
グラフィカルモデルによる推論 — 確率伝搬法(2)

Kenji Fukumizu

The Institute of Statistical Mathematics

計算推論科学概論 II (2010年度, 後期)



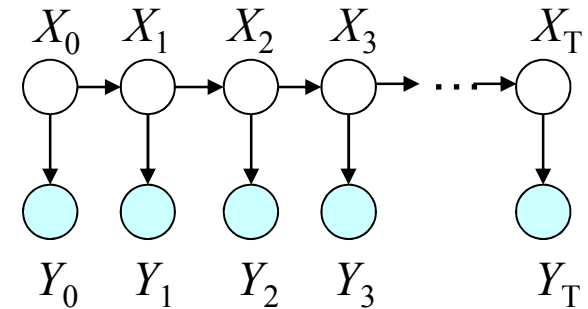
Inference on Hidden Markov Model

Inference on Hidden Markov Model

- Review: HMM model

$$p(X, Y) = p(X_0)p(Y_0 | X_0)\prod_{t=1}^T p(X_t | X_{t-1})p(Y_t | X_t)$$

X_t : hidden state, finite



- Inference

- Compute

$$p(X_t | Y_1, \dots, Y_T) \quad \text{for any } t$$

Naïve computation requires $O(K^T)$ operations, exponential on the sequence length.

K : number of hidden states

Belief Propagation on HMM

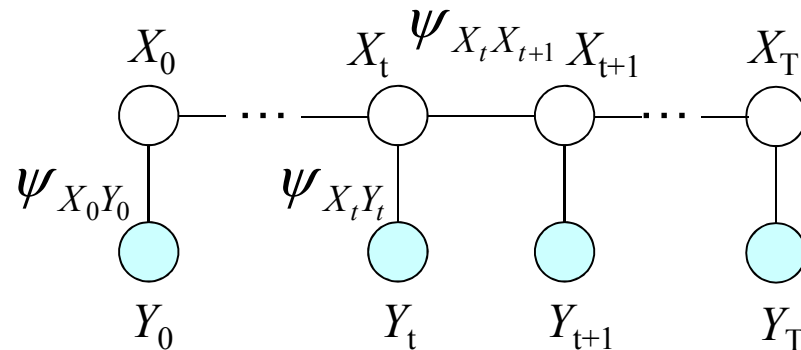
- BP for undirected tree representation
 - Clique potentials

$$\psi_{X_0Y_0}(X_0, Y_0) = p(X_0)p(Y_0 | X_0) = p(X_0, Y_0)$$

$$\psi_{X_tY_t}(X_t, Y_t) = p(Y_t | X_t) \quad (1 \leq t \leq T)$$

$$\psi_{X_{t-1}X_t}(X_{t-1}, X_t) = p(X_t | X_{t-1}) \quad (1 \leq t \leq T)$$

$$p(X, Y) = \psi_{X_0Y_0}(X_0, Y_0) \prod_{t=1}^T \psi_{X_{t-1}X_t}(X_{t-1}, X_t) \psi_{X_tY_t}(X_t, Y_t)$$

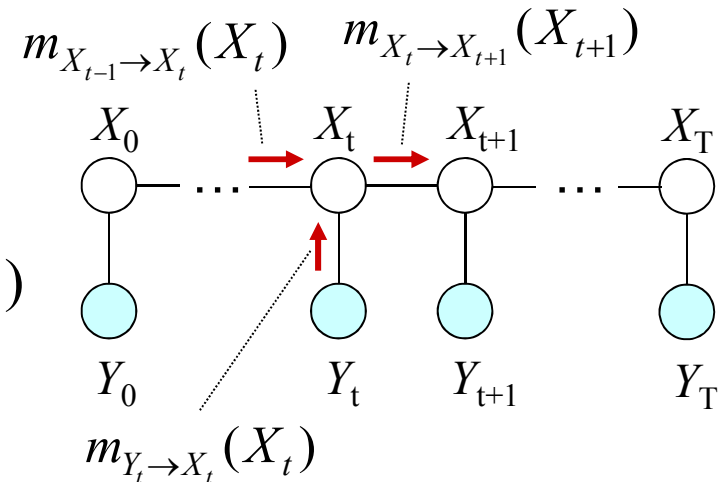


Belief Propagation on HMM

- Upward message passing

$$m_{X_t \rightarrow X_{t+1}}(X_{t+1}) = \sum_{X_t} \psi_{X_t X_{t+1}}(X_t, X_{t+1}) \times m_{X_{t-1} \rightarrow X_t}(X_t) m_{Y_t \rightarrow X_t}(X_t)$$

$$m_{Y_t \rightarrow X_t}(X_t) = p(Y_t | X_t) \quad (Y_t \text{ is given.})$$



$$m_{X_t \rightarrow X_{t+1}}(X_{t+1}) = \sum_{X_t} p(X_{t+1} | X_t) m_{X_{t-1} \rightarrow X_t}(X_t) p(Y_t | X_t)$$

$$= \sum_{X_t} A_{X_t, X_{t+1}} \boxed{m_{X_{t-1} \rightarrow X_t}(X_t) p(Y_t | X_t)} \quad (A: \text{transition matrix})$$

$$\alpha(X_t)$$

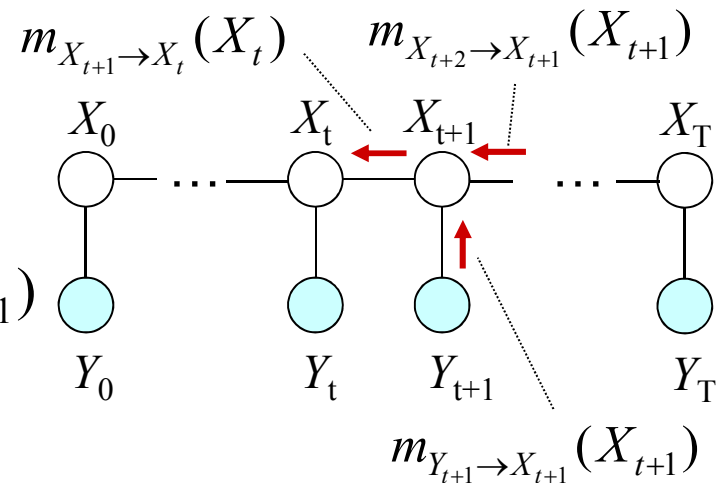


$$\alpha(X_{t+1}) = \sum_{X_t} A_{X_t, X_{t+1}} \alpha(X_t) p(Y_{t+1} | X_{t+1}) \quad \text{update rule}$$

Belief Propagation on HMM

- Downward message passing

$$m_{X_{t+1} \rightarrow X_t}(X_t) = \sum_{X_{t+1}} \psi_{X_t X_{t+1}}(X_t, X_{t+1}) \times m_{X_{t+2} \rightarrow X_{t+1}}(X_{t+1}) m_{Y_{t+1} \rightarrow X_{t+1}}(X_{t+1})$$



$$\begin{aligned} m_{X_{t+1} \rightarrow X_t}(X_t) &= \sum_{X_{t+1}} p(X_{t+1} | X_t) m_{X_{t+2} \rightarrow X_{t+1}}(X_{t+1}) p(Y_{t+1} | X_{t+1}) \\ &= \sum_{X_{t+1}} A_{X_t, X_{t+1}} m_{X_{t+2} \rightarrow X_{t+1}}(X_{t+1}) p(Y_{t+1} | X_{t+1}) \end{aligned}$$

$$\beta(X_t) \equiv m_{X_{t+1} \rightarrow X_t}(X_t)$$



$$\beta(X_t) = \sum_{X_{t+1}} A_{X_t, X_{t+1}} \beta(X_{t+1}) p(Y_{t+1} | X_{t+1}) \quad \text{update rule}$$

Belief Propagation on HMM

- Marginals

$$\begin{aligned} p(X_t, Y_0, \dots, Y_T) &= m_{Y_t \rightarrow X_t}(X_t) m_{X_{t-1} \rightarrow X_t}(X_t) m_{X_{t+1} \rightarrow X_t}(X_t) \\ &= \boxed{p(Y_t | X_t) m_{X_{t-1} \rightarrow X_t}(X_t)} \boxed{m_{X_{t+1} \rightarrow X_t}(X_t)} \\ &\qquad \qquad \qquad \alpha(X_t) \qquad \qquad \qquad \beta(X_t) \end{aligned}$$



$$p(X_t, Y_0, \dots, Y_T) = \alpha(X_t) \beta(X_t) \quad (Y_t\text{'s are given.})$$

Hence,

$$p(Y_0, \dots, Y_T) = \sum_{X_t} \alpha(X_t) \beta(X_t) \quad (t \text{ is arbitrary})$$

and

$$p(X_t | Y_0, \dots, Y_T) = \frac{\alpha(X_t) \beta(X_t)}{\sum_{X_t} \alpha(X_t) \beta(X_t)}$$

Forward-Backward Algorithm

■ Summary

- Forward-backward algorithm (α - β algorithm)

$$\alpha(X_{t+1}) = \sum_{X_t} A_{X_t, X_{t+1}} \alpha(X_t) p(Y_{t+1} | X_{t+1})$$

$$\beta(X_t) = \sum_{X_{t+1}} A_{X_t, X_{t+1}} \beta(X_{t+1}) p(Y_{t+1} | X_{t+1})$$

$$\alpha(X_0) = p(X_0, Y_0), \quad \beta(X_T) = 1$$

- Marginals

$$p(X_t, Y_0, \dots, Y_T) = \alpha(X_t) \beta(X_t)$$

$$p(Y_0, \dots, Y_T) = \sum_{X_t} \alpha(X_t) \beta(X_t) \quad - \text{likelihood of } Y \text{ (any } t)$$

$$p(X_t | Y_0, \dots, Y_T) = \frac{\alpha(X_t) \beta(X_t)}{\sum_{X_t} \alpha(X_t) \beta(X_t)} \quad - \text{smoothing}$$

Forward-Backward Algorithm

- Meaning of α and β

$$\alpha(X_t) = p(Y_0, \dots, Y_t, X_t) \quad (0 \leq t \leq T)$$

$$\beta(X_t) = p(Y_{t+1}, \dots, Y_T | X_t) \quad (0 \leq t \leq T - 1)$$

(Proof: homework)

Forward-Backward Algorithm

- The ordinary derivation of α - β algorithm uses

$$\alpha(X_t) = p(Y_0, \dots, Y_t, X_t), \quad \beta(X_t) = p(Y_{t+1}, \dots, Y_T | X_t)$$

as definitions, and derives the update rules.

- Confirm again

by
$$p(X_t, Y_0, \dots, Y_T) = \alpha(X_t)\beta(X_t)$$

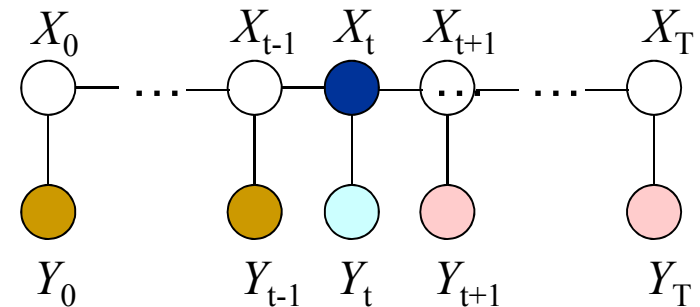
$$p(X_t, Y_0, \dots, Y_T)$$

$$= p(X_t)p(Y_1, \dots, Y_T | X_t)$$

$$= \frac{p(X_t)p(Y_1, \dots, Y_t | X_t)}{\alpha(X_t)} \frac{p(Y_{t+1}, \dots, Y_T | X_t)}{\beta(X_t)}$$

$$\alpha(X_t)$$

$$\beta(X_t)$$



Forward-Backward Algorithm

- Computational cost of the forward-backward algorithm cost is $O(K^2T)$, which is **linear** to the sequence length.
- Smoothing, filtering, and prediction are done by the algorithm;

- **smoothing:**

$$p(X_t | Y_0, \dots, Y_T) = \frac{\alpha(X_t)\beta(X_t)}{\sum_{X_t} \alpha(X_t)\beta(X_t)}$$

- **filtering:**

$$\alpha(X_t) = p(Y_0, \dots, Y_t, X_t),$$

$$\longrightarrow p(X_t | Y_0, \dots, Y_t) = \frac{p(Y_0, \dots, Y_t, X_t)}{p(Y_0, \dots, Y_t)} = \frac{\alpha(X_t)}{\sum_{X_t} \alpha(X_t)}$$

- **prediction:**

$$p(X_{t+1} | Y_0, \dots, Y_t) = \sum_{X_t} p(X_{t+1} | X_t) p(X_t | Y_0, \dots, Y_t) = \frac{\sum_{X_t} A_{X_t, X_{t+1}} \alpha(X_t)}{\sum_{X_t} \alpha(X_t)}$$

Forward-Backward Algorithm

- Prediction and filtering are computed **sequentially**.

For each time step, the update of $\alpha(X_t)$ with the new observation Y_t is sufficient.

We do not need to access the older variables of Y_s .

Mini-Summary

- Belief propagation is applicable to the inference of HMM
 - HMM is a tree → BP is applicable.
 - BP for smoothing derives the forward-backward algorithm.
 - Smoothing for all the hidden variables is done by the computation of the linear cost in the length.
 - BP for prediction and filtering derive sequential (forward) algorithm.



Inference on Non-Tree Graphs

Methods for Non-tree Graphs

■ Loopy Belief Propagation

- Application of BP updates to general graphs, though they have loops.
- An approximation algorithm.
- There is no theoretical guarantee for convergence or correctness.

■ Junction Tree Algorithm

- Propagation algorithm on the “clique tree”.
- Exactness of the resulting marginals are guaranteed, while the marginals are obtained only for the cliques.
- Efficiency of the algorithm depends on the clique tree derived from the original graph.

Loopy Belief Propagation

(Murphy, K., Weiss, Y., and Jordan, M. 1999).

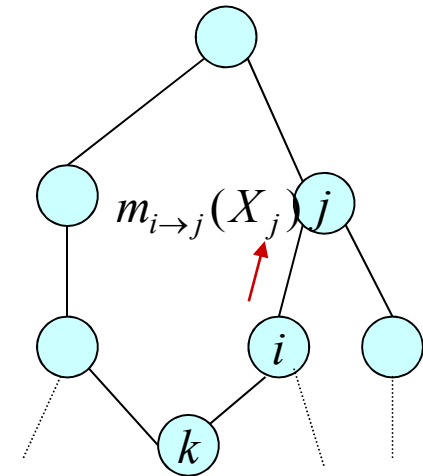
ALGORITHM

- The update rule is the same as the BP for trees.

$$m_{i \rightarrow j}(X_j) = \sum_{X_i} \psi_{ji}(X_j, X_i) \prod_{k \in \text{ne}(i) \setminus \{j\}} m_{k \rightarrow i}(X_i)$$

- The order of updates is arbitrary:
an arbitrary ordering, simultaneous updates, etc.
- Repeat the updates until some convergence criterion is satisfied.
- Compute all the (approximated) marginals by

$$p(X_i) = \frac{b(X_i)}{\sum_{X_i} b(X_i)}, \quad b(X_i) = \prod_{j \in \text{ne}(i)} m_{j \rightarrow i}(X_i)$$



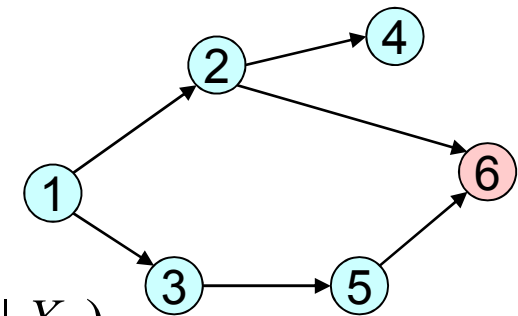
Loopy Belief Propagation

- There are no theoretical guarantees for convergence or correctness.
→ Current research issue.
- In many practical examples, loopy BP shows fast convergence and high accuracy.
 - Decoding method of error correcting codes (turbo-code)

Junction Tree Algorithm

- Basic idea: marginalization by **elimination**

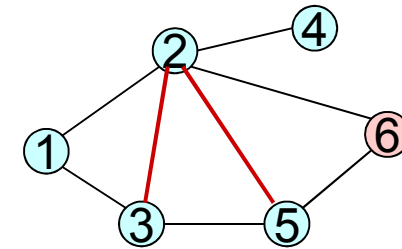
Example



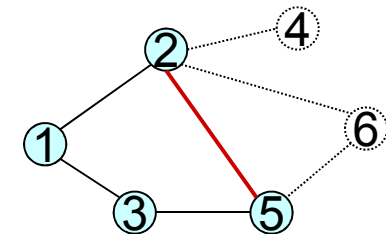
$$\begin{aligned}
 p(X_1, X_6 = e) &= \sum_{X_2, X_3, X_4, X_5, X_6} p(X_1) p(X_2 | X_1) p(X_3 | X_1) p(X_4 | X_2) \\
 &\quad \times p(X_5 | X_3) p(X_6 = e | X_2, X_5) \\
 &= \sum_{X_2, X_3, X_5} p(X_1) p(X_2 | X_1) p(X_3 | X_1) p(X_5 | X_3) \sum_{X_6} p(X_6 = e | X_2, X_5) \sum_{X_4} p(X_4 | X_2) \\
 &= \sum_{X_2, X_3} p(X_1) p(X_2 | X_1) p(X_3 | X_1) \sum_{X_5} p(X_5 | X_3) m_6(X_2, X_5) \\
 &= \sum_{X_2, X_3} p(X_1) p(X_2 | X_1) p(X_3 | X_1) m_5(X_2, X_3) \\
 &= \sum_{X_2, X_3} \psi(X_1, X_2, X_3) \quad \leftarrow \text{marginalization is easy}
 \end{aligned}$$

Junction Tree Algorithm

New cliques appears in the process of successive eliminating variables.



$$p(X_1, X_6 = e) = \sum_{X_2, X_3, X_4, X_5, X_6} p(X_1) p(X_2 | X_1) p(X_3 | X_1) p(X_4 | X_2) \times p(X_5 | X_3) p(X_6 = e | X_2, X_5)$$

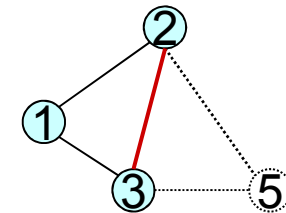


$$= \sum_{X_2, X_3, X_5} p(X_1) p(X_2 | X_1) p(X_3 | X_1) p(X_5 | X_3) \sum_{X_6} p(X_6 = e | X_2, X_5) \sum_{X_4} p(X_4 | X_2)$$

marginalizing out 6 connects 2 and 5



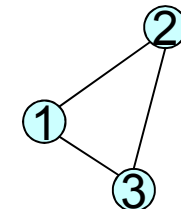
$$= \sum_{X_2, X_3} p(X_1) p(X_2 | X_1) p(X_3 | X_1) \sum_{X_5} p(X_5 | X_3) \underline{m_6(X_2, X_5)}$$



marginalizing out 5 connects 2 and 3

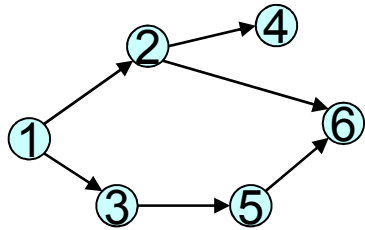


$$= \sum_{X_2, X_3} p(X_1) p(X_2 | X_1) p(X_3 | X_1) \underline{m_5(X_2, X_3)} = \sum_{X_2, X_3} \psi(X_1, X_2, X_3)$$

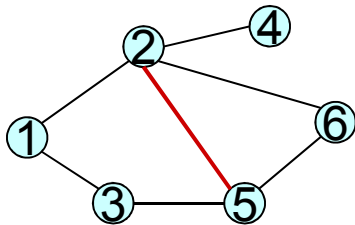


Junction Tree Algorithm

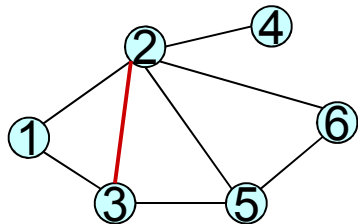
- Sketch of JT algorithm



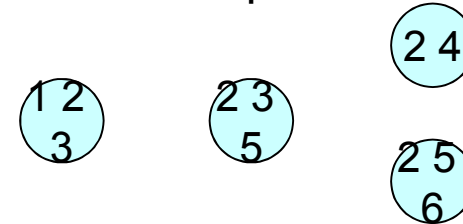
1. Moralization



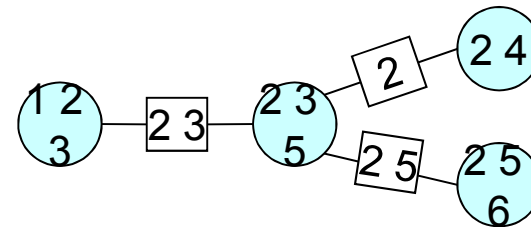
2. Triangulation.



3. Find all the cliques

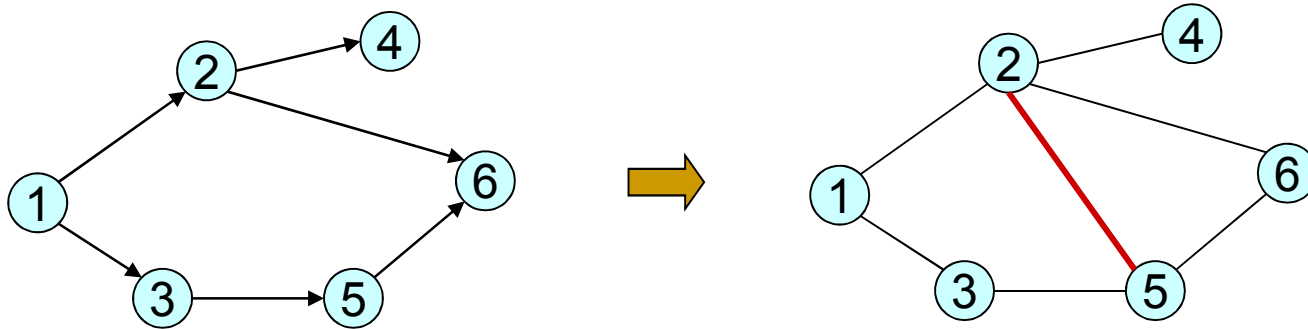


4. Make a junction tree (clique tree).



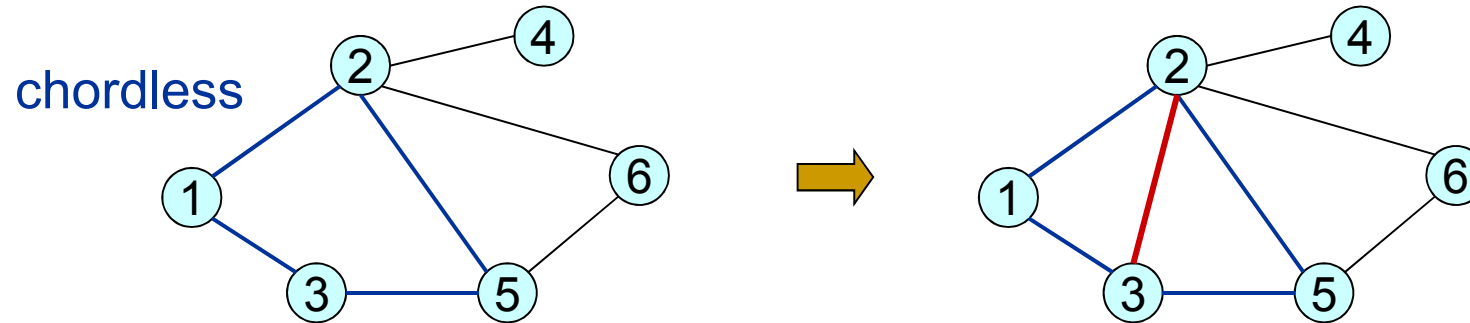
5. Propagate messages → Marginal probabilities of all the cliques.

Moralization



- ❑ **Moralization**: connect parents for each node.
- ❑ Make an undirected graph by removing the directions.
- ❑ No additional conditional independence relations are suggested by moralization.

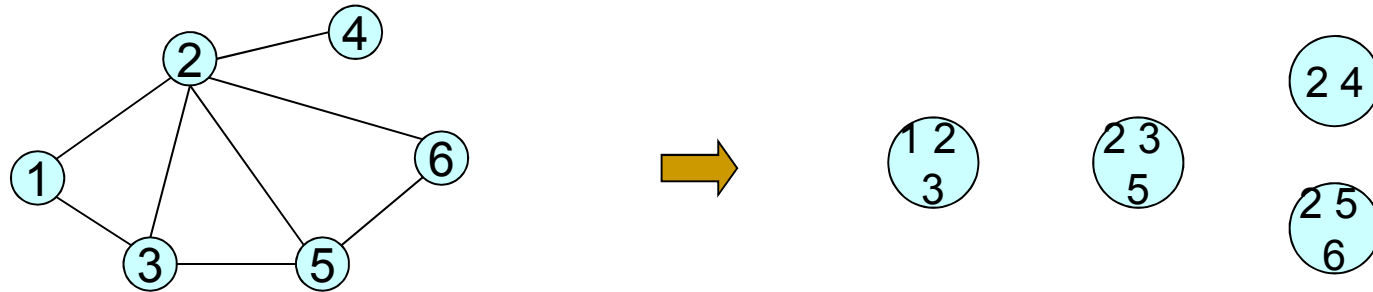
Triangulation



- Triangulation: make a graph such that there are no loops of length larger than 3 without chord.
 - A **chord** is an edge that connects non-consecutive two nodes in a loop.
- Triangulation guarantees the running intersection property of the junction tree.

Junction Tree

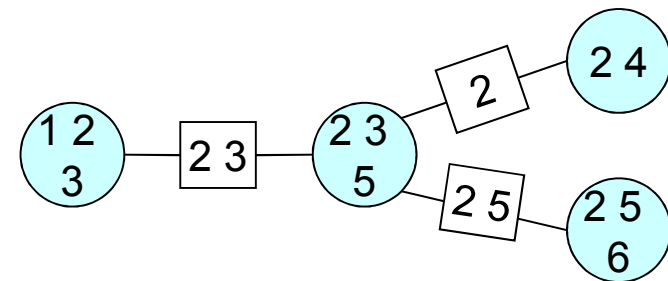
- Find all the cliques



- Make a junction tree

- **Junction tree**

- Tree of the cliques
- Each edge has a **separator** (intersection of connected nodes)

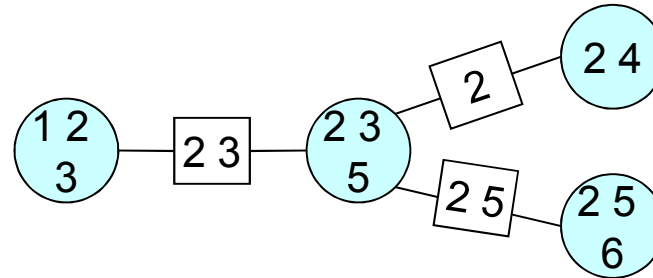


- **Running intersection property:**

if a variable appears in multiple nodes, it must appear in all intermediate nodes in the tree.

Junction Tree Propagation

$$p(X) = \prod_{C: \text{clique}} \psi_C(X_C)$$



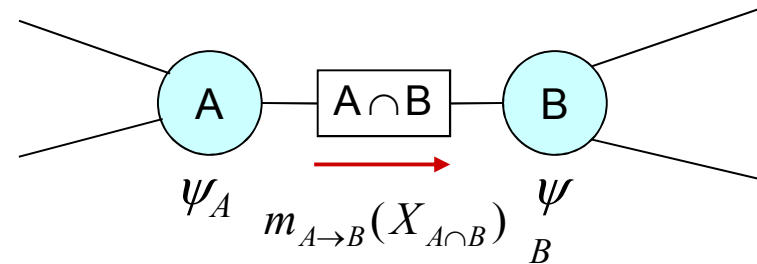
- Initial potentials

$\psi_C(X_C)$ given by the product of potentials in C

- Run belief propagation on the junction tree

$$m_{A \rightarrow B}(X_{A \cap B})$$

$$= \sum_{X_{A \setminus B}} \psi_A(X_A) \prod_{C \in N(A)} m_{C \rightarrow A}(X_{C \cap A})$$



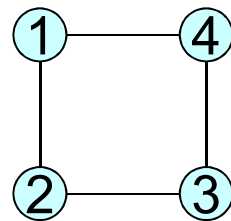
- After the upward and downward updates, the marginals are given by

$$p(X_C) = \psi_C(X_C), \quad p(X_S) = \phi_S(X_S).$$

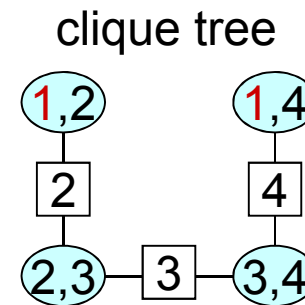
Junction Tree Propagation

Remarks:

- The running intersection property of the junction tree ensures that the propagation procedure gives marginal-out.



Not triangulated



Running intersection
does not hold.

- The computational cost of JT depends on the size of cliques. If the original graph is complete, there is no gain.

Mini-Summary

- Propagation algorithms for non-tree graphs
 - Loopy BP: approximation algorithm
Direct application of the BP update rule for general graphs.
 - Junction tree algorithm: exact marginalization for cliques.
- Extension of propagation algorithm
 - BP-type algorithms are obtained if the two operations satisfy the distribution law.
e.g. max-product, max-sum