Course 16:198:520: Introduction To Artificial Intelligence
Lecture 9

Markov Networks

Abdeslam Boularias

Monday, October 14, 2015

Overview

- Bayesian networks, presented in the previous lecture, are one type of graphical models.
- Bayesian networks are mostly used for diagnostic, such as in medicine and in business analytics.
- In this lecture, we focus on another important class of graphical models known as Markov Networks (a.k.a Markov Random Fields).
- Markov Networks are extensively used in computer vision and image processing.

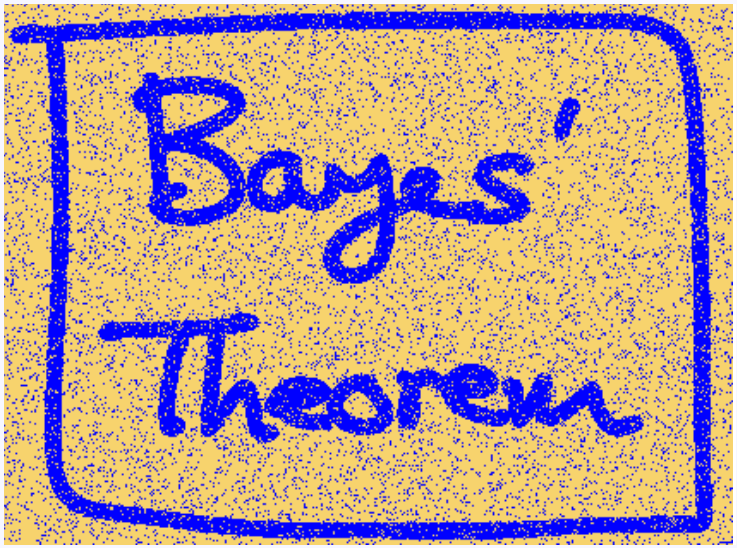
Andrey Markov
1856-1922

Application 1: Image denoising

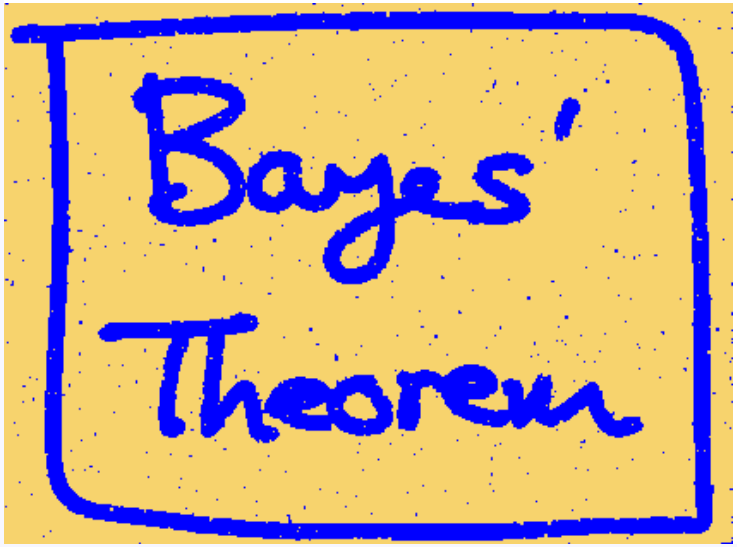
from Christopher Burger, Christian Schuler and Stefan Harmeling. CVPR 2012.

Application 1: Image denoising (from "Pattern Recognition and Machine Learning" by Christopher Bishop)

Original image

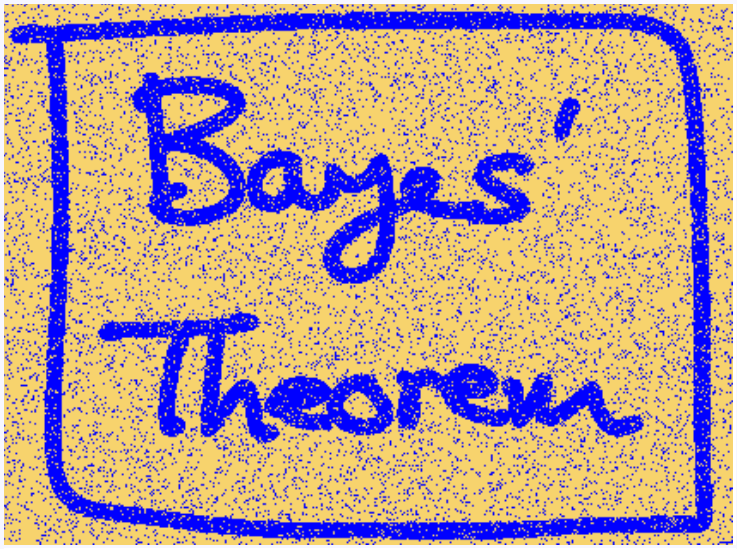


Noisy image

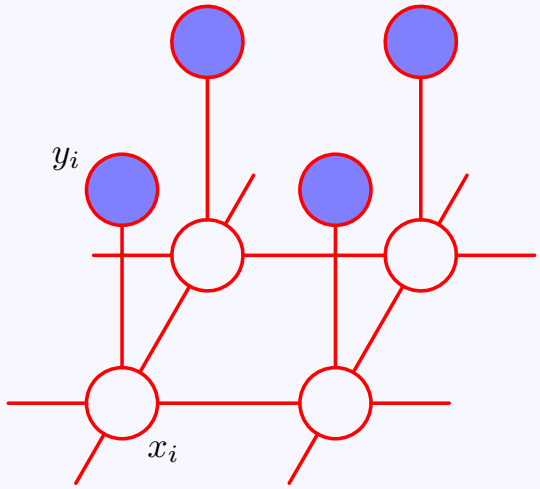


Reconstructed image using a Markov Network

Application 1: Image denoising (from "Pattern Recognition and Machine Learning" by Christopher Bishop)



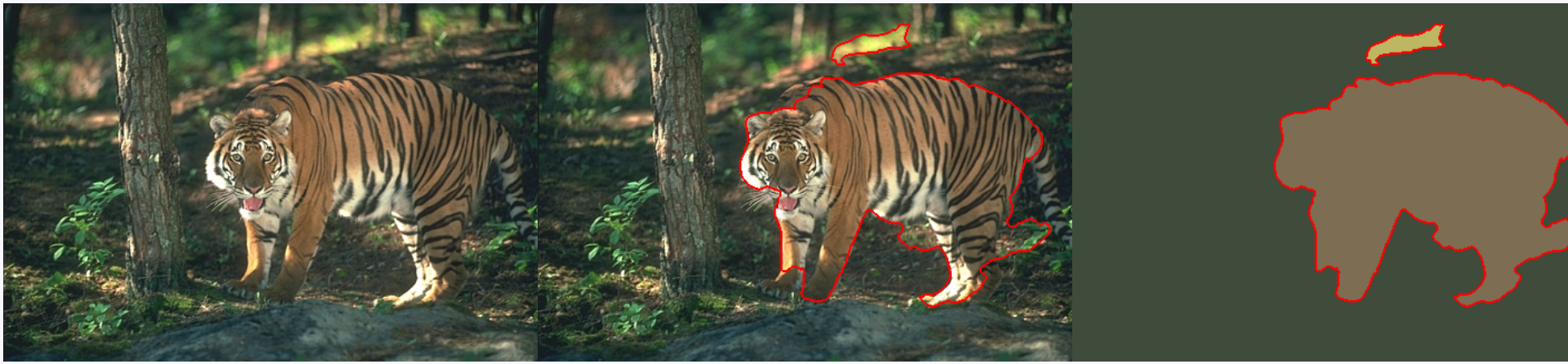
Noisy image



Markov Network

- We construct a graph (network), where each node corresponds to a pixel in the image.
- Nodes are binary random variables (e.g. blue or yellow).
- Nodes X describe the true color of each pixel.
- Nodes Y describe the observed color of each pixel.
- Adjacent pixels correspond to adjacent nodes in the graph.
- Inference problem: Find the *Maximum A Posteriori* (MAP) $\operatorname{argmax}_X P(X \mid Y)$.

Application 2: Background/Foreground separation



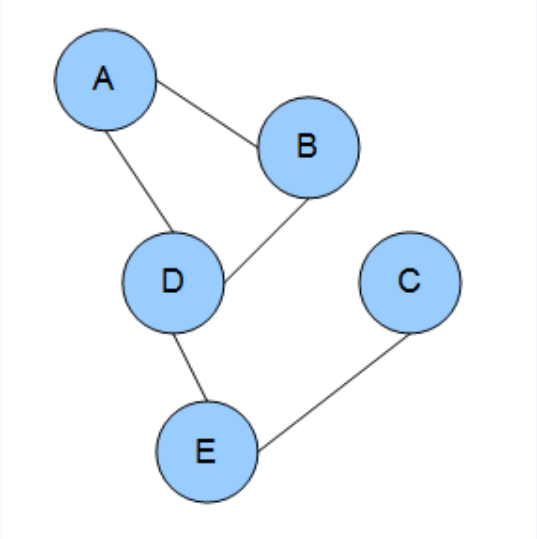
Segmentation of Multivariate Mixed Data via Lossy Coding and Compression. Yi Ma, Harm Derksen, Wei Hong and John Wright. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007.

Application 3: detecting graspable parts of objects from depth images

Markov Networks

- Markov networks are useful for applications where relations between variables are symmetrical.
- Bayesian networks represent causal relations, e.g. $\boxed{Smoking} \rightarrow \boxed{Cancer}$.
- Markov networks represent correlations between variables, e.g. $\boxed{\text{Bob is a democrat}} - \boxed{\text{Alice is a democrat}}$ when Bob and Alice are friends, these two variables depend on each other, but neither is the cause of the other.
- A Markov network is a set of random variables having a *Markov property* described by an **undirected graph**.

Markov Networks



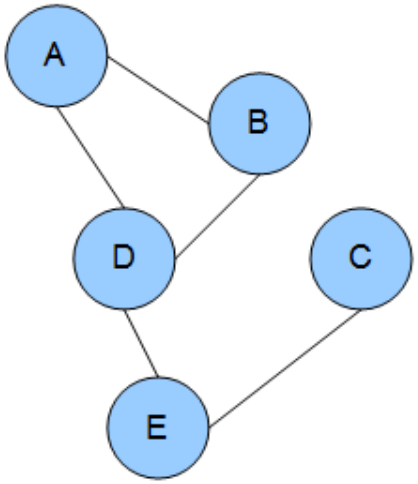
An example of a Markov random field. Each edge represents dependency. In this example: A depends on B and D. B depends on A and D. D depends on A, B, and E. E depends on D and C. C depends on E.

(From Wikipedia)

Complete subgraphs (cliques)

Definition

A *complete subgraph*, or a *clique*, is a subgraph where every two vertices are connected to each other.



Example

- 1-vertex cliques: $C_1 = \{A\}, C_2 = \{B\}, C_3 = \{C\}, C_4 = \{D\}, C_5 = \{E\}$.
- 2-vertex cliques: $C_6 = \{A, B\}, C_7 = \{A, D\}, C_8 = \{B, D\}, C_9 = \{D, E\}, C_{10} = \{E, C\}$.
- 3-vertex cliques: $C_{11} = \{A, B, D\}$.

Factor Potential

Definition

Let $\mathbf{X}$ be a set of random variables. We define a **factor potential** to be a function from values of $\mathbf{X}$ to $\mathbb{R}^+$.

Example

Let $\mathbf{X} = \{BobDemocrat, AliceDemocrat\}$. We define the following factor potential ϕ:

$$\begin{aligned}
\phi([BobDemocrat = true, AliceDemocrat = true]) &= 21 \\
\phi([BobDemocrat = true, AliceDemocrat = false]) &= 2.7 \\
\phi([BobDemocrat = false, AliceDemocrat = true]) &= 3.6 \\
\phi([BobDemocrat = false, AliceDemocrat = false]) &= 18
\end{aligned}$$

Definition of a Markov Network

Definition

Let $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ be a set of random variables and $\mathbf{G}$ a graph that has $\mathbf{X}$ as vertices. Let $\mathbf{C} = \{C_1, C_2, \ldots, C_m\}$ be the set of all the complete subgraphs in $\mathbf{G}$. Let $\phi_1, \phi_2, \ldots, \phi_m$ be factor potentials defined over $C_1, C_2, \ldots, C_m$ respectively.

$\mathbf{X}$ is a Markov network if:

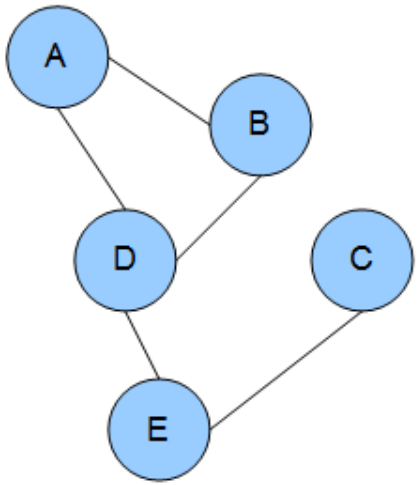
$$P(X_1, X_2, \ldots, X_n) = \frac{1}{Z}\phi_1(C_1) \times \phi_2(C_2) \times \cdots \times \phi_m(C_m).$$

Z is a normalization constant, called *the partition function*, it is defined as

$$Z = \sum_{X_1, X_2, \ldots, X_n} \phi_1(C_1) \times \phi_2(C_2) \times \cdots \times \phi_m(C_m).$$

(each C_i is a subset of $\{X_1, X_2, \ldots, X_n\}$).

Example 1



- 1-vertex cliques: $C_1 = \{A\}, C_2 = \{B\}, C_3 = \{C\}, C_4 = \{D\}, C_5 = \{E\}$.
- 2-vertex cliques: $C_6 = \{A, B\}, C_7 = \{A, D\}, C_8 = \{B, D\}, C_9 = \{D, E\}, C_{10} = \{E, C\}$.
- 3-vertex cliques: $C_{11} = \{A, B, D\}$.

Assume the random variables $\{A, B, C, D, E\}$ are boolean.
Define a factor potential ϕ_i for each clique C_i.
Example: ϕ_6 is defined over the clique $C_6 = \{A, B\}$

$$\begin{aligned} \phi_6([A = true, B = true]) &= 21 \\ \phi_6([A = true, B = false]) &= 2.7 \\ \phi_6([A = false, B = true]) &= 3.6 \\ \phi_6([A = false, B = false]) &= 18 \end{aligned}$$

Example 1

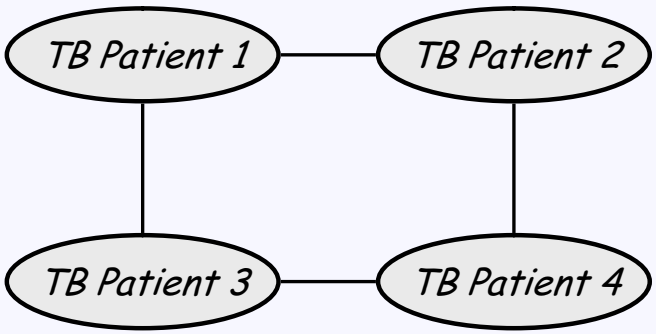
Probability of $(A = true, B = false, C = true, D = true, E = false)$ is given as:

$P(A = true, B = false, C = true, D = true, E = false) = \frac{1}{Z}\phi_1(C_1) \times \phi_2(C_2) \times \phi_3(C_3) \times \phi_4(C_4) \times \phi_5(C_5) \times \phi_6(C_6) \times \phi_7(C_7) \times \phi_8(C_8) \times \phi_9(C_9) \times \phi_{10}(C_{10}) \times \phi_{11}(C_{11})$.

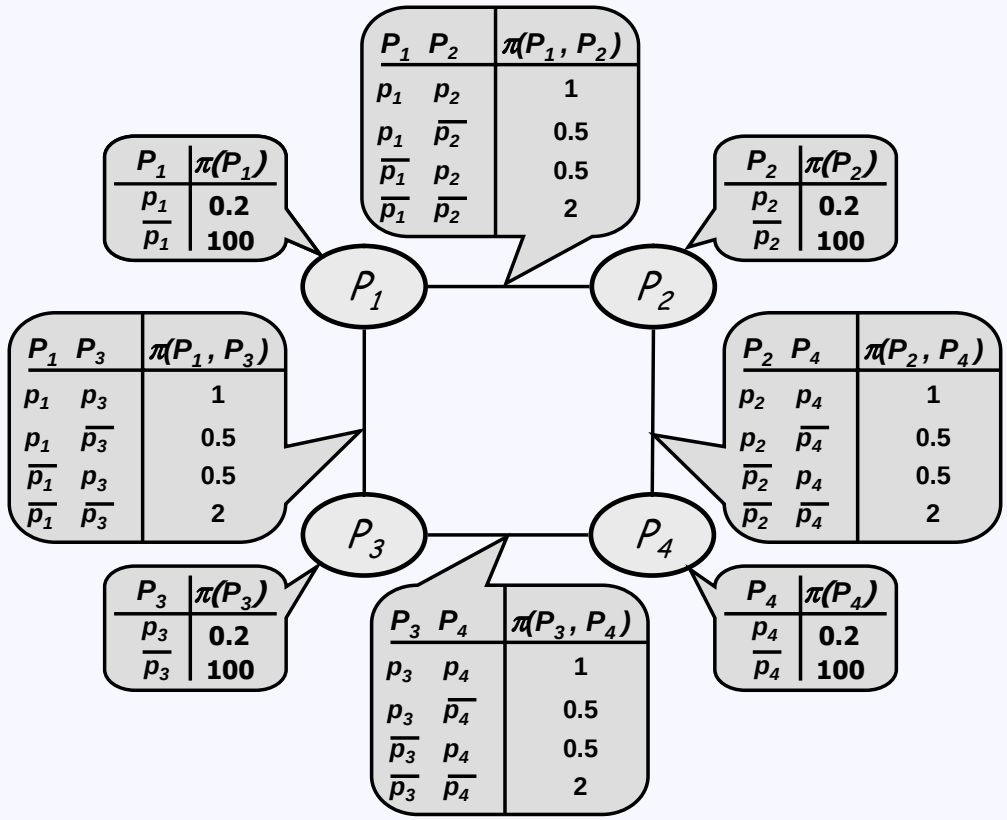
The variables inside each clique C_i take their corresponding values from $(A = true, B = false, C = true, D = true, E = false)$.

$$Z = \sum_{(A,B,C,D,E) \in \{true, false\}^5} \prod_{i=0}^{m} \phi_i(C_i)$$

Example 2



Graph of random variables



Tables of factor potentials.

Potentials of 1-vertex cliques are denoted by $\pi(.)$ and potentials of 2-vertex cliques are denoted by $\pi(.,.)$. Random variable P_i can be interpreted as patient having TB. Two variables are linked if the corresponding patients had a physical contact with each other.

Example taken from Daphne Koller, Nir Friedman, Lise Getoor and Ben Taskar. Graphical Models in a Nutshell

Independence properties in Markov networks

Let $\mathbf{X}$ be a Markov network.
We use the notation $(X \perp Y) \mid Z$ to indicate that X is independent of Y given Z, i.e. $P(X, Y \mid Z) = P(X \mid Z)P(Y \mid Z)$, in other terms $P(X \mid Y, Z) = P(X \mid Z)$.

Independence properties in Markov networks

Let $\mathbf{X}$ be a Markov network.
We use the notation $(X \perp Y) \mid Z$ to indicate that X is independent of Y given Z, i.e. $P(X, Y \mid Z) = P(X \mid Z)P(Y \mid Z)$, in other terms $P(X \mid Y, Z) = P(X \mid Z)$.

- We show here some nice independence properties in Markov networks.
- Independence properties are useful for fast inference, we don't have to enumerate all the possible combinations of values of all the variables.

Independence properties in Markov networks

Let $\mathbf{X}$ be a Markov network.
We use the notation $(X \perp Y) \mid Z$ to indicate that X is independent of Y given Z, i.e. $P(X, Y \mid Z) = P(X \mid Z)P(Y \mid Z)$, in other terms $P(X \mid Y, Z) = P(X \mid Z)$.

- We show here some nice independence properties in Markov networks.
- Independence properties are useful for fast inference, we don't have to enumerate all the possible combinations of values of all the variables.

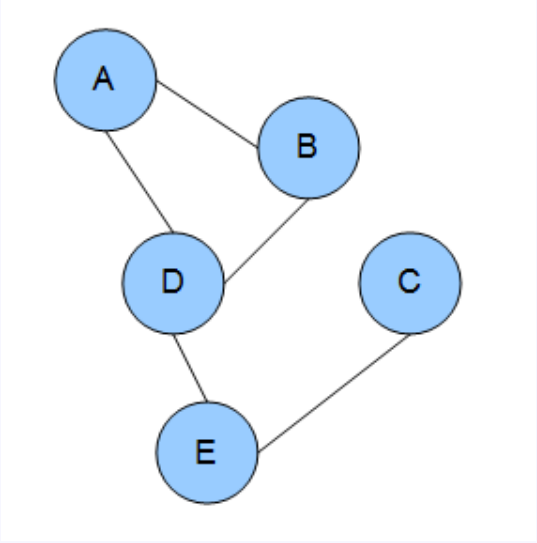
Pairwise Markov property

Any two **non-adjacent** variables, X_i and X_j, are conditionally independent of each other given all other variables $\mathbf{X} - \{X_i, X_j\}$

$$(X_i \perp X_j) \mid \mathbf{X} - \{X_i, X_j\}.$$

In other terms, $P(X_i \mid X_j, \mathbf{X} - \{X_i, X_j\}) = P(X_i \mid \mathbf{X} - \{X_i, X_j\})$.

Example



A and E are independent of each other, given $B, C,$ and D.

Independence properties in Markov networks

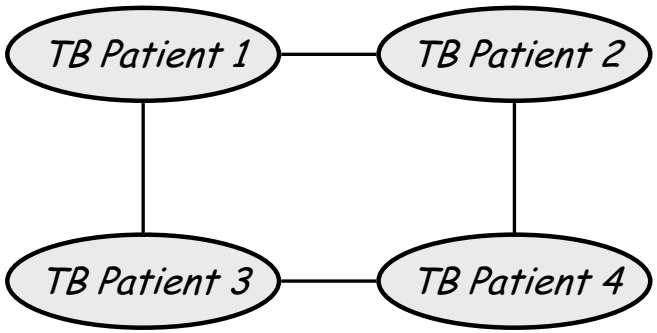
Local Markov property

A variable is conditionally independent of all other variables given **its neighbors**

$$\big(X_i \perp \mathbf{X} - \{X_i, neighbors(X_i)\}\big) \mid neighbors(X_i).$$

Independence properties in Markov networks

Example



This Markov Network describes the following local Markov assumptions:

- $(P_1 \perp P_4 \mid P_2, P_3)$,
- $(P_2 \perp P_3 \mid P_1, P_4)$,

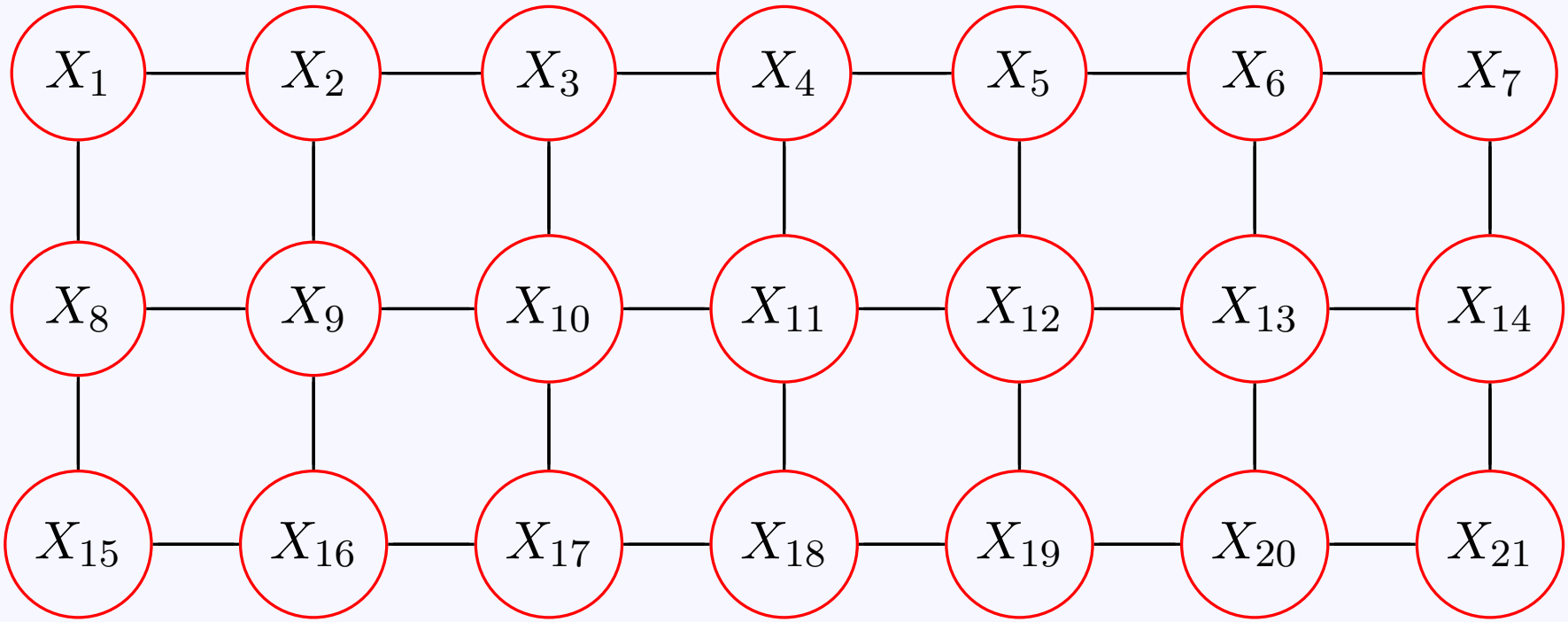
Global Markov property

Any two subsets of variables, $\mathbf{X}_A \subset \mathbf{X}$ and $\mathbf{X}_B \subset \mathbf{X}$, are conditionally independent given a separating subset $\mathbf{X}_S \subset \mathbf{X}$:

$$(\mathbf{X}_A \perp \mathbf{X}_B) \mid \mathbf{X}_S.$$

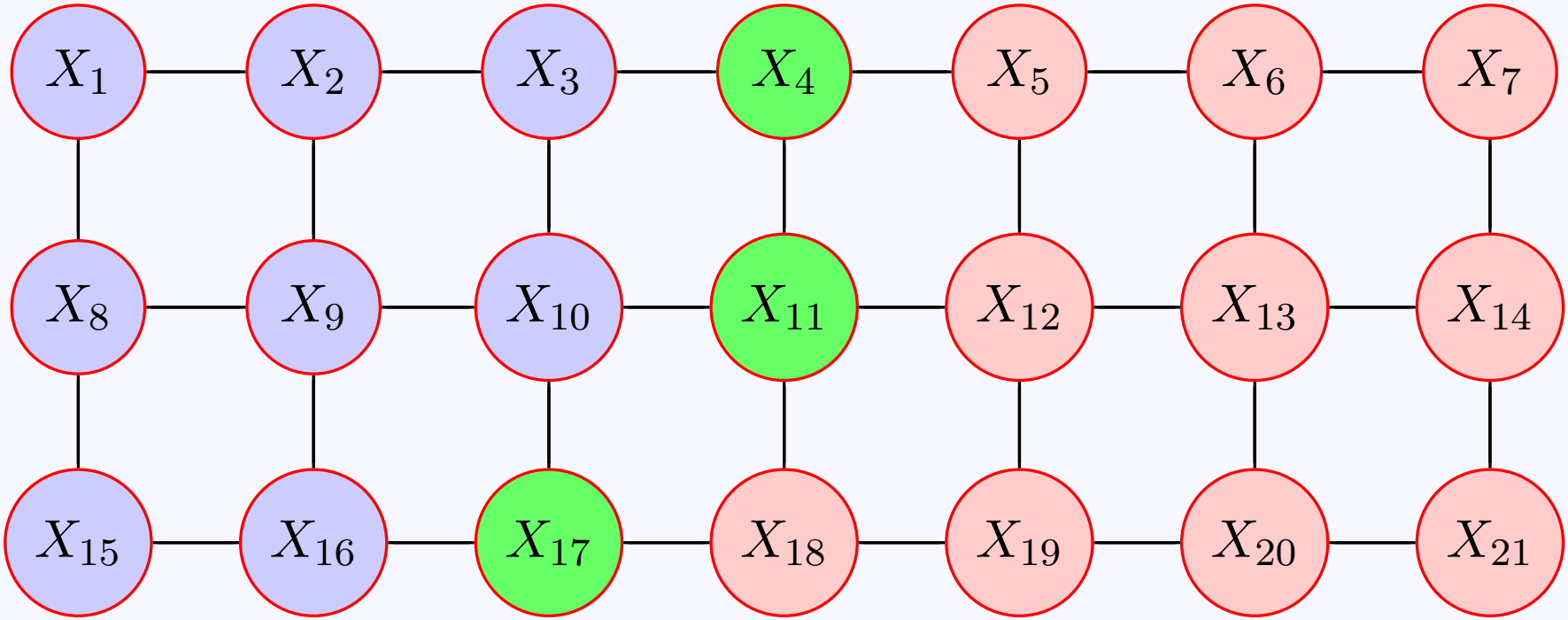
where every path from a node in $\mathbf{X}_A$ to a node in $\mathbf{X}_B$ passes through $\mathbf{X}_S$.

Example



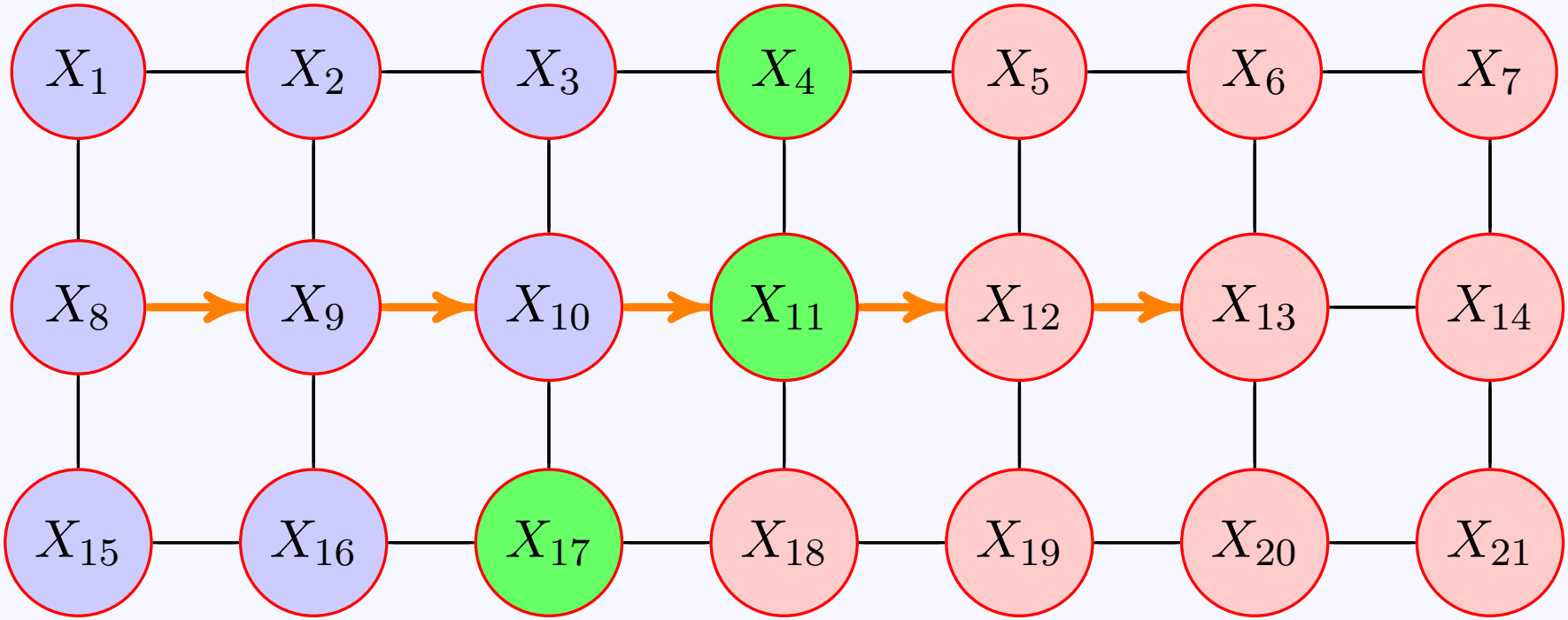
A Markov Network

Example



Set $\mathbf{X}_S = \{X_4, X_{11}, X_{17}\}$ separates sets
$\mathbf{X}_A = \{X_1, X_2, X_3, X_8, X_9, X_{10}, X_{15}, X_{16}\}$ and
$\mathbf{X}_B = \{X_5, X_6, X_7, X_{12}, X_{13}, X_{14}, X_{18}, X_{19}, X_{20}, X_{21}\}$

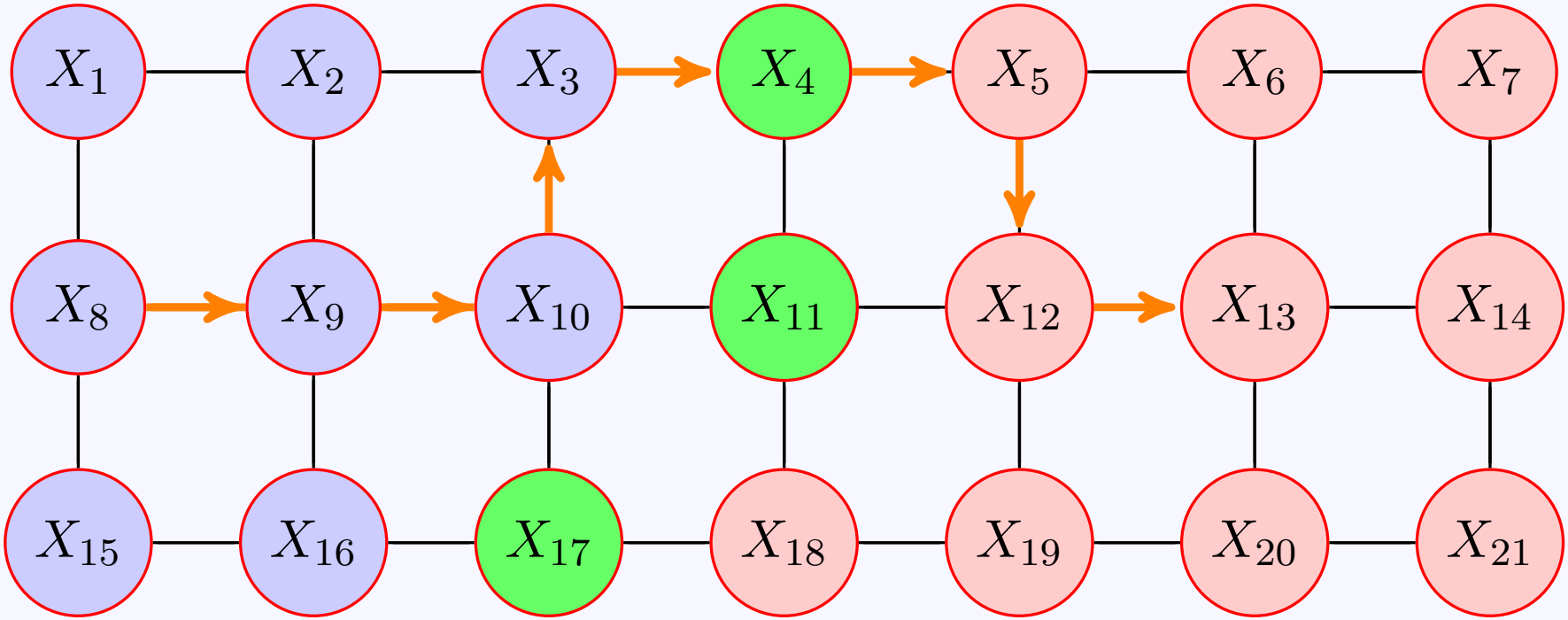
Example



Set $\mathbf{X}_S = \{X_4, X_{11}, X_{17}\}$ separates sets
$\mathbf{X}_A = \{X_1, X_2, X_3, X_8, X_9, X_{10}, X_{15}, X_{16}\}$ and
$\mathbf{X}_B = \{X_5, X_6, X_7, X_{12}, X_{13}, X_{14}, X_{18}, X_{19}, X_{20}, X_{21}\}$

Example: any path between X_8 and X_{13} should pass through X_4, X_{11} or X_{17}.

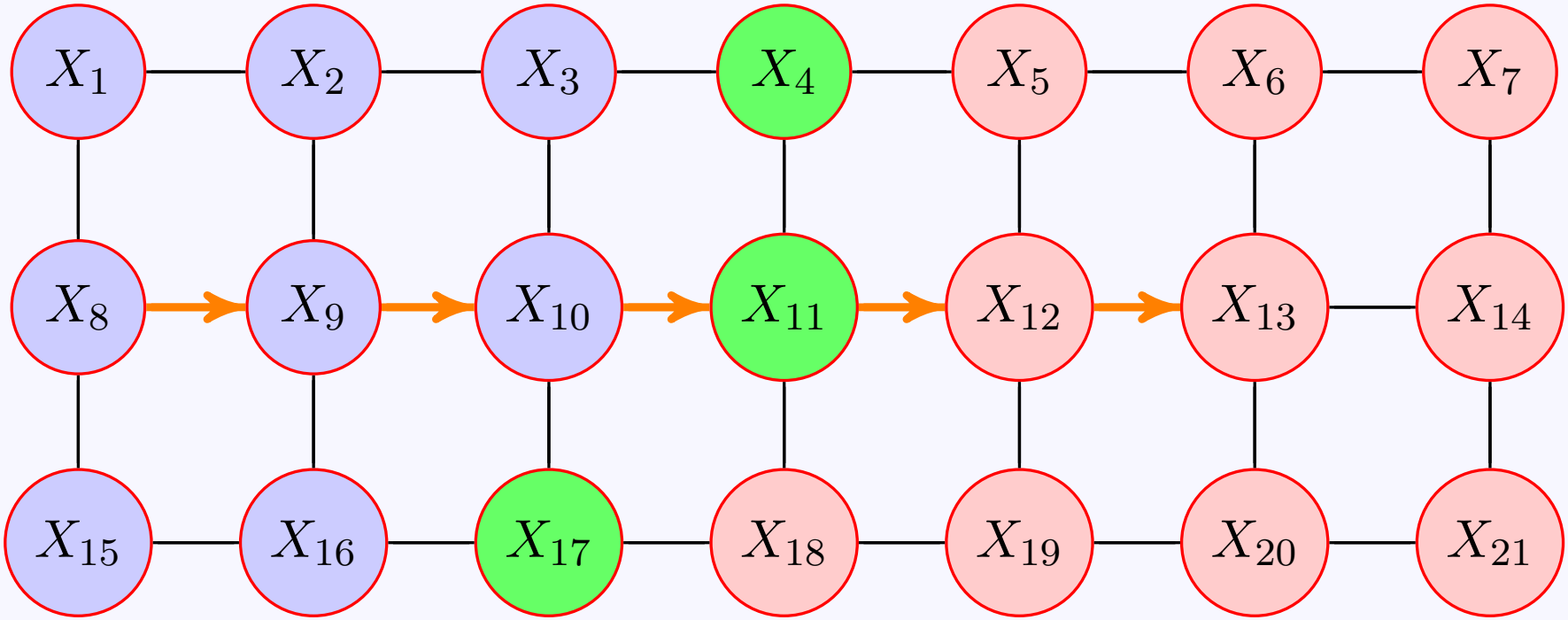
Example



Set $\mathbf{X}_S = \{X_4, X_{11}, X_{17}\}$ separates sets
$\mathbf{X}_A = \{X_1, X_2, X_3, X_8, X_9, X_{10}, X_{15}, X_{16}\}$ and
$\mathbf{X}_B = \{X_5, X_6, X_7, X_{12}, X_{13}, X_{14}, X_{18}, X_{19}, X_{20}, X_{21}\}$

Example: any path between X_8 and X_{13} should pass through X_4, X_{11} or X_{17}.

Example



Set $\mathbf{X}_S = \{X_4, X_{11}, X_{17}\}$ separates sets
$\mathbf{X}_A = \{X_1, X_2, X_3, X_8, X_9, X_{10}, X_{15}, X_{16}\}$ and
$\mathbf{X}_B = \{X_5, X_6, X_7, X_{12}, X_{13}, X_{14}, X_{18}, X_{19}, X_{20}, X_{21}\}$

Example: $P(X_8 \mid X_{13}, X_4, X_{11}, X_{17}) = P(X_8 \mid X_4, X_{11}, X_{17})$, and
$P(X_{13} \mid X_8, X_4, X_{11}, X_{17}) = P(X_{13} \mid X_4, X_{11}, X_{17})$.

Hammersley-Clifford theorem

Theorem

Let $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ *be a set of random variables and* $\mathbf{G}$ *a graph that has* $\mathbf{X}$ *as vertices. If* $\mathbf{X}$ *satisfies the global Markov independence property, then*

$$P(X_1, X_2, \ldots, X_n) = \frac{1}{Z}\phi_1(C_1) \times \phi_2(C_2) \times \cdots \times \phi_m(C_m).$$

where $\mathbf{C} = \{C_1, C_2, \ldots, C_m\}$ *is the set of all the complete subgraphs in* $\mathbf{G}$*, and* $\phi_1, \phi_2, \ldots, \phi_m$ *are factor potentials defined over* $C_1, C_2, \ldots, C_m$ *respectively. In other terms,* $\mathbf{X}$ *is a Markov network.*

Hammersley-Clifford theorem

Theorem

Let $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ be a set of random variables and $\mathbf{G}$ a graph that has $\mathbf{X}$ as vertices. If $\mathbf{X}$ satisfies the global Markov independence property, then

$$P(X_1, X_2, \ldots, X_n) = \frac{1}{Z}\phi_1(C_1) \times \phi_2(C_2) \times \cdots \times \phi_m(C_m).$$

where $\mathbf{C} = \{C_1, C_2, \ldots, C_m\}$ is the set of all the complete subgraphs in $\mathbf{G}$, and $\phi_1, \phi_2, \ldots, \phi_m$ are factor potentials defined over $C_1, C_2, \ldots, C_m$ respectively. In other terms, $\mathbf{X}$ is a Markov network.

Global Markov independence $\equiv$ Factorization into potentials

What exactly are the factor potentials $\phi_i(C_i)$?

- In the previous examples, the factor potentials $\phi_i(C_i)$ look only at the values of the variables contained in clique C_i. For example:

$$\begin{aligned}
\phi([\text{Patient 1 has TB} = true]) &= 30 \\
\phi([\text{Patient 1 has TB} = false]) &= 3 \\
\phi([\text{Patient 1 has TB} = true, \text{Patient 2 has TB} = true]) &= 21 \\
\phi([\text{Patient 1 has TB} = true, \text{Patient 2 has TB} = false]) &= 2.7
\end{aligned}$$

- Clearly, we cannot just use the same $\phi([\text{Patient 1 has TB} = true])$ for every patient x. Every patient is different.
- Also, we cannot write down $\phi([\text{Patient x has TB} = true])$ for every possible patient x, there are infinitely many patients.

What exactly are the factor potentials $\phi_i(C_i)$?

- In the previous examples, the factor potentials $\phi_i(C_i)$ look only at the values of the variables contained in clique C_i. For example:

$$\begin{aligned}
\phi([\text{Patient 1 has TB} = true]) &= 30 \\
\phi([\text{Patient 1 has TB} = false]) &= 3 \\
\phi([\text{Patient 1 has TB} = true, \text{Patient 2 has TB} = true]) &= 21 \\
\phi([\text{Patient 1 has TB} = true, \text{Patient 2 has TB} = false]) &= 2.7
\end{aligned}$$

- Clearly, we cannot just use the same $\phi([\text{Patient 1 has TB} = true])$ for every patient x. Every patient is different.
- Also, we cannot write down $\phi([\text{Patient x has TB} = true])$ for every possible patient x, there are infinitely many patients.

- We include **side information**, or **features**, regarding each variable and each clique. For example, the medical checkup of Patient x.
- $\phi([\text{Patient x has TB} = true])$ is not directly a function of Patient x, but it is a function of the features of Patient x.

Clique features

- Each clique C_i can be described by a vector of features f_i.
- Example 1: Clique $C_1 = \{$ Patient 1$\}$ is described by a vector of features f_1:

$$\begin{cases} f_1[0] & = \text{Age of Patient 1} \\ f_1[1] & = \text{Blood pressure of Patient 1} \\ f_1[2] & = \text{Body temperature of Patient 1} \end{cases}$$

- Example 1: Clique $C_2 = \{$ Patient 1, Patient 2$\}$ is described by a vector of features f_2:

$$\begin{cases} f_2[0] & = \text{Type of interaction between Patients 1 and 2} \\ f_2[1] & = \text{Duration of interaction between Patients 1 and 2} \end{cases}$$

Logistic model

Typically, we use a log-linear model with feature functions f_i to represent factor potentials ϕ_i:

$$\phi_i(C_i) = \exp\Big(\sum_{j=0}^{k} w_{C_i}[j] f_i[j]\Big),$$

Where $f_i[j]$ is the j^{th} feature of the clique C_i, and $w_{C_i}[j]$ is its corresponding weight (a value in $\mathbb{R}$).
w_{C_i}, like $\phi_i(C_i)$, depends on the values taken by the variables in clique C_i.

Logistic model

Typically, we use a log-linear model with feature functions f_i to represent factor potentials ϕ_i:

$$\phi_i(C_i) = \exp\Big(\sum_{j=0}^{k} w_i[j] f_i[j]\Big),$$

Where $f_i[j]$ si the j^{th} feature of the clique C_i, and $w_i[j]$ is its corresponding weight (a value in $\mathbb{R}$).
w_{C_i}, like $\phi_i(C_i)$, depends on the values taken by the variables in clique C_i.

Therefore,

$$P(X_1, X_2, \ldots, X_n) = \frac{1}{Z}\prod_{i=0}^{m}\phi_i(C_i) = \frac{1}{Z}\exp\Big(\sum_{i=0}^{m}\sum_{j=0}^{k} w_{C_i}[j] f_i[j]\Big)$$

Associative Markov Networks

- **Associative Markov Networks** are a popular variant of Markov Networks that have been successfully used in computer vision. They are a special variant of **Pairwise Markov Networks**.
- In **Pairwise Markov Networks**, there are two types of factor potentials,
  - potentials ϕ_{node} associated with individual variables, and
  - potentials ϕ_{edge} associated with edges (links between variables).
- A logistic model is used to represent the potentials:

$$\begin{cases} \phi_{node}(X_i) & = \exp\left(\sum_k sign(X_i) w_{node}[k] f_i[k]\right) \\ \phi_{edge}(X_i, X_j) & = \exp\left(\sum_k w_{edge}[k] f_{(i,j)}[k]\right) \end{cases}$$

$sign(X_i) = +1$ if $X_i = true$ and $sign(X_i) = -1$ if $X_i = false$

Associative Markov Networks

- **Associative Markov Networks** are a popular variant of Markov Networks that have been successfully used in computer vision. They are a special variant of **Pairwise Markov Networks**.
- In **Pairwise Markov Networks**, there are two types of factor potentials,
  - potentials ϕ_{node} associated with individual variables, and
  - potentials ϕ_{edge} associated with edges (links between variables).
- A logistic model is used to represent the potentials:

$$\begin{cases} \phi_{node}(X_i) & = \exp\left(\sum_k sign(X_i) w_{node}[k] f_i[k]\right) \\ \phi_{edge}(X_i, X_j) & = \exp\left(\sum_k w_{edge}[k] f_{(i,j)}[k]\right) \end{cases}$$

$sign(X_i) = +1$ if $X_i = true$ and $sign(X_i) = -1$ if $X_i = false$

The name *logistic model* comes from the fact that $\log \phi(X_i) = \sum_k sign(X_i) w[k] f_i[k]$

Associative Markov Networks

In Associative Markov Networks, edge potentials are defined as

$$\begin{cases} \phi_{edge}(X_i, X_j) & = \exp\left(\sum_k w_{edge}[k] f_{(i,j)}[k]\right) & \text{if } X_i = X_j. \\ \phi_{edge}(X_i, X_j) & = \exp\left(0\right) = 1 & \text{if } X_i \neq X_j. \end{cases}$$

Associative Markov Networks

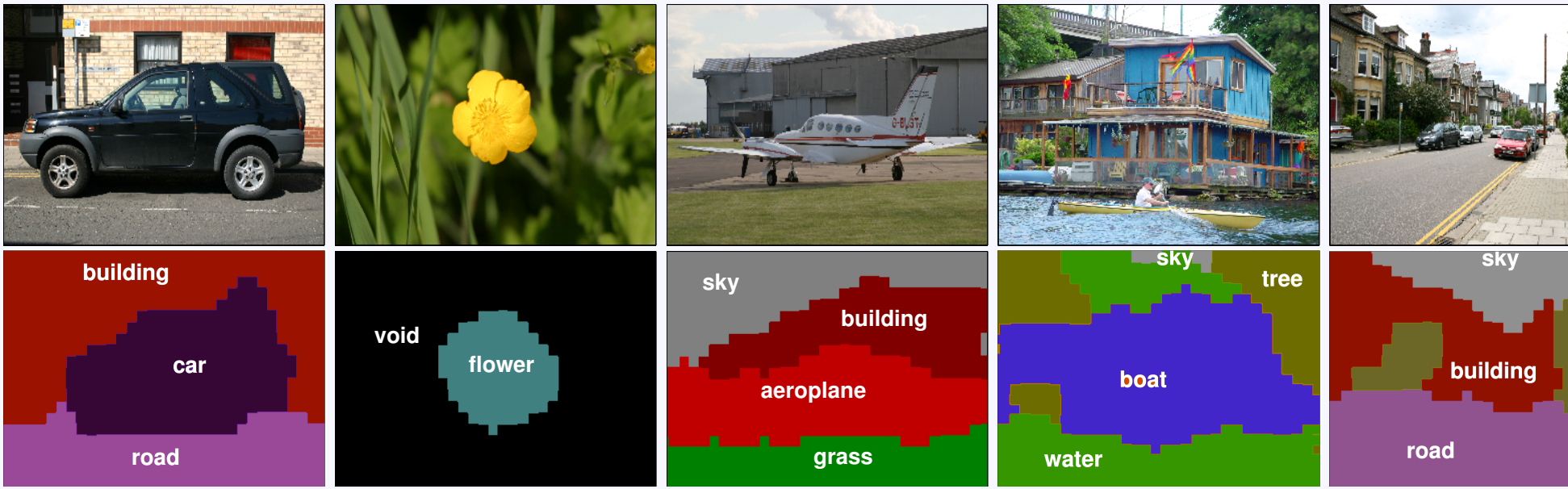
In Associative Markov Networks, edge potentials are defined as

$$\begin{cases} \phi_{edge}(X_i, X_j) & = \exp\left(\sum_k w_{edge}[k] f_{(i,j)}[k]\right) & \text{if } X_i = X_j. \\ \phi_{edge}(X_i, X_j) & = \exp\left(0\right) = 1 & \text{if } X_i \neq X_j. \end{cases}$$

The joint probability distribution is given as

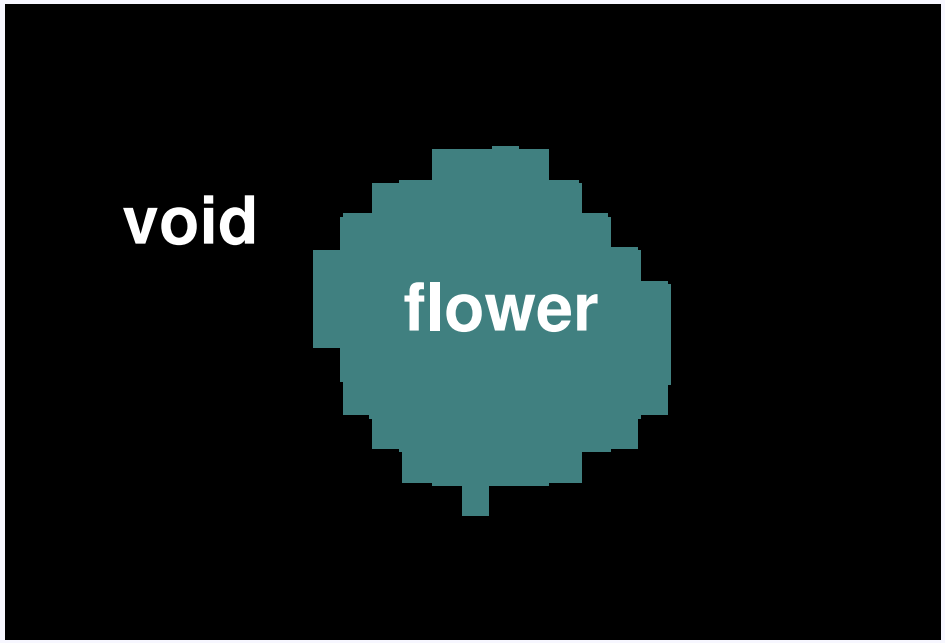
$$P(X) = \frac{1}{Z} \exp\Bigg(\sum_{X_i} \sum_{k} sign(X_i) w_{node}[k] f_i[k] + \sum_{\substack{(X_i, X_j) \\ \text{s.t. } X_i = X_j \text{ and} \\ X_i, X_j \text{ are neighbors}}} \sum_{k} w_{edge}[k] f_{(i,j)}[k]\Bigg).$$

Application of Associative Markov Networks: Image segmentation



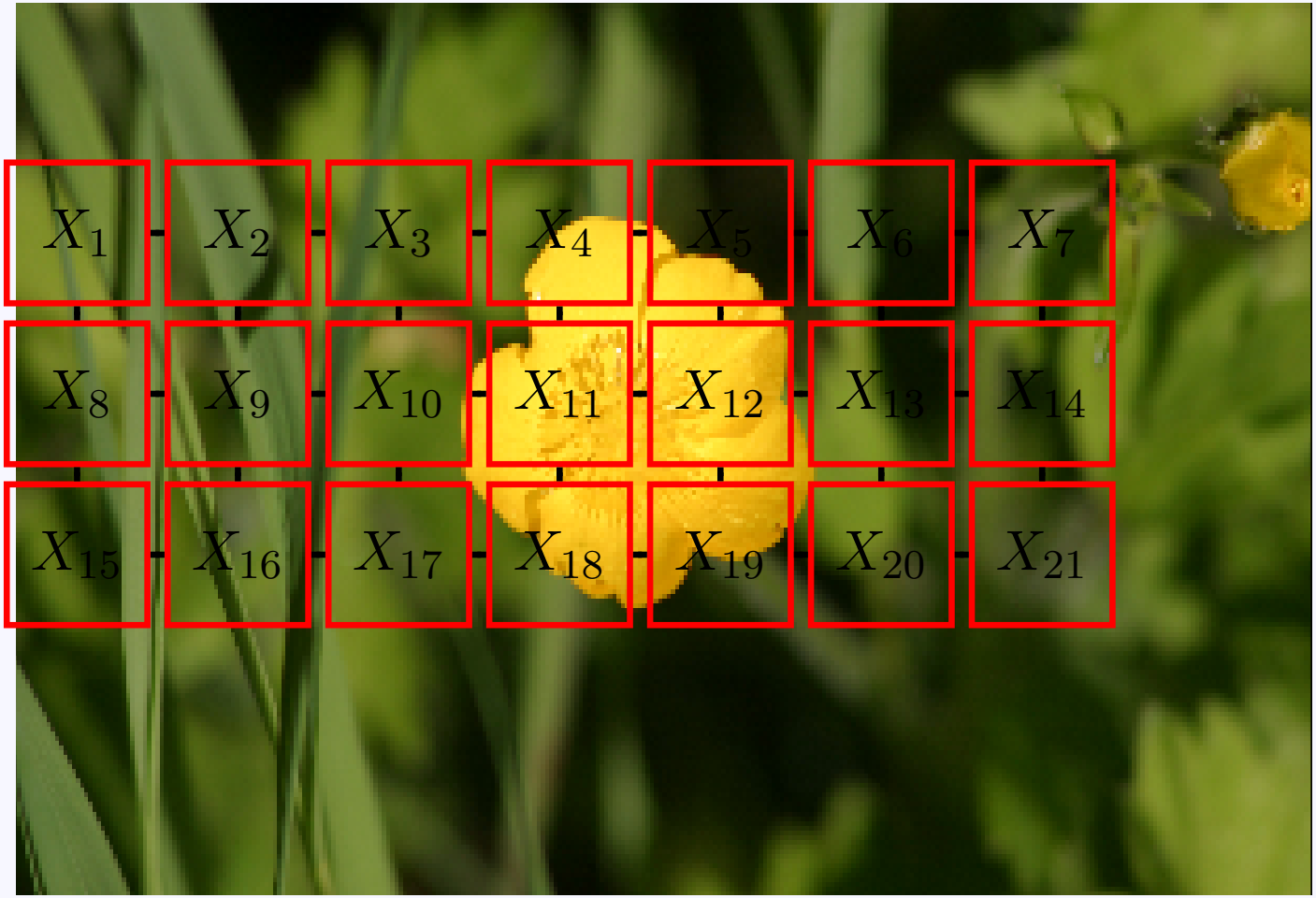
Jakob Verbeek, Bill Triggs. Region Classification with Markov Field Aspect Models. In CVPR 2007.

Application of Associative Markov Networks: Image segmentation



Jakob Verbeek, Bill Triggs. Region Classification with Markov Field Aspect Models. In CVPR 2007.

Application of Associative Markov Networks: Image segmentation



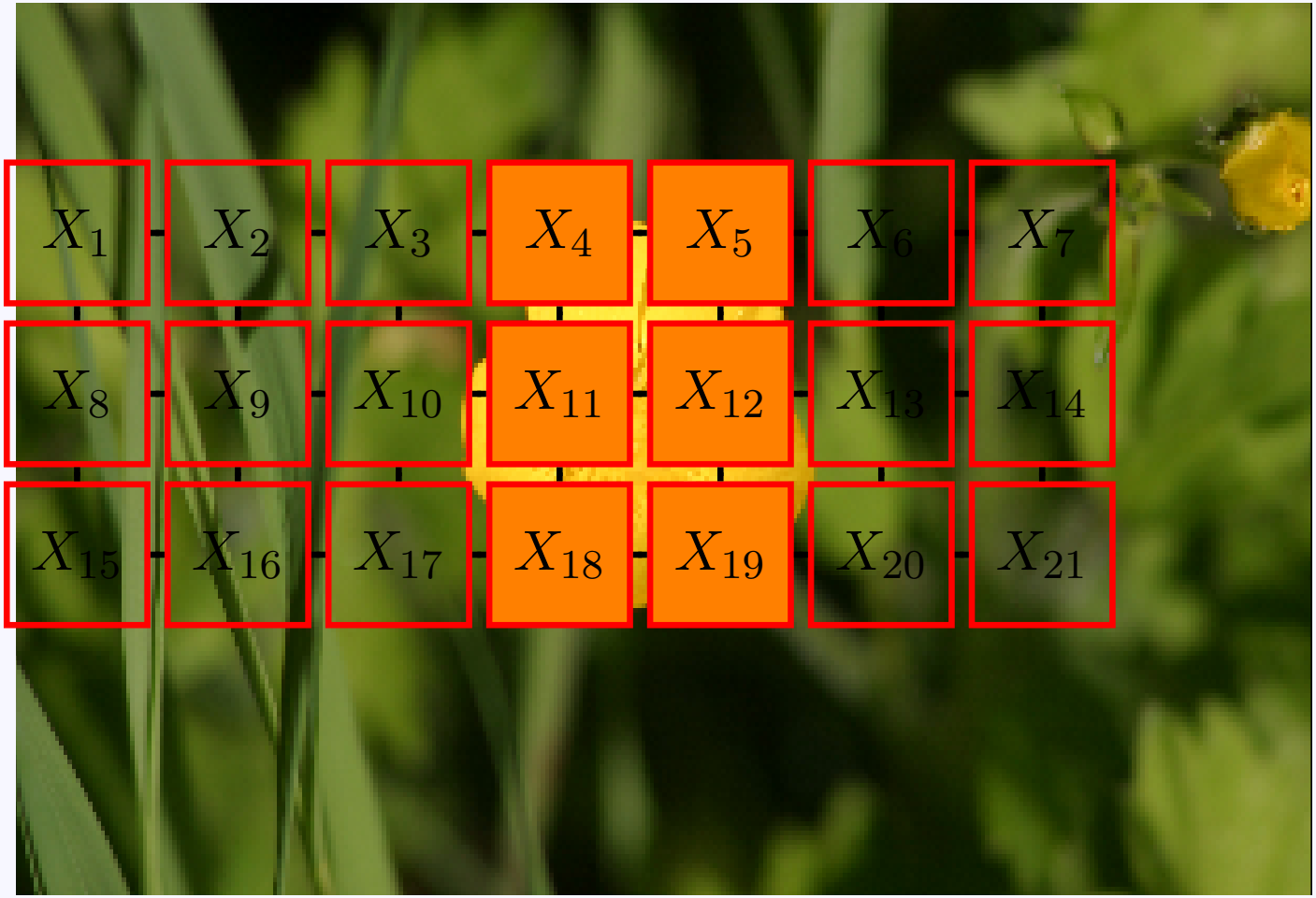
Learning Associative Markov Networks

- Each variable X_i in the network corresponds to a small patch in the image.
- $X_i = true$ means patch i in the image corresponds to a flower.
- $X_i = false$ means patch i in the image does not correspond to a flower.
- We represent the joint probability distribution over all possible values of variables X_i as

$$P(X) = \frac{1}{Z} \exp\Big(\sum_{X_i} \sum_{k} sign(X_i) w_{node}[k] f_i[k] + \sum_{\substack{(X_i, X_j) \\ \text{s.t. } X_i = X_j \text{ and} \\ X_i, X_j \text{ are neighbors}}} \sum_{k} w_{edge}[k] f_{(i,j)}[k] \Big).$$

- The instance that has the highest probability should be when variables $X_4, X_5, X_{11}, X_{12}, X_{18}$ and X_{19} are all true while all the remaining variables are false.

Application of Associative Markov Networks: Image segmentation



Application of Associative Markov Networks: Image segmentation

Edge features

We can use a unique constant feature for the edges (links):
$f_{(i,j)}[0] = 1, \forall (i,j)$ s.t X_i is neighbor of X_j

Node features

- We can use the *Histograms of Oriented Gradients* (HOG) features to represent the patch corresponding to each variable.
- $f_i[k]$ counts the number of occurrences of image gradient orientation $2\pi k/N$ in the portion i of the image (where N is the maximum number or orientations considered).
- We can also add RGB colours as features.

Learning Associative Markov Networks

Now that we know what the feature vectors f_i and $f_{i,j}$ are, how can we find their weight vectors w_{node} and w_{edge}?

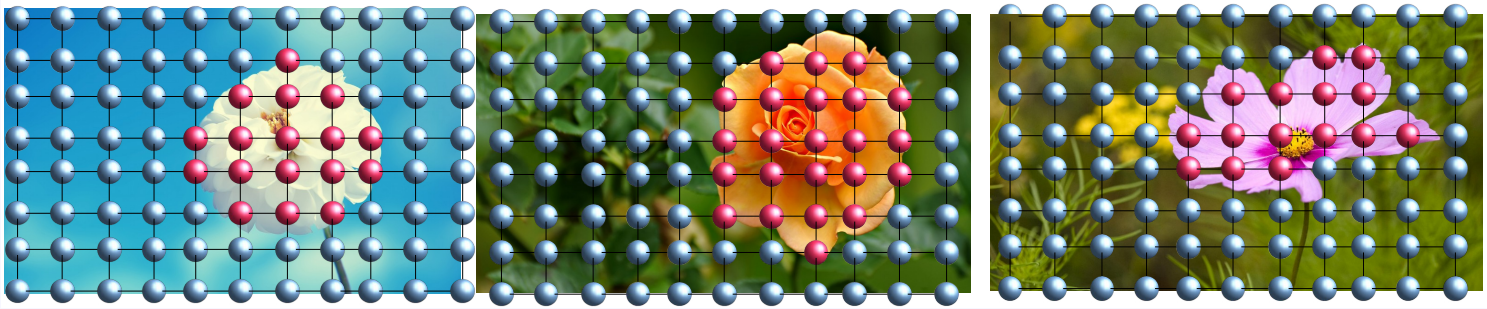
- We can collect a lot of annotated examples, and find weights w_{node} and w_{edge} that maximize the likelihood.
- The likelihood function is concave and can be maximized using a gradient ascent.

Learning Associative Markov Networks

To learn weights w_{node} and w_{edge}, we start by collecting examples of images of a particular object (e.g, flower).

Examples of images containing a flower

For each image, we create an Associative Markov Network and label its nodes as **true** (object detected), or **false**.



Red nodes are variables set to true, blue nodes are variables set to true.

Maximum Likelihood

Let X^{ex} be an assignment of values to the variables (nodes) in one of the examples. We have

$$L_{X^{\text{ex}}}(w) = \frac{1}{Z_w} \exp\Big(\sum_{X_i \in X^{\text{ex}}} \sum_k sign(X_i) w_{node}[k] f_i[k] + \sum_{\substack{(X_i, X_j) \\ \text{s.t. } X_i = X_j \text{ in } X^{\text{ex}}, \\ X_i, X_j \text{ are neighbors}}} \sum_k w_{edge}[k] f_{(i,j)}[k] \Big).$$

where $w = [w_{node}, w_{edge}]$.

- $L_{X^{\text{ex}}}(w)$ is the **likelihood** of w given X^{ex}.
- $L_{X^{\text{ex}}}(w)$ is a function of both w and X^{ex}. But here, X^{ex} is known from the annotated example.
- We want to find w that maximizes $L_{X^{\text{ex}}}(w)$.
- We need to compute
  $\nabla_w L_{X^{\text{ex}}}(w) = \left(\frac{\partial L_{X^{\text{ex}}}(w)}{\partial w_{node}[0]}, \frac{\partial L_{X^{\text{ex}}}(w)}{\partial w_{node}[1]}, \ldots, \frac{\partial L_{X^{\text{ex}}}(w)}{\partial w_{edge}[0]}, \frac{\partial L_{X^{\text{ex}}}(w)}{\partial w_{edge}[1]}, \ldots \right).$

Maximum Likelihood

$$L_{X^{\text{ex}}}(w) = \frac{1}{Z_w} \exp\Big(\sum_{X_i \in X^{\text{ex}}} \sum_k sign(X_i) w_{node}[k] f_i[k] + \sum_{\substack{(X_i,X_j) \\ \text{s.t. } X_i=X_j \text{ in } X^{\text{ex}}, \\ X_i,X_j \text{ are neighbors}}} \sum_k w_{edge}[k] f_{(i,j)}[k] \Big).$$

$$\frac{\partial L_{X^{\text{ex}}}}{\partial w_{node}[k]}(w) = L_{X^{\text{ex}}}(w) \Big(\sum_{X_i \in X^{\text{ex}}} sign(X_i) f_i[k] - \sum_X L_X(w) \sum_{X_i \in X} sign(X_i) f_i[k] \Big).$$

where X is an arbitrary assignments of values to the variables.

$$\frac{\partial L_{X^{\text{ex}}}}{\partial w_{edge}[k]}(w) = L_{X^{\text{ex}}}(w) \Big(\sum_{\substack{(X_i,X_j) \\ \text{s.t. } X_i=X_j \text{ in } X^{\text{ex}}, \\ X_i,X_j \text{ are neighbors}}} \sum_k f_{(i,j)}[k] - \sum_X L_X(w) \sum_{\substack{(X_i,X_j) \\ \text{s.t. } X_i=X_j \text{ in } X, \\ X_i,X_j \text{ are neighbors}}} \sum_k f_{(i,j)}[k] \Big).$$

Most Likely Explanation

- After we learn the weights of the model using gradient ascent, we are given a completely new image and we are asked to label the nodes in it into *true* (e.g, flower), or *false*.
- Notice that we do not need to compute the probability of each combination of values, we only need to find the combination that has the maximum propagation.
- For binary variables, this problem can be transformed into **finding a minimum cut in a graph**, and solved in polynomial time.

Inference in Markov Networks

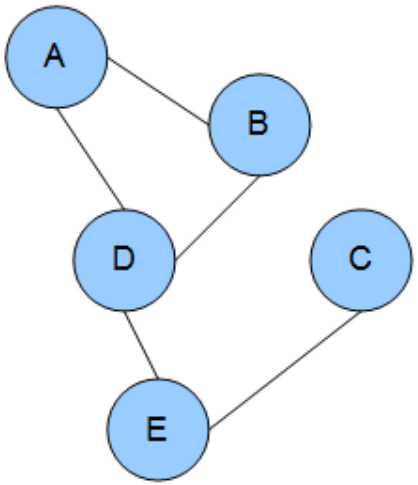
We return now to general Markov Networks, and let's say we want to compute $P(X_i)$ for some variable $X_i \in \mathbf{X}$.

The **belief propagation algorithm**, a.k.a the **sum-product algorithm**, provides an exact answer for tree-structured graphs, and an approximate answer for general graphs (for which the algorithm is known as **loopy belief propagation**).

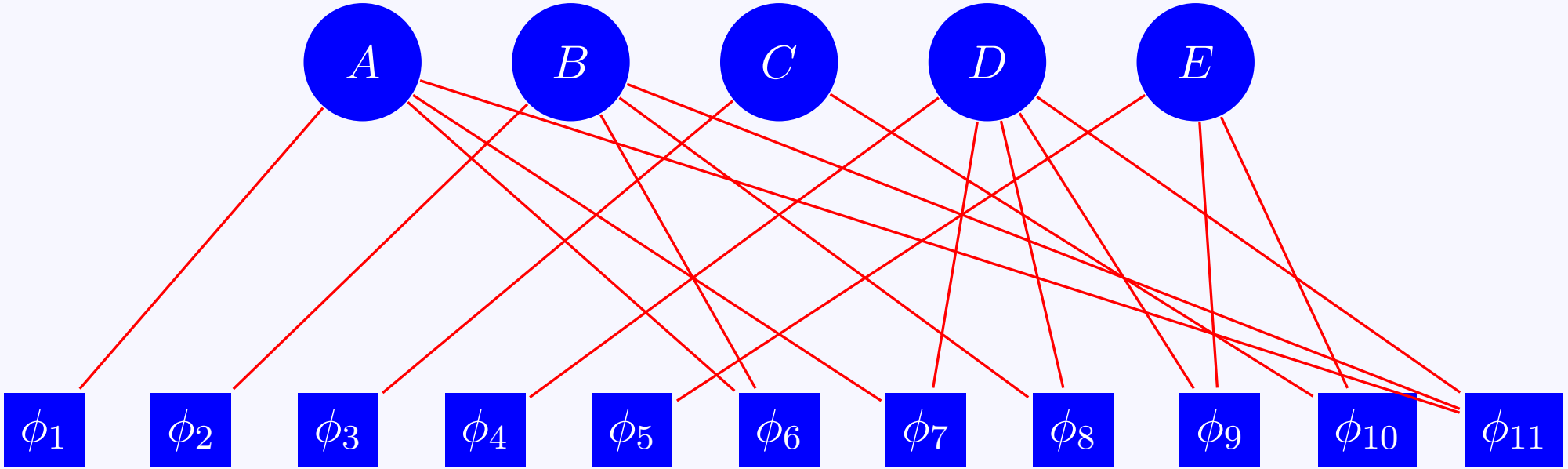
Factor graphs

- We have seen that the joint probability is the product of factors ϕ_i, defined over cliques C_i.
- Each factor involves one or more variables.
- Each variable is involved in one or more factors.

Factor graphs

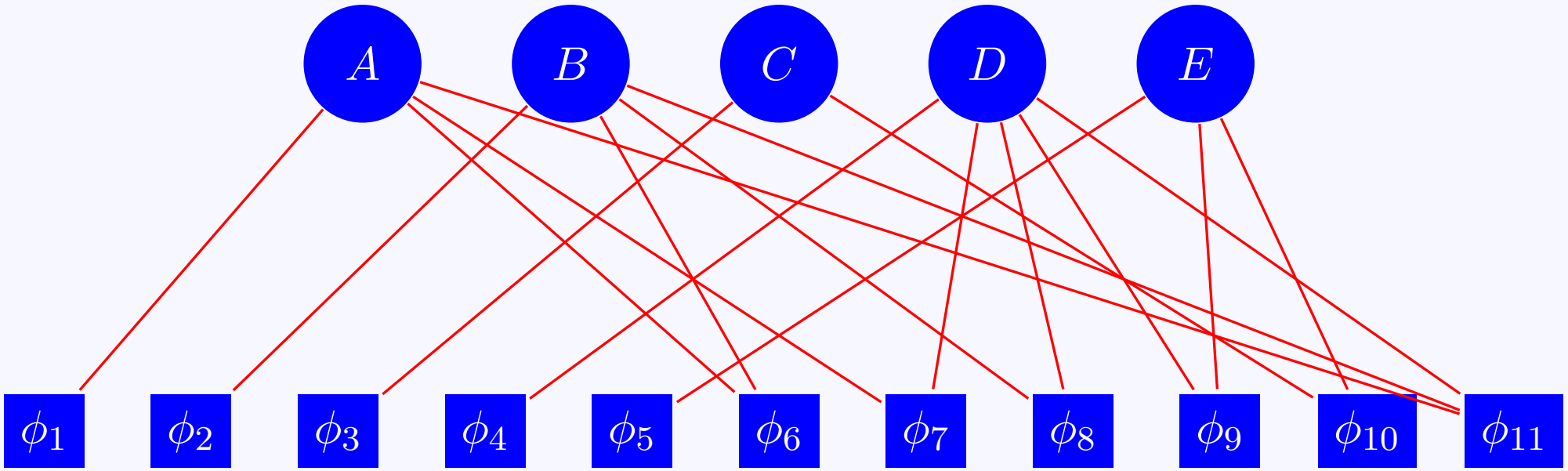


- 1-vertex cliques: $C_1 = \{A\}, C_2 = \{B\}, C_3 = \{C\}, C_4 = \{D\}, C_5 = \{E\}$.
- 2-vertex cliques: $C_6 = \{A, B\}, C_7 = \{A, D\}, C_8 = \{B, D\}, C_9 = \{D, E\}, C_{10} = \{E, C\}$.
- 3-vertex cliques: $C_{11} = \{A, B, D\}$.



Factor graphs

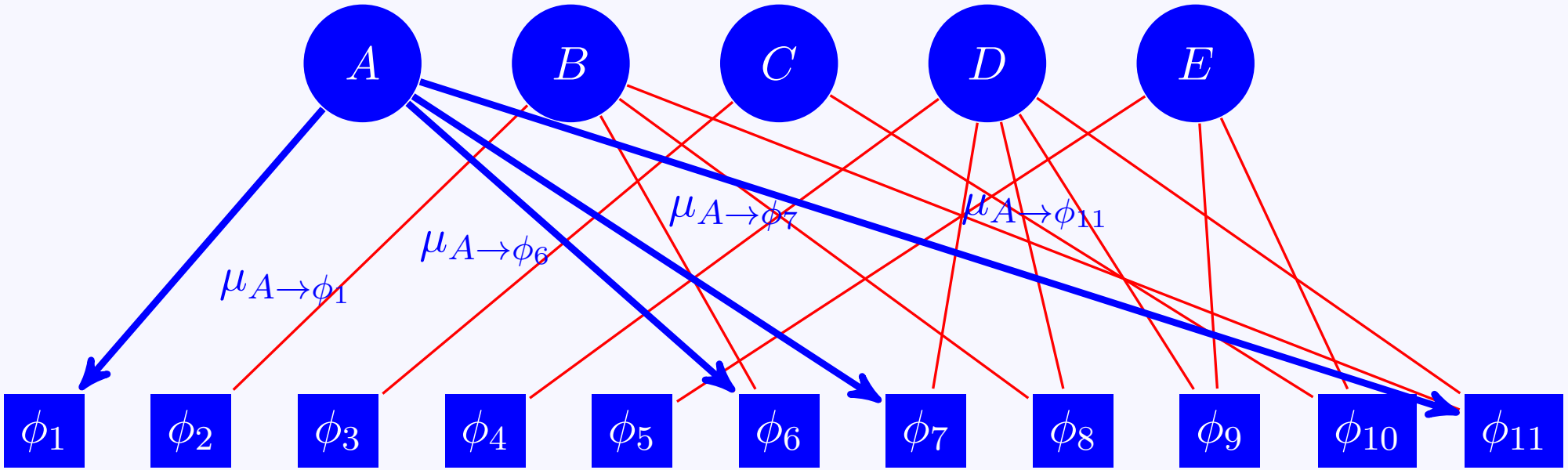
The sum-product algorithm is based on repeatedly passing messages between variables and factors until convergence.



The sum-product algorithm

We have two types of messages:

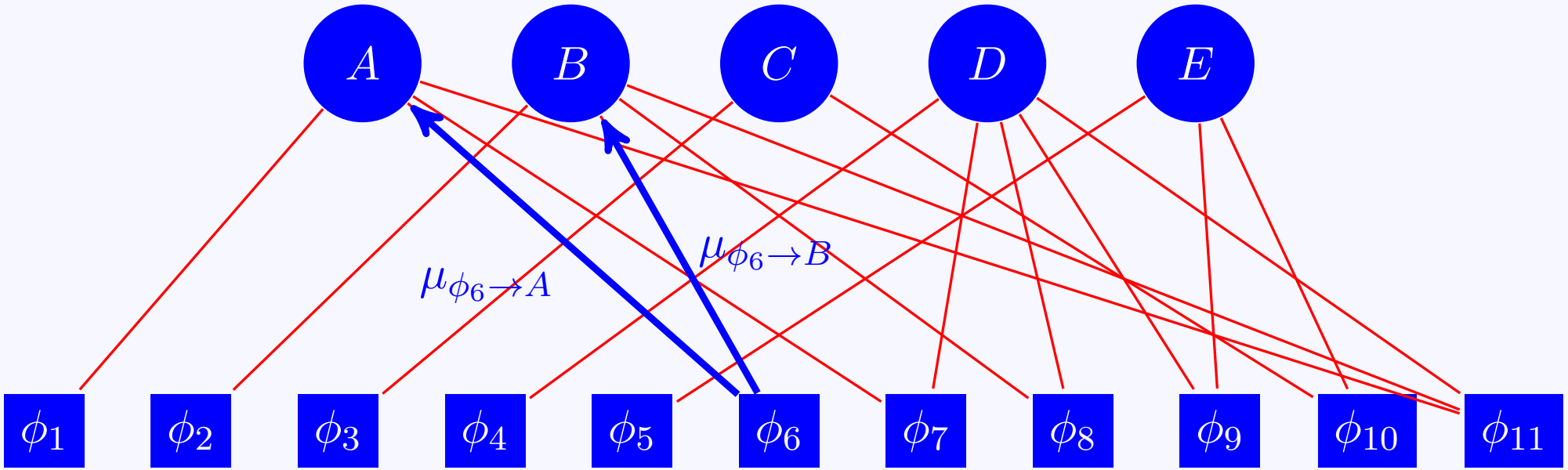
- Message $\mu_{X_i \to \phi}$ from a variable X_i to a factor ϕ
- Message $\mu_{\phi \to X_i}$ from a factor ϕ to a variable X_i.



The sum-product algorithm

We have two types of messages:

- Message $\mu_{X_i \to \phi}$ from a variable X_i to a factor ϕ
- Message $\mu_{\phi \to X_i}$ from a factor ϕ to a variable X_i.



The sum-product algorithm

A message $\mu_{X_i \to \phi}$ from a variable X_i to a factor ϕ is the product of the messages from all other factors involving X_i (except the recipient)

$$\forall x_i \in domain(X_i) : \mu_{X_i \to \phi}(x_i) = \prod_{\phi_j \neq \phi \text{ s.t } X_i \in C_j} \mu_{\phi_j \to X_i}(x_i).$$

where ϕ_j are the factors of all the cliques that contain X_i, except the factor ϕ (the recipient).

The sum-product algorithm

A message $\mu_{\phi \to X_i}$ from factor ϕ to variable X_i is the product of the factor with messages from all other variables, marginalized over all variables except the one associated with X_i

$$\forall x_i \in domain(X_i) : \mu_{\phi \to X_i}(x_i) = \sum_{\mathbf{X} \text{ s.t } X_i = x_i} \phi(\mathbf{X}) \prod_{X_j \in C - \{X_i\}} \mu_{X_j \to \phi}(x_j).$$

where C is the clique of variables associated with ϕ, and $\mathbf{X}$ are joint values of all the variables in clique C. We sum over all the possible values of the variables in the clique (except for X_i which is set to x_i). x_j is the value that variable X_j takes in $\mathbf{X}$.

The sum-product algorithm

The initial messages (at the first iteration) are all equal to 1.

Upon convergence (if convergence happened), the estimated marginal distribution of each node is:

$$P(X_i = x_i) \propto \prod_{\phi_j \text{ s.t } X_i \in C_j} \mu_{\phi_j \to X_i}(x_i)$$

The same algorithm can be used for finding the Maximum A Posteriori (MAP) by replacing the sums by maxima. This algorithm is known as Max-Product.