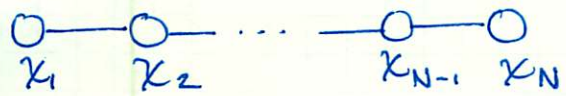
1 491-L7 Inf GM2



$\psi_{ij} = \psi_{ij}(x_i, x_j)$

$p(\underline{x}) = \frac{1}{Z} \psi_{12} \psi_{23} \cdots \psi_{N-1,N}$ full joint distr.

each x_i has K states

$\Rightarrow$ the potential function $\psi_{n-1,n}(x_{n-1}, x_n)$ is a $K \times K$ table.

$\Rightarrow$ joint distr. $p(\underline{x})$ has $(N-1)K^2$ params

$\boxed{?}$ How do we compute the marginal distri $p(x_n)$?

- Marginalize over joint:

$$p(x_n) = \sum_{x_1} \cdots \sum_{x_{n-1}} \sum_{x_{n+1}} \cdots \sum_{x_N} p(\underline{x})$$

$$= \sum_{x_i \in G \setminus n} p(\underline{x})$$

- Each x_n has K states $\Rightarrow K^N$ values for $\underline{x}$

$\Rightarrow$ naïve calculation of $p(x_n)$ is exponential in N.

- Can do better, just like variable elimination in prev. lecture for DAGs

By inspecting sum $\psi_{N-1,N}(x_{N-1}, x_N)$ is the only term that depends on x_N

$\Rightarrow$ sum this first $\sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N) = f(x_{N-1})$

- Now this is the only term that depends of x_{N-1}, so compute $\sum_{x_{N-1}} \psi_{N-2,N-1}(x_{N-2}, x_{N-1}) = f(x_{N-2})$

- and so on...

- Similarly: $\psi_{1,2}(x_1, x_2)$ is the only one that depends on x_1
  - so perform this: $\sum_{x_1} \psi_{1,2}(x_1, x_2) = f(x_2)$
  - then compute: $\sum_{x_2} \psi_{2,3}(x_2, x_3) = f(x_3)$
  - and so on...

- Each sum removes a variable, which is equiv. to removing a node from the graph

- Can group the potentials and summations this way as:

$$p(x_n) = \frac{1}{Z} \underbrace{\left[\sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \cdots \left[\sum_{x_2} \psi_{2,3}(x_2, x_3)\left[\sum_{x_1} \psi_{1,2}(x_1, x_2)\right]\right]\cdots\right]}_{\mu_\alpha(x_n)}$$

only var. left

$$\times \underbrace{\left[\sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1}) \cdots \left[\sum_{x_{N-1}} \psi_{N-2,N-1}(x_{N-2}, x_{N-1})\left[\sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N)\right]\right]\right]}_{\mu_\beta(x_n)}$$

$$ab + ac = a(b+c)$$

$$\frac{1}{Z}\left[\sum_{x_{n-1}} \psi_{n-1,n} \cdots \sum_{x_2} \psi_{2,3}\left[\sum_{x_1} \psi_{1,2}\right]\cdots\right]$$

$$\times \left[\sum_{x_{n+1}} \psi_{n,n+1} \cdots \left[\sum_{x_{N-1}} \psi_{N-2,N-1}\left[\sum_{x_N} \psi_{N-1,N}\right]\right]\cdots\right]$$

? What's the cost of computing $\mu_\alpha(x_n)\mu_\beta(x_n)$?

- How many summations? Over everything but $x_n \Rightarrow N-1$
- How many terms in each? Each var has K states.

Each Ψ is a function of two vars, e.g.

$$\Psi_{12}(x_1, x_2) \Rightarrow K \times K \text{ table}$$

$$\sum_{x_1} \Psi_{1,2}(x_1, x_2)$$ sum over K #'s for each val

$\Rightarrow$ K^2 cost

$$\boxed{\Rightarrow \text{Computing } p(x_n) \text{ is } (N-1)K^2 = O(NK^2)}$$

Naïve approach was exponential in N.

- This only works because the graph is much less than fully connected.
- Fully connected $\Rightarrow$ need to use full joint

~~Message passing:~~ ~~$p(\underline{x}) = \frac{1}{Z}\mu_\alpha(x_n)\mu_\beta(x_n)$~~

Message passing:

$$P(x) = \frac{1}{Z} \mu_\alpha(x_n) \mu_\beta(x_n)$$

Can be evaluated recursively:

$$\mu_\alpha(x_n) = \sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \left[\sum_{x_{n-2}} \cdots \right]$$

$$= \sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \mu_\alpha(x_{n-1})$$

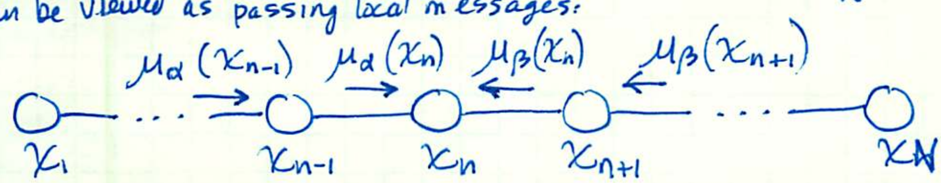
base case: $\mu_\alpha(x_2) = \sum_{x_1} \psi_{1,2}(x_1, x_2)$

$$\mu_\beta(x_n) = \sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1}) \left[\sum_{x_{n+2}} \cdots \right]$$

$$= \sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1}) \mu_\beta(x_{n+1})$$

base case: $\mu_\alpha(x_{N-1}) = \sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N)$

Can be viewed as passing local messages:



Each has this form. Outgoing message is a point wise multiplication ~~over~~ of the incomming message and the local potential fn and summing over the different vals of the node var.

$$\underbrace{\mu_\alpha(x_n)}_{\text{outgoing}} = \sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \overbrace{\mu_\alpha(x_{n-1})}^{\text{incomming}}$$

$$\mu_\beta(x_n) = \sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1}) \mu_\beta(x_{n+1})$$

⑤ 491 - L7 Inf. GM2

- What about Z? $p(x_n) = \frac{1}{Z}\mu_\alpha(x_n)\mu_\beta(x_n)$

$$Z = \sum_{x} \prod_{C} \psi_c(\underline{x}_c)$$

in general, here cliques are just pairwise.

- Now we have:

$$p(x_n) = \frac{1}{Z}\mu_\alpha(x_n)\mu_\beta(x_n)$$

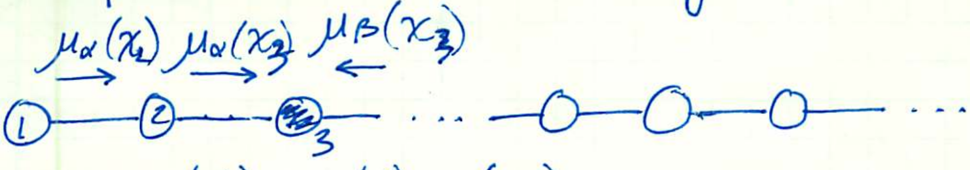
$$\text{So} \quad Z = \sum_{x_n} \mu_\alpha(x_n)\mu_\beta(x_n)$$

- Only depends of x_n, so computation is $O(K)$.

What about computing $p(x_n)$ for other nodes?

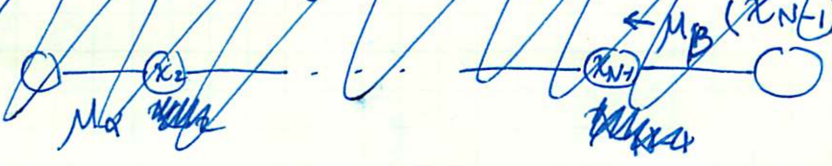
If we do the procedure above to every node, it's $O(N \times NK^2)$
$= O(N^2K^2)$

$\boxed{?}$ Is this optimal in terms of efficiency?



$\boxed{A:}$ No it's very redundant.

~~$\boxed{Idea:}$ Store messages so they don't have to be recomputed.~~



$\boxed{\text{Idea:}}$ Store messages for efficient computation of marginals.

$\boxed{?}$ What's the cost now? Was $O(N \times NK^2)$

Messages only pass once in each direction.

Now: $O(2 \times NK^2) = O(NK^2)$

$\boxed{\text{Only twice as much computation to compute every marginal!}}$

$\boxed{?}$ What about Z, the normalization constant?

A: Only needs to be computed once, it's the same for everything.

$\boxed{?}$ What if some of the nodes are observed?

~~Eg. $\mu_\alpha(x_{n+1}) = \sum_{x_{n}} \psi_{n,n+1}(x_n, x_{n+1})\, \mu_\alpha(x_n)$~~

If $x_n = a$ is given, then this is equivant to setting the joint distri. $p(\underline{x}) = a$ for x_n.

This means all values in the definition of $p(\underline{x})$ where x_n could vary are now fixed to a. Same for computation of Z.

$$\mu_\alpha(x_{n+1}) = \cancel{\sum_{x_n = a}} \psi_{n,n+1}(x_n = a, x_{n+1})\, \mu_\alpha(x_n = a)$$

$\boxed{3}$ What about ~~...~~ computing other quantities?

Eg. $p(x_{n-1}, x_n)$? (two neighboring nodes on chain)

$$p(x_{n-1}, x_n) = \frac{1}{Z} \sum_{\underline{x} \setminus x_n \setminus x_{n-1}} p(\underline{x})$$

$$= \sum_{x_1} \cdots \sum_{x_{n-2}} \sum_{x_{n+1}} \cdots \sum_{x_N} p(\underline{x})$$

i.e. sum over all but x_{n-1} & x_n

Now it proceeds as before, the only difference, however, is that we're not summing over x_{n-1}. Therefore we get:

$$p(x_{n-1}, x_n) = \frac{1}{Z} \mu_\alpha(x_{n-1}) \psi_{n-1,n}(x_{n-1}, x_n) \mu_\beta(x_n)$$

(refer to eqn that defines $\mu_\alpha(x_n)$ & $\mu_\beta(x_n)$)

→ This means its easy to compute joint distributions over sets of vars, once we've computed μ_α & μ_β for every node.

This gives us parametric forms for clique potentials,

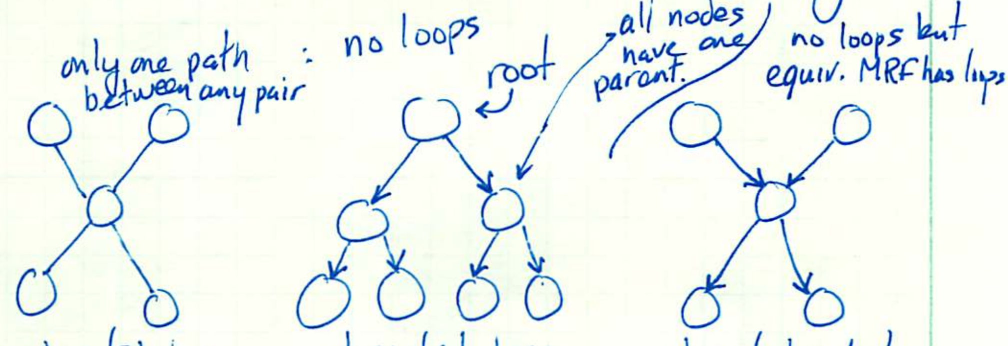
→ or the conditional distributions, if we start with a directed graph.

Trees

We can do exact inference on a chain in linear time.

$\boxed{?}$ Can we do it on other types of graphs?

We will show that inference can be done efficiently on trees:



undirected tree — directed tree — directed poly tree

Sum-product algorithm : a generalization of the message passing alg. which provides an efficient framework for exact inference on tree-structured graphs.

Sum product alg applies to : undirected trees
directed trees
poly trees

First: introduce a new graphical construction:
factor graphs

Factor graphs

Both directed and undirected graphs decompose joint pdfs into products:

$$p(\underline{x}) = \prod_i p(x_i \mid pa(x_i))$$ cond. prob of node given parents

$$p(\underline{x}) = \frac{1}{Z} \prod_C \psi_C(\underline{x}_C)$$ clique potential functions

More generally write pdf as a product of factors:

$$p(\underline{x}) = \prod_S f_S(\underline{x}_S) \qquad \underline{x}_S = \text{some subset of vars}$$

for DGs f_S = local conditional distr.

UGs f_S = potential fns over maximal cliques

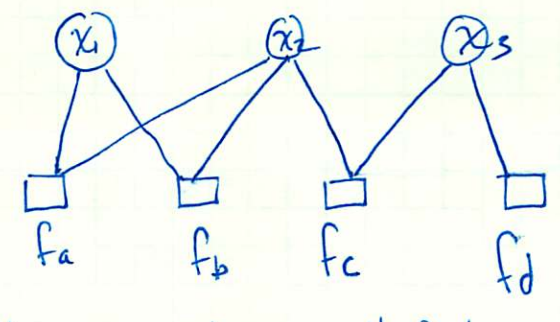
Note: Z isn't represented explicitly but could be defined as a factor that doesn't depend on the vars.

Examples:

note. these are two factors for same vars

$$p(\underline{x}) = f_a(x_1, x_2)\, f_b(x_1, x_2)\, f_c(x_2, x_3)\, \underbrace{f_d(x_3)}$$

this is only over one var.

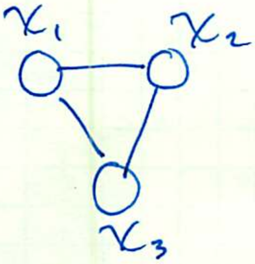


nodes

factors

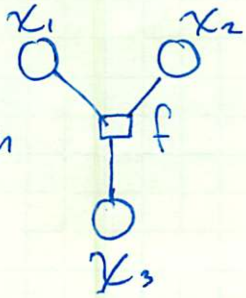
Factor graphs keeps all factors explicit.

MRF

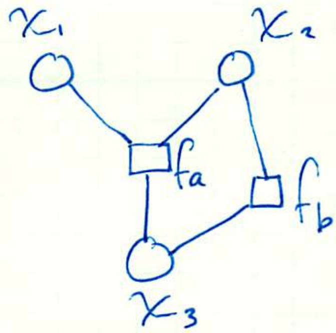


$$p(\underline{x}) = \frac{1}{Z}\psi(x_1, x_2, x_3)$$

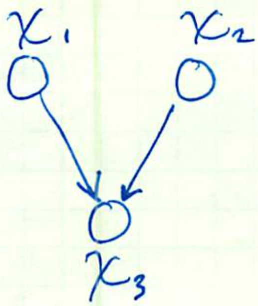
Factor graph



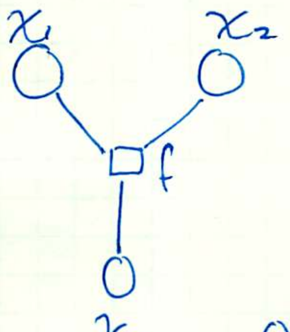
could add:



$$f_a(x_1, x_2, x_3)\, f_b(x_2, x_3) = \psi(x_1, x_2, x_3)$$

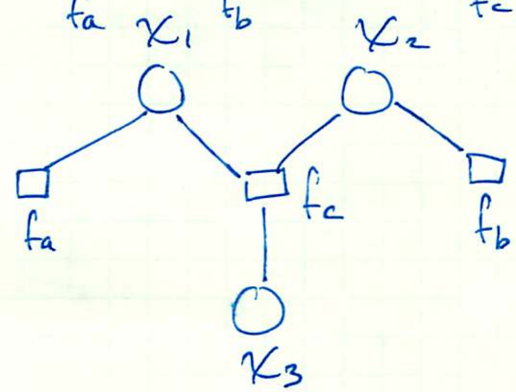


$$p(\underline{x}) = p(x_1)\,p(x_2)\,p(x_3 \mid x_1, x_2)$$



$$f(x_1, x_2, x_3) = \underbrace{p(x_1)}_{f_a}\underbrace{p(x_2)}_{f_b}\underbrace{p(x_3 \mid x_1, x_2)}_{f_c}$$

OR:



Factor graphs are bipartite:

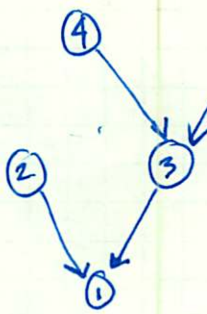
- two kinds of nodes
- all links are between nodes of opposite types

Could make all factor graphs like this

Factor graphs (cont'd):

For UGs:
3) $f_s(x_s) =$ clique potentials $\psi_c(\underline{x}_c)$
1) factor nodes are maximal cliques x_s.
2) var nodes are same

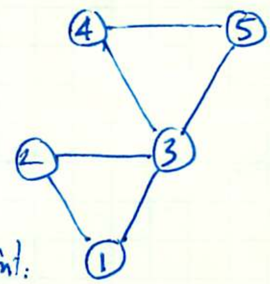
For a polytree:



Could convert to UG:

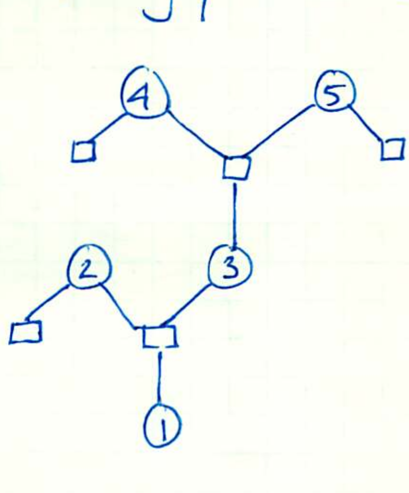
$$p(\underline{x}) = p(x_4)\,p(x_5)\,p(x_3 | x_4, x_5) = \psi_2$$
$$\times\, p(x_2)\,p(x_1 | x_2, x_3) = \psi_1$$

The equiv. UG:
has loops



$$p(\underline{x}) = \frac{1}{Z}\,\psi_1(x_1, x_2, x_3) \times \psi_2(x_3, x_4, x_5)$$

But the equiv. factorgraph doesn't:



Note: It should be clear that the factorization f_s does not correspond to any cond. indep. properties.