Probabilistic Graph Models: Belief Propagation

ECE/CS 498 DS U/G

Lecture 18

Ravi K. Iyer

Dept. of Electrical and Computer Engineering

University of Illinois at Urbana Champaign

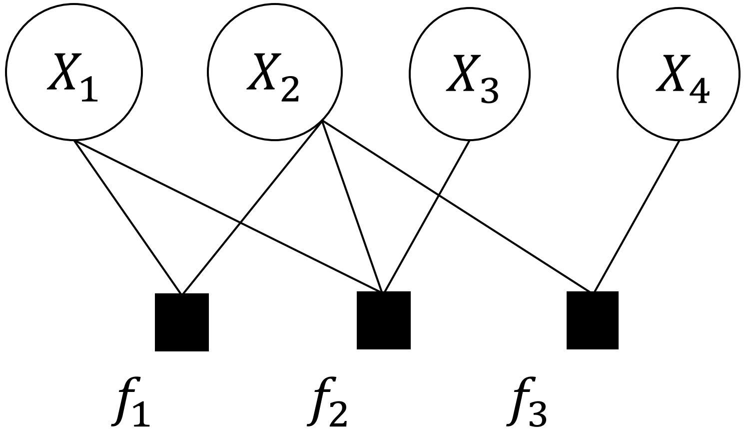
Announcements

- Graduate projects proposal due on Friday, April 5
- No discussion section on Friday, April 5
  - Additional office hours will be held in place of it in CSL 141 from 4-5pm
- Mid course feedback summary
- Schedule of the class moving forward

Week 13	4/8	Belief Propagation continued
	4/10	In-class Activity 5 on PGMs
Week 14	4/15	Supervised Learning (SVM, decision trees, RF)
	4/17	Perceptron Model and Neural Networks
Week 15	4/22	In-class Activity 6 (tentative) on Neural Network + Supervised Learning
	4/24	Intro to Deep Learning Challenges in Deep Learning Compare Deep Neural Nets (DNN) vs. Probabilistic Graphical Models (PGMs) using an Example Dataset
Week 16	4/29	Review problems for Final Exam
	5/2	Reading Day

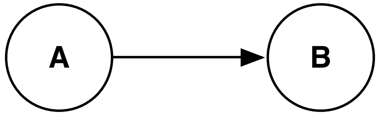
Recap: Definition of a Factor Graph

A factor graph is a **bipartite, undirected graph** of **random variables and factor functions. [Frey et. al. 01].**

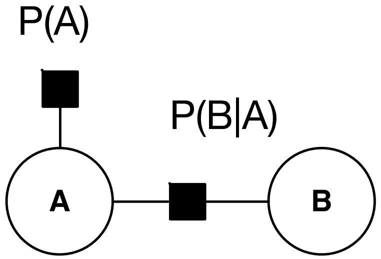


G (graph) = (X,f,E); E denotes the edges

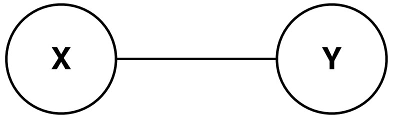
FG can represent both ***causal and non-causal relations****.*



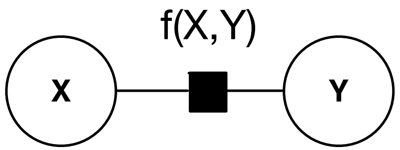
Bayesian Network (BN)



Factor Graph equivalent of BN



Undirected Graph



Factor Graph equivalent of UG

Modeling the credential stealing attack using Factor Graphs - Data

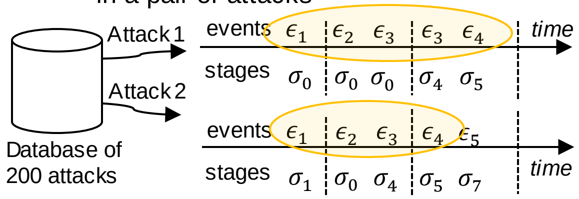
State space of variables

Attack stage: $X = \{\sigma_0, \sigma_1, \dots, \sigma_7\}$

(Observed) Events: $E = \{\epsilon_1, \dots, \epsilon_5\}$

OFFLINE ANNOTATION ON PAST ATTACKS

a) Annotated events and attack stages in a pair of attacks



b) Event-stage annotation table for the attack pair (Attack 1 and Attack 2)

Event	Attack stage
$\{\epsilon_1\}$	$\{\sigma_0 \vert \sigma_1\}$
$\{\epsilon_2\}$	$\{\sigma_0\}$
$\{\epsilon_3\}$	$\{\sigma_4\}$
$\{\epsilon_4\}$	$\{\sigma_5\}$
$\{\epsilon_5\}$	$\{\sigma_7\}$

- Multi-stage credential stealing attack where the attack stage is not observed; however events which are related to the attack stage are observed
- Goal is to detect and pre-empt the attack
- **Model assumptions**
  - There are multivariate relationships among the events
  - There is no restriction on order of the relationships (can be non-causal or correlation based)

- Markov Model and Bayesian Networks cannot be used in this scenarios

- Factor graphs can be used for modeling highly complex attacks, where the causal relations among the events are not immediately clear.

ϵ_1	vulnerability scan	σ_0	benign
ϵ_2	login	σ_1	discovery
ϵ_3	sensitive_uri	σ_4	privilege escalation
ϵ_4	new_library	σ_5	persistence

Modeling the credential stealing attack using Factor Graphs

OFFLINE LEARNING OF FACTOR FUNCTIONS

Example patterns, stages, probabilities, and significance learned from the attack pair

Pattern	Attack stages	Probability in past attacks	Significance (p-value)
$[\epsilon_1, \epsilon_3, \epsilon_4]$	$[\sigma_1, \sigma_4, \sigma_5]$	q_a	p_a
$[\epsilon_1]$	$[\sigma_0 \| \sigma_1]$	q_b	p_b

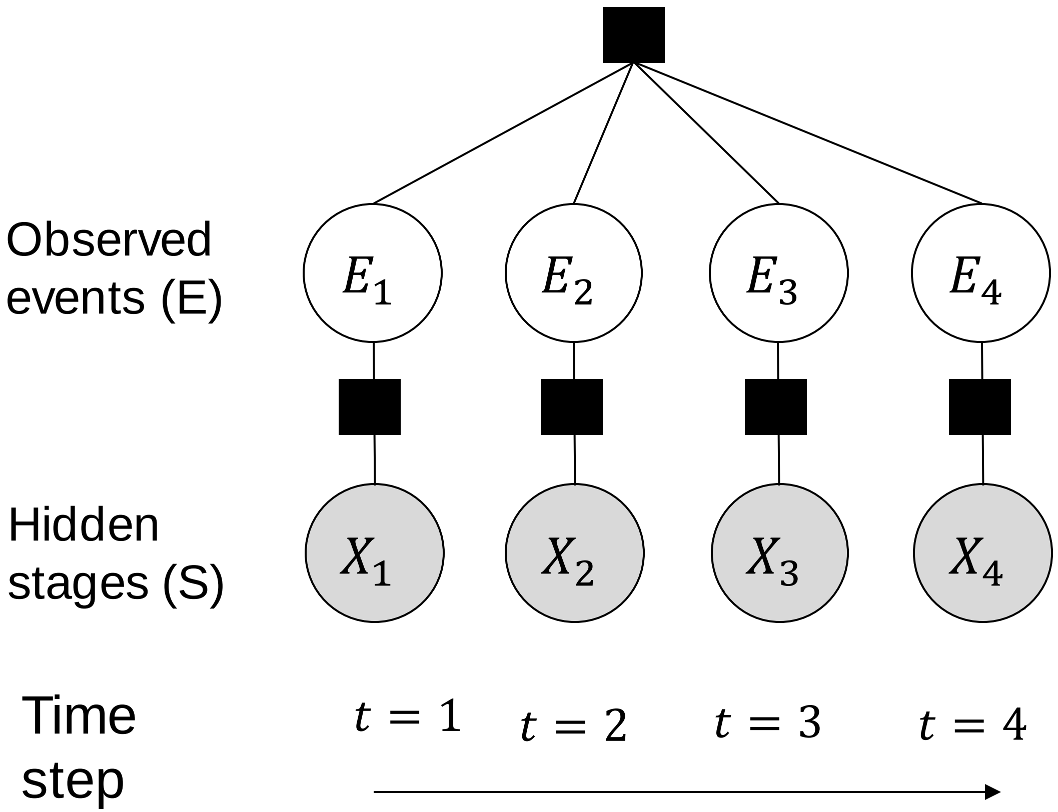
…

$$f(E) = \exp\{q_E(1 - p_E)\}$$

A factor function defined on the learned pattern, stages, and its significance

DETECTION OF UNSEEN ATTACKS

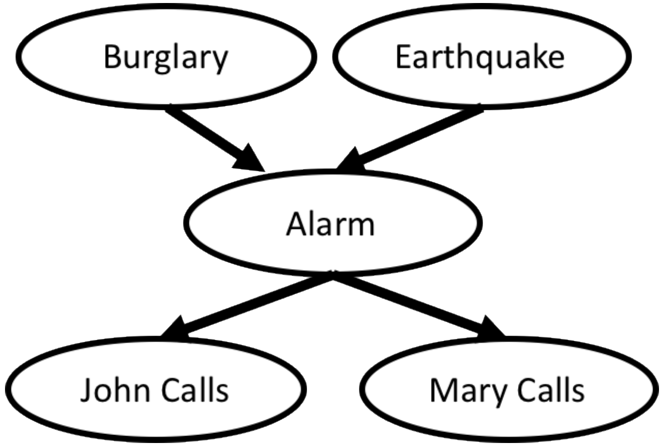
Factor Graph



Inference on Graphical Models

Problems we have already looked at:

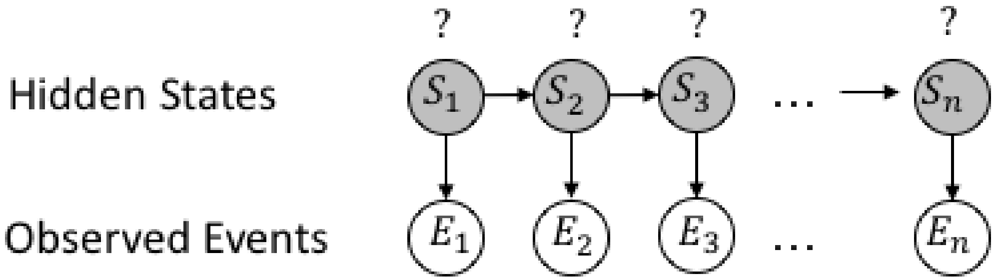
Bayes Network



Calculate the joint probability $P(B,J,A,E,M) = P(J|A)\, P(M|A) P(A|B,E) P(B) P(E)$

Calculate the state probability $P(B) = \sum_{J,A,E,M} P(B,J,A,E,M)$

Hidden Markov Model



Calculate the conditional distribution

$P(S_t|E_1, \dots, E_n)$

Factorize (Bayes Theorem)

$\propto P(S_t|E_1, \dots, E_t) * P(E_{t+1}, \dots, E_n|S_t, E_1, \dots, E_t)$

Use the Markov Property

$= P(S_t|E_1, \dots, E_t) * P(E_{t+1}, \dots, E_n|S_t)$

Forward Backward Algorithm

$= \alpha_t \odot \beta_t$

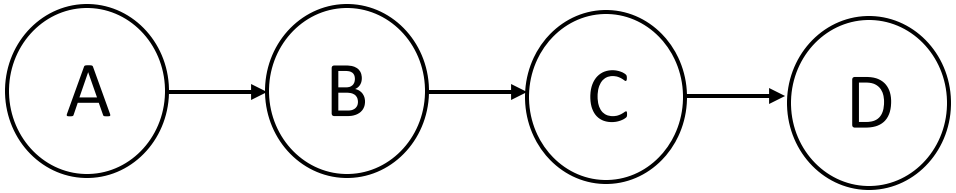
Just involves computation of joint distributions and its marginalization

Example of inference on a Bayesian Network

Consider the following Bayesian Network

- $A \in \{a^1, a^2\}, B \in \{b^1, b^2\}, C \in \{c^1, c^2\}, D \in \{d^1, d^2\}$

Inference task: Compute P(D)



$$P(D) = \sum_{A,B,C} P(A,B,C,D) = \sum_{A,B,C} P(A)P(B|A)P(C|B)P(D|C)$$

- Simple way would be to generate each possible sequence (A,B,C,D) and sum over them
  - Exponential in the number of variables

Example of inference on Bayesian Network

- Enumerating all combinations
- Each term has 3 multiplications
- 8+8 = 16 terms
- Total multiplication ops = 16x3 = 48

- 7 additions for d^1 and 7 additions for d^2
- Total additions ops = 7+7=14

$$
\begin{array}{rllll}
 & P(a^1) & P(b^1 \mid a^1) & P(c^1 \mid b^1) & P(d^1 \mid c^1) \\
+ & P(a^2) & P(b^1 \mid a^2) & P(c^1 \mid b^1) & P(d^1 \mid c^1) \\
+ & P(a^1) & P(b^2 \mid a^1) & P(c^1 \mid b^2) & P(d^1 \mid c^1) \\
+ & P(a^2) & P(b^2 \mid a^2) & P(c^1 \mid b^2) & P(d^1 \mid c^1) \\
+ & P(a^1) & P(b^1 \mid a^1) & P(c^2 \mid b^1) & P(d^1 \mid c^2) \\
+ & P(a^2) & P(b^1 \mid a^2) & P(c^2 \mid b^1) & P(d^1 \mid c^2) \\
+ & P(a^1) & P(b^2 \mid a^1) & P(c^2 \mid b^2) & P(d^1 \mid c^2) \\
+ & P(a^2) & P(b^2 \mid a^2) & P(c^2 \mid b^2) & P(d^1 \mid c^2)
\end{array}
$$

$$
\begin{array}{rllll}
 & P(a^1) & P(b^1 \mid a^1) & P(c^1 \mid b^1) & P(d^2 \mid c^1) \\
+ & P(a^2) & P(b^1 \mid a^2) & P(c^1 \mid b^1) & P(d^2 \mid c^1) \\
+ & P(a^1) & P(b^2 \mid a^1) & P(c^1 \mid b^2) & P(d^2 \mid c^1) \\
+ & P(a^2) & P(b^2 \mid a^2) & P(c^1 \mid b^2) & P(d^2 \mid c^1) \\
+ & P(a^1) & P(b^1 \mid a^1) & P(c^2 \mid b^1) & P(d^2 \mid c^2) \\
+ & P(a^2) & P(b^1 \mid a^2) & P(c^2 \mid b^1) & P(d^2 \mid c^2) \\
+ & P(a^1) & P(b^2 \mid a^1) & P(c^2 \mid b^2) & P(d^2 \mid c^2) \\
+ & P(a^2) & P(b^2 \mid a^2) & P(c^2 \mid b^2) & P(d^2 \mid c^2)
\end{array}
$$

All terms involved in computation of $P(d^1)$ and $P(d^2)$ respectively.

Example of inference on Bayesian Network

Can we reduce the number of computations?

- Many terms are common; they can be computed once and reused

Consider the orange highlighted box, $P(c^1|b^1)P(d^1|c^1)$ is common.

Compute: $P(a^1)P(b^1|a^1) + P(a^2)P(b^1|a^2)$

Consider the blue highlighted box, $P(c^1|b^2)P(d^1|c^1)$ is common.

Compute: $P(a^1)P(b^2|a^1) + P(a^2)P(b^2|a^2)$

Define: $\tau_1(B) = P(a^1)P(B|a^1) + P(a^2)P(B|a^2)$

where $B \in \{b^1, b^2\}$

$$
\begin{array}{lllll}
 & P(a^1) & P(b^1 \mid a^1) & P(c^1 \mid b^1) & P(d^1 \mid c^1) \\
+ & P(a^2) & P(b^1 \mid a^2) & P(c^1 \mid b^1) & P(d^1 \mid c^1) \\
+ & P(a^1) & P(b^2 \mid a^1) & P(c^1 \mid b^2) & P(d^1 \mid c^1) \\
+ & P(a^2) & P(b^2 \mid a^2) & P(c^1 \mid b^2) & P(d^1 \mid c^1) \\
+ & P(a^1) & P(b^1 \mid a^1) & P(c^2 \mid b^1) & P(d^1 \mid c^2) \\
+ & P(a^2) & P(b^1 \mid a^2) & P(c^2 \mid b^1) & P(d^1 \mid c^2) \\
+ & P(a^1) & P(b^2 \mid a^1) & P(c^2 \mid b^2) & P(d^1 \mid c^2) \\
+ & P(a^2) & P(b^2 \mid a^2) & P(c^2 \mid b^2) & P(d^1 \mid c^2)
\end{array}
$$

$$
\begin{array}{lllll}
 & P(a^1) & P(b^1 \mid a^1) & P(c^1 \mid b^1) & P(d^2 \mid c^1) \\
+ & P(a^2) & P(b^1 \mid a^2) & P(c^1 \mid b^1) & P(d^2 \mid c^1) \\
+ & P(a^1) & P(b^2 \mid a^1) & P(c^1 \mid b^2) & P(d^2 \mid c^1) \\
+ & P(a^2) & P(b^2 \mid a^2) & P(c^1 \mid b^2) & P(d^2 \mid c^1) \\
+ & P(a^1) & P(b^1 \mid a^1) & P(c^2 \mid b^1) & P(d^2 \mid c^2) \\
+ & P(a^2) & P(b^1 \mid a^2) & P(c^2 \mid b^1) & P(d^2 \mid c^2) \\
+ & P(a^1) & P(b^2 \mid a^1) & P(c^2 \mid b^2) & P(d^2 \mid c^2) \\
+ & P(a^2) & P(b^2 \mid a^2) & P(c^2 \mid b^2) & P(d^2 \mid c^2)
\end{array}
$$

All terms involved in computation of $P(d^1)$ and $P(d^2)$ respectively.

Example of inference on Bayesian Network

Consider the orange highlighted box, $P(d^1|c^1)$ is common.

Compute: $\tau_1(b^1)P(c^1|b^1) \ + \tau_1(b^2)P(c^1|b^2)$

Consider the blue highlighted box, $P(d^1|c^2)$ is common.

Compute: $\tau_1(b^1)P(c^2|b^1) \ + \tau_1(b^2)P(c^2|b^2)$

Define: $\tau_2(C) \ = \tau_1(b^1)P(C|b^1) + \tau_1(b^2)P(C|b^2)$

where $C \in \{c^1, c^2\}$

$$
\begin{array}{lllll}
 & \tau_1(b^1) & P(c^1 \mid b^1) & P(d^1 \mid c^1) \\
+ & \tau_1(b^2) & P(c^1 \mid b^2) & P(d^1 \mid c^1) \\
+ & \tau_1(b^1) & P(c^2 \mid b^1) & P(d^1 \mid c^2) \\
+ & \tau_1(b^2) & P(c^2 \mid b^2) & P(d^1 \mid c^2)
\end{array}
$$

$$
\begin{array}{lllll}
 & \tau_1(b^1) & P(c^1 \mid b^1) & P(d^2 \mid c^1) \\
+ & \tau_1(b^2) & P(c^1 \mid b^2) & P(d^2 \mid c^1) \\
+ & \tau_1(b^1) & P(c^2 \mid b^1) & P(d^2 \mid c^2) \\
+ & \tau_1(b^2) & P(c^2 \mid b^2) & P(d^2 \mid c^2)
\end{array}
$$

All terms involved in computation of $P(d^1)$ and $P(d^2)$ respectively. The sum is simplified because of use of $\tau_1(B)$.

Example of inference on Bayesian Network

- Computation shown alongside is easy and gives $P(D)$
- Previous steps are equivalent to **pushing summation inside**

$$P(D) = \sum_C \sum_B \sum_A P(A)P(B|A)P(C|B)P(D|C)$$

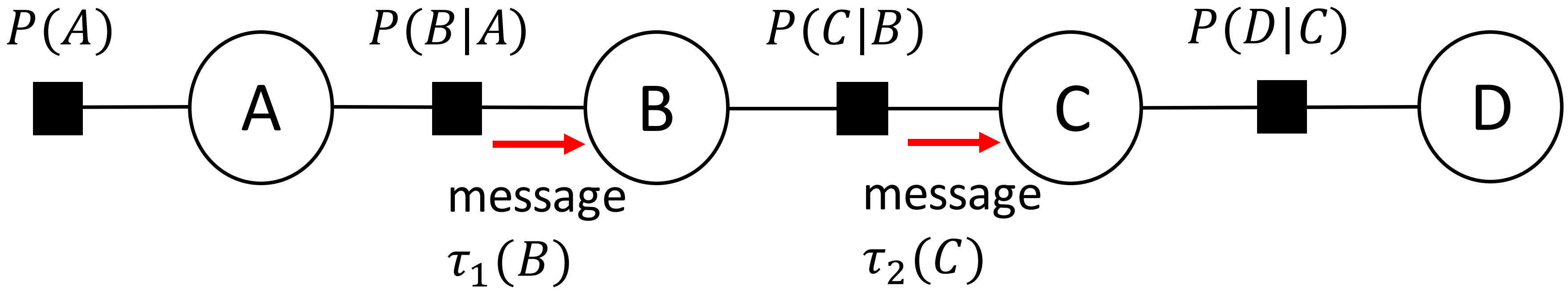
$$P(D) = \sum_C P(D|C) \overbrace{\sum_B P(C|B) \underbrace{\sum_A P(A)P(B|A)}_{\tau_1(B)}}^{\tau_2(C)}$$

$$\begin{array}{rll} & \tau_2(c^1) & P(d^1 \mid c^1) \\ + & \tau_2(c^2) & P(d^1 \mid c^2) \end{array}$$

$$\begin{array}{rll} & \tau_2(c^1) & P(d^2 \mid c^1) \\ + & \tau_2(c^2) & P(d^2 \mid c^2) \end{array}$$

Computation of $P(D)$ is simplified because of use of $\tau_1(B)$, $\tau_2(C)$.

Flow of computations (messages) in Factor Graph corresponding to the given BN



Sum-product algorithm reduces computations

- Pushing summations inside reduced the number of computations
  - Simple way: 48 multiplications + 14 additions
  - Pushing summations inside: 4x3 multiplications + 2x3 additions
  - Can be up to linear in number of variables (much better than exponential!)

- What helped in addressing the exponential blowup of marginalizing the joint distribution?
  - Graph structure – because of structure of Bayesian Network, some subexpression in the joint depend only on a small number of variables
  - Pushing summation inside – by computing these expressions once and caching the results, we can avoid generating them exponentially many times

- Referred to as **sum-product algorithm** or **Belief Propagation**

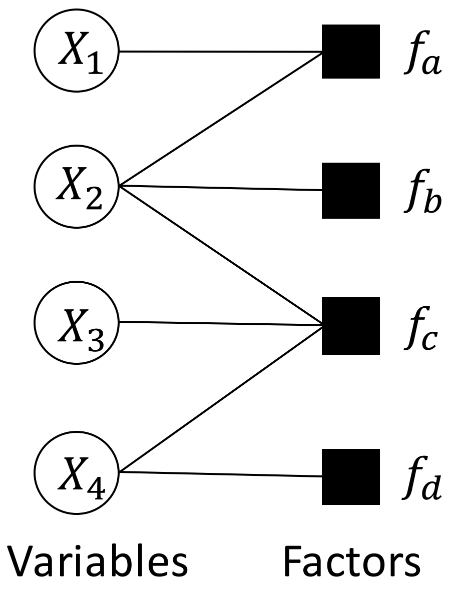
Inference problem on Factor Graphs

What is the problem we are trying to solve?

- Marginalization on Factor Graphs

Marginal probability $$P(X_i) = \sum_{\boldsymbol{X} \backslash X_i} P(\boldsymbol{X})$$

All variables except X_i



Challenge:

- Computationally expensive because the sum is calculated on all variables except one

$X_i \in \{0,1\}$ is a discrete variable
e.g., a Boolean variable

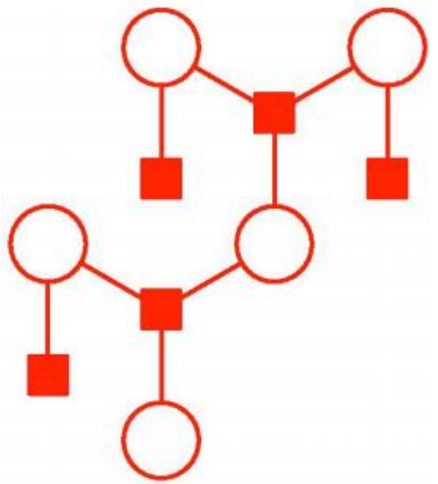
Approach:

- Factorize the joint distribution according the structure
- Use belief propagation to reduce computations

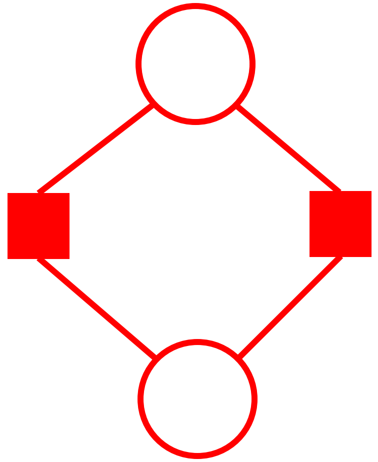
$f_c(X_c)$ is a tensor
on a set of variables X_c

Belief propagation

- Also known as sum-product algorithm
- Computes marginal distributions by “pushing in summations”
- Exact inference for linear graphs and trees
- Approximate inference for graphs with loops; performs remarkably well
- In case of Factor Graphs, involves two types of **messages**
    - From factor to variables
    - From variables to factors



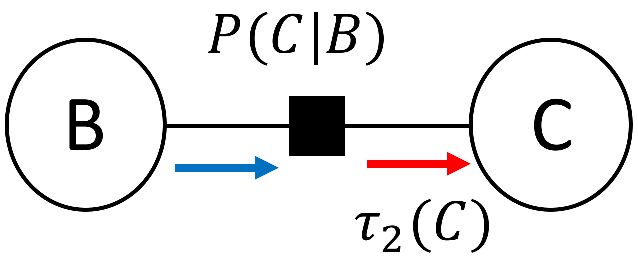
Tree Factor graph



Factor graph with loop

Belief Propagation – Message from factor to variable

Recall from previous example:

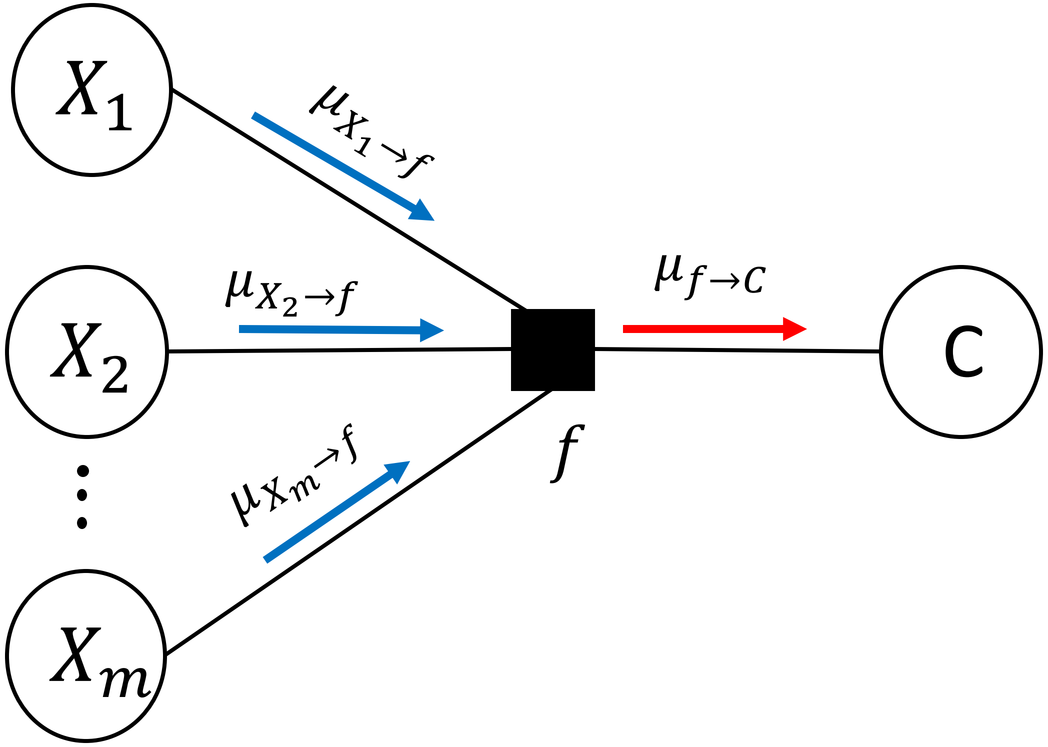


$$\tau_2(C) = \sum_B P(C|B)\tau_1(B)$$

To get the general expression, denote by:

$$f(B,C) = P(C|B)$$
$$\mu_{f\to C}(C) = \tau_2(C)$$
$$\mu_{B\to f}(B) = \tau_1(B)$$

In general:

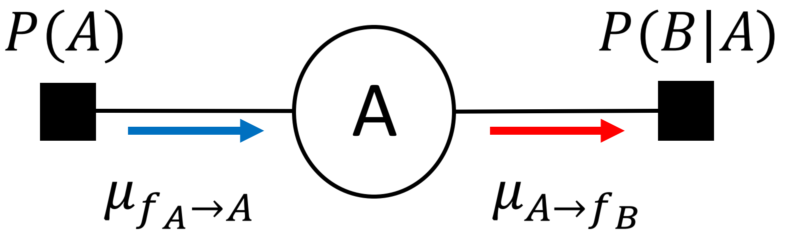


$$\mu_{f\to C}(C) = \sum_{X_1,X_2,\ldots,X_m} f(C,X_1,\ldots,X_m)\prod_{i=1}^{m}\mu_{X_i\to f}(X_i)$$

Message from factor to variable: Product of all incoming messages and factor, sum out previous variables

Belief Propagation – Message variable to factor

Recall from previous example:

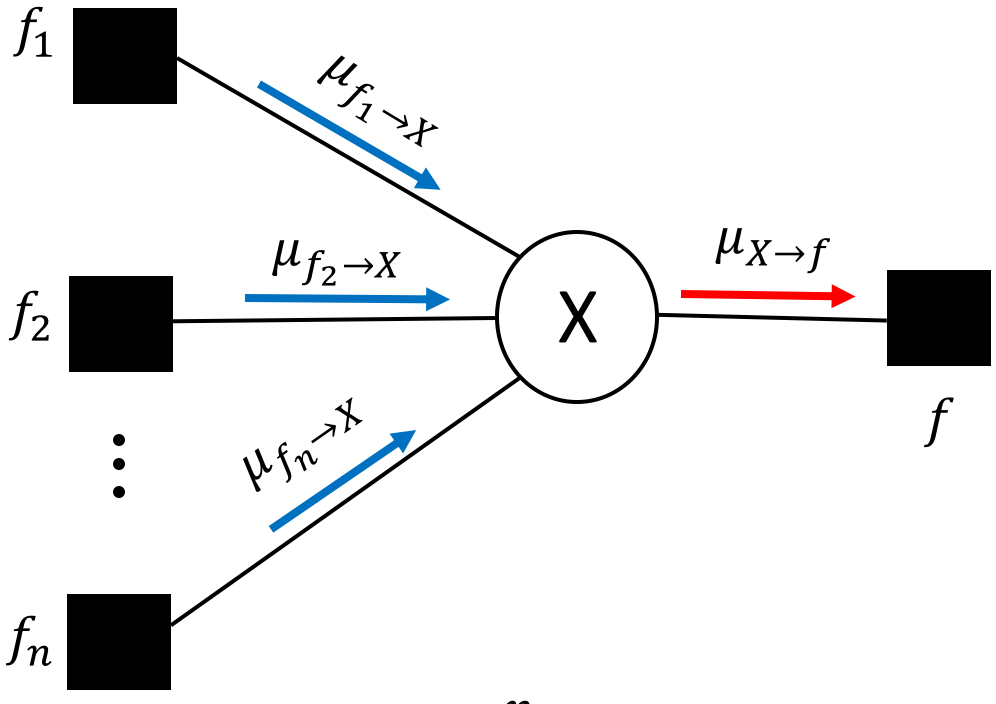


$$\mu_{A \to f_B}(A) = \mu_{f_A \to A}(A) = P(A)$$

Where,

$$f_A(A) = P(A)$$
$$f_B(A, B) = P(B|A)$$

In general:



$$\mu_{X \to f}(X) = \prod_{i=1}^{n} \mu_{f_i \to X}(X)$$

Message from factor to variable: Product of all incoming messages

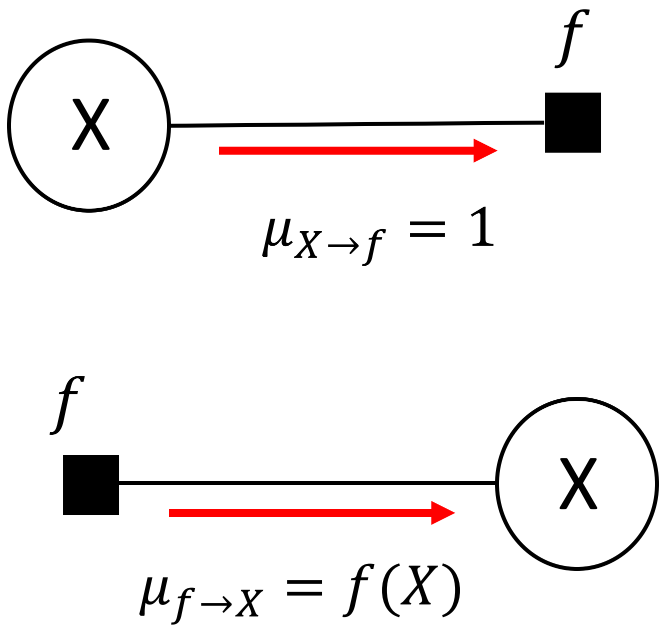
Belief Propagation: General Algorithm

Steps to compute marginal distribution for all variables

- How to start the algorithm
  - Choose a node in the factor graph as root node
  - Compute all the leaf-to-root messages
  - Compute all the root-to-leaf messages

- Initial Conditions
  - Starting from a factor leaf/root node, the initial factor-to-variable message is the factor itself
  - Starting from a variable leaf/root node, the initial variable-to-factor message is a vector of ones

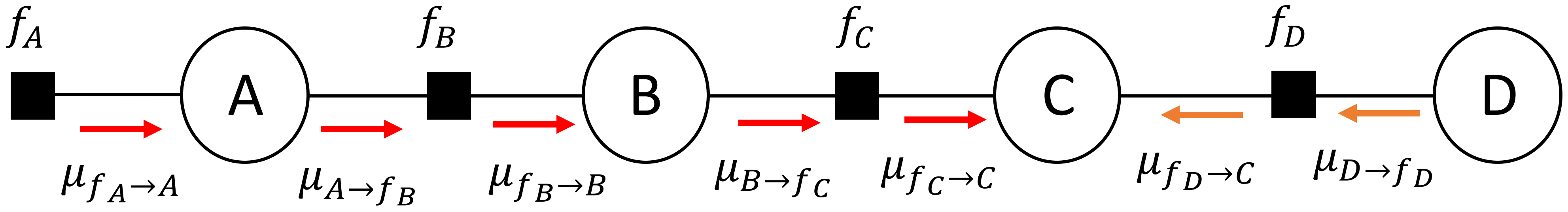
- Computing marginals
  - Marginal is given by the product of all incoming messages; normalize if necessary



Initialization

Example of belief propagation

Compute $P(C)$



$$\mu_{f_A \to A}(A) = f_A(A) = P(A)$$

$$\mu_{A \to f_B}(A) = \mu_{f_A \to A}(A) = P(A)$$

$$\mu_{f_B \to B}(B) = \sum_A f_B(A,B)\, \mu_{f_A \to A}(A)$$

$$= \sum_A P(B|A)\, P(A)$$

$$\mu_{B \to f_C}(B) = \mu_{f_B \to B}(B)$$

$$= \sum_A P(B|A)\, P(A)$$

$$\mu_{f_C \to C}(C) = \sum_B f_C(B,C)\, \mu_{B \to f_B}(B)$$

$$= \sum_B P(C|B) \sum_A P(B|A) P(A)$$

$$\mu_{D \to f_D}(D) = 1$$

$$\mu_{f_D \to C}(C)$$

$$= \sum_D f_D(C,D)\, \mu_{f_D \to C}(C)$$

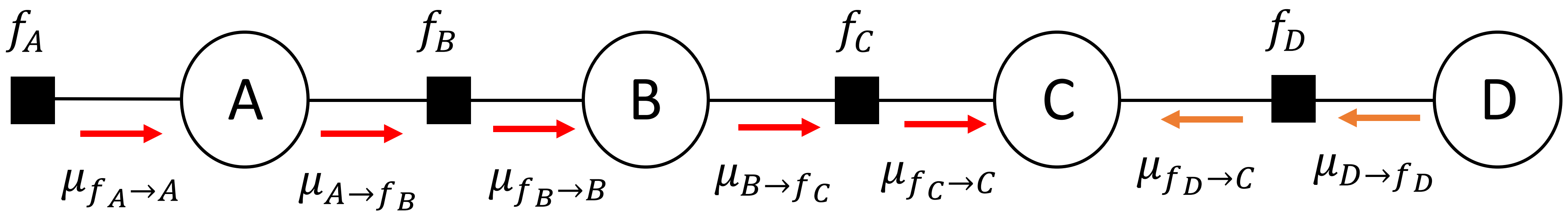
$$= \sum_D P(D|C)$$

Example of belief propagation

$$\mu_{f_D \to C}(C) = \sum_D P(D|C)$$

$$\mu_{f_C \to C}(C) = \sum_B P(C|B) \sum_A P(B|A)P(A)$$

Compute $P(C)$



$$P(C) = \mu_{f_C \to C}(C)\mu_{f_D \to C}(C)$$

Verifying that the above computation gives the marginal distribution

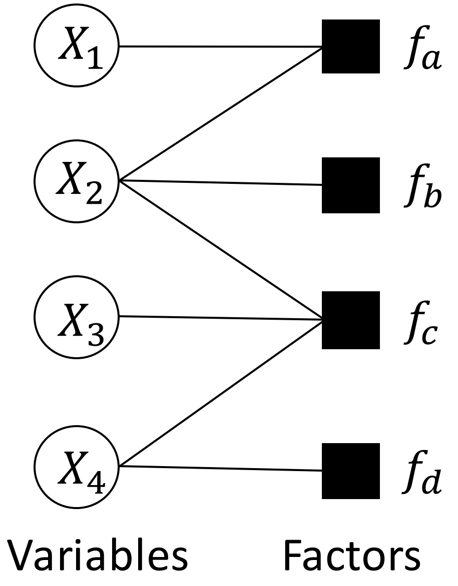
$$P(C) = \left(\sum_B P(C|B) \sum_A P(B|A)P(A)\right)\left(\sum_D P(D|C)\right) = \sum_B P(C|B) \sum_A P(B|A)P(A) \sum_D P(D|C)$$

$$= \sum_A \sum_B \sum_D P(A)P(B|A)P(C|B)P(D|C) = \sum_{A,B,D} P(A,B,C,D)$$

Marginalization on Factor Graphs

Marginal probability $$P(X_i) = \sum_{\boldsymbol{X} \backslash X_i} P(\boldsymbol{X})$$

All variables except X_i



Inference method	Description
Belief Propagation	Exact inference on non-loop FG
Sampling - Markov Chain Monte Carlo, Gibbs	Approximate inference
Variational Inference	Approximate inference

$X_i \in \{0,1\}$ is a discrete variable
e.g., a Boolean variable

$f_c(X_c)$ is a tensor
on a set of variables X_c

References

- Daphne Koller, Nir Friedman's textbook on Graphical Models
- https://www.psi.toronto.edu/~jimmy/ece521/Tut10.pdf
- https://www.doc.ic.ac.uk/~mpd37/teaching/ml_tutorials/2016-11-09-Svensson-BP.pdf