Proof of Hardness

$\boxed{C_1 = x_1 \vee \bar{x}_2}$

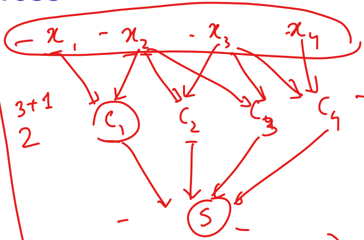
$C_2 = x_3 \vee x_2$

$C_3 = x_2 \vee \bar{x}_3$

$C_4 = x_3 \vee x_4$

$\boxed{S = C_1 \wedge C_2 \wedge C_3 \wedge C_4}$

2^{3+1}



$P(x_i) = \boxed{1/2 \mid 1/2}$

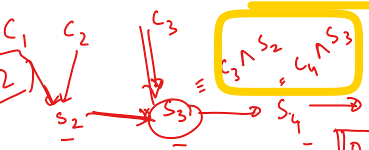
CPD

$P(C_1 | x_1, x_2) \Leftrightarrow C_1 = x_1 \vee \bar{x}_2$

2^3

	00	01	10	11
0	0	1	0	0
1	1	0	1	1

$P(S | C_1, C_2 \cdots C_k)$ $2^{k+1} \neq$ polynomial in $n \mid k$



$\equiv C_3 \wedge S_2$, $C_4 \wedge S_3$

$\equiv S$

$\boxed{P(S=1)} > 0$

space reqd: $(k-1)2^3 + k2^4 + 2$

polynomial in n, k

doubt

Proof of Hardness

Variable elimination on general graphs

- Given, arbitrary sets of potentials $\psi_C(x_C)$, C = cliques in a graph G.
- Find, $Z = \sum_{x_1,\ldots,x_n} \prod_C \psi_C(x_C)$

$P(x_j)$ $\underset{x_1 \ldots x_n}{\text{argmax}}$ $P(\cancel{x_1}, \ldots x_n)$ such that x_j is last.

~~$x_1, \ldots x_n$ = good ordering of variables~~

$\mathcal{F} = \{\psi_C(x_C), C = \text{cliques in a graph } G\}$ {can be non-maximal}

for $i = 1 \ldots n^{-1}$ **do**

$\quad \mathcal{F}_i$ = factors in $\mathcal{F}$ that contain x_i $\binom{w_i}{m}$

$\quad M_i$ = product of factors in $\mathcal{F}_i$ ← w_i

$\quad m_i = \sum_{x_i} M_i$ $\quad \hat{m}_i = \max_{x_i} M_i$ { keep around the maximizing assignment }.

$\quad \mathcal{F} = \mathcal{F} - \mathcal{F}_i \cup \{m_i\}$

end for

$\mathcal{F}$ will consist of a constant $= Z$. $\quad \mathcal{F}$ will contain max value and maximizing assignment.

$\mathcal{F}$ will consist of $m_{n-1}(x_j)/Z = P(x_j)$

Example: Variable elimination

x_2 $M_2(x_1, x_2, x_3, x_4)$ $w=4$ $n m^4$

nm^3

- Given, $F = \{\psi_{12}(x_1, x_2),\ \psi_{24}(x_2, x_4),\ \psi_{23}(x_2, x_3),\ \psi_{45}(x_4, x_5),\ ,\ \psi_{35}(x_3, x_5)\}$
- Find, $Z =$ Brute-force: $\to M(x_1, x_2, x_3, x_4, x_5)$ 2^5 m^5
$\sum_{x_1,\ldots,x_5} \psi_{12}(x_1, x_2)\psi_{24}(x_2, x_4)\psi_{23}(x_2, x_3)\psi_{45}(x_4, x_5)\psi_{35}(x_3, x_5)$.

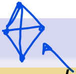
1. x_1: $\prod\{\overset{F_1}{\psi_{12}(x_1, x_2)}\} \to M_1(x_1, x_2) \xrightarrow{\sum_{x_1}} m_1(x_2)$
2. x_2: $\prod\{\psi_{24}(x_2, x_4), \psi_{23}(x_2, x_3), m_1(x_2)\} \to M_2(x_2, x_3, x_4) \xrightarrow{\sum_{x_2}} m_2(x_3, x_4)$

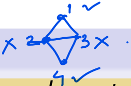
   $F = \{\psi_{35}, \overset{F_2}{}\psi_{45}, m_2\}$
3. x_3: $\prod\{\psi_{35}(x_3, x_5), m_2(x_3, x_4)\} \to M_3(x_3, x_4, x_5) \xrightarrow{\sum_{x_3}} m_3(x_4, x_5)$
4. x_4: $\prod\{\psi_{45}(x_4, x_5), \overset{F_3}{m_3}(x_4, x_5)\} \to M_4(x_4, x_5) \xrightarrow{\sum_{x_4}} m_4(x_5)$
5. x_5: $\prod\{m_5(x_5)\} \to M_5(x_5) \xrightarrow{\sum_{x_5}} Z$

Choosing a variable elimination order

- Complexity of VE $O(nm^w)$ where w is the maximum number of variables in any factor.
- Wrong elimination order can give rise to very large intermediate factors.
- Example: eliminating x_2 first will give a factor of size 4.
- Given an example where the penalty can be really severe (?) m^n vs $n m^2$
- Choosing the optimal elimination order is NP hard for general graphs.
- Polynomial time algorithm exists for chordal graphs.
  - A graph is chordal or triangulated if all cycles of length greater than three have a shortcut.
- Optimal triangulation of graphs is NP hard. (Many heuristics)

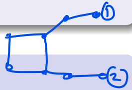
Finding optimal order in a triangulated graph





Theorem

Every triangulated graph is either complete or has at least two non-adjacent simplicial vertices. A vertex is simplicial if its neighbors form a complete set.



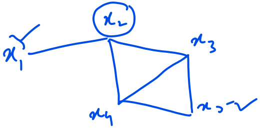
Proof.

In supplementary. (not in syllabus) □

Goal: find optimal ordering for $P(x_1)$ inference. x_1 has to be last in the ordering.

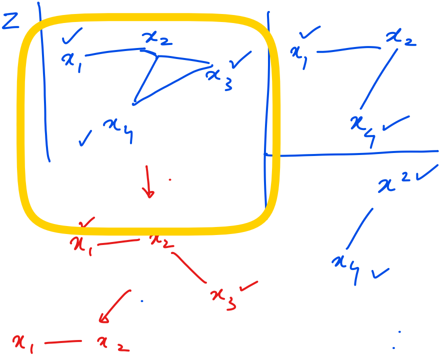
Input: Graph G. n = number of vertices of G
for $i = 2, \ldots, n$ **do**
  π_i = pick any simplicial vertex in G other than 1.
  remove π_i from G
end for
Return ordering $(\pi_1, \pi_2, \ldots, \pi_{n-1})$

Example



$x_5 \quad x_3 \quad x_1 \quad x_2 \quad x_4$

$x_5 \quad x_4 \quad x_3 \quad x_2 \quad x_1$



Reusing computation across multiple inference queries

Given a chain graph with potentials $\psi_{i,i+1}(x_i, x_{i+1})$, suppose we need to compute all n marginals $\boxed{P(x_1), \ldots, P(x_n)}$. $P(x_j)$

Invoking variable elimination algorithm n times for each x_i will entail a cost of $(n \times \boxed{nm^2})$. Can we go faster by reusing work across computations?

$x_1 - x_2 - x_3 - x_4 - x_5$

ψ_{34} ψ_{45}

x_5 x_4 x_3 x_2 m_3

(1) $P(x_1)$

$P(x_2)$

$\sum_{x_1} \psi_{12}(x_1, x_2)\, m_3(x_3)$

$x_1 \rightarrow m_3(x_3)$

(2) $P(x_4)$

$m_1(x_2)$ $m_5(x_4)$

$x_2 \rightarrow x_3 \rightarrow$

$\boxed{2T}$

Junction tree algorithm

- An optimal general-purpose algorithm for exact marginal/MAP queries
- Simultaneous computation of many queries
- Efficient data structures
- Complexity: $O(m^w N)$ $w=$ size of the largest clique in (triangulated) graph, m = number of values of each discrete variable in the clique. $\rightarrow$ linear for trees.
- Basis for many approximate algorithms.
- Many popular inference algorithms special cases of junction trees
  - Viterbi algorithm of HMMs
  - Forward-backward algorithm of Kalman filters