Inference as Optimization

Sargur Srihari
srihari@cedar.buffalo.edu

Topics

- Approximate Inference
  - Exact Inference revisited
  - The Energy Functional
  - Optimizing the Energy Functional
- Exact Inference as Optimization
- Propagation-based Approximations

What is Exact Inference?

- We have a factorized distribution of the form

$$\boxed{P_{\Phi}(X) = \frac{1}{Z}\prod_{\phi\in\Phi}\phi(U_{\phi})}$$

  - where $U_{\phi} = Scope\,(\phi)$
  - Factors are:
    - CPDs in a BN or
    - potentials in a MN
- We are interested in answering queries:
  - about marginal probabilities of variables and
  - about the partition function

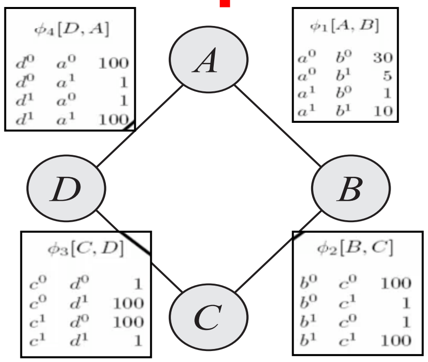
Approximate Inference

- Exact inference may not be possible
  - Time and Space Complexity of Clique Trees is exponential in tree-width of network
- Approach:
  - Find a target class Q of “easy” distributions and
  - Search for an instance within that class that best approximates P_Φ
  - Queries are then answered using inference on Q rather than P_Φ
  - Methods optimize a target function for measuring similarity between Q and P_Φ

Three Categories of Approximation

1. Message passing on Clique Tree
   - Loopy belief propagation
     - Optimize approximate versions of the energy functional
2. Message passing on Clique Trees with approximate messages
   - Called expectation propagation
     - Maximize exact energy functional but with relaxed constraints on Q
3. Mean-field method
   - Originates in statistical physics
     - Focus on Q that has simple factorization

Clique Tree MN representation



1.Gibbs Distribution

$$P(A,B,C,D) = \frac{1}{Z}\phi_1(A,B)\cdot\phi_2(B,C)\cdot\phi_3(C,D)\cdot\phi_4(D,A)$$

where

$$Z = \sum_{A,B,C,D}\phi_1(A,B)\cdot\phi_2(B,C)\cdot\phi_3(C,D)\cdot\phi_4(D,A)$$

$Z=7,201,840$

$$\tilde{P}_\Phi(A,B,C,D) = \phi_1(A,B)\phi_2(B,C)\phi_3(C,D)\phi_4(D,A)$$

Assignment				*Unnormalized*
a^0	b^0	c^0	d^0	300000
a^0	b^0	c^0	d^1	300000
a^0	b^0	c^1	d^0	300000
a^0	b^0	c^1	d^1	30
a^0	b^1	c^0	d^0	500
a^0	b^1	c^0	d^1	500
a^0	b^1	c^1	d^0	5000000
a^0	b^1	c^1	d^1	500
a^1	b^0	c^0	d^0	100
a^1	b^0	c^0	d^1	1000000
a^1	b^0	c^1	d^0	100
a^1	b^0	c^1	d^1	100
a^1	b^1	c^0	d^0	10
a^1	b^1	c^0	d^1	100000
a^1	b^1	c^1	d^0	100000
a^1	b^1	c^1	d^1	100000

2. Clique Tree (triangulated):

$1. A,B,D$ —$\{B,D\}$— $2. B,C,D$

Initial Potentials:

$$\psi_1(A,B,D) = \phi_1(A,B)\phi_2(B,C)\phi_3(C,D)\phi_4(D,A)$$
$$\psi_2(B,C,D) = \phi_1(A,B)\phi_2(B,C)\phi_3(C,D)\phi_4(D,A)$$

Beliefs (Clique and Sepset)

$$\beta_1(A,B,D) = \tilde{P}_\Phi(A,B,D) = \sum_C \psi_1(A,B,D) = \sum_C \phi_1(A,B)\phi_2(B,C)\phi_3(C,D)\phi_4(D,A)$$

e.g., $\beta_1(a^0,b^0,d^0) = 300,000 + 300,000 = 600,000$

$$\mu_{1,2}(B,D) = \sum_{C_1 - S_{1,2}} \beta_1(C_1) = \sum_A \beta_1(A,B,D)$$

e.g., $\mu_{1,2}(b^0,d^0) = 600,000 + 200 = 600,200$

$$\beta_2(B,C,D) = \tilde{P}_\Phi(B,C,D) = \sum_A \mu_{1,2}(B,D)\cdot\psi_2(B,C,D) = \sum_A \psi_2(B,C,D)$$

e.g., $\beta_2(b^0,c^0,d^0) = 300,000 + 100 = 300,100$

Assignment			$\max_C$
a^0	b^0	d^0	600,000
a^0	b^0	d^1	300,030
a^0	b^1	d^0	5,000,500
a^0	b^1	d^1	1,000
a^1	b^0	d^0	200
a^1	b^0	d^1	1,000,100
a^1	b^1	d^0	100,010
a^1	b^1	d^1	200,000

$\beta_1(A, B, D)$

Assignment		$\max_{A,C}$
b^0	d^0	600,200
b^0	d^1	1,300,130
b^1	d^0	5,100,510
b^1	d^1	201,000

$\mu_{1,2}(B, D)$

Assignment			[illegible]
b^0	c^0	d^0	300,[illegible]
b^0	c^0	d^1	1,300,[illegible]
b^0	c^1	d^0	300,[illegible]
b^0	c^1	d^1	[illegible]
b^1	c^0	d^0	[illegible]
b^1	c^0	d^1	100,[illegible]
b^1	c^1	d^0	5,100,[illegible]
b^1	c^1	d^1	100,[illegible]

$\beta_2(B, C, D)$

$$\tilde{P}_\Phi(a^1,b^0,c^1,d^0) = 100$$

$$\frac{\beta_1(a^1,b^0,d^0)\beta_2(b^0,c^1,d^0)}{\mu_{1,2}(b^0,d^0)} = \frac{200\cdot 300\cdot 100}{600\cdot 200} = 100$$

Constrained Optimization

- Inference task is one of optimizing an objective function over the class Q
- Lagrangian multipliers is most common method used
  - Produces a set of equations that characterize the optima of the objective
    - A set of fixed point equations that define each variable in terms of others
  - Fixed point equations derived from constrained optimization can be viewed as passing messages over a graph object

Cluster Tree Representation

- End-product of Belief Propagation is a calibrated cluster tree
- A calibrated set of beliefs represents a distribution
- We view exact inference as searching over the set of distributions Q that are representable by the cluster tree to find a distribution Q^* that matches P_Φ

Objective Function

- Search for a distribution Q that *minimizes* $D(Q \,||\, P_\Phi)$ where
  - The relative entropy between P_1 and P_2 is defined as

$$\boxed{D\left(P_1 \,||\, P_2\right) = E_{P_1}\left[\frac{\ln P_1\left[\chi\right]}{\ln P_2\left[\chi\right]}\right]}$$

  - It is always non-negative
  - Equal to 0 if and only if $P_1 = P_2$

The Optimization Task

- Find distribution Q that minimizes $D(Q \,||\, P_\Phi)$
- We are given:
  - a clique tree structure T for P_Φ
  - a set of beliefs

$$Q=\{\beta_i: i \in V_T\} \cup \{\mu_{i,j}: (i\text{-}j) \in E_T\}$$

where C_i are clusters in T, β_i denote beliefs over C_i and $\mu_{i,j}$ denotes beliefs $S_{i,j}$ of edges in T

- Set of beliefs in T defines a distribution Q by

$$\boxed{Q(\chi)=\frac{\prod_{i\in V_T}\beta_i}{\prod_{(i-j)\in V_T}\mu_{i,j}}}$$

Exact Inference as Optimization

- Exact inference is one of maximizing $-D(Q \| P_\Phi)$ over the space of calibrated sets Q

Ctree-Optimize-KL

- **Find** $Q=\{\beta_i: i \in V_T\} \cup \{\mu_{i,j}: (i\text{-}j) \in E_T\}$
- **Maximizing** $-D(Q \| P_\Phi)$
- **Subject to**

$$\boxed{\begin{array}{l} \mu_{i,j}\left[s_{i,j}\right]=\sum_{C_i-S_{i,j}} \beta_i\left(c_i\right) \quad \forall\left(i-j\right) \in E_T, \forall s_{i,j} \in Val\left(S_{i,j}\right) \\ \sum_{c_i} \beta_i\left(c_i\right)=1 \qquad \forall \mathrm{i} \in \mathrm{V}_T \end{array}}$$

Energy Functional

- Direct evaluation of $D(Q \| P_\Phi)$ is unwieldy

$$D(P_1 \| P_2) = E_{P_1}\left[\frac{\ln P_1[\chi]}{\ln P_2[\chi]}\right] = \sum_{\chi} P_1[\chi]\left[\frac{\ln P_1[\chi]}{\ln P_2[\chi]}\right]$$

  - Because summation over all χ is infeasible in practice
- Instead use equivalent form $D(Q \| P_\Phi) = \ln Z - F(\tilde{P}_\Phi, Q)$
  - Where F is the energy functional

$$F[\tilde{P}_\Phi, Q] = E_Q[\ln \tilde{P}(\chi)] + H_Q(\chi) = \sum_{\phi \in \Phi} E_Q[\ln \phi] + H_Q(\chi)$$

- Since the term $\ln Z$ does not depend on Q,
  - minimizing relative entropy $D(Q \| P_\Phi)$ is equivalent to maximizing the energy functional $F(\tilde{P}_\Phi, Q)$
- Energy functional $F[\tilde{P}_\Phi, Q] = \sum_{\phi \in \Phi} E_Q[\ln \phi] + H_Q(\chi)$ has two terms:
  - energy term (expectation of logs of factors in Φ) and entropy term

Energy Functional Optimization

- Problem of finding good approximation Q is one of maximizing the energy functional
- Equivalently minimizing the relative entropy
- Choose approximation Q that allows for efficient inference
- Energy Functional is a lower bound on partition function
  - Since $D(Q||PF) \geq 0$ we have $\ln Z \geq F\left[\tilde{P}_{\Phi}, Q\right]$
  - Useful since partition function is usually the hardest part of inference
    - Plays important role in learning

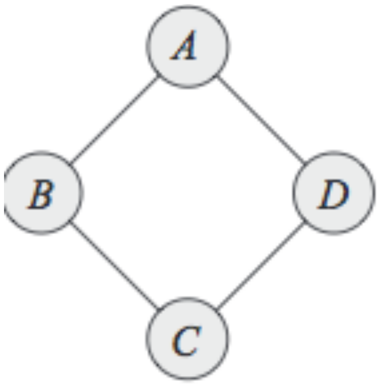
Approximate Inference

- Strategies for optimizing the energy functional
- Referred to as Variational Methods
- Variational calculus: finding optima of a functional
  - E.g., distribution that maximizes entropy

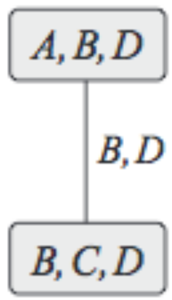
Propagation-based Approaches

- Work with cluster graphs instead of clique trees

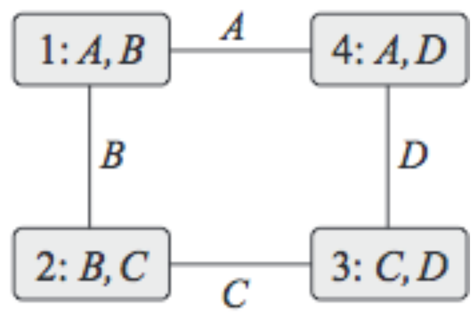
A simple network



A Clique Tree

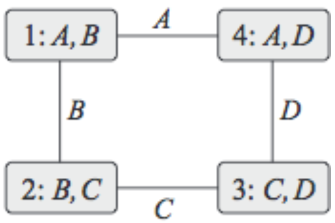


A Cluster Graph

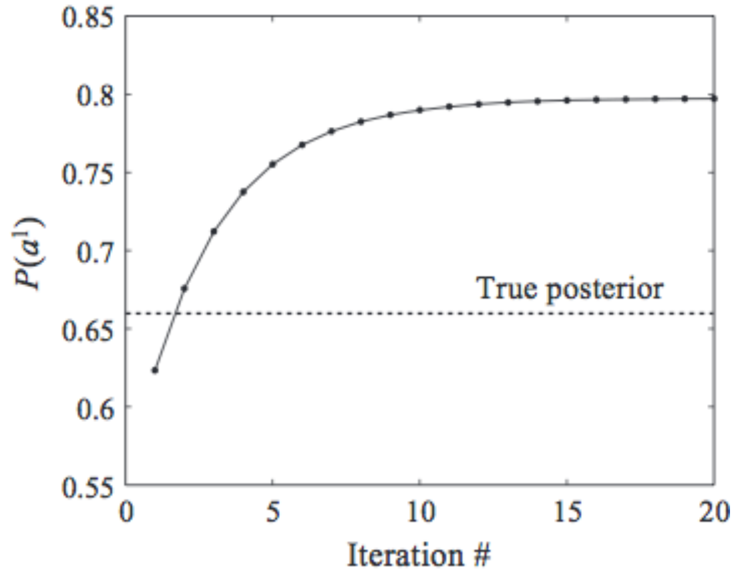


- Cluster graph may contain loops
  - Algorithm Ctree-BU-calibrate can be used
  - Called *Loopy Belief Propagation*
- Clusters are smaller than in Clique Tree

Loopy Belief Propagation



- Propagate messages
  - in following order $\mu_{1,2}, \mu_{2,3}, \mu_{3,4}, \mu_{4,1}$
  - Cluster $\{A,B\}$ passes information to cluster $\{B,C\}$ through a marginal distribution on B
    - In final message $\mu_{4,1}$ information reaches original cluster
  - All potentials prefer consensus assignment



Coding Theory and Loopy BP

- Sending messages over a noisy channel and recovering
- We wish to send a k-bit message $u_1,\ldots u_k$
- Encode the message using n bits $x_1,..x_n$
- Resulting in corrupted outputs $y_1,..,y_n$
- Task is to recover an estimate $\tilde{u}_1,\ldots,\tilde{u}_k$ from $y_1,.., y_n$
- Message decoding can be formulated as a probabilistic inference task

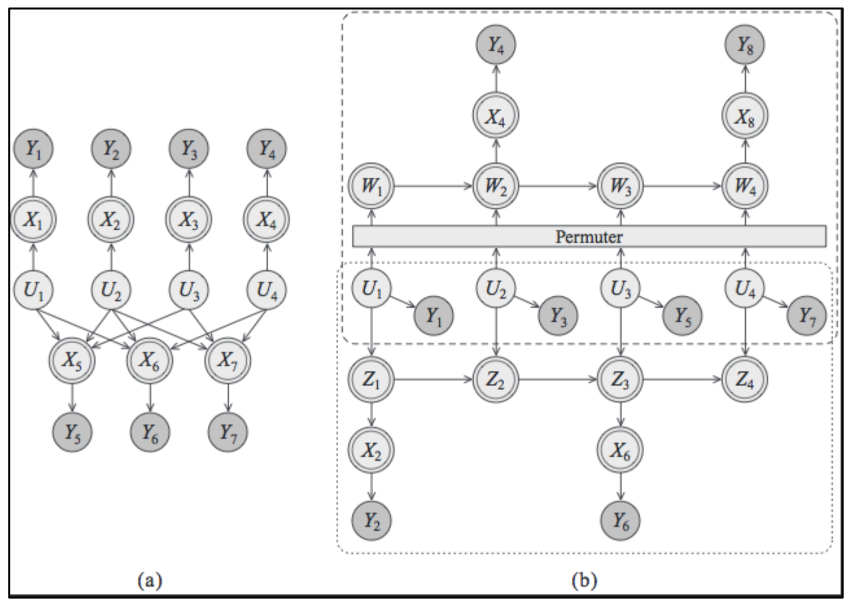
Noise Models and Error Rate

- Outputs can be discrete or continuous
  - Different channels introduce different noise
    - Addition of Gaussian noise
    - Flip bits independently with some probability p
    - Noise is added in a correlated way
- Bit error rate
  - Probability that bit is decoded incorrectly
- Rate of a code
  - k/n: ratio of no. of msg bits to no. of transmit bits
    - Repetition code: transmit each bit 3 times, decode by majority vote, has bit error rate p^3+3p^2
    - Shannon: for a given rate, max noise level tolerated while achieving a certain bit error rate

Two Examples of Codes

- A $k=4$, $n=7$ parity check code where every four message bits are sent along with three bits that encode parity checks
- A $k=4$, $n=8$ turbocode

BN formulation
With Belief propagation



Cluster Graph Belief Propagation

- Sum-Product BP in a Cluster Graph
  - Procedure Cgraph-SP-Calibrate (

11.3. Propagation-Based Approximation

edge $(i–j)$, connecting the clusters C_i and C_j, w

$$\sum_{C_i - S_{i,j}} \beta_i = \sum_{C_j - S_{i,j}} \beta_j;$$

that is, the two clusters agree on the marginal
tion is weaker than cluster tree calibration, since
joint marginal of all the variables they have in c
sepset. However, if a calibrated cluster graph satis
the marginal of a variable X is identical in all the

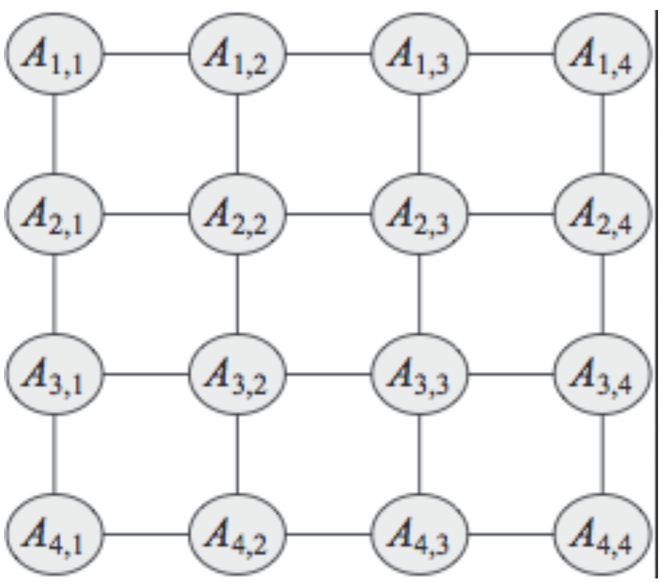
Algorithm 11.1 Calibration using sum-product be

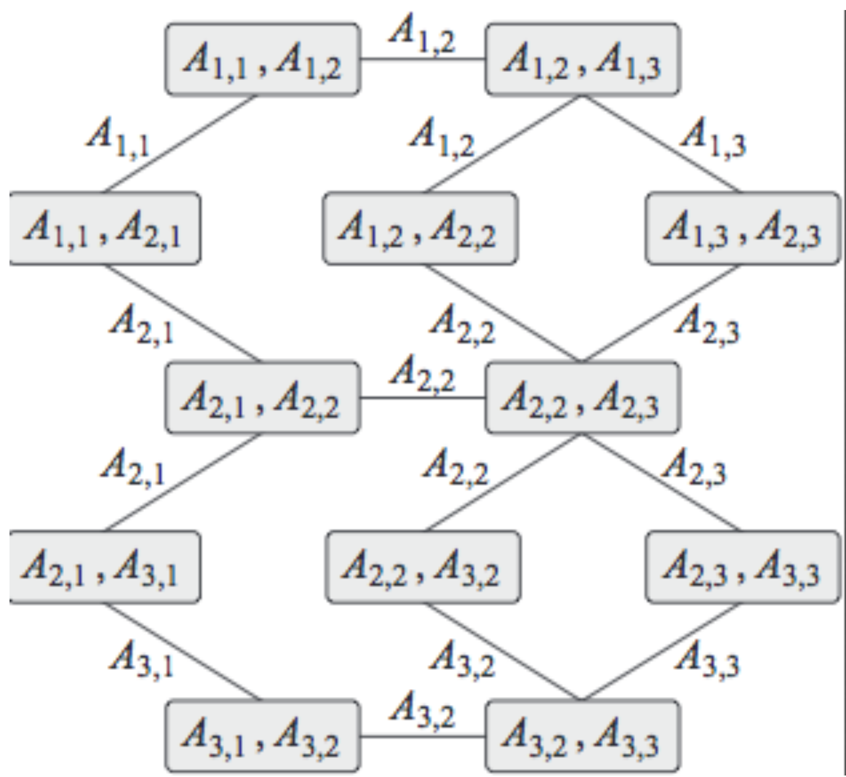
```
Procedure CGraph-SP-Calibrate (
    Φ,    // Set of factors
    U     // Generalized cluster graph Φ
  )
    Initialize-CGraph
    while graph is not calibrated
      Select (i–j) ∈ E_U
      δ_{i→j}(S_{i,j}) ← SP-Message(i, j)
    for each clique i
      β_i ← ψ_i · ∏_{k∈Nb_i} δ_{k→i}
    return {β_i}

Procedure Initialize-CGraph (
    U
  )
    for each cluster C_i
      β_i ← ∏_{φ : α(φ)=i} φ
    for each edge (i–j) ∈ E_U
      δ_{i→j} ← 1
      δ_{j→i} ← 1

Procedure SP-Message (
    i,   // sending clique
    j    // receiving clique
  )
    ψ(C_i) ← ψ_i · ∏_{k∈(Nb_i−{j})} δ_{k→i}
    τ(S_{i,j}) ← Σ_{C_i−S_{i,j}} ψ(C_i)
    return τ(S_{i,j})
```

How do we calibrate a cluster graph? Because cal
adjoining clusters, we want to try to ensure that each

Cluster Graph Belief Propagation

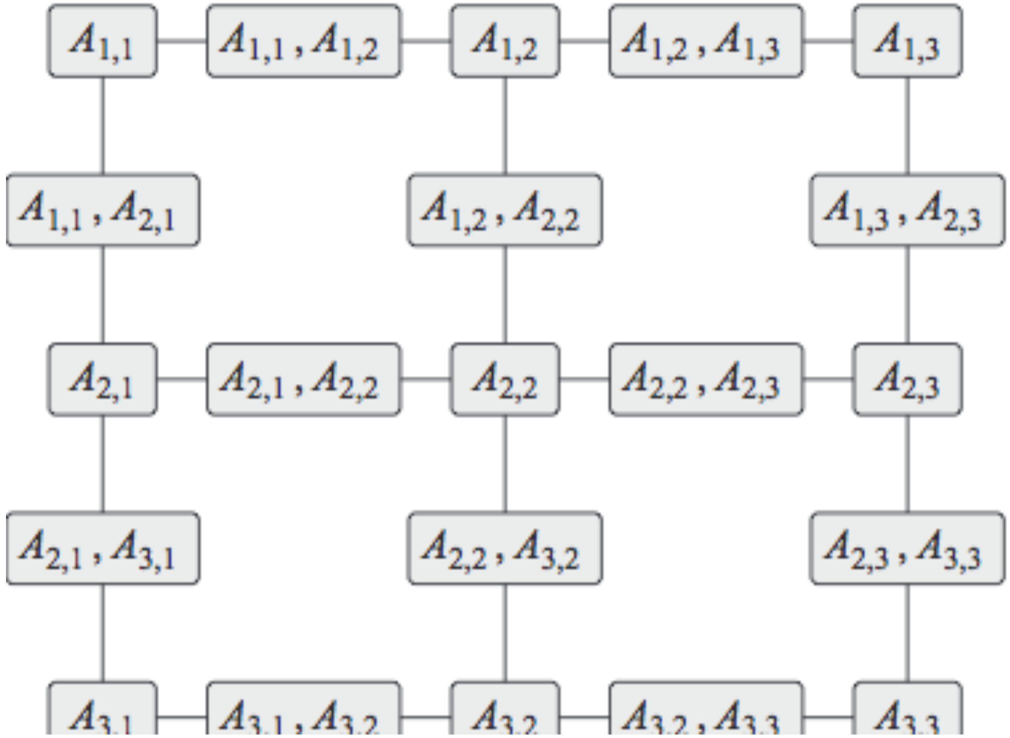
- 4x4 two-dimensional grid network
- Generalized cluster graph for 3 x 3 network





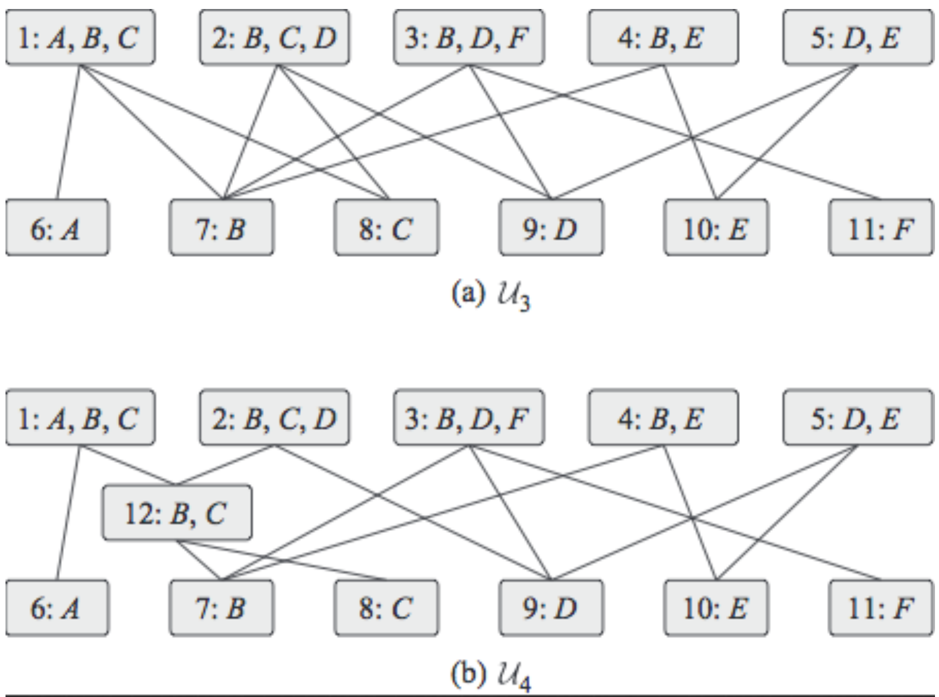
Cluster Graph for Pairwise Markov Network

- Potentials defined over nodes and edges
- For a 3x3 grid



Bethe Cluster Graph

- Generalizes pairwise clustering
- Bipartite graph; first layer of large clusters and second layer of univariate clusters



(a) $\mathcal{U}_3$

(b) $\mathcal{U}_4$

Use of Cluster Graphs

- Cluster graph belief propagation are a general purpose approximation inference method
- Can be used with trees of high width
- Many applications
  - Message decoding in communications
  - Predicting protein structure
  - Image segmentation
- Some Caveats: Need not converge, multiple optima