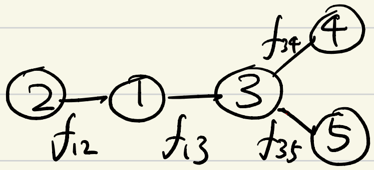
* Elimination Algorithm on a tree T.
- Message-passing on T
- Parallelized
- Messages can be re-used

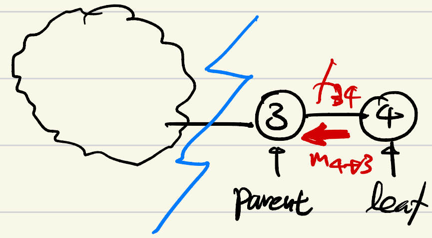
$\Big\} \Longrightarrow$ Sum-Product Algorithm.



elimination ordering $(4, 5, 3, 2, \underline{1})$.

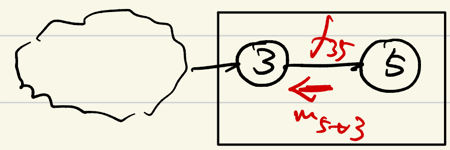
$$\mathbb{P}(X_1) = \sum_{X_2,X_3,X_5} f_{12}\cdot f_{13}\cdot f_{35} \underbrace{\sum_{X_4} f_{34}(X_3,X_4)}_{m_{4\to3}(X_3)}$$

message/belief



meaning:
If the whole graph $③ \xrightarrow{f_{34}} ④$
then $m_{4\to3}(X_3) \propto P(x_3)$

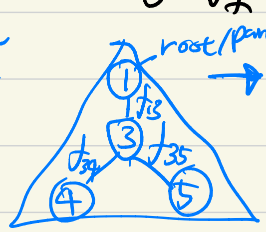
$$= \sum_{X_2,X_3} f_{12}\cdot f_{13}\; m_{4\to3}(X_3)\cdot \underbrace{\sum_{X_5} f_{35}(X_3,X_5)}_{m_{5\to3}(X_3)}$$



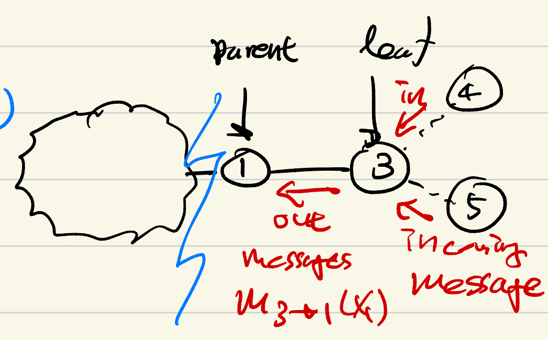
- note that $m_{5\to3}(X_3)$ & $m_{4\to3}(X_3)$ can be computed in parallel.

$$= \sum_{X_2} f_{12}\cdot \underbrace{\sum_{X_3} f_{13}(X_1,X_3)\times m_{5\to3}(X_3)\times m_{4\to3}(X_3)}_{m_{3\to1}(X_1)}$$

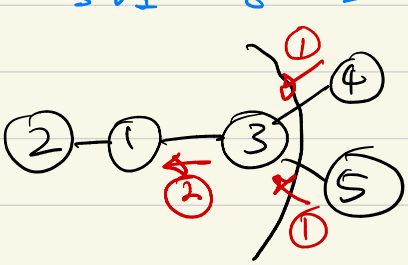
* for $\tilde{G}$



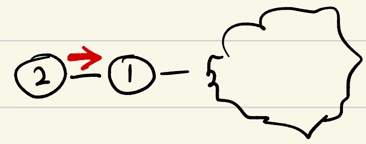
$\rightarrow \tilde{P}(X_1) \propto m_{3\to1}(X_1)$



* $m_{3\to1}$ requires $m_{5\to3}$, $m_{4\to3}$

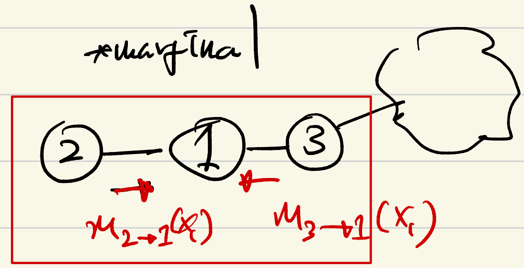


$$= \frac{1}{Z}\, m_{3\to1}(x_1)\cdot \underbrace{\sum_{x_2} f_{12}(x_1,x_2)}_{m_{2\to1}(x_1)}$$

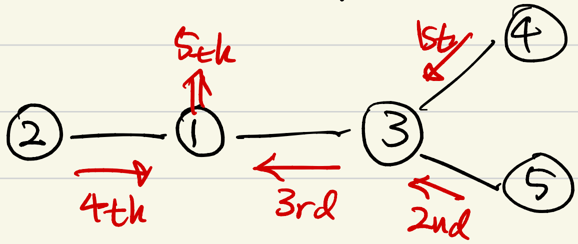


$$= \frac{1}{Z}\, \underbrace{m_{3\to1}(x_1) \times m_{2\to1}(x_1)}_{m_1(x_1)\ :\ \text{marginal}}$$

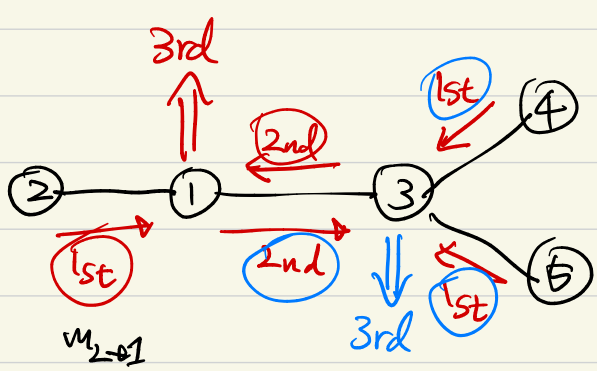
$$P(x_1) = \frac{m_1(x_1)}{\sum_{x_1'} m_1(x_1')}$$



* Message Passing algorithm on a graph.



elimination (4, 5, 3, 2, 1)

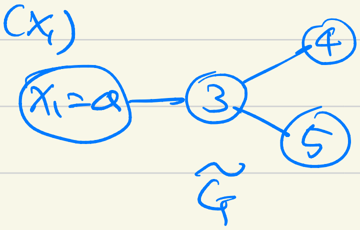


· all the messages can be re-used for computing other marginals.

if you want $P(x_1)$, & $P(x_3)$, for example.

* Decimation: $P(X_1) \longrightarrow P(X_4 \mid X_1 = a)$

Compute $P(X_1, X_4) \longrightarrow P(X_1)$



$\longrightarrow \tilde{P}(X_4) = P(X_4 \mid X_1 = a)$

* Serial Elimination Algorithm on Tree for $P(x_1)$

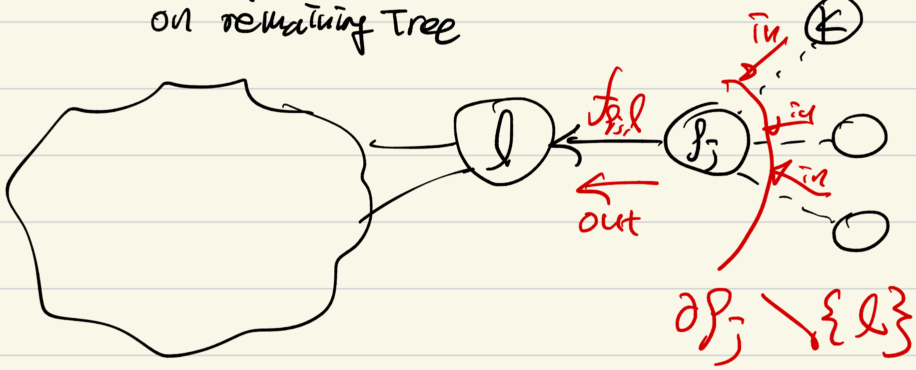
- ordering $\rho = (\rho_1, \cdots, \rho_n = x_1)$ s.t. any node is a leaf on remaining graph

- for $j = 1 \cdots, n$ do

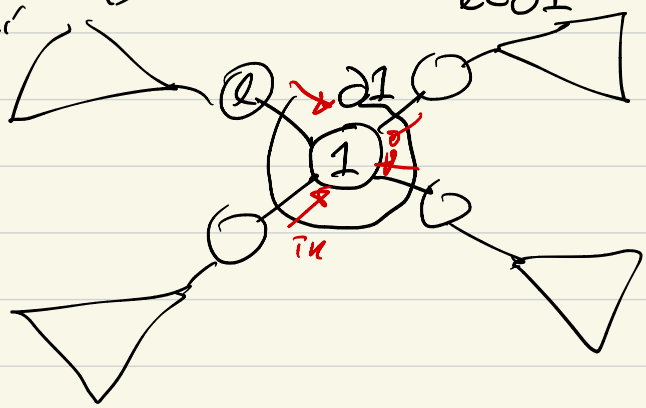
$$m_{\rho_j \to \ell}(x_\ell) = \sum_{x_{\rho_j}} f_{\rho_j, \ell}(x_{\rho_j}, x_\ell) \cdot \prod_{k \in \partial \rho_j \setminus \{\ell\}} m_{k \to \rho_j}(x_{\rho_j})$$

ρ_j: leaf

ℓ: Parent of j uniquely defined on remaining Tree



in the end. $P(x_1) = \dfrac{m_1(x_1)}{\sum_{x_1'} m_1(x_1')}$, $m_1(x_1) = \prod_{\ell \in \partial 1} \cdot m_{\ell \to 1}(x_1)$.



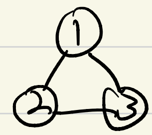
* Sum-Product Algorithm = Belief Propagation.
  - applied to any G non-tree.
  - only for Pairwise MRF
  - Exact if G is a tree, good approx if G is non-tree
    HW2.

Def. Pairwise MRF on $G=(V,E)$ is defined as

$$P(x) = \frac{1}{Z} \prod_{(i,j)\in E} f_{ij}(x_i, x_j)$$

general MRF

$f(x_1, x_2, x_3)$



pairwise MRF

$\neq$

$P(x) = \frac{1}{Z} f_{12}(x_1, x_2) f_{23}(x_2, x_3) \times f_{13}(x_1, x_3)$

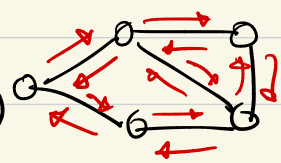
* Sum-Product Algorithm
  - input: $G=(V,E)$, $\{f_{ij}(x_i,x_j)\}_{(i,j)\in E}$, $\mathcal{X}$, T: # of iterations
  - Output: $\{\hat{P}(x_i)\}_{i=1}^{n}$
  - Initialize messages: $\{ m_{i\to j}(x_j), m_{j\to i}(x_i) \}_{(i,j)\in E} \overset{\text{Random}}{=} \mathbb{1}$

$[\;]\}^{|\mathcal{X}|}$ $\quad 2\cdot|E|$

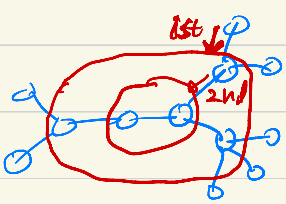
  - for $t=1, \dots, T$

    for all $(i,j)\in E$ & $(j,i)\in E$

  - update $\quad m_{i\to j}(x_j) = \sum_{x_i} f_{ij}(x_i,x_j) \cdot \prod_{l\in \partial i \setminus \{j\}} m_{l\to i}(x_i)$



  - for all $i\in V$: $\quad M_i(x_i) = \prod_{l\in\partial i} m_{l\to i}(x_i)$
  - $\hat{P}_i(x_i) = \dfrac{M_i(x_i)}{\sum_{x_i} M_i(x_i)}$

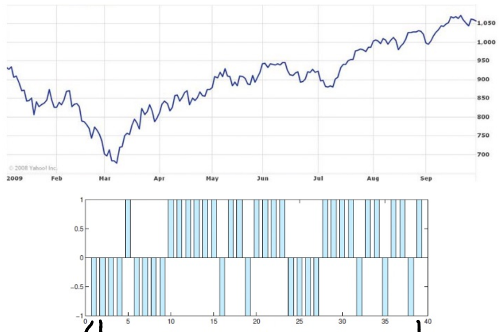


* If G is a tree of Diameter d, then $T\geq d$ ensures $\hat{P}_i(x_i) = P_i(x_i)$

length of longest path

(HW2)

Hidden Markov Model (HMM)



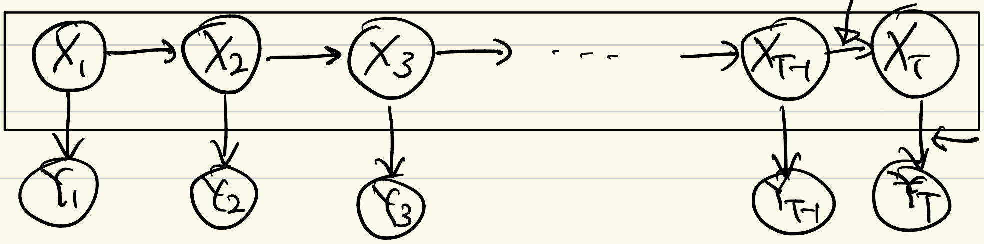
$Y_1\ Y_2\ \cdots\cdots\ Y_T \in \{+1, -1\}$

- S&P 500 index over a period of time
- For each week, measure the price movement relative to the previous week: $+1$ indicates up and -1 indicates down
- a hidden Markov model in which x_t denotes the economic state (good or bad) of week t and y_t denotes the price movement (up or down)
- $x_{t+1} = x_t$ with probability 0.8
- $\mathbb{P}_{Y_t|X_t}(y_t = +1|x_t = \text{'good'}) = \mathbb{P}_{Y_t|X_t}(y_t = -1|x_t = \text{'bad'}) = q$

$X_t \in \{Good, Bad\}$

$$P_{X_{t+1}|X_t} = \begin{array}{c|cc} & X_t{=}G & X_t{=}B \\ \hline X_{t+1}{=}G & 0.8 & 0.2 \\ X_{t+1}{=}B & 0.2 & 0.8 \end{array}$$

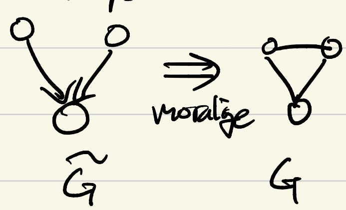
BN:

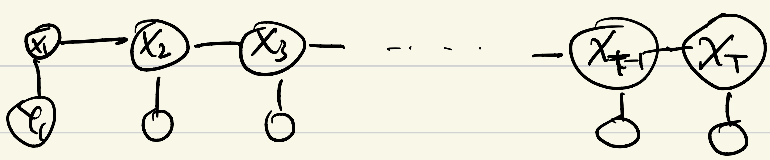


$$P_{Y_t|X_t} = \begin{array}{c|cc} & X_t{=}+1 & X_t{=}-1 \\ \hline Y_t{=}+1 & q & 1-q \\ Y_t{=}-1 & 1-q & q \end{array}$$

① Moralize BN $\tilde{G}$ to get MRF G.

V-shape





② write the factors.

$$P(x) = \frac{1}{Z} \prod_{i \in [T-1]} f_{i,i+1}(X_i, X_{i+1}) = \begin{array}{c|cc} & X_i{=}G & X_i{=}B \\ \hline X_{i+1}{=}G & & \\ X_{i+1}{=}B & & \end{array}$$

$$f_{i,i+1} = \begin{cases} P(X_{i+1}|X_i) \cdot P(Y_i|X_i) & i < T-1 \\ P(X_T|X_{T-1})\, P(Y_T|X_T)\, P(Y_{T-1}|X_{T-1}) & i = T-1 \end{cases}$$

③ Write down the BP update

$$m_{i\to i+1}(x_{i+1}) = \sum_{x_i} f_{i,i+1}(x_i, x_{i+1})\, m_{i-1\to i}(x_i)$$

$$m_{i+1\to i}(x_i) = \sum_{x_{i+1}} f_{i,i+1}(x_i, x_{i+1})\, m_{i+2\to i+1}(x_{i+1})$$

$$m_i(x_i) = m_{i-1\to i}(x_i)\, m_{i+1\to i}(x_i)$$

Elimin. Algo. on T for $P(x_1)$ → Sum-Product Algorithm ✓

Elimin. Algo. on T for $\max_x P(x)$ → Max-Product Algorithm.

Def. Max-marginal: $\tilde{P}_i(x_i) \triangleq \max_{x_{-i}} P(x)$

if $\tilde{P}_i(x_i) = \begin{array}{c|c} \begin{bmatrix} .3 \\ .5 \end{bmatrix} & \begin{array}{c} 0 \\ 1 \end{array} \end{array}$ unique maximizer

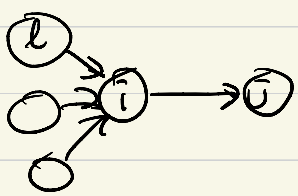
$x_i^* = 1$

⇓

$(x_1^*, \dots, x_n^*)$ maximize $P(x)$

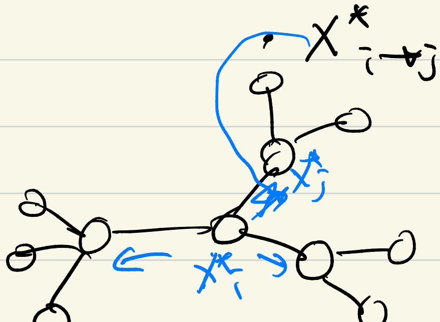
$$\underset{\uparrow\ \mathcal{X}^{n-1}}{x^*_{-i}(x_i)} \in \arg\max_{x_{-i}} P(x_i, x_{-i})$$

Update Rule for Max-Product Algorithm.



$$\cdot\ m^*_{i\to j}(x_j) = \max_{x_i} f_{ij}(x_i, x_j) \prod_{l\in\partial i\setminus\{j\}} m^*_{l\to i}(x_i)$$

$$\cdot\ x^*_{i\to j}(x_j) = \arg\max_{x_i} f_{ij}(x_i, x_j) \prod_{l\in\partial i\setminus\{j\}} m^*_{l\to i}(x_i)$$



argmax $P(x)$

Dynamic Programming on T = Max-Product algorithm

· max-marginals

$$m_i^*(x_i) = \prod_{l \in \partial i} m_{l \to i}^*(x_i)$$

$$\tilde{p}(x_i) = \frac{m_i^*(x_i)}{\sum_{x_i'} m_i^*(x_i')}$$

To get x^*, you fix x_i^* and back-track.

If $G = T$, & # of iterations > diameter, then <u>exact</u>.