Lecture 14 - scribbles

Tuesday, October 22, 2019 14:25

today: inference – graph eliminate
– sum-product

graph elimination alg. (for inference)

- consider $p \in \mathcal{G}(\mathcal{G})$ ($\mathcal{G}$ undirected) $\quad p(x) = \frac{1}{Z} \prod_{C \in \mathcal{G}} \psi_C(x_C)$

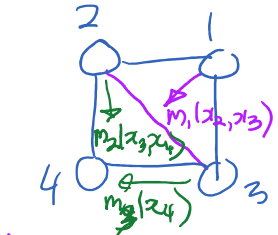
say want to compute $p(x_F)$ for $F \subseteq V$ "query nodes"

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

$$\sum_{x_1, x_2} f(x_1) g(x_2) = \left(\sum_{x_1} f(x_1)\right)\left(\sum_{x_2} g(x_2)\right)$$ [convince yourself]

↳ first term $\sum_{x_2} f(x_1 = 1) g(x_2)$

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



$F = \{4\}$

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

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

$\sum_{x_1} \psi(x_1, x_2)\psi(x_1, x_3) = m_1(x_2, x_3)$ "message"

↳ stored as a table

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

$$= \frac{1}{Z} m_3(x_4)$$

last message is proportional to marginal $p(x_4)$

$\sum_{x_4} m_3(x_4) = Z$

general alg: GraphEliminate

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

b) put all $\psi_C(x_C)$ on "active list"

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

"update" c) repeat in order of variables to eliminate
(say x_i is variable to eliminate)

loop

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

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

2) sum x_i to get a new factor $m_i(x_{S_i})$

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

get $m_i(x_{S_i}) \triangleq \sum_{x_i} \prod_{\alpha \text{ s.t. } i \in \alpha} \psi_\alpha(x_\alpha)$

new clique to sum over $S_i \cup \{i\}$

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

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

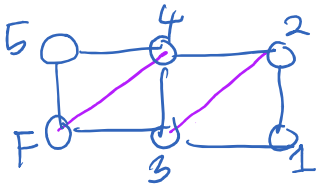
"normalize" d) last factor left has only $x_F \Rightarrow$ proportional to $p(x_F)$

memory needed? $\simeq 2^{\max_i |S_i|} \cdot (\#\text{ of factors})$

computational cost $\simeq 2^{\max_i |S_i|+1} \cdot n$

↳ later, related "treewidth" of a graph

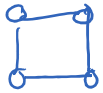
(*) "augmented graph" → graph obtained by running graph eliminate (for a fixed ordering) + keeping track of all edges added



"augmented graph" after graph Eliminate is always a triangulated graph

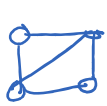
def.: graph with no cycle of size 4 or more that cannot be broken by a "chord"

↳ edge between two non-neighboring nodes in cycle



not triangulated

triangulated



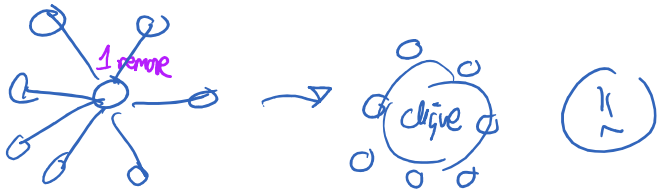
treewidth of a graph $\triangleq$ min { size of biggest clique formed -1 }
over all elimination ordering

→ convention treewidth(tree) = 1

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

best ordering gives $\approx 2^{\text{treewidth}+1}$

not all orderings are good



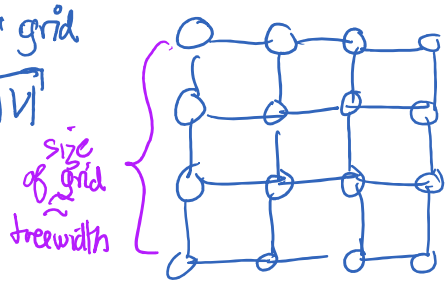
bad news:

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

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

$\Rightarrow$ need approximate methods

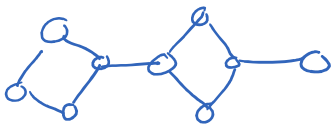
Example: treewidth of a grid $\approx \sqrt{|V|}$



Good news: $|V|+|E|$

* inference is linear time for trees (treewidth = 1) ("sum-product alg.")
(HMM, Markov chain)

* efficient for "small treewidth graph"
use → junction tree alg.



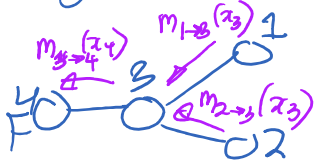
15h36

Inference on trees

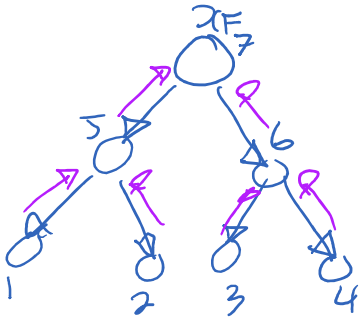
graphEliminate on a tree

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

good order: eliminate leaves first



order: make a directed tree by using x_F as a root
(singleton)



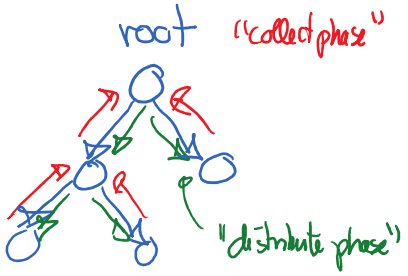
$$m_{i \to j}(x_j) = \sum_{x_i} \psi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in \text{children}(i)} m_{k \to i}(x_i)$$

i: child, j: parent

"how factors containing i on active list"

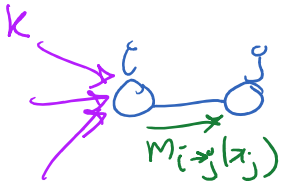
sum-product alg. (for trees)

get all marginals cheaply by sharing (caching) & re-using messages (dynamic programming)



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

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



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

$N(i)$: neighbors

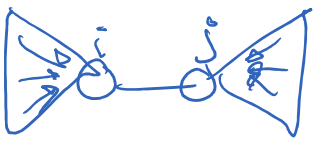
at end (node i)

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

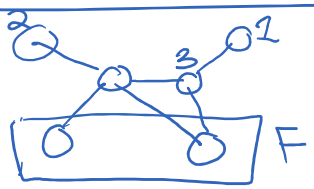
at end (node marginal) $p(x_i) \propto \prod_{j \in N(i)} m_{j \to i}(x_i) \cdot \psi_i(x_i)$

normalization $Z = \sum_{x_i} (\quad)$

(edge marginal)



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



here need graph Eliminate

Sum-product schedules:

a) above, distribute/collect schedule

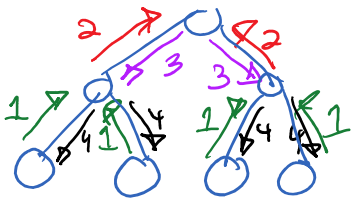
b) (flooding) parallel schedule:

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

2) at every step (in parallel) compute $m_{i \to j}^{new}(x_j)$

as if the neighbor messages were already correctly computed

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



Loopy Belief Propagation (Loopy BP): approximate inference for graph with cycles.

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

$\alpha \in [0,1]$ "damping"

* this gives exact answers on trees (fixed pt. yields correct marginals)
* on (not loopy) graphs → approximate solution

Getting conditionals:

$$p(x_i \mid \overline{x}_E) \propto p(x_i, \overline{x}_E)$$

indicates values we are conditioning on

keep this fixed during marginalization for each $j \in E$

(formal trick): redefine $\tilde{\psi}_j(x_j) \triangleq \psi_j(x_j) \cdot \delta(x_j, \overline{x}_j)$

Kronecker-delta

ex. $\delta(a,b) \triangleq \begin{cases} 1 & \text{if } a=b \\ 0 & \text{o.w.} \end{cases}$

computing $m_{j \to i}(x_i)$ $\sum_{x_j} \tilde{\psi}_j(x_j)\, \text{shift}(x_j, x_i)$

$= \psi_j(\overline{x}_j)\, \text{shift}(\overline{x}_j, x_i)$

at the end, result of sum-product will give $p(x_i, \overline{x}_E) = \frac{1}{Z} \psi_i(x_i) \prod_{k \in N(i)} m_{k \to i}(x_i)$

renormalize over x_i to get $p(x_i \mid \overline{x}_E)$