Agenda

- Mini-bucket elimination
- Mini-clustering
- Iterative Belief propagation
- Iterative-join-graph propagation
Cluster Tree Elimination - properties

- Correctness and completeness: Algorithm CTE is correct, i.e. it computes the exact joint probability of a single variable and the evidence.

- Time complexity: $O(\text{deg} \times (n+N) \times d^{w^*+1})$

- Space complexity: $O(\text{N} \times d^{\text{sep}})$

where

- $\text{deg} =$ the maximum degree of a node
- $n =$ number of variables (= number of CPTs)
- $N =$ number of nodes in the tree decomposition
- $d =$ the maximum domain size of a variable
- $w^* =$ the induced width
- $\text{sep} =$ the separator size
Join-Tree Clustering

\[ h_{(1,2)}(b, c) = \sum_a p(a) \cdot p(b \mid a) \cdot p(c \mid a, b) \]
\[ h_{(2,1)}(b, c) = \sum_{d,f} p(d \mid b) \cdot p(f \mid c, d) \cdot h_{(3,2)}(b, f) \]
\[ h_{(2,3)}(b, f) = \sum_{c,d} p(d \mid b) \cdot p(f \mid c, d) \cdot h_{(1,2)}(b, c) \]
\[ h_{(3,2)}(b, f) = \sum_e p(e \mid b, f) \cdot h_{(4,3)}(e, f) \]
\[ h_{(3,4)}(e, f) = \sum_b p(e \mid b, f) \cdot h_{(2,3)}(b, f) \]
\[ h_{(4,3)}(e, f) = p(G = g_e \mid e, f) \]

**EXACT algorithm**

**Time and space:**
\[ \exp(\text{cluster size}) = \exp(\text{treewidth}) \]
Mini-Clustering

Split a cluster into mini-clusters => bound complexity

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

\[ \sum_{\text{elim}} \prod_{i=1}^{n} h_i \leq \left( \sum_{\text{elim}} \prod_{i=1}^{r} h_i \right) \cdot \left( \sum_{\text{elim}} \prod_{i=r+1}^{n} h_i \right) \]

Exponential complexity decrease \[ O(e^n) \rightarrow O(e^{\text{var}(r)}) + O(e^{\text{var}(n-r)}) \]
Mini-Clustering, i-bound=3

\[ h_{(1,2)}^{(b,c)} = \sum_a p(a) \cdot p(b \mid a) \cdot p(c \mid a,b) \]

\[ h_{(2,3)}^{(b)} = \sum_{d,c} p(d \mid b) \cdot h_{(1,2)}^{(b,c)} \]

\[ h_{(2,3)}^{(f)} = \max_{c,d} p(f \mid c,d) \]

**APPROXIMATE algorithm**

Time and space: \( \exp(i\text{-bound}) \)

Number of variables in a mini-cluster
Mini-Clustering

- **Correctness and completeness**: Algorithm MC-bel(\(i\)) computes a bound (or an approximation) on the joint probability \(P(X_i,e)\) of each variable and each of its values.

- **Time & space complexity**: \(O(n \times hw^* \times k^i)\)

where \(hw^* = \max_u | \{f | f \cap \chi(u) \neq \emptyset\} |\)
Lower bounds and mean approximations

We can replace $max$ operator by

- $min$  $=>$ lower bound on the joint
- $mean$  $=>$ approximation of the joint
Grid 15x15 - 10 evidence

Grid 15x15, evid=10, w*=22, 10 instances

Grid 15x15, evid=10, w*=22, 10 instances

Grid 15x15, evid=10, w*=22, 10 instances

Grid 15x15, evid=10, w*=22, 10 instances
CPCS 422, evid=0, w* = 23, 1 instance

CPCS 422, evid=10, w* = 23, 1 instance

evidence=0
evidence=10
Coding networks - Bit Error Rate

Coding networks, $N=100$, $P=4$, $\sigma=.22$, $w^*=12$, 50 instances

Coding networks, $N=100$, $P=4$, $\sigma=.51$, $w^*=12$, 50 instances

$\sigma=0.22$  

$\sigma=.51$
**Heuristic for partitioning**

**Scope-based Partitioning Heuristic.** The *scope-based* partition heuristic (SCP) aims at minimizing the number of mini-buckets in the partition by including in each minibucket as many functions as possible as long as the $i$ bound is satisfied. First, single function mini-buckets are decreasingly ordered according to their arity. Then, each minibucket is absorbed into the left-most mini-bucket with whom it can be merged.

The time and space complexity of $\text{Partition}(B, i)$, where $B$ is the partitioned bucket, using the SCP heuristic is $O(|B| \log |B| + |B|^2)$ and $O(\exp(i))$, respectively.

The scope-based heuristic is quite fast, its shortcoming is that it does not consider the actual information in the functions.
Content-based heuristics
(Rollon and Dechter 2010)

- Log relative error:
  \[ RE(f, h) = \sum_i (\log(f(t)) - \log(h(t))) \]

- Max log relative error:
  \[ MRE(f, h) = \max_t \{\log(f(t)) - \log(h(t))\} \]

Partitioning lattice of bucket \( \{f_1, f_2, f_3, f_4\} \).

Use greedy heuristic derived from a distance function to decide which functions go into a single mini-bucket.
Agenda

- Mini-bucket elimination
- Mini-clustering
- Iterative Belief propagation
- Iterative-join-graph propagation
- Use of Mini-bucket for Heuristic search
Agenda

- Mini-bucket elimination
- Mini-clustering
- Iterative Belief propagation
- Iterative-join-graph propagation
  - IJGP complexity
  - Convergence and pair-wise consistency
  - Accuracy when converged
  - Belief Propagation and constraint propagation
- Using Mini-bucket as heuristics for optimization
Probability of evidence (or partition function)

\[
P(e) = \sum_{X \, \text{var}(e)} P(x_i \mid p_a_i) \bigg|_e \quad Z = \sum_X \prod_i \psi_i(C_i)
\]

- **Posterior marginal (beliefs):**

\[
P(x_i \mid e) = \frac{P(x_i, e)}{P(e)} = \frac{\sum_{X \, \text{var}(e) - X_i} \prod_{j=1}^n P(x_j \mid p_a_j) \bigg|_e}{\sum_{X \, \text{var}(e)} \prod_{j=1}^n P(x_j \mid p_a_j) \bigg|_e}
\]

- **Most Probable Explanation**

\[
\bar{x}^* = \arg \max_{\bar{x}} P(\bar{x}, e)
\]
Iterative Belief Propagation

- Belief propagation is exact for poly-trees
- IBP - applying BP iteratively to cyclic networks
- No guarantees for convergence
- Works well for many coding networks
Belief network

Loopy BP graph
CPCS422 - Absolute error

CPCS 422, evid=0, w*=23, 1 instance

CPCS 422, evid=10, w*=23, 1 instance

Absolute error

i-bound

evidence=0

evidence=10
MBE-mpe vs. IBP

mb-e - mpe is better on low - w * codes
IBP is better on randomly generated (high - w *) codes

Bit error rate (BER) as a function of noise (sigma):

Structured (50,25) block code, P=7

Random (100,50) block code, P=4
Iterative Join Graph Propagation

- Loopy Belief Propagation
  - Cyclic graphs
  - Iterative
  - Converges fast in practice (no guarantees though)
  - Very good approximations (e.g., turbo decoding, LDPC codes, SAT – survey propagation)

- Mini-Clustering(i)
  - Tree decompositions
  - Only two sets of messages (inward, outward)
  - Anytime behavior – can improve with more time by increasing the i-bound

- We want to combine:
  - Iterative virtues of Loopy BP
  - Anytime behavior of Mini-Clustering(i)
IJGP - The basic idea

- Apply Cluster Tree Elimination to any join-graph
- We commit to graphs that are I-maps
- Avoid cycles as long as I-mapness is not violated
- Result: use minimal arc-labeled join-graphs
Minimal arc-labeled join-graph

Figure 1.17: a) A belief network; b) A dual join-graph with singleton labels; c) A dual join-graph which is a join-tree

Figure 1.15: An arc-labeled decomposition
IJGP - Example

Belief network

Loopy BP graph
Arcs labeled with any single variable should form a TREE
Collapsing Clusters
Join-Graphs

more accuracy

less complexity
Message propagation

Minimal arc-labeled:
sep(1,2)={D,E}
elim(1,2)={A,B,C}

Non-minimal arc-labeled:
sep(1,2)={C,D,E}
elim(1,2)={A,B}

\[
h_{(1,2)}(de) = \sum_{a,b,c} p(a)p(c)p(b|ac)p(d|abe)p(e|bc)h_{(3,1)}(bc)
\]

\[
h_{(1,2)}(cde) = \sum_{a,b} p(a)p(c)p(b|ac)p(d|abe)p(e|bc)h_{(3,1)}(bc)
\]
Bounded decompositions

- We want arc-labeled decompositions such that:
  - the cluster size (internal width) is bounded by $i$ (the accuracy parameter)
  - the width of the decomposition as a graph (external width) is as small as possible

- Possible approaches to build decompositions:
  - partition-based algorithms - inspired by the mini-bucket decomposition
  - grouping-based algorithms
Constructing Join-Graphs

a) schematic mini-bucket(i), i=3

b) arc-labeled join-graph decomposition
Empirical evaluation

- **Algorithms:**
  - Exact
  - IBP
  - MC
  - IJGP

- **Measures:**
  - Absolute error
  - Relative error
  - Kulbach-Leibler (KL) distance
  - Bit Error Rate
  - Time

- **Networks (all variables are binary):**
  - Random networks
  - Grid networks (MxM)
  - CPCS 54, 360, 422
  - Coding networks
Coding networks - BER

Coding, N=400, 1000 instances, 30 it, w*=43, sigma=.22

\begin{align*}
\text{sigma} &= .22 \\
\text{BER} &= \{1e-5, 1e-4, 1e-3, 1e-2, 1e-1\}
\end{align*}

Coding, N=400, 500 instances, 30 it, w*=43, sigma=.32

\begin{align*}
\text{sigma} &= .32 \\
\text{BER} &= \{0.00237, 0.00238, 0.00239, 0.00240, 0.00241, 0.00242, 0.00243\}
\end{align*}

Coding, N=400, 500 instances, 30 it, w*=43, sigma=.51

\begin{align*}
\text{sigma} &= .51 \\
\text{BER} &= \{0.0745, 0.0750, 0.0755, 0.0760, 0.0765, 0.0770, 0.0775, 0.0780, 0.0785\}
\end{align*}

Coding, N=400, 500 instances, 30 it, w*=43, sigma=.65

\begin{align*}
\text{sigma} &= .65 \\
\text{BER} &= \{0.1900, 0.1902, 0.1904, 0.1906, 0.1908, 0.1910, 0.1912, 0.1914\}
\end{align*}
CPCS 422 – KL Distance

CPCS 422, evid=0, w*=23, 1instance

CPCS 422, evid=30, w*=23, 1instance

evidence=0

evidence=30
CPCS 422 – KL vs. Iterations

CPCS 422, evid=0, w*=23, 1instance

number of iterations

KL distance

evidence=0

CPCS 422, evid=30, w*=23, 1instance

number of iterations

KL distance

evidence=30
Coding networks - Time

Coding, N=400, 500 instances, 30 iterations, w*=43

The graph shows the time (in seconds) taken for various iterations. The x-axis represents the i-bound, while the y-axis represents the time (seconds).

- IJGP 30 iterations
- MC
- IBP 30 iterations
Agenda

- Mini-bucket elimination
- Mini-clustering
- Iterative Belief propagation
- Iterative-join-graph propagation
  - IJGP complexity
  - Convergence and pair-wise consistency
  - Accuracy when converged
  - Belief Propagation and constraint propagation
- Using Mini-bucket as heuristics for optimization
IJGP properties

- IJGP($i$) applies BP to min arc-labeled join-graph, whose cluster size is bounded by $i$

- On join-trees IJGP finds exact beliefs

- IJGP is a Generalized Belief Propagation algorithm (Yedidia, Freeman, Weiss 2001)

- Complexity of one iteration:
  - time: $O(deg \times (n+N) \times d^{i+1})$
  - space: $O(N \times d^0)$
Agenda

- Mini-bucket elimination
- Mini-clustering
- Iterative Belief propagation
- Iterative-join-graph propagation
  - IJGP complexity
  - Convergence and pair-wise consistency
  - Accuracy when converged
Important IJGP properties

- IJGP achieves pairwise consistency if it converges.
- If IJGP converges, the normalizing constants are unique.
Join-graph decomposition

**Definition 1 (join-graph decompositions)** A join-graph decomposition $JG$ for $\mathcal{M} = \langle X, D, F, \otimes, \downarrow \rangle$ is a triple $JG = \langle G, \chi, \psi \rangle$, where $G = (V, E)$ is a graph, and $\chi$ and $\psi$ are labeling functions which associate each vertex $v \in V$ with two sets, $\chi(v) \subseteq X$ and $\psi(v) \subseteq F$ such that:

I. For each $f \in F$, there is exactly one vertex $v \in V$ such that $f \in \psi(v)$, and scope$(f) \subseteq \chi(v)$.

II. (connectedness) For each variable $X_i \in X$, the set $\{v \in V | X_i \in \chi(v)\}$ induces a connected subgraph of $G$. The connectedness requirement is also called the running intersection property.
**Pairwise consistency**

**Definition 2 (Pairwise-consistency (pwc))** Given a join-graph decomposition $\mathcal{JG} = \langle G, \chi, \Psi \rangle$, $G = (V, E)$ of a graphical model $\mathcal{M} = \langle X, D, F \rangle$, then $\mathcal{JG}$ is pairwise-consistent (pwc) relative to a set of messages $H = \{h_{u\rightarrow v}, h_{v\rightarrow u} | (u, v) \in E\}$, iff for every $(u, v) \in E$ we have:

\[
\sum_{\chi(u) - \chi(uv)} \psi_u \cdot \prod_{h \in H_u} h = \sum_{\chi(v) - \chi(uv)} \psi_v \cdot \prod_{h \in H_v} h
\]  

(1)

**Definition 3 (Beliefs)** Given a $\mathcal{JG} = \langle G, \chi, \Psi \rangle$, $G = (V, E)$ of a graphical model $\mathcal{M} = \langle X, D, F \rangle$, and a set of messages $H$ for $JG$ then we define the beliefs for every $u \in G$ by:

\[
b(x_u) = \psi_u(x_u) \cdot \prod_{h \in H_u} h(x_u)
\]  

(2)

\[
b_{uv}(x_{uv}) = \sum_{\chi(u) - \chi_{uv}} b_u(x_u)
\]  

(3)
**Pseudo marginals**

**Definition 5 (p-marginal functions)** Given a graphical model for $M = \langle X, D, F \rangle$, the p-marginal function of $M$ is the unnormalized probability distribution defined by

$$\tilde{P}_X(x) = \prod_{f \in F} f(x_f),$$

The p-marginal for a scope $S \subseteq X$ is defined by:

$$\tilde{P}_S(x_S) = \sum_{(X-S)} \tilde{P}_X(x) = \sum_{(X-S)} \prod_{f \in F} f(x_f) \quad (7)$$
Algorithm PWC-propagation

**Algorithm 1: Algorithm Pairwise-Consistency (PWC)**

**Input:** a Join-graph representation $\mathcal{JG} = (G, \chi, \psi)$, $G = (V, E)$ of a graphical model $\mathcal{M} =< X, D, F >$. $\psi_u = \prod_{f \in \psi(u)} f$

**Output:** A set of messages $\mathcal{H}$ of JG and the corresponding augmented join-graph.

**Initialize:** $h_{u \rightarrow v} \leftarrow 1$.

**Repeat**

For every $u \in G$ do

For every neighbor $v$ of $u$ in $G$, node $u$ sends the message $h_{u \rightarrow v}(x_{uv})$ to $v$ defined by:

$$h_{u \rightarrow v}(x_{uv}) \leftarrow \sum_{\chi(u) - \chi(uv)} \psi_u(x_u) \cdot \prod_{(r, v) \in E, r \neq v} h_{r \rightarrow u}(x_{ru})$$  \hspace{2cm} (9)

**endfor**

**Until** there is no change (the algorithm converged) or a time bound

**Return:** $\mathcal{JG}$ augmented by the messages $\mathcal{H} = \{h_{v \leftarrow u}|(u, v) \in E\}$.

Figure 1: Algorithm Pairwise Consistency (PWC)
The main theorem

**Theorem 2** The following hold.

1. If algorithm PWC converged then its output $JG_H$ is PWC.
Proof. Part a: If the algorithm converges then from Eq. 5 it follows that the messages satisfy:

\[ h_{u \rightarrow v}(x_{uv}) = \sum_{\chi(u) - \chi(uv)} \psi(x_u) \prod_{r \in ne(u), r \neq v} h_{r \rightarrow u}(x_{ru}) \]

From this, multiplying both sides by \( h_{v \rightarrow u} \) we get

\[ h_{u \rightarrow v}(x_{uv}) \cdot h_{v \rightarrow u}(x_{vu}) = \sum_{\chi(u) - \chi(uv)} \psi(x_u) \prod_{r \in ne(u)} h_{r \rightarrow u}(x_{ru}) = \sum_{\chi_u - \chi_{u,v}} b_H(x_u) = b_H(x_{vu}) \quad (10) \]

Exchanging \( u \) and \( v \) everywhere we get also that

\[ h_{v \rightarrow u}(x_{uv}) \cdot h_{u \rightarrow v}(x_{vu}) = \sum_{\chi_u - \chi_{u,v}} b_H(x_u) = b_H(x_{uv}) \quad (11) \]

and therefore since the left handside of Equations 10and 10 are the same we get that:

\[ b_H(x_{uv}) = b_H(x_{vu}) \]

which expresses the notion of PWC relative to \( JG_H \).

parts b and c are well known.

\[ \Box \]
Symmetry and pwc

**Definition 6** (Pairwise-consistency (pwc)) Given a join-graph decomposition $\mathcal{JG} = \langle G, \chi, \Psi \rangle$, $G = (V, E)$ of a graphical model $\mathcal{M} = \langle X, D, F \rangle$, then $\mathcal{JG}_H$ is pairwise-consistent (pwc) relative to $H = \{h_{u \rightarrow v}(x_{uv}), h_{v \rightarrow u}(x_{vu}) | (u, v) \in E\}$, iff for every $(u, v) \in E$ we have:

$$\sum_{\chi(w) - \chi(uv)} \psi_u(X_u) \cdot \prod_{k \neq (v)} h_{k \rightarrow v}(X_{ku}) = \sum_{\chi(v) - \chi(uv)} \psi_v(x_u) \cdot \prod_{k \neq (u)} h_{k \rightarrow u}(X_{kv})$$

(7)

**Definition 7** (Symmetry) Given a join-graph decomposition $\mathcal{JG} = \langle G, \chi, \Psi \rangle$, $G = (V, E)$ of a graphical model $\mathcal{M} = \langle X, D, F \rangle$, then $\mathcal{JG}_H$ is symmetric relative to $H$ iff $\forall (u, v) \in E$.

$$b_H(x_{uv}) = h_{u \rightarrow v}(x_{uv}) \cdot h_{v \rightarrow u}(x_{vu})$$

(8)
Fixed point iff symmetry

**Theorem 1** Given a join-graph decomposition $\mathcal{JG} = \langle G, \chi, \Psi \rangle$, $G = (V, E)$ of a graphical model $\mathcal{M} = \langle X, D, F \rangle$ and given a set of messages $H_{JG}$.

I. If a set of messages $H$ is a fixed point of algorithm PWC when applied to $JG$ then $JG_H$ is symmetric.

II. If we have a set of messages $H_{JG}$ such that $JG_H$ is symmetric than $H_{JG}$ is a fixed point of algorithm PWC.
**Proposition 1** If $JG_H$ is symmetric then $JG_H$ is pairwise consistent, but not vice-versa. We can have a pairwise consistent $JG_H$ which is not symmetric.

**Proof.** It is trivial to show that symmetry implies pwc since by definition of equation 8 it is defined in a symmetric way for $u$ and $v$. To show that the pwc does not imply symmetry consider the graphical model having three variables $X, Y, Z$ and two potentials that are marginals of the same distribution, $P(X, Y)$ and $P(Y, Z)$. Assume constant messages $h = 1$ and a $JG$ which is the dual graph of the graphical models (each function is a cluster). Clearly $JG_H$ is pwc relative to the dual graph since we have only two nodes and marginalizing over $X$ yield the same marginal. However $JG_H$ is clearly not symmetric since $b_H(Y) = P(Y) \neq 1$. \[ \square \]
**Proposition 1** A join graph is pwc relative to $\mathcal{H}$ iff we have:

\[
b_{uv}(x_{uv}) = \sum_{\chi(u)-\chi_{uv}} b_u(x_u) = \sum_{\chi(v)-\chi_{vu}} b_v(x_v) = b_{vu}(x_{vu})
\] (4)

**Definition 4** (normalizing constant) Given a $\mathcal{JG} = < G, \chi, \Psi >$, $G = (V, E)$, and a set of messages $H$ for $JG$ then $\forall u \in V$ we define the belief’s normalized constant by

\[
K(u) = \sum_{x_u} b_u(x_u)
\] (5)

\[
K(uv) = \sum_{x_{uv}} b_{uv}(x_{uv})
\] (6)
Theorem 1  If $\mathcal{J}G = \langle G, \chi, \Psi \rangle$ is pwc relative to messages $\mathcal{H}$ then, $\forall u, v, (u, v) \in E$

$$K(u) = K(v) = K(uv)$$

Proof. If $\mathcal{J}G = \langle G, \chi, \Psi \rangle$ is pwc relative to messages $\mathcal{H}$ then

$$K(u) = \sum_{x-u} b_u(x_u) =$$

$$= \sum_{x_{uv}} \sum_{x_{uv} - x_{uv}} b_u(x_u) =$$

and because of pwc holds

$$K(u) = \sum_{x_{uv}} b_{uv}(x_{uv}) = \sum_{x_{vu}} b_{vu}(x_{vu}) = \sum_{x_{vu} \chi(v) - x_{vu}} b_v(x_v) = \sum_{\chi(v)} b_v(x_v) = K(v)$$
Repatameterization

\[ Q(x) = \frac{\prod_{v \in V} b_H(x_v)}{\prod_{(u,v) \in E} h_{u \rightarrow v}(x_{uv}) \cdot h_{v \rightarrow u}(x_{vu})} \]
Agenda

- Mini-bucket elimination
- Mini-clustering
- Iterative Belief propagation
- Iterative-join-graph propagation
  - IJGP complexity
  - Convergence and pair-wise consistency
  - Accuracy when converged
  - BP and constraint propagation
More On the Power of Belief Propagation

- BP as local minima of KL distance
- BP’s power from constraint propagation perspective.
More On the Power of Belief Propagation

- BP as local minima of KL distance
- BP’s power from constraint propagation perspective.
The Kullback-Leibler Divergence

The Kullback-Leibler divergence (KL–divergence)

\[
\text{KL}(\text{Pr}'(X|e), \text{Pr}(X|e)) = \sum x \text{Pr}'(x|e) \log \frac{\text{Pr}'(x|e)}{\text{Pr}(x|e)}
\]

- KL(Pr'(X|e), Pr(X|e)) is non-negative
- equal to zero if and only if Pr'(X|e) and Pr(X|e) are equivalent.
The Kullback-Leibler Divergence

KL–divergence is not a true distance measure in that it is not symmetric. In general:

$$KL(\Pr'(\mathbf{X}|\mathbf{e}), \Pr(\mathbf{X}|\mathbf{e})) \neq KL(\Pr(\mathbf{X}|\mathbf{e}), \Pr'(\mathbf{X}|\mathbf{e})).$$

- KL($\Pr'(\mathbf{X}|\mathbf{e}), \Pr(\mathbf{X}|\mathbf{e})$) weighting the KL–divergence by the approximate distribution $\Pr'$
- We shall indeed focus on the KL–divergence weighted by the approximate distribution as it has some useful computational properties.
The Kullback-Leibler Divergence

Let $\text{Pr}(\mathbf{X})$ be a distribution induced by a Bayesian network $\mathcal{N}$ having families $\mathbf{XU}$.

The KL–divergence between $\text{Pr}$ and another distribution $\text{Pr}'$ can be written as a sum of three components:

$$\text{KL}(\text{Pr}'(\mathbf{X}|\mathbf{e}), \text{Pr}(\mathbf{X}|\mathbf{e})) = -\text{ENT}'(\mathbf{X}|\mathbf{e}) - \sum_{\mathbf{XU}} \text{AVG}'(\log \lambda_e(\mathbf{X})\Theta_{\mathbf{X}|\mathbf{U}}) + \log \text{Pr}(\mathbf{e}),$$

where

- $\text{ENT}'(\mathbf{X}|\mathbf{e}) = -\sum_{\mathbf{x}} \text{Pr}'(\mathbf{x}|\mathbf{e}) \log \text{Pr}'(\mathbf{x}|\mathbf{e})$ is the entropy of the conditioned approximate distribution $\text{Pr}'(\mathbf{X}|\mathbf{e})$.

- $\text{AVG}'(\log \lambda_e(\mathbf{X})\Theta_{\mathbf{X}|\mathbf{U}}) = \sum_{\mathbf{xU}} \text{Pr}'(\mathbf{xU}|\mathbf{e}) \log \lambda_e(\mathbf{x})\theta_{\mathbf{x}|\mathbf{U}}$ is a set of expectations over the original network parameters weighted by the conditioned approximate distribution.
The Kullback-Leibler Divergence

A distribution $\Pr' (X|e)$ minimizes the KL-divergence $\text{KL}(\Pr' (X|e), \Pr (X|e))$ if it maximizes

$$\text{ENT}' (X|e) + \sum_{X \in U} \text{AVG}' (\log \lambda_e (X) \Theta X|u)$$

Competing properties of $\Pr' (X|e)$ that minimize the KL–divergence:

- $\Pr' (X|e)$ should match the original distribution by giving more weight to more likely parameters $\lambda_e (x) \theta_{x|u}$ (i.e., maximize the expectations).

- $\Pr' (X|e)$ should not favor unnecessarily one network instantiation over another by being evenly distributed (i.e., maximize the entropy).
Optimizing the KL-Divergence

The approximations computed by IBP are based on assuming an approximate distribution $\Pr'(X)$ that factors as follows:

$$\Pr'(X|e) = \prod_{X \cup U} \frac{\Pr'(X \cup U|e)}{\prod_{U \in u} \Pr'(U|e)}$$

- This choice of $\Pr'(X|e)$ is expressive enough to describe distributions $\Pr(X|e)$ induced by polytree networks $\mathcal{N}$.
- In the case where $\mathcal{N}$ is not a polytree, then we are simply trying to fit $\Pr(X|e)$ into an approximation $\Pr'(X|e)$ as if it were generated by a polytree network.
- The entropy of distribution $\Pr'(X|e)$ can be expressed as:

$$\text{ENT}'(X|e) = - \sum_{X \cup U} \sum_{x_{u}} \Pr'(x_{u}|e) \log \frac{\Pr'(x_{u}|e)}{\prod_{u \sim u} \Pr'(u|e)}$$
Optimizing the KL-Divergence

Let $P_r(X)$ be a distribution induced by a Bayesian network $N$ having families $X|U$. Then IBP messages are a fixed point if and only if IBP marginals $\mu_u = BEL(u)$ and $\mu_{xu} = BEL(x|u)$ are a stationary point of:

$$\text{ENT}'(X|e) + \sum_{X|U} \text{AVG}'(\log \lambda_e(X) \Theta_{X|U})$$

$$= - \sum_{X|U} \sum_{x|u} \mu_{xu} \log \frac{\mu_{xu}}{\prod_{u \sim u} \mu_u} + \sum_{X|U} \sum_{x|u} \mu_{xu} \log \lambda_e(x) \theta_{x|u},$$

under normalization constraints:

$$\sum_u \mu_u = \sum_{x|u} \mu_{xu} = 1$$

for each family $X|U$ and parent $U$, and under consistency constraints:

$$\sum_{x|u \sim y} \mu_{xu} = \mu_y$$

for each family instantiation $x|u$ and value $y$ of family member $Y \in X|U$. 

IBP fixed points are stationary points of the KL–divergence: they may only be local minima, or they may not be minima.

When IBP performs well, it will often have fixed points that are indeed minima of the KL–divergence.

For problems where IBP does not behave as well, we will next seek approximations $P_{r'}$ whose factorizations are more expressive than that of the polytree-based factorization.
If a distribution $\Pr'(X|e)$ has the form:

$$\Pr'(X|e) = \frac{\prod_C \Pr'(C|e)}{\prod_S \Pr'(S|e)},$$

then its entropy has the form:

$$\text{ENT}'(X|e) = \sum_C \text{ENT}'(C|e) - \sum_S \text{ENT}'(S|e).$$

When the marginals $\Pr'(C|e)$ and $\Pr'(S|e)$ are readily available, the ENT component of the KL–divergence can be computed efficiently.
While a jointree induces an exact factorization of a distribution, a jointgraph $G$ induces an approximate factorization:

$$\Pr'(X|e) = \frac{\prod_i \Pr'(C_i|e)}{\prod_{ij} \Pr'(S_{ij}|e)}$$

which is a product of cluster marginals over a product of separator marginals. When the jointgraph corresponds to a jointree, the above factorization will be exact.
A dual joingraph $G$ for network $\mathcal{N}$ is obtained as follows:

- $G$ has the same undirected structure of network $\mathcal{N}$.
- For each family $XU$ in network $\mathcal{N}$, the corresponding node $i$ in joingraph $G$ will have the cluster $C_i = XU$.
- For each $U \rightarrow X$ in network $\mathcal{N}$, the corresponding edge $i-j$ in joingraph $G$ will have the separator $S_{ij} = U$. 
Computing cluster marginals $\mu_{c_i} = \Pr'(c_i|e)$ and separator marginals $\mu_{s_{ij}} = \Pr'(s_{ij}|e)$ that minimize the KL–divergence between $\Pr'(X|e)$ and $\Pr(X|e)$.

This optimization problem can be solved using a generalization of IBP, called iterative join graph propagation (IJGP), which is a message passing algorithm that operates on a join graph.
Iterative JoinGraph Propagation

\textbf{IJGP}(G, \Phi)

\textbf{input:}
\begin{itemize}
  \item $G$: a joingraph
  \item $\Phi$: factors assigned to clusters of $G$
\end{itemize}

\textbf{output:} approximate marginal $BEL(C_i)$ for each node $i$ in the joingraph $G$.

\textbf{main:}
\begin{enumerate}
  \item $t \leftarrow 0$
  \item initialize all messages $M^t_{ij}$ (uniformly)
  \item while messages have not converged do
  \item \hspace{1em} $t \leftarrow t + 1$
  \item \hspace{1em} for each joingraph edge $i-j$ do
  \item \hspace{2em} $M^t_{ij} \leftarrow \eta \ \sum_{c_i \setminus s_{ij}} \Phi_i \ \Pi_{k \neq j} M^t_{ki}^{-1}$
  \item \hspace{2em} $M^t_{ji} \leftarrow \eta \ \sum_{c_j \setminus s_{ij}} \Phi_j \ \Pi_{k \neq i} M^t_{kj}^{-1}$
  \item \hspace{1em} end for
  \item end while
  \item return $BEL(C_i) \leftarrow \eta \ \Phi_i \ \Pi_{k} M^t_{ki}$ for each node $i$
\end{enumerate}
Let $P_r(X)$ be a distribution induced by a Bayesian network $\mathcal{N}$ having families $XU$, and let $C_i$ and $S_{ij}$ be the clusters and separators of a joingraph for $\mathcal{N}$.

Then messages $M_{ij}$ are a fixed point of IJGP if and only if IJGP marginals $\mu_{c_i} = BEL(c_i)$ and $\mu_{s_{ij}} = BEL(s_{ij})$ are a stationary point of:

$$\text{ENT}'(X|e) + \sum\limits_{C_i} \text{AVG}'(\log \Phi_i)$$

$$= -\sum\sum \mu_{c_i} \log \mu_{c_i} + \sum\sum \mu_{s_{ij}} \log \mu_{s_{ij}} + \sum\sum \mu_{c_i} \log \Phi_i(c_i),$$

under normalization constraints:

$$\sum_{C_i} \mu_{c_i} = \sum_{S_{ij}} \mu_{s_{ij}} = 1$$

for each cluster $C_i$ and separator $S_{ij}$, and under consistency constraints:

$$\sum_{C_i \sim S_{ij}} \mu_{c_i} = \mu_{s_{ij}} = \sum_{C_j \sim S_{ij}} \mu_{c_j}$$

for each separator $S_{ij}$ and neighboring clusters $C_i$ and $C_j$. 

Iterative Joingraph Propagation
Summary of IJGP so far

A spectrum of approximations.

IBP: results from applying IJGP to the dual joingraph.

Jointree algorithm: results from applying IJGP to a jointree (as a joingraph).

In between these two ends, we have a spectrum of joingraphs and corresponding factorizations, where IJGP seeks stationary points of the KL–divergence between these factorizations and the original distribution.
Agenda

- Mini-bucket elimination
- Mini-clustering
- Iterative Belief propagation
- Iterative-join-graph propagation
  - IJGP complexity
  - Convergence and pair-wise consistency
  - Accuracy when converged
  - Belief Propagation and constraint propagation
- Using Mini-bucket as heuristics for optimization
More On the Power of Belief Propagation

- BP as local minima of KL distance
- BP’s power from constraint propagation perspective.
Inference Power of Loopy BP

- Comparison with iterative algorithms in constraint networks
- Zero-beliefs \iff\ inconsistent assignments
- $\varepsilon$-Small beliefs – experimental study
Constraint networks

Map coloring

Variables: countries (A B C etc.)

Values: colors (red green blue)

Constraints: $A \neq B, A \neq D, D \neq E, \text{ etc.}$
Arc-consistency

- Sound
- Incomplete
- Always converges (polynomial)
Relational Distributed Arc-Consistency

Primal

Dual

A
1
2
3

B
1
2
3

C
1
2
3

D
1
2
3

A < B
1 2
2 3

A < D
1 2
2 3

B = C
1 1
2 2
3 3

D < C
1 2
2 3

AB
1 2
2 3

AD
1 2
2 3

AB
1 2 1 2
2 3 2 3

AD
1 2
2 3

BC
1 1
2 2
3 3

DC
1 2
2 3

AB
1 2 2 2
2 3 3 3

BC
2 2 1 2
3 3 2 3

A

B

C

D

95
Flattening the Bayesian Network

Belief network

Flat constraint network
Belief Zero Propagation = Arc-Consistency

\[ h_i^j = \sum_{\text{elim}(i,j)} (p_i \cdot \prod_{k \in \text{ne}(i)} h_k^i) \]

\[ h_i^j = \pi_{t_j} (R_i \otimes (\otimes_{k \in \text{ne}(i)} h_k^i)) \]

Updated belief:

\[ Bel(A, B) = P(B \mid A) \cdot h_1^2 \cdot h_4^2 \cdot h_5^2 = \]

\[ Bel(A, B) = \begin{array}{cc}
1 & 2 \\
1 & 3 \\
n & 2 \\
n & 3 \\
\end{array} \]

Updated relation:

\[ R(A, B) = R(A, B) \otimes h_1^2 \otimes h_4^2 \otimes h_5^2 = \]

\[ R(A, B) = \begin{array}{cc}
1 & 2 \\
1 & 3 \\
n & 2 \\
n & 3 \\
\end{array} \]
Flat Network - Example

\[ \begin{array}{c|c} A & P(A) \\ \hline 1 & .2 \\ 2 & .5 \\ 3 & .3 \\ \vdots & 0 \end{array} \]

\[ \begin{array}{c|c|c|c} A & B & P(B|A) \\ \hline 1 & 2 & .3 \\ 1 & 3 & .7 \\ 2 & 1 & .4 \\ 2 & 3 & .6 \\ 3 & 1 & .1 \\ 3 & 2 & .9 \\ \vdots & \vdots & 0 \end{array} \]

\[ \begin{array}{c|c|c|c} A & C & P(C|A) \\ \hline 1 & 2 & 1 \\ 3 & 2 & 1 \\ \vdots & \vdots & 0 \end{array} \]

\[ \begin{array}{c|c|c|c} B & C & F & P(F|B,C) \\ \hline 1 & 2 & 3 & 1 \\ 3 & 2 & 1 & 1 \\ \vdots & \vdots & \vdots & 0 \end{array} \]
IBP Example – Iteration 2

**R1**

<table>
<thead>
<tr>
<th>A</th>
<th>P(A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&gt;0</td>
</tr>
<tr>
<td>3</td>
<td>&gt;0</td>
</tr>
<tr>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>

**R2**

| A | B | P(B|A) |
|---|---|-------|
| 1 | 3 | 1     |
| 3 | 1 | 1     |
| ... | ... | 0    |

**R3**

| A | C | P(C|A) |
|---|---|-------|
| 1 | 2 | 1     |
| 3 | 2 | 1     |
| ... | ... | 0    |

**R4**

| A | B | D | P(D|A,B) |
|---|---|---|---------|
| 1 | 3 | 2 | 1       |
| 3 | 1 | 2 | 1       |
| ... | ... | ... | 0      |

**R5**

| B | C | F | P(F|B,C) |
|---|---|---|---------|
| 3 | 2 | 1 | 1       |
| ... | ... | ... | 0      |

**R6**

| D | F | G | P(G|D,F) |
|---|---|---|---------|
| 2 | 1 | 3 | 1       |
| ... | ... | ... | 0      |
IBP Example – Iteration 3
IBP Example – Iteration 4
IBP Example – Iteration 5

\[
\begin{array}{c|c}
\text{P}(A) & 1 \\
\hline
\text{P}(B|A) & 1 \\
\end{array}
\]

\[
\begin{array}{c|c}
\text{P}(C|A) & 1 \\
\hline
\text{P}(F|B,C) & 1 \\
\end{array}
\]

\[
\begin{array}{c|c}
\text{P}(D|A,B) & 1 \\
\hline
\text{P}(G|D,F) & 1 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c|c|c}
\text{A} & \text{B} & \text{C} & \text{D} & \text{F} & \text{G} & \text{Belief} \\
\hline
1 & 3 & 2 & 2 & 1 & 3 & 1 \\
... & ... & ... & ... & ... & ... & 0 \\
\end{array}
\]

\[
\begin{array}{c|c}
\text{P}(C|A) & 1 \\
\hline
\text{P}(F|B,C) & 1 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c|c|c}
\text{A} & \text{C} & \text{P}(C|A) & 1 \\
\hline
1 & 2 & 1 & 1 \\
... & ... & 0 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c|c|c}
\text{B} & \text{C} & \text{F} & \text{P}(F|B,C) & 1 \\
\hline
3 & 2 & 1 & 1 \\
... & ... & ... & 0 \\
\end{array}
\]
Theorem:
Trace of zero beliefs of Iterative Belief Propagation =
Trace of invalid tuples of arc-consistency on flat network

Soundness:
- The inference of zero beliefs by IBP converges in a finite number of iterations
- All the inferred zero beliefs are correct

Incompleteness:
- IBP may not infer all the true zero beliefs
Zero and $\varepsilon$-Small Beliefs

- Zero beliefs
- $\varepsilon$-small beliefs

- # of tuples
- LBP zeros
- True zeros

- # of tuples
- LBP $\varepsilon$-small
- True $\varepsilon$-small
N=200, 1000 instances, treewidth=15
N=100, 100 instances, w*=15
Random Networks

N=80, 100 instances, w*=15
CPCS360: 5 instances, w*=20
CPCS54: 100 instances, w*=15
Experimental Results

We investigated empirically if the results for zero beliefs extend to $\varepsilon$-small beliefs ($\varepsilon > 0$)

Network types:
- Coding
- Linkage analysis*
- Grids*
- Two-layer noisy-OR*
- CPCS54, CPCS360

Measures:
- Exact/IJGP histogram
- Recall absolute error
- Precision absolute error

Algorithms:
- IBP
- IJGP

* Instances from the UAI08 competition
Networks with Determinism: Coding

N=200, 1000 instances, w*=15
Nets w/o Determinism: bn2o

\( w^* = 24 \)

\( w^* = 27 \)

\( w^* = 26 \)
Some competition comparison
IJGP on UAI06 problems

![Graph showing IJGP performance on UAI06 problems](Image)
IJGP on Set Relational

Approximate Mar Problem Set Relational

![Graph showing sum score over time for different methods.](chart.png)

- **SampleSearch**
- **EDBP**
- **IJGP**
- **TLSBP**
- **EPIS**
Agenda

- Mini-bucket elimination
- Mini-clustering
- Iterative Belief propagation
- Iterative-join-graph propagation
  - IJGP complexity
  - Convergence and pair-wise consistency
  - Accuracy when converged
  - Belief Propagation and constraint propagation
- Using Mini-bucket as heuristics for optimization

(did not go beyond this slides)
Mini-Bucket can be used to guide more than one solution

L = lower bound
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 (or more)

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

[Diagram showing the pruning process in Branch-and-Bound and the expansion order in Best-First Search]
Heuristic search

- Mini-buckets record upper-bound heuristics
- The evaluation function over

- **Best-first**: expand a node with maximal evaluation function
- **Branch and Bound**: prune if $f \leq$ upper bound
- **Properties**:
  - an exact algorithm
  - Better heuristics lead to more pruning
Given a cost function

\[ P(a,b,c,d,e) = P(a) \cdot P(b|a) \cdot P(c|a) \cdot P(e|b,c) \cdot P(d|b,a) \]

Define an evaluation function over a partial assignment as the probability of it’s best extension

\[
 f^*(a,e,d) = \max_{b,c} P(a,b,c,d,e) = \\
 = P(a) \cdot \max_{b,c} P(b|a) \cdot P(c|a) \cdot P(e|b,c) \cdot P(d|b,a) \\
 = g(a,e,d) \cdot H^*(a,e,d)
\]
MBE Heuristics

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

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

$$f(a, e, D) = F(a) + h^B(D,a) + h^C(e,a)$$

$g$ and $h$ – is admissible.
Properties

- Heuristic is consistent/monotone
- Heuristic is admissible
- Heuristic is computed in linear time
- IMPORTANT:
  - Mini-buckets generate heuristics of varying strength using control parameter – bound i
  - Higher bound -> more preprocessing -> stronger heuristics -> less search
  - Allows controlled trade-off between preprocessing and search
Classic Branch-and-Bound

Upper Bound UB

Lower Bound LB

\[ LB(n) = g(n) + h(n) \]

Prune if LB(n) ≥ UB

h(n) estimates
Optimal cost below n

OR Search Tree
Empirical Evaluation of mini-bucket heuristics

Random Coding, K=100, noise 0.32
AND/OR Branch-and-Bound Search

\[ f(T') \geq UB \]
Heuristic Evaluation Function

\[ f(T') = w(A,0) + w(B,1) + w(C,0) + w(D,0) + h(D,0) + h(F) = 12 \leq f^*(T') \]
Software & Competitions

- **How to use the software**
  - [http://graphmod.ics.uci.edu/group/Software](http://graphmod.ics.uci.edu/group/Software)
  - [http://mulcyber.toulouse.inra.fr/projects/toulbar2](http://mulcyber.toulouse.inra.fr/projects/toulbar2)

- **Reports on competitions**
  - UAI-2006, 2008, 2010 Competitions
    - PE, MAR, MPE tasks
  - CP-2006 Competition
    - WCSP task
Toulbar2 and aolib

- **toulbar2**
  
  [http://mulcyber.toulouse.inra.fr/gf/project/toulbar2](http://mulcyber.toulouse.inra.fr/gf/project/toulbar2)
  
  (Open source WCSP, MPE solver in C++)

- **aolib**
  
  [http://graphmod.ics.uci.edu/group/Software](http://graphmod.ics.uci.edu/group/Software)
  
  (WCSP, MPE, ILP solver in C++, inference and counting)

- **Large set of benchmarks**
  
  [http://carlit.toulouse.inra.fr/cgi-bin/awki.cgi/SoftCSP](http://carlit.toulouse.inra.fr/cgi-bin/awki.cgi/SoftCSP)
UAI-2006 Competition

- **Team 1 (UCLA)**
  - David Allen, Mark Chavira, Arthur Choi, Adnan Darwiche

- **Team 2 (IET)**
  - Masami Takikawa, Hans Dettmar, Francis Fung, Rick Kissh

- **Team 5 (UCI)**
  - Radu Marinescu, Robert Mateescu, Rina Dechter
  - Used AOBB-C+SMB(i) solver for MPE
UAI-2006 Results

Rank Proportions (how often was each team a particular rank, rank 1 is best)
**UAI-2008 Competition**

- **AOBB-C+SMB(i) – (i = 18, 20, 22)**
  - AND/OR Branch-and-Bound with pre-compiled mini-bucket heuristics (i-bound), full caching, static pseudo-trees, constraint propagation

- **AOBF-C+SMB(i) – (i = 18, 20, 22)**
  - AND/OR Best-First search with pre-compiled mini-bucket heuristics (i-bound), full caching, static pseudo-trees, no constraint propagation

- **Toulbar2**
  - OR Branch-and-Bound, dynamic variable/value orderings, EDAC consistency for binary and ternary cost functions, variable elimination of small degree (2) during search

- **Toulbar2/BTD**
  - DFBB exploiting a tree decomposition (AND/OR), same search inside clusters as toulbar2, full caching (no cluster merging), combines RDS and EDAC, and caching lower bounds
UAI-2008 Results

MPE : All

Instances solved vs. Minutes

- inra
- inra-mf
- aobb1
- aobb2
- aobb3
- aobf1
- aobf2
- aobf3
UAI-2008 Results (contd.)
UAI-2010 Competition

- **Tasks**
  - PR: probability of evidence
  - MAR: posterior marginals
  - MPE: most probable explanation

- **3 tracks: 20 sec, 20 min, 1 hour**
  - PR, MAR - 204 instances; MPE - 442 instances
    - CSP, grids, image alignment, medical diagnosis, object detection, pedigree, protein folding, protein-protein interaction, relational model, segmentation

- **Exact and approximate solvers**
UAI-2010 Results

- **MAR task**
  - *1st place* (20 min, 1 hour) – (impl. by Vibhav Gogate)
  - Anytime *IJGP(i)* with randomized orderings and SAT based domain pruning

- **PR task**
  - *1st place* (20 min, 1 hour) – (impl. by Vibhav Gogate)
  - Formula *SampleSearch* with *IJGP(3)* based importance distribution, *w*-cutset sampling, *minisat* based search, rejection control

- **MPE task**
  - *3rd place* (all tracks) – (impl. by Lars Otten)
  - **AND/OR BnB** with mini-buckets, randomized min-fill based pseudo tree, LDS based search for initial upper bound
Winning the PASCAL 2011 MAP Challenge with Enhanced AND/OR Branch-and-Bound

Lars Otten, Alexander Ihler, Kalev Kask, Rina Dechter

Dept. of Computer Science
University of California, Irvine
Overview

- Placed 1st in all three MPE tracks.
  - Close competition, congratulations to runner-ups!
- Baseline: AND/OR Branch-and-Bound with mini-bucket heuristic.
  - 3rd place for MPE at UAI 2010 Evaluation.
- Our solver DAOOPT is AOBBS “on steroids”:
  - Several enhancements / extensions.
    - All useful in themselves, but hard to quantify.
- Source code available online:
  - http://github.com/lotten/daoopt
AND/OR Branch-and-Bound

Problem decomposition and caching.

- Mini-bucket heuristic for pruning.

Guided by pseudo tree
AND/OR Branch-and-Bound

- Problem decomposition and caching.
- Mini-bucket heuristic for pruning.

Problem decomposition

Guided by pseudo tree
AND/OR Branch-and-Bound

Problem decomposition and caching.

- Mini-bucket heuristic for pruning.
Central Enhancements

\[
\min_{\lambda} \sum_{(ij)} \max_X (f_{ij}(X_i, X_j) + \lambda_{ij}(X_i), \lambda_{ji}(X_j))
\]

**Cost-shifting (MPLP) Re-parametrization**
Tighter bounds by iteratively solving linear programming relaxations and message passing on join graph.

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

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

- **20 sec, 20 min, 1 hour categories**
  - Score computed relative to a baseline/BP solution.
  - \( E(x) = -\sum \log f_i(x), \quad \text{Score}(x) = \frac{E(x) - \min\{E(x^{bp}), E(x^{df})\}}{|\min\{E(x^{bp}), E(x^{df})\}|} \)
  - **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>dfbbvemcs</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>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Grids</td>
<td>-0.3403</td>
<td>-0.3210</td>
<td>-0.3174</td>
</tr>
<tr>
<td>Image alignment</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Medical diagnosis</td>
<td>-0.0028</td>
<td>-0.0046</td>
<td>-0.0460</td>
</tr>
<tr>
<td>Object detection</td>
<td>-4.8201</td>
<td>-4.8287</td>
<td>-4.8023</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>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Relational</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Segmentation</td>
<td>-0.0300</td>
<td>-0.0300</td>
<td>-0.0298</td>
</tr>
</tbody>
</table>