# Lecture 12: February 25, 2015
# cs 573: Probabilistic Reasoning
# Professor Nevatia
# Spring 2015

# Admin

- Assignment # 4 due March 9
- **Office hours Th, Feb 26: 1:30-3:00 P.M.**
- Exam 1
  - March 2, class period, here (except for remote students)
  - Closed book, Closed Notes
    - Will not need to memorize long formulas
    - Calculators will not be needed
  - Style: similar to assignments + qualitative ("theory") questions
  - Content: defined by what is covered in class, including parts of this week's classes
  - Representations: Chapters 2-4 except for sections 3.4.3, 4.4.2, 4.6.2; chapter 5, focus on 5.4 (exclude 5.4.4)
  - Inference: Chapter 9, excluding 9.6; Chapter 10; Chapter 11: 11.3 (except 11.3.4 and 11.3.7)

# Review

- Last lecture:
  - Belief Propagation Algorithm
    - Note: algorithm as presented in the book does not give a clear termination criterion
      - Easy to define some (new message passed only if a clique receives input that is not "1", except the first time).
    - Introducing evidence
    - Answering queries (not covered last time)
  - Intro to need for approximate inference
- Today's objective
  - Loopy Belief Propagation (LBP)
  - Intro to Gaussian Networks

# Convergence of Belief Propagation

- Algorithm 10.3 not completely specified. When is a clique "uninformed"?
  - Clique is "informed", when it has received messages from all neighbors which should also be informed
  - Note: if a clique gets no new info, it will transmit "1" which can be omitted (after initialization)
- In parallel implementation, each clique sends messages to neighbors on each iteration
  - Maximum number is length of longest path in the tree
- In serial implementation, start from leaf nodes, go to root and downward pass (similar to sum-product)
- Sum-product-divide algorithm does not need to store original clique potentials, only the sepset potentials ($\mu_{ij}$) which will have fewer variables then the cliques in them; hence savings in space.

# Queries outside a Clique

- Variables may be in more than one clique.
- Find a sub-tree that contains all the variables
- Perform VE on it

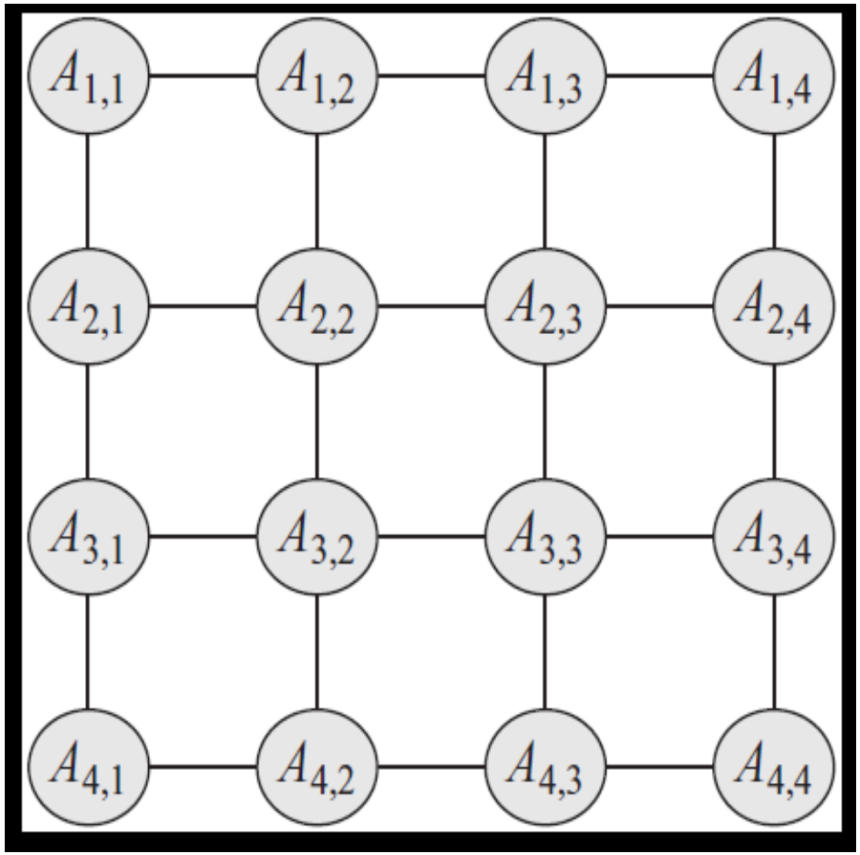
**Algorithm 10.4 Out-of-clique inference in clique tree**

**Procedure** CTree-Query (
  $\mathcal{T}$, // Clique tree over $\Phi$
  $\{\beta_i\}, \{\mu_{i,j}\}$, // Calibrated clique and sepset beliefs for $\mathcal{T}$
  $\mathbf{Y}$ // A query
)

1 Let $\mathcal{T}'$ be a subtree of $\mathcal{T}$ such that $\mathbf{Y} \subseteq Scope[\mathcal{T}']$
2 Select a clique $r \in \mathcal{V}_{\mathcal{T}'}$ to be the root
3 $\Phi \leftarrow \beta_r$
4 **for** each $i \in \mathcal{V}'_{\mathcal{T}}$
5 $\phi \leftarrow \frac{\beta_i}{\mu_{i,p_r(i)}}$
6 $\Phi \leftarrow \Phi \cup \{\phi\}$
7 $\mathbf{Z} \leftarrow Scope[\mathcal{T}'] - \mathbf{Y}$
8 Let $\prec$ be some ordering over $\mathbf{Z}$
9 **return** Sum-Product-VE($\Phi, \mathbf{Z}, \prec$)

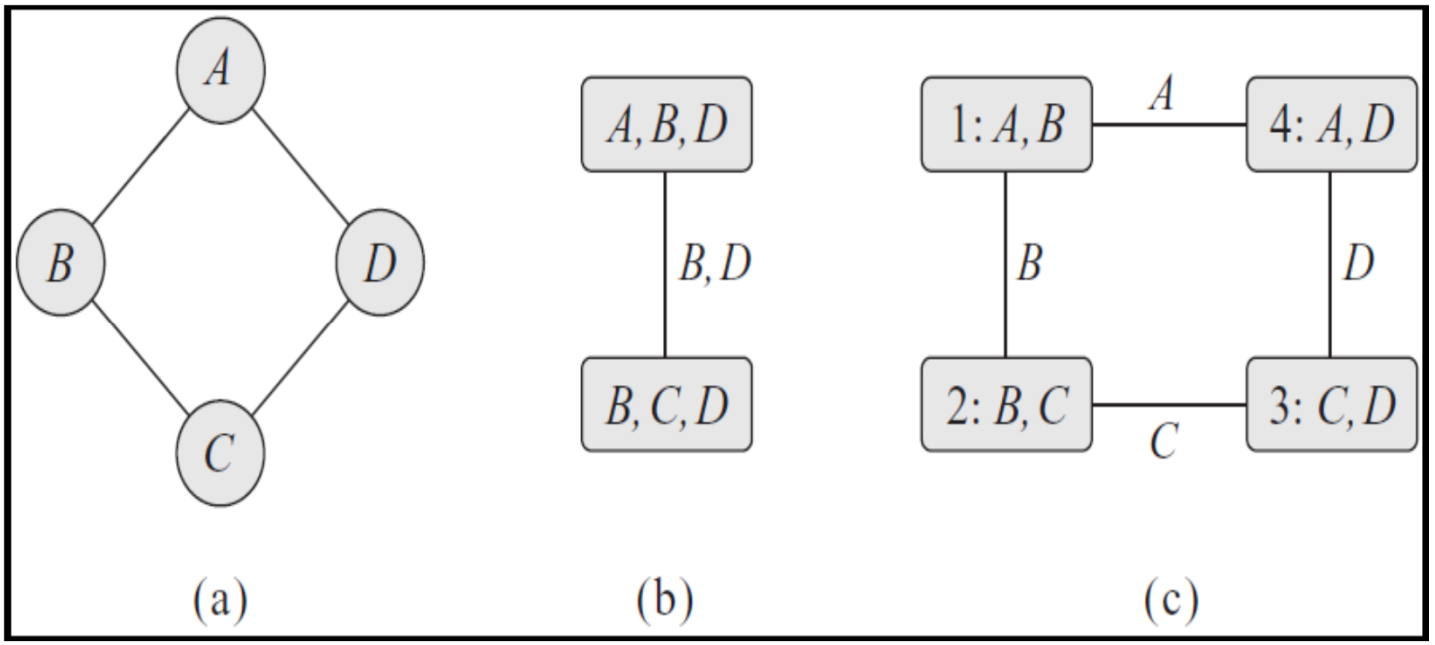
# Approximate Inference in Graphical Models

- Exact inference is NP-hard, may not be practical for large networks
  - Some factors may become exponentially large
- Consider a Grid MRF
- Can we convert to a clique tree
  - Size of maximal clique?
- Need for approximation



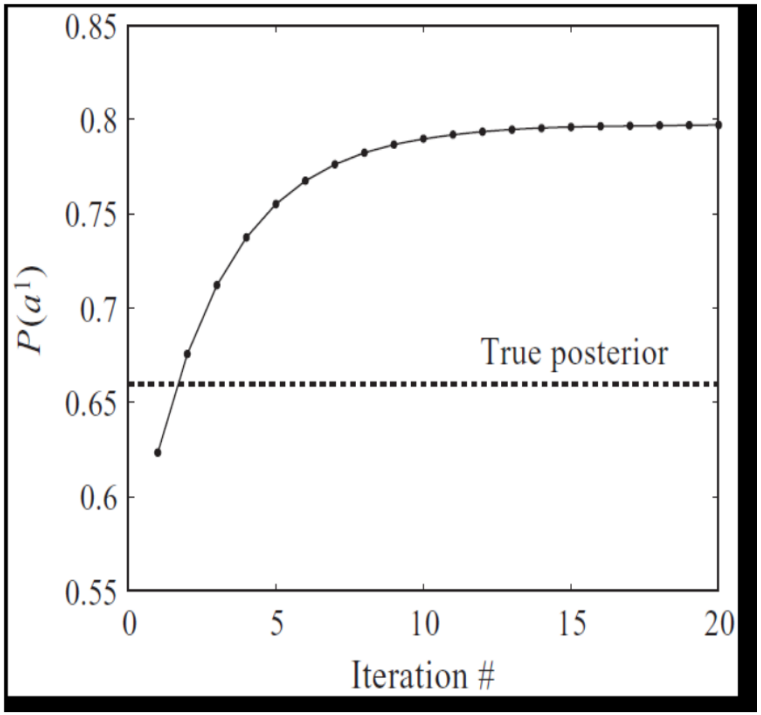
# A simple Graph with a Cycle

- Can convert to a tree, as in part (b) but cliques are larger.
  - Even for a small graph, if variables are multi-valued, size of factors grows fast.
- Part (c) is a cluster graph; can we apply BU (belief update) to it also?



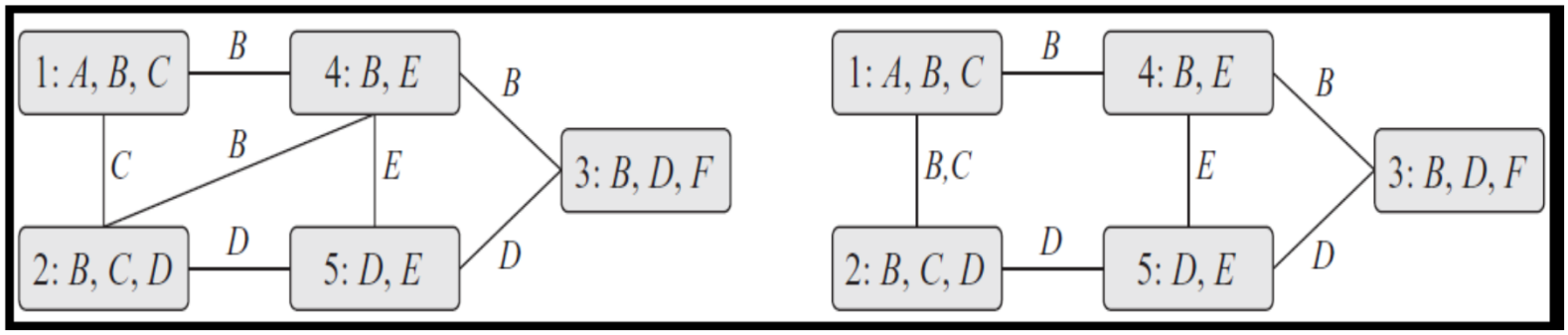
# Convergence

- Suppose all the potentials favor giving same values to variables in the pair in the potential (both taking value 1 or both taking value 0)
- Suppose a message comes back to {A, B} with a high value for $P(a^1)$. In the next iteration, this value will be favored and increased over and over again. Thus, there may be an overestimate (for this example)



# Cluster Graph BP

- Generalized Running Intersection Property:
  - *If X is in $\mathbf{C}_i$ and also in $\mathbf{C}_j$ then there is a single path between $\mathbf{C}_i$ and $\mathbf{C}_j$ , for which X $\in \mathbf{S}_e$ (sepset) for all edges e in the path*
  - Informally: one and only one path along which info about X can flow directly.
- Left figure satisfies running intersection property but circular influence is still possible: info about B flows from C3 to C2, via C4, but also via C5 (marginalize D between C3 and C5, passing a factor containing B, then again the same between C5 and C2).




- Note $\mathbf{S}_{i,j}$ not always = $\mathbf{C}_i \cap \mathbf{C}_j$ .

# Calibrated Cluster Graph

- A cluster graph is calibrated, if for each edge (i-j)

$$\sum_{\boldsymbol{C}_i - \boldsymbol{S}_{i,j}} \beta_i = \sum_{\boldsymbol{C}_j - \boldsymbol{S}_{i,j}} \beta_j;$$

- Note: this is different than similar equation for a cluster *tree* as in the latter case, sepset contains *all* the variables in common.
- In a calibrated graph with running intersection property, marginals of a variable X are identical in all clusters containing it

# Calibrated Cluster Graph

- To calibrate a graph, pass messages as for the cluster tree algorithm
  - In a loopy graph, no clique may be ready to transmit as they could all be waiting for messages from others
  - Initialize to ***1*** to avoid this (later passes will change the values)
- Cluster graph BP algorithm (11.1)
- All nodes can send messages synchronously at each time step

# Loopy Belief Propagation (LBP): Algorithm 11.1

- To calibrate a graph, pass messages as for the cluster tree algorithm
  - In a loopy graph, no clique may be ready to transmit as they could all be waiting for messages from others
  - Initialize to ***1*** to avoid this (later passes will change the values)
- All nodes can send messages synchronously at each time step
- In a graph with cycles, messages may circulate among the nodes in a cycle indefinitely
- LBP allows messages to keep being passed until there is no change (convergence is achieved) or some preset limit is reached

# LBP Scheduling

- Belief propagation needs to be scheduled
  - If a node has received all the input messages, it has a pending message for another node
  - If any input of a node changes, it has a new pending message
  - In each cycle, transmit all pending messages (in parallel)
  - To initiate, pass a unit message along each link
  - Repeat until no new messages are generated
- Convergence is not guaranteed but empirical evidence suggests that LBP converges to good solutions in many cases.
  - Works very well for *turbocodes*

**Algorithm 11.1 Calibration using sum-product belief propagation in a cluster graph**

**Procedure** CGraph-SP-Calibrate (
  $\Phi$, // Set of factors
  $\mathcal{U}$ // Generalized cluster graph $\Phi$
)
1 Initialize-CGraph
2 **while** graph is not calibrated
3 Select $(i–j) \in \mathcal{E}_{\mathcal{U}}$
4 $\delta_{i \to j}(\boldsymbol{S}_{i,j}) \leftarrow$ SP-Message$(i, j)$
5 **for** each clique $i$
6 $\beta_i \leftarrow \psi_i \cdot \prod_{k \in \text{Nb}_i} \delta_{k \to i}$
7 **return** $\{\beta_i\}$

**Procedure** Initialize-CGraph (
  $\mathcal{U}$
)
1 **for** each cluster $\boldsymbol{C}_i$
2 $\beta_i \leftarrow \prod_{\phi \,:\, \alpha(\phi)=i} \phi$
3 **for** each edge $(i–j) \in \mathcal{E}_{\mathcal{U}}$
4 $\delta_{i \to j} \leftarrow 1$
5 $\delta_{j \to i} \leftarrow 1$
6

**Procedure** SP-Message (
  i, // sending clique
  j // receiving clique
)
1 $\psi(\boldsymbol{C}_i) \leftarrow \psi_i \cdot \prod_{k \in (\text{Nb}_i - \{j\})} \delta_{k \to i}$
2 $\tau(\boldsymbol{S}_{i,j}) \leftarrow \sum_{\boldsymbol{C}_i - \boldsymbol{S}_{i,j}} \psi(\boldsymbol{C}_i)$
3 **return** $\tau(\boldsymbol{S}_{i,j})$

# Properties of Graph Calibrate

- Cluster graph invariant: multiplying clique beliefs still represents the original distribution, but each clique belief may not correspond to the correct marginal distribution

*Let $\mathcal{U}$ be a generalized cluster graph over a set of factors $\Phi$. Consider the set of beliefs $\{\beta_i\}$ and sepsets $\{\mu_{i,j}\}$ at any iteration of* CGraph-BU-Calibrate; *then*

$$\tilde{P}_\Phi(\mathcal{X}) = \frac{\prod_{i \in \mathcal{V}_\mathcal{U}} \beta_i[C_i]}{\prod_{(i-j) \in \mathcal{E}_\mathcal{U}} \mu_{i,j}[S_{i,j}]}.$$

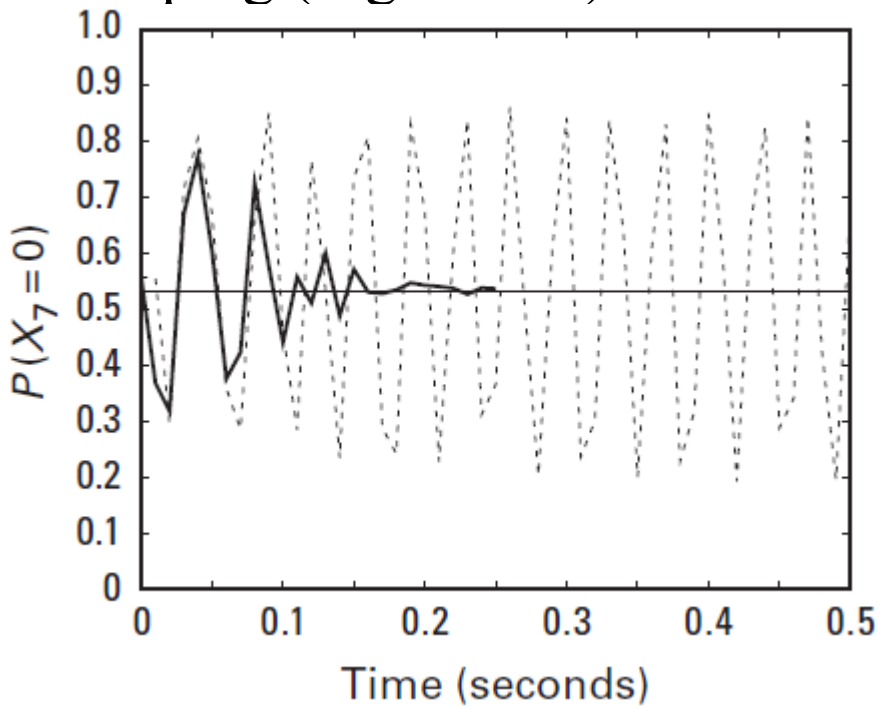
*where* $\tilde{P}_\Phi(\mathcal{X}) = \prod_{\phi \in \Phi} \phi$ *is the unnormalized distribution defined by* $\Phi$.

# Convergence

- No guarantee of convergence
- If it converges, it may not be to the correct marginals
- *Damping* may help convergence

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

- Some results of damping (Fig 11.C.1)

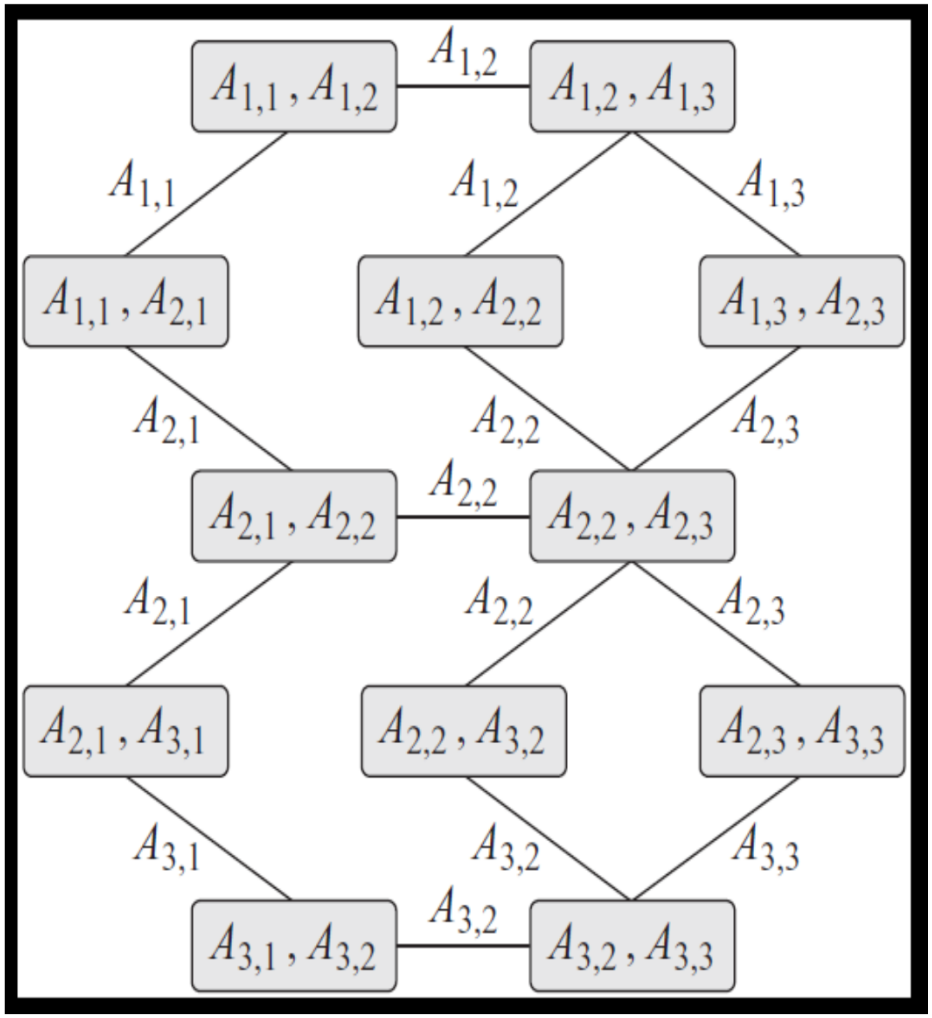


# Asynchronous Updating

- Algorithm does not require synchronous updates or updates in a "fair" order. Some orders may be better than others.
- Tree reparameterization (TRP)
  - Form a set of trees from subsets of nodes in the graph
    - Ensure that all nodes and links are in at least one tree
  - Select a tree and calibrate nodes in it (2-pass algorithm)
  - Repeat for other trees (from subset of nodes) and iterate
- Residual Belief Propagation (RBP)
  - Focus on part of graph where disagreement between neighbors is strongest (different distributions for sepset variables)
- Neither of above is guaranteed to converge or to converge to the right result but seem to be reasonable heuristics

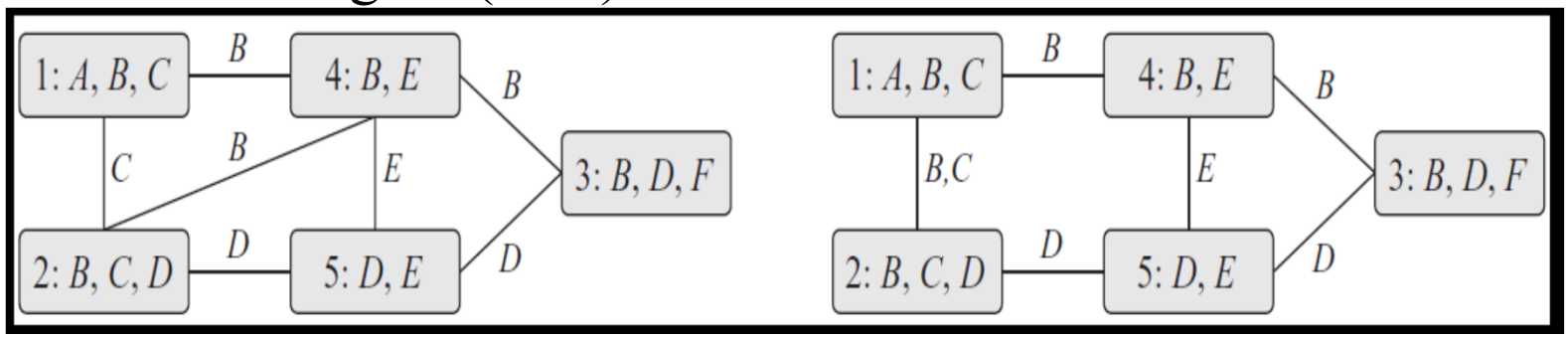
# Cluster Graph for Grid MRF

- Note: only pairwise cliques



# Constructing Cluster Graphs

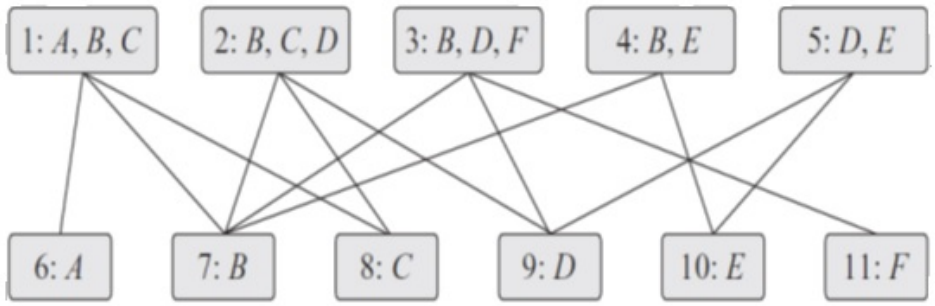
- Multiple graphs are possible for the same distribution
  - Also the case for cluster *trees* but they all give the same answer
  - Different graphs may converge to different approximations
- Consider earlier figure (11.3)




  - Left graph: Only propagates marginal over C bet $C_1$ and $C_2$. If $C_1$ induces strong correlation between B and C (say B=C), this is not communicated directly: marginal on C via edge (1-2), marginal over B via edge (4-2)
  - Right graph: Better at preserving the correlation but if we include all common variables in all sepsets, we may not be able to preserve the running intersection property (e.g. if we also connect $C_2$ and $C_4$)

# Bethe Cluster Graphs

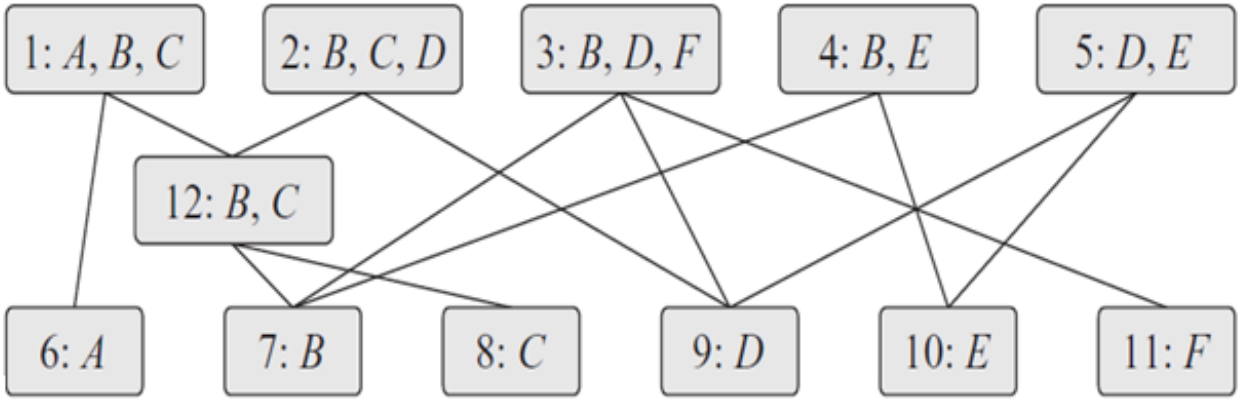
- How to construct clique graphs (from BNs) in general?
  - Connection to work in physics (Bethe)
- One layer corresponds to each factor
  - Family preservation property is satisfied
- Second layer is each individual variable
- Edge between each node, X, in second layer and cluster in first layer that includes the variable X
  - Satisfies running intersection property
  - Natural for pair-wise MRF (images, volumes…)
- However, strong dependencies may not get propagated




(a) $\mathcal{U}_3$

# Alternate Construction

- Introduce pair-wise nodes also but running intersection property may not be satisfied. Some pair-wise nodes may then be removed to restore RIP
- No known ideal configuration algorithm




(b) $\mathcal{U}_4$

# Continuous Distributions

- Distributions are not in the form of tables; use *density* functions instead of *distribution* functions
- In principle, sum product like algorithms still apply
  - Integration in place of summation
  - Functions may not be integrable in closed form; resulting distributions may not be in the same parametric family
- One option is to discretize variables but CPD can get very large for a fine grain discretization
- Gaussian distributions (including linear Gaussian Models) have particularly attractive properties
  - Representations are compact
  - Closed form solutions are possible
  - Intermediate factors remain Gaussian
  - We will mostly study such distributions only in this class

# Multi-variate Gaussian Distribution

- Eq. 7.1

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]$$

  - Mean vector, $\boldsymbol{\mu}$
  - $\Sigma$ is n x n covariance matrix,
    - must be positive definite, $\mathbf{x}^T\Sigma\,\mathbf{x} > 0$, $\mathbf{x} \neq 0$, for density to be well defined
    - Equivalent property: all eigenvalues are > 0
  - 2-D example $\boldsymbol{\mu} = \begin{pmatrix}\mu_x \\ \mu_y\end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix}\sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2\end{pmatrix}.$
  - Standard Multivariate Gaussian:

    $\boldsymbol{\mu} = \mathbf{0}$ (vector), $\Sigma = I$ (identity matrix) (1s on diagonal, 0s elsewhere)

# Next Class

- Read chapter 7, Chapter 14, sections 14.1, 14.2