today: inference: graph Eliminate
sum-product alg.

graph elimination alg. (for inference in generic UGM)

- consider $p \in \mathcal{L}(G)$ (undirected) $\quad p(x) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \psi_c(x_c)$

say want to compute $p(x_F)$ for some $F \subseteq V$ "query nodes"

main trick: use distributivity of $\oplus$ over $\odot$ $\leadsto c \odot (a \oplus b) = c \odot a \oplus c \odot b$

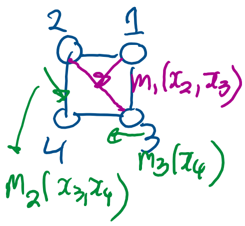
$$\sum_{x_1, x_2} f(x_1) g(x_2) = \left(\sum_{x_1} f(x_1)\right)\left(\sum_{x_2} f(x_2)\right)$$

[convince yourself]

more generally
$$\sum_{x_{1:n}} \prod_i f_i(x_i) = \prod_{i=1}^{n} \left(\sum_{x_i} f(x_i)\right)$$

$$O(k^n n) \longrightarrow O(k \cdot n)$$

x_i takes k values



$F = \{4\}$

$$p(x_4) = \frac{1}{Z} \sum_{x_1, x_2, x_3} \psi(x_1, x_2)\, \psi(x_2, x_4)\, \psi(x_1, x_3)\, \psi(x_3, x_4)$$

$$= \frac{1}{Z}\left[\sum_{x_3} \psi(x_3, x_4)\left[\sum_{x_2} \psi(x_2, x_4) \underbrace{\left(\sum_{x_1} \psi(x_1, x_2)\, \psi(x_1, x_3)\right)}_{m_1(x_2, x_3)}\right]\right]$$

"message"

$$\underbrace{\sum_{x_2} \psi(x_2, x_4)\, m_1(x_2, x_3)}_{m_2(x_3, x_4)}$$

$= \frac{1}{Z} m_3(x_4)$ $\hookrightarrow$ last message is proportional to marginal $p(x_4)$

$\sum_{x_4} m_3(x_4) = Z \qquad \boxed{p(x_4) = \frac{m_3(x_4)}{Z}} = \frac{m_3(x_4)}{\sum_{x_4'} m_3(x_4')}$

general alg: graph Eliminate

init: a) choose an elimination ordering s.t. F are the last nodes

b) put all $\psi_c(x_c)$ on "active list"

b) put all $\psi_c(x_c)$ on "active list"

"update" c) repeat, in order of variables to eliminate

1) (say x_i is variable to eliminate)

remove all factors from active list with x_i in it & take their product

i.e. $\prod_{\substack{\alpha \in \text{s.t.} \\ i \in \alpha}} \psi_\alpha(x_\alpha)$

2) sum over x_i to get a new factor $m_i(x_{S_i})$ (think as $\psi_{S_i}(x_{S_i})$)

i.e. S_i are all variables in these factors except i

get $m_i(x_{S_i}) \triangleq \sum_{x_i} \underbrace{\prod_{\substack{\alpha \text{ s.t.} \\ i \in \alpha}}}_{} \psi_\alpha(x_\alpha)$ new clique to sum over $S_i \cup \{i\}$

$S_i \triangleq \left(\bigcup_{\substack{\alpha \text{ s.t.} \\ i \in \alpha}} \alpha \right) \setminus \{i\}$

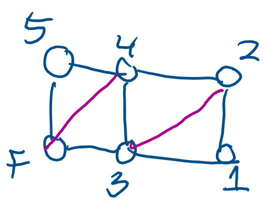
3) put back $m_i(x_{S_i})$ in active list $(\psi_{S_i}(x_{S_i}))$

"normalize": d) last product of factors has only $x_F \Rightarrow$ proportional $p(x_F)$

suppose $x_i \in \{0,1\}$ memory needed? $\approx 2^{\max_i |S_i|}$ (max size of active list $\leq |\mathcal{C}|$)

computational $= \underbrace{\left(2^{(\max_i |S_i| + 1)}\right)}_{\text{later, related "tree width" of a graph}} n$

(*) "augmented graph" $\rightarrow$ graph obtained by running graph Eliminate (for specific ordering) + keeping track of all edge added

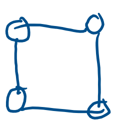


(*) note augmented graph obtained after graph eliminate is always

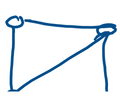
a triangulated graph

def.: graph with no cycle of size 4 or more that cannot be broken by a "chord"

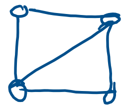
an edge between two non-neighboring nodes in a cycle



$\leftarrow$ not triangulated



$\leftarrow$ Δ-graph



← Δ-graph

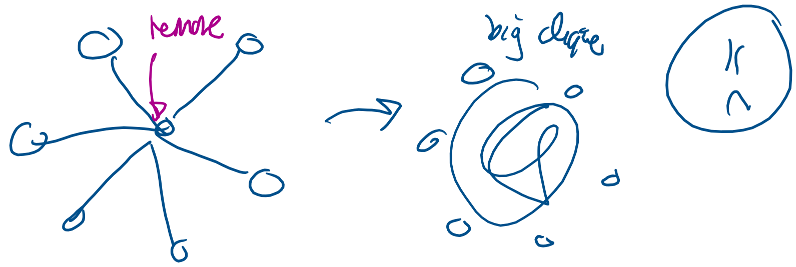
treewidth of a graph ≜ min over all possible elimination orderings { size of biggest clique in augmented graph − 1 }

↳ convention: treewidth(tree) = 1

both memory & running time of graph Eliminate is dominated by $2^{\text{size of biggest clique}}$

when using best ordering $\quad 2^{\text{treewidth}+1}$

not all orderings are good



bad news:

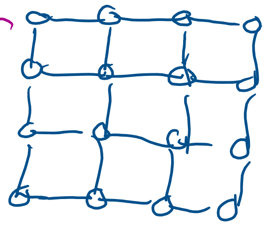
a) NP hard to compute treewidth of a general graph (or find best ordering)

b) NP hard to do (exact) inference in general UGM

$\Longrightarrow$ need approximate methods

example: treewidth of a grid $\approx \sqrt{|V|}$

side length $\approx$ treewidth

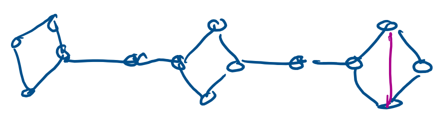


15h26 good news:

* inference is linear time for tree (treewidth 1) ("sum-product alg.")
  ↳ $|V|+|E|$
  (HMM, Markov chain)

* efficient for "small treewidth" graph

  → use junction tree alg.



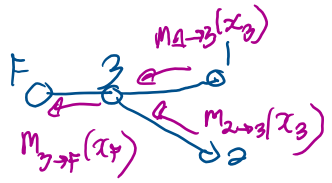
inference on trees

graph Eliminate on a tree

inference on trees

graph Eliminate on a tree

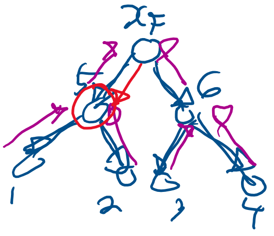
→ good order is to eliminate leaves first



$$p(x) = \frac{1}{Z} \prod_i \psi_i(x_i) \prod_{\{i,j\}\in E} \psi_{ij}(x_i, x_j)$$

(assuming F is singleton)

Order: make a directed tree by using x_F as a root



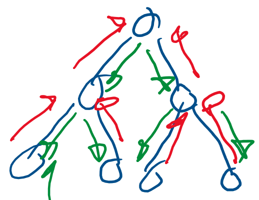
$$\boxed{m_{i\to j}(x_j) = \sum_{x_i} \psi_i(x_i)\psi_{ij}(x_i,x_j) \underbrace{\prod_{k\in \text{children}(i)} m_{k\to i}(x_i)}_{\text{new factors containing } i \text{ in the active list}}}$$

i: child, j: parent

sum-product alg. (for trees)

alg. to get all node/edge marginals cheaply by storing (caching) & re-using messages (dynamic programming)

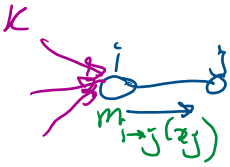
"collect phase"



"distribute phase"

goal: $\{i,j\}\in E$, compute $m_{i\to j}(x_j)$
$m_{j\to i}(x_i)$

rule: i can only send messages to neighbor j when it has received all messages from other neighbors



$$\boxed{m_{i\to j}(x_j) \triangleq \sum_{x_i} \psi_i(x_i)\psi_{ij}(x_i,x_j) \prod_{k\in N(i)\setminus\{j\}} m_{k\to i}(x_i)}$$

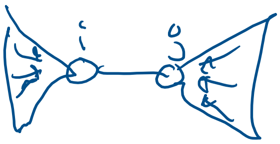
at end

(node marginal) $p(x_i) \propto \underbrace{\psi_i(x_i) \prod_{j\in N(i)} m_{j\to i}(x_i)}$

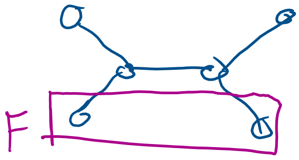
normalize $Z = \sum_{x_i} (\ \text{"}\ )$

(edge marginal)

(edge marginal)



$$p(x_i, x_j) = \frac{1}{Z} \psi_i(x_i)\psi_j(x_j)\psi_{ij}(x_i,x_j) \cdot \prod_{k\in N(i)\setminus\{j\}} m_{k\to i}(x_i) \cdot \prod_{k'\in N(j)\setminus\{i\}} m_{k'\to j}(x_j)$$



for this marginal need to use graph Elimination

<u>sum-product schedules</u>

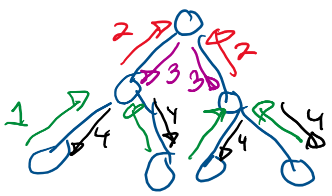
a) above, collect/distribute schedule

b) (flooding) parallel schedule

1) initialize all $m_{i\to j}(x_j)$ to uniform dist. $\forall (i,j)$ s.t. $\{i,j\} \in E$ (j,i)

2) at every step (in parallel) compute $m_{i\to j}^{(new)}(x_j)$ <u>as if</u> the neighbors were correctly computed

→ one can prove that after "diameter of tree" # of steps all messages are correctly computed (for a tree) (and are fixed to a fixed point)



<u>loopy belief propagation</u> (loopy BP): approximate inference for graphs with <u>cycles</u>

$$m_{i\to j}^{new}(x_j) = \left(m_{i\to j}^{old}(x_j)\right)^{1-\alpha} \left(\sum_{x_i} \psi_i(x_i)\psi_{ij}(x_i,x_j) \prod_{k\in N(i)\setminus\{j\}} m_{k\to i}^{old}(x_i)\right)^{\alpha}$$

$\alpha \in [0,1]$ step-size

"damping"

⊛ this gives exact answer on trees (fixed pt. → yields correct marginals)

✱ on (not too loopy) graph → "nice" approximate solution