CAP6938-02
Plan, Activity, and Intent Recognition

Lecture 7:

Introduction to Graphical Models:
Part 2 (Exact Inference)

Instructor: Dr. Gita Sukthankar
Email: gitars@eecs.ucf.edu

Reminder

- Homework:
  - Tuesday: In-class demonstration of system
- Calendar:
  - Sept 25: in-class demo and presentation (no writeup)
  - Oct 4: Exam 1
  - Oct 11: System evaluation writeup

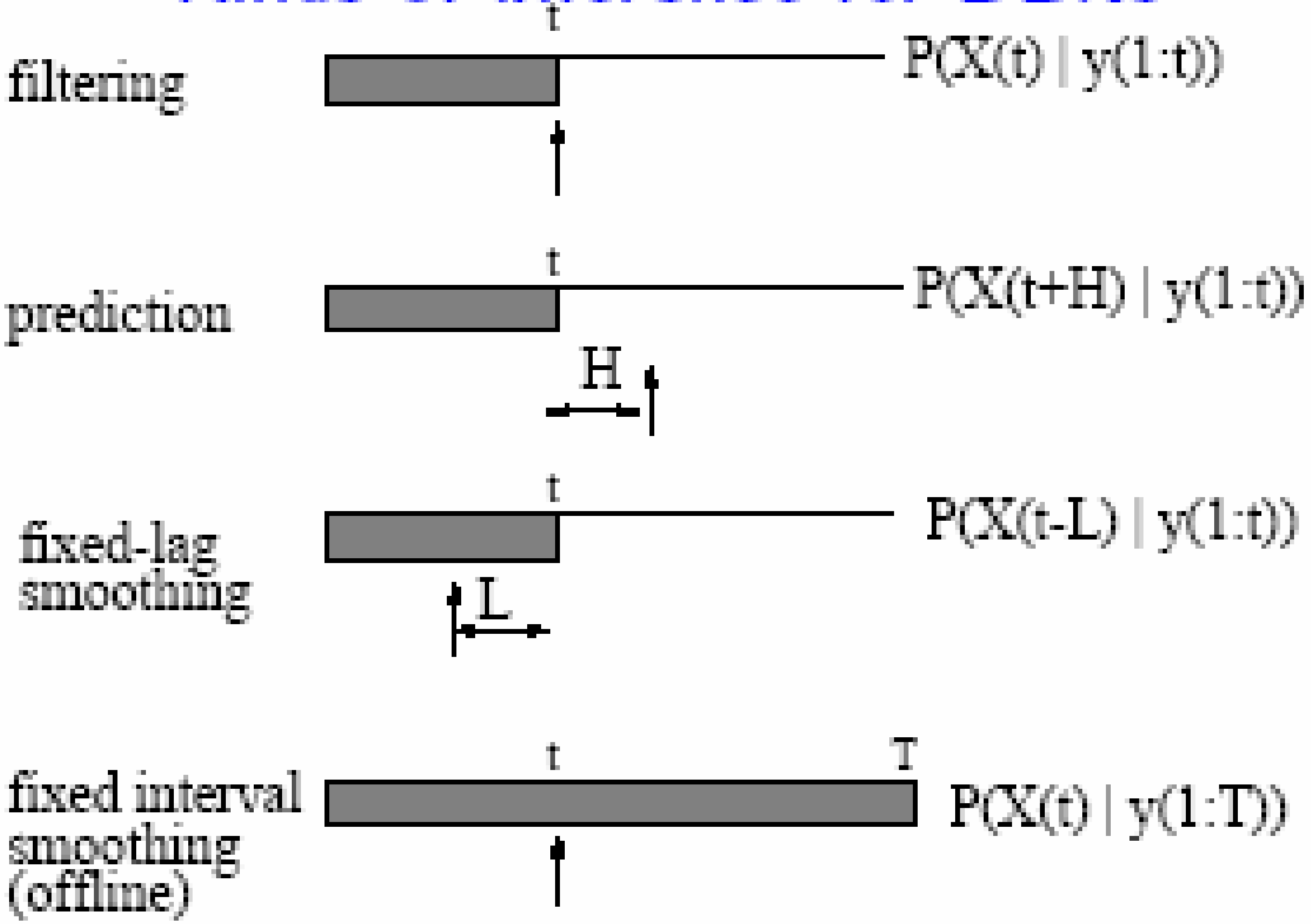
Question

- What is inference?

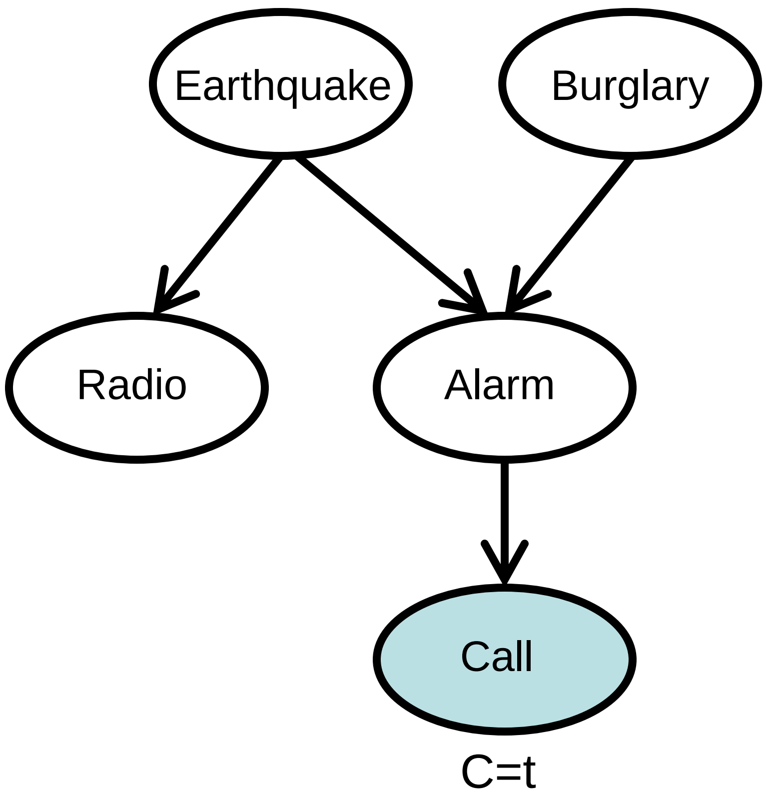
Graphical Model Definitions

- Representation: compactly representing joint probability distributions
- Inference: determine hidden states of a system given noisy observations
- Learning: how to estimate parameters and structure of the model
- Decision theory: how to convert belief into action

Kinds of inference for DBNs

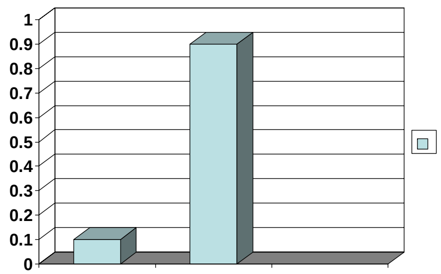


Inference (state estimation)

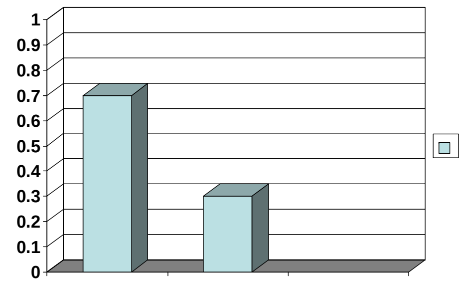


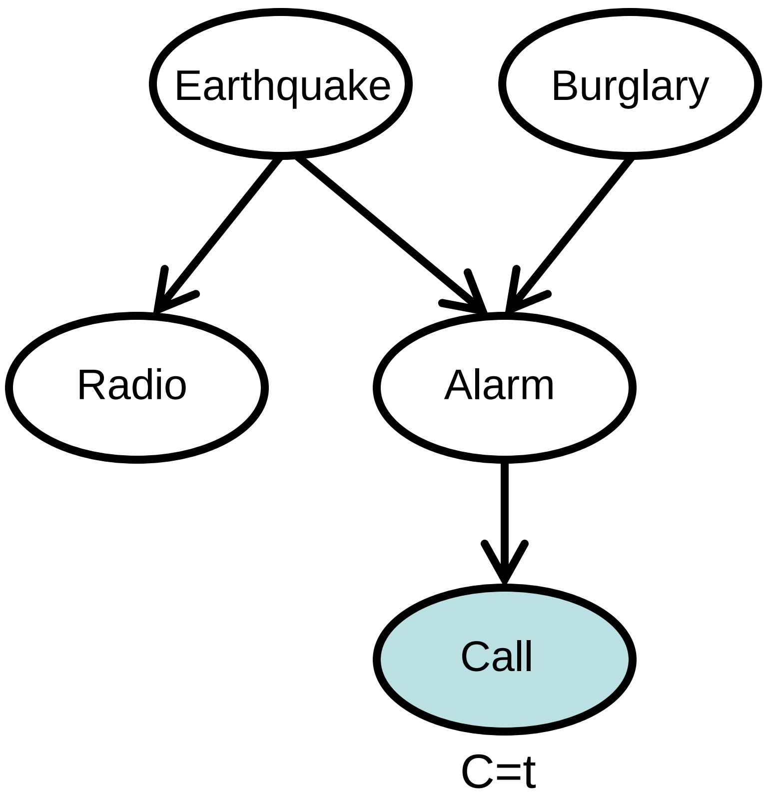
Inference

P(E=t|C=t)=0.1



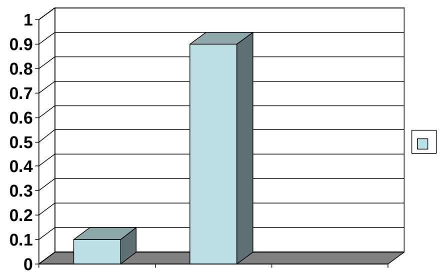
P(B=t|C=t) = 0.7



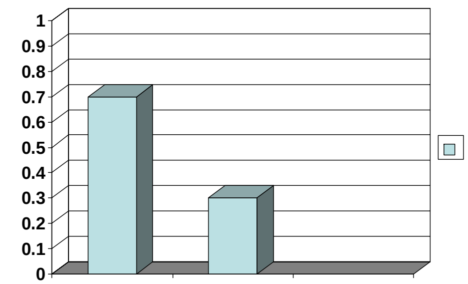


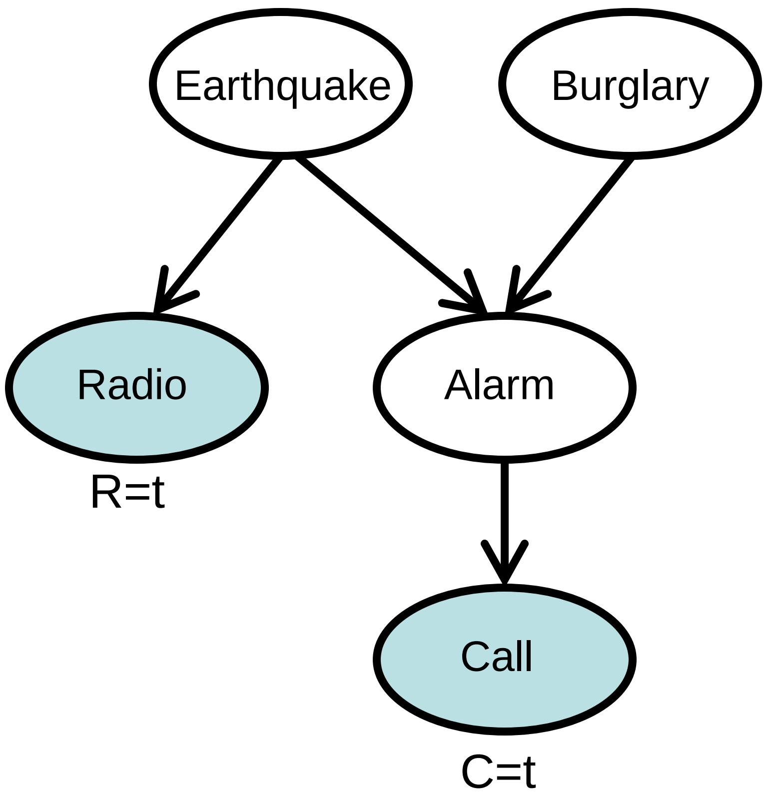
Inference

P(E=t|C=t)=0.1



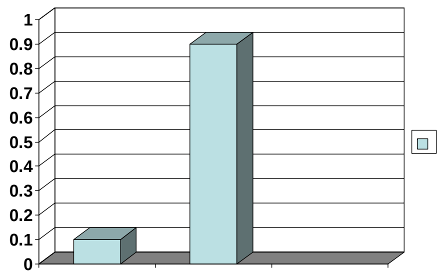
P(B=t|C=t) = 0.7



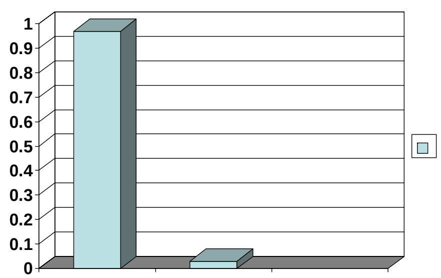


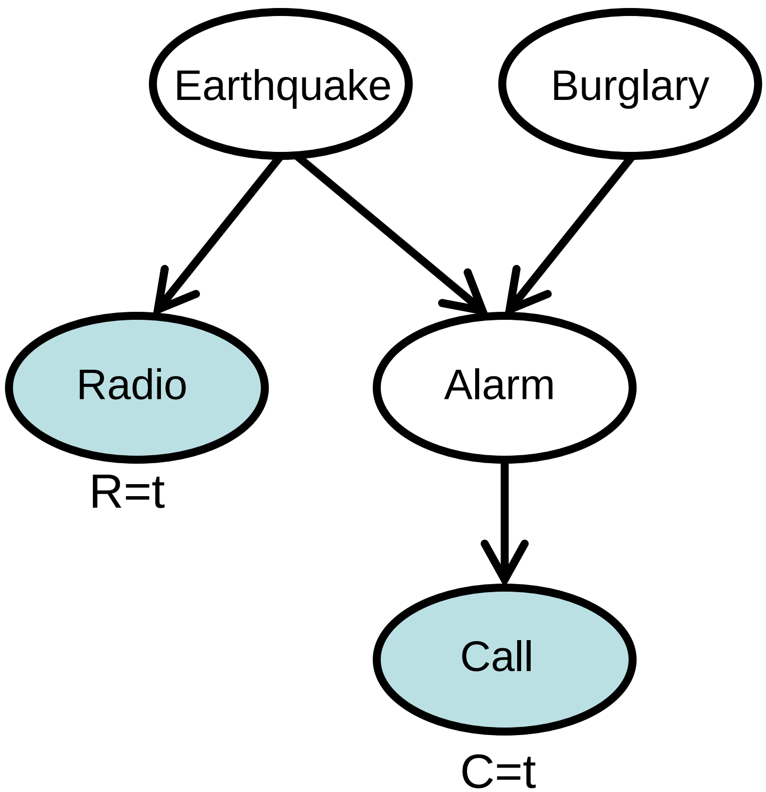
Inference

P(E=t|C=t)=0.1

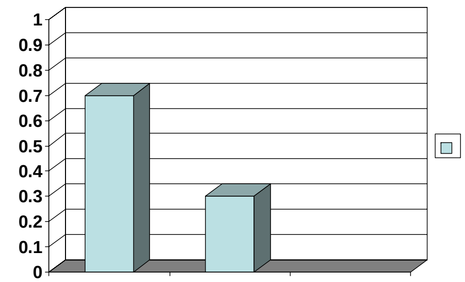


P(E=t|C=t,R=t)=0.97

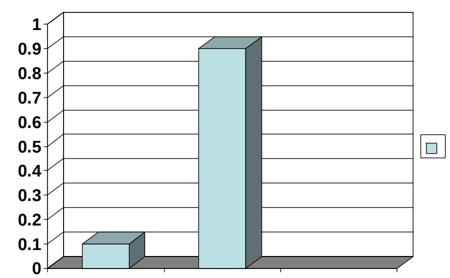




P(B=t|C=t) = 0.7

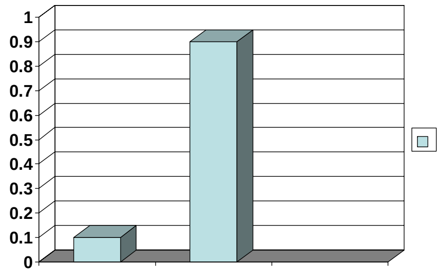


P(B=t|C=t,R=t) = 0.1

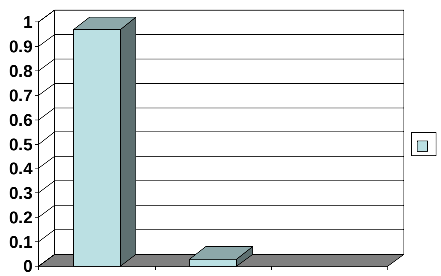


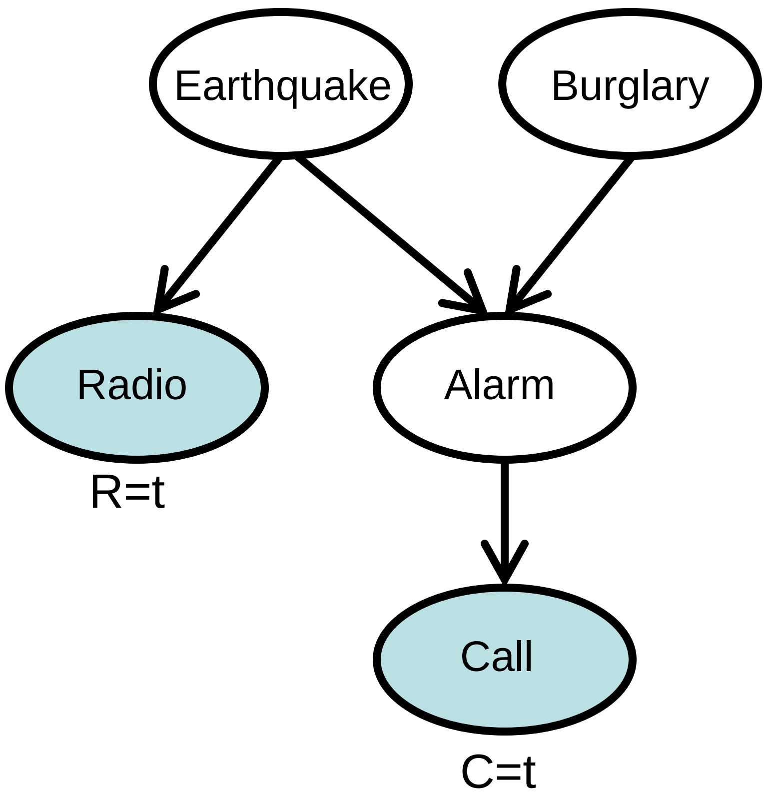
Inference

P(E=t|C=t)=0.1

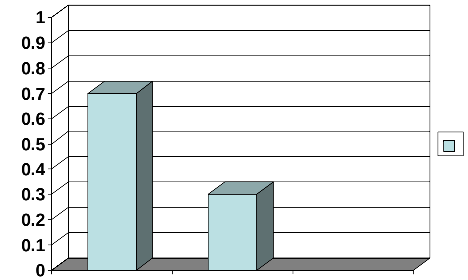


P(E=t|C=t,R=t)=0.97

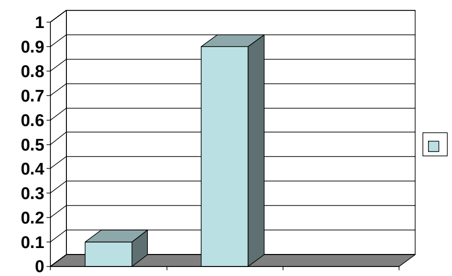




P(B=t|C=t) = 0.7

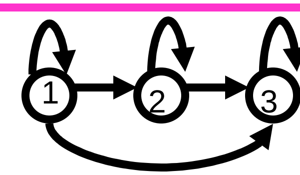


P(B=t|C=t,R=t) = 0.1



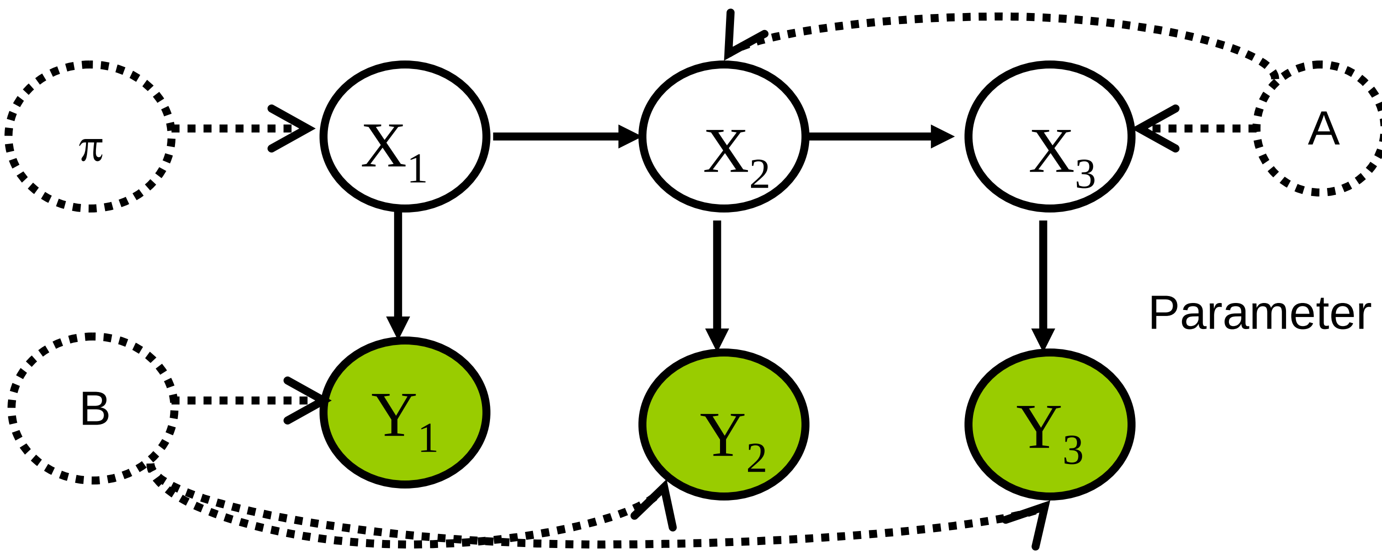
Explaining away effect

CPDs for HMMs



$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{pmatrix}$$

A=state transition matrix



Parameter tying

$$P(X_{1:T}, Y_{1:T}) = P(X_1)P(Y_1|X_1) \prod_{t=2}^{T} P(X_t|X_{t-1})P(Y_t|X_t)$$

Transition matrix $P(X_t = j|X_{t-1} = i) \;=\; A(i,j)$

Observation matrix $P(Y_t = j|X_t = i) \;=\; B(i,j)$

Initial state distribution $P(X_1 = i) \;=\; \pi(i)$

Question

- Why do we need multiple types of graph structures?

Other HMM Variants

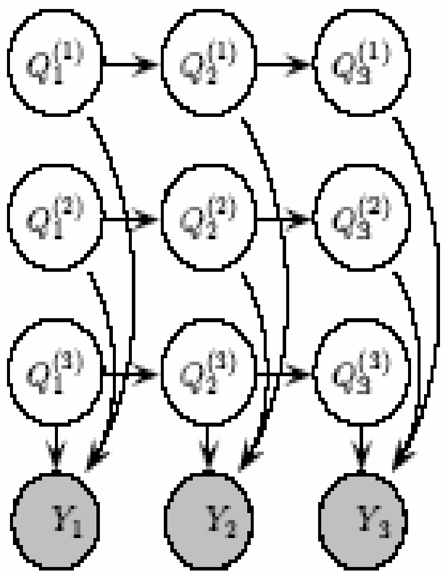


Figure 7: A factorial HMM with 3 hidden chains.

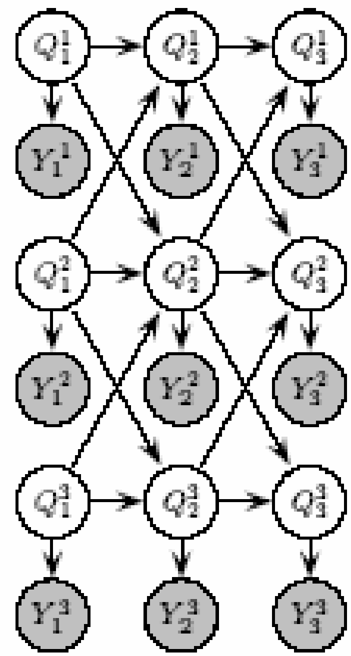


Figure 9: A coupled HMM with 3 chains.

Factorial **Coupled**

Nuisance variable=hidden node that we don't care about but that we don't know the value for

Inference tasks

- Posterior probabilities of Query given Evidence
  - Marginalize out Nuisance variables
  - Sum-product

$$P(X_Q|X_E = x_e) = \frac{\sum_{x_n} P(X_Q, x_n, x_e)}{\sum_{x_q} \sum_{x_n} P(x_q, x_n, x_e)}$$

- Most Probable Explanation (MPE)/ Viterbi
  - max-product

$$x_q^* = \arg\max_{x_q} P(x_q|x_e) = \arg\max_{x_q} P(x_q, x_e)$$

- "Marginal Maximum A Posteriori (MAP)"
  - max-sum-product

$$x_q^* = \arg\max_{x_q} P(x_q|x_e) = \arg\max_{x_q} \sum_{x_n} P(x_q, x_n, x_e)$$

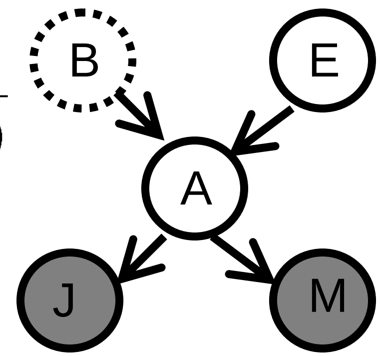
Outline

- Exact inference
  - Brute force enumeration
  - Variable elimination algorithm
  - Loopy graphs
  - Forwards-backwards algorithm

Brute force enumeration

- We can compute $P(X_Q|x_e) = \sum_{x_n} P(X_Q, x_n, x_e)$ in $O(K^N)$ time, where $K=|X_i|$

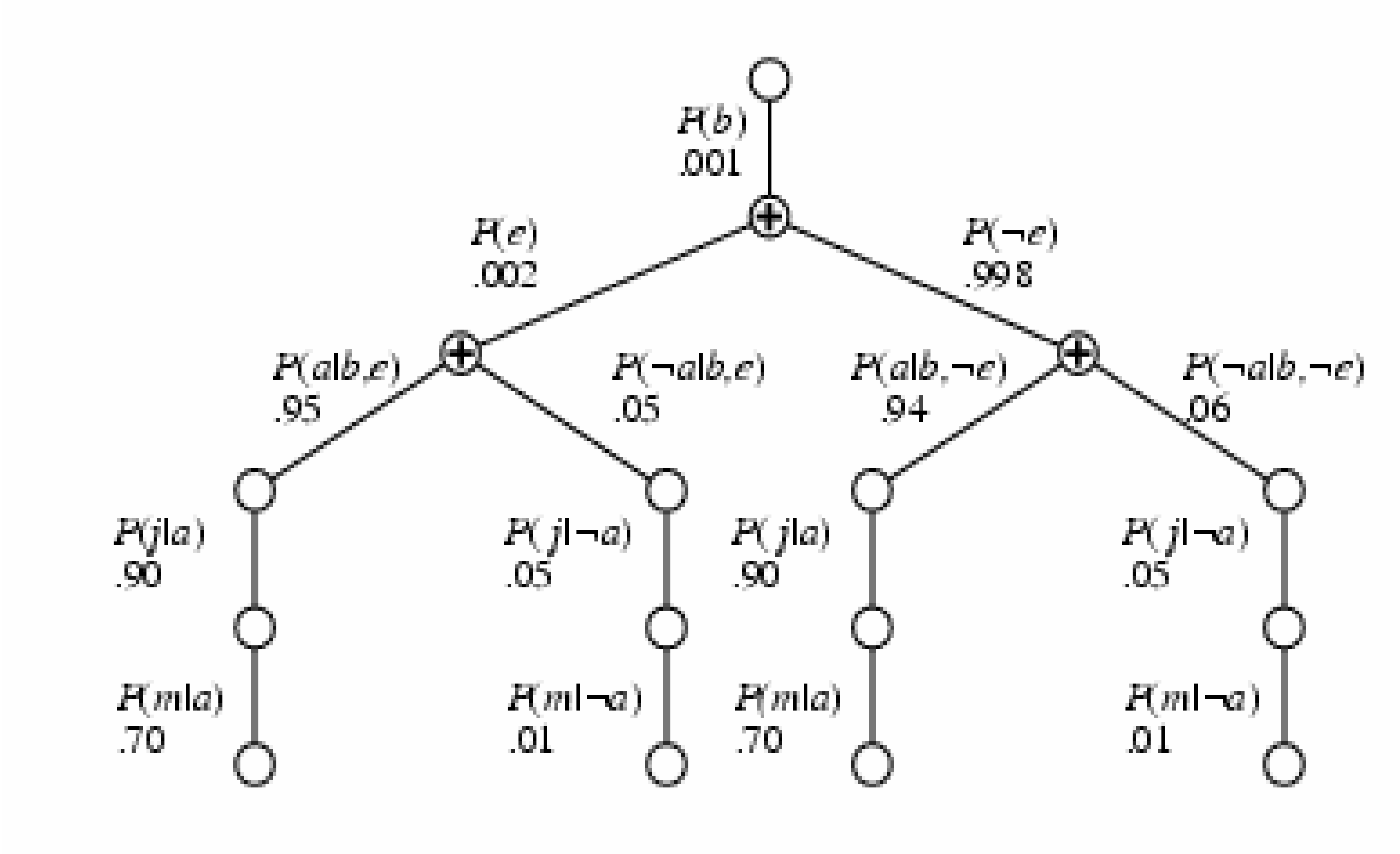
$$
\begin{aligned}
P(b|j,m) &= \frac{P(b,j,m)}{P(j,m)} = \frac{P(b,j,m)}{\sum_{b'} P(b',j,m)} \\
&= \alpha P(b,j,m) \\
&= \alpha \sum_e \sum_a P(b,e,a,j,m)
\end{aligned}
$$



- By using BN, we can represent joint in O(N) space

$$P(b|j,m) = \alpha \sum_e \sum_a P(b)P(e)P(a|b,e)P(j|a)P(m|a)$$

Enumeration tree

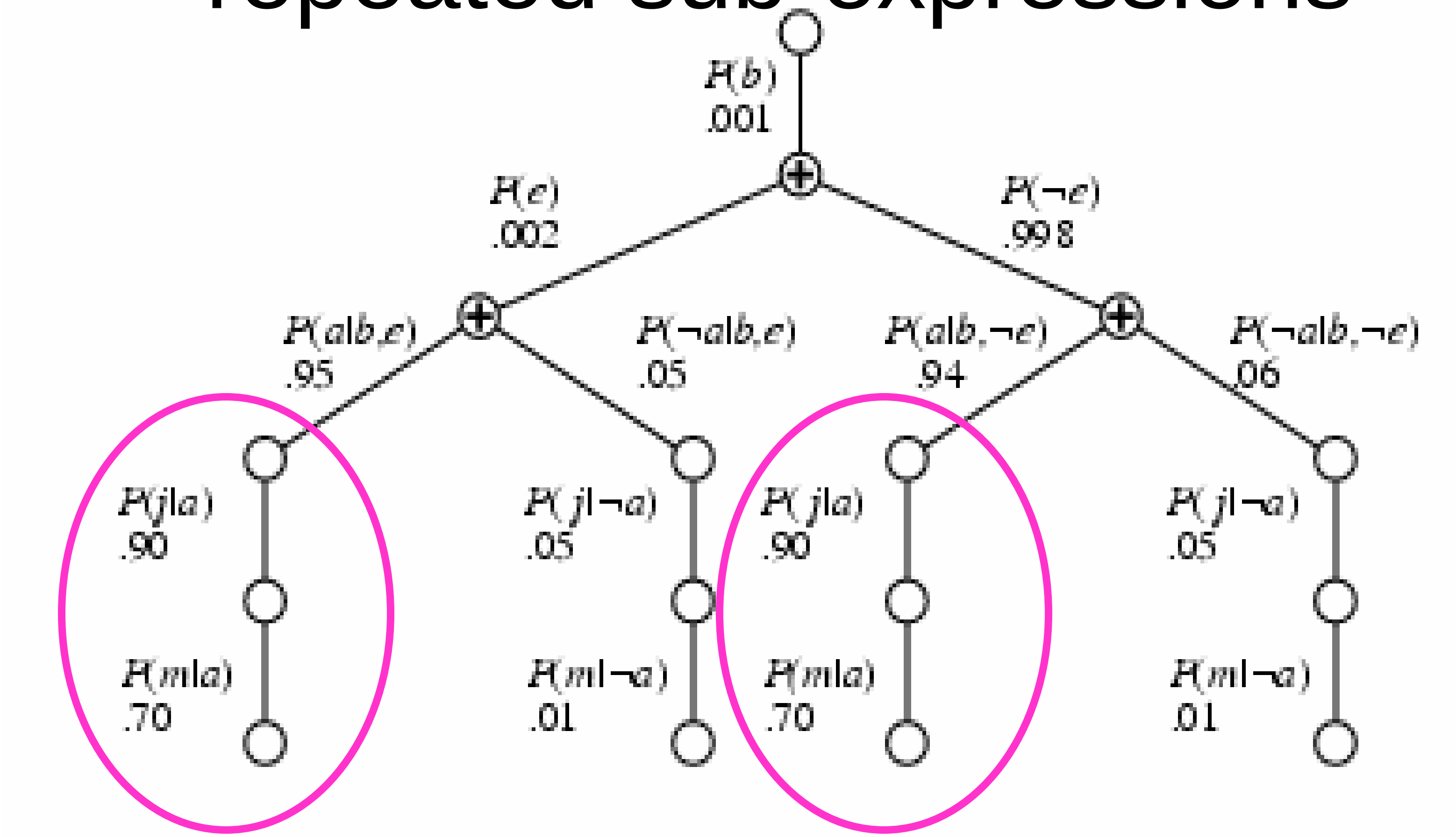


$$P(b|j,m) = \alpha \sum_e \sum_a P(b)P(e)P(a|b,e)P(j|a)P(m|a)$$

(E and A are nuisance variables)

Russell & Norvig

Enumeration tree contains repeated sub-expressions



$$P(b|j,m) \;=\; \alpha \sum_{e} \sum_{a} P(b)P(e)P(a|b,e)P(j|a)P(m|a)$$

Outline

- Exact inference
  - Brute force enumeration
  - Variable elimination algorithm
  - Loopy graphs
  - Forwards-backwards algorithm

Kschischang01,Dechter96

Variable/bucket elimination

- Push sums inside products (generalized distributive law)
- Carry out summations right to left, storing intermediate results (factors) to avoid recomputation (dynamic programming)

$$\begin{aligned} P(b|j,m) &= \alpha \sum_e \sum_a P(b)P(e)P(a|b,e)P(j|a)P(m|a) \\ &= \alpha P(b) \sum_e P(e) \sum_a P(a|b,e)P(j|a)P(m|a) \end{aligned}$$

VarElim: basic operations

- Pointwise product

$$f_1(a,b) \times f_2(b,c) = f_{12}(a,b,c)$$

- Summing out

$$\begin{aligned} & \sum_c f_1(a,b) f_2(b,c) f_3(c,d) \\ = & \; f_1(a,b) \sum_c f_4(b,c,d) \\ = & \; f_1(a,b) f_5(b,d) \end{aligned}$$

Variable elimination

$$
\begin{aligned}
P(b|j,m) &= \alpha \underbrace{P(b)}_{B} \sum_e \underbrace{P(e)}_{E} \sum_a \underbrace{P(a|b,e)}_{A} \underbrace{P(j|a)}_{J} \underbrace{P(m|a)}_{M} \\
&= \alpha P(b) \sum_e P(e) \sum_a P(a|b,e) P(j|a) F_M(A) \\
&= \alpha P(b) \sum_e P(e) \sum_a P(a|b,e) F_J(a) F_M(a) \\
&= \alpha P(b) \sum_e P(e) \sum_a F_A(a,b,e) F_J(a) F_M(a) \\
&= \alpha P(b) \sum_e P(e) F_{\overline{A}JM}(b,e) \quad \text{sum out A} \\
&= \alpha P(b) \sum_e F_E(e) F_{\overline{A}JM}(b,e) \\
&= \alpha P(b) F_{\overline{EA}JM}(b) \quad \text{sum out E}
\end{aligned}
$$

Variable elimination

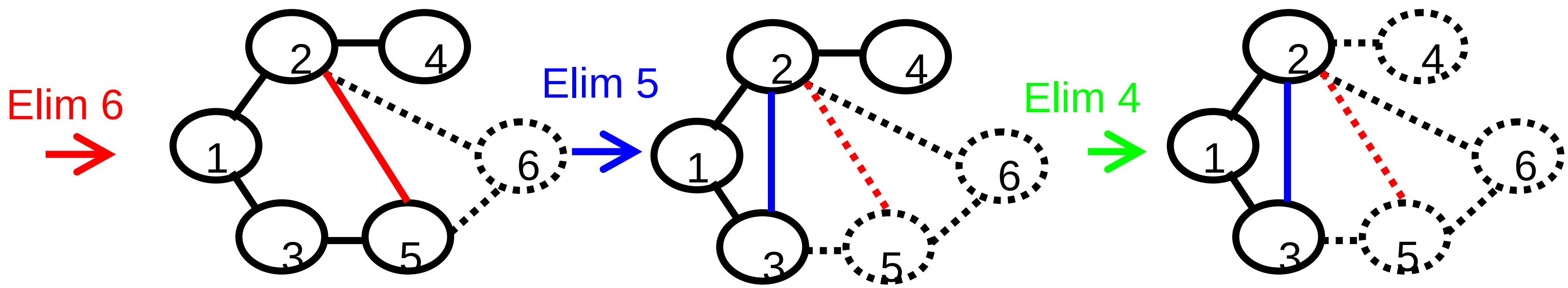
```
function ELIMINATION-ASK(X, e, bn) returns a distribution over X
   inputs: X, the query variable
           e, evidence specified as an event
           bn, a belief network specifying joint distribution P(X_1, ..., X_n)

   factors ← []; vars ← REVERSE(VARS[bn])
   for each var in vars do
      factors ← [MAKE-FACTOR(var, e)|factors]
      if var is a hidden variable then factors ← SUM-OUT(var, factors)
   return NORMALIZE(POINTWISE-PRODUCT(factors))
```

Outline

- Exact inference
  - Brute force enumeration
  - Variable elimination algorithm
  - Loopy graph
  - Forwards-backwards algorithm

VarElim on loopy graphs

Let us work right-to-left, eliminating variables, and adding arcs to ensure that any two terms that co-occur in a factor are connected in the graph



$$
\begin{aligned}
P(x_1|x_6^*) &\propto P(x_1)\sum_{x_2} P(x_2|x_1)\sum_{x_3} P(x_3|x_1)\sum_{x_4} P(x_4|x_2)\sum_{x_5} P(x_5|x_3)\sum_{x_6^*} P(x_6^*|x_2,x_5) \\
&= P(x_1)\sum_{x_2} P(x_2|x_1)\sum_{x_3} P(x_3|x_1)\sum_{x_4} P(x_4|x_2)\sum_{x_5} \underbrace{P(x_5|x_3) f_6(x_2,x_5)}_{x_2,x_3,x_5} \\
&= P(x_1)\sum_{x_2} P(x_2|x_1)\sum_{x_3} P(x_3|x_1)\sum_{x_4} P(x_4|x_2)\sum_{x_5} f(x_2,x_3,x_5) \\
&= P(x_1)\sum_{x_2} P(x_2|x_1)\sum_{x_3} P(x_3|x_1) m_5(x_2,x_3)\sum_{x_4} P(x_4|x_2)
\end{aligned}
$$

Complexity of VarElim

- Time/space for single query = $O(N K^{w+1})$ for N nodes of K states, where $w=w(G, \pi)$ = width of graph induced by elimination order π
- $w^* = \text{argmin}_\pi w(G,\pi)$ = treewidth of G
- Thm: finding an order to minimize treewidth is NP-complete Yannakakis81
- Does there exist a more efficient exact inference algorithm?

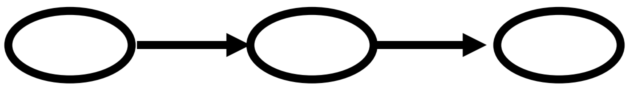
Summary so far

- Brute force enumeration $O(K^N)$ time, $O(N\ K^C)$ space (where C=max clique size)
- VarElim $O(N\ K^{w+1})$ time/space
  - $w = w(G,\pi)$ = induced treewidth
- Exact inference is #P-complete
  - Motivates need for approximate inference

Treewidth

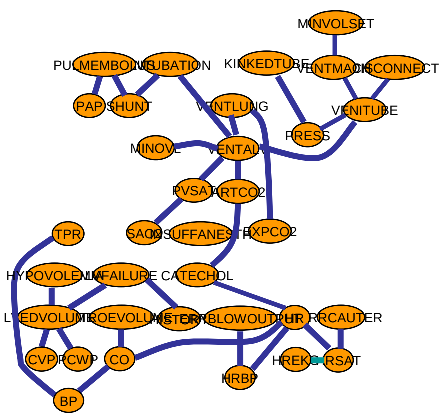
Low treewidth

Chains



$W^* = 1$

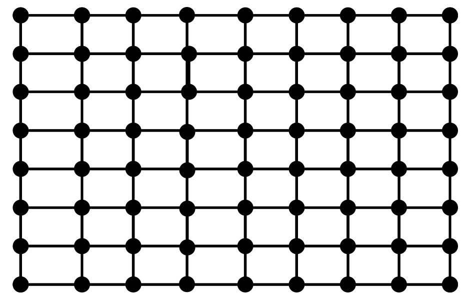
Trees (no loops)



$W^* = \#parents$

High tree width

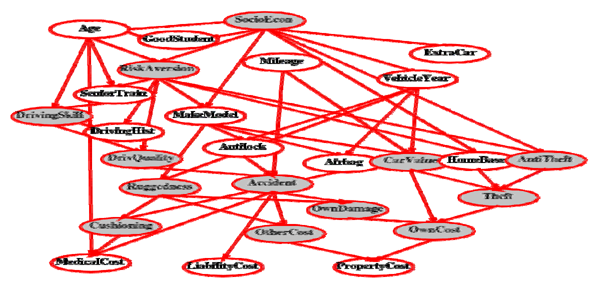
N=nxn grid



$W^* = O(n) = O(p\ N)$

Arnborg85

Loopy graphs

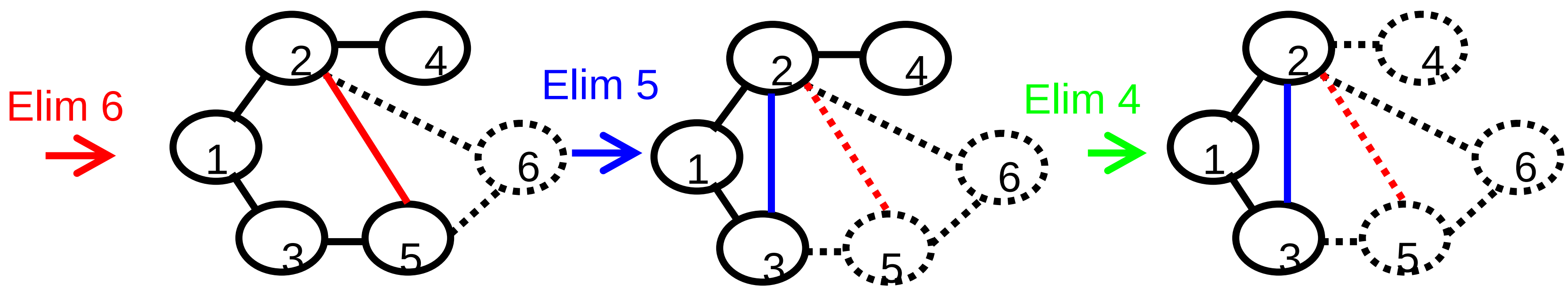


$W^* =$ NP-hard to find

Graph triangulation

Golumbic80

- A graph is triangulated (chordal, perfect) if it has no chordless cycles of length > 3.
- To triangulate a graph, for each node X_i in order π, ensure all neighbors of X_i form a clique by adding fill-in edges; then remove X_i



Finding an elimination order

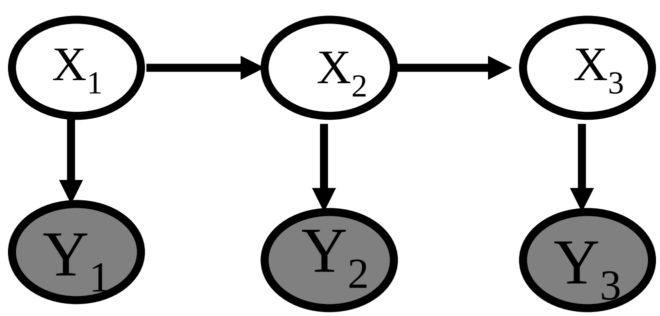
- The size of the induced clique depends on the elimination order.
- Since this is NP-hard to optimize, it is common to apply greedy search techniques: Kjaerulff90
- At each iteration, eliminate the node that would result in the smallest
  - Num. fill-in edges [min-fill]
  - Resulting clique weight [min-weight] (Weight of clique = product of number of states per node in clique)
- There are some approximation algorithms Amir01

Outline

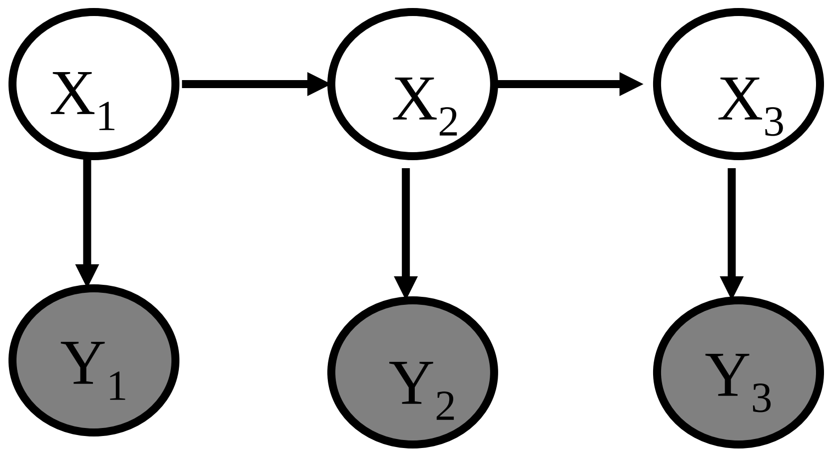
- Exact inference
  - Brute force enumeration
  - Variable elimination algorithm
  - Loopy graph
  - Forwards-backwards algorithm

What's wrong with VarElim

- Often we want to query all hidden nodes.
- VarElim takes $O(N^2 K^{w+1})$ time to compute $P(X_i|x_e)$ for all (hidden) nodes i.
- There exist message passing algorithms that can do this in $O(N K^{w+1})$ time.
- Later, we will use these to do approximate inference in $O(N K^2)$ time, indep of w.



Repeated variable elimination leads to redundant calculations



$$P(x_1|y_{1:3}) \propto f(x_1)\sum_{x_2} f(x_1,x_2)\sum_{x_3} f(x_2,x_3)$$

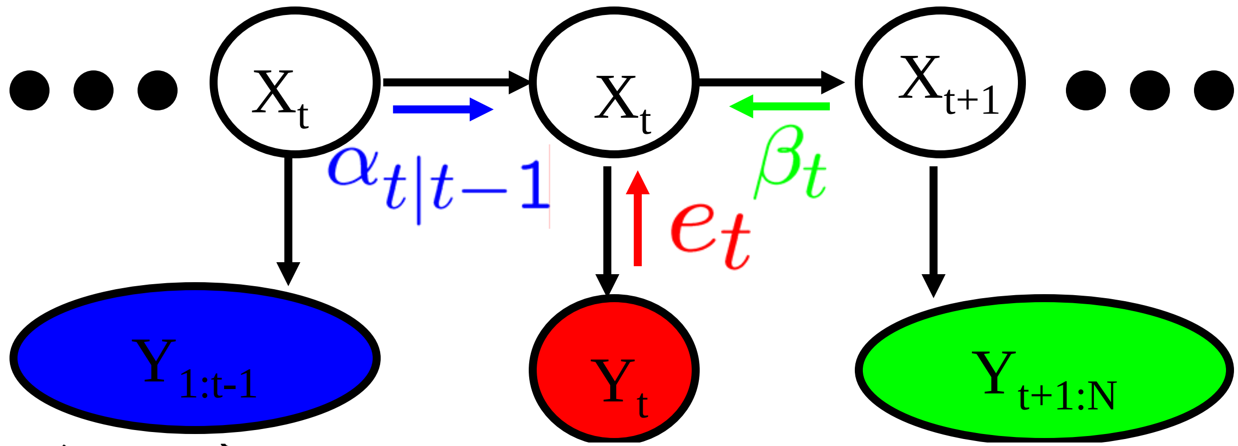
$$P(x_2|y_{1:3}) \propto \sum_{x_1} f(x_1)f(x_1,x_2)\sum_{x_3} f(x_2,x_3)$$

$$P(x_3|y_{1:3}) \propto \sum_{x_1} f(x_1)\sum_{x_2} f(x_1,x_2)f(x_2,x_3)$$

$O(N^2 K^2)$ time to compute all N marginals

Forwards-backwards algorithm

Rabiner89,etc

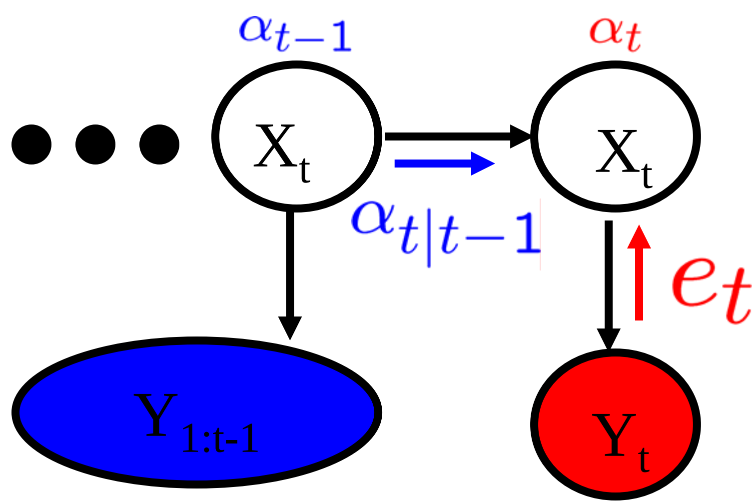


$$
\begin{aligned}
P(X_t = j | y_{1:N}) \\
\propto \quad & P(X_t, y_{1:t-1}, y_t, y_{t+1:N}) \\
\propto \quad & P(y_{t+1:N} | X_t, \cancel{y_t}, \cancel{y_{1:t-1}}) P(y_t | X_t, \cancel{y_{1:t-1}}) P(X_t | y_{1:t-1}) \\
\propto \quad & P(X_t | y_{1:t-1}) P(y_t | X_t) P(y_{t+1:N} | X_t) \\
\stackrel{\text{def}}{=} \quad & \alpha_{t|t-1}(j) e_t(j) \beta_t(j)
\end{aligned}
$$

(Use dynamic programming to compute these)

Forwards prediction — Local evidence — Backwards prediction

Forwards algorithm (filtering)

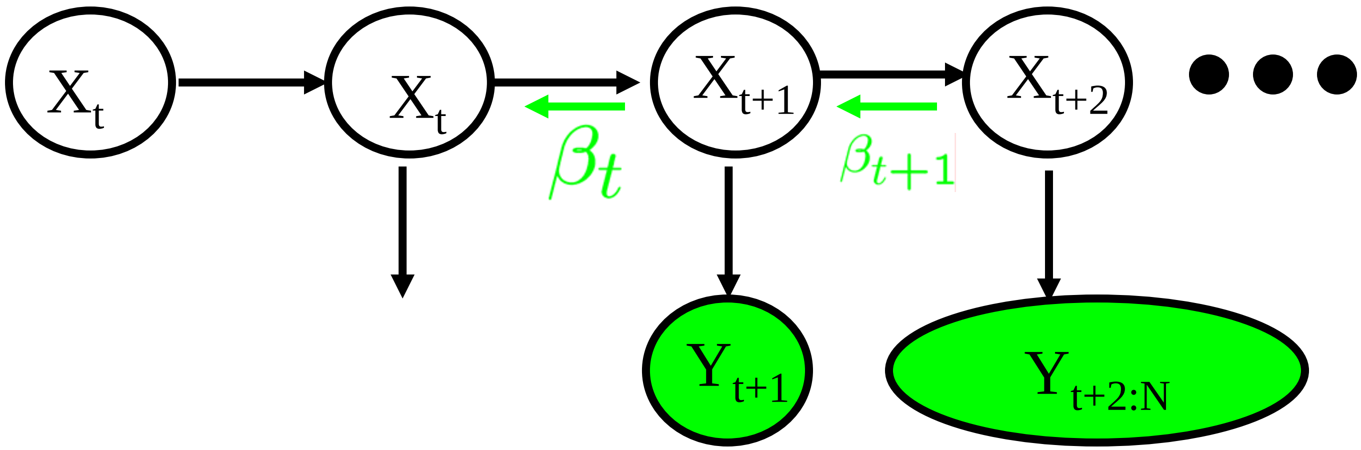


Use the Markov assumptions

$$\begin{aligned}
\alpha_t(j) &\overset{\text{def}}{=} P(X_t = j | y_{1:t}) \\
&\propto P(y_t | X_t = j, \cancel{y_{1:t-1}}) P(X_t = j | y_{1:t-1}) \\
&= P(y_t | X_t = j) \sum_i P(X_t | X_{t-1}, \cancel{y_{1:t-1}}) P(X_{t-1} | y_{1:t-1}) \\
&= e_t(j) \sum_i A(i,j) \alpha_{t-1}(i)
\end{aligned}$$

$$\alpha_t \propto e_t .* A^T \alpha_{t-1}$$

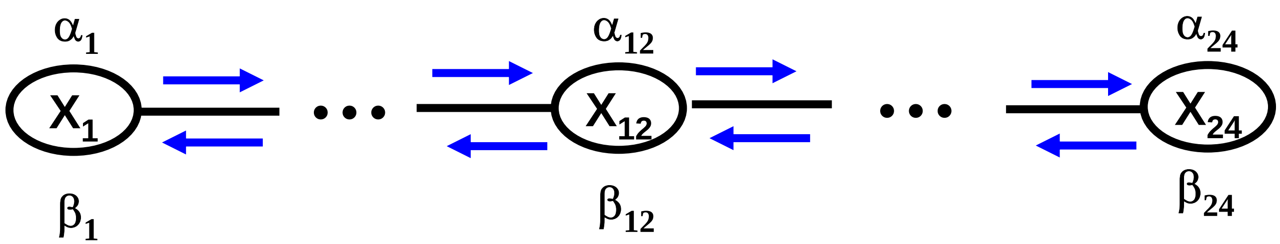
Backwards algorithm



$$
\begin{aligned}
\beta_t(i) &\stackrel{\text{def}}{=} P(y_{t+1:N}|X_t = i) \\
&= \sum_j P(y_{t+1}, y_{t+2:N}, X_{t+1} = j|X_t = i) \\
&= \sum_j P(y_{t+2:N}|X_{t+1}, \cancel{X_t}, \cancel{y_{t+1}})P(y_{t+1}|X_{t+1}, \cancel{X_t})P(X_{t+1}|X_t) \\
&= \sum_j \beta_{t+1}(j)e_{t+1}(j)A(i,j)
\end{aligned}
$$

$$\beta_t \propto A(e_{t+1} .* \beta_{t+1})$$

Forwards-backwards algorithm



- Forwards

$$\alpha_t \propto (A^T \alpha_{t-1}) .* e_t$$

- Backwards

$$\beta_t \propto A(\beta_{t+1} .* e_{t+1})$$

Backwards messages independent of forwards messages

- Combine

$$P(X_t = i | y_{1:T}) \propto \alpha_t(i) \beta_t(i)$$

O(N K^2) time to compute all N marginals, not O(N^2 K^2)

References

- K. Murphy, Exact inference in graphical models, AAAI tutorial 2004