Lecture 13 - scribbles

Tuesday, October 17, 2017
14:24

today: inference : elimination alg.
sum-product "

Inference:

present for UGM

make DGM $\rightsquigarrow \in$ UGM via moralization

ie. DGM: $p(x) = \prod_i p(x_i | x_{\pi_i})$

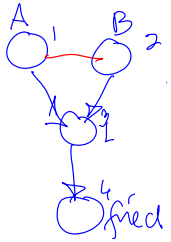
moralization: $C = \{i\} \cup \pi_i$

(UGM): $= \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(x_C)$ $\quad \psi_C(x_C) \triangleq p(x_i | x_{\pi_i})$

$Z = 1$

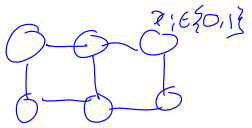
$\psi_C : \prod_{i \in C} \Omega_{x_i} \rightarrow \mathbb{R}_+$

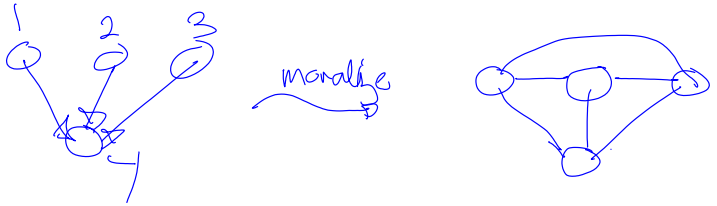
DGM:



$\psi_1(x_1)$
$\psi_2(x_2)$
$\psi_{\{1,2,3\}}(x_1, x_2, x_3) = p(x_3 | x_1, x_2)$

Ising model:





inference: want to compute:

a) marginal: $p(x_F)$ for some $F \subseteq V$

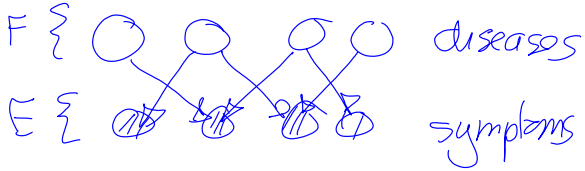
b) conditional: $p(x_F | x_E)$ ("evidence")

query → usually 1 or 2 nodes

c) for UGM: partition function $Z = \sum_{x_V} \prod_{C \in \mathcal{C}} \psi_C(x_C)$ $C \subseteq V$

why? whole pt. of graph model is getting a model for $p(x_F | x_E)$ (query, observation)

- missing data
- prediction
- "latent cause"



latent cause: $p(x_{cause} | x_{obs.})$

prediction: $p(x_{future} | x_{past})$

missing data: $p(x_{unobs} | x_{obs.})$ → in vision:

also related inference & $\operatorname{argmax}_{x_F} p(x_F | x_E)$ ↳ could be big

inference is also needed during estimation (parameter fitting MLE)

(e.g. during E-step $p(z|x)$)

graph elimination alg. (for inference)

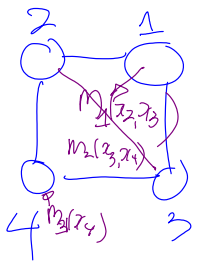
- consider UGM $p(x) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(x_C)$

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

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

$$\sum_{x_1,x_2} f(x_1)g(x_2) = \left(\sum_{x_1} f(x_1)\right)\left(\sum_{x_2} g(x_2)\right) \qquad \text{[conv. yourself]}$$

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



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

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

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

(stored as table)

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

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

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

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

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"

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

(say x_i is variable to eliminate)

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

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

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

ie. S_i all variables in those factors except x_i

get $m_i(x_{S_i}) \triangleq \sum_{x_i} \prod_{\alpha} \psi_\alpha(x_\alpha)$

$S_i = (\bigcup_{\alpha \text{ s.t. } i \in \alpha} \alpha) \setminus \{i\}$ new clique to sum over $S_i \cup \{i\}$

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

new table to store

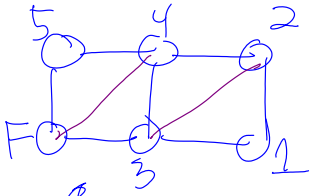
("normalize") 4) last factor left has only x_F $\Rightarrow$ proportional to $p(x_F)$

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

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

clique tree:

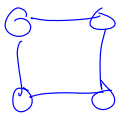
cliques formed during graph eliminate



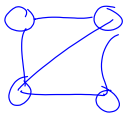
↑ "augmented graph after graph Eliminate

(*) running graph Eliminate + keep track of all edges added yields a triangulated graph

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



← not triangulated


← triangulated

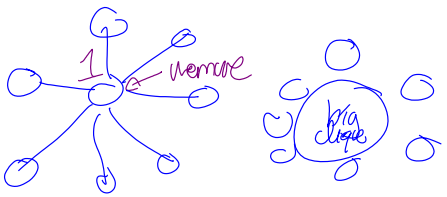
treewidth of graph = min elimination orderings { size of biggest clique formed − 1 }

↳ convention treewidth(tree) = 1

both memory & running time of graph Eliminate is dominated by $2^{\text{size of biggest clique formed} + 1}$

best order gives $2^{\text{treewidth} + 2}$

not all orderings are good:

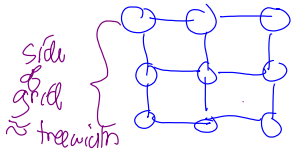


bad news:

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

b) NP hard to do inference in general UGM
⇒ need approximate methods

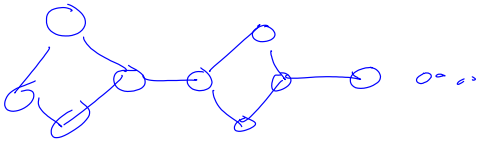
example: treewidth of a grid $\approx \sqrt{|V|}$



good news:

- inference in linear time ($|V| + |E|$) for trees (treewidth = 1)
  (HMM, Markov chain)
- efficient for "small treewidth graph"

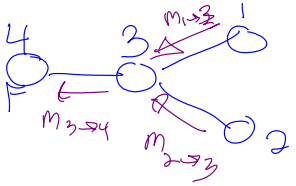
- efficient for "small treewidth graph"
  ↝ use junction tree alg.



Inference on a tree:

idea is to eliminate leaves first

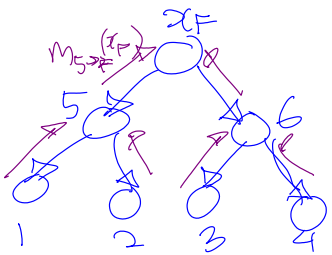
$$p(x) = \frac{1}{Z} \prod_i \psi_i(x_i) \prod_{\{i,j\} \in E} \psi_{ij}(x_i, x_j)$$



$$p(x_4) = \frac{\psi_4(x_4)}{Z} \sum_{x_3} \psi_3(x_3)\psi(x_4,x_3) \sum_{x_2} \psi(x_2)\psi(x_3,x_2) \underbrace{\sum_{x_1} \psi(x_1)\psi(x_1,x_3)}_{m_{1\to 3}(x_3)}$$

$$\underbrace{\qquad\qquad\qquad\qquad}_{m_{2\to 3}(x_3)}$$

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



start from leaves up to root

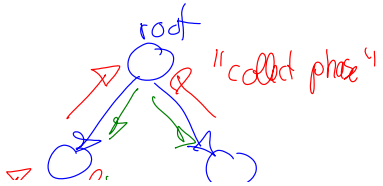
$$m_{i\to j}(x_j) = \sum_{x_i} \psi_i(x_i)\psi_{ij}(x_i,x_j) \prod_{k \in \text{children}(i)} m_{k\to i}(x_i)$$

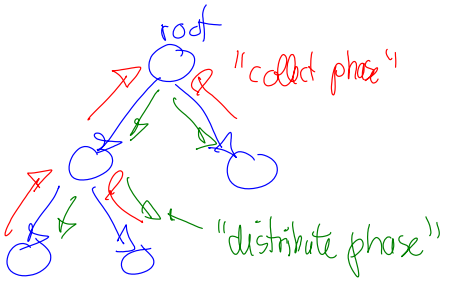
i: child, j: parent

$k \in$ children (i): new factors containing i on active list

Sum-product alg (for trees)

get all marginals cheaply by storing & reusing messages (dynamic programming)



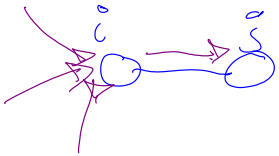


(dynamic programming)

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

$m_{j \to i}(x_i)$

rule: i can only send message to neighbor j $(m_{i \to j}(x_j))$
when it has received all messages from other neighbors

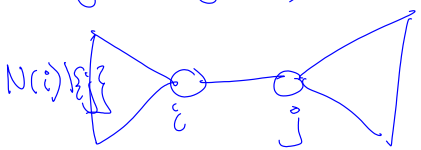


$$m_{i \to j}(x_j) = \sum_{x_i} \psi_i(x_i)\, \psi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus \{j\}} m_{k \to i}(x_i)$$

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

$$Z = \sum_{x_i} \left(\quad \right)$$

(edge marginal)



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