COMP9418: Advanced Topics in Statistical Machine Learning

Belief Propagation

Instructor: Gustavo Batista

University of New South Wales

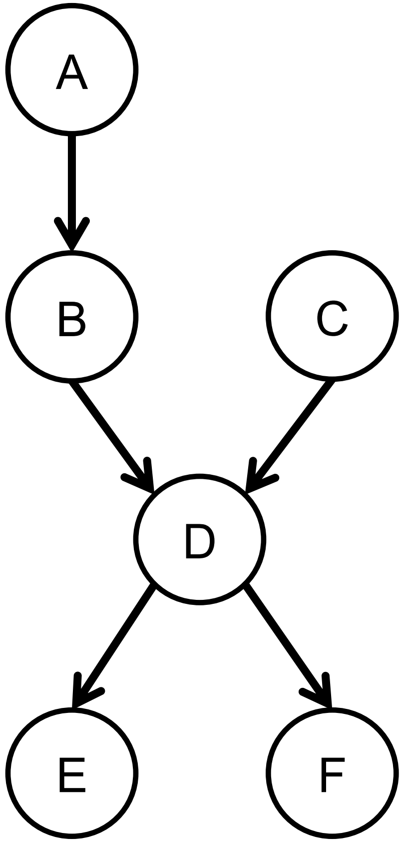
Introduction

- In this lecture, we discuss a class of approximate inference algorithms based on belief propagation
  - Belief propagation was introduced as an exact algorithm for networks with polytree structure
  - Later, applied to networks with arbitrary structure and produced high-quality approximations in certain cases
- We introduce generalization of the algorithm with a full spectrum of approximations
  - Belief propagation approximation at one end
  - Exact results at the other

Belief Propagation

- Belief propagation is a messaging-passing algorithm
  - Originally developed for exact inference in polytrees networks
  - A polytree is a network with only one undirected path between any two nodes

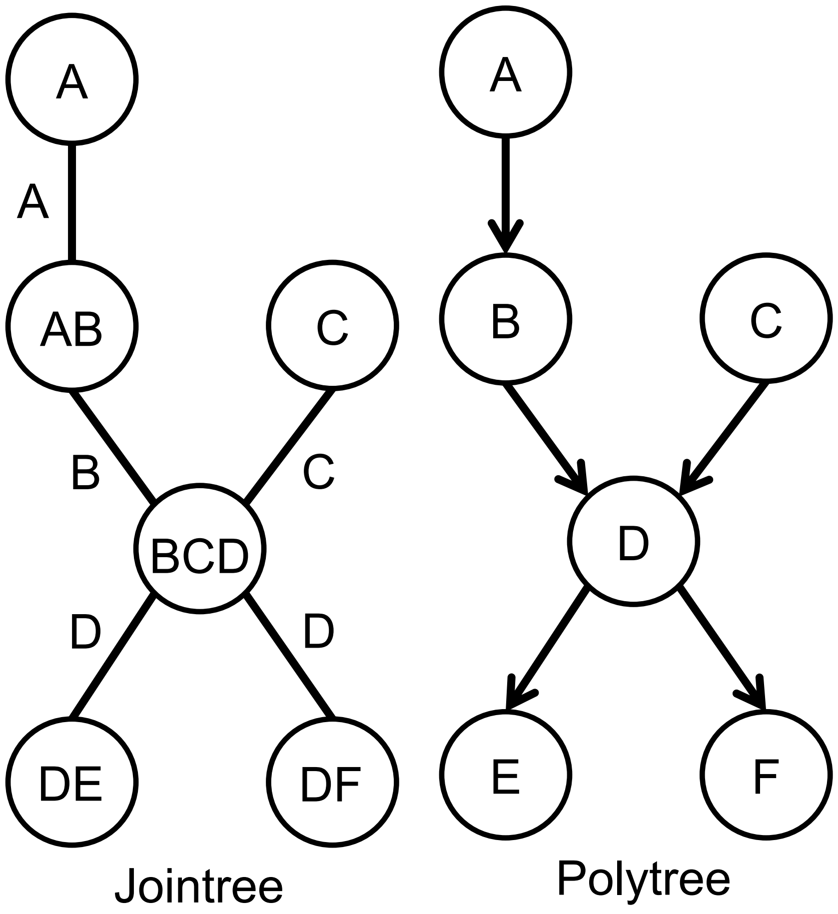
- The exact algorithm is a variation of the jointree
  - It computes $P(X, \boldsymbol{e})$ for every variable in the polytree
  - We discuss the approximate algorithm later on



Polytree

Belief Propagation

- Suppose we want to apply the jointree algorithm under evidence $E = true$
  - In this case, we can create a "special" jointree has the same structure as the polytree
  - A node i in the jointree has cluster $\boldsymbol{C}_i = X\boldsymbol{U}$, where $\boldsymbol{U}$ are the parents of X
  - Edge $U \rightarrow X$ in the jointree has separator $\boldsymbol{S}_{ij} = U$
- Therefore
  - Jointree width equals polytree treewidth
  - Each jointree message is over a single variable

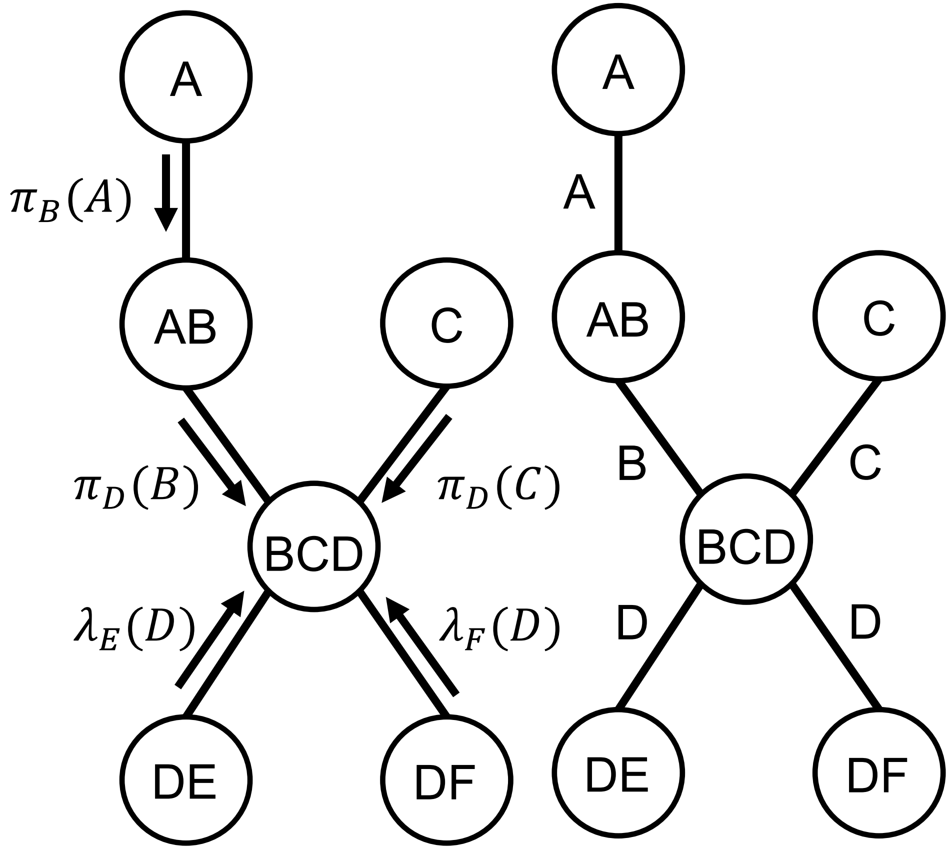


Belief Propagation

- *Belief propagation* is the jointree algorithm under these circumstances
  - Messages are notated differently based on the polytree
  - Message from node U to child X is denoted $\pi_X(U)$ (*causal support*)
  - Messages from node Y to parent X is denoted $\lambda_Y(X)$ (*diagnostic support*)

- The joint marginal for the family of variable X with parents U_i and children Y_i is given by

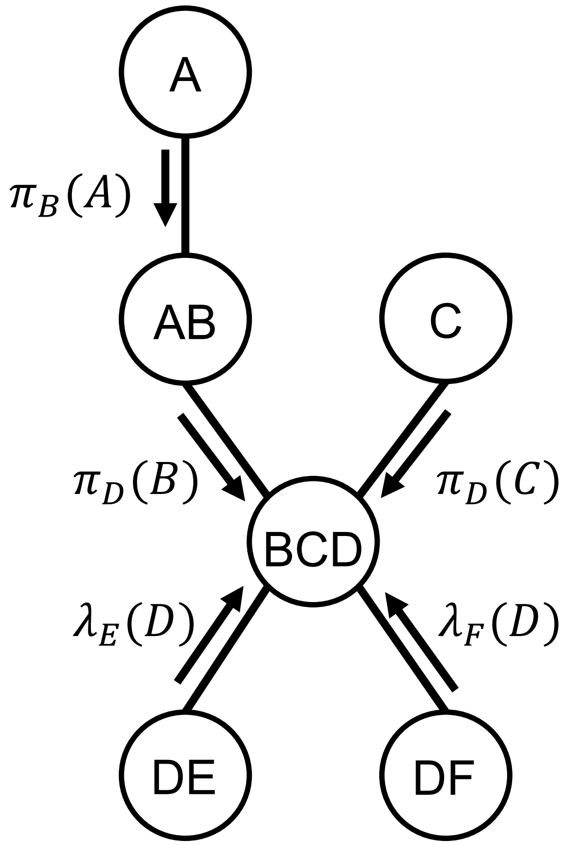
$$P(X\boldsymbol{U}) = \phi_X(X, \boldsymbol{U}) \prod_i \pi_X(U_i) \prod_j \lambda_{Y_j}(X)$$



Belief Propagation with Evidence

- In the presence of evidence, the belief propagation uses an evidence indicator $\lambda_{\boldsymbol{e}}(X)$
  - $\lambda_{\boldsymbol{e}}(x)$ = 1 if x is consistent with the evidence $\boldsymbol{e}$ and zero otherwise
  - We can rewrite the joint marginal for the family of variable X with parents U_i, children Y_i and evidence $\boldsymbol{e}$ as

$$P(X\boldsymbol{U}, \boldsymbol{e}) = \lambda_{\boldsymbol{e}}(X)\ \phi_X(X, \boldsymbol{U}) \prod_i \pi_X(U_i) \prod_j \lambda_{Y_j}(X)$$



Belief Propagation with Evidence

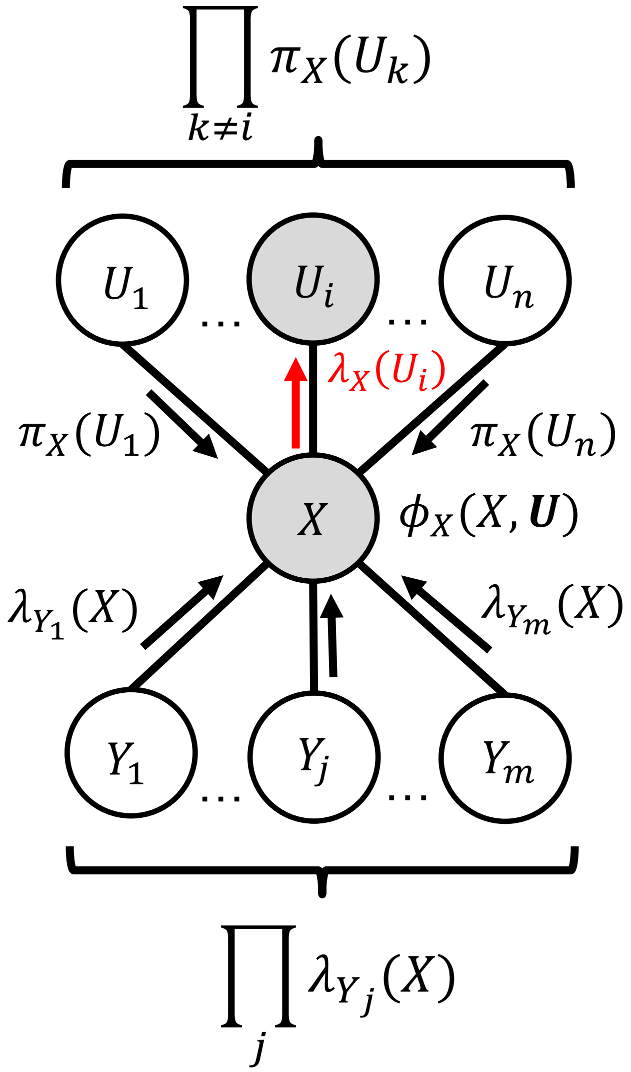
- Using this notation, diagnostic messages can be defined as

$$\lambda_X(U_i) = \sum_{X\boldsymbol{U}\setminus\{U_i\}} \lambda_e(X)\,\phi_X(X,\boldsymbol{U}) \prod_{k\neq i} \pi_X(U_k) \prod_j \lambda_{Y_j}(X)$$

- And causal messages as

$$\pi_{Y_j}(X) = \sum_{\boldsymbol{U}} \lambda_e(X)\,\phi_X(X,\boldsymbol{U}) \prod_i \pi_X(U_i) \prod_{k\neq j} \lambda_{Y_k}(X)$$

- A node can send a message to a neighbour only after it has received messages from all other neighbours



Belief Propagation with Evidence

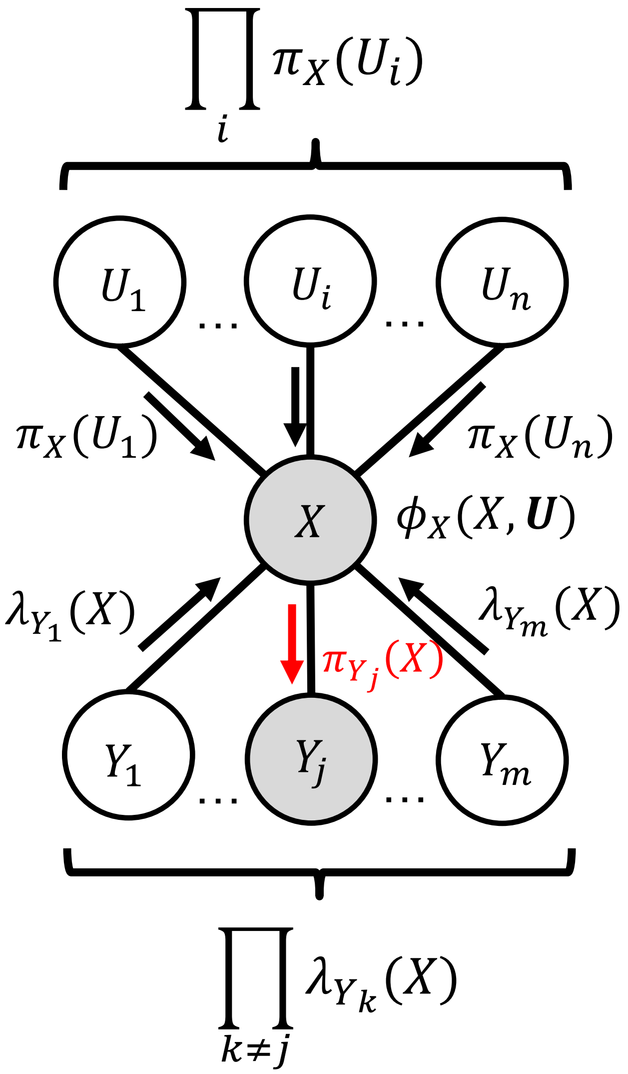
- Using this notation, diagnostic messages can be defined as

$$\lambda_X(U_i) = \sum_{X\boldsymbol{U}\setminus\{U_i\}} \lambda_e(X)\,\phi_X(X,\boldsymbol{U}) \prod_{k\neq i} \pi_X(U_k) \prod_j \lambda_{Y_j}(X)$$

- And causal messages as

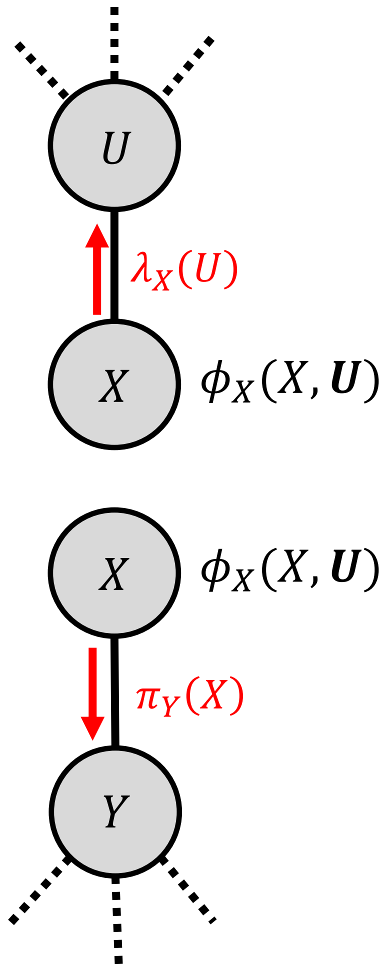
$$\pi_{Y_j}(X) = \sum_{\boldsymbol{U}} \lambda_e(X)\,\phi_X(X,\boldsymbol{U}) \prod_i \pi_X(U_i) \prod_{k\neq j} \lambda_{Y_k}(X)$$

- A node can send a message to a neighbour only after it has received messages from all other neighbours



Belief Propagation with Evidence

- When a node has a single neighbour, it can immediately send a message to that neighbour
  - This includes a leaf node X with a single parent U
$$\lambda_X(U) = \sum_X \lambda_e(X)\, \phi_X(X, U)$$
  - And a root node X with a single child Y
$$\pi_Y(X) = \lambda_e(X)\phi_X(X, U)$$
- These are the base cases for belief propagation
  - These messages can be computed immediately as they do not depend on any other messages
  - Typically, messages are first propagated toward a root and them pushed away from root



Belief Propagation: Example

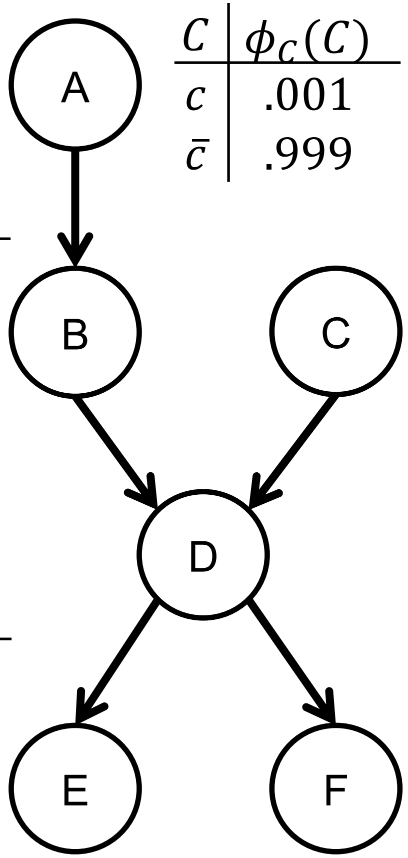
$$P(B,C,D,\boldsymbol{e}) = \phi_D(D,B,C)\pi_D(B)\pi_D(C)\lambda_E(D)\lambda_F(D)$$

$\boldsymbol{e}$: $\{E = true\}$

A	$\phi_A(A)$
a	.01
$\bar{a}$	.99

A	B	$\phi_B(B,A)$
a	b	.100
a	$\bar{b}$	.900
$\bar{a}$	b	.001
$\bar{a}$	$\bar{b}$	.999

D	E	$\phi_E(E,D)$
d	e	.9
d	$\bar{e}$	.1
$\bar{d}$	e	.3
$\bar{d}$	$\bar{e}$	.7



C	$\phi_C(C)$
c	.001
$\bar{c}$	.999

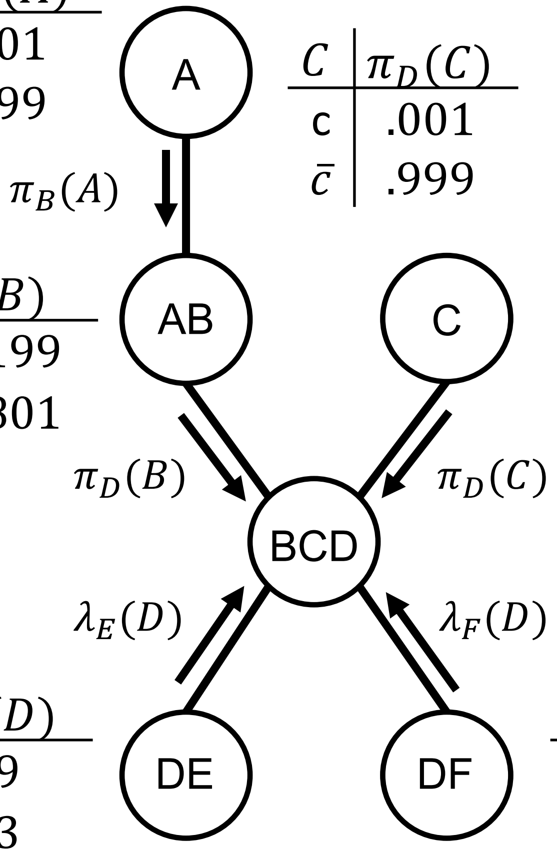
B	C	D	$\phi_D(D,B,C)$
b	c	d	.99
b	c	$\bar{d}$	.01
b	$\bar{c}$	d	.90
b	$\bar{c}$	$\bar{d}$	.10
$\bar{b}$	c	d	.95
$\bar{b}$	c	$\bar{d}$	.05
$\bar{b}$	$\bar{c}$	d	.01
$\bar{b}$	$\bar{c}$	$\bar{d}$	.99

D	F	$\phi_F(F,D)$
d	f	.2
d	$\bar{f}$	.8
$\bar{d}$	f	.1
$\bar{d}$	$\bar{f}$	.9

A	$\pi_B(A)$
a	.01
$\bar{a}$	.99

B	$\pi_D(B)$
b	.00199
$\bar{b}$	.99801

D	$\lambda_E(D)$
d	.9
$\bar{d}$	.3



C	$\pi_D(C)$
c	.001
$\bar{c}$	.999

D	$\lambda_F(D)$
d	1
$\bar{d}$	1

B	C	D	$P(B,C,D,\boldsymbol{e})$
b	c	d	1.7731×10^{-6}
b	c	$\bar{d}$	5.9700×10^{-9}
b	$\bar{c}$	d	1.6103×10^{-3}
b	$\bar{c}$	$\bar{d}$	5.9640×10^{-5}
$\bar{b}$	c	d	8.5330×10^{-4}
$\bar{b}$	c	$\bar{d}$	1.4970×10^{-5}
$\bar{b}$	$\bar{c}$	d	8.9731×10^{-3}
$\bar{b}$	$\bar{c}$	$\bar{d}$	2.9611×10^{-1}

Belief Propagation: Example

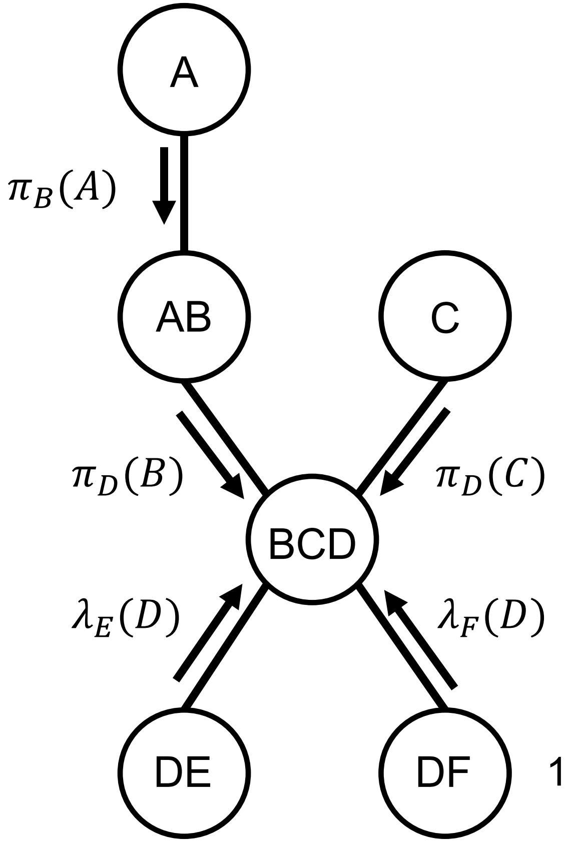
- We can use $P(B, C, D, \boldsymbol{e})$ to compute marginals for the variables B, C and D. For instance
  - We can also compute the joint marginal for C once we compute the message from D to C
  - To compute conditional marginals, we simply normalize joint marginals
- Another approach is to use a constant η that normalizes the factor to sum to one

$$P(X\boldsymbol{U}|\boldsymbol{e}) = \eta\, \lambda_{\boldsymbol{e}}(X)\, \phi_X(X, \boldsymbol{U}) \prod_i \pi_X(U_i) \prod_j \lambda_{Y_j}(X)$$

$$\lambda_X(U_i) = \eta \sum_{X\boldsymbol{U}\setminus\{U_i\}} \lambda_e(X)\, \phi_X(X, \boldsymbol{U}) \prod_{k \neq i} \pi_X(U_k) \prod_j \lambda_{Y_j}(X)$$

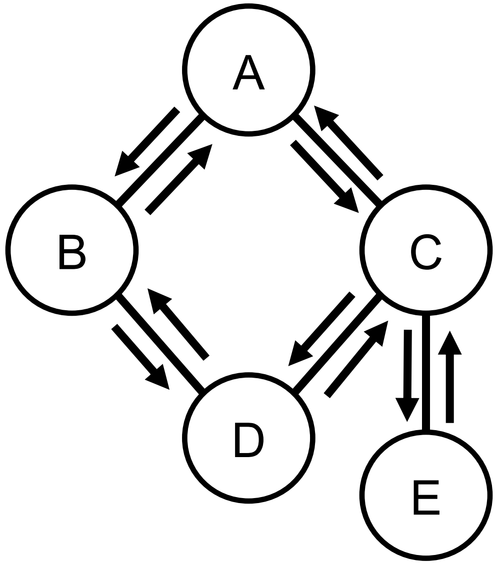
$$\pi_{Y_j}(X) = \eta \sum_{\boldsymbol{U}} \lambda_e(X)\, \phi_X(X, \boldsymbol{U}) \prod_i \pi_X(U_i) \prod_{k \neq j} \lambda_{Y_k}(X)$$

C	$P(C, \boldsymbol{e})$
c	.0009
$\bar{c}$	.3067



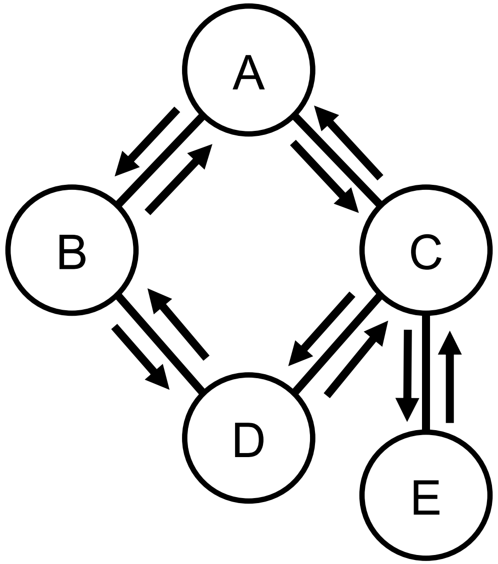
Belief Propagation in Connected Networks

- Belief propagation was designed as an exact algorithm for polytrees
  - However, it was later applied to connected networks
- This application poses some difficulties
  - A message can be sent from X to Y only when X has received all messages from other neighbours
  - The correctness of belief propagation depends on the underlying polytree
- The results can be incorrect if applied to connected networks
  - The algorithm is no longer always correct
  - But can still provide some high-quality approximations in many cases
- In the figure, after node E send a message to C no other message can be propagated
  - Since each is dependent on others that are waiting to be propagated



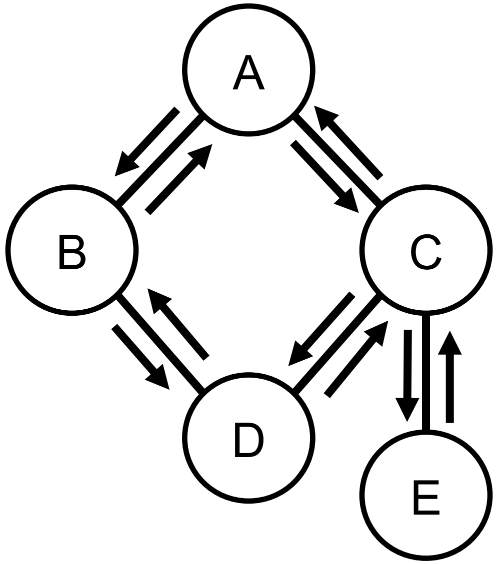
Iterative Belief Propagation (IBP)

- *Iterative* or *Loopy Belief Propagation* assumes some initial value to each message in the network
  - Given these initial values, each node is ready to send a message to each of its neighbours
  - At each iteration t, every node X send a message to its neighbours using the messages received from its other neighbours in $t-1$
- The algorithm iterates until message convergence
  - The value of messages at the current iteration are within some threshold from their values at the previous iteration
  - When IBP converges, the values of the messages at convergence are called *fixed point*
  - IBP may have multiple fixed points on a given network



Message Schedule

- For some networks, IBP can oscillate and never converge
- The convergence rate can depend on the order the messages are propagated, which is known as *message schedule*
  - Parallel schedule: the order of the messages does not affect the algorithm
  - Sequential schedule: messages are propagates as soon as they are computed
- Sequential schedules are flexible in when and how quickly information is propagated
- Although one schedule may converge and other may not, all schedules have the same fixed points



Parallel Iterative Belief Propagation

$t \leftarrow 0$

initialize all messages

while messages have not converged **do**

$t \leftarrow t + 1$

for each node X with parents $\boldsymbol{U}$ **do**

for each parent U_i **do**

$\lambda_X^t(U_i) = \eta \sum_{X\boldsymbol{U}\setminus\{U_i\}} \lambda_e(X)\, \phi_X(X, \boldsymbol{U}) \prod_{k \neq i} \pi_X^{t-1}(U_k) \prod_j \lambda_{Y_j}^{t-1}(X)$

for each child Y_j **do**

$\pi_{Y_j}^t(X) = \eta \sum_{\boldsymbol{U}} \lambda_e(X)\, \phi_X(X, \boldsymbol{U}) \prod_i \pi_X^{t-1}(U_i) \prod_{k \neq j} \lambda_{Y_k}^{t-1}(X)$

return $\beta(X\boldsymbol{U}) = P(X\boldsymbol{U}|\boldsymbol{e}) = \eta\, \lambda_{\boldsymbol{e}}(X)\, \phi_X(X, \boldsymbol{U}) \prod_i \pi_X^t(U_{\boldsymbol{i}}) \prod_j \lambda_{Y_j}^t(X)$

The Kullback-Leibler Divergence

- The *Kullback-Leibler divergence*, known as *KL divergence,* between two distributions P and P' conditioned on $\boldsymbol{e}$

$$KL(P'(\boldsymbol{X}|\boldsymbol{e}), P(\boldsymbol{X}|\boldsymbol{e})) \stackrel{\text{def}}{=} \sum_{\boldsymbol{x}} P'(\boldsymbol{x}|\boldsymbol{e}) \log \frac{P'(\boldsymbol{x}|\boldsymbol{e})}{P(\boldsymbol{x}|\boldsymbol{e})}$$

- $KL(P'(\boldsymbol{X}|\boldsymbol{e}), P(\boldsymbol{X}|\boldsymbol{e}))$ is non-negative and equal to zero if and only if $P'(\boldsymbol{X}|\boldsymbol{e})$ and $P(\boldsymbol{X}|\boldsymbol{e})$ are equivalent
  - However, KL divergence is not a true distance since it is not symmetric. In general
    $$KL\big(P'(\boldsymbol{X}|\boldsymbol{e}), P(\boldsymbol{X}|\boldsymbol{e})\big) \neq KL(P(\boldsymbol{X}|\boldsymbol{e}), P'(\boldsymbol{X}|\boldsymbol{e}))$$
  - We say we are weighting the KL divergence by the approximate distribution
  - This variation has some useful computational properties

Optimizing KL Divergence

- The approximate inference can be posed as an optimization problem
  - The goal is to search for an approximate distribution P' that minimizes KL divergence with P
  - We can assume a parametrized form for P' and search for the best instance, i.e., the best set of parameters

- The Iterative Belief Propagation algorithm presented before assumes that the approximate distribution $P'(X)$ factors as

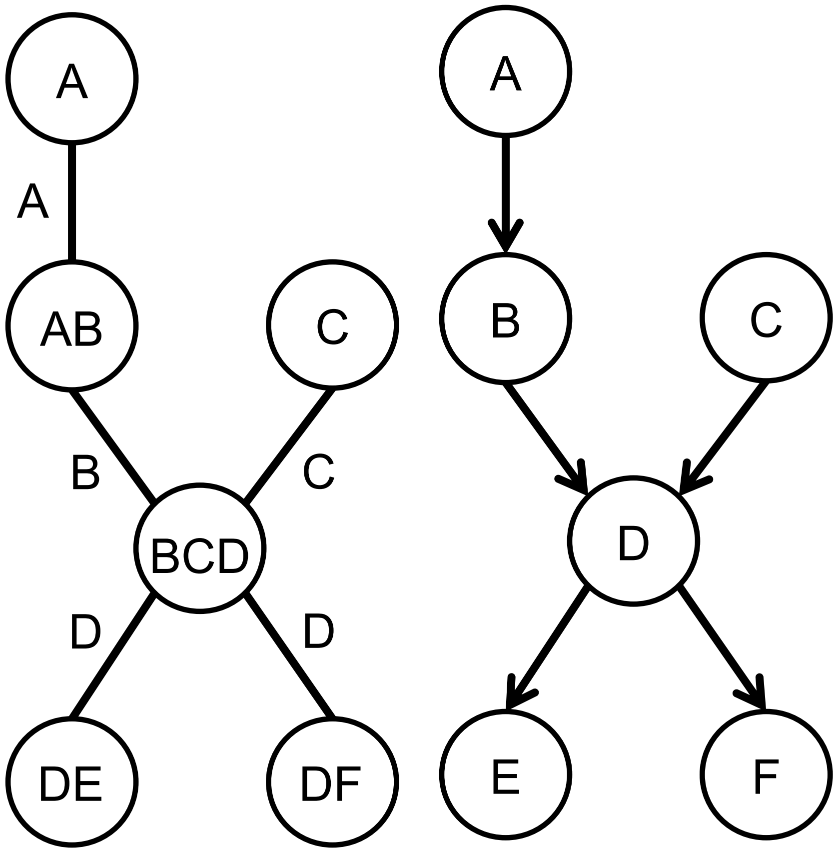
$$P'(\boldsymbol{X}|\boldsymbol{e}) = \prod_{X\boldsymbol{U}} \frac{P'(X\boldsymbol{U}|\boldsymbol{e})}{\prod_{U \in \boldsymbol{U}} P'(U|\boldsymbol{e})}$$

  - $X\boldsymbol{U}$ ranges over the families of the network N
  - U ranges over nodes that appear as parents in N

Optimizing KL Divergence

- The approximate distribution $P'(X)$ factors as
$$P'(\boldsymbol{X}|\boldsymbol{e}) = \prod_{X\boldsymbol{U}} \frac{P'(X\boldsymbol{U}|\boldsymbol{e})}{\prod_{U\in\boldsymbol{U}} P'(U|\boldsymbol{e})}$$
  - $X\boldsymbol{U}$ ranges over the families of the network N
  - U ranges over nodes that appear as parents in N

- Some observations about this assumption
  - This choice of $P'(\boldsymbol{X}|\boldsymbol{e})$ is expressive enough to describe distributions induced by polytree networks
  - If the network N is a polytree then $P(\boldsymbol{X}|\boldsymbol{e})$ does factor according to this equation (see figure for an example)
  - If N is not a polytree, then we are trying to fit $P(\boldsymbol{X}|\boldsymbol{e})$ into an approximation $P'(\boldsymbol{X}|\boldsymbol{e})$ as if it were generated by a polytree



$$P(A,B,C,D,E,F) = \frac{P(A)P(C)P(B,A)\ P(D,B,C)P(E,D)P(F,D)}{P(A)P(B)P(C)P(D)P(D)}$$

Optimizing KL Divergence

- The previous correspondence that IBP fixed points are stationary points of the KL divergence
  - They may or may not be local minima
  - When IBP performs well, it often has fixed points that are minima of the KL divergence
  - Otherwise, we need to seek approximations P' whose factorizations are more expressive than the polytree-based factorization

- If we do not insist on marginals being over families and individual variables, we can have a more general form that covers every distribution

Generalized Belief Propagation

- We saw in the previous lecture that a network can be factorized according to this this expression if
  - $\boldsymbol{C}$ corresponds to the clusters of a jointree
  - $\boldsymbol{S}$ corresponds to the separators
- If we base our factorization in a jointree
  - Solving the previous optimization problem yields the same update equations of the jointree algorithm
- Therefore, the factorizations used by IBP and the factorization based on jointrees can be viewed as two extremes
  - One efficient but approximate
  - The other expensive but exact

$$P'(\boldsymbol{X}|\boldsymbol{e}) = \frac{\prod_{\boldsymbol{C}} P'(\boldsymbol{C}|\boldsymbol{e})}{\prod_{\boldsymbol{S}} P'(\boldsymbol{S}|\boldsymbol{e})}$$

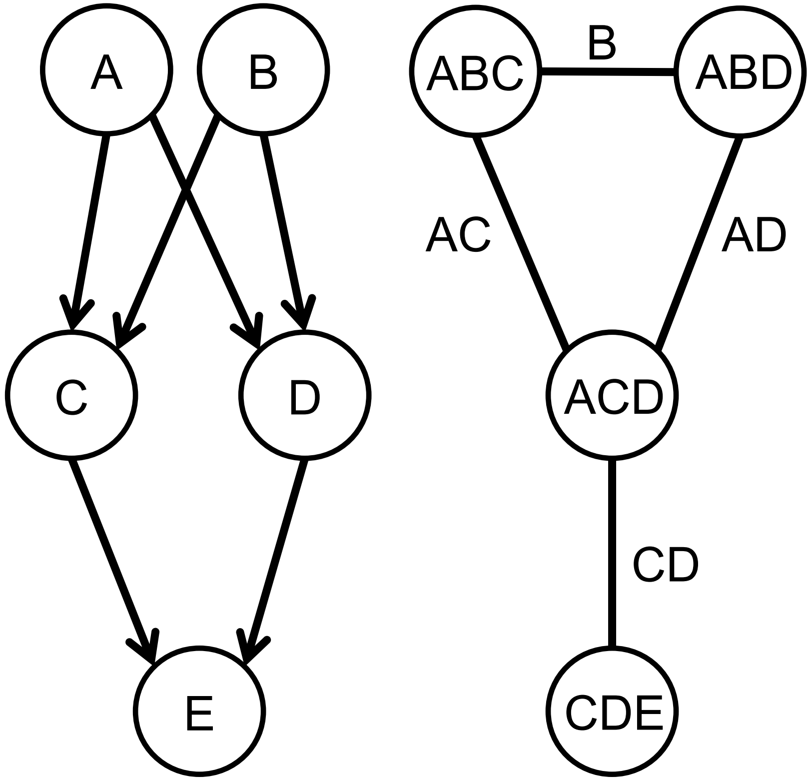
Joingraphs

- There is a spectrum of factorizations that fall in between these two extremes
  - This allows a trade-off quality and efficiency
  - The notion of joingraph is one way to obtain such a spectrum

- *Joingraphs* are generalizations of jointrees
  - They can be used to obtain factorizations according to $P'(\boldsymbol{X}|\boldsymbol{e}) = \frac{\prod_{\boldsymbol{C}} P'(\boldsymbol{C}|\boldsymbol{e})}{\prod_{\boldsymbol{S}} P'(\boldsymbol{S}|\boldsymbol{e})}$
  - They are used to formulate a message-passing algorithm similar to IBF, known as *iterative joingraph propagation*

Joingraphs

- A *joingraph* G for a network N is a graph where nodes i are labelled by cluster $\boldsymbol{C}_i$, and edges $i - j$ are labelled by separators $\boldsymbol{S}_{ij}$. Moreover, G satisfies the following properties
  - Clusters $\boldsymbol{C}_i$ and separators $\boldsymbol{S}_{ij}$ are sets of nodes from N
  - Each factor in N must appear in some cluster $\boldsymbol{C}_i$
  - If a variable X appears in two clusters $\boldsymbol{C}_i$ and $\boldsymbol{C}_j$, then there exists a unique path connecting i and j in the joingraph such that X appears in every cluster and separator on that path
  - For every edge $i - j$ in the joingraph, $\boldsymbol{S}_{ij} \subseteq \boldsymbol{C}_i \cap \boldsymbol{C}_j$

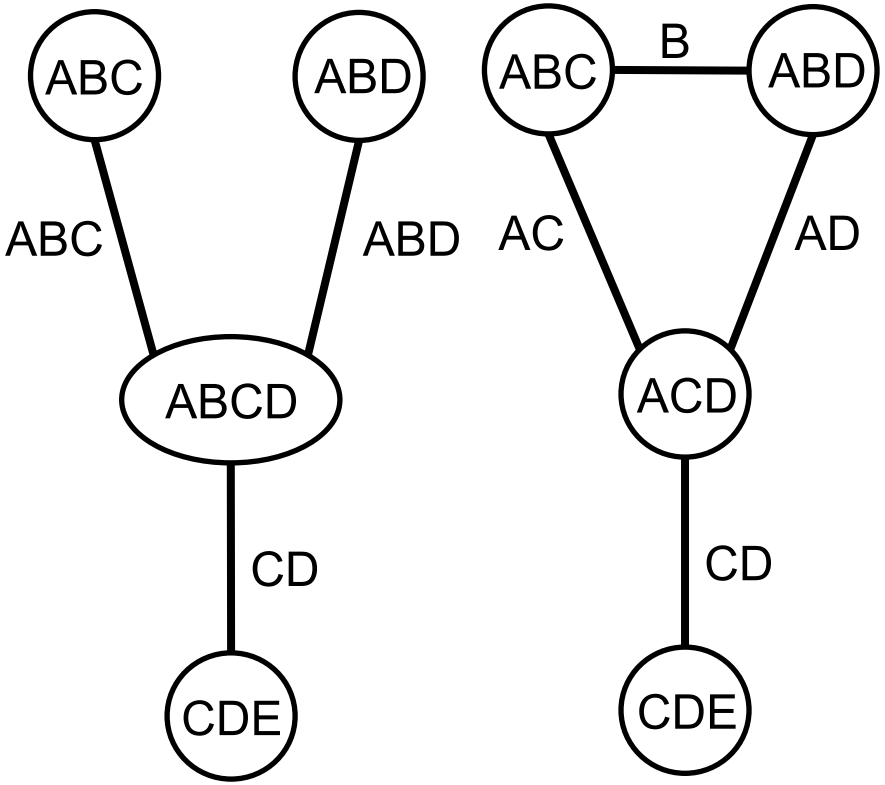


Bayes net

Joingraph

Jointrees and Joingraphs

- We can think of a jointgraph as a way of relaxing some constraints of jointrees
  - In a jointree, if two clusters $\boldsymbol{C}_i$ and $\boldsymbol{C}_j$ share a set of variables $\boldsymbol{X}$ then every cluster and separator on the path connecting $\boldsymbol{C}_i$ and $\boldsymbol{C}_j$ must contain $\boldsymbol{X}$
  - In a joingraph, we assert each variable $X \in \boldsymbol{X}$ be contained in clusters and separators of some path connecting $\boldsymbol{C}_i$ and $\boldsymbol{C}_j$
  - We do not require separators $\boldsymbol{S}_{ij}$ to be precisely the intersection of $\boldsymbol{C}_i$ and $\boldsymbol{C}_j$, as in the case of jointrees

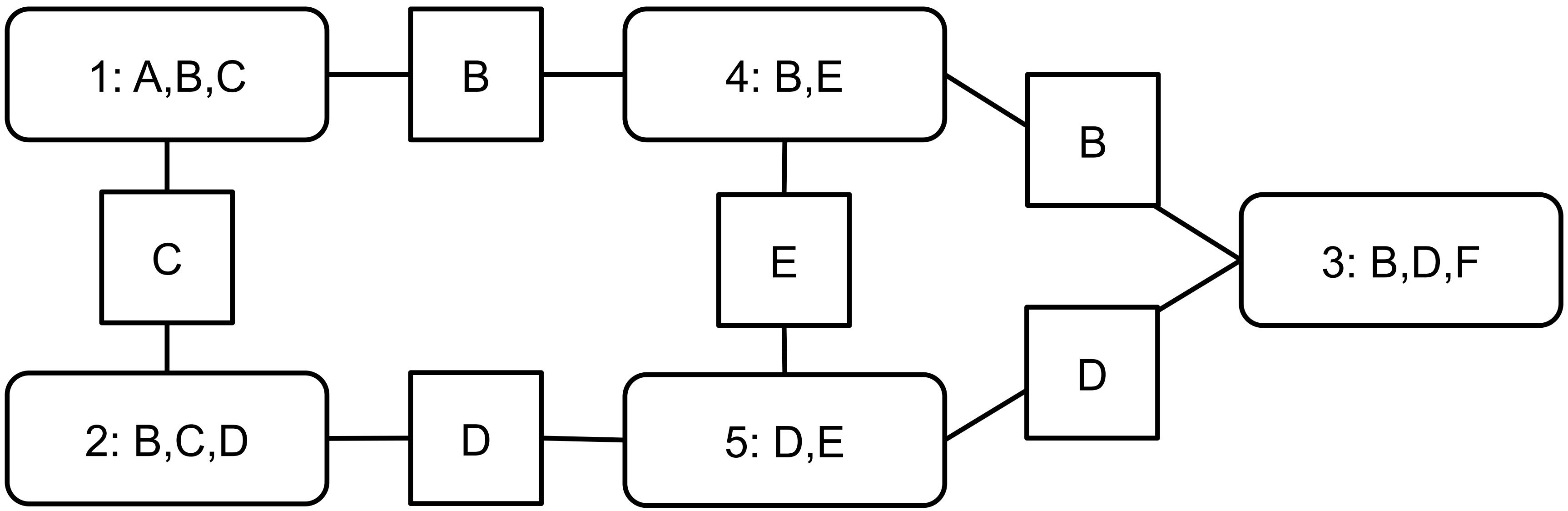


Jointree

Joingraph

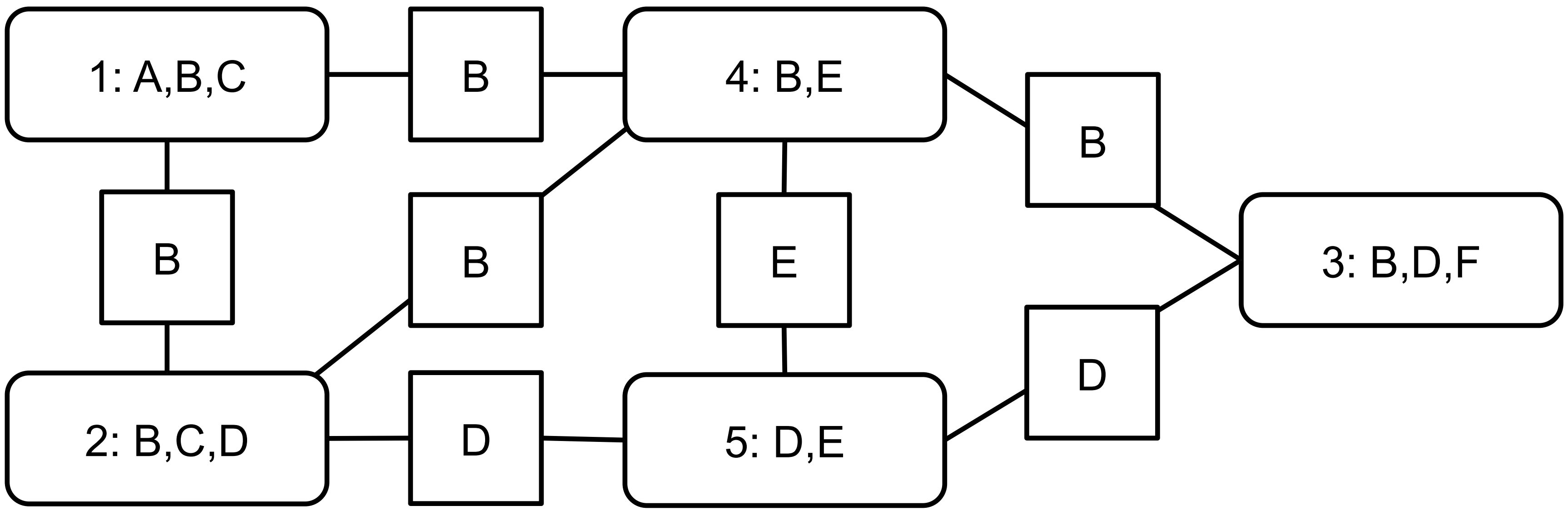
Valid Joingraph?

$$\phi_1(A,B,C), \phi_2(B,C), \phi_3(B,D), \phi_4(D,E), \phi_5(B,E), \phi_6(B,D), \phi_7(B,D,F)$$



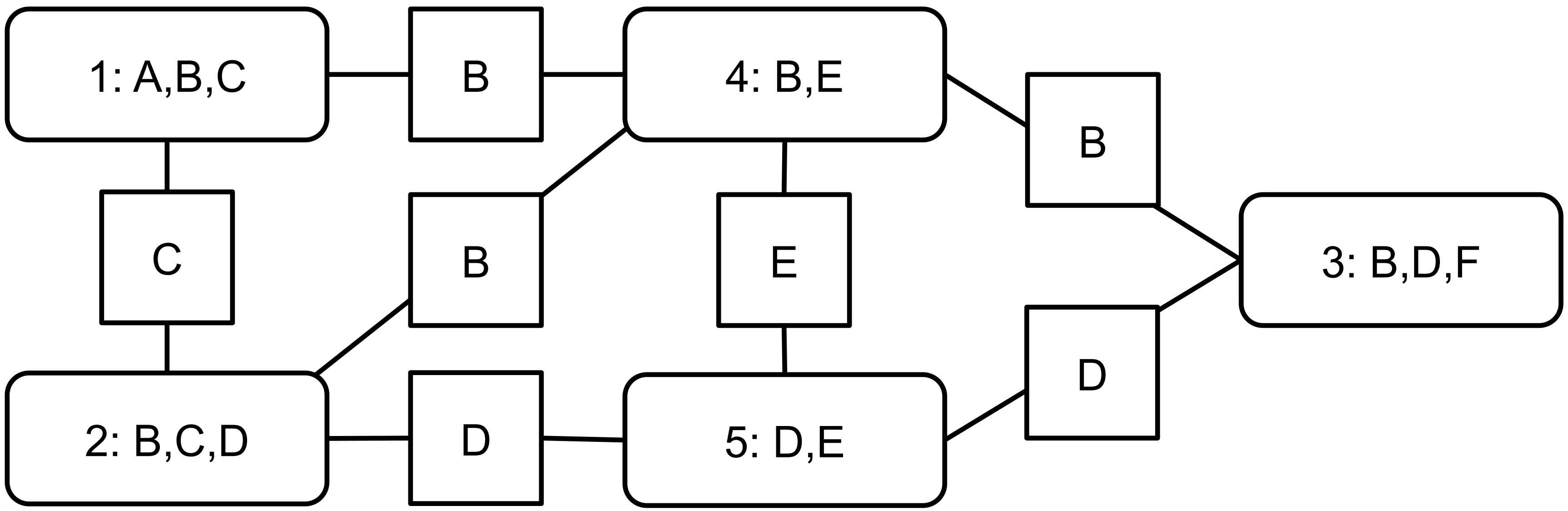
Valid Joingraph?

$$\phi_1(A,B,C), \phi_2(B,C), \phi_3(B,D), \phi_4(D,E), \phi_5(B,E), \phi_6(B,D), \phi_7(B,D,F)$$



Valid Joingraph?

$$\phi_1(A,B,C), \phi_2(B,C), \phi_3(B,D), \phi_4(D,E), \phi_5(B,E), \phi_6(B,D), \phi_7(B,D,F)$$

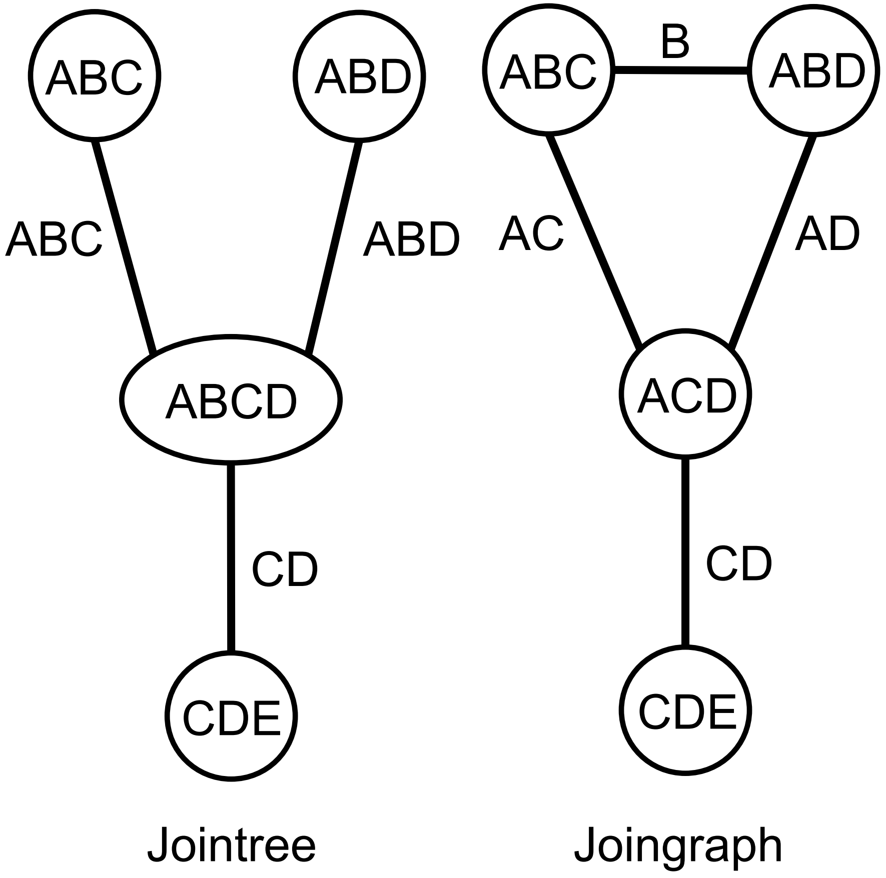


Joingraph Factorization

- A joingraph induces an approximate factorization

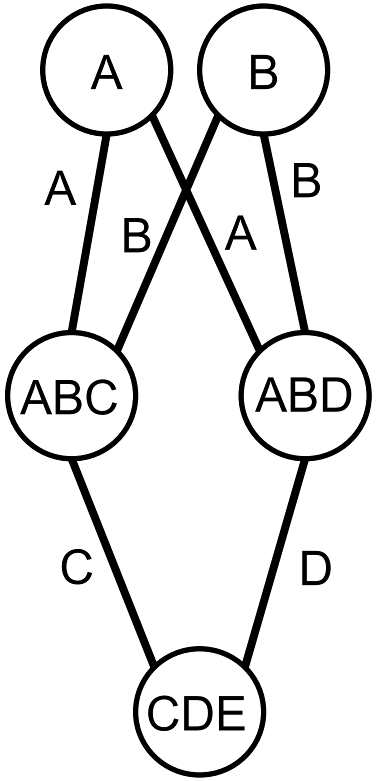
$$P'(\boldsymbol{X}|\boldsymbol{e}) = \frac{\prod_i P'(\boldsymbol{C}_i|\boldsymbol{e})}{\prod_{ij} P'(\boldsymbol{S}_{ij}|\boldsymbol{e})}$$

- When the joingraph corresponds to a jointree, the factorization is exact

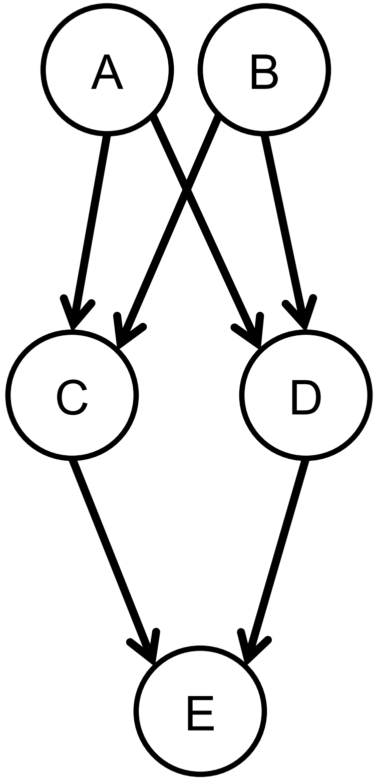


Dual Joingraph

- A *dual joingraph* is a special joingraph whose factorization reduces to the one used by IBP
- A dual joingraph G for a network N is obtained as follows
  - G has the same undirected structure as N
  - For each family $X\boldsymbol{U}$ in N, the corresponding node i in G has the cluster $\boldsymbol{C}_i = X\boldsymbol{U}$
  - For each $U \rightarrow X$ in N, the corresponding edge $i - j$ in G has separator $\boldsymbol{S}_{ij} = U$



Dual Joingraph

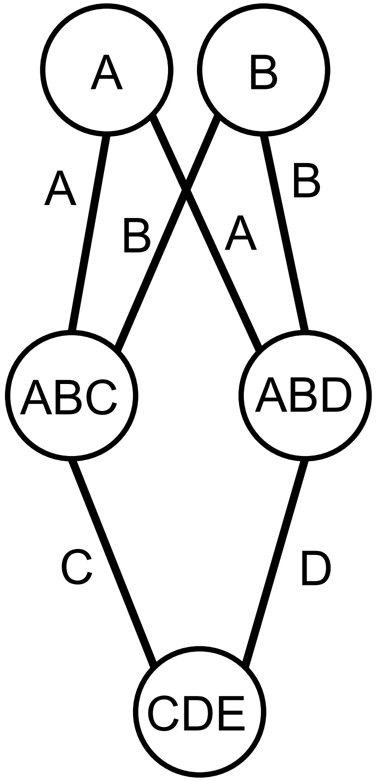


Bayes net

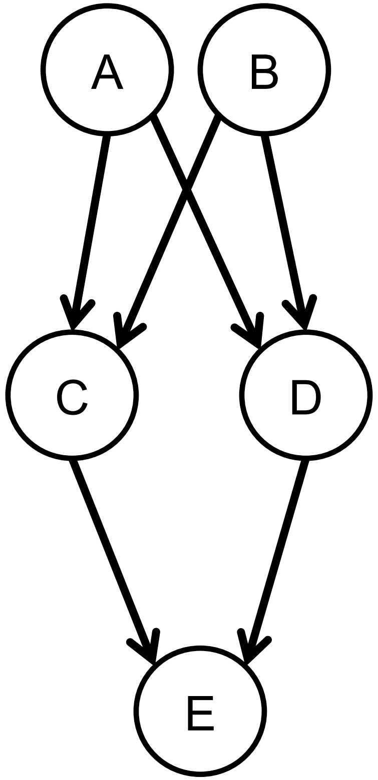
Factorization Comparison

- Dual jointgraph (approximate, same as IBP)

$$P'(\boldsymbol{X}|\boldsymbol{e}) = \frac{P'(A|\boldsymbol{e})P'(B|\boldsymbol{e})P'(A,B,C|\boldsymbol{e})P'(A,B,D|\boldsymbol{e})P'(C,D,E|\boldsymbol{e})}{P'(A|\boldsymbol{e})^2 P'(B|\boldsymbol{e})^2 P'(C|\boldsymbol{e})P'(D|\boldsymbol{e})}$$



Dual Joingraph



Bayes net

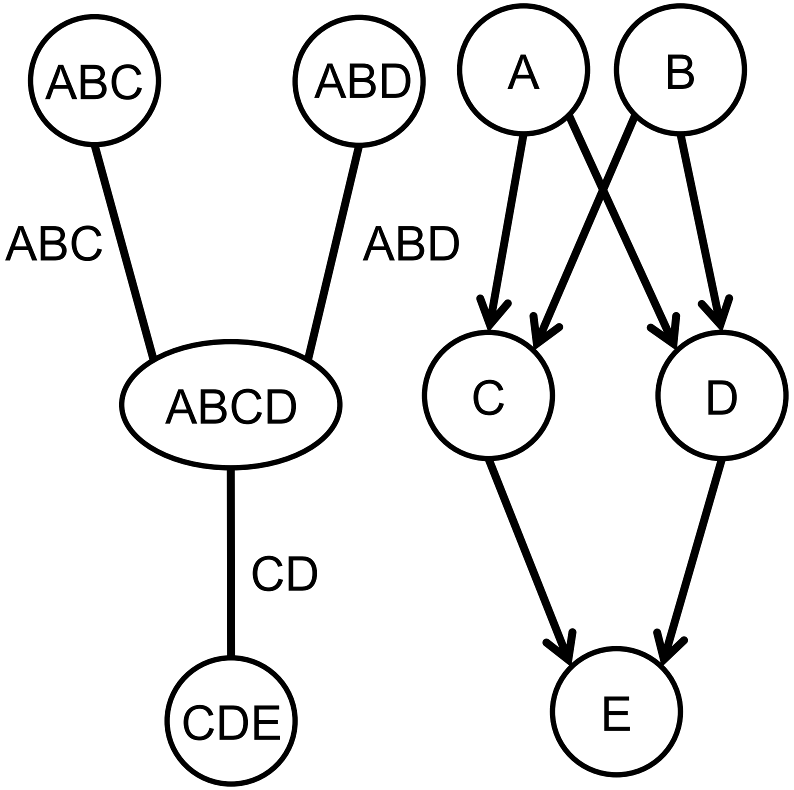
Factorization Comparison

- Dual jointgraph (approximate, same as IBP)

$$P'(\boldsymbol{X}|\boldsymbol{e}) = \frac{P'(A|\boldsymbol{e})P'(B|\boldsymbol{e})P'(A,B,C|\boldsymbol{e})P'(A,B,D|\boldsymbol{e})P'(C,D,E|\boldsymbol{e})}{P'(A|\boldsymbol{e})^2 P'(B|\boldsymbol{e})^2 P'(C|\boldsymbol{e})P'(D|\boldsymbol{e})}$$

- Jointree (exact)

$$P'(\boldsymbol{X}|\boldsymbol{e}) = \frac{P'(A,B,C|\boldsymbol{e})P'(A,B,D|\boldsymbol{e})P'(A,B,C,D|\boldsymbol{e})P'(C,D,E|\boldsymbol{e})}{P'(A,B,C|\boldsymbol{e})P'(A,B,D|\boldsymbol{e})P'(C,D|\boldsymbol{e})}$$



Jointree Bayes net

Factorization Comparison

- Dual jointgraph (approximate, same as IBP)

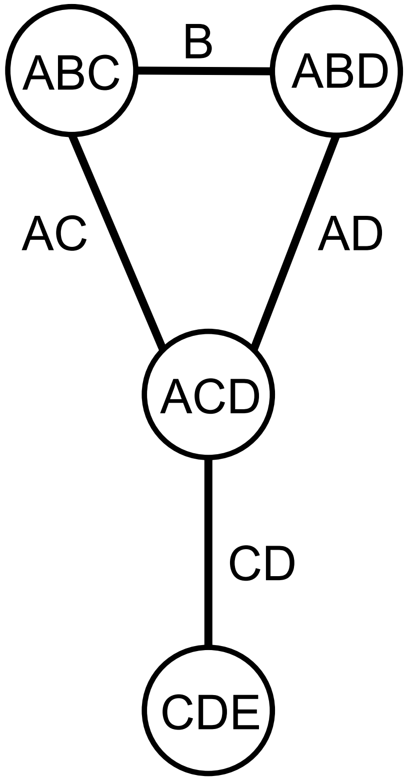
$$P'(\boldsymbol{X}|\boldsymbol{e}) = \frac{P'(A|\boldsymbol{e})P'(B|\boldsymbol{e})P'(A,B,C|\boldsymbol{e})P'(A,B,D|\boldsymbol{e})P'(C,D,E|\boldsymbol{e})}{P'(A|\boldsymbol{e})^2 P'(B|\boldsymbol{e})^2 P'(C|\boldsymbol{e})P'(D|\boldsymbol{e})}$$

- Jointree (exact)

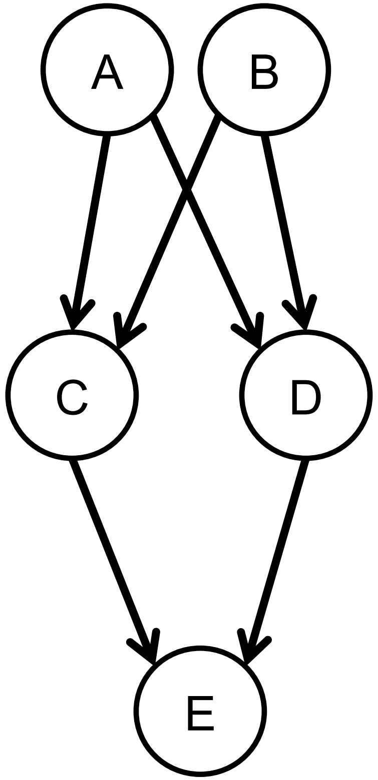
$$P'(\boldsymbol{X}|\boldsymbol{e}) = \frac{P'(A,B,C|\boldsymbol{e})P'(A,B,D|\boldsymbol{e})P'(A,B,C,D|\boldsymbol{e})P'(C,D,E|\boldsymbol{e})}{P'(A,B,C|\boldsymbol{e})P'(A,B,D|\boldsymbol{e})P'(C,D|\boldsymbol{e})}$$

- Joingraph (trade complexity and quality)

$$P'(\boldsymbol{X}|\boldsymbol{e}) = \frac{P'(A,B,C|\boldsymbol{e})P'(A,B,D|\boldsymbol{e})P'(A,C,D|\boldsymbol{e})P'(C,D,E|\boldsymbol{e})}{P'(B|\boldsymbol{e})P'(A,C|\boldsymbol{e})P'(A,D|\boldsymbol{e})P'(C,D|\boldsymbol{e})}$$



Joingraph

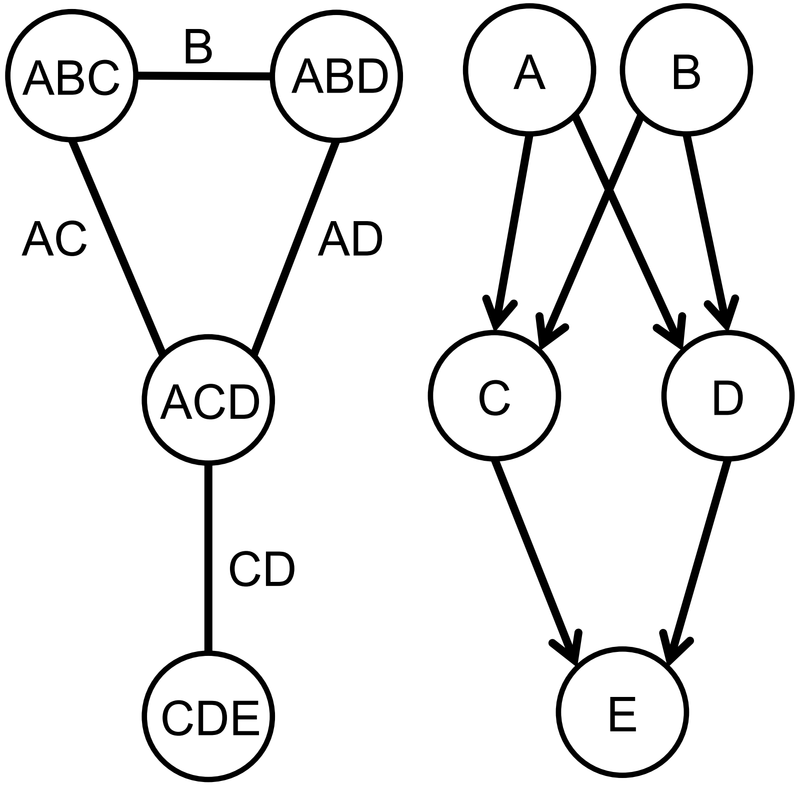


Bayes net

Factorization Comparison

- Joingraph (trade complexity and quality)

$$P'(\boldsymbol{X}|\boldsymbol{e}) = \frac{P'(A,B,C|\boldsymbol{e})P'(A,B,D|\boldsymbol{e})P'(A,C,D|\boldsymbol{e})P'(C,D,E|\boldsymbol{e})}{P'(B|\boldsymbol{e})P'(A,C|\boldsymbol{e})P'(A,D|\boldsymbol{e})P'(C,D|\boldsymbol{e})}$$



Joingraph

Bayes net

Iterative Joingraph Propagation

- Suppose we have a network N that induces a distribution P
  - And a corresponding joingraph that induces a factorization P'
  - Also, we want to compute cluster marginals $P'(\boldsymbol{C}_i|\boldsymbol{e})$ and separator marginals $P'(\boldsymbol{S}_{ij}|\boldsymbol{e})$ that minimize the KL divergence between $P(\boldsymbol{X}|\boldsymbol{e})$ and $P'(\boldsymbol{X}|\boldsymbol{e})$

- This optimization problem can be solved with a generalization of IBP called *interactive joingraph propagation* (IJGP)

Iterative Joingraph Propagation

- The algorithm starts assigning each network factor ϕ and evidence indicator $\lambda_{\boldsymbol{e}}$ to some cluster $\boldsymbol{C}_i$ that contains variables in ϕ
  - All factors are associated to some cluster (no information loss)
  - No factor is present in more than one cluster (no overcounting of information)

- It propagates messages with the equations
  - $M_{ij} = \eta \sum_{\boldsymbol{C}_i \backslash \boldsymbol{S}_{ij}} \psi_i \prod_{k \neq j} M_{ki}$
  - where ψ_i is the product of all CPTs and evidence indicators assigned to cluster $\boldsymbol{C}_i$
  - M_{ij} is the message sent from cluster i to cluster j

Parallel Iterative Joingraph Propagation

$t \leftarrow 0$

initialize all messages

while messages have not converged **do**

  $t \leftarrow t + 1$

  for each joingraph edge $i - j$ **do**

    $M_{ij}^{t} = \eta \sum_{\boldsymbol{C}_i \setminus \boldsymbol{S}_{ij}} \psi_i \prod_{k \neq j} M_{ki}^{t-1}$

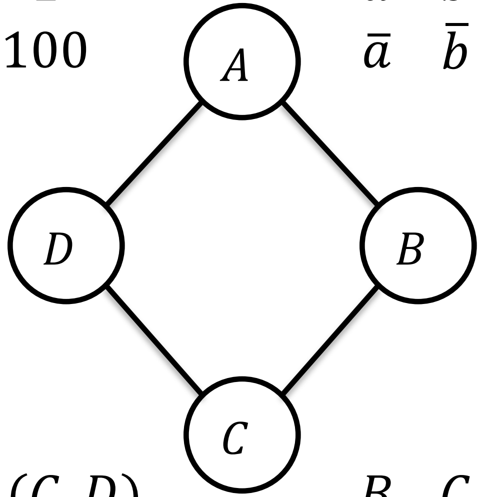
    $M_{ji}^{t} = \eta \sum_{\boldsymbol{C}_i \setminus \boldsymbol{S}_{ij}} \psi_i \prod_{k \neq i} M_{kj}^{t-1}$

return $\beta(\boldsymbol{C}_i) = \eta\, \psi_i \prod_k M_{ki}^{t}$ for each node i

Joingraph Example with Markov Nets

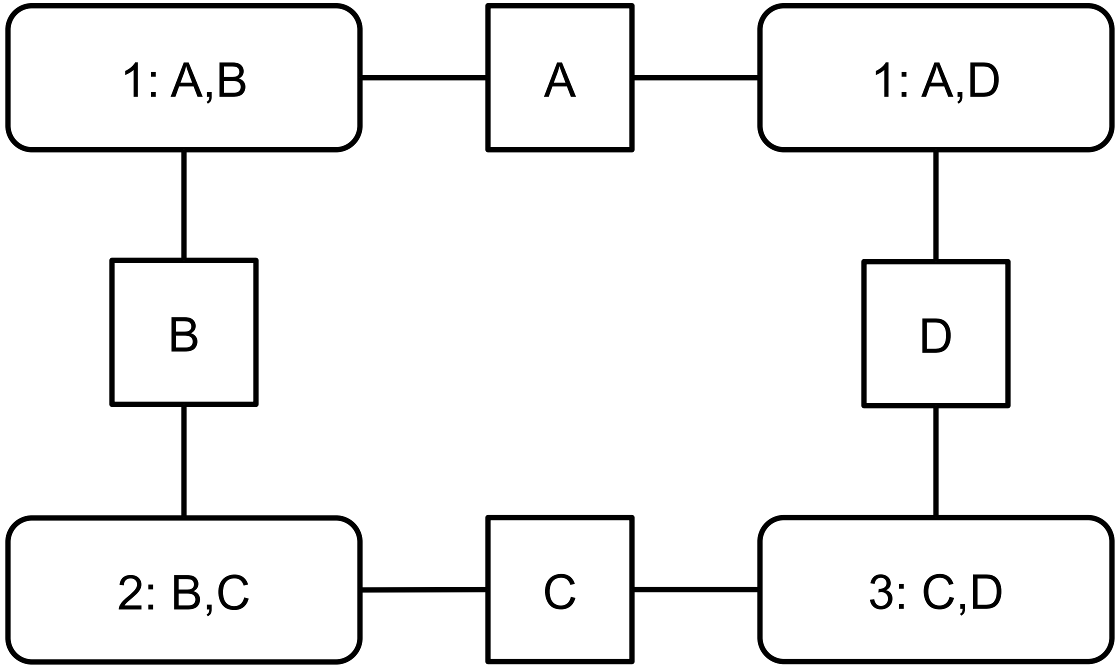
D	A	$\phi_4(D,A)$
d	a	100
d	$\bar{a}$	1
$\bar{d}$	a	1
$\bar{d}$	$\bar{a}$	100

A	B	$\phi_1(A,B)$
a	b	30
a	$\bar{b}$	5
$\bar{a}$	b	1
$\bar{a}$	$\bar{b}$	10



C	D	$\phi_3(C,D)$
c	d	1
c	$\bar{d}$	100
$\bar{c}$	d	100
$\bar{c}$	$\bar{d}$	1

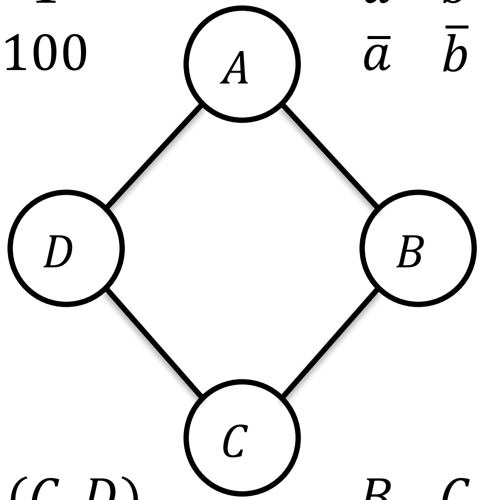
B	C	$\phi_2(B,C)$
b	c	100
b	$\bar{c}$	1
$\bar{b}$	c	1
$\bar{b}$	$\bar{c}$	100



Joingraph Example with Markov Nets

D	A	$\phi_4(D,A)$
d	a	100
d	$\bar{a}$	1
$\bar{d}$	a	1
$\bar{d}$	$\bar{a}$	100

A	B	$\phi_1(A,B)$
a	b	30
a	$\bar{b}$	5
$\bar{a}$	b	1
$\bar{a}$	$\bar{b}$	10

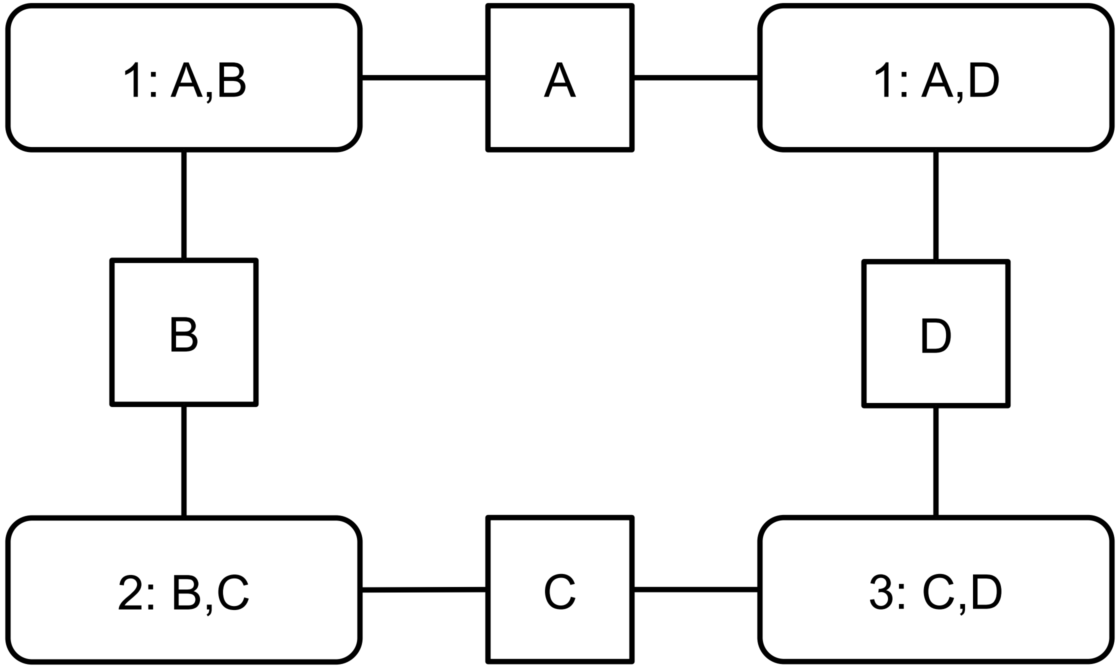


C	D	$\phi_3(C,D)$
c	d	1
c	$\bar{d}$	100
$\bar{c}$	d	100
$\bar{c}$	$\bar{d}$	1

B	C	$\phi_2(B,C)$
b	c	100
b	$\bar{c}$	1
$\bar{b}$	c	1
$\bar{b}$	$\bar{c}$	100

$\psi_1(A,B) = \phi_1(A,B)$

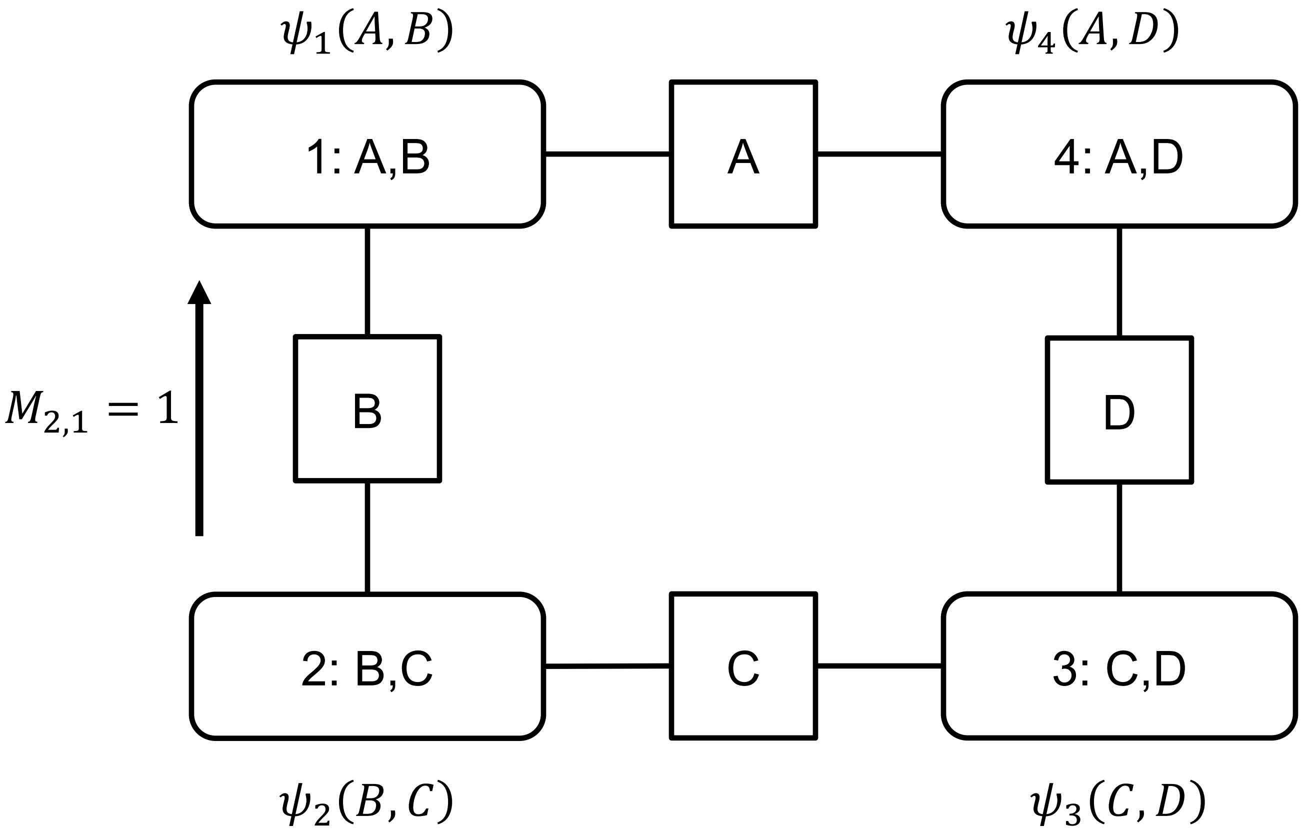
$\psi_4(A,D) = \phi_4(A,D)$



$\psi_2(B,C) = \phi_2(B,C)$

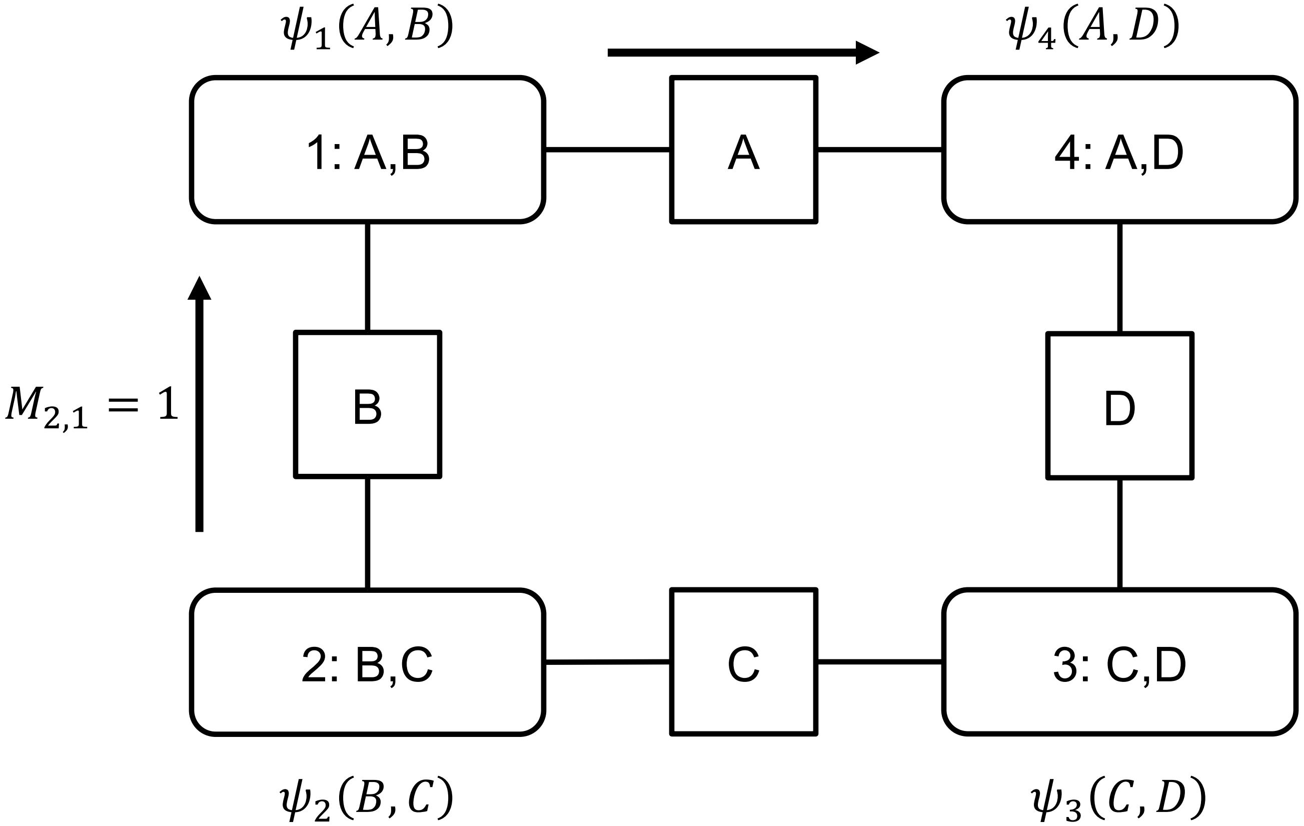
$\psi_3(C,D) = \phi_3(C,D)$

Joingraph Example with Markov Nets



Message Passing with Markov Nets

$M_{1,4} = \sum_B \psi_1(A,B)\, M_{2,1}(B)$



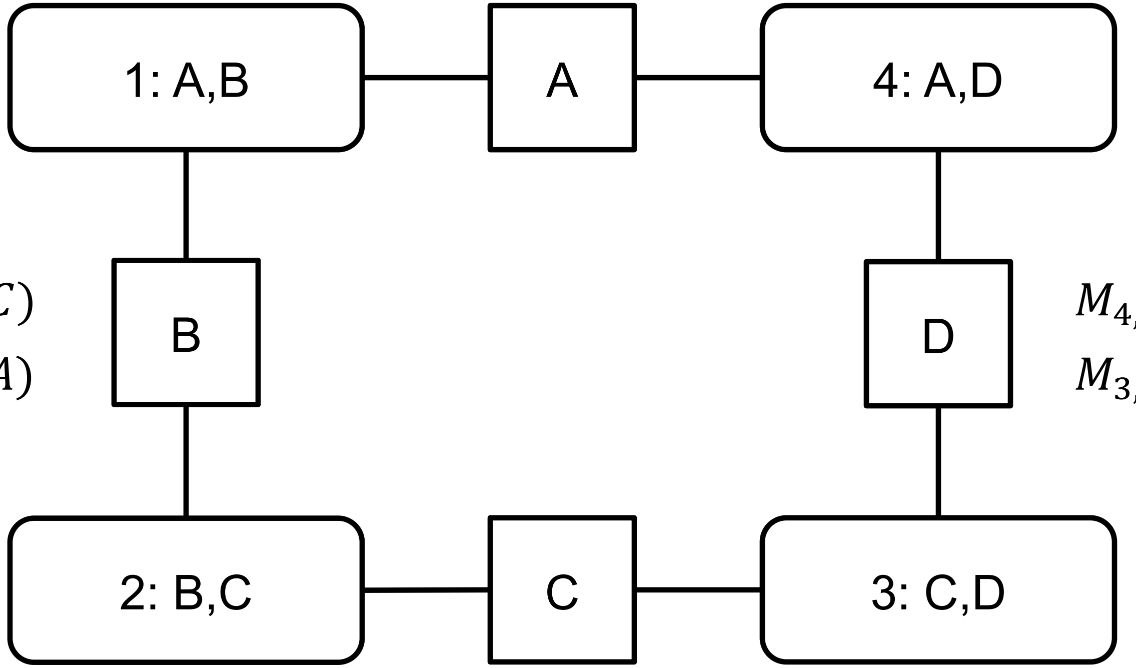
Message Passing with Markov Nets

$M_{1,4} = \sum_B \psi_1(A,B)\ M_{2,1}(B)$

$M_{4,1} = \sum_D \psi_4(A,D)\ M_{3,4}(D)$

$\psi_1(A,B)$

$\psi_4(A,D)$



$M_{2,1} = \sum_C \psi_2(B,C)\ M_{3,2}(C)$

$M_{1,2} = \sum_A \psi_1(A,B)\ M_{4,1}(A)$

$M_{4,3} = \sum_A \psi_4(A,D)\ M_{1,4}(A)$

$M_{3,4} = \sum_C \psi_3(C,D)\ M_{2,3}(C)$

$\psi_2(B,C)$

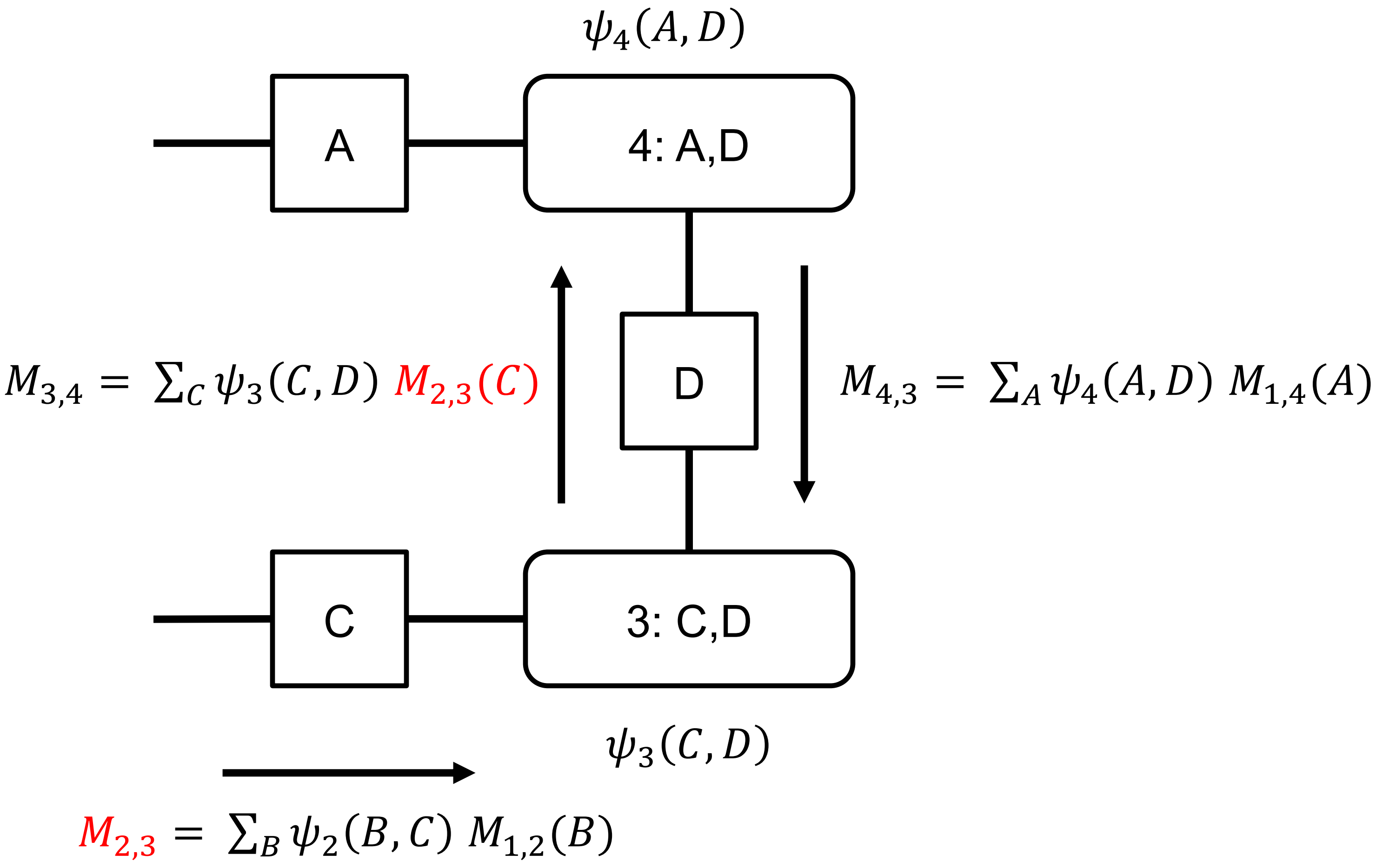
$\psi_3(C,D)$

$M_{2,3} = \sum_B \psi_2(B,C)\ M_{1,2}(B)$

$M_{3,2} = \sum_D \psi_3(C,D)\ M_{4,3}(D)$

Message Passing: Avoid Self-Beliefs

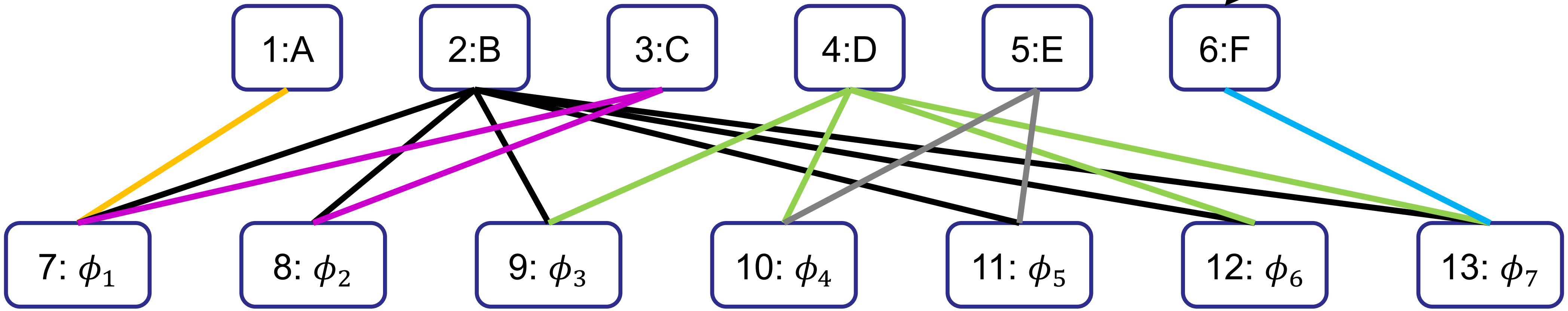
- Notice that
  - $M_{3,4}$ only considers information received from 2 ($M_{2,3}$)
  - Therefore, ignoring information from 4
  - This helps to avoid reinforcing self-beliefs



Bethe Graph

- A simple way to generate a valid joingraph
  - For each ϕ_k, make a cluster $\boldsymbol{C}_k = Vars(\phi_k)$
  - For each variable X_i, create a singleton cluster $\{X_i\}$
  - Edge $(\boldsymbol{C}_k, X_i)$ if $X_i \in \boldsymbol{C}_k$

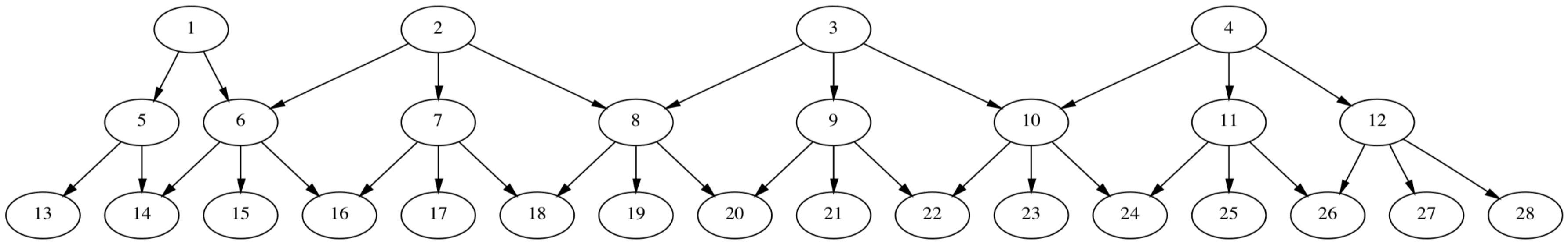
Which factor goes here? $\psi_i = 1$



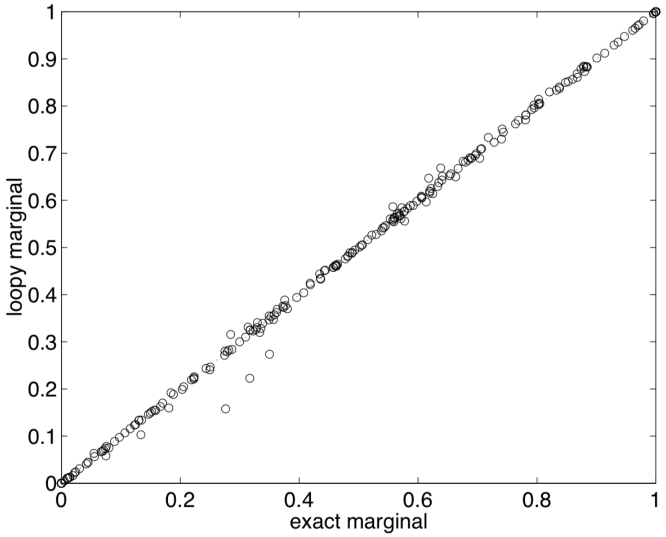
$$\phi_1(A,B,C), \phi_2(B,C), \phi_3(B,D), \phi_4(D,E), \phi_5(B,E), \phi_6(B,D), \phi_7(B,D,F)$$

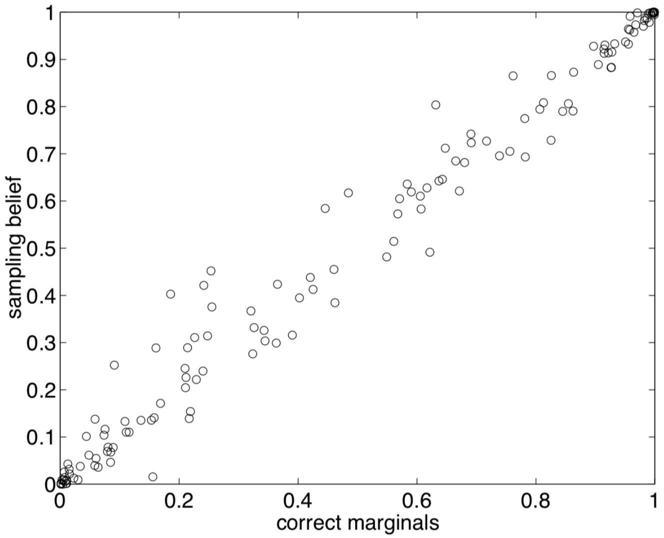
Belief Propagation Error

Pyramid network



Loopy belief propagation



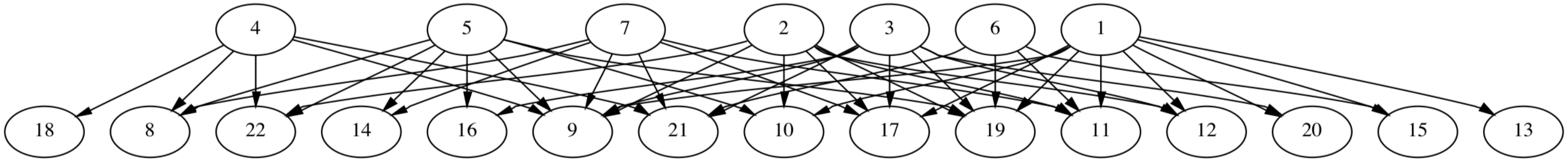


Likelihood sampling with 200 cases

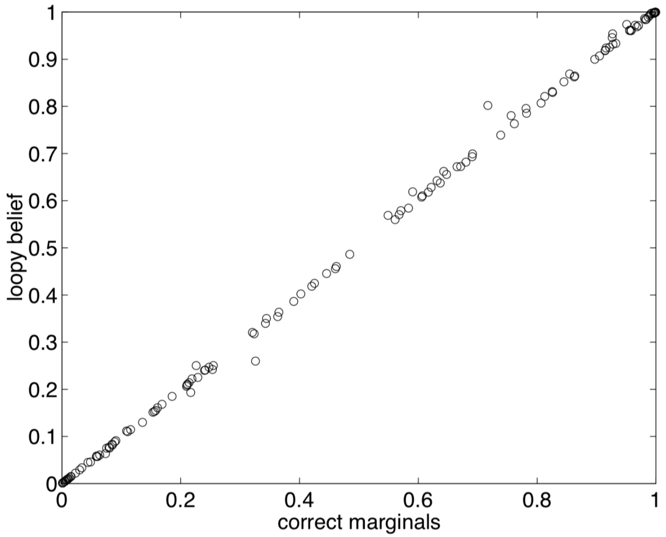
Murphy, K., Weiss, Y., & Jordan, M. I. (2013). Loopy belief propagation for approximate inference: An empirical study. *arXiv preprint arXiv:1301.6725*.

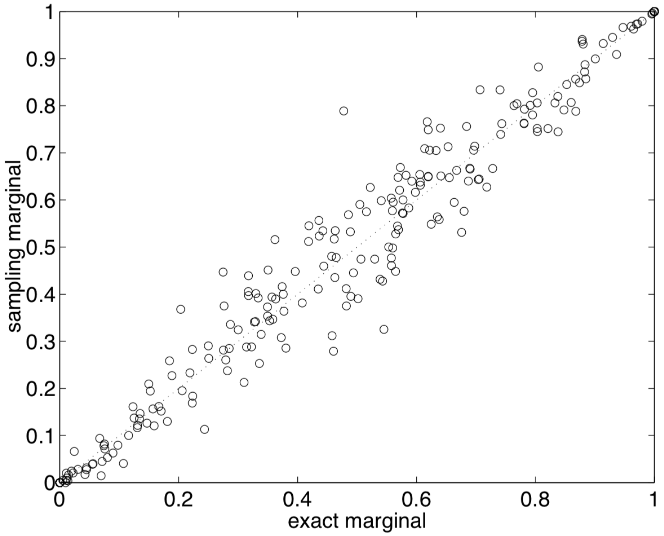
Belief Propagation Error

toyQMR network



Loopy belief propagation



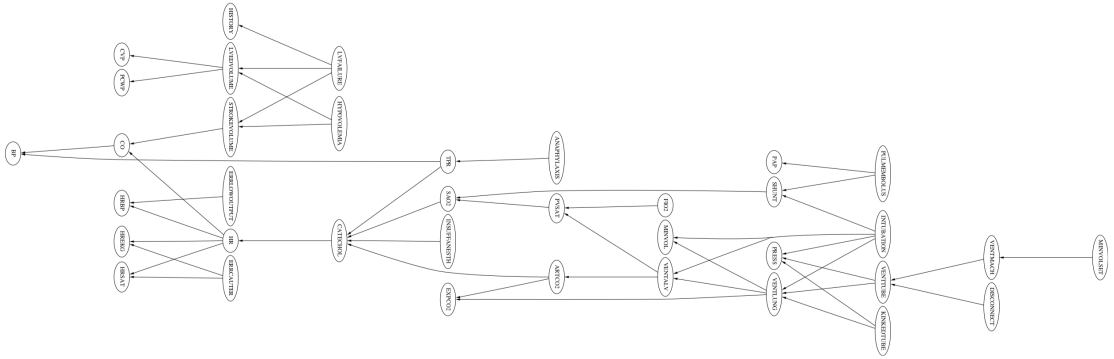


Likelihood sampling with 200 cases

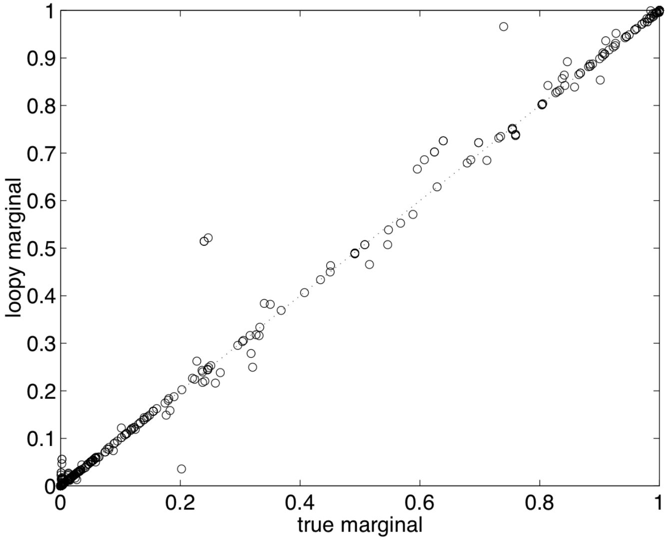
Murphy, K., Weiss, Y., & Jordan, M. I. (2013). Loopy belief propagation for approximate inference: An empirical study. *arXiv preprint arXiv:1301.6725*.

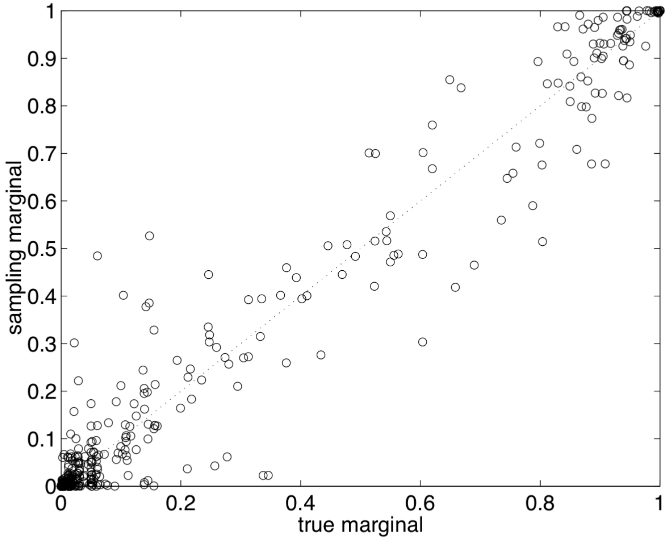
Belief Propagation Error

Full scale ICU network



Loopy belief propagation





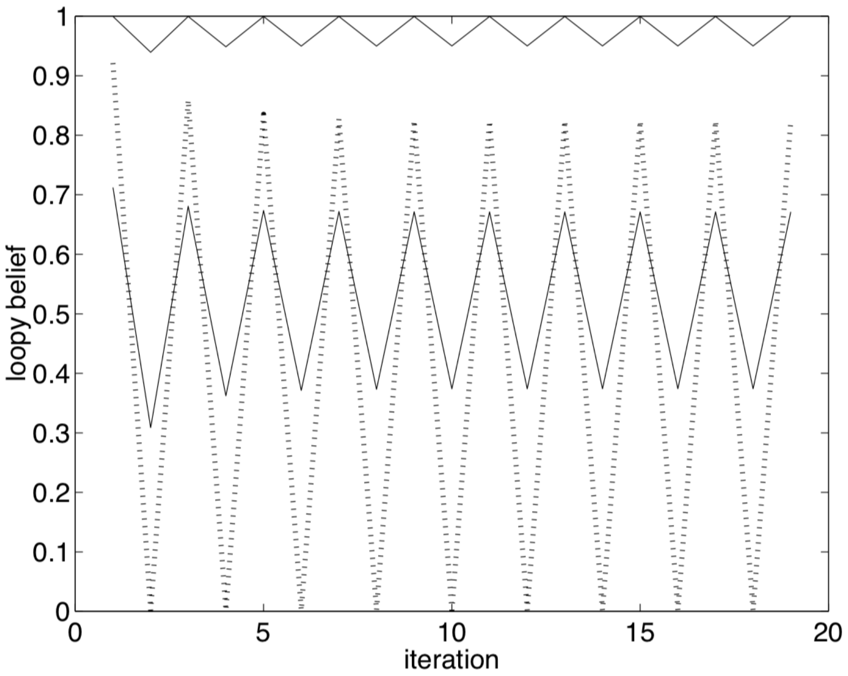
Likelihood sampling with 200 cases

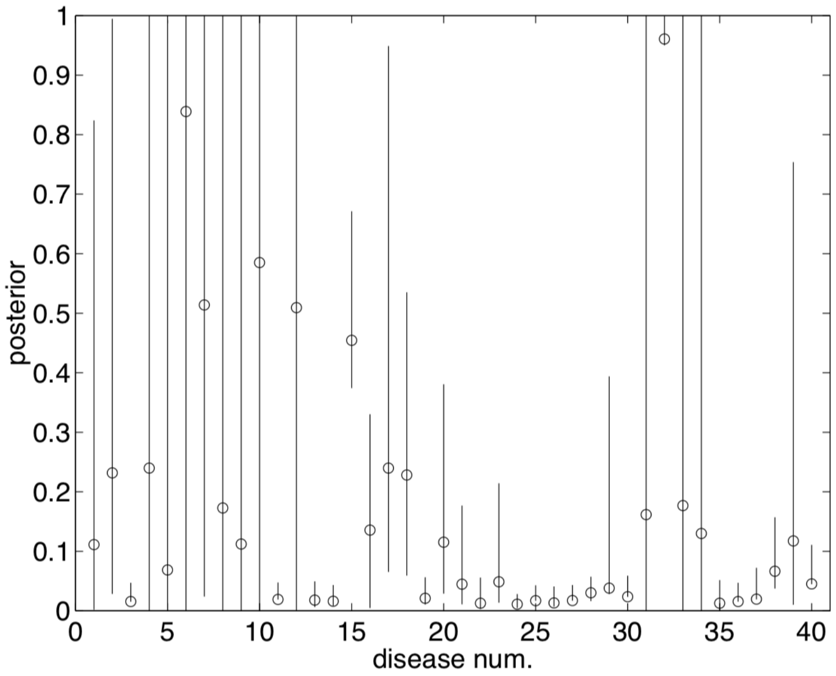
Murphy, K., Weiss, Y., & Jordan, M. I. (2013). Loopy belief propagation for approximate inference: An empirical study. *arXiv preprint arXiv:1301.6725*.

Belief Propagation Error

QMR-DT network

Posterior marginals for three nodes



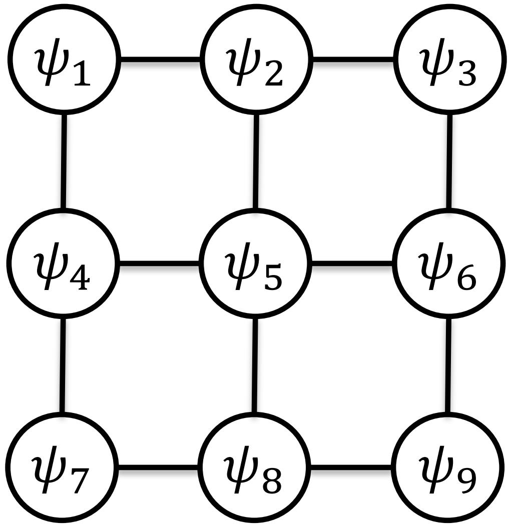


Exact marginals (circles) and error bars

Murphy, K., Weiss, Y., & Jordan, M. I. (2013). Loopy belief propagation for approximate inference: An empirical study. *arXiv preprint arXiv:1301.6725*.

Convergence and Message Schedule

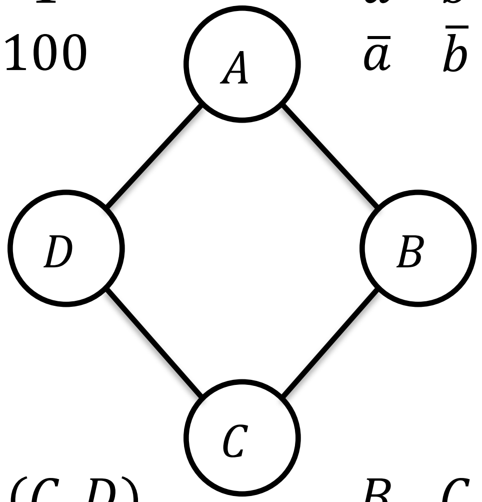
- In their analysis, Murphy, Weiss & Jordan used synchronous (parallel) message passaging
  - However, convergence can be improved with asynchronous approaches
- Some approaches for asynchronous message scheduling
  - Tree reparameterization (TRP): Choose a tree (spanning tree is a good choice) and pass messages. The trees must cover all edges
  - Residual belief propagation (RBP): Pass messages between two clusters whose beliefs over separators disagree the most. Usually organised with a priority queue
- Smoothing messages
  - $M_{ij} = \lambda \left(\eta \sum_{\boldsymbol{C}_i \setminus \boldsymbol{S}_{ij}} \psi_i \prod_{k \neq j} M_{ki} \right) + (1 - \lambda) M_{ij}^{old}$



Joingraph Example with Markov Nets

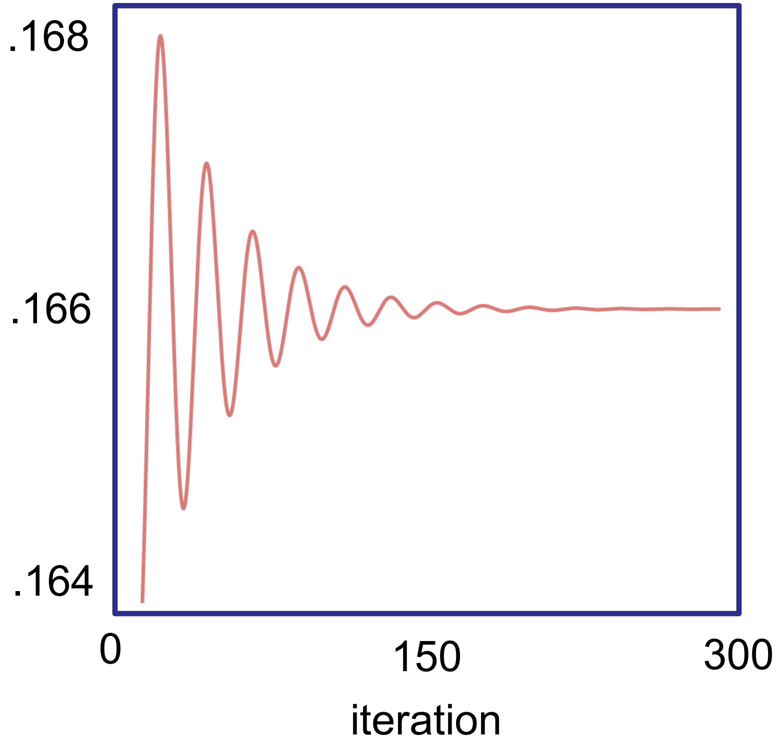
D	A	$\phi_4(D,A)$
d	a	100
d	$\bar{a}$	1
$\bar{d}$	a	1
$\bar{d}$	$\bar{a}$	100

A	B	$\phi_1(A,B)$
a	b	100
a	$\bar{b}$	2
$\bar{a}$	b	1
$\bar{a}$	$\bar{b}$	100

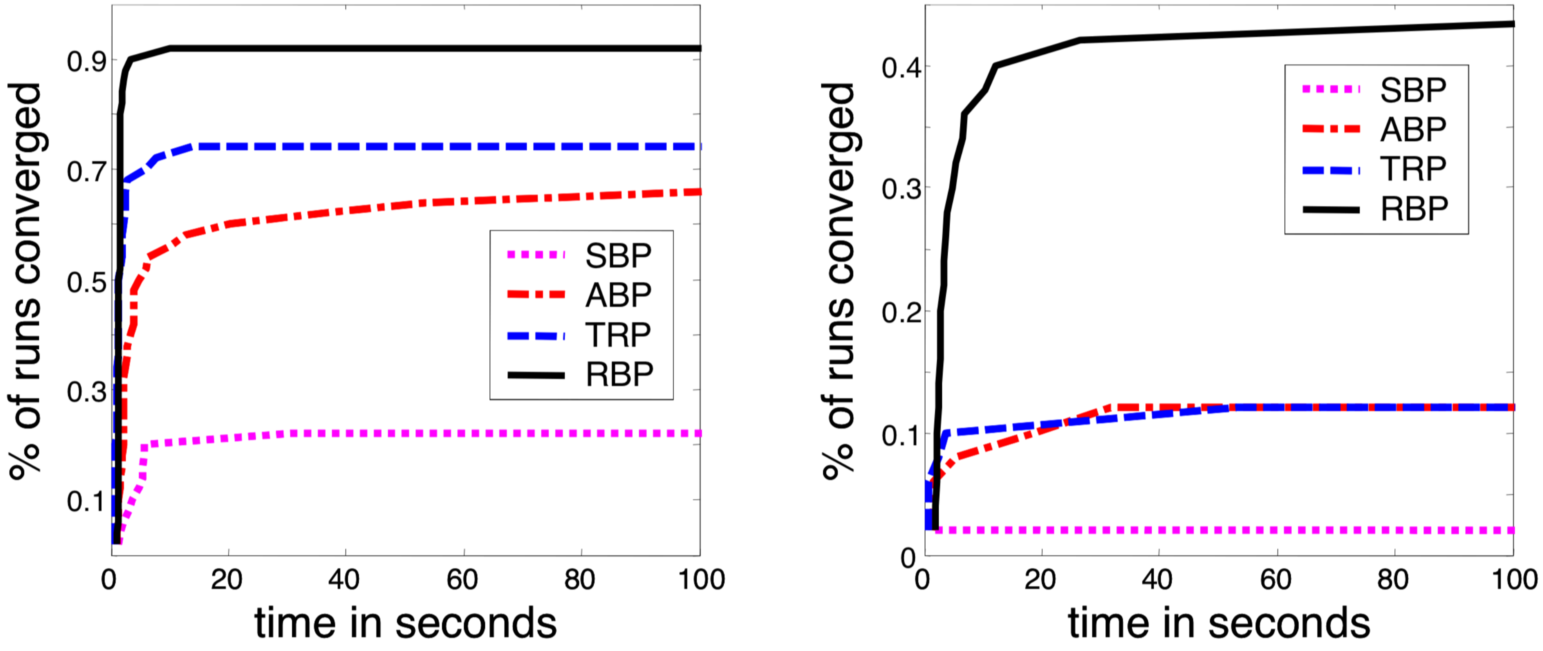


C	D	$\phi_3(C,D)$
c	d	1
c	$\bar{d}$	100
$\bar{c}$	d	100
$\bar{c}$	$\bar{d}$	1

B	C	$\phi_2(B,C)$
b	c	100
b	$\bar{c}$	1
$\bar{b}$	c	1
$\bar{b}$	$\bar{c}$	100



Convergence and Message Schedule



50 random grids of size 11 $\times$ 11 and C = 11 (left) and C = 13 (right)

Elidan, G., McGraw, I., & Koller, D. (2012). Residual belief propagation: Informed scheduling for asynchronous message passing. *arXiv preprint arXiv:1206.6837*.

Conclusion

- Belief propagation extends the paradigm of message passing
  - It provides a full spectrum of possibilities from exact to approximate inference
- Interactive joingraph propagation (IJGP) algorithm
  - Can be interpreted as an approach that minimizes the KL divergence between
    - The factorization induced by the network
    - The factorization induced by the joingraph
- IJGP messages convergence
  - Guaranteed in a single iteration if the joingraph is a tree (jointree)
  - Otherwise, convergence is not guaranteed
  - Even if the messages converge, its beliefs may not be necessarily equal the true marginals
  - Although very often in practice they will be close
- Task
  - Read chapter 14 (but 14.8)