Approximation Techniques
Iterative bounded inference

COMPSCI 276, Spring 2013

Set 11: Rina Dechter

(Reading: Primary: Class Notes (10)
Secondary: , Darwiche chapters 14)

Agenda

- Mini-bucket elimination
- Mini-clustering
- Iterative Belief propagation
- Iterative-join-graph propagation

Cluster Tree Elimination - properties

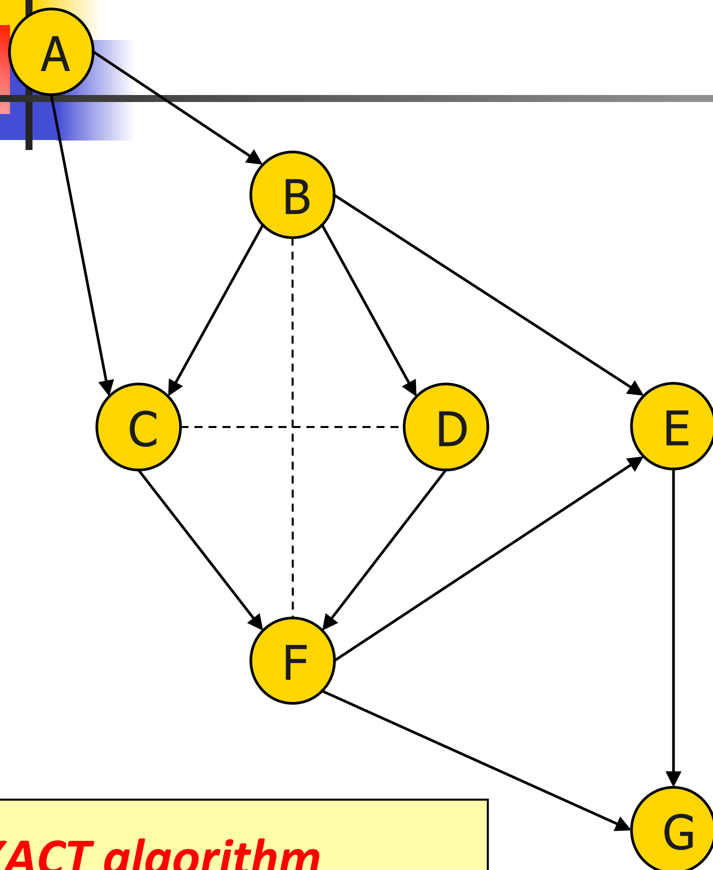
- Correctness and completeness: Algorithm CTE is correct, i.e. it computes the exact joint probability of a single variable and the evidence.

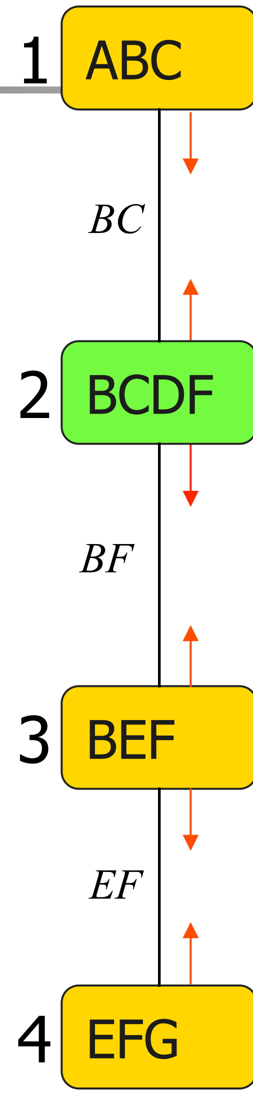
- Time complexity: $O(\, deg \times (n+N) \times d^{\,w^*+1}\,)$

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

where *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
sep = the separator size

Join-Tree Clustering





$$h_{(1,2)}(b,c) = \sum_{a} p(a) \cdot p(b|a) \cdot p(c|a,b)$$

$$h_{(2,1)}(b,c) = \sum_{d,f} p(d|b) \cdot p(f|c,d) \cdot h_{(3,2)}(b,f)$$

$$h_{(2,3)}(b,f) = \sum_{c,d} p(d|b) \cdot p(f|c,d) \cdot h_{(1,2)}(b,c)$$

$$h_{(3,2)}(b,f) = \sum_{e} p(e|b,f) \cdot h_{(4,3)}(e,f)$$

$$h_{(3,4)}(e,f) = \sum_{b} p(e|b,f) \cdot h_{(2,3)}(b,f)$$

$$h_{(4,3)}(e,f) = p(G = g_e | e,f)$$

EXACT algorithm

Time and space:
exp(cluster size)= exp(treewidth)

Mini-Clustering

Split a cluster into mini-clusters => bound complexity

$$\underbrace{\{h_1,\ldots,h_r,h_{r+1},\ldots,h_n\}}$$

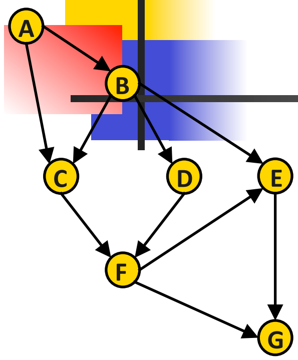
APPROXIMATE algorithm

$$\underbrace{\{h_1,\ldots,h_r\}} \qquad \underbrace{\{h_{r+1},\ldots,h_n\}}$$

$$\sum_{elim}\prod_{i=1}^{n} h_i \le \left(\sum_{elim}\prod_{i=1}^{r} h_i\right)\cdot\left(\sum_{elim}\prod_{i=r+1}^{n} h_i\right)$$

Exponential complexity decrease $O(e^n) \to O(e^{\text{var}(r)}) + O(e^{\text{var}(n-r)})$

Mini-Clustering, i-bound=3



1 **A B C**
p(a), p(b|a), p(c|a,b)

BC

2 **B C D**
p(d|b), h(1,2)(b,c)

C D F
p(f|c,d)

BF

3 **B E F**
p(e|b,f),
h¹(2,3)(b), h²(2,3)(f)

EF

4 **E F G**
p(g|e,f)

$$h^1_{(1,2)}(b,c) = \sum_a p(a) \cdot p(b \mid a) \cdot p(c \mid a,b)$$

$$h^1_{(2,3)}(b) = \sum_{c,d} p(d \mid b) \cdot h^1_{(1,2)}(b,c)$$

$$h^2_{(2,3)}(f) = \max_{c,d} p(f \mid c,d)$$

APPROXIMATE algorithm

Time and space:
exp(i-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: $O(n \times hw^* \times k^i)$

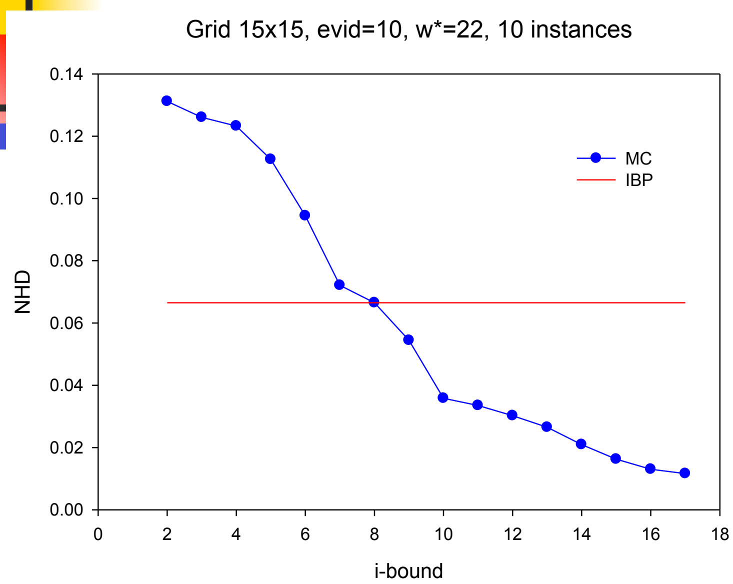
where $hw^* = max_u \mid \{f \mid f \cap \chi(u) \neq \phi\} \mid$

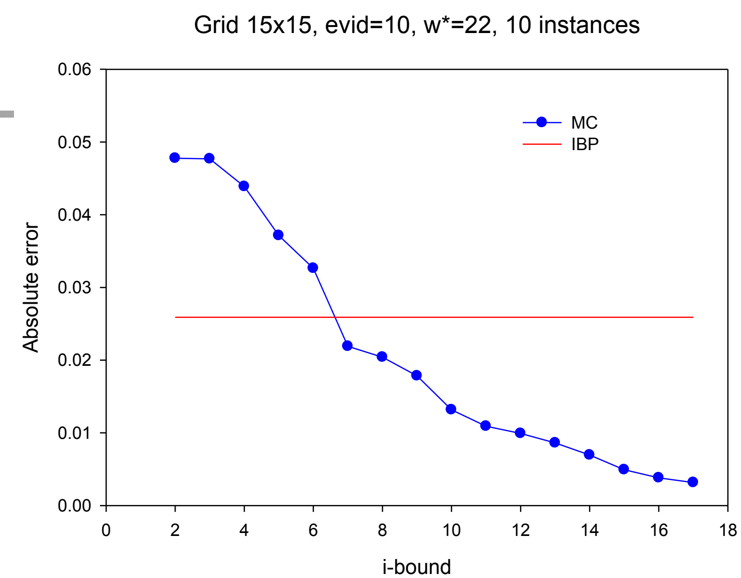
Lower bounds and mean approximations

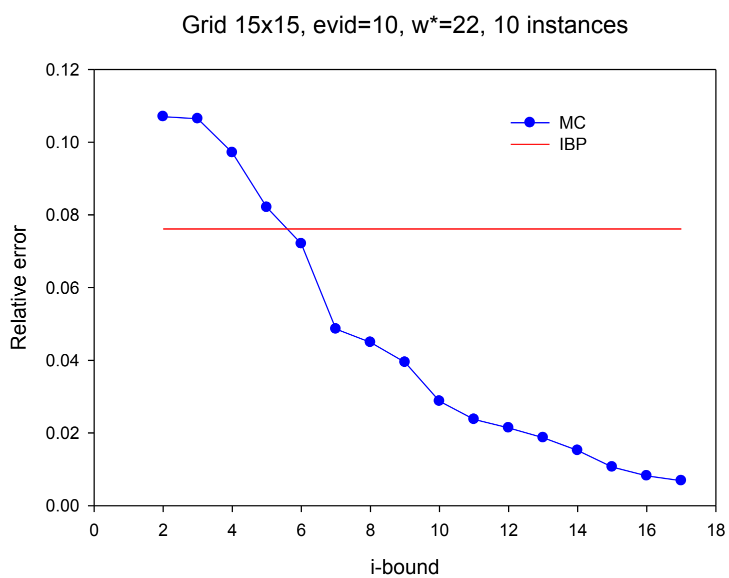
We can replace *max* operator by

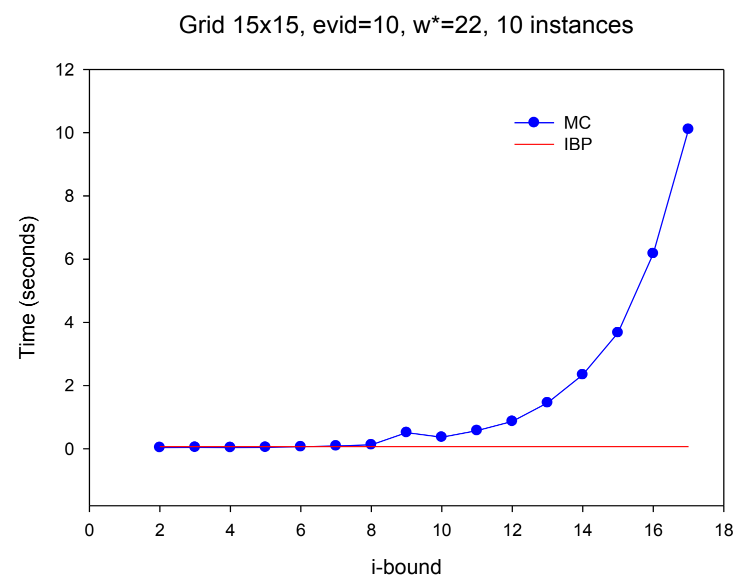
- *min* => lower bound on the joint
- *mean* => approximation of the joint

Grid 15x15 - 10 evidence

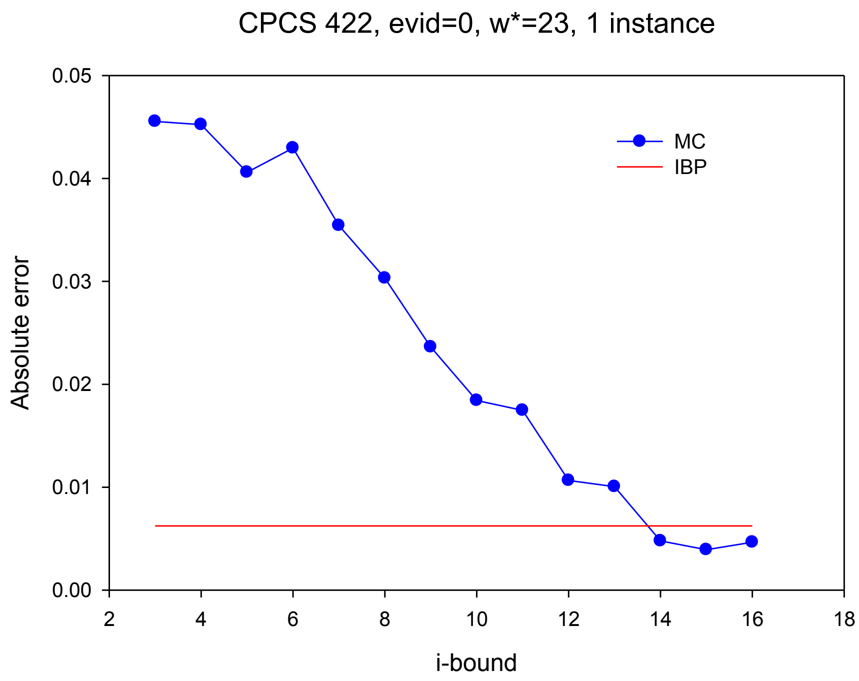


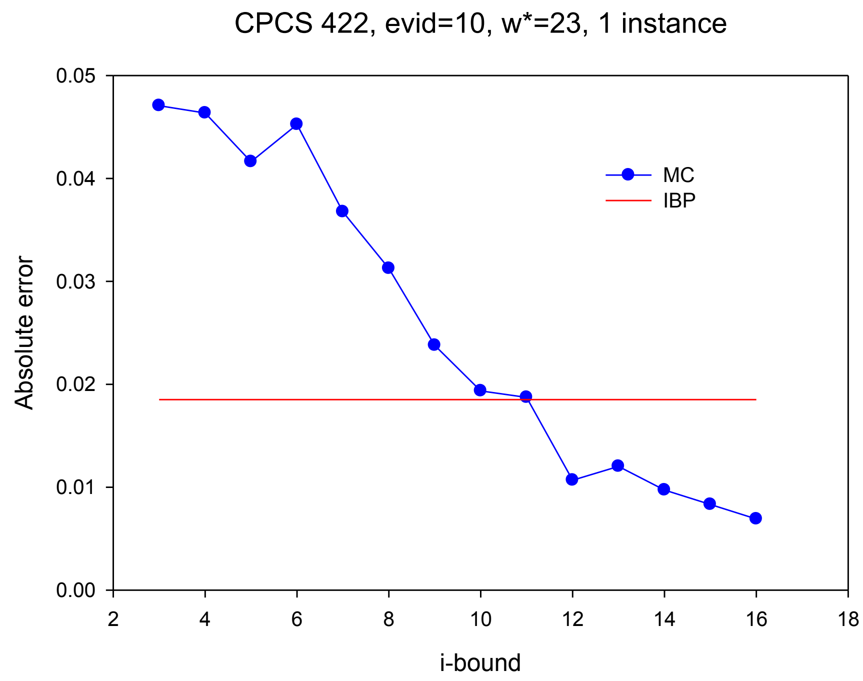






CPCS422 - Absolute error

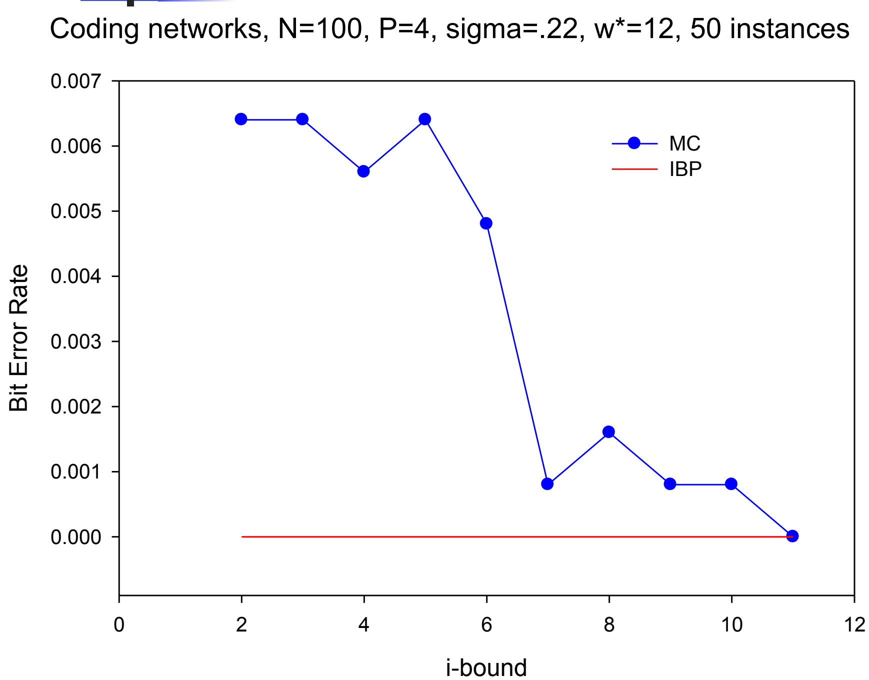


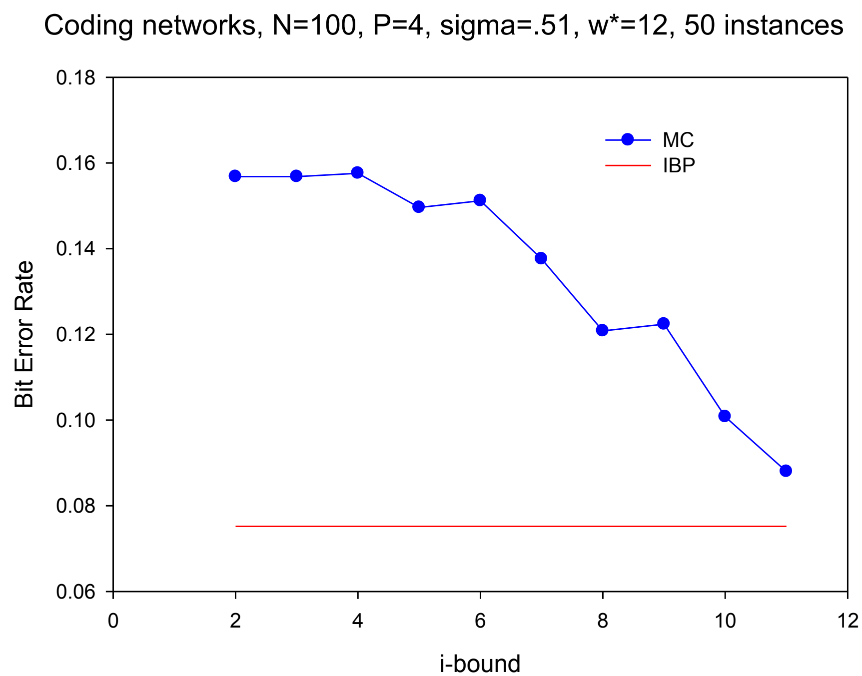


evidence=0

evidence=10

Coding networks - Bit Error Rate





sigma=0.22

sigma=.51

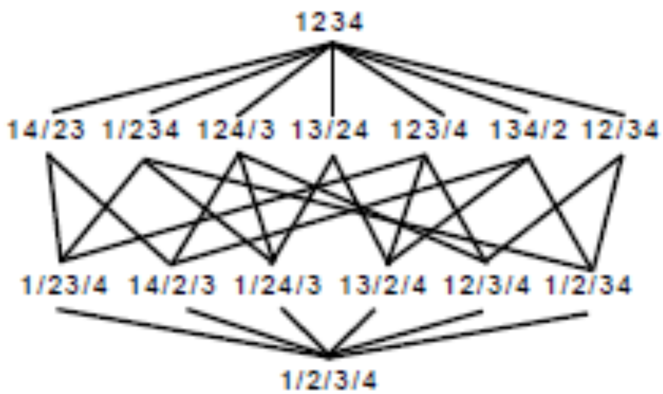
Heuristic for partitioning

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 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 is quite fast, its shortcoming is that it does not consider the actual information in the functions.

Content-based heuristics
(Rollon and Dechter 2010)



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

- *Log relative error*:

$$RE(f,h) = \sum_t (\log(f(t)) - \log(h(t)))$$

- *Max log relative error*:

$$MRE(f,h) = \max_t \{\log(f(t)) - \log(h(t))\}$$

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

Queries

- **Probability of evidence (or partition function)**

$$P(e) = \sum_{X - \text{var}(e)} \prod_{i=1}^{n} P(x_i \mid pa_i)|_e \qquad Z = \sum_X \prod_i \psi_i(C_i)$$

- **Posterior marginal (beliefs):**

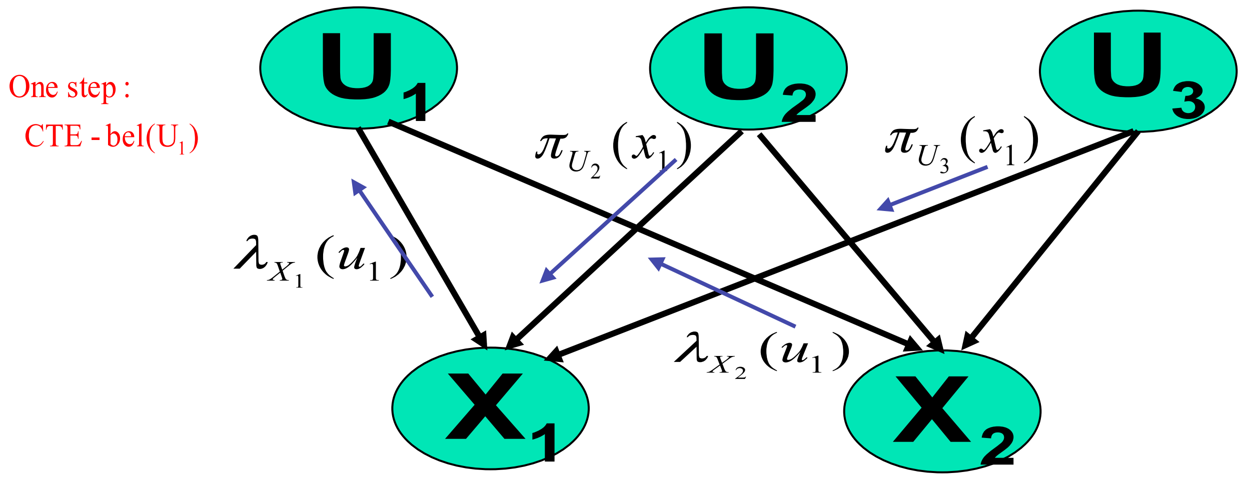
$$P(x_i \mid e) = \frac{P(x_i, e)}{P(e)} = \frac{\sum_{X - \text{var}(e) - X_i} \prod_{j=1}^{n} P(x_j \mid pa_j)|_e}{\sum_{X - \text{var}(e)} \prod_{j=1}^{n} P(x_j \mid pa_j)|_e}$$

- **Most Probable Explanation**

$$\bar{x}^* = \arg\max_{\bar{x}} P(\bar{x}, e)$$

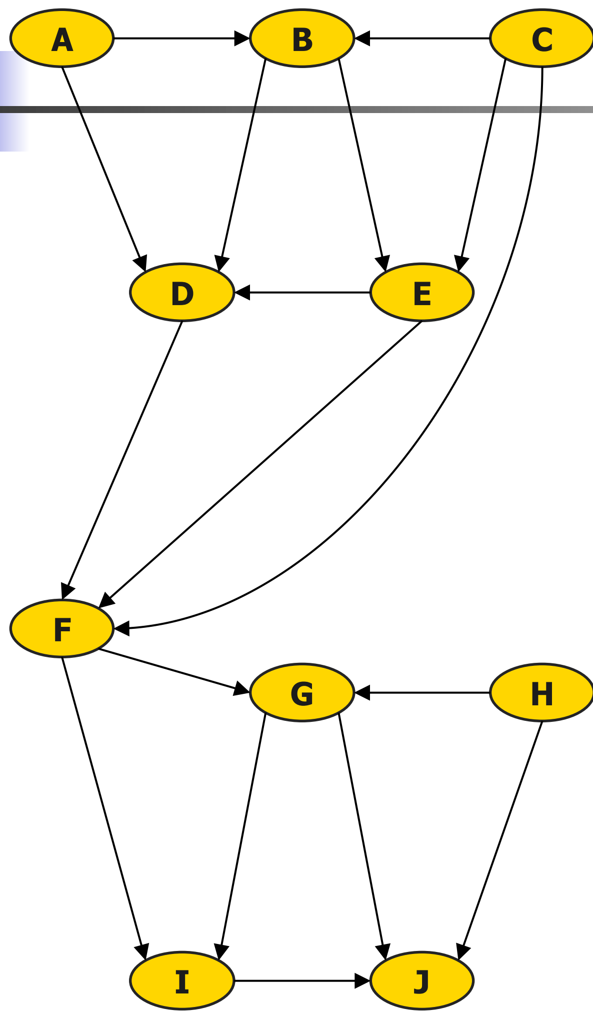
Iterative Belief Proapagation

- Belief propagation is exact for poly-trees
- IBP - applying BP iteratively to cyclic networks

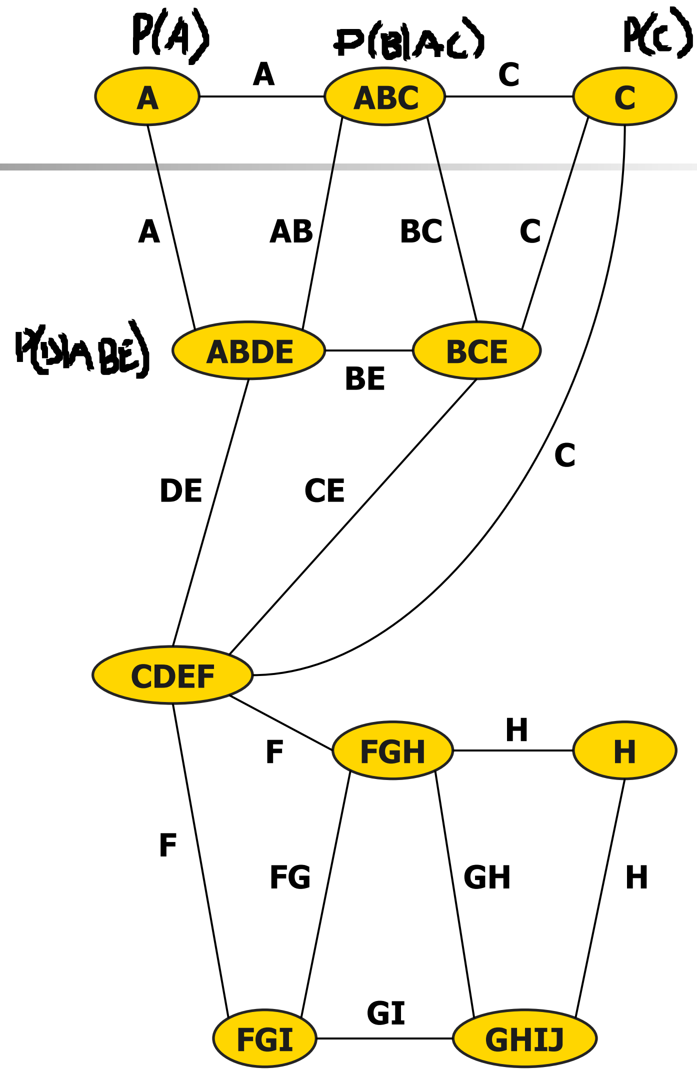


- No guarantees for convergence
- Works well for many coding networks

IJGP - Example

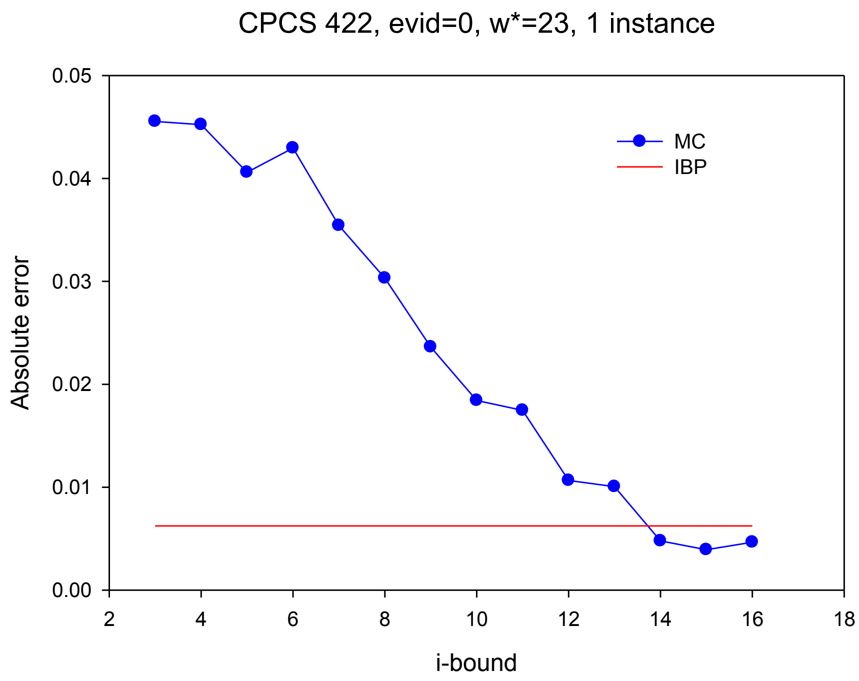


Belief network

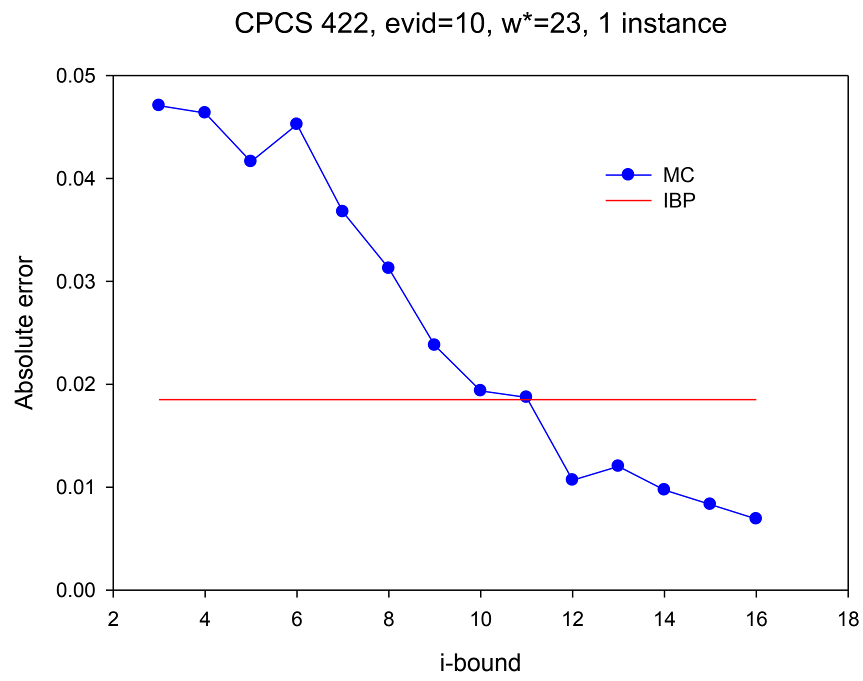


Loopy BP graph

CPCS422 - Absolute error



evidence=0



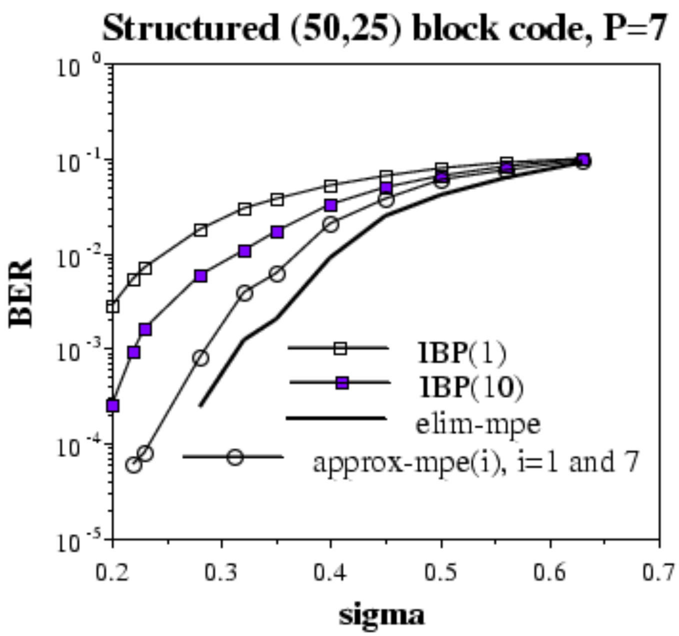
evidence=10

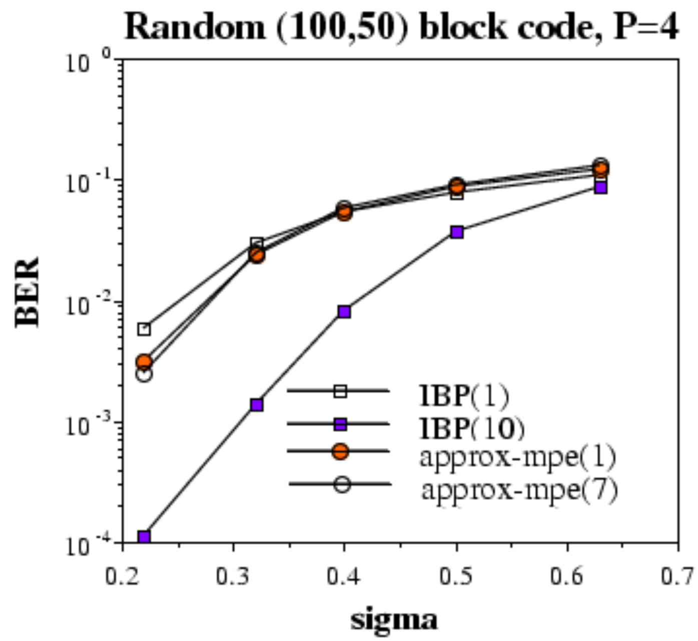
MBE-mpe vs. IBP

mbe - 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):





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

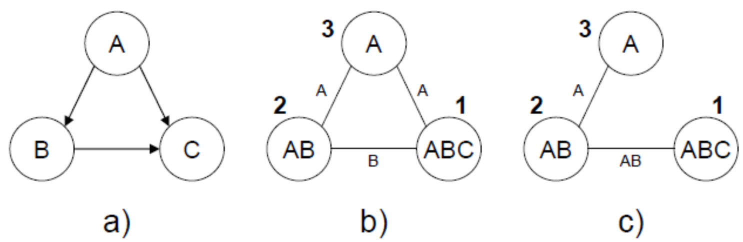


Figure 1.17: a) A belief network; b) A dual join-graph with singleton labels; c) A dual join-graph which is a join-tree

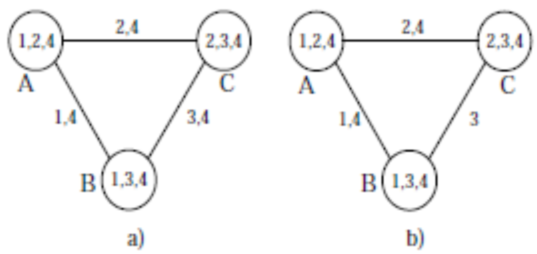
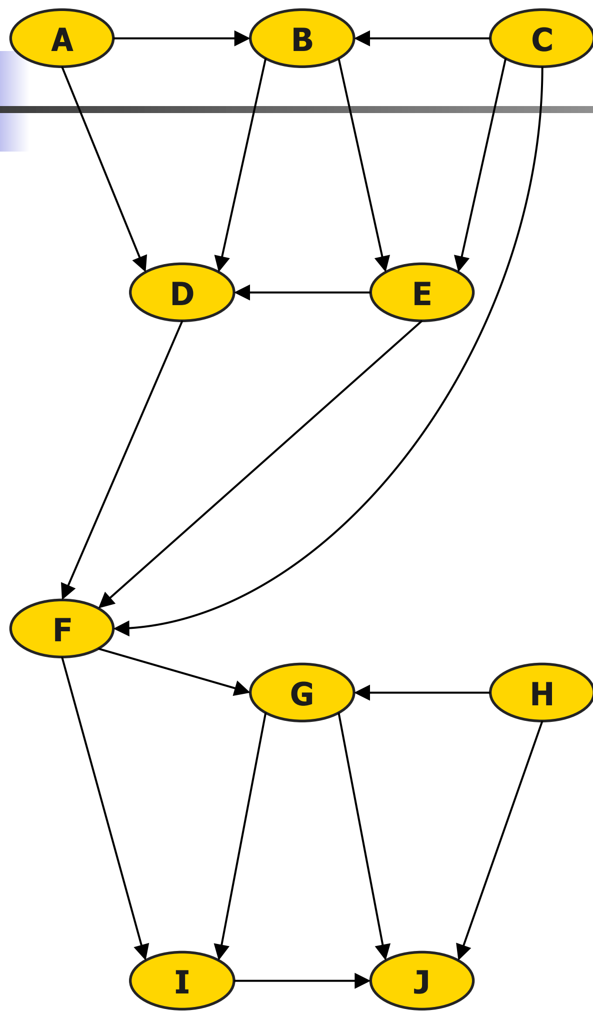
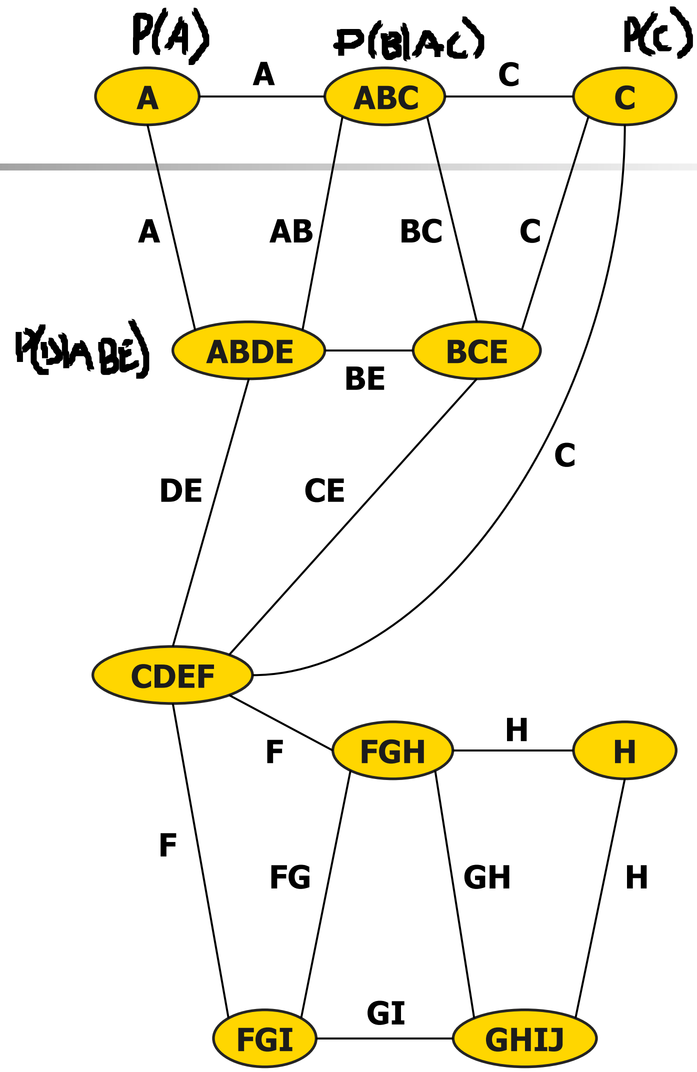


Figure 1.15: An arc-labeled decomposition

IJGP - Example

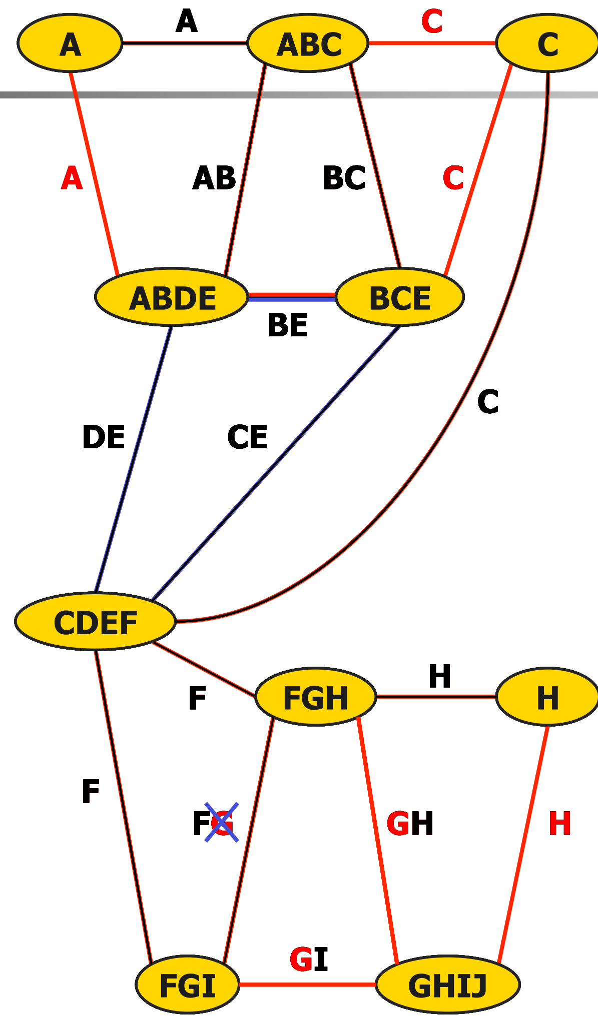


Belief network



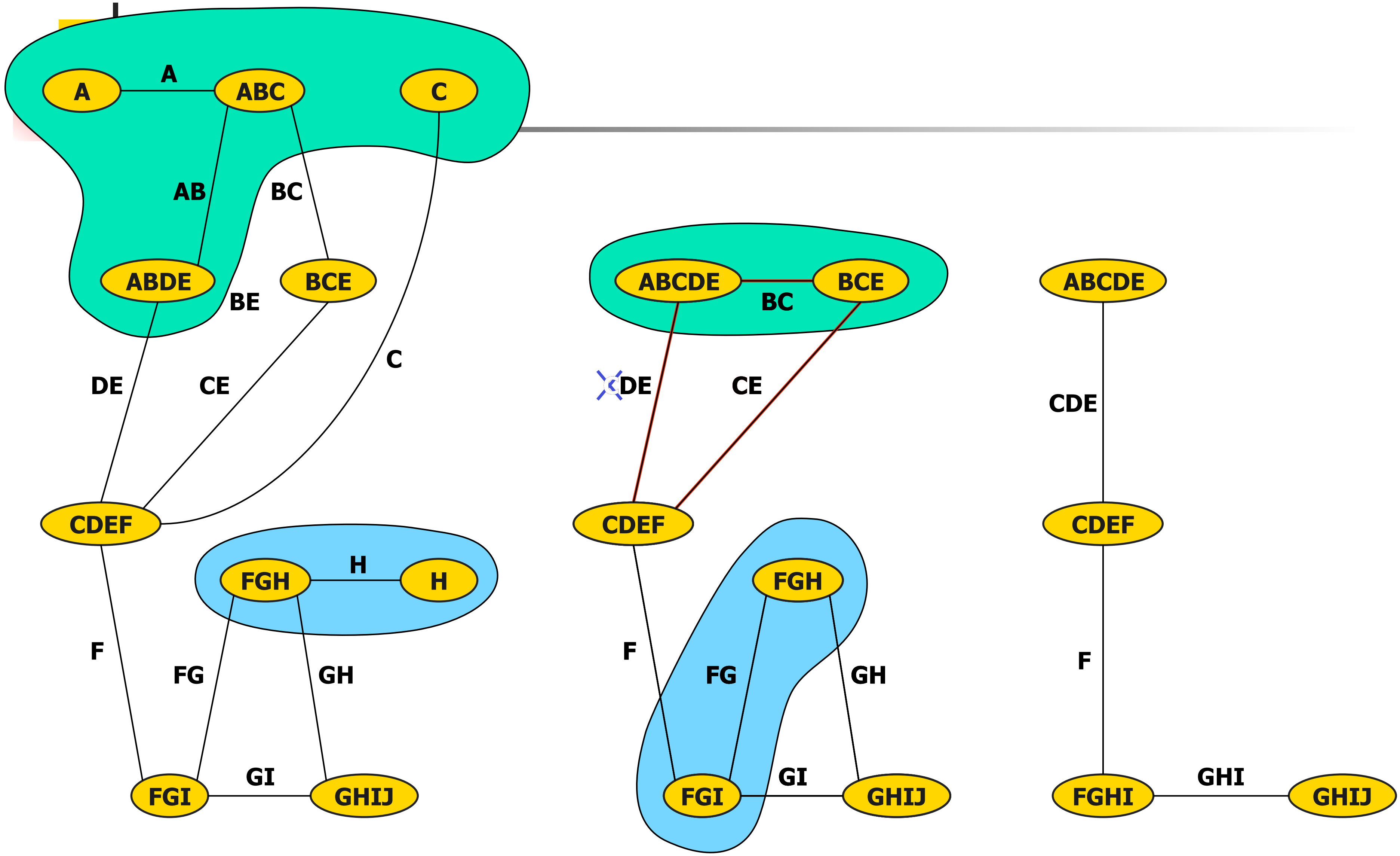
Loopy BP graph

Arc-Minimal Join-Graph

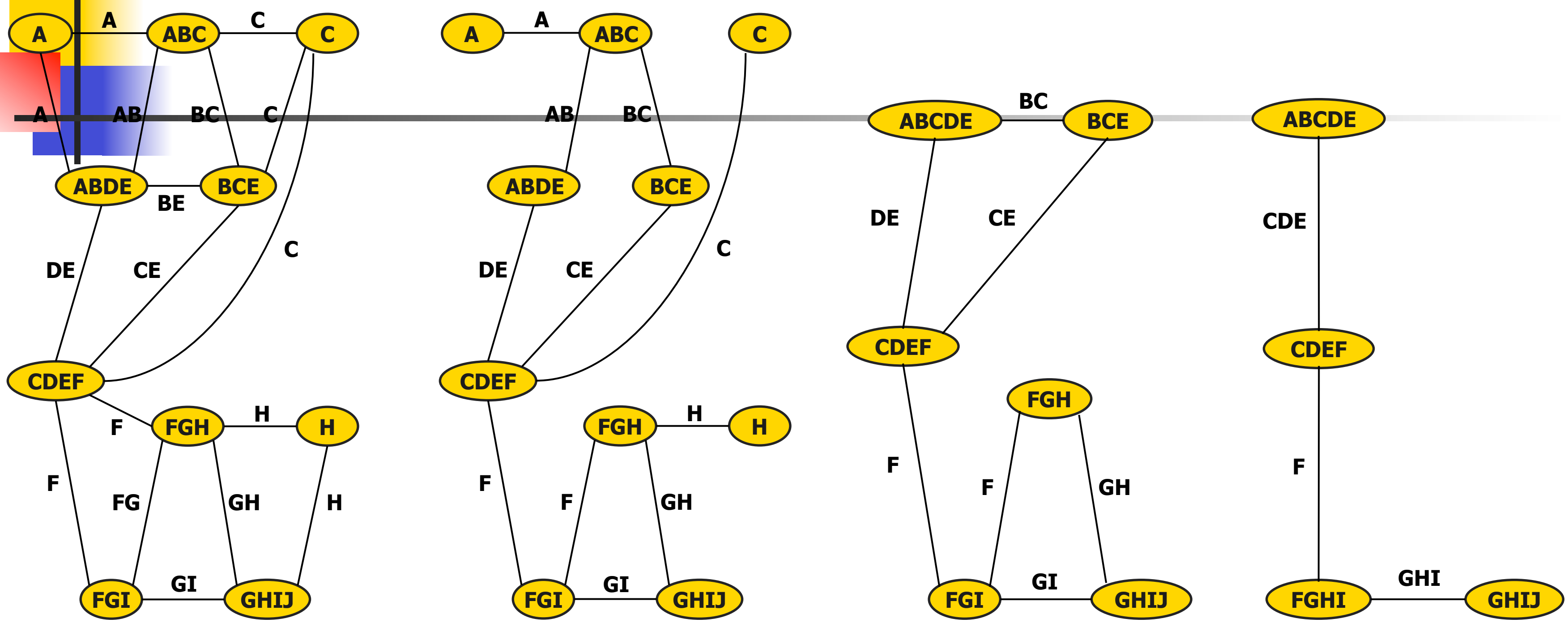


Arcs labeled with any single variable should form a TREE

Collapsing Clusters



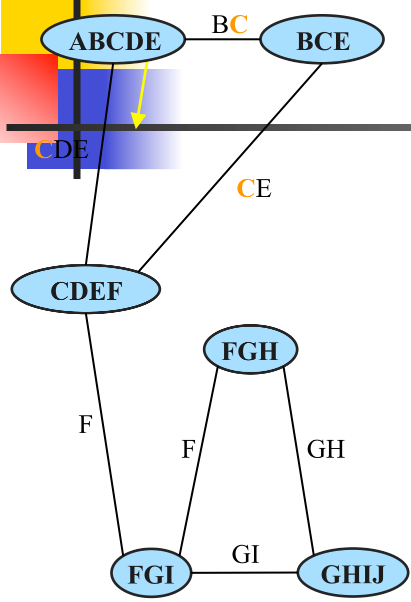
Join-Graphs

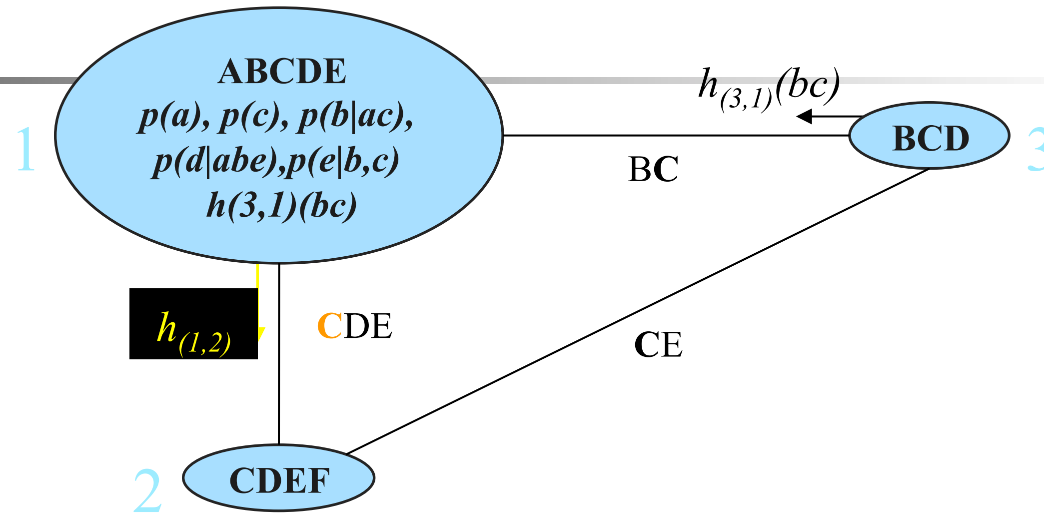


more accuracy

less complexity

Message propagation





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

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

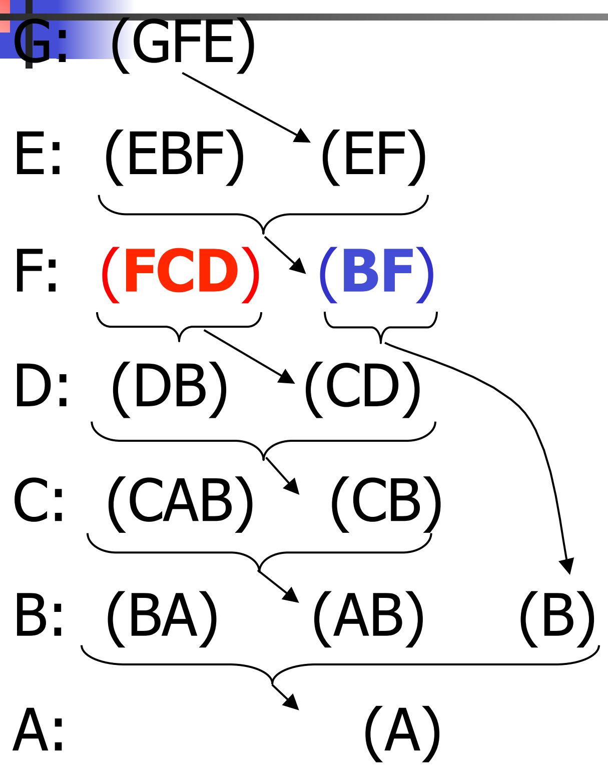
Non-minimal arc-labeled:
sep(1,2)={C,D,E}
elim(1,2)={A,B}

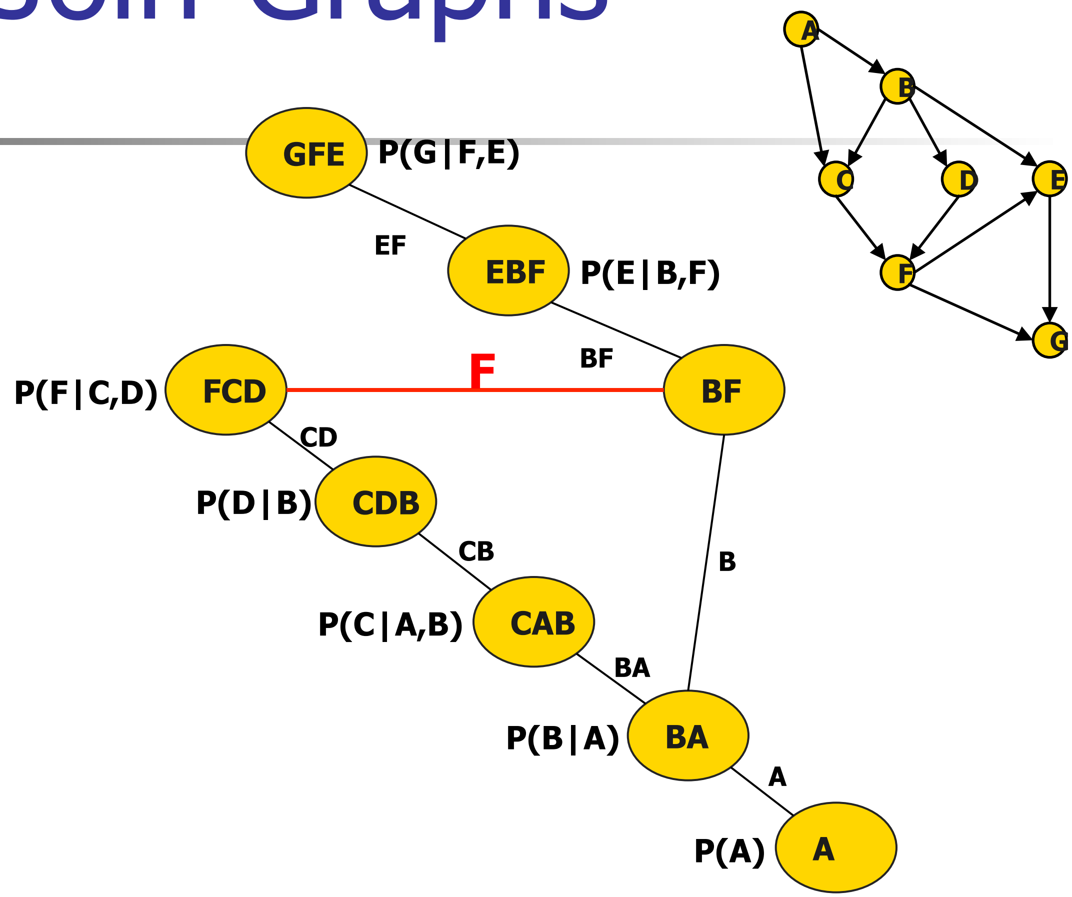
$$h_{(1,2)}(cde) = \sum_{a,b} p(a)p(c)p(b|ac)p(d|abe)p(e|bc)h_{(3,1)}(bc)$$

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





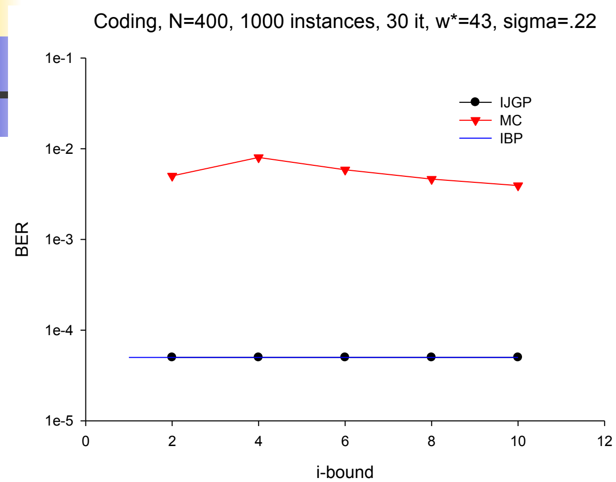
a) schematic mini-bucket(i), i=3

b) arc-labeled join-graph decomposition

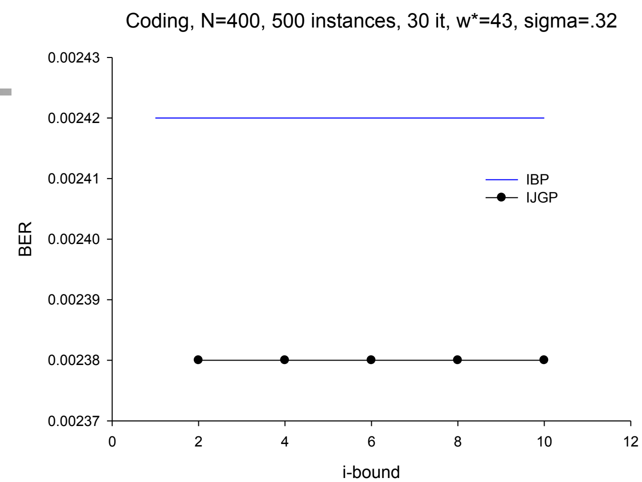
Empirical evaluation

- 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

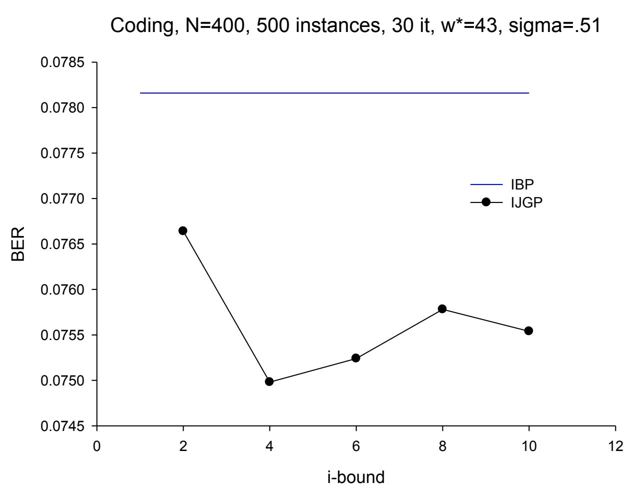
Coding networks - BER



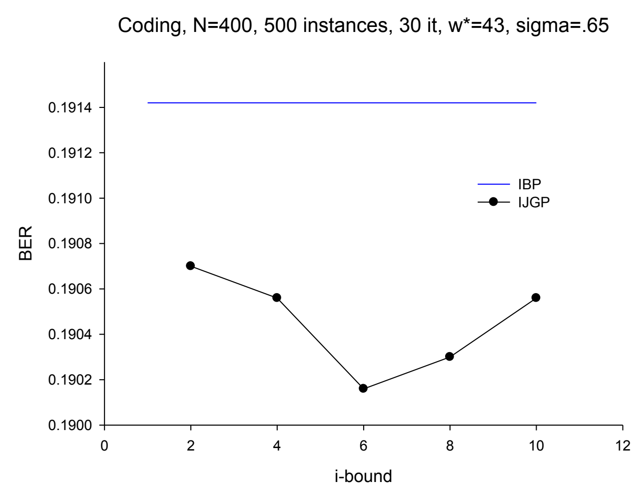
sigma=.22



sigma=.32

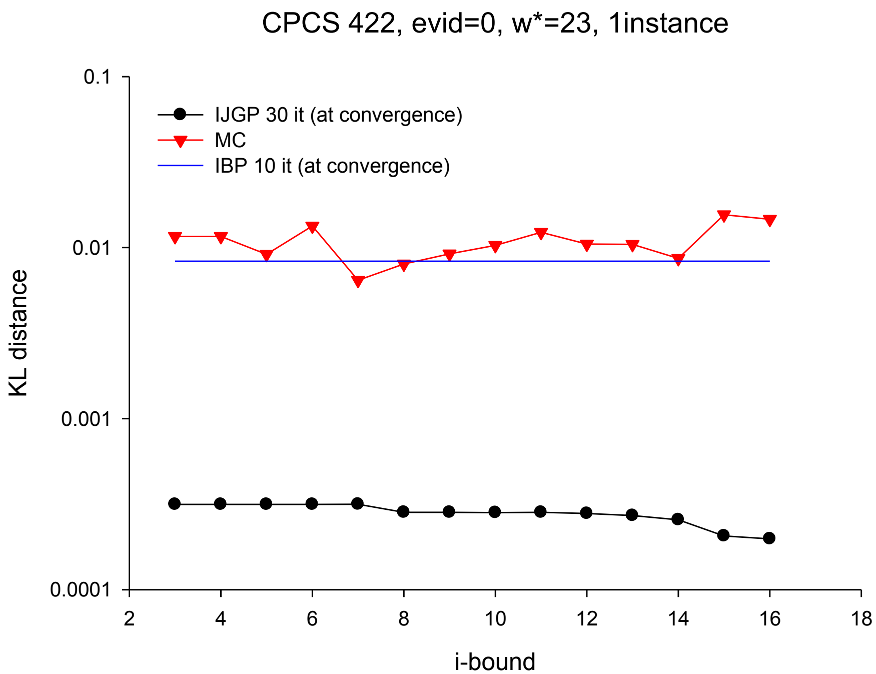


sigma=.51

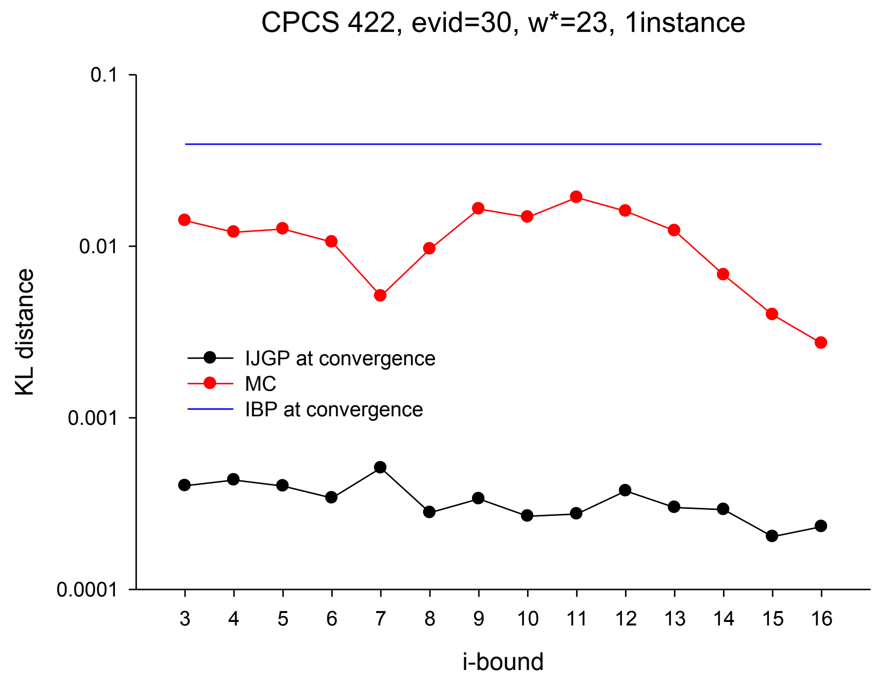


sigma=.65

CPCS 422 – KL Distance

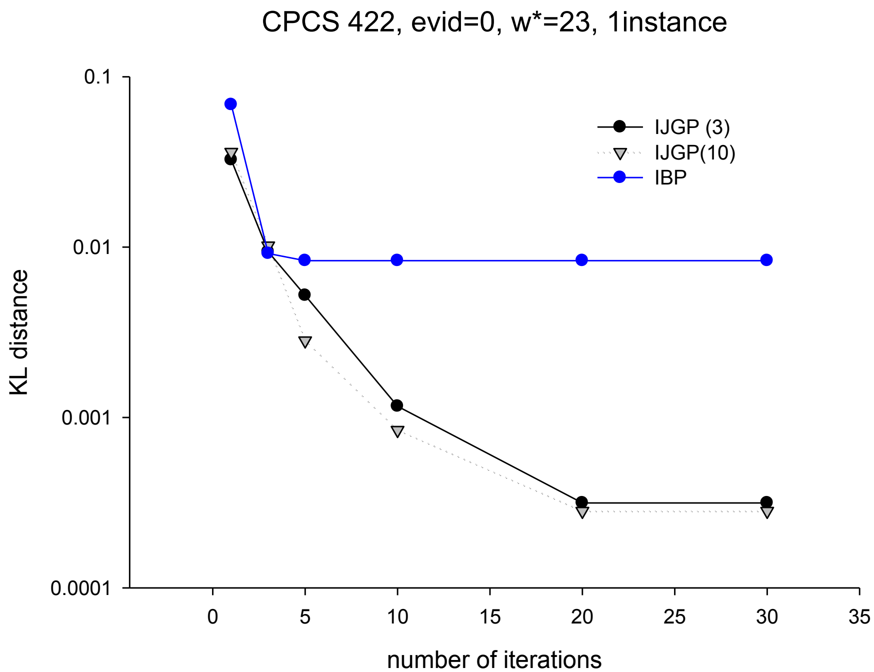


evidence=0

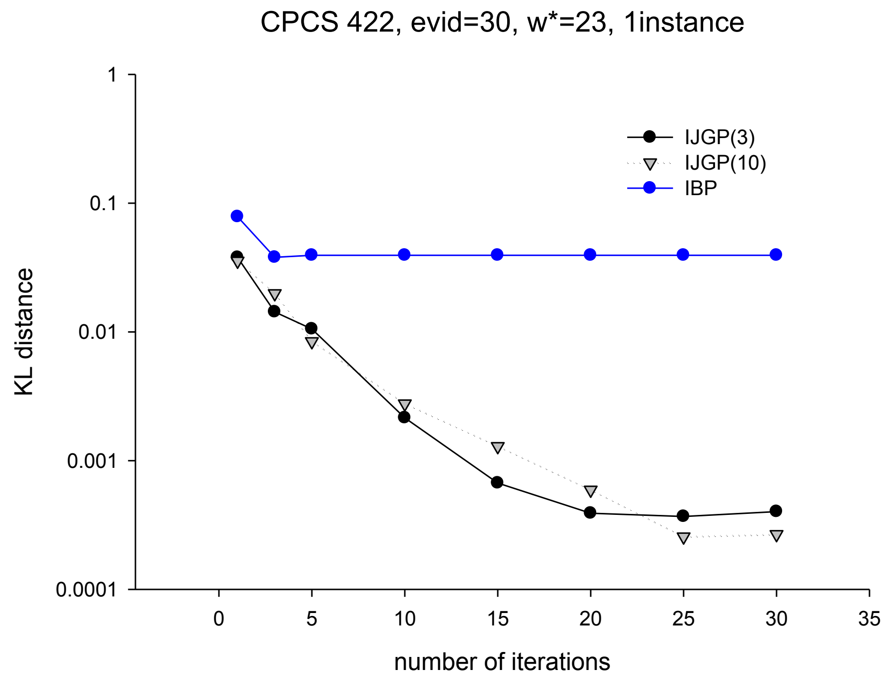


evidence=30

CPCS 422 – KL vs. Iterations

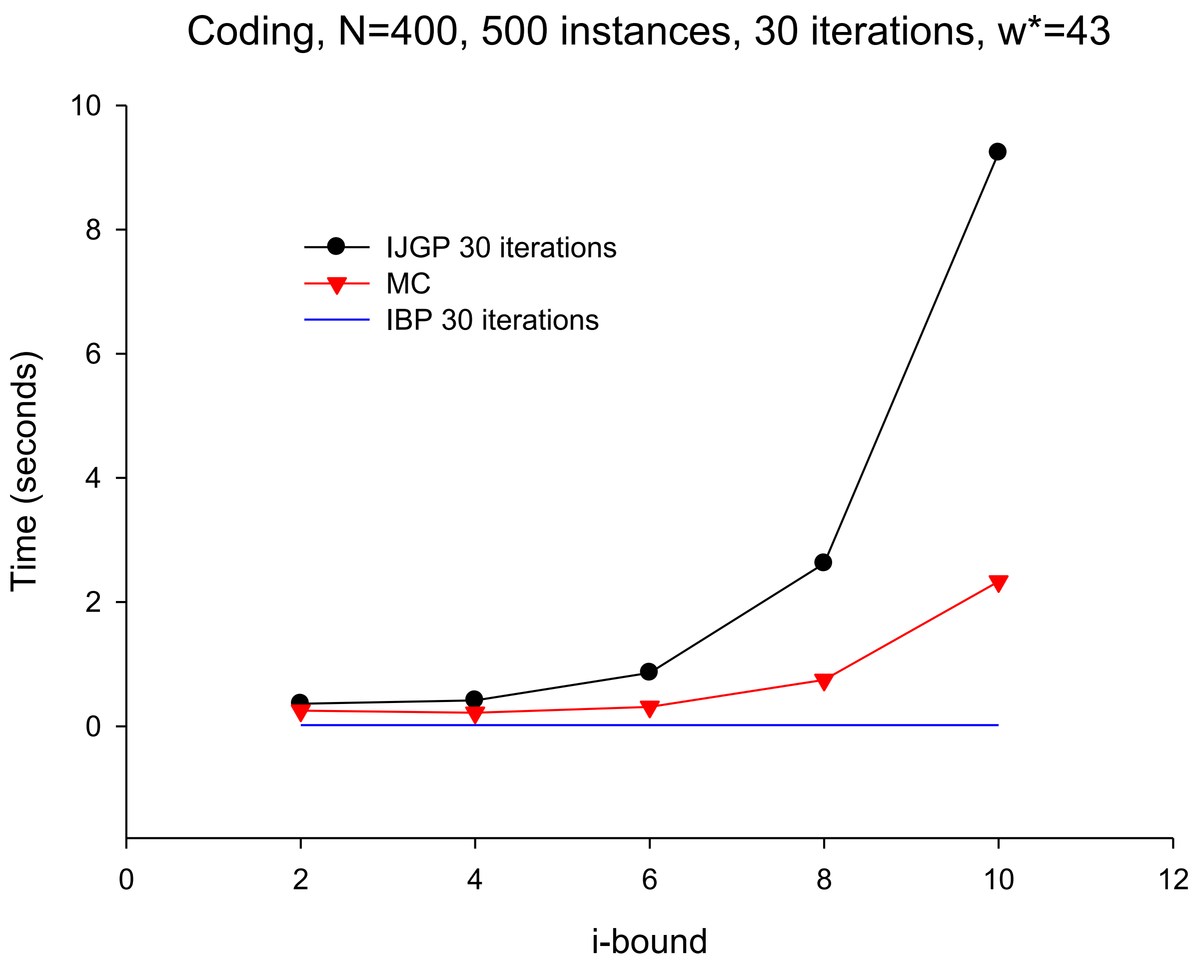


evidence=0



evidence=30

Coding networks - Time



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: $O(deg \bullet (n+N) \bullet d^{i+1})$
  - space: $O(N \bullet d^{\theta})$

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 converges
- If IJGP converges, the normalizing constants are unique

Join-graph decomposition

Definition 1 (join-graph decompositions) *A* join-graph decomposition JG *for* $\mathcal{M} =< X, D, F, \otimes, \Downarrow>$ *is a triple* $\mathcal{JG} =< G, \chi, \psi >$*, where* $G = (V, E)$ *is a graph, and* χ *and* ψ *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* $\mathcal{JG} =< G, \chi, \Psi >$, $G = (V, E)$ *of a graphical model* $\mathcal{M} =< X, D, F >$, *then* $\mathcal{JG}$ *is pairwise-consistent (pwc) relative to a set of messages* $H = \{h_{u \to v}, h_{v \to u} | (u, v) \in E\}$, *iff for every* $(u, v) \in E$ *we have:*

$$\sum_{\chi(u) - \chi(uv)} \psi_u \cdot \prod_{h \in H_u} h = \sum_{\chi(v) - \chi(uv)} \psi_v \cdot \prod_{h \in H_v} h \tag{1}$$

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

$$b(x_u) = \psi_u(x_u) \cdot \prod_{h \in H_u} h(x_u) \tag{2}$$

$$b_{uv}(x_{uv}) = \sum_{\chi(u) - \chi_{uv}} b_u(x_u) \tag{3}$$

Pseudo marginals

DEFINITION 5 (p-marginal functions) *Given a graphical model for* $\mathcal{M} =< X, D, F >$, *the* p-marginal function *of* $\mathcal{M}$ *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:*

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

3

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: $h_{u \to v} \leftarrow 1$.

Repeat

For every $u \in G$ do

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

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

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.*

I. If algorithm PWC converged then its output JG_H *is PWC.*

Proof

Proof. Part a: If the algorithm converges then from Eq. 5 it follows that the messages satisfy:

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

From this, multiplying both sides by $h_{v\to u}$ we get

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

Exchanging u and v everywhere we get also that

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

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.

□

Symmetry and pwc

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

$$\sum_{\chi(u)-\chi(uv)} \psi_u(X_u) \cdot \prod_{k \neq (v)} h_{k \to v}(X_{ku}) = \sum_{\chi(v)-\chi(uv)} \psi_v(x_u) \cdot \prod_{k \neq (u)} h_{k \to u}(X_{kv}) \tag{7}$$

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

$$b_H(x_{uv}) = h_{u \to v}(x_{uv}) \cdot h_{v \to u}(x_{vu}) \tag{8}$$

Fixed point iff symmetry

THEOREM 1 *Given a join-graph decomposition* $\mathcal{JG} =< G, \chi, \Psi >$, $G = (V, E)$ *of a graphical model* $\mathcal{M} =< X, D, F >$ *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.

Symmetry -→ 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 $b_H(Y) = P(Y) \neq 1$. □

PWC and Normalizing constants

PROPOSITION 1 *A joingraph is pwc relative to $\mathcal{H}$ iff we have:*

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

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*

$$K(u) = \sum_{x_u} b_u(x_u) \tag{5}$$

$$K(uv) = \sum_{x_u} b_{uv}(x_{uv}) \tag{6}$$

PWC implies unique normalizing constants

THEOREM 1 *If* $\mathcal{JG} =< G, \chi, \Psi >$ *is pwc relative to messages* $\mathcal{H}$ *then,* $\forall, u, v, \in V, (u, v) \in E$

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

Proof. If $\mathcal{JG} =< G, \chi, \Psi >$ is pwc relative to messages $\mathcal{H}$ then

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

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

and because of pwc holds

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

□

Repatameterization

$$Q(x) = \frac{\prod_{v \in V} b_H(x_v)}{\prod_{(u,v) \in E} h_{u \to v}(x_{uv}) \cdot h_{v \to u}(x_{vu})}$$

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)

$$\mathrm{KL}(\Pr'(\mathbf{X}|\mathbf{e}), \Pr(\mathbf{X}|\mathbf{e})) = \sum_{\mathbf{x}} \Pr'(\mathbf{x}|\mathbf{e}) \log \frac{\Pr'(\mathbf{x}|\mathbf{e})}{\Pr(\mathbf{x}|\mathbf{e})}$$

- $\mathrm{KL}(\Pr'(\mathbf{X}|\mathbf{e}), \Pr(\mathbf{X}|\mathbf{e}))$ is non-negative
- equal to zero if and only if $\Pr'(\mathbf{X}|\mathbf{e})$ and $\Pr(\mathbf{X}|\mathbf{e})$ are equivalent.

The Kullback-Leibler Divergence

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

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

- $\mathrm{KL}(\Pr'(\mathbf{X}|\mathbf{e}), \Pr(\mathbf{X}|\mathbf{e}))$ 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 $\Pr(\mathbf{X})$ be a distribution induced by a Bayesian network $\mathcal{N}$ having families $X\mathbf{U}$

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

$$\begin{aligned} & \mathrm{KL}(\Pr'(\mathbf{X}|\mathbf{e}), \Pr(\mathbf{X}|\mathbf{e})) \\ = \ & -\mathrm{ENT}'(\mathbf{X}|\mathbf{e}) - \sum_{X\mathbf{U}} \mathrm{AVG}'(\log \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}}) + \log \Pr(\mathbf{e}), \end{aligned}$$

where

- $\mathrm{ENT}'(\mathbf{X}|\mathbf{e}) = -\sum_{\mathbf{x}} \Pr'(\mathbf{x}|\mathbf{e}) \log \Pr'(\mathbf{x}|\mathbf{e})$ is the entropy of the conditioned approximate distribution $\Pr'(\mathbf{X}|\mathbf{e})$.
- $\mathrm{AVG}'(\log \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}}) = \sum_{x\mathbf{u}} \Pr'(x\mathbf{u}|\mathbf{e}) \log \lambda_{\mathbf{e}}(x)\theta_{x|\mathbf{u}}$ is a set of expectations over the original network parameters weighted by the conditioned approximate distribution.

The Kullback-Leibler Divergence

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

$$\mathrm{ENT}'(\mathbf{X}|\mathbf{e}) + \sum_{X\mathbf{U}} \mathrm{AVG}'(\log \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}})$$

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

- $\Pr'(\mathbf{X}|\mathbf{e})$ should match the original distribution by giving more weight to more likely parameters $\lambda_{\mathbf{e}}(x)\theta_{x|\mathbf{u}}$ (i.e, maximize the expectations).
- $\Pr'(\mathbf{X}|\mathbf{e})$ 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 $\mathrm{Pr}'(\mathbf{X})$ that factors as follows:

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

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

$$\mathrm{ENT}'(\mathbf{X}|\mathbf{e}) = -\sum_{X\mathbf{U}} \sum_{x\mathbf{u}} \mathrm{Pr}'(x\mathbf{u}|\mathbf{e}) \log \frac{\mathrm{Pr}'(x\mathbf{u}|\mathbf{e})}{\prod_{u \sim \mathbf{u}} \mathrm{Pr}'(u|\mathbf{e})}$$

Optimizing the KL-Divergence

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

$$\begin{aligned} & \mathrm{ENT}'(\mathbf{X}|\mathbf{e}) + \sum_{X\mathbf{U}} \mathrm{AVG}'(\log \lambda_{\mathbf{e}}(X)\Theta_{X|\mathbf{U}}) \\ = \quad & -\sum_{X\mathbf{U}} \sum_{x\mathbf{u}} \mu_{x\mathbf{u}} \log \frac{\mu_{x\mathbf{u}}}{\prod_{u \sim \mathbf{u}} \mu_u} + \sum_{X\mathbf{U}} \sum_{x\mathbf{u}} \mu_{x\mathbf{u}} \log \lambda_{\mathbf{e}}(x)\theta_{x|\mathbf{u}}, \end{aligned}$$

under normalization constraints:

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

for each family $X\mathbf{U}$ and parent U, and under consistency constraints:

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

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

Optimizing the KL-Divergence

- 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 $\Pr'$ whose factorizations are more expressive than that of the polytree-based factorization.

Generalized Belief Propagation

If a distribution $\Pr'$ has the form:

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

then its entropy has the form:

$$\mathrm{ENT}'(\mathbf{X}|\mathbf{e}) = \sum_{\mathbf{C}} \mathrm{ENT}'(\mathbf{C}|\mathbf{e}) - \sum_{\mathbf{S}} \mathrm{ENT}'(\mathbf{S}|\mathbf{e}).$$

When the marginals $\Pr'(\mathbf{C}|\mathbf{e})$ and $\Pr'(\mathbf{S}|\mathbf{e})$ are readily available, the ENT component of the KL–divergence can be computed efficiently.

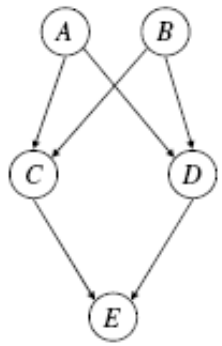
Joingraphs

While a jointree induces an exact factorization of a distribution, a joingraph G induces an approximate factorization:

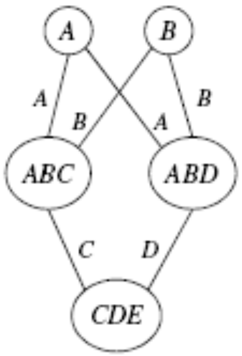
$$\Pr'(\mathbf{X}|\mathbf{e}) = \frac{\prod_i \Pr'(\mathbf{C}_i|\mathbf{e})}{\prod_{ij} \Pr'(\mathbf{S}_{ij}|\mathbf{e})}$$

which is a product of cluster marginals over a product of separator marginals. When the joingraph corresponds to a jointree, the above factorization will be exact.

Joingraphs



Bayesian network



dual joingraph

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 $X\mathbf{U}$ in network $\mathcal{N}$, the corresponding node i in joingraph G will have the cluster $\mathbf{C}_i = X\mathbf{U}$.
- For each $U \rightarrow X$ in network $\mathcal{N}$, the corresponding edge i–j in joingraph G will have the separator $\mathbf{S}_{ij} = U$.

terative Joingraph Propagation

Computing cluster marginals $\mu_{\mathbf{c}_i} = \Pr'(\mathbf{c}_i|\mathbf{e})$ and separator marginals $\mu_{\mathbf{s}_{ij}} = \Pr'(\mathbf{s}_{ij}|\mathbf{e})$ that minimize the KL–divergence between $\Pr'(\mathbf{X}|\mathbf{e})$ and $\Pr(\mathbf{X}|\mathbf{e})$

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

Iterative Joingraph Propagation

$\text{IJGP}(G, \Phi)$

input:

G: a joingraph

Φ: factors assigned to clusters of G

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

main:

```
t ← 0
initialize all messages M^t_ij (uniformly)
while messages have not converged do
    t ← t + 1
    for each joingraph edge i–j do
        M^t_ij ← η Σ_{C_i \ S_ij} Φ_i Π_{k≠j} M^{t−1}_ki
        M^t_ji ← η Σ_{C_j \ S_ij} Φ_j Π_{k≠i} M^{t−1}_kj
    end for
end while
return BEL(C_i) ← η Φ_i Π_k M^t_ki for each node i
```

terative Joingraph Propagation

Let $\Pr(\mathbf{X})$ be a distribution induced by a Bayesian network $\mathcal{N}$ having families $X\mathbf{U}$, and let $\mathbf{C}_i$ and $\mathbf{S}_{ij}$ 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 $\mu_{\mathbf{c}_i} = BEL(\mathbf{c}_i)$ and $\mu_{\mathbf{s}_{ij}} = BEL(\mathbf{s}_{ij})$ are a stationary point of:

$$\begin{aligned} & \mathrm{ENT}'(\mathbf{X}|\mathbf{e}) + \sum_{\mathbf{C}_i} \mathrm{AVG}'(\log \Phi_i) \\ = \quad & -\sum_{\mathbf{C}_i}\sum_{\mathbf{c}_i} \mu_{\mathbf{c}_i} \log \mu_{\mathbf{c}_i} + \sum_{\mathbf{S}_{ij}}\sum_{\mathbf{s}_{ij}} \mu_{\mathbf{s}_{ij}} \log \mu_{\mathbf{s}_{ij}} + \sum_{\mathbf{C}_i}\sum_{\mathbf{c}_i} \mu_{\mathbf{c}_i} \log \Phi_i(\mathbf{c}_i), \end{aligned}$$

under normalization constraints:

$$\sum_{\mathbf{c}_i} \mu_{\mathbf{c}_i} = \sum_{\mathbf{s}_{ij}} \mu_{\mathbf{s}_{ij}} = 1$$

for each cluster $\mathbf{C}_i$ and separator $\mathbf{S}_{ij}$, and under consistency constraints:

$$\sum_{\mathbf{c}_i \sim \mathbf{s}_{ij}} \mu_{\mathbf{c}_i} = \mu_{\mathbf{s}_{ij}} = \sum_{\mathbf{c}_j \sim \mathbf{s}_{ij}} \mu_{\mathbf{c}_j}$$

for each separator $\mathbf{S}_{ij}$ and neighboring clusters $\mathbf{C}_i$ and $\mathbf{C}_j$.

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 $\Leftrightarrow$ inconsistent assignments
- ε -small beliefs – experimental study

Constraint networks

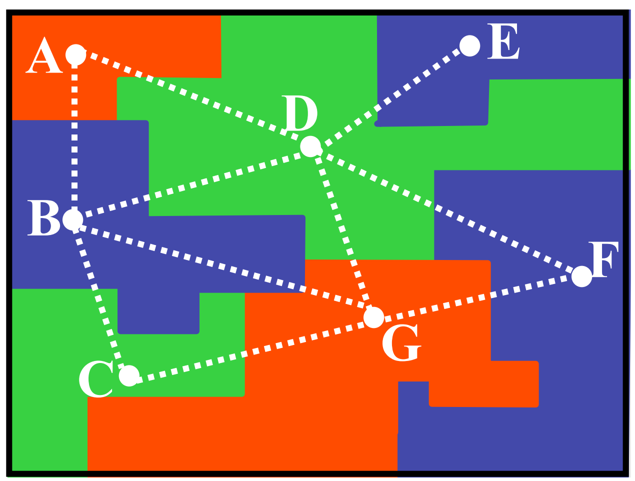
Map coloring

Variables: countries (A B C etc.)

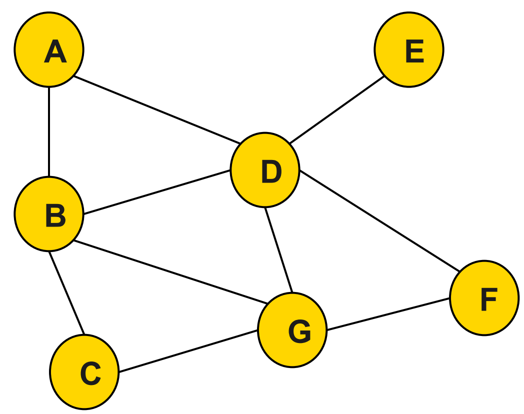
Values: colors (red green blue)

Constraints: **A ≠ B, A ≠ D, D ≠ E**, *etc.*

A	B
red	green
red	yellow
green	red
green	yellow
yellow	green
yellow	red

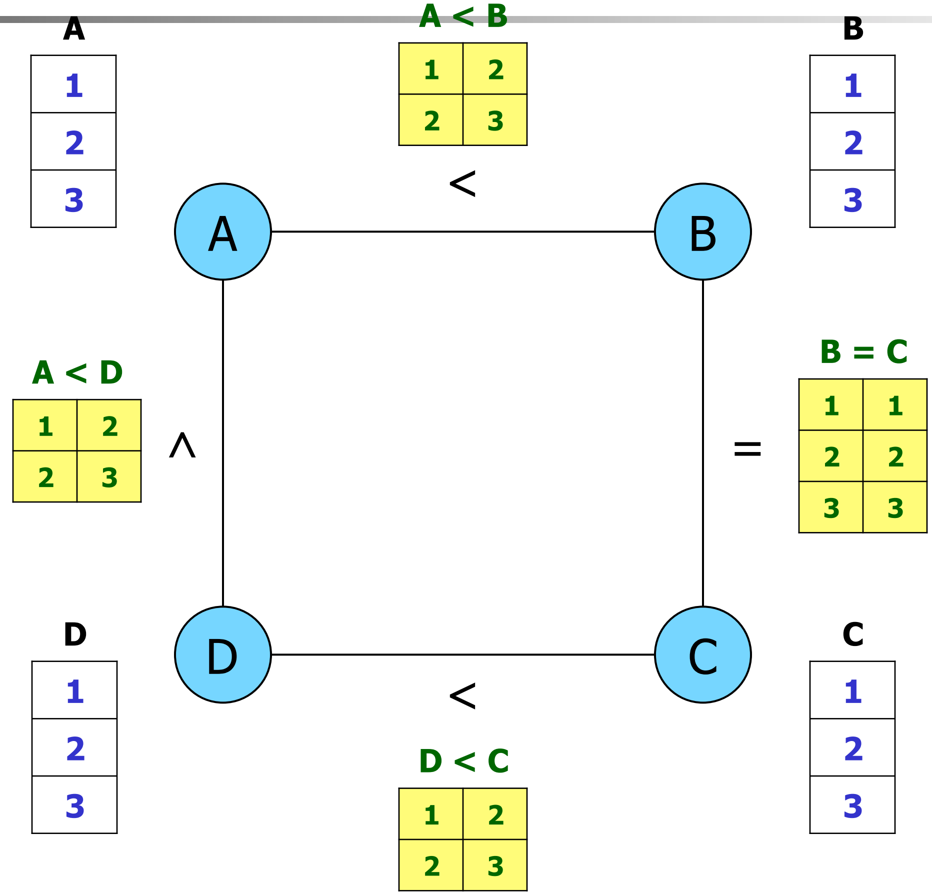


Constraint graph

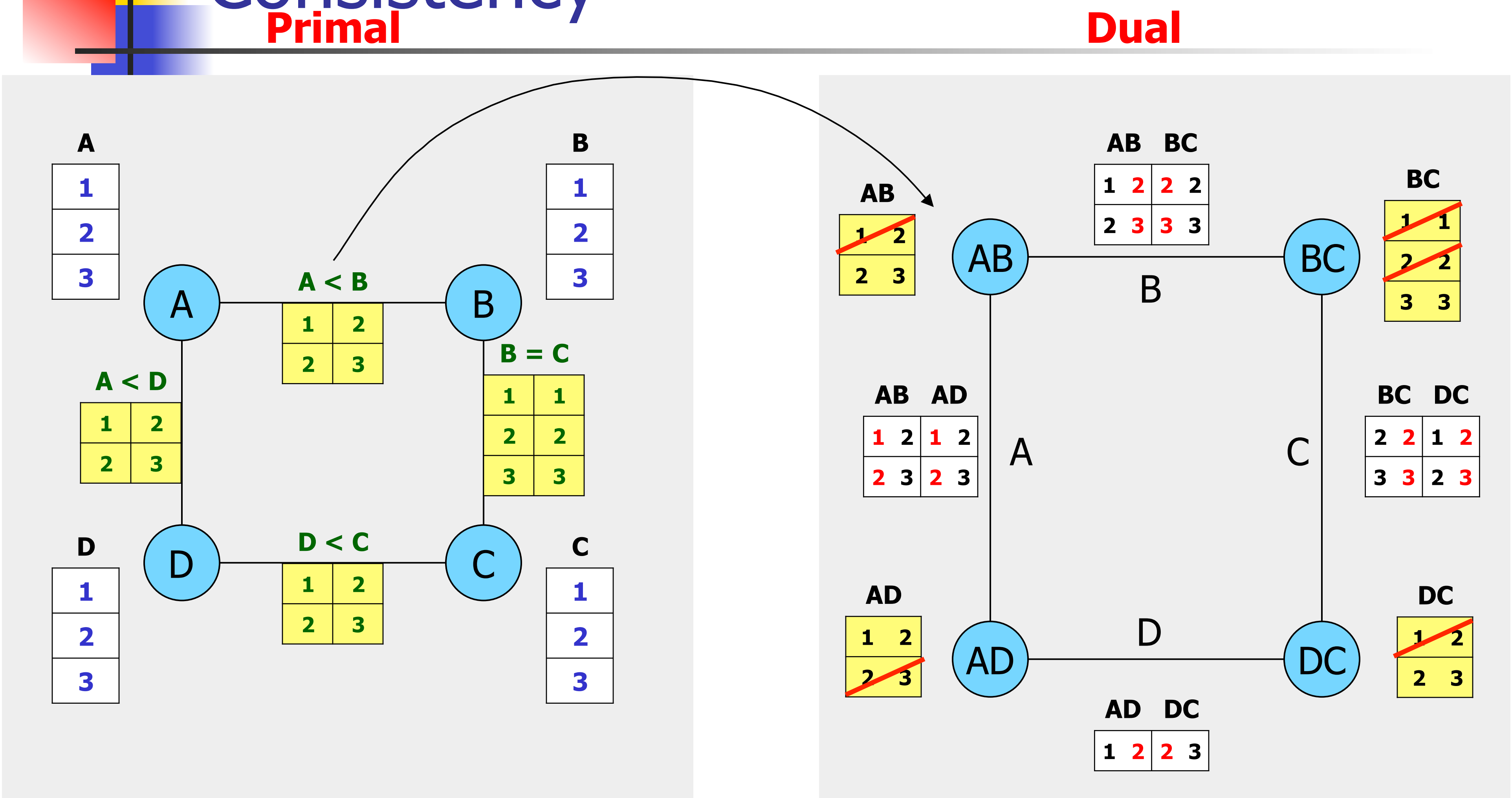


Arc-consistency

- Sound
- Incomplete
- Always converges (polynomial)



Relational Distributed Arc-Consistency

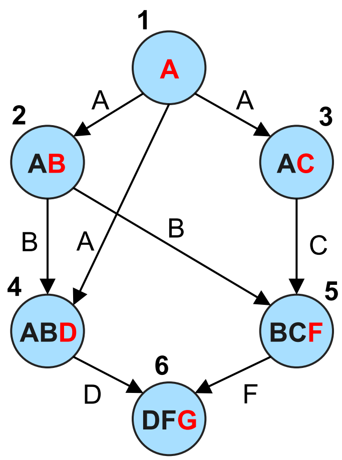


Flattening the Bayesian Network

A	P(A)
1	.2
2	.5
3	.3
…	0

A	B	P(B|A)
1	2	.3
1	3	.7
2	1	.4
2	3	.6
3	1	.1
3	2	.9
…	…	0

A	C	P(C|A)
1	2	1
3	2	1
…	…	0



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

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

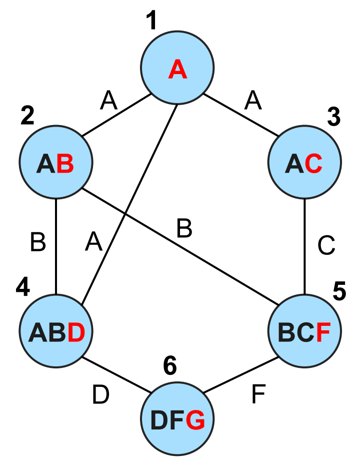
D	F	G	P(G|D,F)
1	2	3	1
2	1	3	1
…	…	…	0

Belief network

A
1
2
3

A	B
1	2
1	3
2	1
2	3
3	1
3	2

A	C
1	2
3	2



A	B	D
1	2	3
1	3	2
2	1	3
2	3	1
3	1	2
3	2	1

B	C	F
1	2	3
3	2	1

D	F	G
1	2	3
2	1	3

Flat constraint network

Belief Zero Propagation = Arc-Consistency

$$h_i^j = \sum_{elim(i,j)} \left(p_i \cdot \left(\prod_{\{k \in ne_j(i)\}} h_k^i\right)\right)$$

$$h_i^j = \pi_{l_{ij}} \left(R_i \bowtie \left(\bowtie_{k \in ne(i)} h_k^i\right)\right)$$

A	B	P(B|A)
1	2	>0
1	3	>0
2	1	>0
2	3	>0
3	1	>0
3	2	>0
…	…	0

h_1^2

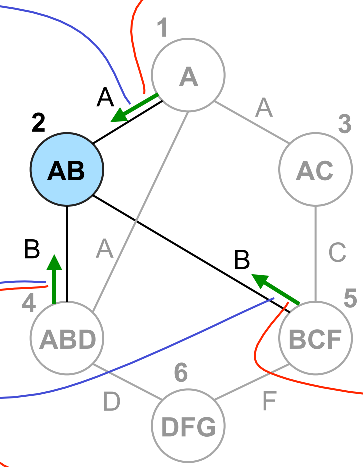
A	$h_1^2(A)$
1	>0
2	>0
3	>0
…	0

h_4^2

B	$h_1^2(B)$
1	>0
2	>0
3	>0
…	0

h_5^2

B	$h_1^2(B)$
1	>0
3	>0
…	0



A	B
1	2
1	3
2	1
2	3
3	1
3	2

h_1^2

A
1
2
3

h_4^2

B
1
2
3

h_5^2

B
1
3

Updated belief:

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

=

A	B	Bel (A,B)
1	3	>0
2	1	>0
2	3	>0
3	1	>0
…	…	0

Updated relation:

$$R(A,B) = R(A,B) \bowtie h_1^2 \bowtie h_4^2 \bowtie h_5^2 =$$

=

A	B
1	3
2	1
2	3
3	1

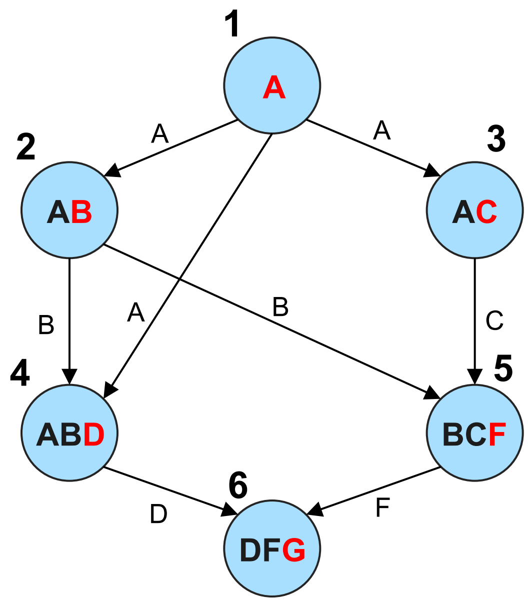
Flat Network - Example

R_1

A	P(A)
1	.2
2	.5
3	.3
…	0

R_2

A	B	P(B|A)
1	2	.3
1	3	.7
2	1	.4
2	3	.6
3	1	.1
3	2	.9
…	…	0



R_3

A	C	P(C|A)
1	2	1
3	2	1
…	…	0

R_4

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

R_5

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

R_6

D	F	G	P(G|D,F)
1	2	3	1
2	1	3	1
…	…	…	0

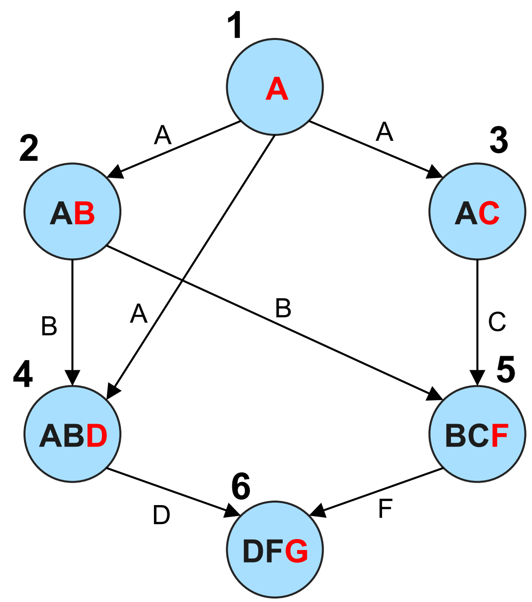
IBP Example – Iteration 1

R_1

A	P(A)
1	>0
3	>0
…	0

R_2

A	B	P(B|A)
1	3	1
2	1	>0
2	3	>0
3	1	1
…	…	0



R_3

A	C	P(C|A)
1	2	1
3	2	1
…	…	0

R_4

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

R_5

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

R_6

D	F	G	P(G|D,F)
2	1	3	1
…	…	…	0

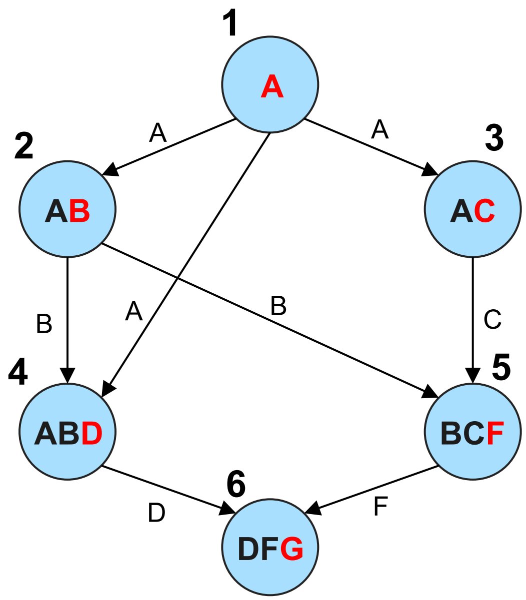
IBP Example – Iteration 2

R_1

A	P(A)
1	>0
3	>0
…	0

R_2

A	B	P(B|A)
1	3	1
3	1	1
…	…	0



R_3

A	C	P(C|A)
1	2	1
3	2	1
…	…	0

R_4

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

R_5

B	C	F	P(F|B,C)
3	2	1	1
…	…	…	0

R_6

D	F	G	P(G|D,F)
2	1	3	1
…	…	…	0

IBP Example – Iteration 3

R_1

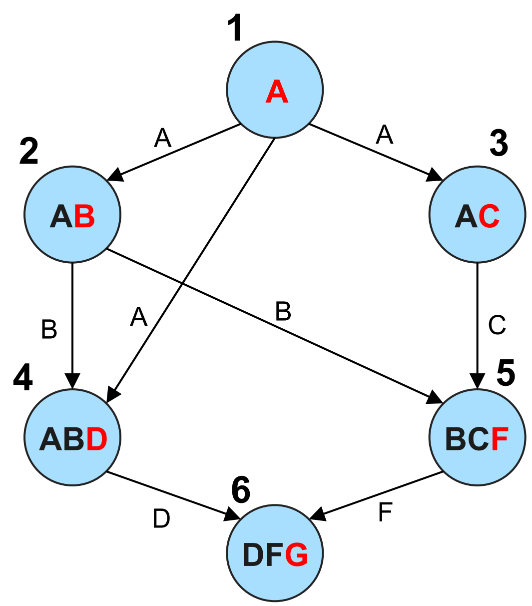
A	P(A)
1	>0
3	>0
…	0

R_2

A	B	P(B|A)
1	3	1
…	…	0

R_3

A	C	P(C|A)
1	2	1
3	2	1
…	…	0



R_4

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

R_5

B	C	F	P(F|B,C)
3	2	1	1
…	…	…	0

R_6

D	F	G	P(G|D,F)
2	1	3	1
…	…	…	0

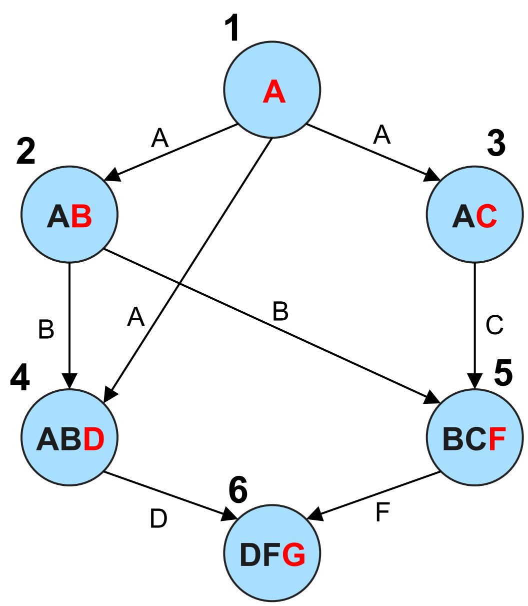
IBP Example – Iteration 4

R_1

A	P(A)
1	1
…	0

R_2

A	B	P(B|A)
1	3	1
…	…	0



R_3

A	C	P(C|A)
1	2	1
3	2	1
…	…	0

R_4

A	B	D	P(D|A,B)
1	3	2	1
…	…	…	0

R_5

B	C	F	P(F|B,C)
3	2	1	1
…	…	…	0

R_6

D	F	G	P(G|D,F)
2	1	3	1
…	…	…	0

IBP Example – Iteration 5

R_1

A	P(A)
1	1
…	0

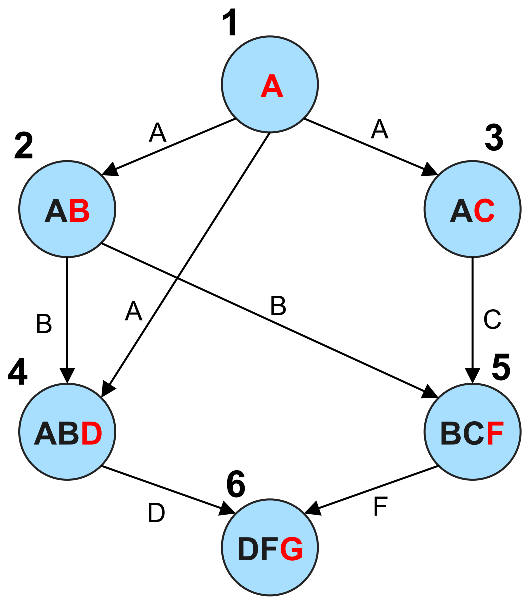
A	B	C	D	F	G	Belief
1	3	2	2	1	3	1
…	…	…	…	…	…	0

R_2

A	B	P(B|A)
1	3	1
…	…	0

R_3

A	C	P(C|A)
1	2	1
…	…	0



R_4

A	B	D	P(D|A,B)
1	3	2	1
…	…	…	0

R_5

B	C	F	P(F|B,C)
3	2	1	1
…	…	…	0

R_6

D	F	G	P(G|D,F)
2	1	3	1
…	…	…	0

IBP – inference power for zero beliefs

- **Theorem:**

  Trace of zero beliefs of Iterative Belief Propagation =
  Trace of invalid tuples of arc-consistency on flat network

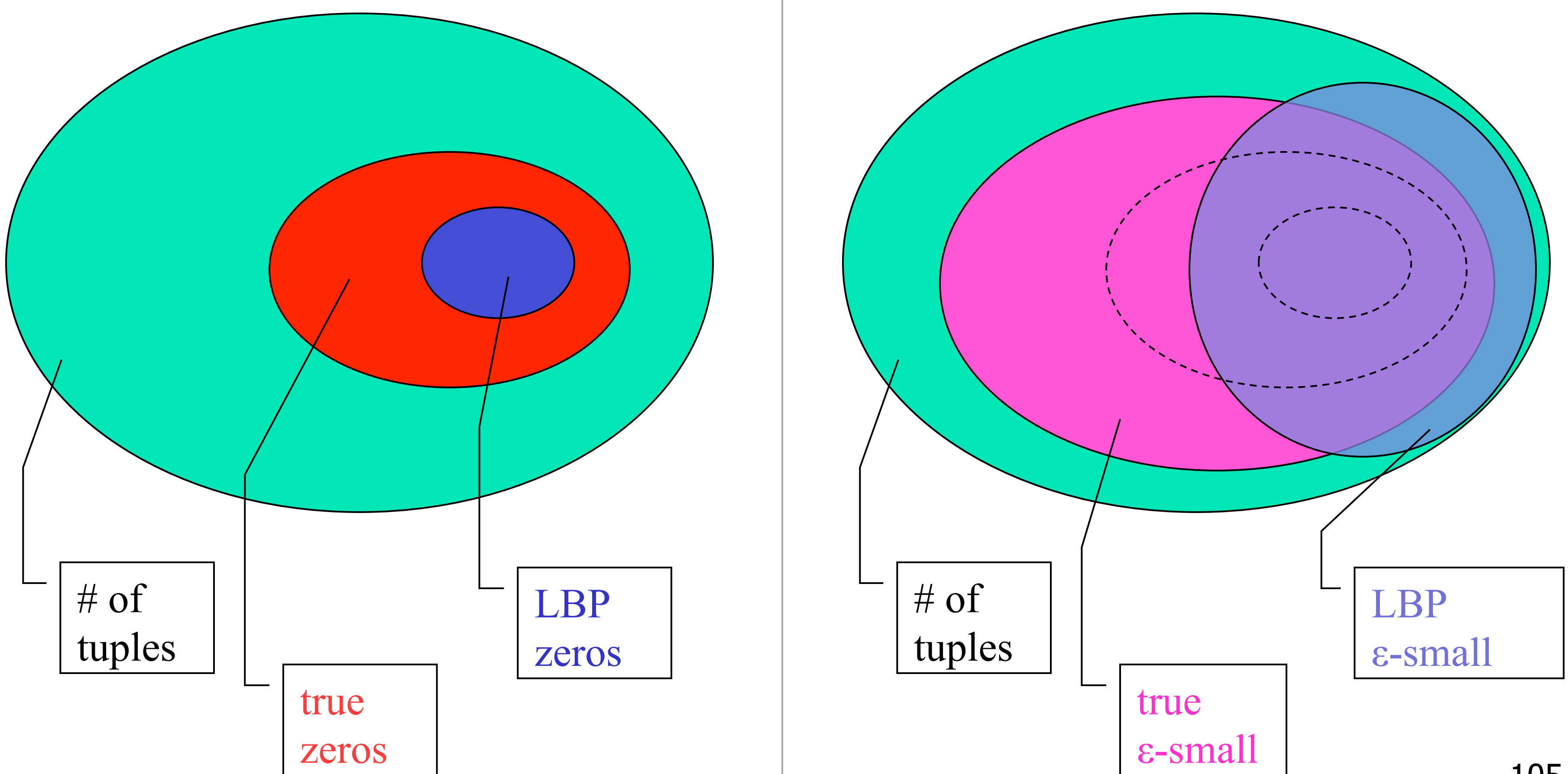
- **Soundness:**
  - The inference of zero beliefs by IBP converges in a finite number of iterations
  - all the inferred zero beliefs are correct

- **Incompleteness:**
  - IBP may not infer all the true zero beliefs

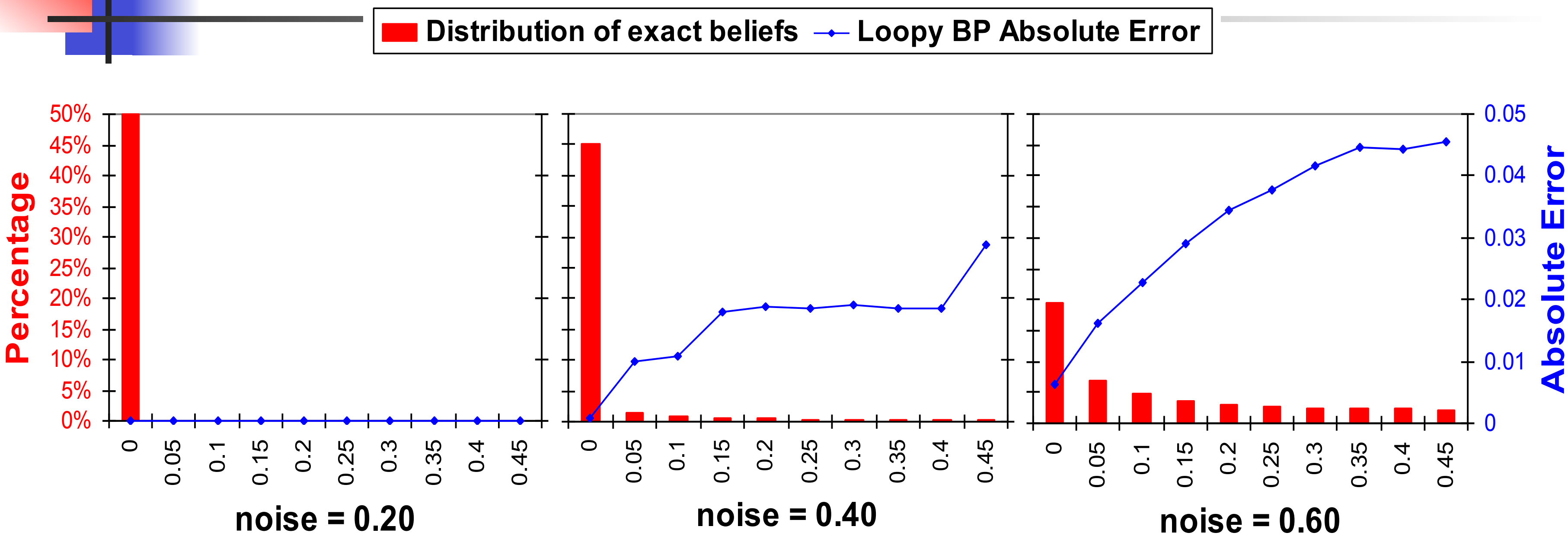
Zero and ε -Small Beliefs

Zero beliefs

ε -small beliefs

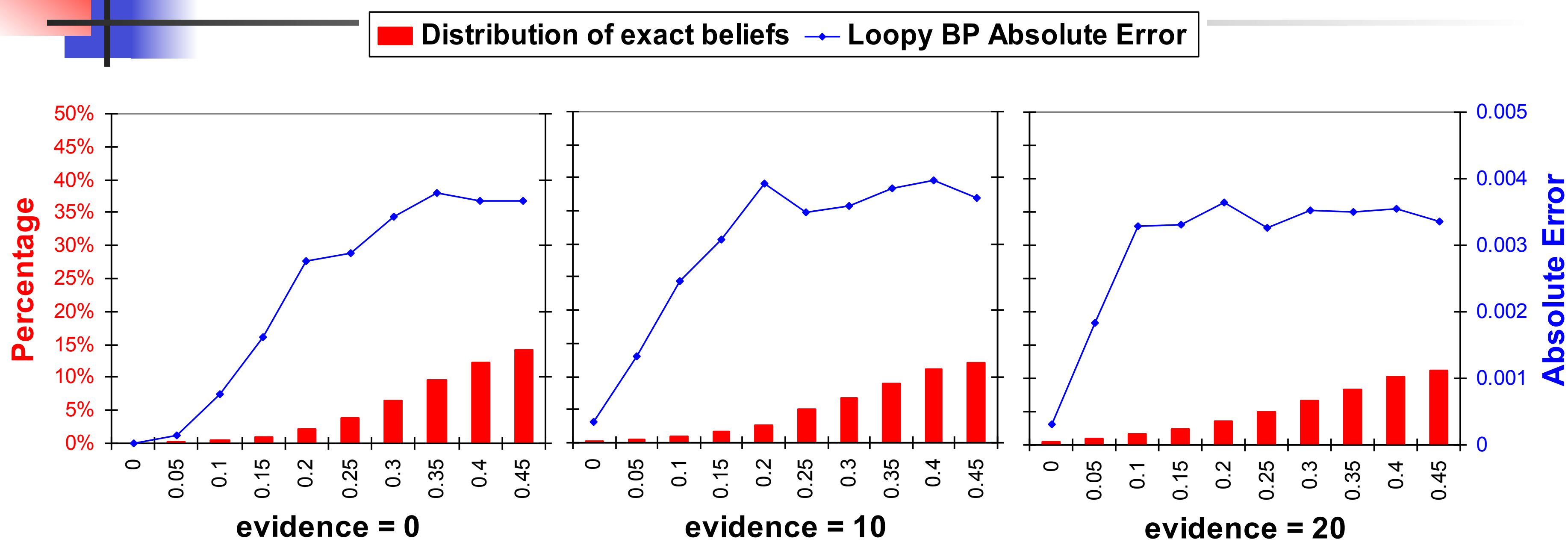


Coding Networks



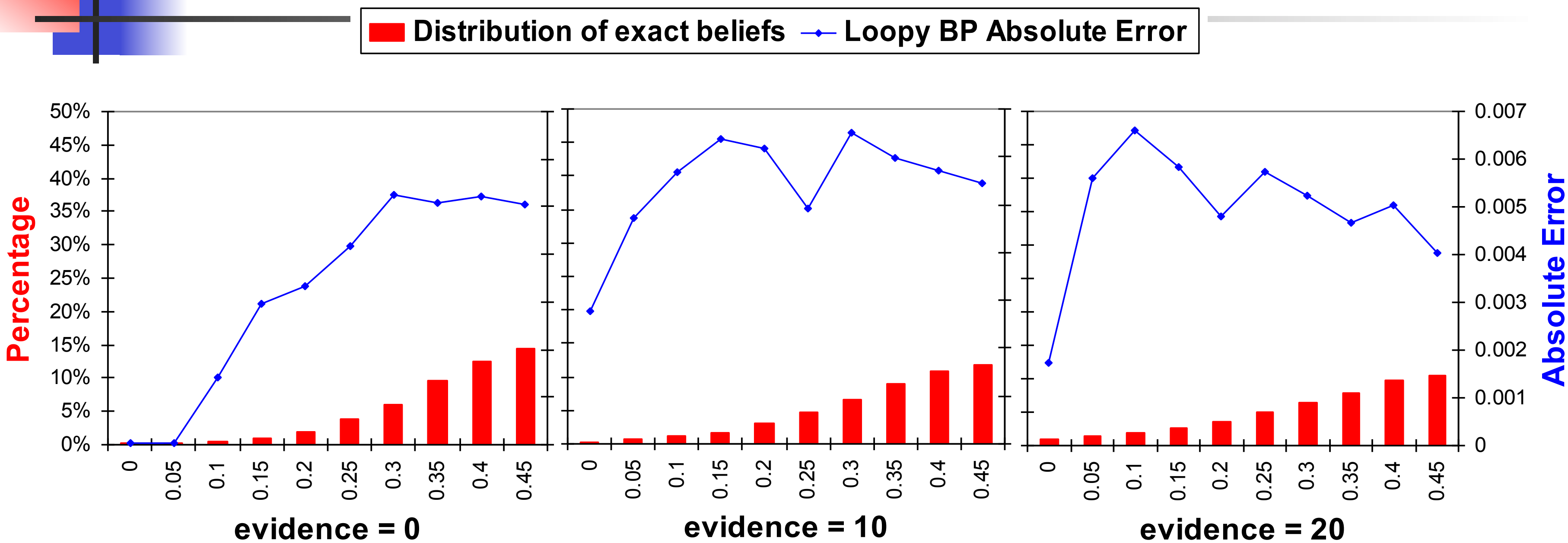
N=200, 1000 instances, treewidth=15

10x10 Grids



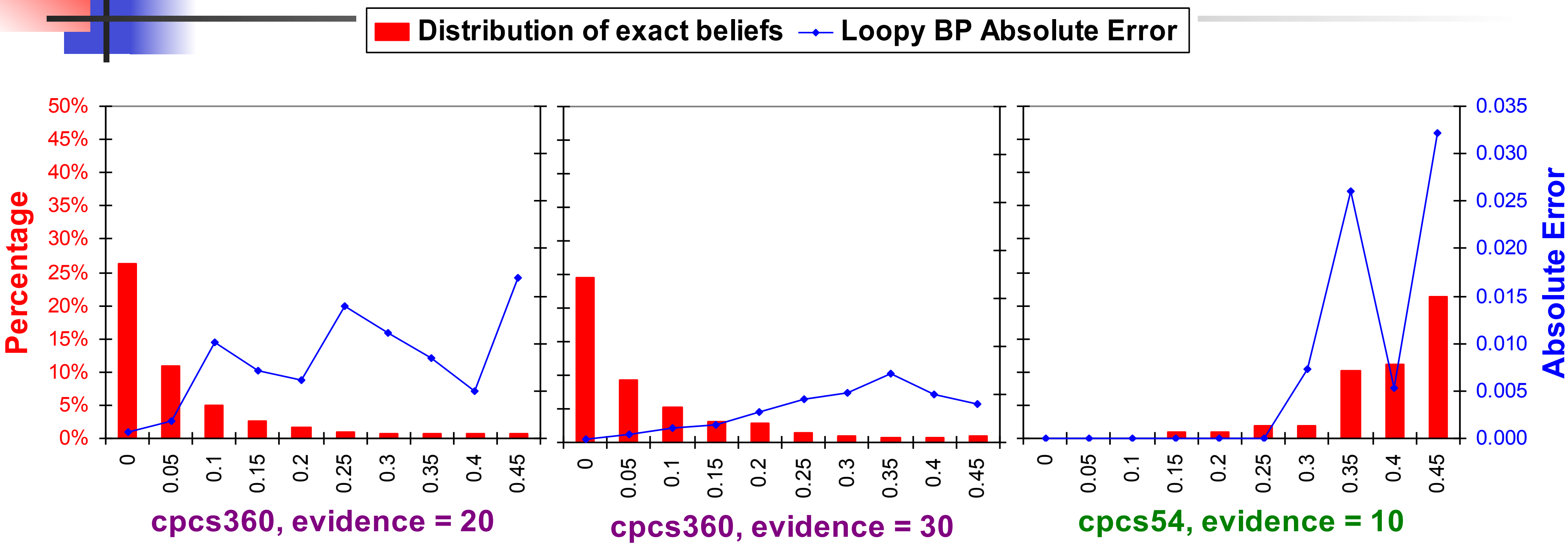
N=100, 100 instances, w*=15

Random Networks



N=80, 100 instances, w*=15

CPCS 54, CPCS360



CPCS360: 5 instances, w*=20

CPCS54: 100 instances, w*=15

Experimental Results

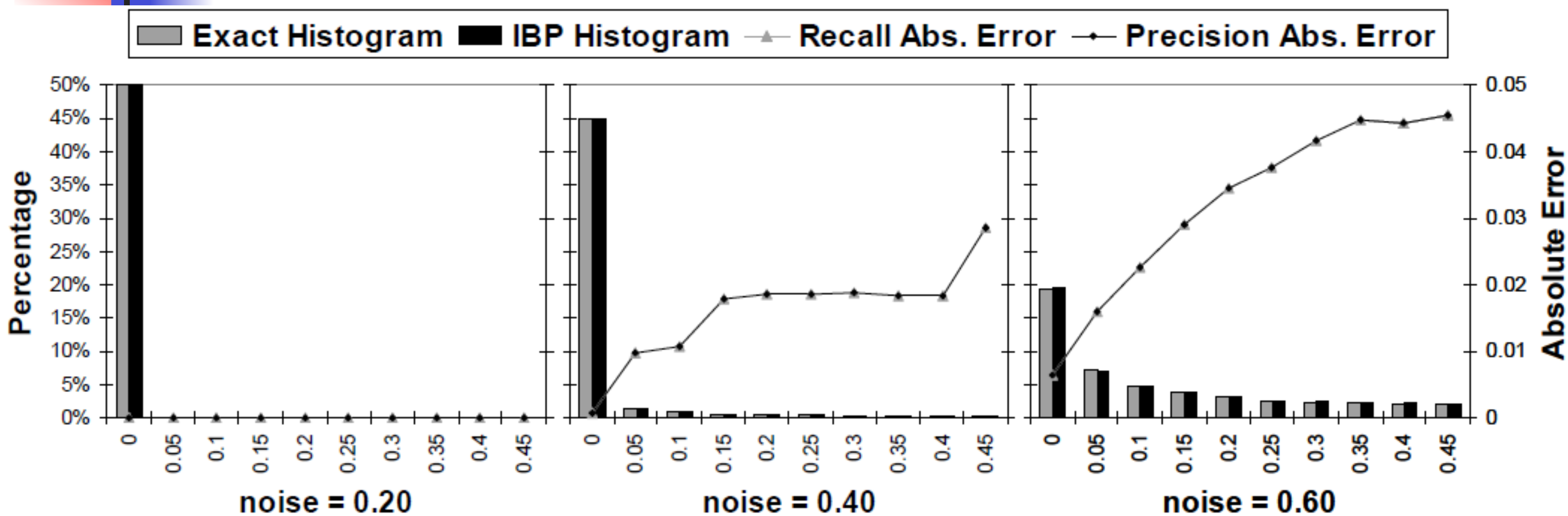
We investigated empirically if the results for zero beliefs extend to ε-small beliefs ($\varepsilon > 0$)

Have determinism?

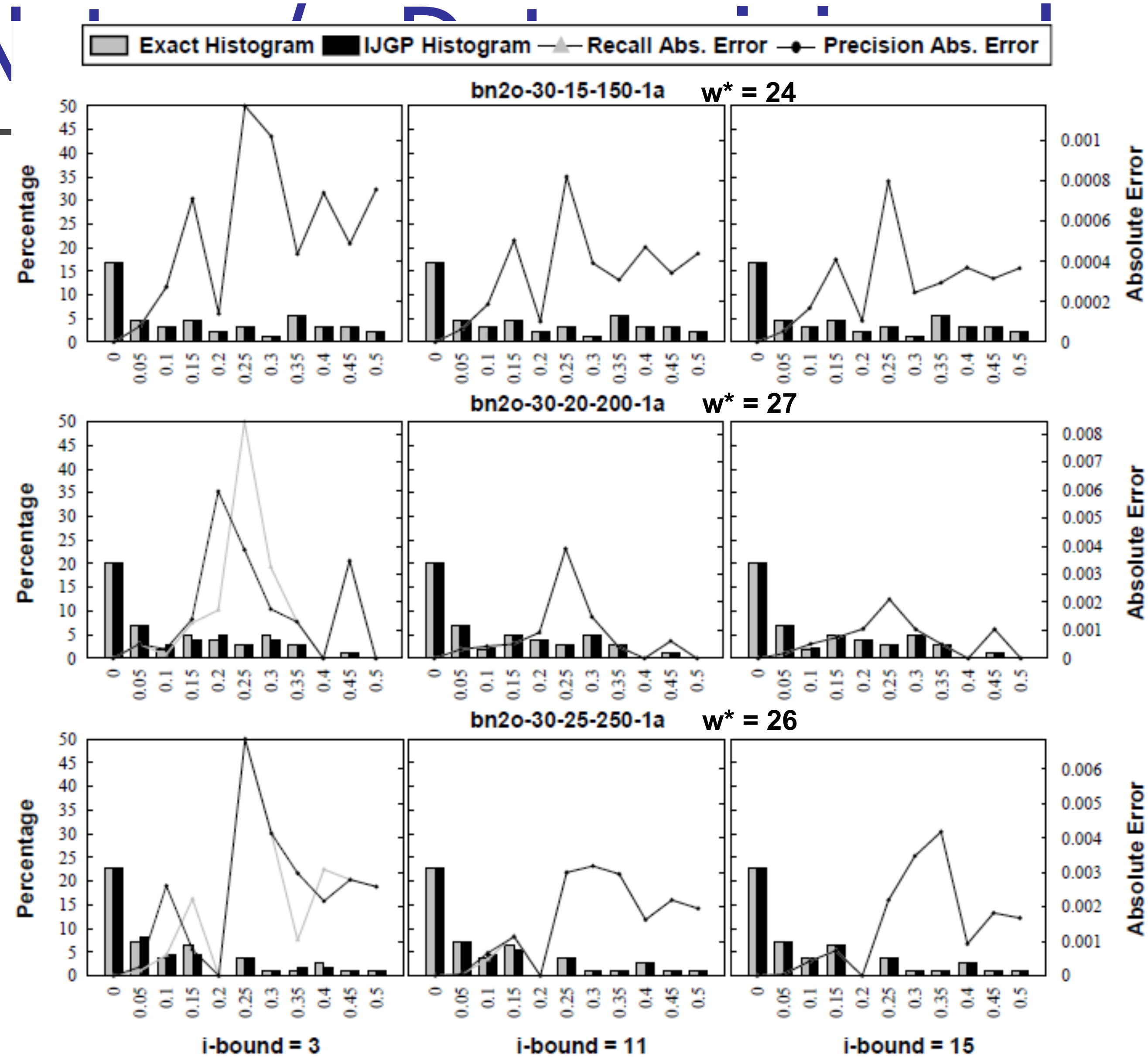
- Network types:
  - YES
    - Coding
    - Linkage analysis*
    - Grids*
  - NO
    - Two-layer noisy-OR*
    - CPCS54, CPCS360
- Measures:
  - Exact/IJGP histogram
  - Recall absolute error
  - Precision absolute error
- Algorithms:
  - IBP
  - IJGP

* Instances from the UAI08 competition

Networks with Determinism: Coding

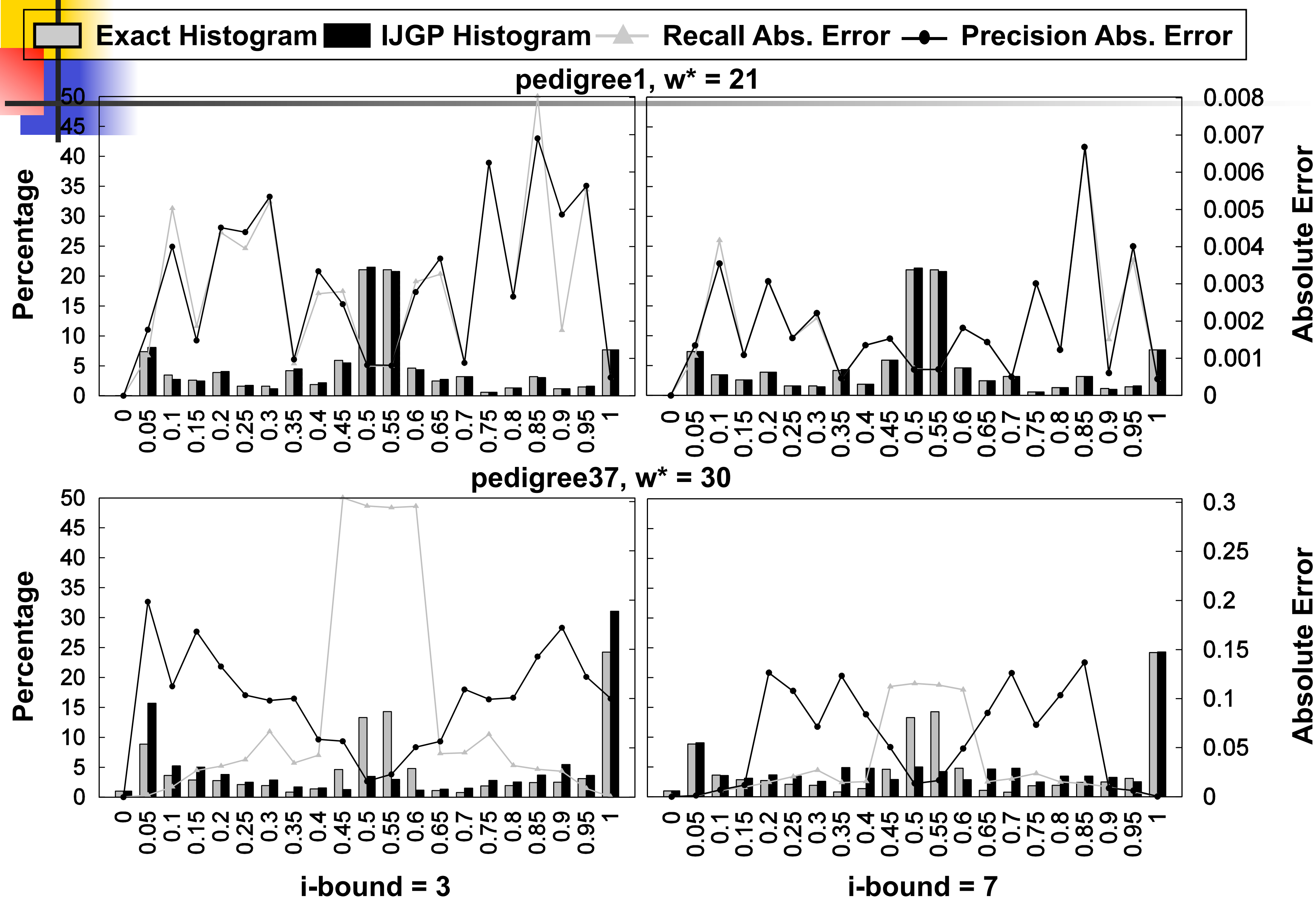


N=200, 1000 instances, w*=15



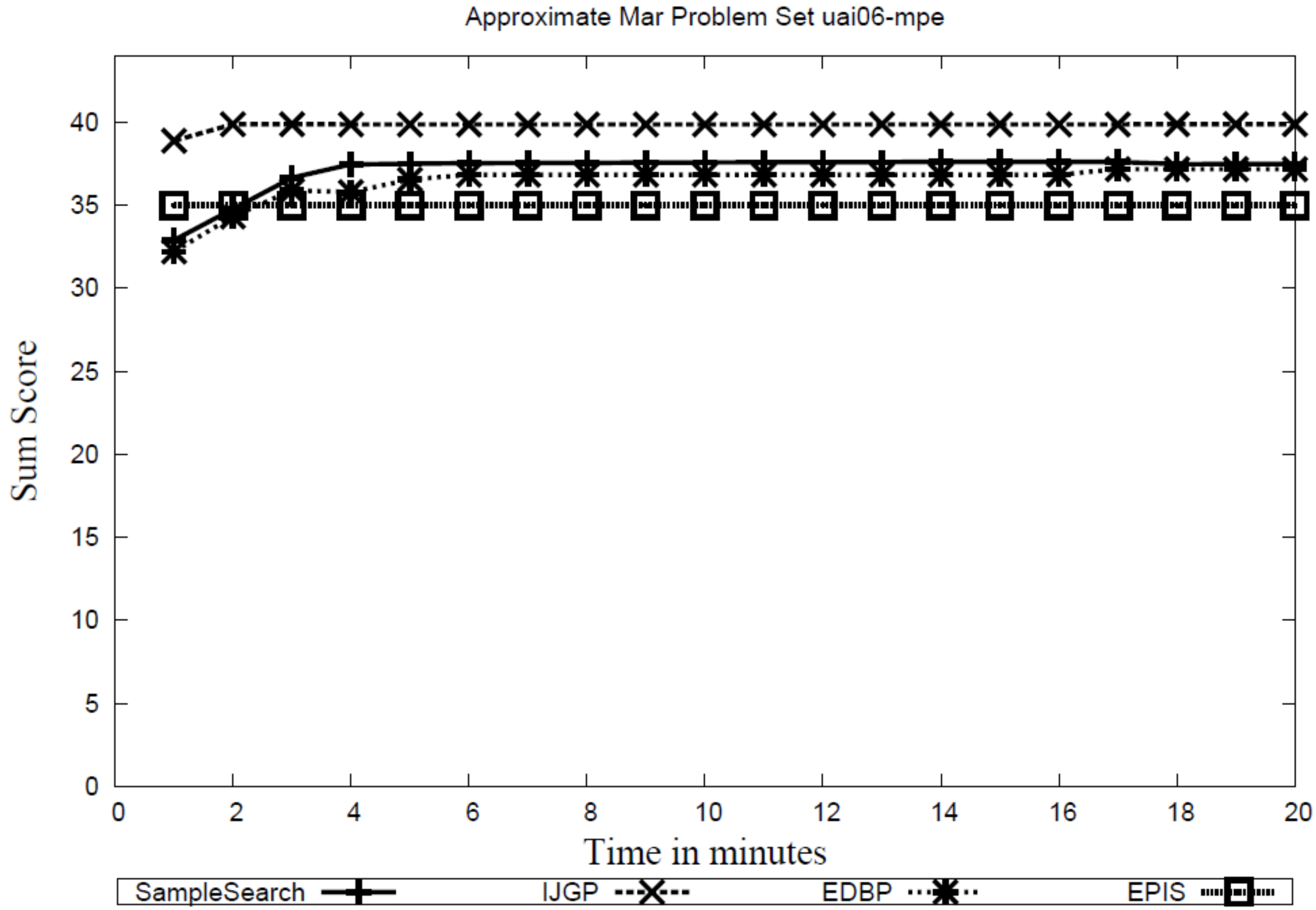
Exact Histogram
IJGP Histogram
Recall Abs. Error
Precision Abs. Error
bn2o-30-15-150-1a
w* = 24
bn2o-30-20-200-1a
w* = 27
bn2o-30-25-250-1a
w* = 26
Percentage
Absolute Error
i-bound = 3
i-bound = 11
i-bound = 15

Nets with Determinism: Linkage

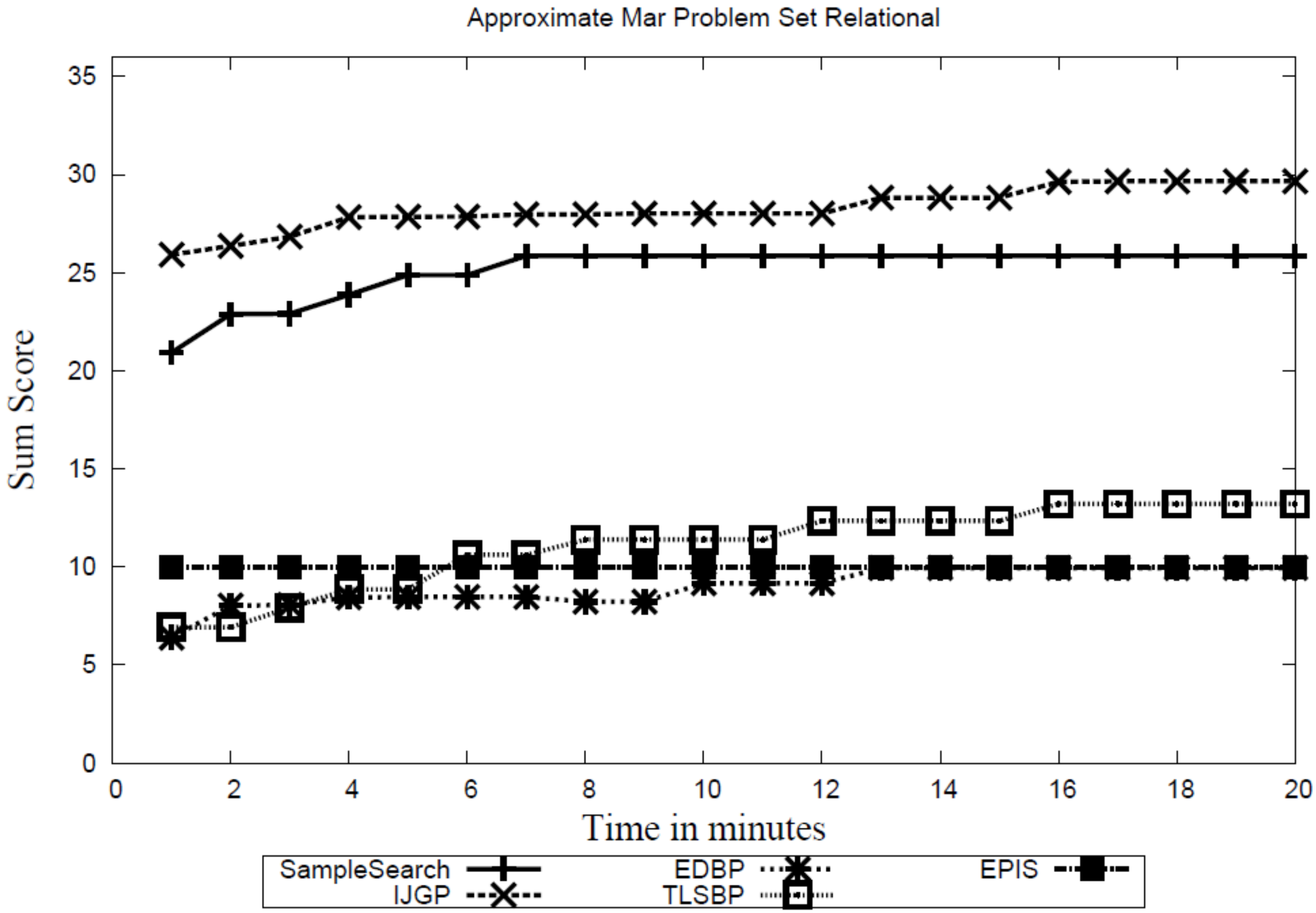


Some competition comparison

IJGP on UAI06 problems



IJGP on Set Relational

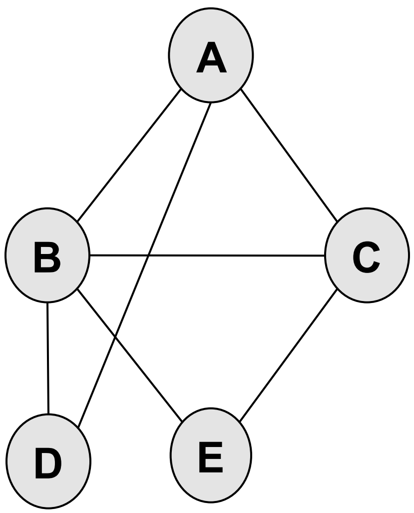


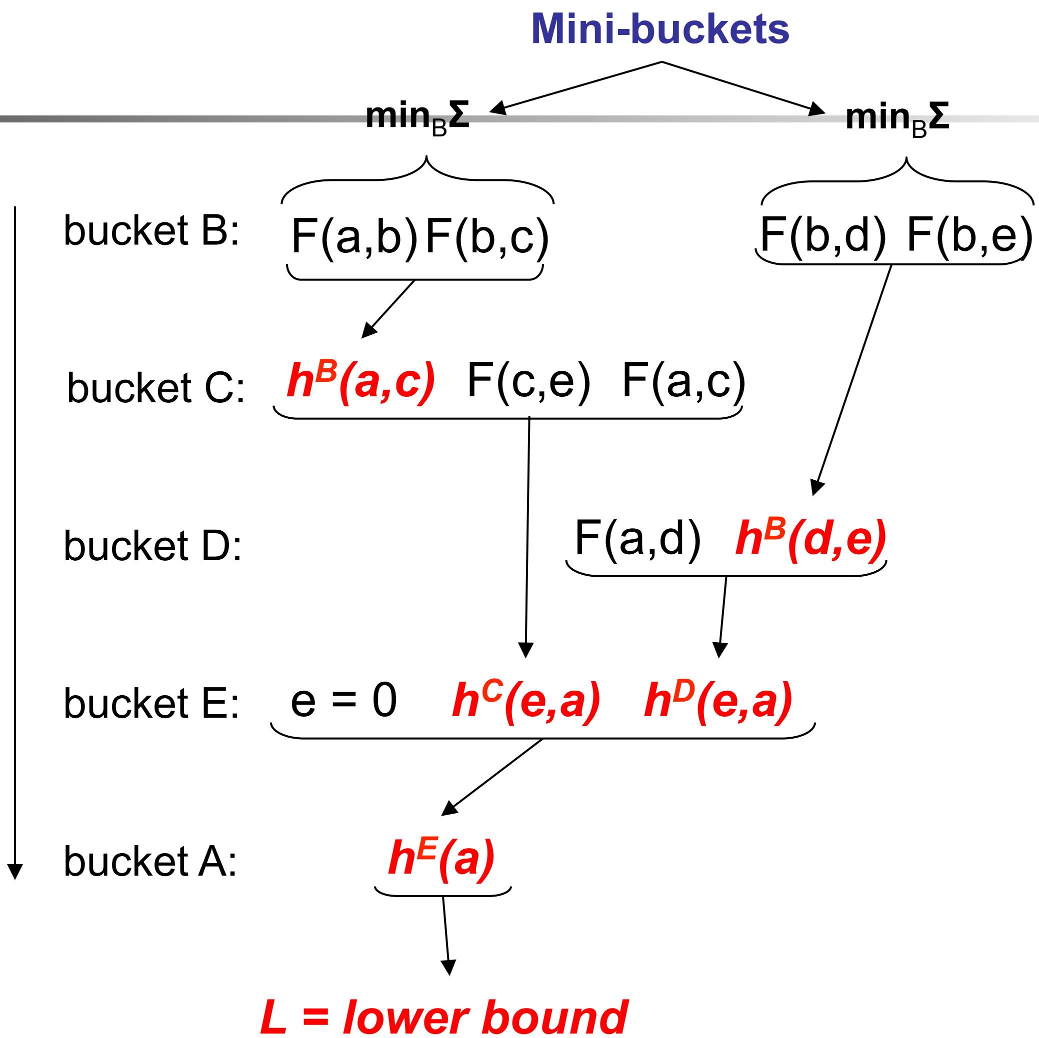
Agenda

- 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





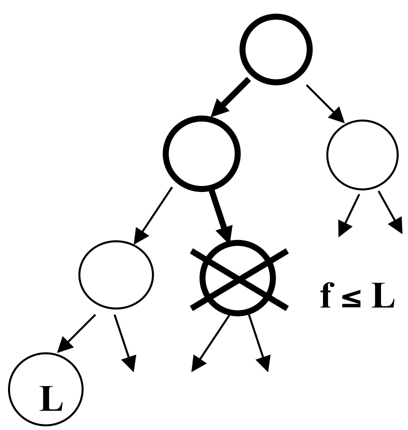
Basic Heuristic Search Schemes

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

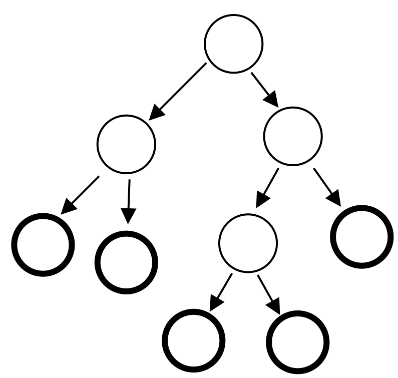
Linear space (or more)



2. Best-First Search

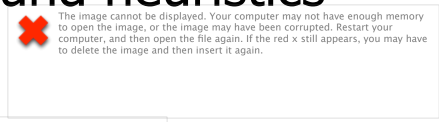
Always expand the node with the highest heuristic value **$f(x^p)$**

Needs lots of memory



Heuristic search

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



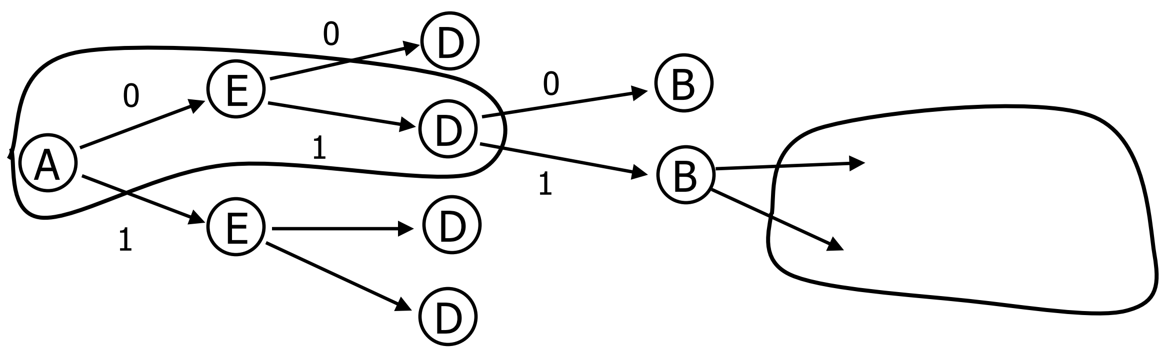
- **Best-first:** expand a node with maximal evaluation function
- **Branch and Bound:** prune if f <= upper bound
- **Properties:**
  - an exact algorithm
  - Better heuristics lead to more pruning

Heuristic Function

Given a cost function

$$P(a,b,c,d,e) = P(a) \bullet P(b|a) \bullet P(c|a) \bullet P(e|b,c) \bullet P(d|b,a)$$

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

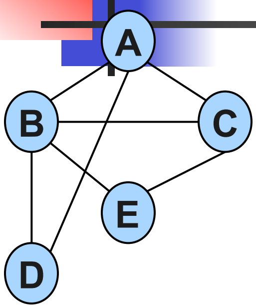


$$f^*(a,e,d) = \max_{b,c} P(a,b,c,d,e) =$$

$$= P(a) \bullet \max_{b,c} P(b|a) \bullet P(c|a) \bullet P(e|b,c) \bullet P(d|a,b)$$

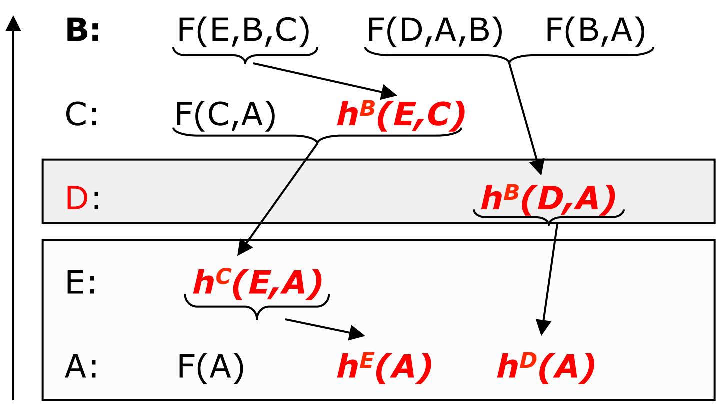
$$= g(a,e,d) \bullet H^*(a,e,d)$$

MBE Heuristics

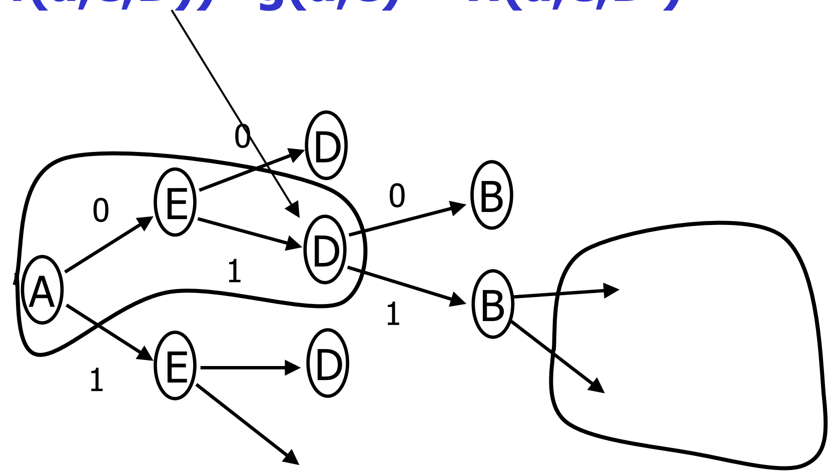


Cost Network

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



$$\mathbf{f(a,e,D))=g(a,e) + H(a,e,D\ )}$$

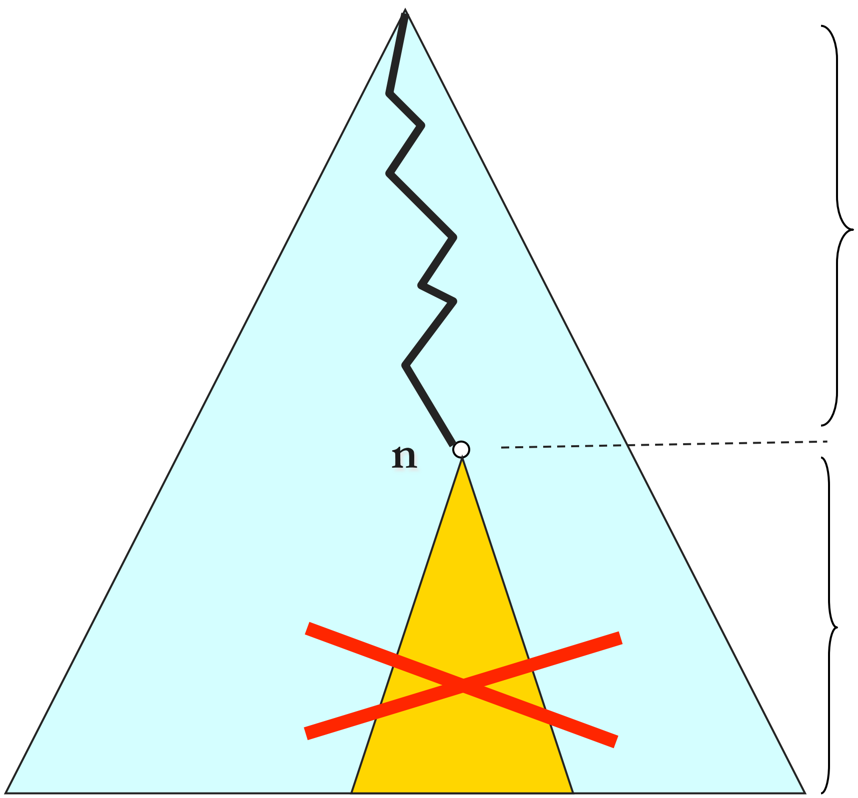


$$\mathbf{f(a,e,D) = \underbrace{F(a)}_{g} + \underbrace{h^B(D,a) + h^C(e,a)}_{h\text{ – is admissible}}}$$

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



OR Search Tree

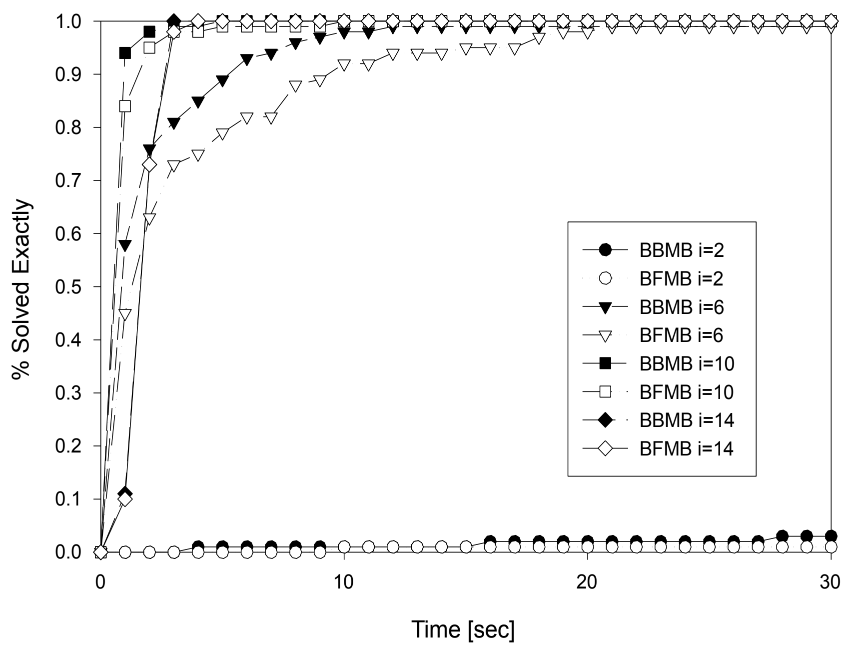
Upper Bound **UB**

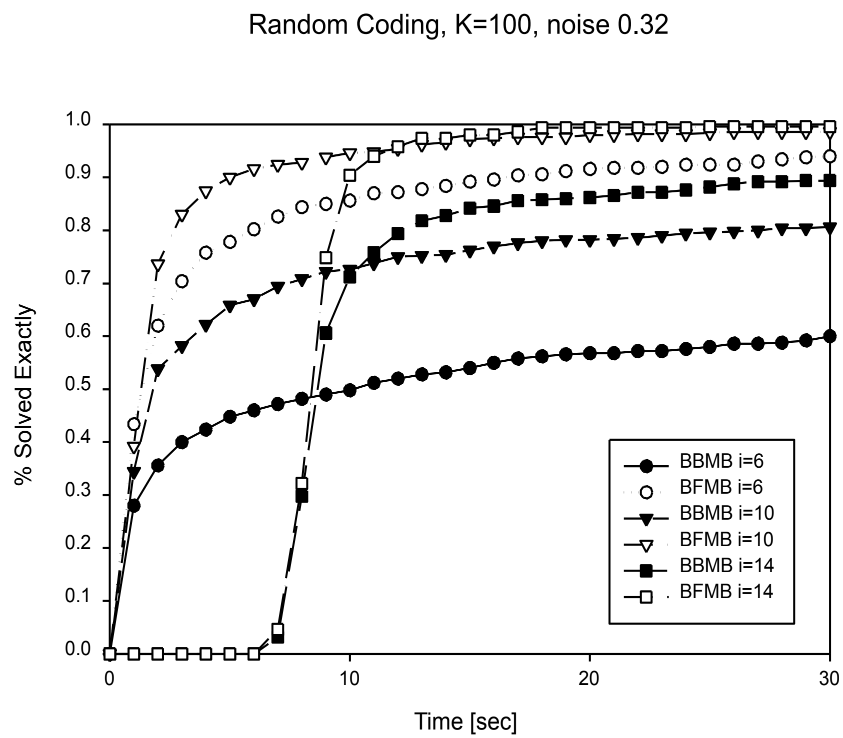
Lower Bound **LB**

LB(n) = g(n) + h(n)

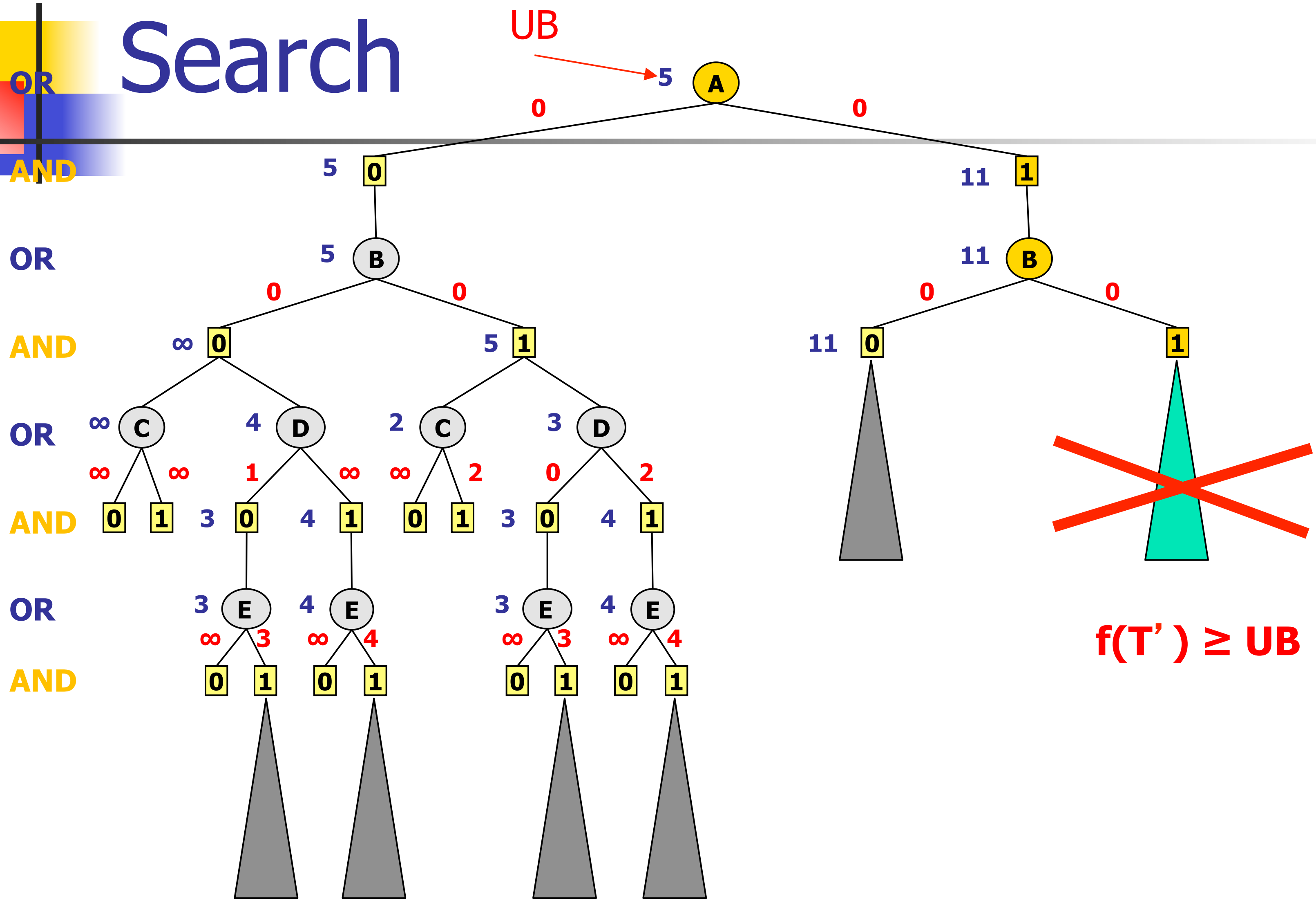
Prune if LB(n) ≥ UB

Empirical Evaluation of mini-bucket heuristics

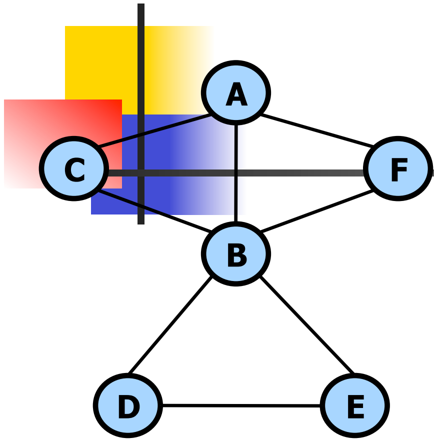




AND/OR Branch-and-Bound Search



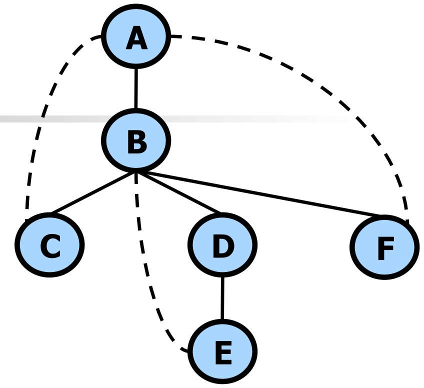
Heuristic Evaluation Function

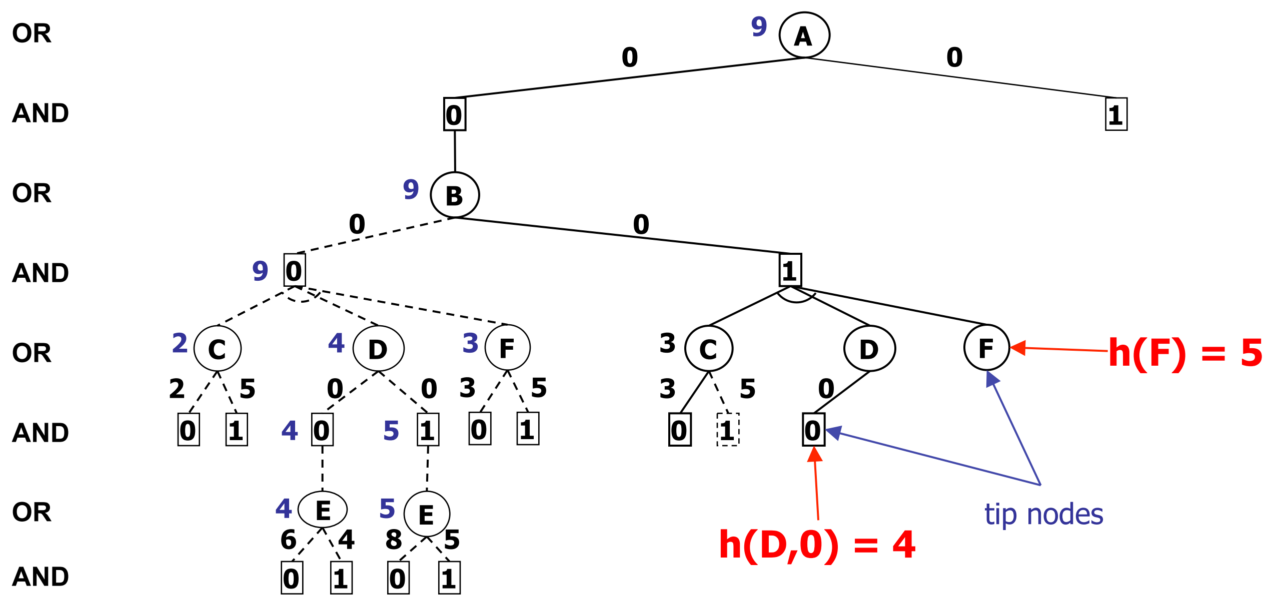


A	B	C	$f_1(ABC)$
0	0	0	2
0	0	1	5
0	1	0	3
0	1	1	5
1	0	0	9
1	0	1	3
1	1	0	7
1	1	1	2

A	B	F	$f_2(ABF)$
0	0	0	3
0	0	1	5
0	1	0	1
0	1	1	4
1	0	0	6
1	0	1	5
1	1	0	6
1	1	1	5

B	D	E	$f_3(BDE)$
0	0	0	6
0	0	1	4
0	1	0	8
0	1	1	5
1	0	0	9
1	0	1	3
1	1	0	7
1	1	1	4





$$f(T') = w(A,0) + w(B,1) + w(C,0) + w(D,0) + h(D,0) + h(F) = 12 \le f^*(T')$$

Software & Competitions

- **How to use the software**
  - http://graphmod.ics.uci.edu/group/Software
  - 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

  (Open source WCSP, MPE solver in C++)

- aolib

  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

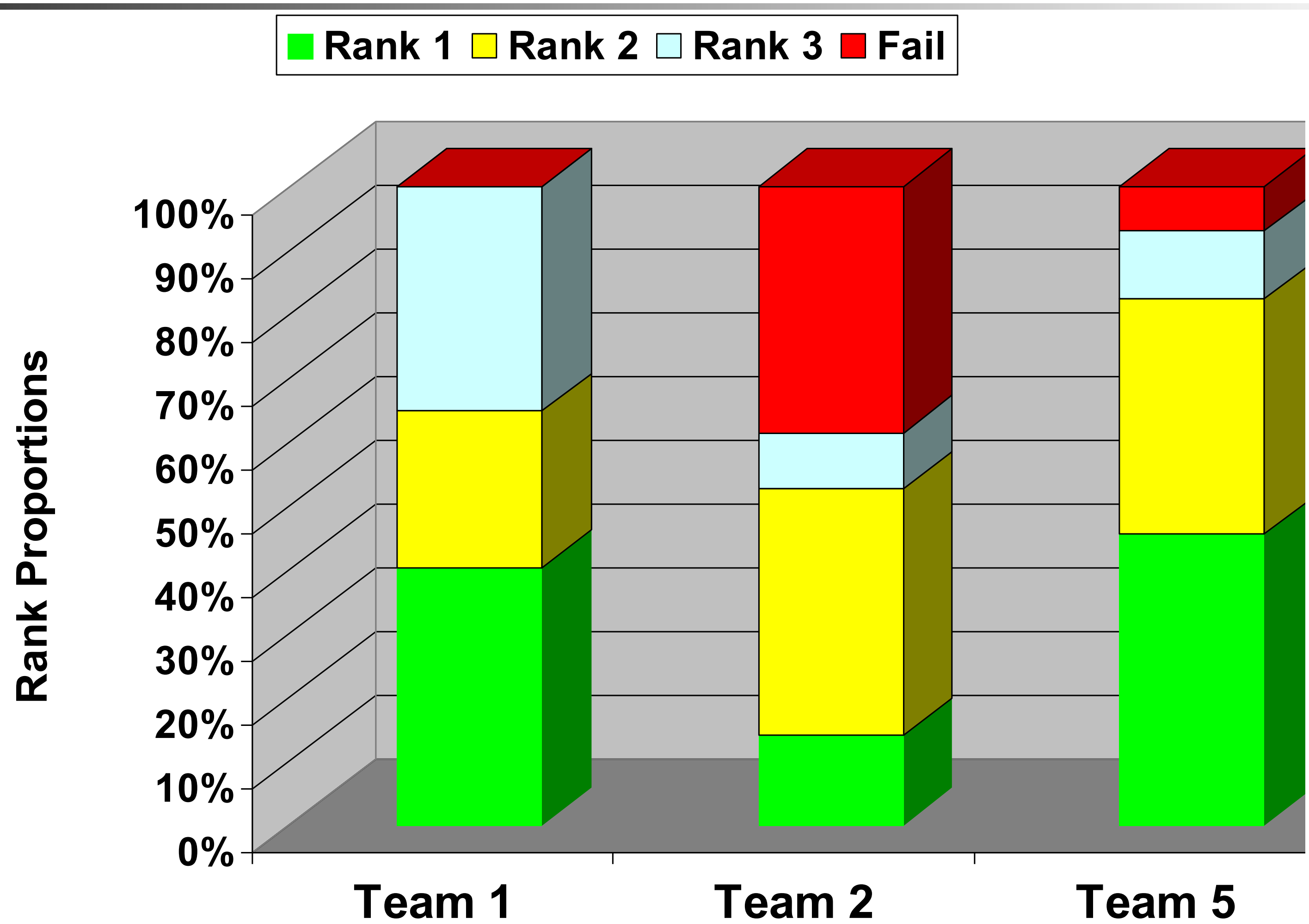
  http://graphmod.ics.uci.edu/group/Repository

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

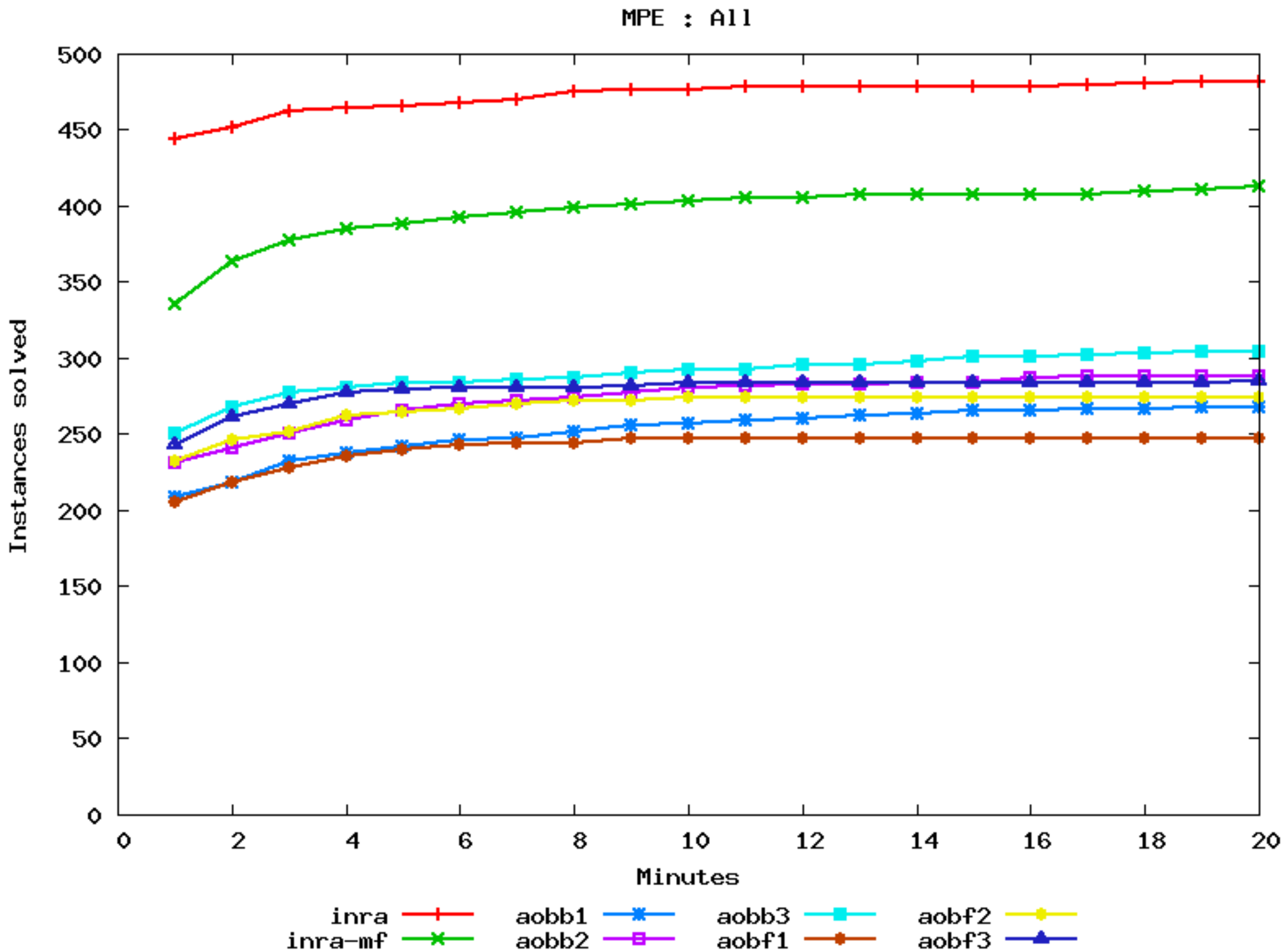
Rank Proportions (how often was each team a particular rank, rank 1 is best)



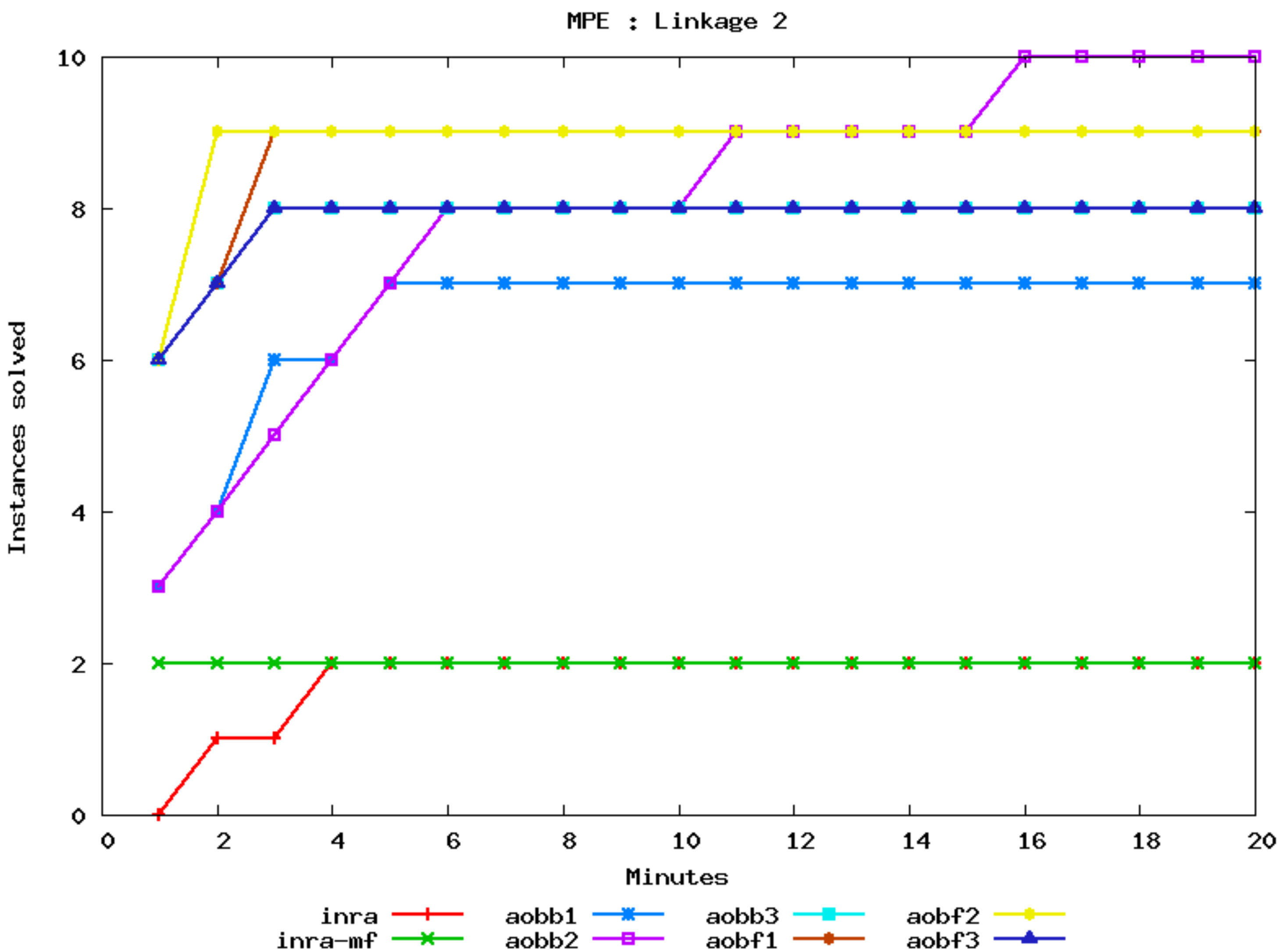
UAI-2008 Competition

- **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



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 (Mateescu et al, JAIR2010), (Dechter et al, UAI2002)
  - **1st place** (20 min, 1 hour) – (impl. by Vibhav Gogate)
  - Anytime **IJGP(i)** with randomized orderings and SAT based domain pruning
- PR task (Gogate, Domingos and Dechter UAI2010)
  - **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 (Marinescu and Dechter, AIJ2009), (Otten and Dechter, ISAIM2010)
  - **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

DISCML 2012 – NIPS Workshop

Winning the PASCAL 2011 MAP Challenge with Enhanced AND/OR Branch-and-Bound

Lars Otten, Alexander Ihler,
Kalev Kask, Rina Dechter

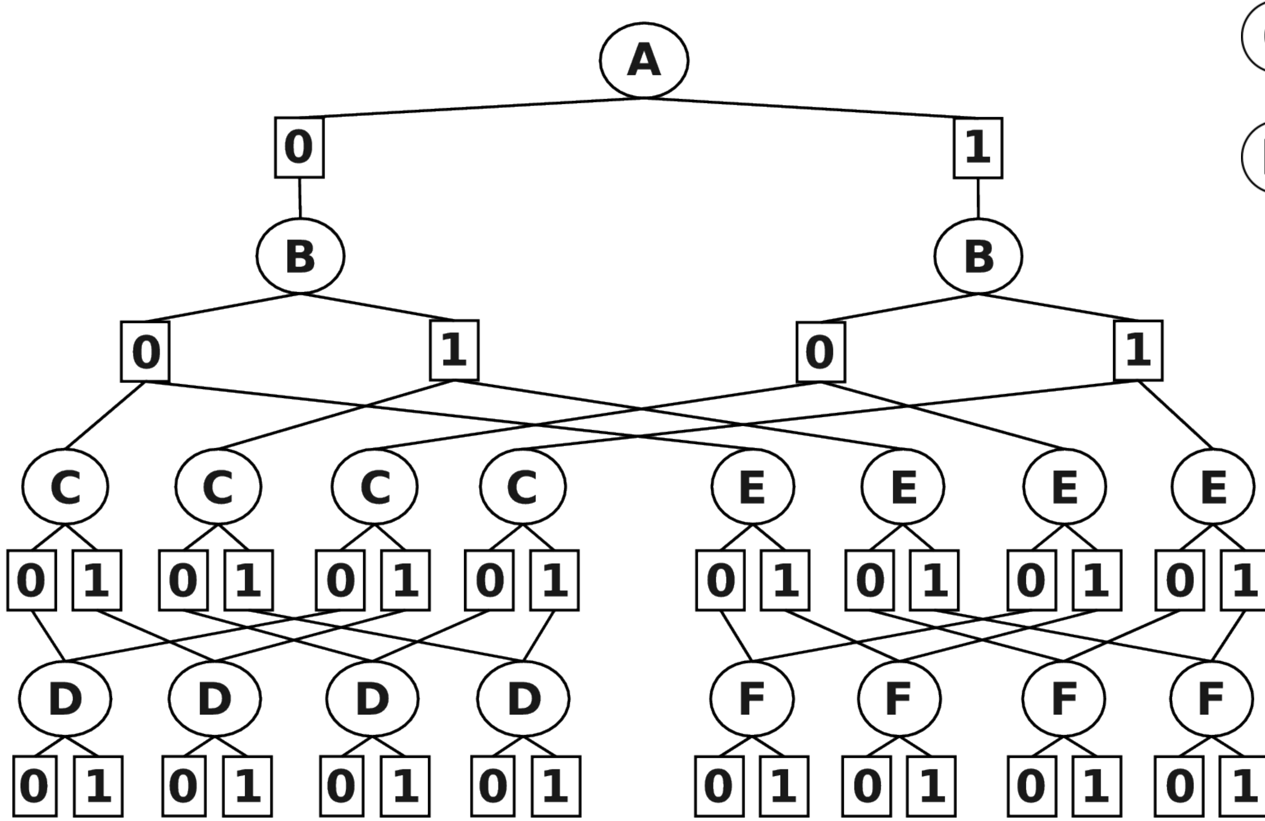
Dept. of Computer Science
University of California, Irvine

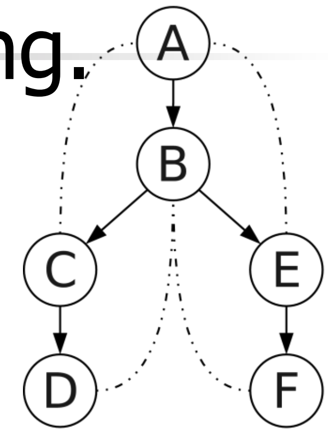
Overview

- 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 AOBB “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.

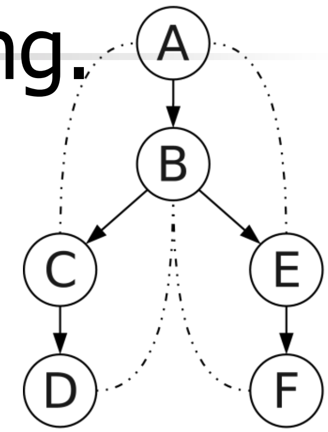




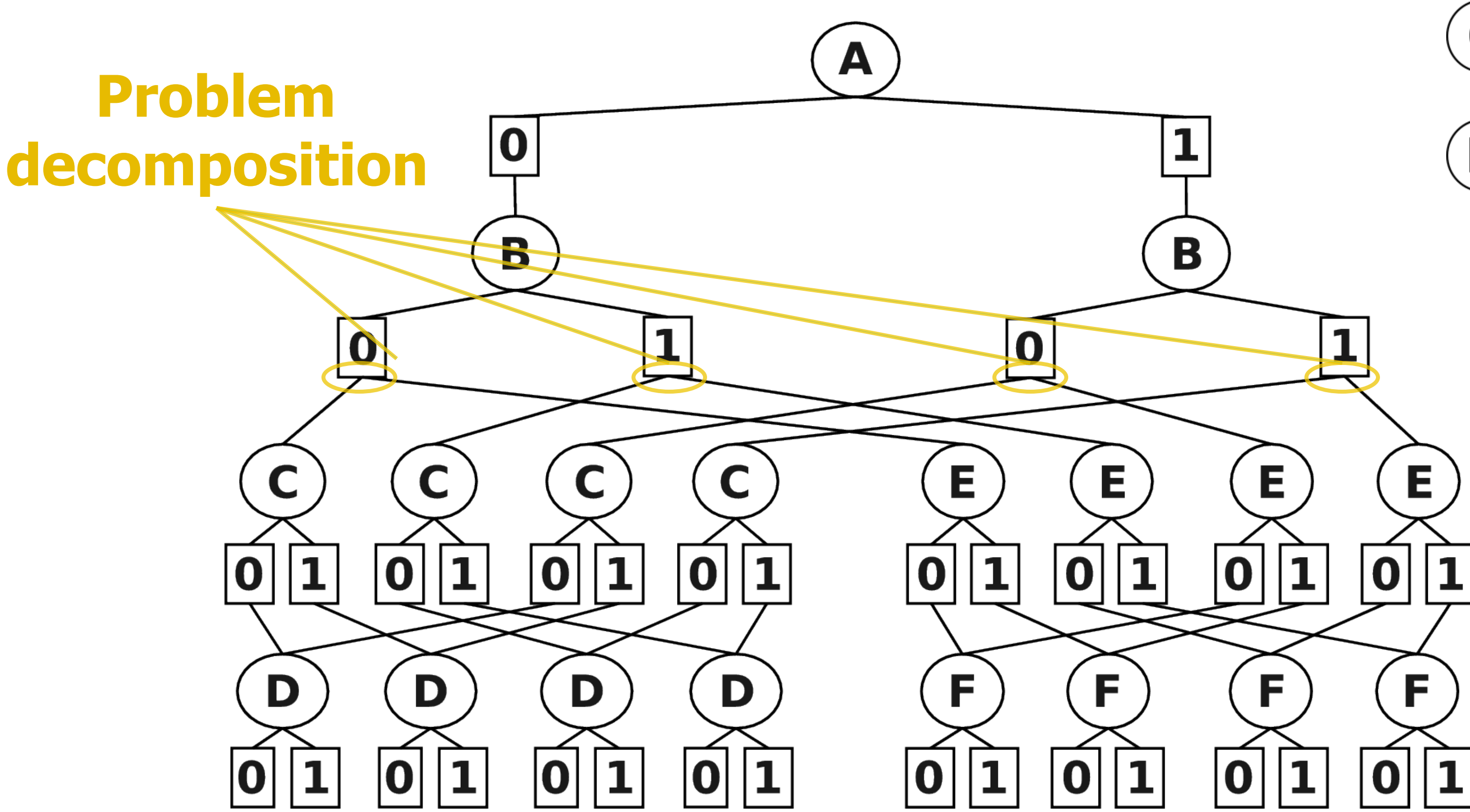
Guided by pseudo tree

AND/OR Branch-and-Bound

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

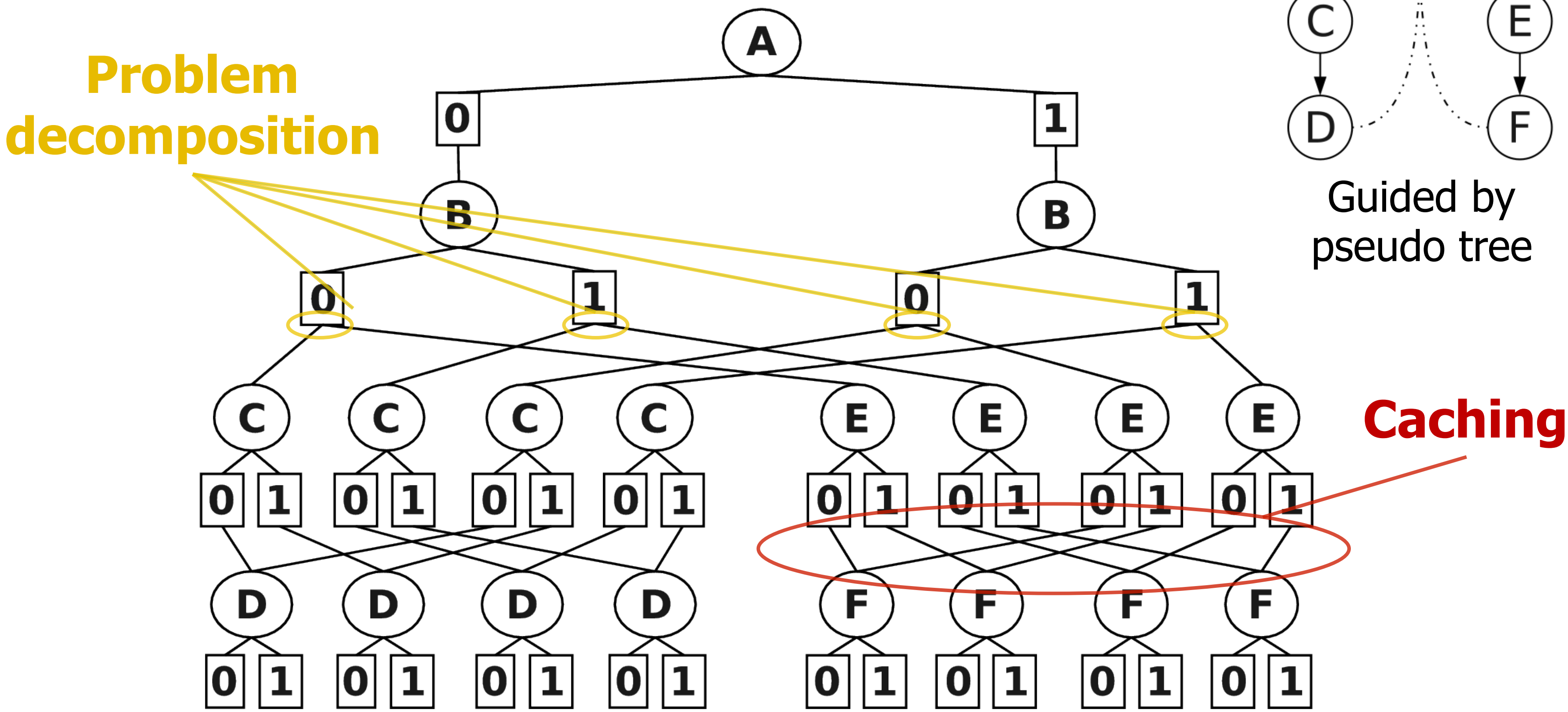


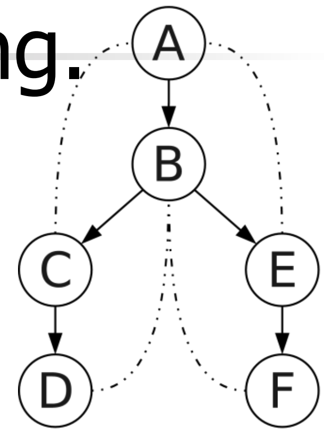
Guided by pseudo tree



AND/OR Branch-and-Bound

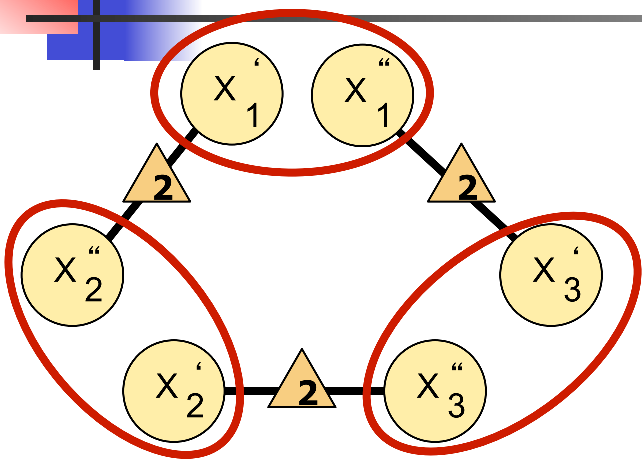
- Problem decomposition and caching.
  - Mini-bucket heuristic for pruning.



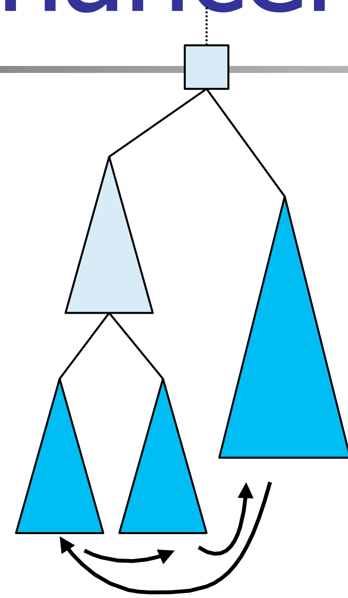


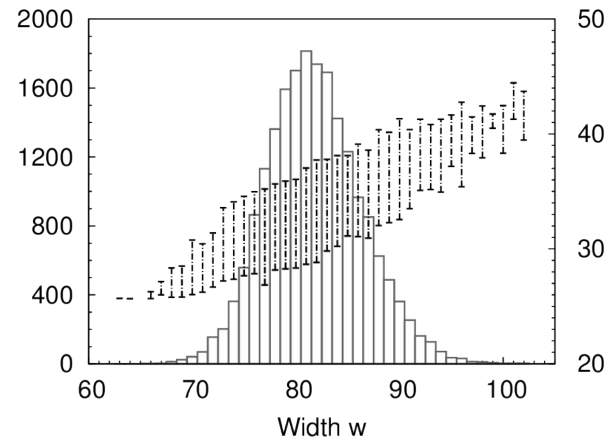
Guided by pseudo tree

Central Enhancements



$\min_{\lambda} \; \Sigma_{(ij)} \max_{X}(f_{ij}(X_i, X_j) + \lambda_{ij}(X_i), \lambda_{ji}(X_j))$





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.
    - $\mathrm{E}(\mathrm{x}) = -\sum \log f_i(x)$, $Score(x^s) = \frac{E(x^s) - \min\{E(x^{bp}), E(x^{df})\}}{|\min\{E(x^{bp}), E(x^{df})\}|}$
  - **1st place** in all three categories!

	20 sec			20 min			1 hour		
Category	*daoopt*	*ficolofo*	*dfbbvemcs*	*daoopt*	*dfbbvecms*	*ficolofo*	*daoopt*	*ficolofo*	*vns/lds+cp*
CSP	**-0.9123**	-0.8669	-0.8669	**-0.8739**	-0.7862	-0.7862	**-0.8442**	-0.6958	-0.6975
Deep belief nets	-	-	-	-1.6286	**-1.6342**	**-1.6342**	-5.0470	-5.1707	**-5.1709**
Grids	**-0.3403**	-0.3210	-0.3174	**-0.2437**	-0.2241	-0.2241	**-0.1721**	-0.1590	-0.1589
Image alignment	0.0000	0.0000	0.0000	**-0.0006**	0.0000	**-0.0006**	-0.0006	-0.0006	-0.0006
Medical diagnosis	-0.0028	-0.0046	**-0.0460**	-0.0037	**-0.0043**	**-0.0043**	-0.0041	**-0.0043**	**-0.0043**
Object detection	-4.8201	**-4.8287**	-4.8023	-4.8237	**-4.8743**	**-4.8743**	-1.9368	**-1.9628**	-1.9572
Protein folding	-0.0308	-0.0308	-0.0308	-0.1135	**-0.1187**	**-0.1187**	-0.1146	**-0.1183**	**-0.1183**
Prot/prot inter.	-	-	-	**-0.1341**	-0.1317	-0.1317	-0.1681	**-0.1744**	-0.1735
Relational	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Segmentation	**-0.0300**	**-0.0300**	-0.0298	-0.0300	-0.0300	-0.0300	-0.0338	-0.0338	-0.0338
Overall	**-6.3164**	-6.0819	-6.0518	**-7.8519**	-7.8041	-7.8000	**-8.3214**	-8.3196	-8.3150