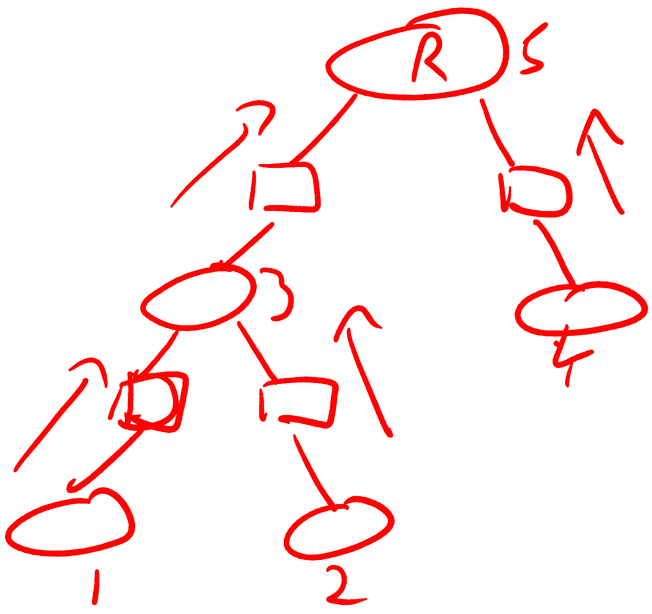
Stat 521A
Lecture 9

Outline

- Exact inference in clique trees (10.2, 10.3)
- Approximate inference – overview
- Loopy belief propagation (11.3)
- Other entropy approximations (11.3.7)

Message passing on a clique tree

- To compute $p(X_i)$, find a clique that contains X_i, make it the root, and send messages to it from all other nodes.
- A clique cannot send a node to its parent until it is ready, ie. Has received msgs from all its children.
- Hence we send from leaves to root.



Upwards pass (collect to root)

Procedure CTree Sum Product Up (
Φ, // Set of factors
$\mathcal{T}$, // Clique tree over Φ
α, // Initial assignment of factors to cliques
C_r // Some selected root clique
)

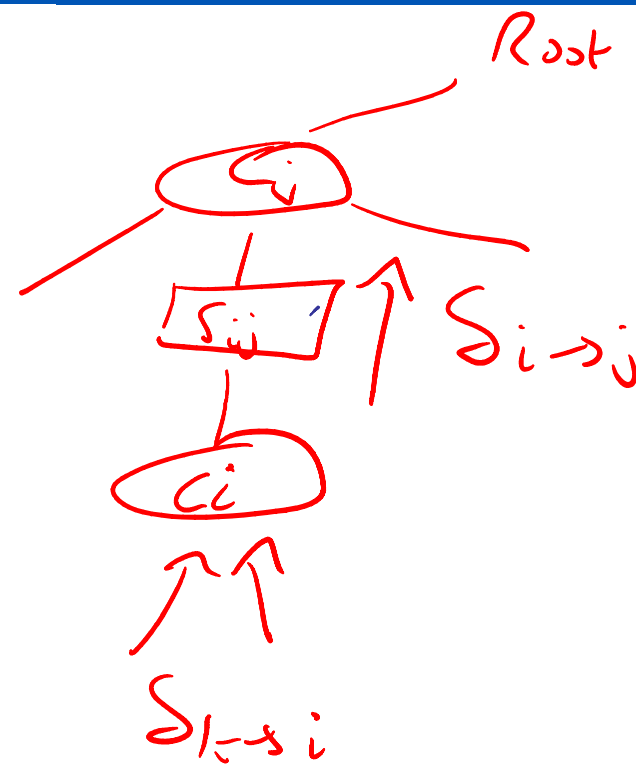
Initialize Cliques
while C_r is not ready
  Let C_i be a ready clique
  $\delta_{i \to p_r(i)}(S_{i,p_r(i)}) \leftarrow$ SP Message$(i, p_r(i))$
$\beta_r \leftarrow \psi_r \cdot \prod_{k \in \mathrm{Nb}_{C_r}} \delta_{k \to r}$
return β_r

Procedure Initialize Cliques (
)

for each clique C_i
  $\psi_i[C_i] \leftarrow \prod_{\phi_j \,:\, \alpha(\phi_j)=i} \phi$

Procedure SP Message (
i, // sending clique
j // receiving clique
)

$\psi(C_i) \leftarrow \psi_i \cdot \prod_{k \in (\mathrm{Nb}_i - \{j\})} \delta_{k \to i}$
$\tau(S_{i,j}) \leftarrow \sum_{C_i - S_{i,j}} \psi(C_i)$
return $\tau(S_{i,j})$

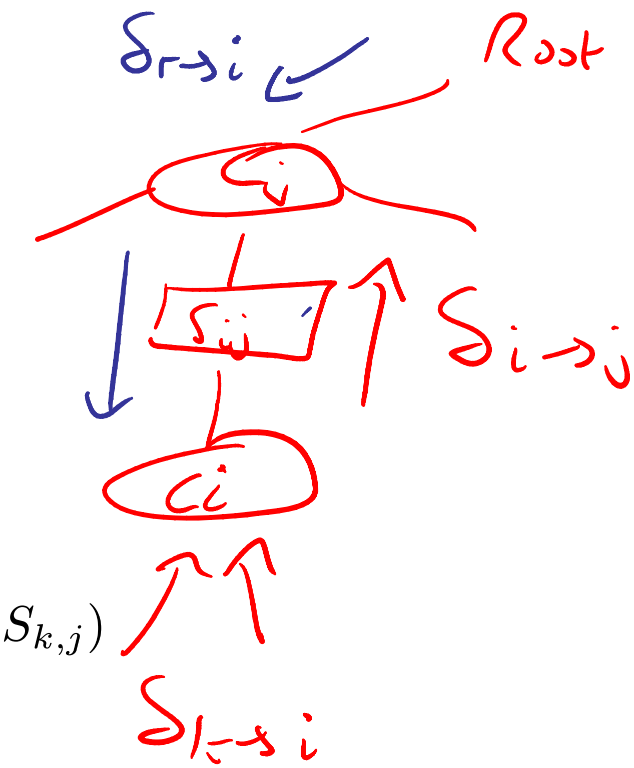


$$\beta_i(C_i) = \phi_i(C_i) \prod_{k \in n_i, k \neq j} \delta_{k \to i}(S_{k,i})$$

$$\delta_{i \to j}(S_{ij}) = \sum_{C_i \setminus S_{ij}} \beta_i(C_i)$$

Downwards pass (distribute from root)

- At the end of the upwards pass, the root has seen all the evidence.
- We send back down from root to leaves.



$$
\begin{aligned}
\beta_j(C_j) &= \phi_j(C_j) \prod_{k \in n_j} \delta_{k\to j}(S_{k,j}) \\
\delta_{j\to i}(S_{ij}) &= \sum_{C_j \backslash S_{ij}} \phi_j(C_j) \prod_{k \in n_j, i \neq k} \delta_{k\to j}(S_{k,j}) \quad \text{SP} \\
&= \sum_{C_j \backslash S_{ij}} \frac{\beta_j(C_j)}{\delta_{i\to j}(S_{ij})} \quad \text{BU}
\end{aligned}
$$

Use division operator to avoid double counting

Beliefs

- Thm 10.2.7. After collect/distribute, each clique potential represents a marginal probability (conditioned on the evidence)

$$\beta_i(C_i) = \sum_{\mathbf{x} \backslash C_i} \tilde{P}(\mathbf{x})$$

- If we get new evidence on X_i, we can multiply it in to any clique containing i, and then distribute messages outwards from that clique to restore consistency.

MAP configuration

- We can generalize the Viterbi algorithm from HMMs to find a MAP configuration of a general graph as follows.
- On the upwards pass, replace sum with max.
- At the root, find the most probable joint setting and send this as evidence to the root's children.
- Each child finds its most probable setting and sends this to its children.
- The jtree property ensures that when the state of a variable is fixed in one clique, that variable assumes the same state in all other cliques.

Samples

- We can generalize forwards-filtering backwards-sampling to draw exact samples from any GM as follows.
- Do a collect pass to the root as usual.
- Sample xR from the root marginal, and then enter it as evidence in all the children.
- Each child then samples itself from its updated local distribution and sends this to its children.

Calibrated clique tree

- Def 102.8. A clique tree is calibrated if, for all pairs of neighboring cliques, we have

$$\sum_{C_i \setminus S_{i,j}} \beta_i(C_i) = \sum_{C_j \setminus S_{i,j}} \beta_j(C_j) = \mu_{i,j}(S_{i,j})$$

- Eg. A-B-C clq tree AB – [B] – BC. We require

$$\sum_a \beta_{ab}(a,b) = \sum_c \beta_{bc}(b,c)$$

- Def 10.2.11. The measure defined by a calibrated tree is defined as

$$\beta_T(x) = \frac{\prod_i \beta_i(C_i)}{\prod_{<ij>} \mu_{i,j}(S_{ij})}$$

Calibrated clique tree

- Thm 10.2.12. For a calibrated clique tree, $p(x) \propto \beta_T(x)$ iff $\beta_i(C_i) \propto p(C_i)$

- Pf (sketch).

$$p(A,B,C) = \frac{p(A,B)p(B,C)}{p(C)} = p(A,B)p(C|B) = p(A|B)p(B,C)$$

Clique tree invariant

- Suppose at every step, clique i sends a msg to clique j, and stores it in $\mu_{i,j}$:

Procedure Send-BU-Msg (
  i, // sending clique
  j // receiving clique
)
  $\sigma_{i \to j} \leftarrow \sum_{C_i - S_{i,j}} \beta_i$
    // marginalize the clique over the sepset
  $\beta_j \leftarrow \beta_j \cdot \frac{\sigma_{i \to j}}{\mu_{i,j}}$
  $\mu_{i,j} \leftarrow \sigma_{i \to j}$

- Initially $\mu_{i,j}$=1 and $\beta_i = \prod_{f:\ f \text{ ass to } i} \phi_f$. Hence the following holds.

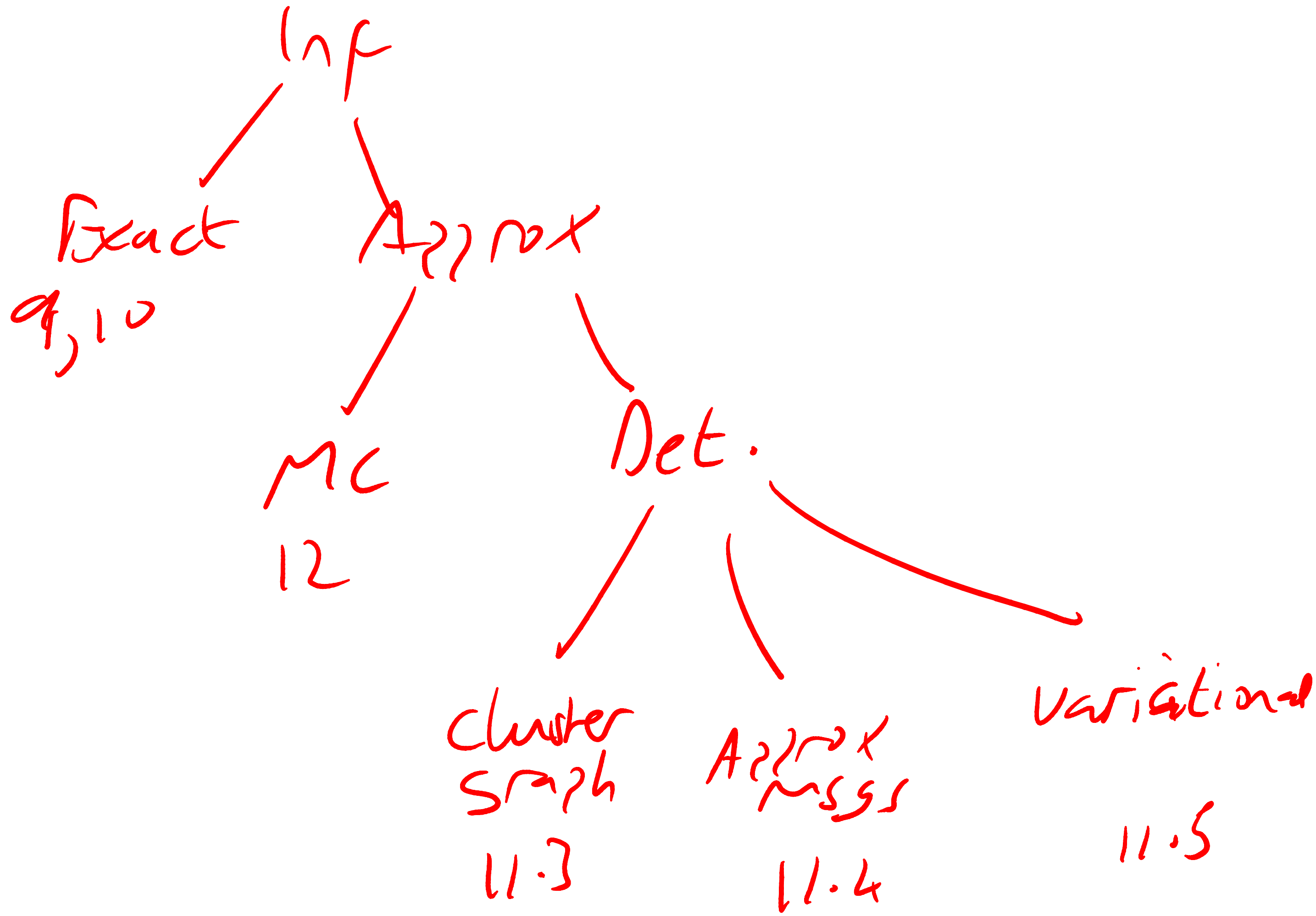
$$p(x) = \frac{\prod_i \beta_i(C_i)}{\prod_{<ij>} \mu_{i,j}(S_{ij})}$$

- Thm 10.3.4. This property holds after every belief updating operation. (But only when fully calibrated do clq pots = marginals.)

Summary of exact inference

- Build clique tree
  - eliminate nodes in some order
  - collect maximal cliques
  - Build a weighted graph where $W_{ij} = |C_i \text{ intersect } C_j|$
  - Find max weight spanning tree
- Initialize clique potentials with model potentials and evidence
- Do message passing on the tree

Approximate inference



Inference as optimization (11.1)

- Goal: find $\min_Q D(Q||P)$

- Thm 11.1.2

$$\begin{aligned}
\min_Q D(Q||P) &= D(Q||\frac{1}{Z}\tilde{P}) = \sum_x Q(x)\log Q(X) - Q(x)\ln \tilde{P}(x) + \ln Z \\
&= lnZ - F(\tilde{P}, Q) \\
F(\tilde{P}, Q) &= H_Q(x) + \sum_c E_{x_c \sim Q} \ln \phi(x_c)
\end{aligned}$$

  where F is the energy functional, and –F is the Helmholtz free energy

- Since D(Q||P) >=0, ln Z >= F(P,Q). We will maximize a lower bound on the log likelihood wrt Q.

Factored energy functional

- Consider a Q based on a cluster graph

$$Q = \{\beta_i : i \in \mathcal{V}\} \cup \{\mu_{i,j} : (i,j) \in \mathcal{E}\}$$

- Def 11.2.1. The factored energy functional is given by the following, where we approximate the entropy of Q

$$\tilde{F}(\tilde{P}, Q) = \sum_i E_{C_i \sim \beta_i} \ln \psi_i + \sum_i H_{\beta_i}(C_i) - \sum_{<ij>} H_{\mu_{i,j}}(S_{i,j})$$

- Thm 11.2.2. If Q is a set of calibrated beliefs for a tree, and Q has the form $Q(x) = \frac{\prod_i \beta_i(C_i)}{\prod_{<ij>} \mu_{i,j}(S_{ij})}$ then

$$\tilde{F}(\tilde{P}, Q) = F(\tilde{P}, Q)$$

Exact inference as optimization

- Define the local consistency polytope as (p381) the set of distributions

$$Q = \{\beta_i : i \in \mathcal{V}\} \cup \{\mu_{i,j} : (i,j) \in \mathcal{E}\}$$

which satisfy

$$\begin{aligned} \mu_{i,j}(S_{i,j}) &= \sum_{C_i \backslash S_{i,j}} \beta_i(C_i) \\ \sum_{c_i} \beta_i(c_i) &= 1 \\ \beta_i(c_i) &\geq 0 \end{aligned}$$

- Thm 11.1.1 If T is an I-map of P, and Q is a calibrated clique tree, then

$$max_{Q \in \text{Local}} \tilde{F}(\tilde{P}, Q)$$

has a unique global optimum, in which Q=P

Constrained optimization

CTree-Optimize

Find $\quad Q = \{\beta_i : i \in \mathcal{V}_T\} \cup \{\mu_{i,j} : (i\ j) \in \mathcal{E}_T\}$

that maximize $\quad \tilde{F}[\tilde{P}_\Phi, Q]$

subject to

$$\begin{array}{rcll} \mu_{i,j}[s_{i,j}] & = & \sum_{\boldsymbol{C}_i - \boldsymbol{S}_{i,j}} \beta_i[\boldsymbol{c}_i] & \\ & & \forall (i\ j) \in \mathcal{E}_T, \forall s_{i,j} \in Val(\boldsymbol{S}_{i,j}) & \\ \sum_{\boldsymbol{c}_i} \beta_i[\boldsymbol{c}_i] & = & 1 & \forall i \in \mathcal{V}_T \\ \beta_i[\boldsymbol{c}_i] & \geq & 0 & \forall i \in \mathcal{V}_T, \boldsymbol{c}_i \in Val(\boldsymbol{C}_i) \end{array}$$

$$\begin{array}{rcl} \mathcal{J} & = & \tilde{F}[\tilde{P}_\Phi, Q] \\ & & - \sum_{i \in \mathcal{V}_T} \lambda_i \left(\sum_{\boldsymbol{c}_i} \beta_i[c_i] - 1 \right) \\ & & - \sum_i \sum_{j \in \mathrm{Nb}_i} \sum_{\boldsymbol{s}_{i,j}} \lambda_{j \to i}[s_{i,j}] \left(\sum_{\boldsymbol{c}_i \sim \boldsymbol{s}_{i,j}} \beta_i[\boldsymbol{c}_i] - \mu_{i,j}[s_{i,j}] \right), \end{array}$$

Msgs = Lagrange multipliers

$$
\begin{aligned}
\mathcal{J} &= \tilde{F}[\tilde{P}_\Phi, Q] \\
&\quad - \sum_{i \in \mathcal{V}_T} \lambda_i \left(\sum_{c_i} \beta_i[c_i] - 1 \right) \\
&\quad - \sum_i \sum_{j \in \mathrm{Nb}_i} \sum_{s_{i,j}} \lambda_{j \to i}[s_{i,j}] \left(\sum_{c_i \sim s_{i,j}} \beta_i[c_i] - \mu_{i,j}[s_{i,j}] \right),
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial}{\partial \beta_i[c_i]} \mathcal{J} &= \ln \psi_i[c_i] - \ln \beta_i[c_i] - 1 - \lambda_i - \sum_{j \in \mathrm{Nb}_i} \lambda_{j \to i}[s_{i,j}] \\
\frac{\partial}{\partial \mu_{i,j}[s_{i,j}]} \mathcal{J} &= \ln \mu_{i,j}[s_{i,j}] + 1 + \lambda_{i \to j}[s_{i,j}] + \lambda_{j \to i}[s_{i,j}].
\end{aligned}
$$

$$
\begin{aligned}
\beta_i[c_i] &= \exp\{-1 - \lambda_i\} \psi_i[c_i] \prod_{j \in \mathrm{Nb}_i} \exp\{-\lambda_{j \to i}[s_{i,j}]\} \\
\mu_{i,j}[s_{i,j}] &= \exp\{-1\} \exp\{-\lambda_{i \to j}[s_{i,j}]\} \exp\{-\lambda_{j \to i}[s_{i,j}]\}.
\end{aligned}
$$

$$
\delta_{i \to j}[s_{i,j}] \triangleq \exp\left\{ -\lambda_{i \to j}[s_{i,j}] - \frac{1}{2} \right\}.
$$

Msgs = Lagrange multipliers

- Thm 11.2.3. A set of beliefs Q is a stationary point of CTreeOptimize iff there exist a set of messages such that

$$\delta_{i \to j} \propto \sum_{\boldsymbol{C}_i - \boldsymbol{S}_{i,j}} \psi_i \left(\prod_{k \in \mathrm{Nb}_i - \{j\}} \delta_{k \to i} \right)$$

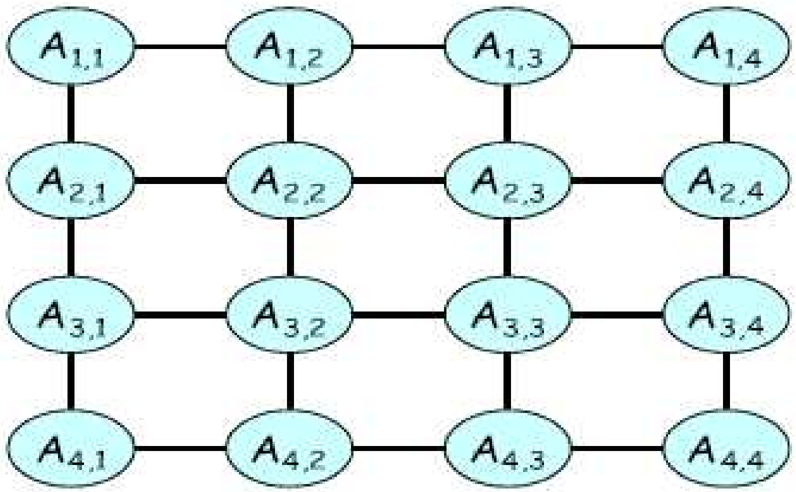
and moreover, we have that

$$\begin{aligned} \beta_i &\propto \psi_i \left(\prod_{j \in \mathrm{Nb}_i} \delta_{j \to i} \right) \\ \mu_{i,j} &= \delta_{j \to i} \cdot \delta_{i \to j}. \end{aligned}$$

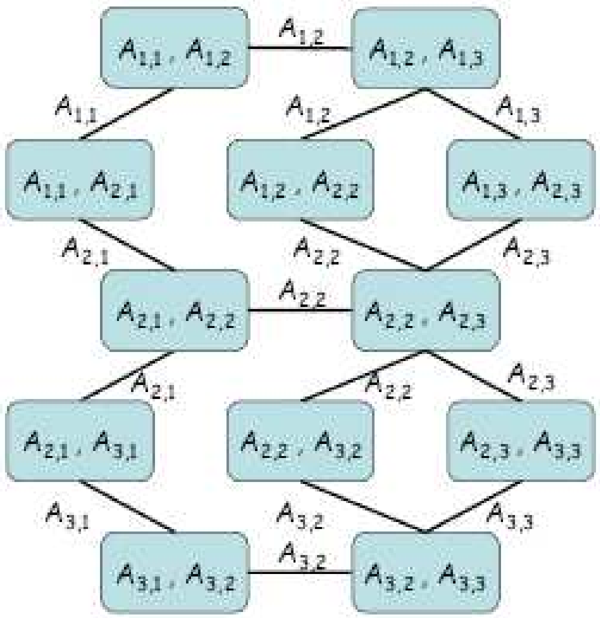
Cluster graphs

- If the cluster graph is a cluster tree with RIP, then the factored energy is equal to the energy, and enforcing local consistency is equivalent to enforcing global consistency.
- However, the cliques may be too big.
- Let us consider general CGs which only have to satisfy the RIP constraint.
- Hence all edges associated with some node X form a tree and all clusters agree on the marginal for each X. However, they may not agree on higher order marginals.

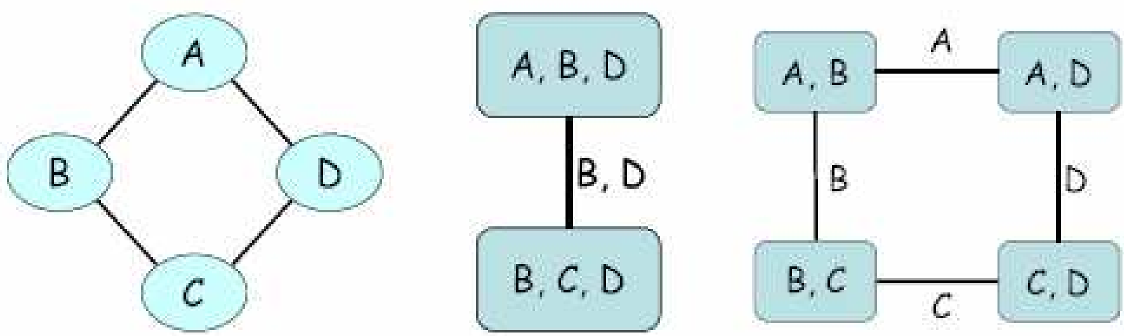
Examples



$A_{1,1}$ $A_{1,2}$ $A_{1,3}$ $A_{1,4}$
$A_{2,1}$ $A_{2,2}$ $A_{2,3}$ $A_{2,4}$
$A_{3,1}$ $A_{3,2}$ $A_{3,3}$ $A_{3,4}$
$A_{4,1}$ $A_{4,2}$ $A_{4,3}$ $A_{4,4}$



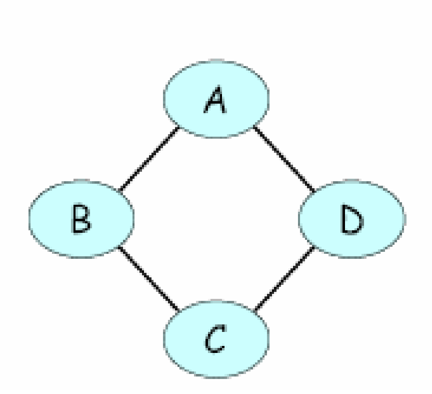
$A_{1,1}, A_{1,2}$
$A_{1,2}$
$A_{1,2}, A_{1,3}$
$A_{1,1}$
$A_{1,2}$
$A_{1,3}$
$A_{1,1}, A_{2,1}$
$A_{1,2}, A_{2,2}$
$A_{1,3}, A_{2,3}$
$A_{2,1}$
$A_{2,2}$
$A_{2,3}$
$A_{2,1}, A_{2,2}$
$A_{2,2}$
$A_{2,2}, A_{2,3}$
$A_{2,1}$
$A_{2,2}$
$A_{2,3}$
$A_{2,1}, A_{3,1}$
$A_{2,2}, A_{3,2}$
$A_{2,3}, A_{3,3}$
$A_{3,1}$
$A_{3,2}$
$A_{3,3}$
$A_{3,1}, A_{3,2}$
$A_{3,2}$
$A_{3,2}, A_{3,3}$

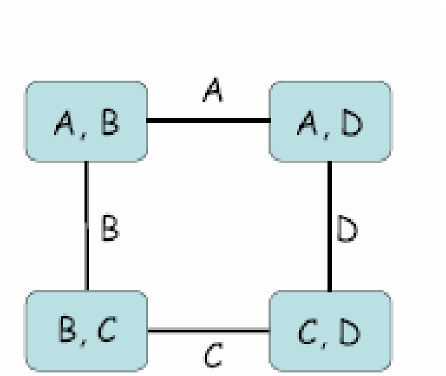


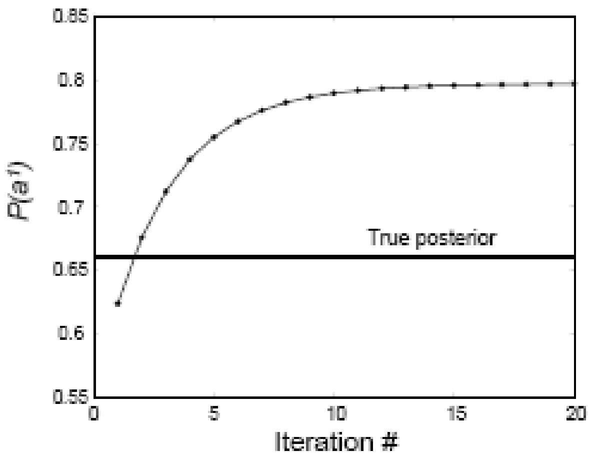
A
B
D
C
A, B, D
B, D
B, C, D
A, B
A
A, D
B
D
B, C
C
C, D

Belief prop on a cluster graph

- We can run the BP algorithm on a CG even if it is not a tree. This is called loopy BP.
- This can fail to converge and give the wrong answers due to double counting of evidence.



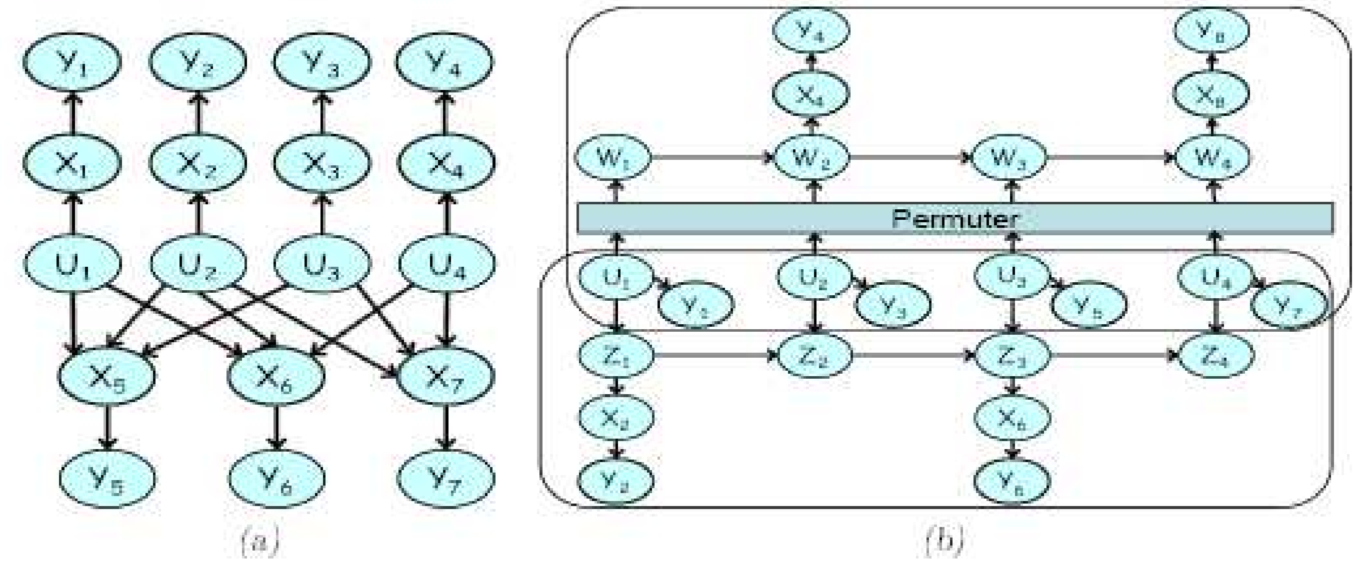




Turbocodes

- Channel coding is a way of encoding msgs that makes them resistant to noise, and hence easier to decode.
- Let us send a k-bit msg u(1:k) using n bits, x(1:n) eg x = 3 copies of u. We receive y(1:n) and estimate u. The rate of the code is k/n.
- Shannon's thm characterizes the best rate one can achieve for a given error rate and noise level.
- Turbodecoding is a method to approximately estimate u from y which achieves near-optimal rate. It is equivalent to loopy BP in a particular DGM.

Turbocodes



K=4,n=7 parity check

K=4,n=8 turbocode

Convergence (11.3.4)

- For discrete networks, one can show that LBP will converge if the connections are not too deterministic.
- Eg for Ising model, sufficient condition is

$$\max_i \max_{j \in \mathrm{Nb}_i} \sum_{k \in \mathrm{Nb}_i - \{j\}} \tanh |\epsilon_{k,i}| < 1.$$

- Similar conditions exist for Gaussian networks.
- Special case analysis has been derived for turbocodes.

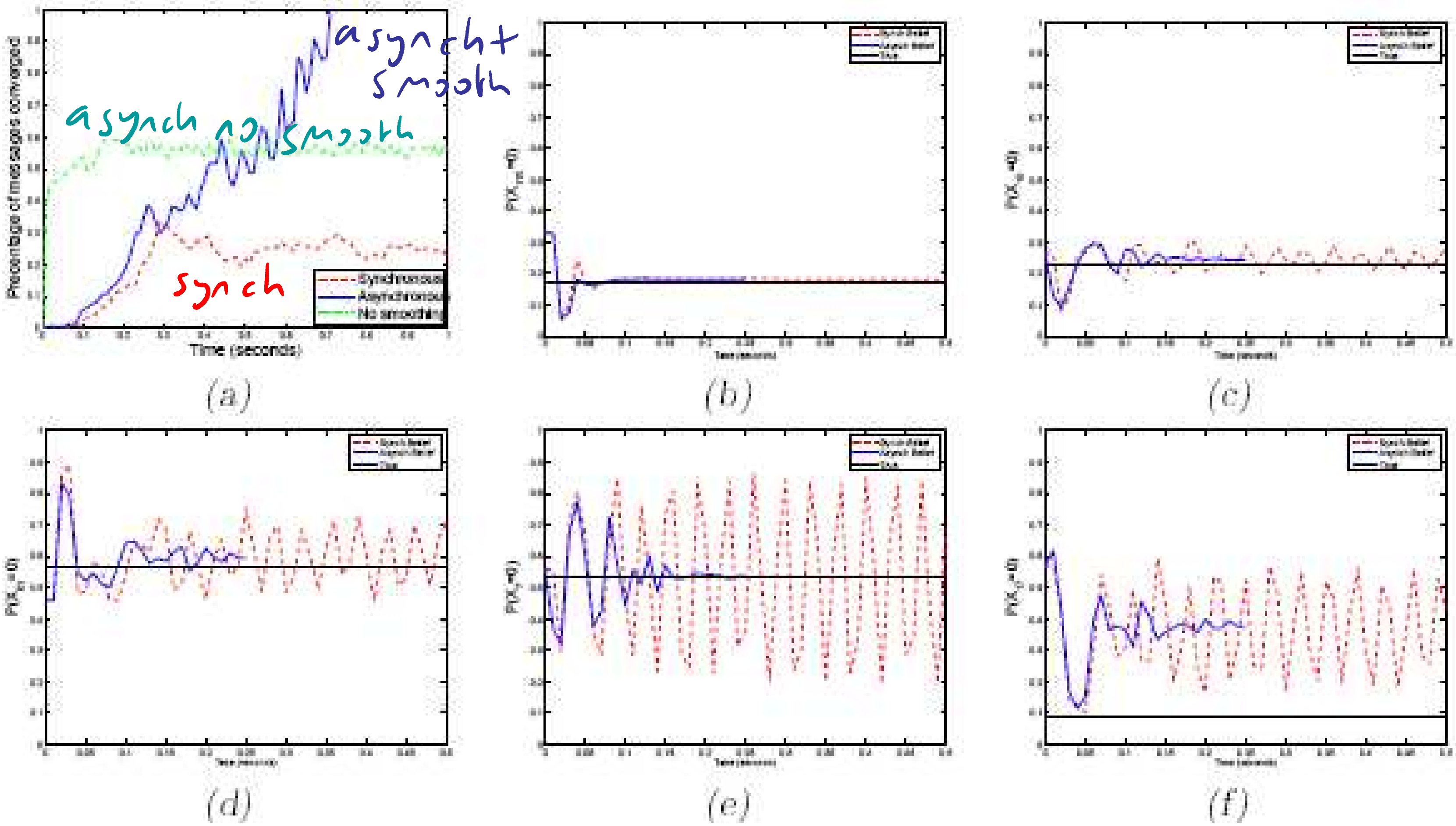
Encouraging convergence

- One can use damped updates

$$\delta_{i\to j} \leftarrow \sum_{C_i - S_{i,j}} \prod_{k\neq j} \delta_{k\to i}. \qquad \delta_{i\to j} \leftarrow \lambda \left(\sum_{C_i - S_{i,j}} \prod_{k\neq j} \delta_{k\to i} \right) + (1-\lambda)\delta^{\text{old}}_{i\to j},$$

- Asychronous updates work better than sychronous.
- Tree reparameterization (TRP) selects a set of trees, each of which spans a large number of clusters, and whose union covers all the edges. It then selects a tree at rnd and calibrates it, treating all other messages as local evidence.
- Priority-queue based msg scheduling also works very well.

Example: 11x11 Ising

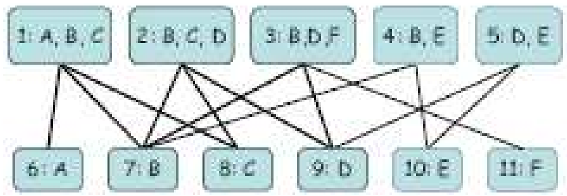


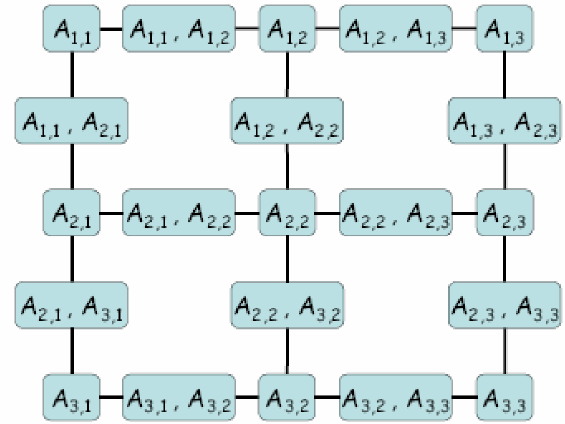
Accuracy

- In general, it is hard to characterize the accuracy of approximate solutions. Often the most probable state is locally correct but is over confident.
- For Gaussian networks, Weiss et al showed that, if the method converges, the means are exact, but the variances are too small.

Bethe cluster graphs

- Suppose we create one cluster for each original factor, and one cluster for each node.

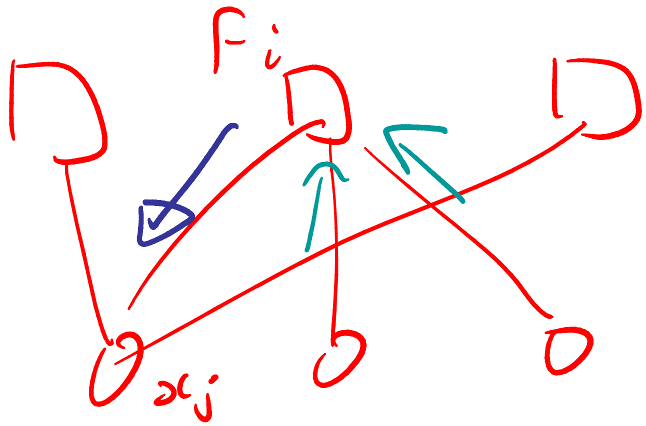




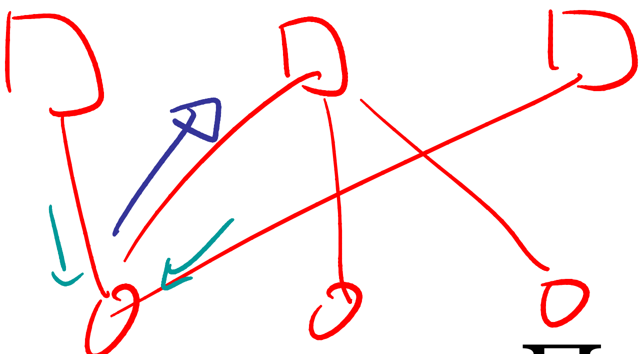
- Then for a pairwise MRF, propagating $C_i - C_{ij} - C_j$ is equivalent to sending msgs from node i to node j via edge ij.
- In general, BP on the Bethe CG = BP on the factor graph.

BP on factor graphs

Bishop p406



$$\mu_{f_i \to x_j}(x_j) \quad = \quad \sum_{c_i \backslash x_j} f(c_i) \prod_{k \in nb(f_i) \backslash x_j} \mu_{x_k \to f_i}(x_k)$$



$$\mu_{x_i \to f_j}(x_i) \quad = \quad \prod_{k \in nb(x_i) \backslash f_j} \mu_{f_k \to x_i}(x_i)$$

Bethe approximation to entropy

- Thm 11.3.10. If Q is a calibrated set of beliefs for a Bethe approximation CG then the factored energy is given by

$$
\begin{aligned}
\tilde{F}(\tilde{P}, Q) &\stackrel{\text{def}}{=} \sum_{\phi} E_{\beta_\phi} \ln \phi + \sum_{\phi} H_{\beta_\phi}(C_\phi) - \sum_{s} H_{\mu_s}(S_s) \\
&= \sum_{\phi} E_{\beta_\phi} \ln \phi + \sum_{\phi} H_{\beta_\phi}(C_\phi) - \sum_{i} (d_i - 1) H_{\beta_i}(X_i)
\end{aligned}
$$

  where d_i = #factors that contain Xi.

- If Xi appears in di factors, by RIP, it appears in (di-1) sepsets. Hence we count the entropy of each Xi once in total.

Weighted approximation to entropy

- Consider a cluster graph, each of whose clusters (regions) has a counting number μ_r. Define the weighted approximate entropy as

$$H_Q^{\mu}(X) = \sum_r \mu_r H_{\beta_r}(C_r)$$

- For a Bethe-structured CG, we set

$$\mu_i = 1 - \sum_{r \in nb_i} \mu_r$$

- If we set μ_r=1, we recover the Bethe approximation.
- Let us consider more general weightings.

Convex approximation to entropy

- Def 11.3.13. We say that μ_r are convex counting numbers if there exist non-negative numbers ν_r, ν_i, $\nu_{r,i}$ st

$$\begin{aligned} \mu_r &= \nu_r + \textstyle\sum_{i \,:\, X_i \in C_r} \nu_{r,i} \quad \textit{for all } r \\ \mu_i &= \nu_i - \textstyle\sum_{r \,:\, X_i \in C_r} \nu_{r,i} \quad \textit{for all } i \end{aligned}$$

- Then

$$\sum_r \mu_r H_{\beta_r}(C_r) + \sum_i \mu_i H_{\beta_i}(X_i) = \sum_r \nu_r H_{\beta_r}(C_r) + \sum_{r, X_i \in C_r} \nu_{r,i}(H_{\beta_r}(C_r) - H_{\beta_i}(X_i)) \sum_i \nu_i H_{\beta_i}(X_i$$

- Thm 11.3.14. The above eqn is concave for any set of beliefs Q which satisfy marginal consistency constraints.

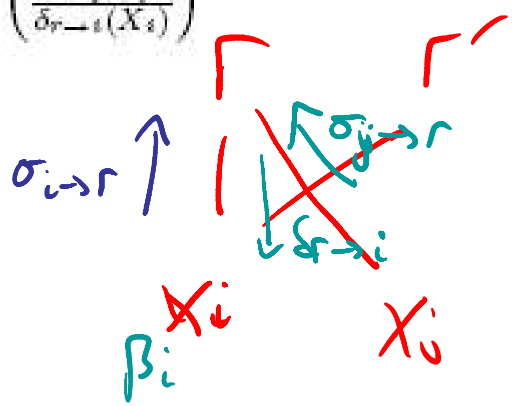
Convex BP

Algorithm 11.2 Convergent message passing algorithm for Bethe-structured region graphs with convex counting numbers

Procedure Convex-BP-Msg (
  $\psi_r[C_r]$ // set of initial potentials
  $\sigma_{i\to r}(C_r)$ // Current node-to-region messages
)

for $i = 1, \ldots, n$
  // Compute incoming messages from neighboring regions to X_i
  for $r \in \mathrm{Nb}_i$
    $\delta_{r\to i}(X_i) \leftarrow \sum_{C_r - X_i}\left(\psi_r[C_r]\prod_{j\in \mathrm{Nb}_r-\{i\}}\sigma_{j\to r}(C_r)\right)^{\frac{1}{\hat{\nu}_{i,r}}}$
  // Compute beliefs for X_i, renormalizing to avoid numerical underflows
  $\beta_i[X_i] \leftarrow \propto \prod_{r\in \mathrm{Nb}_i}(\delta_{r\to i}(X_i))^{\hat{\nu}_{i,r}/\hat{\nu}_i}$
  // Compute outgoing messages from X_i to neighboring regions
  for $r \in \mathrm{Nb}_i$
    $\sigma_{i\to r}(C_r) \leftarrow \left(\psi_r[C_r]\prod_{j\in \mathrm{Nb}_r-\{i\}}\sigma_{j\to r}(C_r)\right)^{-\frac{\nu_{i,r}}{\hat{\nu}_{i,r}}}\left(\frac{\beta_i[X_i]}{\delta_{r\to i}(X_i)}\right)^{\nu_r}$
return $\{\sigma_{i\to r}(C_r)\}_{i,r\in \mathrm{Nb}_i}$

$$\hat{\nu}_i = \nu_i + \sum_{r\in \mathrm{Nb}_i}\nu_r; \qquad \hat{\nu}_{i,r} = \nu_r + \nu_{i,r}.$$



TRW

- Tree reweighting algorithm (TRW) uses the following convex counting numbers, given a distribution over trees T st each edge in the pairwise network is present in at least 1 tree

$$
\begin{aligned}
\mu_i &= -\sum_{T \ni X_i} \rho(T) \\
\mu_{i,j} &= \sum_{T \ni (X_i, X_j)} \rho(T)
\end{aligned}
$$

Convex or not?

- When standard BP converges, the Bethe approximation to the entropy is often more accurate than the convex approximation.
- However, it is desirable to have a convex inference engine in the inner loop of learning.
- If you train with a convex approximation, there are some arguments you should use the same convex approx at test time for decoding.

Regions graphs (11.3.7.3)

- One can use more general CGs than the Bethe construction, which lets you model higher order interactions which are intermediate between the original factors and singletons.
- Resulting algorithm is complex.