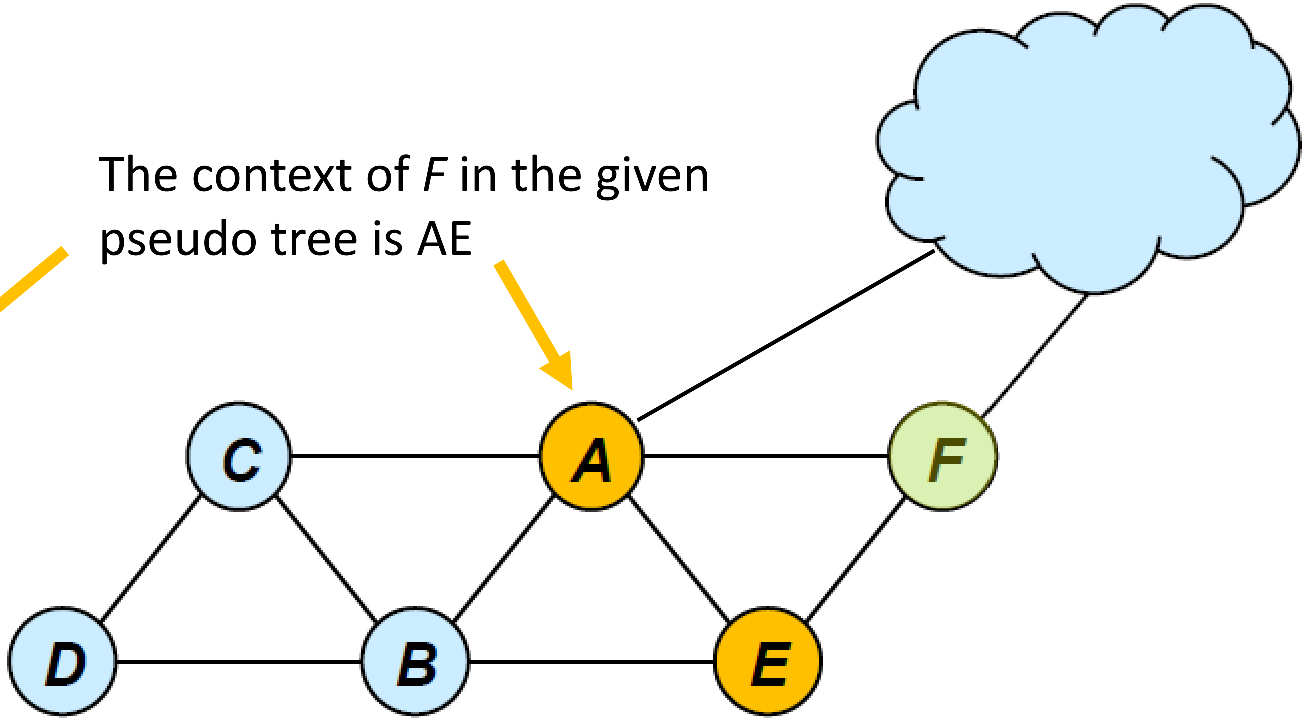
Context-Based Abstractions – Defining Context

Set of pseudo tree ancestors whose assignment causes conditional independence of a variable's subtree with all other variables



Pseudo-Tree

The context of *F* in the given pseudo tree is AE

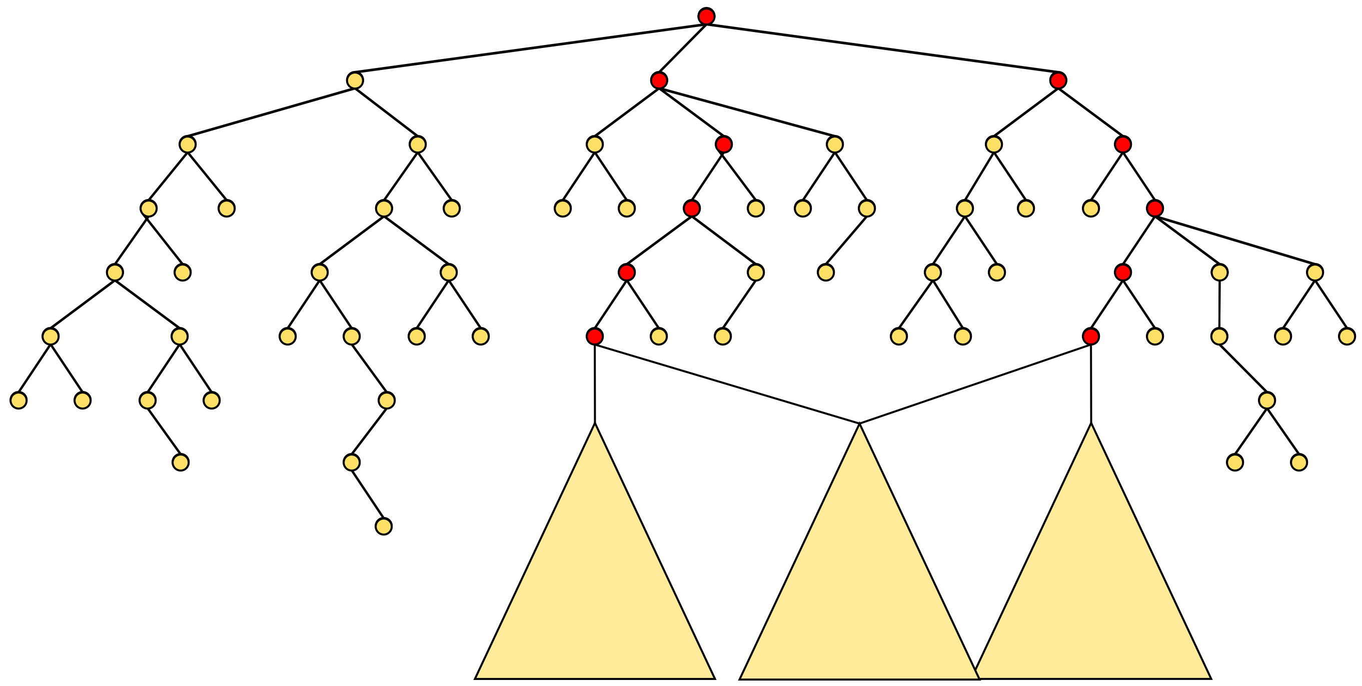


Primal Graph

Context-based Abstractions - Intuition

We know from search that we can merge nodes that root identical subtrees.

[Dechter and Mateescu, 2006]



Paths having the same assignments to the "context" of a variable

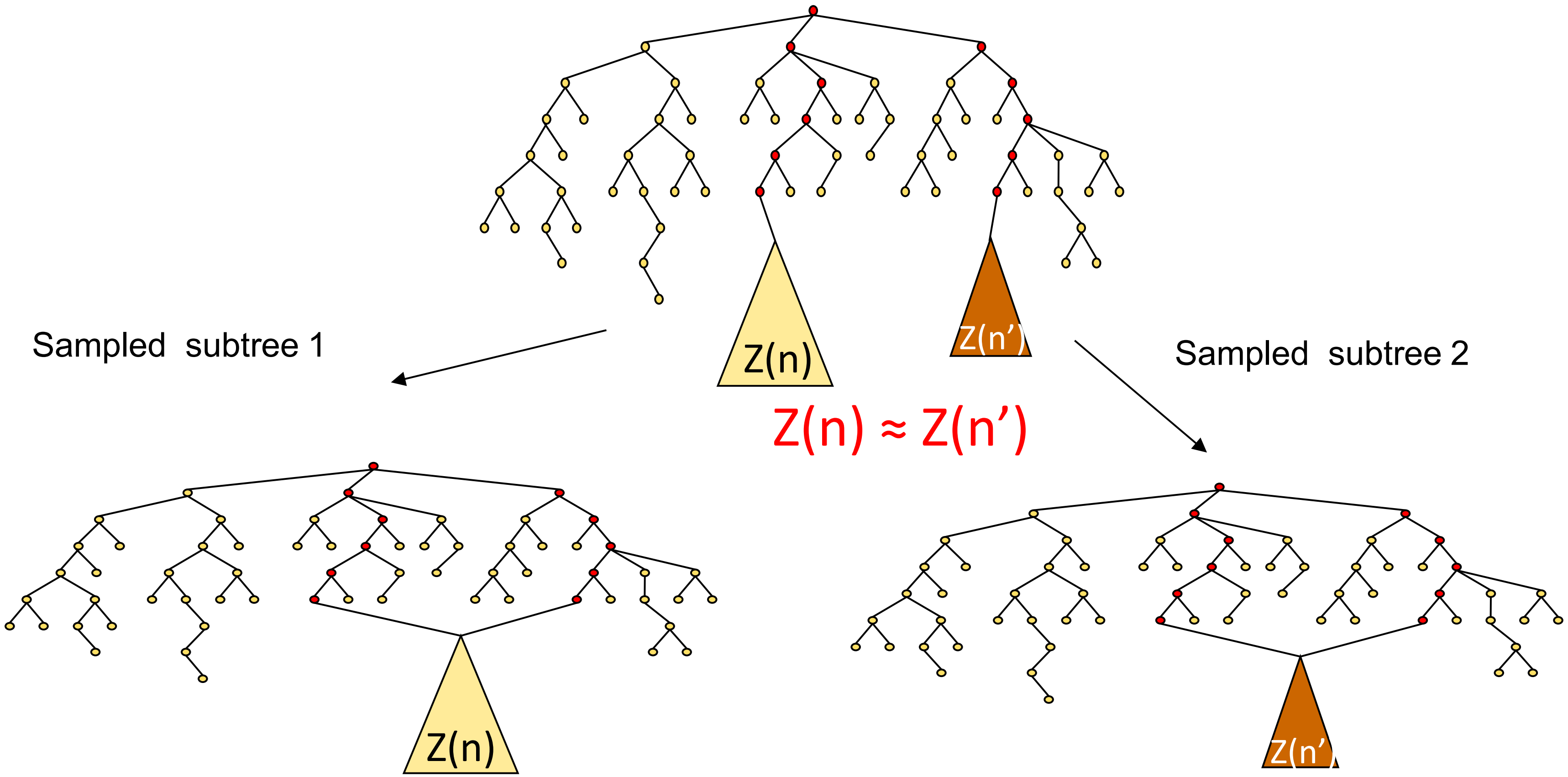
Determinable via structural analysis

⇓

nodes of that variable along those paths root the exact same subtree

Relaxed Context-Based Abstractions – Intuition

What if we abstract nodes that root ~~identical~~ similar subtrees...



Relaxed Context-Based Abstractions

- Relaxed Context-Based Abstractions (RelCB)
  - Use subset of ***most recent*** context variables
  - Granularity parameter limits number context variables

- Randomized Context-Based Abstractions (RandCB)
  - Use full context, but randomly hash into a bounded number of abstract states
  - Granularity parameter limits number of abstract states

AOAS vs. DIS [Lou, Dechter, Ihler, 2019]

Problem: benchmark name

Size: difficulty of subset of problems

Total: total number of instances

∈Bnds: number of times AOAS's estimate fell within DIS's 95% probabilistic bounds

AOAS≥ : number times AOAS's* estimates were comparable to or better than DIS's**

AOAS>: number times AOAS's* estimates were strictly better than DIS's

Problem	Size	Total	∈Bnds	AOAS≥	AOAS>
DBN	small	66	62	57	47
	large	48	40	38	35
Grids	small	8	5	5	2
	large	19	7	7	6
Linkage[1]	large	82	82	82	82
Pedigree	small	24	24	24	19
Promedas	small	65	58	49	29
	large	173	165	141	113

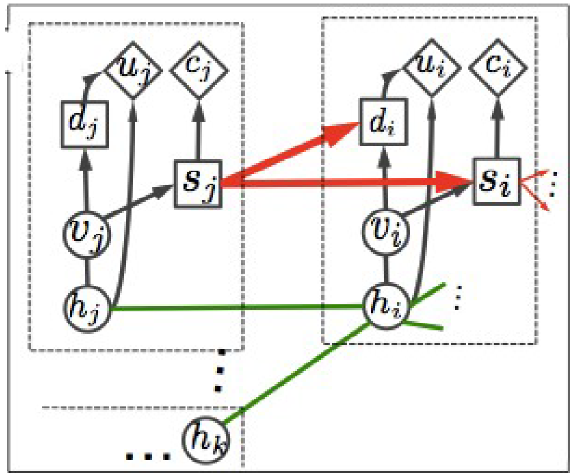
* for this table, AOAS refers to AOAS RandCB-256

**comparable means falling within ±0.1 or ±0.5 of DIS's estimate, for small and large problems respectively

Marginal MAP (MMAP)

$$MMAP(\mathcal{M}, X_{MAP}) = \max_{X_{MAP}} \sum_{\boldsymbol{X}/X_{MAP}} \prod_{\alpha} f_{\alpha}(X_{\alpha})$$

Influence diagram:



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

$MMAP$

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

Example: decision making

Sum over random variables (random effects, etc.)

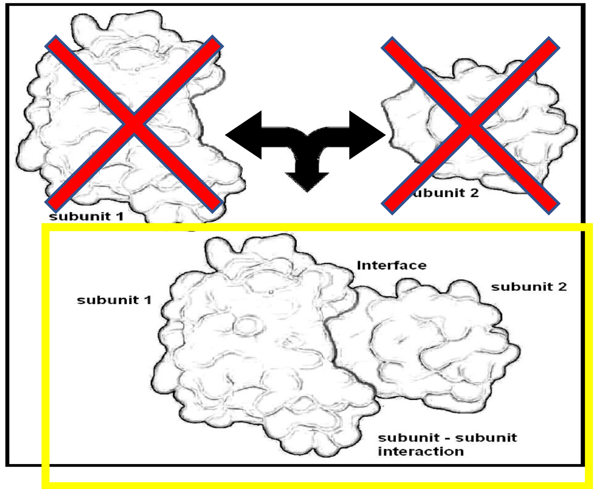
Max over decision variables (specify action policies)

GMEC Objective

Lower Energy ➔ More Stable ➔ Structure More Likely To Exist

Def. Global Minimum-Energy Conformation (GMEC):

- conformation that minimizes the energy of the complex



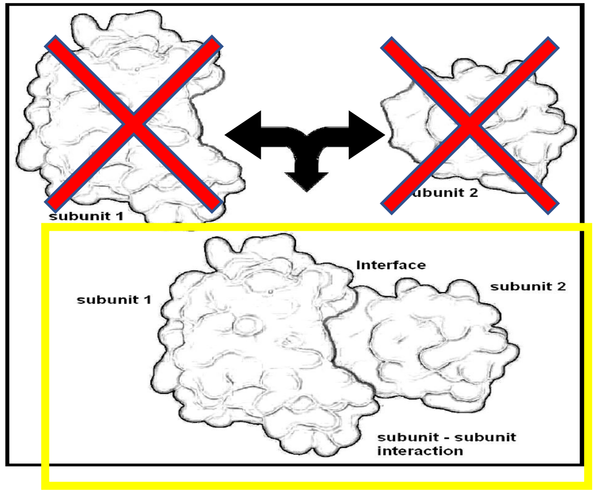
$$GMEC(r) = \min_{c \in C(r)} E(c)$$

r = amino acid assignments to the residues
C(r) = possible rotamer conformations given a.a. sequence r
E(c) = energy given conformation c

GMEC Objective

M = minimum

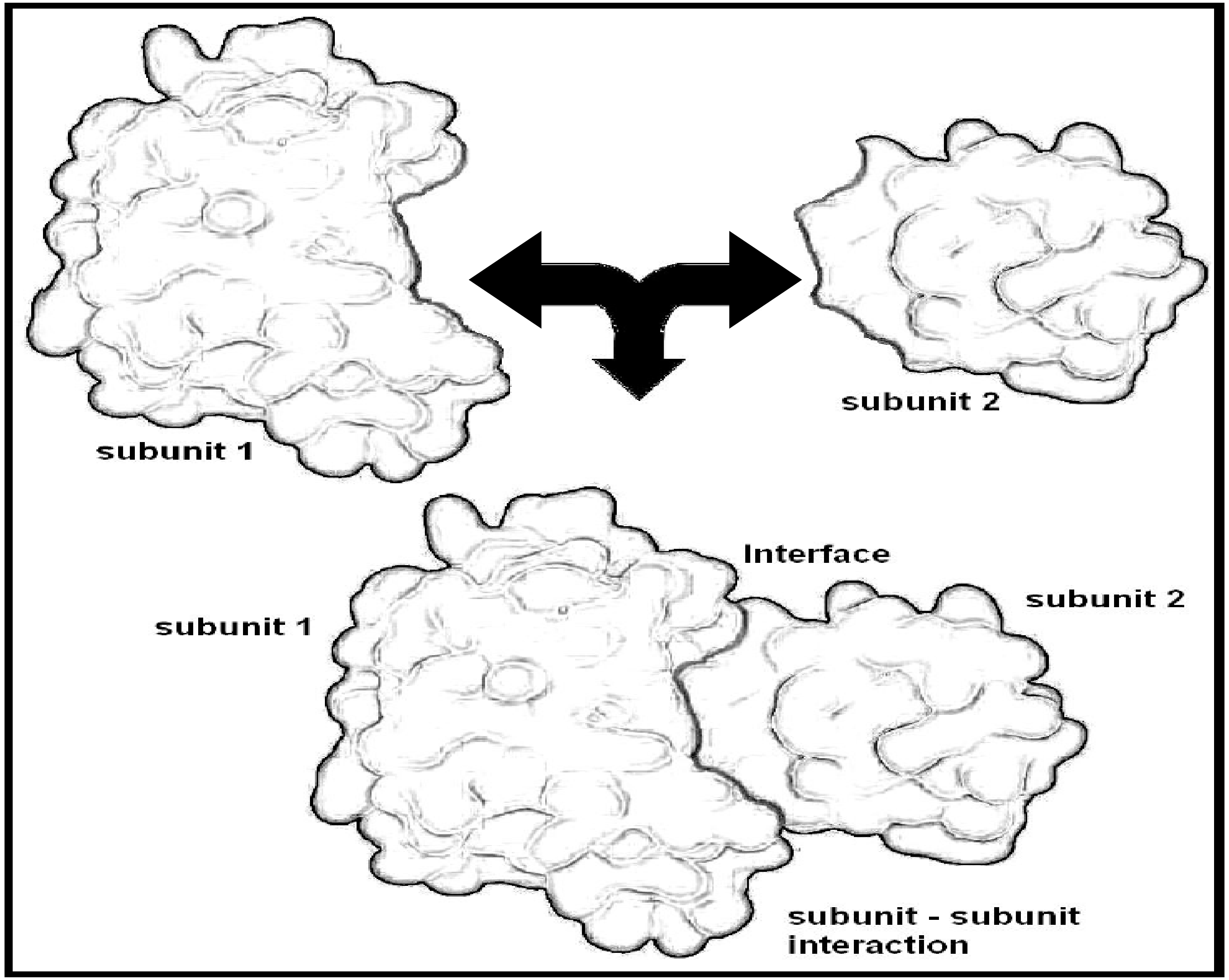
$$GMEC\ MAP = \min_{R} GMEC(r)$$



ie. Find the sequence with the lowest GMEC

- ie. Find sequence that has the most stable conformation

Proteins are Dynamic Structures

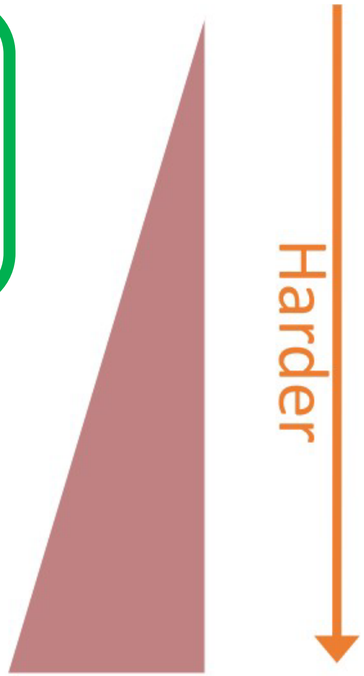


Sowmya, Gopichandran & Vaishnavi, A. & Jigisha, A. & Kangueane, Pandjassarame. (2011). Protein-protein complexes.

GMEC Objective

$GMEC\ MAP = \min_{R} GMEC(r)$

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



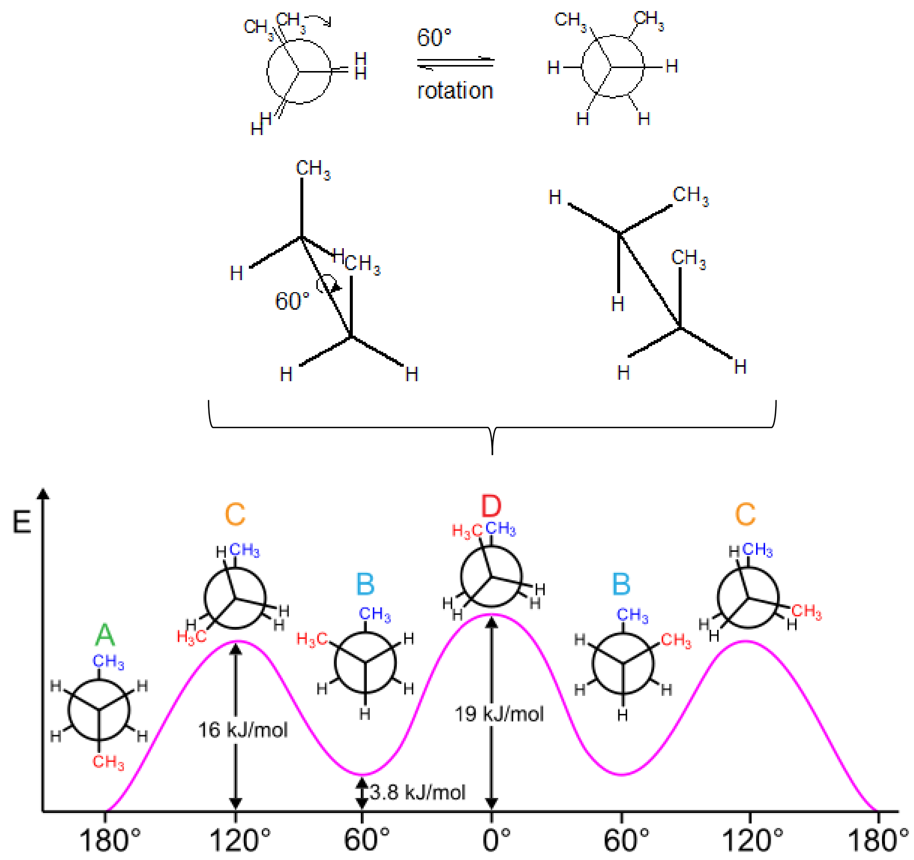
- **NP-hard**: exponentially many terms

Problem Formulation: Functions

$$\boldsymbol{E_X^{sb}} = \left\{E^{sb}_{X(i)}(R_i, C_i) \,\middle|\, i \in \{1, 2, \dots, N\}\right\}$$

- Energy of interactions between residues and their surrounding backbone

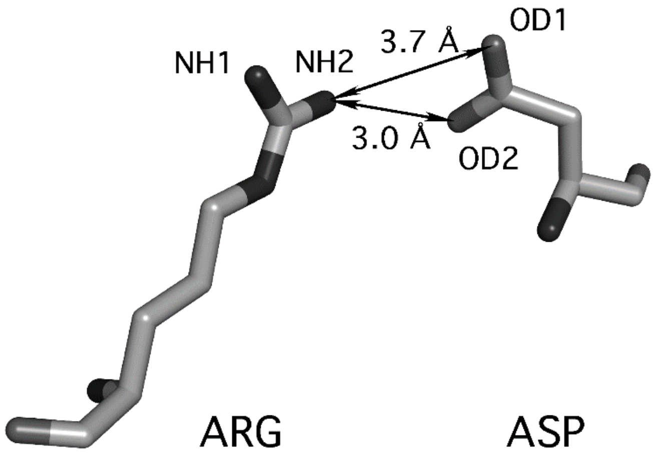
$X \in \{Bound, Dissociate\}$



Problem Formulation: Functions

$$\boldsymbol{E}_{\boldsymbol{X}}^{\boldsymbol{pw}} = \left\{ E_{X(i,j)}^{pw}(R_i, C_i, R_j, C_j) \middle| \ \forall i, j \text{ st. } R_i \text{ and } R_j \text{ interact} \right\}$$

- Energy of interactions between pairs of residues that interact



$X \in \{Bound, Dissociate\}$

K*MAP

$X \in \{Bound, Dissociate\}$

$$Z_X(\boldsymbol{r}) = \sum_{\boldsymbol{C_X}} \prod_{\boldsymbol{E_X^{sb}}} e^{-\frac{E^{sb}_{X(i)}(r_i, C_{X(i)})}{\mathcal{R}T}} \prod_{\boldsymbol{E_X^{pw}}} e^{-\frac{E^{pw}_{X(i,j)}(r_i, C_{X(i)}, r_j, C_{X(j)})}{\mathcal{R}T}}$$

$$K^*(\boldsymbol{r}) = \frac{Z_{Bound}(\boldsymbol{r})}{Z_{Dissociate}(\boldsymbol{r})} = \frac{Z_{complex}(\boldsymbol{r})}{Z_{subunit\ 1}(\boldsymbol{r})\, Z_{subunit\ 2}(\boldsymbol{r})}$$

$$K^*MAP = \max_{\boldsymbol{R}} K^*(\boldsymbol{r})$$

Problem Formulation: Subunit-Stability Constraints

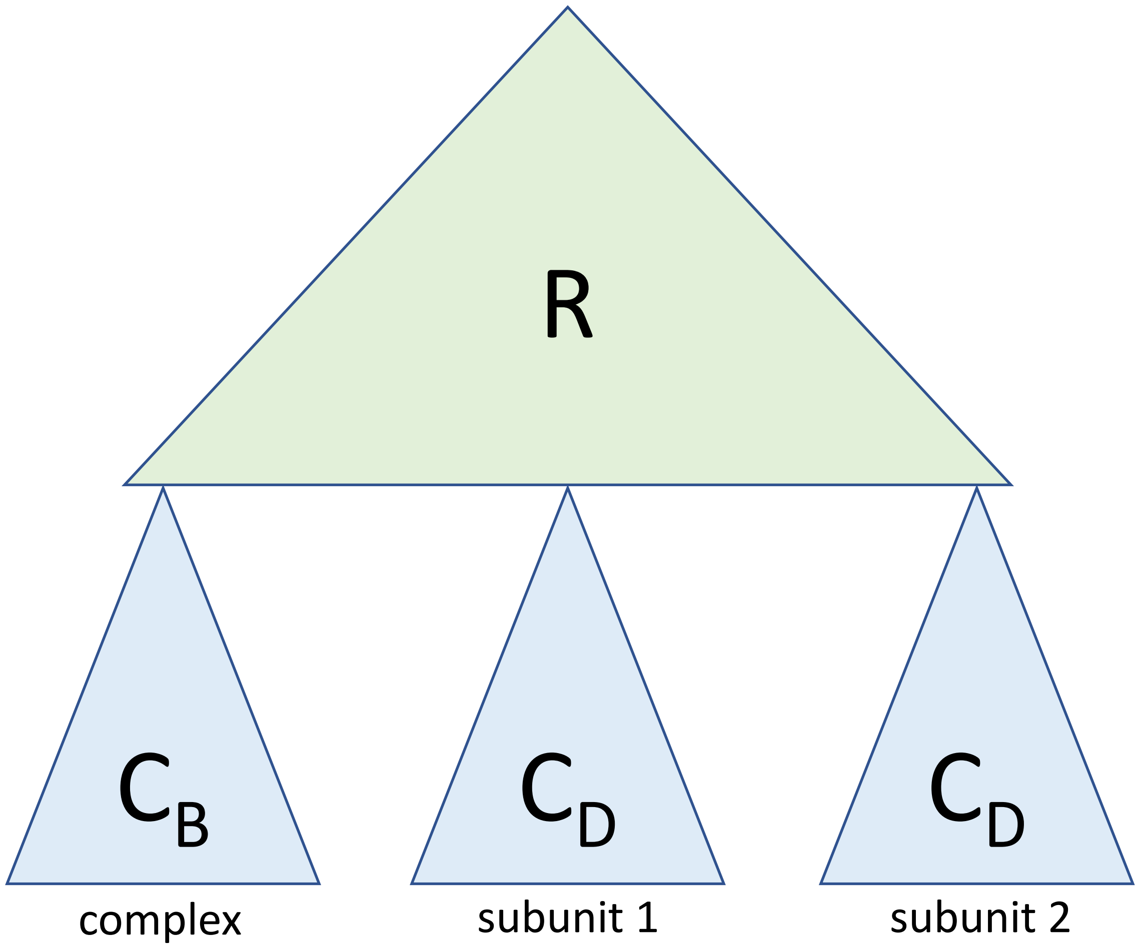
$$K^*(r) = \frac{Z_{complex}(r)}{Z_{subunit\ 1}(r)\, Z_{subunit\ 2}(r)}$$

Do not want dissociate subunits to be too unstable

$$Z_{subunit\ i}(r) > \underbrace{Z_{subunit\ i}(r^{wt})}_{\text{Likelihood of naturally occurring version}} * \underbrace{\exp\{-5/\mathcal{R}\boldsymbol{T}\}}_{\text{Constant factor to threshold with}}$$

i = index of dissociate subunit
r = amino acid sequence assignments
D = indicating dissociate subunit
r^{wt} = naturally occurring in nature amino acid sequence (wild type)
R = universal gas constant (for unit conversion between kJ and K)
T = absolute temperature (Kelvin)

Problem Formulation: Pseudo Tree Overview for K*MAP



ω-Weighted Search

benchmark	ω	iB	w*	d	|X|	UB	pre-t	search	time	K*
					(203 ≤ Dmax ≤ 206)				ω-AOBB-K*	
1gwc_00021*	1	4	4	7	13	28.80	123.8	81.3	205.1	11.92
	0.001	4	4	7	13	28.80	124.3	12.1	136.4	
2hnv_00025*	1	4	6	9	17	42.32	109.8	44.0	153.8	16.18
	0.001	4	6	9	17	42.32	109.3	12.2	121.5	
2rf9_00013*	1	4	6	9	17	37.68	83.0	17.8	100.8	15.03
	0.001	4	6	9	17	37.68	82.8	1.6	84.4	
2rfe_00012*	1	4	5	8	15	34.07	58.3	2.6	60.9	13.93
	0.001	4	5	8	15	34.07	58.6	0.3	58.9	
2rfe_00014*	1	4	5	8	15	35.07	58.2	2.4	60.6	14.36
	0.001	4	5	8	15	35.07	58.2	1.3	59.4	
2rfe_00017*	1	5	5	8	15	26.39	166.8	167.8	334.6	10.86
	0.001	4	5	8	15	27.39	89.1	5.0	94.1	
2rfe_00030*	1	4	5	8	15	31.34	115.2	161.5	276.6	11.1
	0.001	4	5	8	15	31.34	101.4	0.5	101.9	10.9
2xgy_00020*	1	5	5	8	15	26.24	81.8	278.9	360.7	10.96
	0.001	4	5	8	15	27.24	60.4	8.2	68.6	
3u7y_00009*	1	4	4	7	13	11.41	62.6	36.8	99.5	4.51
	0.001	4	4	7	13	11.41	62.5	2.1	64.7	
3u7y_00011*	1	4	4	7	13	28.29	74.0	2.1	76.1	11.85
	0.001	4	4	7	13	28.29	83.4	0.1	83.5	
4wwi_00019*	1	5	5	8	15	36.96	169.2	12.1	181.3	14.99
	0.001	4	5	8	15	37.96	62.0	7.2	69.2	

Boosted Variants of AOBB-K* / wMBE-K*

3 Mut	AOBB-K*-b				AOBB-K*		
Problem	i	Soln	Time	wt K*	i	Soln	Time
00007	4	14.73	269.3	14.08	-	-inf	t/o
00009	4	4.51	79.9	4.09	4	4.51	99.5
00011	4	11.85	102.2	11.75	4	11.85	76.1
00012	4	13.93	69.1	13.93	4	13.93	60.9
00013	4	15.03	101.9	13.25	4	15.03	100.8
00014	4	14.36	70.9	13.96	4	14.36	60.6
00017	4	10.86	118.0	10.52	5	10.86	334.6
00019	4	14.99	77.6	14.99	5	14.99	181.3
00020	4	10.96	101.5	10.60	5	10.96	360.7
00021	4	11.92	200.4	9.37	4	11.92	205.1
00025	4	16.18	168.6	10.74	4	16.18	153.8
00030	4	11.12	154.3	10.35	4	11.12	276.6

Dynamic Heuristics

M	Problem	Algorithm	i	Soln	Time	OR	AND
3	d11-3-1	AOBB-K*-b-DH	3	11.85	24.64	3	15
		AOBB-K*-b	3	11.85	60.99	58	197
		AOBB-K*-b	4	11.85	102.18	3	5
	d12-3-1	AOBB-K*-b-DH	3	13.93	22.06	3	13
		AOBB-K*-b	3	13.93	20.72	21	122
		AOBB-K*-b	4	13.93	69.05	3	4
	d14-3-1	AOBB-K*-b-DH	3	14.36	26.95	3	16
		AOBB-K*-b	3	14.36	21.92	25	132
		AOBB-K*-b	4	14.36	70.88	3	5
	d30-3-1	AOBB-K*-b-DH	3	11.12	54.02	70	141
		AOBB-K*-b	3	11.12	2019.77	254	3666
		AOBB-K*-b	4	11.12	154.28	22	25
4	d18-4-2	AOBB-K*-b-DH	3	16.58	598.38	40	56
		AOBB-K*-b	3	16.58	3488.56	279	1214
	d24-4-1	AOBB-K*-b-DH	3	12.96	407.78	92	251
		AOBB-K*-b	3	12.96	487.66	94	437
	d27-4-1	AOBB-K*-b-DH	3	15.55	405.89	57	137
		AOBB-K*-b	3	15.55	254.67	57	137
	d28-4-1	AOBB-K*-b-DH	3	15.27	37.78	9	12
		AOBB-K*-b	3	15.27	21.62	9	19
	d28-4-2	AOBB-K*-b-DH	3	15.27	576.98	93	230
		AOBB-K*-b	3	15.27	323.45	59	166
	d42-4-1	AOBB-K*-b-DH	3	22.65	2897.35	18	61
		AOBB-K*-b	3	22.65	3025.80	24	114
	d43-4-2	AOBB-K*-b-DH	3	18.04	483.55	346	476
		AOBB-K*-b	3	18.04	112.50	56	75

Conclusion

- Graphical Model formulation for K*MAP task
- wMBE-K*, wMBE-based heuristic for bounding K*
- AOBB-K*, MMAP-like AND/OR search algorithms for K*MAP
- Multiple improvements to improve scalability
  - Weighted Search
  - Tuning of AOBB-K* and wMBE-K*
  - Dynamic Heuristics
  - UFO
    - Also as an independent scheme
- Strong performance in comparison to state-of-the-art BBK*

Future Work

- Test structures that have conditional independences between their residues
- Extend other well-known approximate anytime methods
- More compact sparse representation
- Improve heuristic function
  - Use sampling / search for lower bound?
  - Incorporate pruning constraint
- k-Best Solutions

Redesign of 3 Residues

		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
3	d19-3-1	AOBB-K*-b-UFO	3	14.99	**6.15**	621.83	14.99	14.99	34.00
		AOBB-K*-b-DH	3	14.99	**11.31**	56.05	14.99	14.99	34.00
		AOBB-K*-b	4	14.99	75.99	76.00	14.99	14.99	34.00
	d20-3-1	AOBB-K*-b-UFO	3	10.96	**13.70**	480.77	10.60	10.96	1388.13
		AOBB-K*-b-DH	3	10.96	**39.67**	339.91	10.60	10.96	1388.13
		AOBB-K*-b	4	10.96	100.02	100.03	10.60	10.96	1388.13
	d21-3-1	AOBB-K*-b-UFO	3	11.92	**89.03**	628.59	9.37	11.72	551.27
		AOBB-K*-b-DH	3	11.92	**136.44**	1307.45	9.37	11.72	551.27
		AOBB-K*-b	4	11.92	**193.83**	196.52	9.37	11.72	551.27
	d25-3-1	AOBB-K*-b-UFO	3	16.18	**14.02**	64.82	10.74	13.65	880.46
		AOBB-K*-b-DH	3	16.18	**51.92**	80.22	10.74	13.65	880.46
		AOBB-K*-b	4	16.18	**166.74**	166.75	10.74	13.65	880.46

Redesign of 3 Residues

		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
3	d19-3-1	AOBB-K*-b-UFO	3	14.99	**6.15**	621.83	14.99	14.99	34.00
		AOBB-K*-b-DH	3	14.99	**11.31**	56.05	14.99	14.99	34.00
		AOBB-K*-b	4	14.99	75.99	76.00	14.99	14.99	34.00
	d20-3-1	AOBB-K*-b-UFO	3	10.96	**13.70**	480.77	10.60	10.96	1388.13
		AOBB-K*-b-DH	3	10.96	**39.67**	339.91	10.60	10.96	1388.13
		AOBB-K*-b	4	10.96	100.02	100.03	10.60	10.96	1388.13
	d21-3-1	AOBB-K*-b-UFO	3	**11.92**	**89.03**	628.59	9.37	11.72	551.27
		AOBB-K*-b-DH	3	**11.92**	**136.44**	1307.45	9.37	11.72	551.27
		AOBB-K*-b	4	**11.92**	**193.83**	196.52	9.37	11.72	551.27
	d25-3-1	AOBB-K*-b-UFO	3	**16.18**	**14.02**	64.82	10.74	13.65	880.46
		AOBB-K*-b-DH	3	**16.18**	**51.92**	80.22	10.74	13.65	880.46
		AOBB-K*-b	4	**16.18**	**166.74**	166.75	10.74	13.65	880.46

Redesign of 3 Residues

		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
3	d19-3-1	AOBB-K*-b-UFO	3	14.99	**6.15**	621.83	14.99	14.99	34.00
		AOBB-K*-b-DH	3	14.99	**11.31**	56.05	14.99	14.99	34.00
		AOBB-K*-b	4	14.99	75.99	76.00	14.99	14.99	34.00
	d20-3-1	AOBB-K*-b-UFO	3	10.96	**13.70**	480.77	10.60	10.96	1388.13
		AOBB-K*-b-DH	3	10.96	**39.67**	339.91	10.60	10.96	1388.13
		AOBB-K*-b	4	10.96	100.02	100.03	10.60	10.96	1388.13
	d21-3-1	AOBB-K*-b-UFO	3	11.92	**89.03**	628.59	9.37	11.72	551.27
		AOBB-K*-b-DH	3	11.92	**136.44**	1307.45	9.37	11.72	551.27
		AOBB-K*-b	4	11.92	193.83	196.52	9.37	11.72	551.27
	d25-3-1	AOBB-K*-b-UFO	3	16.18	**14.02**	64.82	10.74	13.65	880.46
		AOBB-K*-b-DH	3	16.18	**51.92**	80.22	10.74	13.65	880.46
		AOBB-K*-b	4	16.18	**166.74**	166.75	10.74	13.65	880.46

Redesign of 3 Residues

		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
3	d19-3-1	AOBB-K*-b-UFO	3	14.99	**6.15**	621.83	14.99	14.99	34.00
		AOBB-K*-b-DH	3	14.99	**11.31**	56.05	14.99	14.99	34.00
		AOBB-K*-b	4	14.99	75.99	76.00	14.99	14.99	34.00
	d20-3-1	AOBB-K*-b-UFO	3	10.96	**13.70**	480.77	10.60	10.96	1388.13
		AOBB-K*-b-DH	3	10.96	**39.67**	339.91	10.60	10.96	1388.13
		AOBB-K*-b	4	10.96	100.02	100.03	10.60	10.96	1388.13
	d21-3-1	AOBB-K*-b-UFO	3	11.92	**89.03**	628.59	9.37	11.72	551.27
		AOBB-K*-b-DH	3	11.92	**136.44**	1307.45	9.37	11.72	551.27
		AOBB-K*-b	4	11.92	193.83	196.52	9.37	11.72	551.27
	d25-3-1	AOBB-K*-b-UFO	3	16.18	**14.02**	64.82	10.74	13.65	880.46
		AOBB-K*-b-DH	3	16.18	**51.92**	80.22	10.74	13.65	880.46
		AOBB-K*-b	4	16.18	166.74	166.75	10.74	13.65	880.46

Redesign of 3 Residues

		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
3	d19-3-1	AOBB-K*-b-UFO	3	14.99	**6.15**	621.83	14.99	14.99	34.00
		AOBB-K*-b-DH	3	14.99	**11.31**	56.05	14.99	14.99	34.00
		AOBB-K*-b	4	14.99	75.99	76.00	14.99	14.99	34.00
	d20-3-1	AOBB-K*-b-UFO	3	10.96	**13.70**	480.77	10.60	10.96	1388.13
		AOBB-K*-b-DH	3	10.96	**39.67**	339.91	10.60	10.96	1388.13
		AOBB-K*-b	4	10.96	100.02	100.03	10.60	10.96	1388.13
	d21-3-1	AOBB-K*-b-UFO	3	11.92	**89.03**	628.59	9.37	11.72	551.27
		AOBB-K*-b-DH	3	11.92	**136.44**	1307.45	9.37	11.72	551.27
		AOBB-K*-b	4	11.92	193.83	196.52	9.37	11.72	551.27
	d25-3-1	AOBB-K*-b-UFO	3	16.18	**14.02**	64.82	10.74	13.65	880.46
		AOBB-K*-b-DH	3	16.18	**51.92**	80.22	10.74	13.65	880.46
		AOBB-K*-b	4	16.18	166.74	166.75	10.74	13.65	880.46

Redesign of 3 Residues

		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
3	d19-3-1	AOBB-K*-b-UFO	3	14.99	**6.15**	621.83	14.99	14.99	34.00
		AOBB-K*-b-DH	3	14.99	**11.31**	56.05	14.99	14.99	34.00
		AOBB-K*-b	4	14.99	75.99	76.00	14.99	14.99	34.00
	d20-3-1	AOBB-K*-b-UFO	3	10.96	**13.70**	480.77	10.60	10.96	1388.13
		AOBB-K*-b-DH	3	10.96	**39.67**	339.91	10.60	10.96	1388.13
		AOBB-K*-b	4	10.96	100.02	100.03	10.60	10.96	1388.13
	d21-3-1	AOBB-K*-b-UFO	3	11.92	**89.03**	628.59	9.37	11.72	551.27
		AOBB-K*-b-DH	3	11.92	**136.44**	1307.45	9.37	11.72	551.27
		AOBB-K*-b	4	11.92	**193.83**	196.52	9.37	11.72	551.27
	d25-3-1	AOBB-K*-b-UFO	3	16.18	**14.02**	64.82	10.74	13.65	880.46
		AOBB-K*-b-DH	3	16.18	**51.92**	80.22	10.74	13.65	880.46
		AOBB-K*-b	4	16.18	**166.74**	166.75	10.74	13.65	880.46

Redesign of 3 Residues

		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
3	d19-3-1	AOBB-K*-b-UFO	3	14.99	**6.15**	621.83	14.99	14.99	34.00
		AOBB-K*-b-DH	3	14.99	**11.31**	56.05	14.99	14.99	34.00
		AOBB-K*-b	4	14.99	75.99	76.00	14.99	14.99	34.00
	d20-3-1	AOBB-K*-b-UFO	3	10.96	**13.70**	480.77	10.60	10.96	1388.13
		AOBB-K*-b-DH	3	10.96	**39.67**	339.91	10.60	10.96	1388.13
		AOBB-K*-b	4	10.96	100.02	100.03	10.60	10.96	1388.13
	d21-3-1	AOBB-K*-b-UFO	3	**11.92**	**89.03**	628.59	9.37	11.72	551.27
		AOBB-K*-b-DH	3	**11.92**	**136.44**	1307.45	9.37	11.72	551.27
		AOBB-K*-b	4	**11.92**	**193.83**	196.52	9.37	11.72	551.27
	d25-3-1	AOBB-K*-b-UFO	3	**16.18**	**14.02**	64.82	10.74	13.65	880.46
		AOBB-K*-b-DH	3	**16.18**	**51.92**	80.22	10.74	13.65	880.46
		AOBB-K*-b	4	**16.18**	**166.74**	166.75	10.74	13.65	880.46

Redesign of 3 Residues

	AOBB-K*-b-[DH/UFO]							BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
3	d19-3-1	AOBB-K*-b-UFO	3	14.99	**6.15**	621.83	14.99	14.99	34.00
		AOBB-K*-b-DH	3	14.99	**11.31**	56.05	14.99	14.99	34.00
		AOBB-K*-b	4	14.99	75.99	76.00	14.99	14.99	34.00
	d20-3-1	AOBB-K*-b-UFO	3	10.96	**13.70**	480.77	10.60	10.96	1388.13
		AOBB-K*-b-DH	3	10.96	**39.67**	339.91	10.60	10.96	1388.13
		AOBB-K*-b	4	10.96	100.02	100.03	10.60	10.96	1388.13
	d21-3-1	AOBB-K*-b-UFO	3	11.92	**89.03**	628.59	9.37	11.72	551.27
		AOBB-K*-b-DH	3	11.92	**136.44**	1307.45	9.37	11.72	551.27
		AOBB-K*-b	4	11.92	**193.83**	196.52	9.37	11.72	551.27
	d25-3-1	AOBB-K*-b-UFO	3	16.18	**14.02**	64.82	10.74	13.65	880.46
		AOBB-K*-b-DH	3	16.18	**51.92**	80.22	10.74	13.65	880.46
		AOBB-K*-b	4	16.18	**166.74**	166.75	10.74	13.65	880.46

Redesign of 3 Residues

		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
3	d19-3-1	AOBB-K*-b-UFO	3	14.99	**6.15**	621.83	14.99	14.99	34.00
		AOBB-K*-b-DH	3	14.99	**11.31**	56.05	14.99	14.99	34.00
		AOBB-K*-b	4	14.99	75.99	76.00	14.99	14.99	34.00
	d20-3-1	AOBB-K*-b-UFO	3	10.96	**13.70**	480.77	10.60	10.96	1388.13
		AOBB-K*-b-DH	3	10.96	**39.67**	339.91	10.60	10.96	1388.13
		AOBB-K*-b	4	10.96	100.02	100.03	10.60	10.96	1388.13
	d21-3-1	AOBB-K*-b-UFO	3	**11.92**	**89.03**	628.59	9.37	11.72	551.27
		AOBB-K*-b-DH	3	**11.92**	**136.44**	1307.45	9.37	11.72	551.27
		AOBB-K*-b	4	**11.92**	**193.83**	196.52	9.37	11.72	551.27
	d25-3-1	AOBB-K*-b-UFO	3	**16.18**	**14.02**	64.82	10.74	13.65	880.46
		AOBB-K*-b-DH	3	**16.18**	**51.92**	80.22	10.74	13.65	880.46
		AOBB-K*-b	4	**16.18**	**166.74**	166.75	10.74	13.65	880.46

Redesign of 4 Residues

M	Problem	AOBB-K*-b-[DH/UFO] Algorithm	i	Soln	Anytime	Time	wt K*	BBK* Soln	Time
4	d7-4-2	AOBB-K*-b-UFO	3	**14.89**	3391.78	timeout	14.08	14.54	278.08
		AOBB-K*-b-DH	3	**14.49**	3543.27	timeout	14.08	14.54	278.08
		AOBB-K*-b	3	14.49	3293.62	timeout	14.08	14.54	278.08
	d13-4-1	AOBB-K*-b-UFO	3	15.03	**12.69**	1974.43	13.25	15.03	46.46
		AOBB-K*-b-DH	3	15.03	**22.05**	79.88	13.25	15.03	46.46
		AOBB-K*-b	4	15.03	165.48	timeout	13.25	15.03	46.46
	d17-4-1	AOBB-K*-b-UFO	3	10.86	**29.39**	timeout	10.52	10.80	89.94
		AOBB-K*-b	4	10.86	**657.54**	timeout	10.52	10.80	89.94
		AOBB-K*-b-DH	3	10.86	660.16	timeout	10.52	10.80	89.94
	d21-4-1	AOBB-K*-b-UFO	3	11.92	**196.30**	timeout	9.37	11.72	687.66
		AOBB-K*-b-DH	3	11.92	614.88	timeout	9.37	11.72	687.66
		AOBB-K*-b	4	11.72	264.92	timeout	9.37	11.72	687.66
	d43-4-1	AOBB-K*-b-UFO	3	18.19	76.49	484.69	18.04	18.18	119.88
		AOBB-K*-b-DH	3	18.19	**386.49**	timeout	18.04	18.18	119.88
		AOBB-K*-b	3	18.19	896.67	timeout	18.04	18.18	119.88
	d47-4-2	AOBB-K*-b-UFO	3	**22.87**	72.53	239.88	22.70	22.83	1339.15
		AOBB-K*-b	3	22.74	**130.95**	timeout	22.70	22.83	1339.15
		AOBB-K*-b-DH	3	22.74	140.66	timeout	22.70	22.83	1339.15

Redesign of 4 Residues

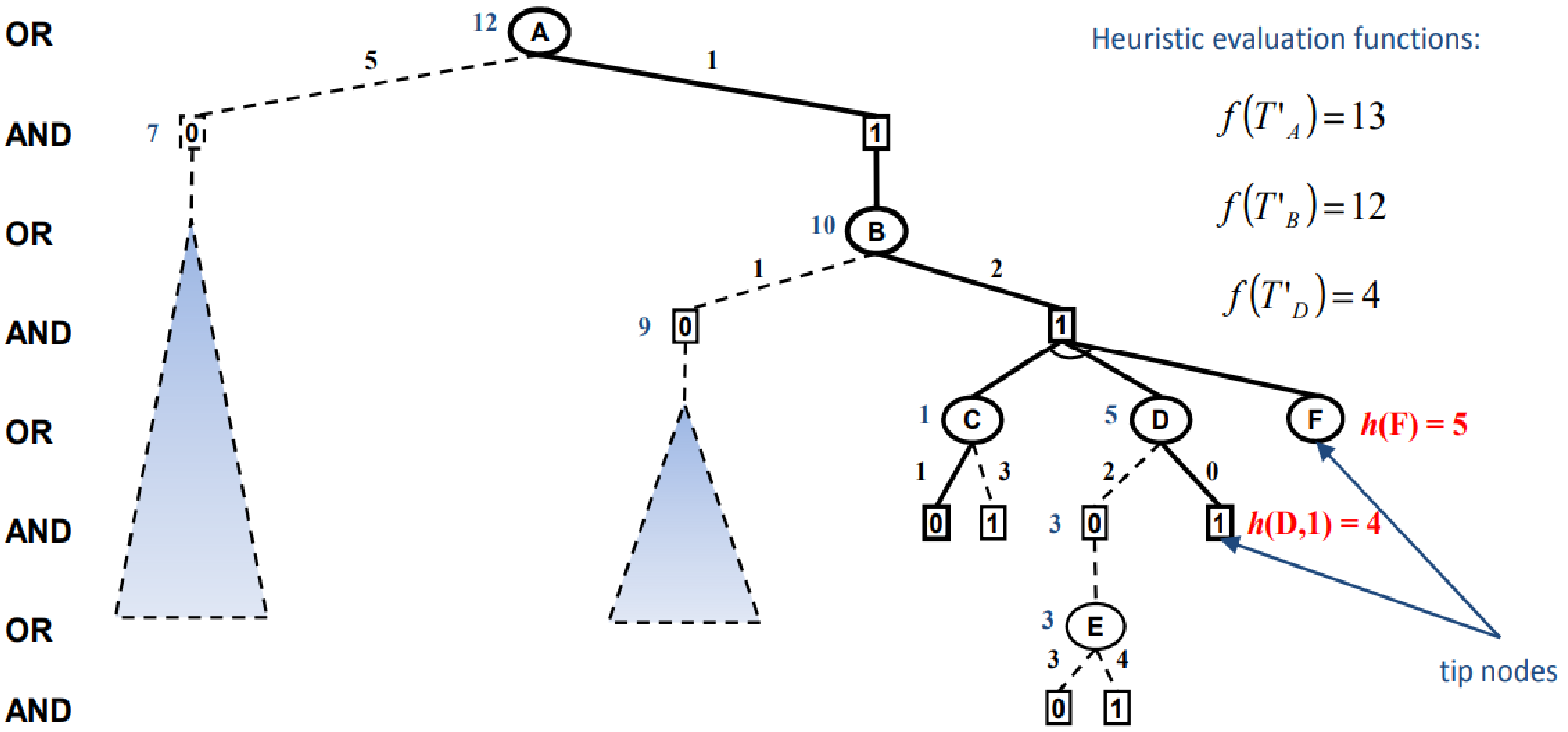
		AOBB-K*-b-[DH/UFO]						BBK*	
M	Problem	Algorithm	i	Soln	Anytime	Time	wt K*	Soln	Time
4	d7-4-2	AOBB-K*-b-UFO	3	**14.89**	3391.78	timeout	14.08	14.54	278.08
		AOBB-K*-b-DH	3	**14.49**	3543.27	timeout	14.08	14.54	278.08
		AOBB-K*-b	3	14.49	3293.62	timeout	14.08	14.54	278.08
	d13-4-1	AOBB-K*-b-UFO	3	15.03	**12.69**	1974.43	13.25	15.03	46.46
		AOBB-K*-b-DH	3	15.03	**22.05**	79.88	13.25	15.03	46.46
		AOBB-K*-b	4	15.03	165.48	timeout	13.25	15.03	46.46
	d17-4-1	AOBB-K*-b-UFO	3	10.86	**29.39**	timeout	10.52	10.80	89.94
		AOBB-K*-b	4	10.86	**657.54**	timeout	10.52	10.80	89.94
		AOBB-K*-b-DH	3	10.86	660.16	timeout	10.52	10.80	89.94
	d21-4-1	AOBB-K*-b-UFO	3	11.92	196.30	timeout	9.37	11.72	687.66
		AOBB-K*-b-DH	3	**11.92**	614.88	timeout	9.37	11.72	687.66
		AOBB-K*-b	4	11.72	264.92	timeout	9.37	11.72	687.66
	d43-4-1	AOBB-K*-b-UFO	3	18.19	76.49	484.69	18.04	18.18	119.88
		AOBB-K*-b-DH	3	18.19	**386.49**	timeout	18.04	18.18	119.88
		AOBB-K*-b	3	18.19	896.67	timeout	18.04	18.18	119.88
	d47-4-2	AOBB-K*-b-UFO	3	**22.87**	72.53	239.88	22.70	22.83	1339.15
		AOBB-K*-b	3	22.74	**130.95**	timeout	22.70	22.83	1339.15
		AOBB-K*-b-DH	3	22.74	140.66	timeout	22.70	22.83	1339.15

Cochrane 1977

This theorem leads to the following rules of conduct. In a given stratum, take a larger sample if

1. The stratum is larger.
2. The stratum is more variable internally.
3. Sampling is cheaper in the stratum.

AOBB-K*MAP K* Pruning Condition



Heuristic evaluation functions:

$f(T'_A) = 13$

$f(T'_B) = 12$

$f(T'_D) = 4$

AOBB-K*MAP
Subunit Stability Pruning Condition

key observation:

for any node n corresponding to subunit X, the progressively improving $ub(Z_x)$ can be computed via the expression:

$$ub(Z_x) = g(n)\, ub_x(n) \cdot \underbrace{r_x(n)}_{\text{MMAP ancestor branching factor}} + \underbrace{sum_x(n)}_{\text{summation term } (n \in SUM)}$$

which can be computed using information from n, $par(n)$, and $sib_x(n)$

AOBB-K*MAP
Subunit Stability Pruning Condition

$UB = UB_x \cdot C_x + S_x$

for each subunit

$UB_R,\ C_R : 1,\ S_R : 0$

$UB_{R_0},\ C_{R_0} : C_R,\ S_{R_0} : S_R$

$UB_A,\ C_A : C_{R_0} \cdot UB_B,\ S_A : S_{R_0}$

$UB_{A_0},\ C_{A_0} : C_A,\ S_{A_0} : S_A$

$UB_C,\ C_C : C_{A_0} \cdot UB_D,\ S_C : S_{A_0}$

$UB_E,\ C_E : C_C,\ S_E : S_A + (UB_C - C_C UB_E)$

$UB_G,\ C_G : C_E \cdot UB_H,\ S_G : S_E$

$UB_K,\ C_K : C_G,\ S_K : S_G + (UB_G - C_G UB_K)$

HK*

H_{UBsub1}

H_{UBsub2}