Modern Exact and Approximate MAP algorithms for Graphical Models

In the pursuit of universal solver

Rina Dechter
Donald Bren school of ICS, University of California, Irvine

TTI, May/2014
Sample Applications for Graphical Models

Computer Vision

Genetic Linkage

Sensor Networks

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

Learning: MAP

Reasoning: MAP

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

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

- Graphical models, Queries
- Inference Algorithms
- AND/OR search
- Optimistic bounding schemes (mini-bucket, re-parameterization, cost-shifting, soft AC)
- BRAOB: anytime DFS for AND/OR
- Experiments/competitions (putting it all together)
- Recent work: Parallelism, m-best, weighted best-first, marginal map, tree-SLS
- Conclusion
Graphical Models, Queries, Algorithms
Graphical Models

- A graphical model \( M = (X,D,F,\otimes) \)
  - \( X = \{X\downarrow 1, \ldots, X\downarrow n\} \) variables
  - \( D = \{D\downarrow 1, \ldots, D\downarrow n\} \) domains
  - \( F = \{f\downarrow 1, \ldots, f\downarrow t\} \) functions over \( \{S\downarrow 1, \ldots, S\downarrow t\} \)
  - Global function \( \otimes \) Combine

\[
F(X) = \bigotimes f \in F f(x)
\]

- Reasoning (queries) \( \downarrow \) marginalize

\[
G(Z) = \downarrow X - Z F(X) = \downarrow X - Z \bigotimes f \in F f(x)
\]

Queries:

- Belief updating: \( \Sigma_{x,y} \Pi_j p_i \)
- MPE: \( \max_x \Pi_j p_j \)
- CSP: \( \Pi_x \times_j c_j \)
- Max-CSP: \( \min_x \Sigma_j F_j \)

When combine and marginalize obey

Some properties they can be solved

By the same algorithms

(Bistareli, Rossi and Montanari, 1995, Shenoy, Shafer, 1990, Kask et. al., 2005.)
Monitoring Intensive-Care Patients

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

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

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

$$MAP = \max_x \prod_i p(x_i \mid parents(x_i))$$
Queries

- **Optimization Queries**: MAP/MPE queries:
  \[ x_{AB}^* = \arg \min_{x_A, x_B} \sum_{x_\alpha} \varphi_\alpha \quad x_{AB}^* = \arg \max_{x_A, x_B} \prod_{x_\alpha} \varphi_\alpha \]

- **Likelihood queries**: (counting, partition function, marginal, probability of evidence)
  \[ Z = \sum_{x_A, x_B} \prod_{x_\alpha} \varphi_\alpha \]

- **Marginal MAP**:
  - **Marginalize (sum) away variables** \( A \), then **find optimal configuration of variables** \( B \)
  \[ x_{B}^* = \arg \max_{x_B} \sum_{x_A} \prod_{\alpha} \psi(x_\alpha) \]

*Also satisfiability and expected utility*
Tree-solving is easy

Belief updating
(sum-prod)

CSP – consistency
(projection-join)

MPE (max-prod)

#CSP (sum-prod)

Trees are processed in linear time and memory
Inference vs conditioning-search

Inference

\[\exp(w^*) \text{ time/space}\]

Search

\[\text{Exp}(n) \text{ time} \quad O(n) \text{ space}\]

Search+inference:

Space: \(\exp(w)\)  \(w: \text{user}\)

Time: \(\exp(w+c(w))\)  \(w: \text{controlled}\)
Inference vs conditioning-search

**Inference**

$\exp(w^*)$ time/space

**Search**

$\exp(w^*)$ time

$O(w^*)$ space

Context minimal AND/OR search graph

18 AND nodes

Search+inference:

Space: $\exp(q)$

Time: $\exp(q+c(q))$

$q$: user controlled
• Solving MAP by Inference:
• Non-serial Dynamic programming
• The induced-width/treewidth
Finding MAP by Bucket Elimination

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

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

bucket B: \( P(b \mid a) \cdot P(d \mid b, a) \cdot P(e \mid b, c) \)

bucket C:

bucket D:

bucket E:

bucket A:

\[ h^B(a, d, c, e) \]
Finding MAP by Bucket Elimination

Algorithm BE-mpe \( \text{(Dechter 1996, Bertele and Briochi, 1977)} \)

\[
MPE = \max_{a,e,d,c,b} P(a)P(c \mid a)P(b \mid a)
\]

\[
\max_{X} \prod
\]

bucket B: 

\[
P(b \mid a) \quad P(d \mid b, a) \quad P(e \mid b, c)
\]

bucket C: 

\[
P(c \mid a)
\]

bucket D: 

\[
h^C(a, d, e)
\]

bucket E: 

\[
e=0 \quad h^D(a, e)
\]

bucket A: 

\[
P(a) \quad h^E(a)
\]

\[
\text{OPT}
\]

\[
W^* = 4
\]

"induced width" 
(max clique size)

ITA, 2/14/2013
Generating the MPE-tuple

1. \( a' = \text{arg max}_a P(a) \cdot h^E(a) \)
2. \( e' = 0 \)
3. \( d' = \text{arg max}_d h^C(a', d, e') \)
4. \( c' = \text{arg max}_c P(c | a') \times h^B(a', d', c, e') \)
5. \( b' = \text{arg max}_b \frac{P(b | a') \times P(d' | b, a') \times P(e' | b, c')}{P(b | a)} \times P(d | b, a) \times P(e | b, c) \)

\[ B: \quad P(b|a) \quad P(d|b,a) \quad P(e|b,c) \]
\[ C: \quad P(c|a) \quad h^B(a,d,c,e) \]
\[ D: \quad h^C(a,d,e) \]
\[ E: \quad e=0 \quad h^D(a,e) \]
\[ A: \quad P(a) \quad h^E(a) \]

Return \((a', b', c', d', e')\)
Generating the MPE-tuple

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

2. $e' = 0$

3. $c' = \max_a \arg \max_{b'} P(c' | b', a')$

4. $d' = \max_{b'} P(d' | b', a') \times P(e' | b', a', c')$

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

$\mathbf{B}: P(b | a) \quad P(d | b, a) \quad P(e | b, c)$

$\mathbf{A}: P(a) \quad h^E(a)$

$\mathbf{C}: P(a)$

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

Time and space exponential in the induced-width / treewidth

$O(n_{k \uparrow \omega^* + 1})$
The Induced-width/treewidth

\[ w^*(d) \] – the induced width of a graph along ordering \( d \)

The effect of the ordering:

"Moral" graph

\[ w^*(d_1) = 4 \]

\[ w^*(d_2) = 2 \]
Bucket Elimination

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

**Messages**

Finding max

**Assignment**

argmax

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

*The AND/OR Search graph*
Classic OR Search Space

Ordering: A B E C D F
AND/OR Search Space

Primal graph

DFS tree
AND/OR vs. OR Spaces

Time $O(nk^h)$
Space $O(n)$

Height is bounded by $(\log n) w^*$

54 nodes
AND/OR vs. OR

AND/OR size: exp(4),
OR size exp(6)
DFS algorithm (#CSP example)  

Value of node = number of solutions below it

OR node: Marginalization operator (summation)

AND node: Combination operator (product)
AND/OR Tree Search for COP

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

**AND node** = Combination operator (summation)

**OR node** = Marginalization operator (minimization)
From Search Trees to Search Graphs

- Any two nodes that root identical sub-trees or sub-graphs can be merged
From Search Trees to Search Graphs

- Any two nodes that root identical sub-trees or sub-graphs can be merged
From AND/OR Tree
An AND/OR Graph
How Big Is The Context?

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

\[ \text{max context size} = \text{treewidth} \]
All Four Search Spaces

<table>
<thead>
<tr>
<th></th>
<th>AND/OR graph</th>
<th>OR graph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space</td>
<td>$O(n , k^{w*})$</td>
<td>$O(n , k^{pw*})$</td>
</tr>
<tr>
<td>Time</td>
<td>$O(n , k^{w*})$</td>
<td>$O(n , k^{pw*})$</td>
</tr>
</tbody>
</table>

Computes any query:
- Constraint satisfaction
- Optimization
- Weighted counting

Any query is best computed Over the c-minimal AO space
The impact of the pseudo-tree

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

Min-Fill (Kjaerulff90)

(W=5,h=6)

(W=4,h=8)
The Effect of Constraint Propagation

Domains are \{1,2,3,4\}

CONSTRANTS ONLY

FORWARD CHECKING

MAINTAINING ARC CONSISTENCY
Basic Heuristic Search Schemes

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

1. **Branch-and-Bound**
   - Use heuristic function \( f(x^p) \) to prune the depth-first search tree
   - Linear space

2. **Best-First Search**
   - Always expand the node with the highest heuristic value
   - \( f(x^p) \) needs lots of memory
Bounding approximations:
• The mini-bucket scheme
• The cost-shifting or re-parameterization scheme
• Combining the two
Two Bounding Schemes

- **Goal:** bound $\min_x \sum_i f_i(x)$ or $\max_x \prod_i f_i(x)$

- **Mini-bucket; Node duplication control:**

- **Reparameterization schemes:**
  - Soft arc-consistency (Bistareli, 2000, Sciex 2000)

- Belief Propagation can be viewed as re-parameterization
Mini-bucket Approximation
(Dechter and Rish, 1997, 2003)

Split a bucket into mini-buckets => bound complexity

\[
\text{bucket}(X) = \{ h_1, \ldots, h_r, h_{r+1}, \ldots, h_n \}
\]

\[
h^X = \max_X \prod_{i=1}^{n} h_i
\]

\[
g^X = (\max_X \prod_{i=1}^{r} h_i) \cdot (\max_X \prod_{i=r+1}^{n} h_i)
\]

\[
h^X \leq g^X
\]

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

\[ \max_B \prod P(E|B,C) \]

Bucket B

\[ P(C|A) h^B (C,E) \]

Bucket C

\[ h^B (A,D) \]

Bucket D

\[ E = 0 h^C (A,E) \]

Bucket E

\[ P(A) h^E (A) h^D (A) \]

Bucket A

\[ \text{W=2} \]

MPE* is an upper bound on MPE --U

Generating a solution yields a lower bound--L

Node duplication, renaming

P(A)

P(B|A)

P(C|A)

P(E|B,C)

P(D|A,B)

P(E|B,C)

W=2

\[ \max_B \prod \]
Properties of Mini-Bucket Elimination

- **Complexity**: $O(r \exp(i))$ time and $O(\exp(i))$ space.
- **Accuracy**: determined by upper/lower (U/L) bound.
- As $i$ increases, both accuracy and complexity increase.
- Possible use of mini-bucket approximations:
  - As anytime algorithms
  - As heuristics in search
Outline

- Graphical models and the MAP task
- Bounding approximations
  - The mini-bucket scheme
  - The cost-shifting or re-parameterization scheme
  - Combining the two
- New Algorithms combinations
- Experiments
Tightening bounds via cost-shifting

$$\max_x \sum_{ij} E_{ij}(x_i, x_j)$$
Tightening bounds via cost-shifting

\[ \max_{x} \sum_{i,j} E_{ij}(x_i, x_j) \leq \sum_{i,j} \max_{x} E_{ij}(x_i, x_j). \]

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

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

\[
\max X \sum_{ij} E_{ij}(x_i, x_j) \leq \min \lambda \sum \max X E_{ij}(x_i, x_j) + \lambda_i(x_i) + \lambda_j(x_j)
\]

\[
\forall i \sum_j \lambda_{ij}(x_i) = 0
\]
Decomposition view

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

\[
\max_x \sum_{ij} E_{ij}(x_i, x_j) \leq \min_x \sum_{ij} \max_x E_{ij}(x_i, x_j) + \lambda_{ij}(x_i) + \lambda_{ji}(x_j)
\]

Same bound by different names

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

- Can use any decomposition updates
  - (message passing, subgradient, augmented, etc.)

- **FGLP**: Update the original factors

- **JGLP**: Update clique function of the join graph

- **MBE-MM**: Update within each bucket only
Factor graph Linear Programming

- Update the original factors (FGLP)
  - Tighten all factors over $x_i$ simultaneously
  - Compute max-marginals $\forall \alpha, \gamma_\alpha(x_i) = \max_{x_\alpha \setminus x_i} f_\alpha$
  - & update:
    $$\forall \alpha, f_\alpha(x_\alpha) \leftarrow f_\alpha(x_\alpha) - \gamma_\alpha(x_i) + \frac{1}{|F_i|} \sum_\beta \gamma_\beta(x_i)$$
Mini-bucket + fixed-point updates

- **JGLP**: Update clique function of the join graph
  - Use MBE to generate the join graph
  - Define function $F_i$ for each clique (mini-bucket) $q^i$
  - Update each edge over separator set

\[
F_i \leftarrow F_i + \frac{1}{2}(\gamma_j(x_s) - \gamma_i(x_s))
\]

- **MBE-MM**: Mini-bucket with moment matching
  - Apply cost-shifting within each bucket only
Join Graph Linear Programming (JGLP(i))

\[ \max_B \Pi \]

Bucket B

**P(E|B,C)**

Bucket C

**P(C|A)**  
**h_B(C,E)**

Bucket D

**P(D|A,B)**  
**P(B|A)**  
**h_B(A,D)**

**E = 0**  
**h_C(A,E)**

Bucket E

**P(A)**  
**h_E(A)**  
**h_D(A)**

Bucket A

**MB defines**  
**A Join Graph**
Iterative tightening as bounding schemes

pedigree9, n=1119, k=5, w=25, h=123, z=10

pedigree41, n=885, k=5, w=33, h=100, z=10

90-30-5, n=900, k=2, w=42, h=151, z=10

90-34-5, n=1156, k=2, w=48, h=186, z=10
Anytime AND/OR Branch and Bound (BRAOBB)
+ MBE/JGLP(i)
Basic Heuristic Search Schemes

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

1. **Branch-and-Bound**
   Use heuristic function \( f(x^p) \) to prune the depth-first search tree
   
   Linear space

2. **Best-First Search**
   Always expand the node with the highest heuristic value
   \( f(x^p) \) needs lots of memory

\[ f \leq L \]
Mini-bucket Heuristics for BB search

\[ f(a, e, D) = P(a) \cdot h^B(D, a) \cdot h^C(e, a) \]
**AND/OR Branch-and-Bound**

- Problem decomposition and caching.

  *Mini-bucket heuristic for pruning.*

---

**Problem decomposition**

**Guided by pseudo tree**

**Caching**
MAP by AND/OR Branch-and-Bound

Decomposition of independent subproblems

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

Cache table for F (independent of A)

[B | E | cost]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>..</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>..</td>
</tr>
</tbody>
</table>

[Marinescu & Dechter, AIJ'09]
Anytime Performance

- OR Branch-and-Bound is anytime.
- But AND/OR breaks anytime behavior of depth-first scheme:
  - First anytime solution delayed until last subproblem starts processing.
  - Very bad anytime behavior for unbalanced subproblem spaces.
    - One complex subproblem early on delays everything.

[Otten & Dechter, AlCom '12]
Anytime Performance

- **Breadth-Rotating AOBB:**
  - Take turns processing subproblems, rotate on:
    - Child subproblem branching.
    - Subproblem solved.
    - Node expansion limit reached.
  - Process each subproblem depth-first.
    - Maintain favorable complexity bounds: linear number of nodes on stack(s).

[Otten & Dechter, AlCom '12]
Stochastic Variable Orderings

- AOBBC complexity: $O(nk^w)$
  - High variance in width of orderings.
- Our implementation:
  - Minfill heuristic.
  - Random tie-breaking.
  - Allow deviation from heuristic optimum.
  - Highly optimized data structures, early termination.
    - Can do many thousands of iterations.

[Kask, Gelfand, Otten, Dechter, AAAI '11]
Empirical Evaluation: Pascal 2011 Competition Comparing with Gurobi
Iterative tightening as heuristic generators

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

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

Time bound – 24 h

pedigree31 (n=1183 k=5 w=30 h=85) i-bound=10

pedigree31 (n=1183 k=5 w=30 h=85) i-bound=15
Empirical Evaluation: Haplotype problems

Pedigree 9 (n=1118 k=7 w=27 h=100) t-bound=10

Time bound – 24 h
DAOOPT: Improving AND/OR Branch-and-Bound for Graphical Models

Lars Otten, Alexander Ihler, Kalev Kask, Rina Dechter

Dept. of Computer Science
University of California, Irvine

ITA, 2/14/2013
State-of-the-art AOB Search

Problem decomposition

Enhanced Variable Ordering Schemes
Highly efficient, stochastic minfill / mindegree implementations for lower-width orderings.

Breadth-First Subproblem Rotation
Improved anytime performance through interleaved processing of independent subproblems.

Mini-Buckets Cost-shifting (MPLP)
Re-parametrization
Tighter bounds by iteratively solving linear programming relaxations and message passing on join graph.
Putting It All Together

- Three time limits: 20 sec, 20 min, 1h.
  - 4 GB memory limit in each case.
- Chose different parameter sets through experimentation.

<table>
<thead>
<tr>
<th></th>
<th>20 seconds</th>
<th>20 minutes</th>
<th>1 hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. MPLP in input graph</td>
<td>2 sec</td>
<td>30 sec / 500 iter</td>
<td>60 sec / 2000 iter</td>
</tr>
<tr>
<td>2. Stochastic Local Search</td>
<td>2x 2 sec</td>
<td>10x 6 sec</td>
<td>20x 10 sec</td>
</tr>
<tr>
<td>3. Iterative Variable Ordering</td>
<td>3 sec / 500 iter</td>
<td>60 sec / 10K iter</td>
<td>180 sec / 30K iter</td>
</tr>
<tr>
<td>4. MPLP on join graph (JGLP)</td>
<td>2 sec</td>
<td>30 sec / 250 iter</td>
<td>60 sec / 1K iter</td>
</tr>
<tr>
<td>5. Mini-buckets + MM</td>
<td>i=15, max.125MB</td>
<td>i=25, max.4GB</td>
<td>i=35, max.4GB</td>
</tr>
<tr>
<td>6. Full BRAOBB</td>
<td>To completion or until timeout</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Competing Solvers

• INRA Toulouse, variants of *Toulbar2*:
  • All methods employ soft local arc consistency.
  • dfbvbemcs:
    • Variable elimination preprocessing, depth-first BaB.
  • ficolofo & vns/lds+cp:
    • Initial greedy search or limited discrepancy search, resp.
    • Interleave variable neighborhood search and exact depth-first BaB, limited to varying subspaces.

• Not as competitive, details/authors unknown:
  • Vanilla-MPLP, nutcracker
Competition Results

- 20 sec, 20 min, 1 hour categories
  - Score computed relative to a baseline/BP solution.
  - \( \text{E}(x) = -\sum \log f(x) \)
  - \( \text{Score}(x\uparrow s) = E(x\uparrow s) - \min\{E(x\uparrow bp), E(x\uparrow df)\} \)
  - **1st place** in all three categories!

<table>
<thead>
<tr>
<th>Category</th>
<th>20 sec</th>
<th>20 min</th>
<th>1 hour</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>daoopt</td>
<td>ficofo</td>
<td>dfbbvecms</td>
</tr>
<tr>
<td>CSP</td>
<td>-0.9123</td>
<td>-0.8669</td>
<td>-0.8669</td>
</tr>
<tr>
<td>Deep belief nets</td>
<td>-0.3403</td>
<td>-0.3210</td>
<td>-0.3174</td>
</tr>
<tr>
<td>Grids</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Image alignment</td>
<td>-0.0028</td>
<td>-0.0046</td>
<td>-0.0460</td>
</tr>
<tr>
<td>Medical diagnosis</td>
<td>-4.8201</td>
<td>-4.8287</td>
<td>-4.8023</td>
</tr>
<tr>
<td>Object detection</td>
<td>-0.0308</td>
<td>-0.0308</td>
<td>-0.0308</td>
</tr>
<tr>
<td>Protein folding</td>
<td>-0.0308</td>
<td>-0.0308</td>
<td>-0.0308</td>
</tr>
<tr>
<td>Prot/prot inter.</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Relational</td>
<td>-0.0300</td>
<td>-0.0300</td>
<td>-0.0298</td>
</tr>
</tbody>
</table>
Competition:
Gurobi vs AOBB on Pascal2 instances (Junkyu and Lam, ongoing)
### Benchmark

<table>
<thead>
<tr>
<th>Problem</th>
<th>PEDIGREE</th>
<th>WCSP</th>
<th>Protein Folding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total #. Instances</td>
<td>22 From UAI'08 Competition</td>
<td>61 From PASCAL2Competition</td>
<td>10 From PASCAL2Competition</td>
</tr>
<tr>
<td>Compared Instances</td>
<td>22</td>
<td>15 Exclude 46 Indeterminate Cases*</td>
<td>7 Exclude 3 Indeterminate Cases*</td>
</tr>
<tr>
<td>BRAOBB (G.M of Time)</td>
<td>51.07 sec G.M of 20 instances terminated for both</td>
<td>15.96 sec G. M of 13 instances terminated for both</td>
<td>1004.25 sec G. M of 7 instances, terminated for AOBB</td>
</tr>
<tr>
<td>GUROBI (G.M of Time)</td>
<td>7.15 sec G.M of 20 instances terminated for both</td>
<td>72.65 sec G. M of 13 instances terminated for both</td>
<td>NA 9/10 Memory Out</td>
</tr>
<tr>
<td>BRAOBB VS. GUROBI</td>
<td>0 VS. 22</td>
<td>12 VS. 3</td>
<td>6 Vs. 1 AOBB won by memory out of Gurobi</td>
</tr>
</tbody>
</table>

**Indeterminate Cases:** G.M = geometric mean

- Both Time out, or BRAOBB Time out and Gurobi Memory out (4GB),
- One of the solver time out earlier than the other’s running time
- MPE to 0/1 ILP Conversion was not available for experiments

**Link to more results**
<table>
<thead>
<tr>
<th>name</th>
<th>(n,f,k,s,i,w,h)</th>
<th>Gurobi Time</th>
<th>Tg/Ta</th>
<th>AOBB Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SPOT5 1502 209,412,4,3,6,6,15</td>
<td>1.72</td>
<td>24.57%</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>SPOT5 29 82,463,4,2,14,14,24</td>
<td>25.31</td>
<td>76.70%</td>
<td>33</td>
</tr>
<tr>
<td>3</td>
<td>SPOT5 404 100,711,4,3,19,19,45</td>
<td>310.58</td>
<td>2218.43%</td>
<td>14</td>
</tr>
<tr>
<td>4</td>
<td>SPOT5 503 143,636,4,3,9,9,44</td>
<td>622.49</td>
<td>8892.71%</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>SPOT5 54 67,272,4,3,11,11,19</td>
<td>28.15</td>
<td>402.14%</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>SPOT5 42 190,1395,4,3,13,26,87</td>
<td>3600 (TO)</td>
<td>NA</td>
<td>289</td>
</tr>
<tr>
<td>7</td>
<td>bwt3ac 45,686,11,2,12,16,27</td>
<td>164.4</td>
<td>342.50%</td>
<td>48</td>
</tr>
<tr>
<td>8</td>
<td>bwt4ac 179,7110,18,2,7,42,90</td>
<td>(MO)</td>
<td>NA</td>
<td>697</td>
</tr>
<tr>
<td>9</td>
<td>driverlog01ac 71,619,4,2,9,9,30</td>
<td>7.74</td>
<td>96.75%</td>
<td>8</td>
</tr>
<tr>
<td>10</td>
<td>GEOM30a_3 30,82,3,2,6,6,15</td>
<td>42.42</td>
<td>707.00%</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>GEOM30a_4 30,82,4,2,6,6,15</td>
<td>6.76</td>
<td>112.67%</td>
<td>6</td>
</tr>
<tr>
<td>12</td>
<td>GEOM30a_5 30,82,5,2,6,6,15</td>
<td>6.69</td>
<td>111.50%</td>
<td>6</td>
</tr>
<tr>
<td>13</td>
<td>myciel5g_3 47,237,3,2,14,21,24</td>
<td>1012.8</td>
<td>1875.56%</td>
<td>54</td>
</tr>
<tr>
<td>14</td>
<td>queen5 _5_3 25,161,3,2,15,18,20</td>
<td>2744.57</td>
<td>8072.26%</td>
<td>34</td>
</tr>
<tr>
<td>15</td>
<td>queen5 _5_4 25,161,4,2,12,18,20</td>
<td>977.24</td>
<td>534.01%</td>
<td>183</td>
</tr>
</tbody>
</table>

**BRAOBB** : Tout 60 min (MBE-MM, MPLP 2sec, JGLP 2 sec, SLS 2x2 sec, Ordering from Kalev’s Code 3sec)

**GUROBI** : Tout 60 min (Default, Single Processor, Dual Simplex at the Root)

**Memory Limit** : Both 4 GB

**Indeterminate Cases Total 46 (Not presented Above)**

✓ Both Time Out : 5/46
✓ AOBB Time Out, Gurobi Memory out 15/46
✓ ILP Conversion NA : 26/46
<table>
<thead>
<tr>
<th>name</th>
<th>Gurobi Time</th>
<th>Tg/Ta</th>
<th>AOBB Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>pdb1b25</td>
<td>(MO)</td>
<td>NA</td>
<td>21600 (TO)</td>
</tr>
<tr>
<td>pdb1d2e</td>
<td>(MO)</td>
<td>NA</td>
<td>557</td>
</tr>
<tr>
<td>pdb1fmj</td>
<td>(MO)</td>
<td>NA</td>
<td>21600 (TO)</td>
</tr>
<tr>
<td>pdb1i24</td>
<td>78.54</td>
<td>21.76%</td>
<td>361</td>
</tr>
<tr>
<td>pdb1iqc</td>
<td>(MO)</td>
<td>NA</td>
<td>1801</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>name</th>
<th>Gurobi Time</th>
<th>Tg/Ta</th>
<th>AOBB Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>pdb1jmx</td>
<td>(MO)</td>
<td>NA</td>
<td>21600 (TO)</td>
</tr>
<tr>
<td>pdb1kgn</td>
<td>(MO)</td>
<td>NA</td>
<td>536</td>
</tr>
<tr>
<td>pdb1kwh</td>
<td>(MO)</td>
<td>NA</td>
<td>16638</td>
</tr>
<tr>
<td>pdb1m3y</td>
<td>(MO)</td>
<td>NA</td>
<td>647</td>
</tr>
<tr>
<td>pdb1qks</td>
<td>(MO)</td>
<td>NA</td>
<td>493</td>
</tr>
</tbody>
</table>

**BRAOBB**: Tout 6Hr (MBE-MM, MPLP 60 sec, JGLP 60 sec, SLS 20x10 sec, Ordering from Kalev’s Code 3min)
**GUROBI**: Tout 6Hr (Default, Single Processor, Dual Simplex at the Root)
**Memory Limit**: Both 4 GB

*Indeterminate Cases Total 3*

✓ AOBB Time Out, Gurobi Memory Out 3
### Gurobi Time

<table>
<thead>
<tr>
<th>Pedigree</th>
<th>Gurobi Time</th>
<th>Tg/Ta</th>
<th>AOBB Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pedigree1</td>
<td>0.7</td>
<td>10.00%</td>
<td>7</td>
</tr>
<tr>
<td>Pedigree13</td>
<td>15.8</td>
<td>2.33%</td>
<td>679</td>
</tr>
<tr>
<td>Pedigree18</td>
<td>6.45</td>
<td>53.75%</td>
<td>12</td>
</tr>
<tr>
<td>Pedigree19</td>
<td>173.66</td>
<td>NA</td>
<td>1800 (TO)</td>
</tr>
<tr>
<td>Pedigree20</td>
<td>11.81</td>
<td>18.75%</td>
<td>63</td>
</tr>
<tr>
<td>Pedigree23</td>
<td>2.72</td>
<td>4.77%</td>
<td>57</td>
</tr>
<tr>
<td>Pedigree25</td>
<td>3.1</td>
<td>16.32%</td>
<td>19</td>
</tr>
<tr>
<td>Pedigree30</td>
<td>7.25</td>
<td>55.77%</td>
<td>13</td>
</tr>
<tr>
<td>Pedigree31</td>
<td>20.32</td>
<td>18.81%</td>
<td>108</td>
</tr>
<tr>
<td>Pedigree33</td>
<td>4.12</td>
<td>9.36%</td>
<td>44</td>
</tr>
<tr>
<td>Pedigree34</td>
<td>40.06</td>
<td>41.30%</td>
<td>97</td>
</tr>
</tbody>
</table>

### AOBBB Time

<table>
<thead>
<tr>
<th>Pedigree</th>
<th>Gurobi Time</th>
<th>Tg/Ta</th>
<th>AOBB Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pedigree1</td>
<td>3.09</td>
<td>11.44%</td>
<td>27</td>
</tr>
<tr>
<td>Pedigree13</td>
<td>5.99</td>
<td>13.31%</td>
<td>45</td>
</tr>
<tr>
<td>Pedigree18</td>
<td>6.01</td>
<td>30.05%</td>
<td>20</td>
</tr>
<tr>
<td>Pedigree19</td>
<td>221.15</td>
<td>NA</td>
<td>1800 (TO)</td>
</tr>
<tr>
<td>Pedigree20</td>
<td>24.5</td>
<td>8.75%</td>
<td>280</td>
</tr>
<tr>
<td>Pedigree23</td>
<td>2.36</td>
<td>3.58%</td>
<td>66</td>
</tr>
<tr>
<td>Pedigree25</td>
<td>9.83</td>
<td>18.90%</td>
<td>52</td>
</tr>
<tr>
<td>Pedigree30</td>
<td>12.05</td>
<td>48.20%</td>
<td>25</td>
</tr>
<tr>
<td>Pedigree31</td>
<td>14.64</td>
<td>3.30%</td>
<td>443</td>
</tr>
<tr>
<td>Pedigree33</td>
<td>8.01</td>
<td>10.27%</td>
<td>78</td>
</tr>
<tr>
<td>Pedigree34</td>
<td>7.85</td>
<td>25.32%</td>
<td>31</td>
</tr>
</tbody>
</table>

**BRAOBB** : Tout 30min (MBE-MM, MPLP 2ec, JGLP 2 sec, SLS 2x2 sec, Ordering from Kalev’s Code 3sec)

**GUROBI** : Tout 30min (Default, Single Processor, Dual Simplex at the Root)

**Memory Limit** : Both 4 GB
Recent work:

- Parallelism AOBB,
- Using External memory
- M-best search
- Weighted Best-first
- Marginal MAP
- STLS- Tree-based SLS
A New Algorithm for Marginal MAP

- (Submitted to UAI-2014) Improving Marginal Map for Graphical Models
- Radu Marinescu, Rina Dechter, Alex Ihler.

• Problem: \[ x_B^* = \arg \max_{x_B} \sum_{x_A} \prod_{\alpha} \psi(x_{\alpha}) \]

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

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

- Branch and Bound Search of AND/OR search

Figure 2: AND/OR search spaces for marginal MAP

Figure 5: Number of instances solved (top) and number of wins (bottom) by benchmark.
Recent work:
• Parallelism AOBB,
• Using External memory
• M-best search
• Weighted Best-first
• Marginal MAP
• STLS- Tree-based SLS
Weighted AND/OR Search

Paper submitted: “Evaluating Weighted DFS Branch and Bound over Graphical Models” Natalia Flerova, Radu Marinescu, Rina Dechter

• Empirically evaluation proposed algorithms wAOBB and wBRAOBB against Weighted Best-First search (wAOBB) and Breadth-First AND/OR Branch and Bound (BRAOBB)
Conclusion

- Search+bounded-Inference lead to effective anytime schemes and effective approximations, with bounding guarantees.
- Heuristic evaluation function can be improved
- Static/dynamic heuristic, during search with dynamic variable-ordering (tradeoff of time overhead against pruning).
- Portfolio approach
UCI Library: Summary

- **Exact/anytime:**
  - **Likelihood:** BE, BEEM, VEC(w), AOlibPE(c-bound)
  - **MAP:** VE, BEEM (external memory/multi-core), AOBB(i), BRAOBB(i), DAOOPT(Distributed AOBB).
  - Marginal Map (currently developed)

- **Approximation/anytime, for all queries:**
  - BP, IJGP(i-bound)
  - IJGP-Importance Sampling(i-bound)
  - IJGP-SampleSearch(i-bound)
  - MBE (mini-bucket), Weighted-mini-bucket, reparameterized MB
  - STLS (currently developed, for MAP)

- **Supporting schemes:** Variable-ordering (IGVO)
For publication see:
http://www.ics.uci.edu/~dechter/publications.html

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