Recap.

Inference Algorithms

Marginalization	Maximization			
√ Elimination Algo	Elimination Algo	Exact,	any G,	$O(\lvert\mathcal{X}\rvert^{\text{Treewidth}})$ at best but NP-hard to find ordering
√ sum-product = Belief Propagation	Max-product	Approx.,	Pairwise MRF,	$O(\lvert\mathcal{X}\rvert^2)$
√ Sum-product algorithm on factor graphs	Max-product algo on factor graphs	Approx,	any G,	$O(\lvert\mathcal{X}\rvert^{\text{max-degree}})$
Elimination algorithm on Junction trees	Elimination Algorithm on Junction trees	Exact,	Junction Trees,	$O(\lvert\mathcal{X}\rvert^{\text{Treewidth}})$ NP-hard to find the best ordering.

an exercise to get familiarized with BP updates & derive 2 different BP for Pairwise MRF.

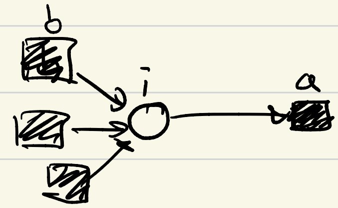
From BP for FG to BP for pairwise MRF.

BP for Factor Graphs.

Repeat $t=1\dots T$

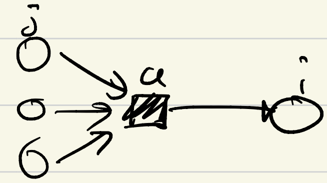
Update $m_{i\to a}$'s :

$$m_{i\to a}(x_i) = \prod_{b\in\partial i\setminus\{a\}} \tilde{m}_{b\to i}(x_i)$$



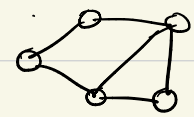
Update $\tilde{m}_{a\to i}$'s :

$$\tilde{m}_{a\to i}(x_i) = \sum_{\partial a\setminus\{i\}} f_a(x_{\partial a}) \prod_{j\in\partial a\setminus\{i\}} m_{j\to a}(x_j)$$

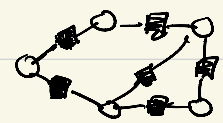


Let's apply it to Pairwise MRF:

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

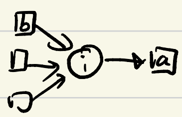


Pairwise MRF

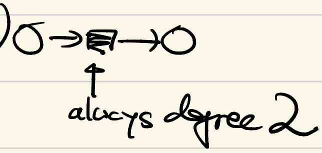


F.G.

$$m_{i\to a}(x_i) = \prod_{b\in\partial i\setminus\{a\}} \tilde{m}_{b\to i}(x_i)$$

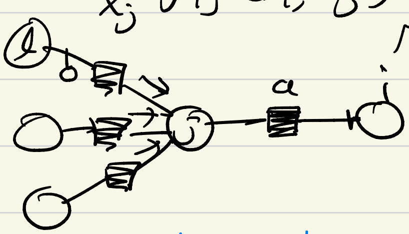


$$\tilde{m}_{a\to i}(x_i) = \sum_{x_j} f_{ij}(x_i, x_j)\, m_{j\to a}(x_j)$$



let's simplify by plugging $m_{i\to a}(x_i)$ into the update for $\tilde{m}_{a\to i}(x_i)$

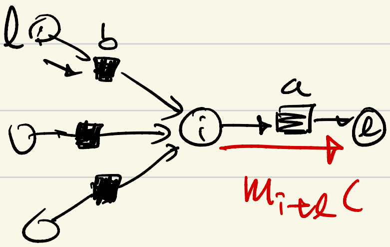
$$\tilde{m}_{\substack{\cancel{a\to i}\\ j\to i}}(x_i) = \sum_{x_j} f_{ij}(x_i, x_j) \prod_{\substack{\cancel{b\in\partial j\setminus\{a\}}\\ l\in\partial j\setminus\{i\}}} \tilde{m}_{\substack{\cancel{b\to j}\\ l\to j}}(x_j)$$



this recovers the BP update for Pairwise MRF as we learned.

let's try simplifying the other way.

$$m_{i\to a}(x_i) = \prod_{b\in\partial i\setminus\{a\}} \left\{ \sum_{x_l \in\partial b\setminus\{i\}} f_{il}(x_i, x_l)\, m_{l\to b}(x_l) \right\}$$



$$m_{i\to j}(x_i) = \prod_{l\in\partial i\setminus\{j\}} \left\{ \sum_{x_l} f_{il}(x_i, x_l)\, m_{l\to i}(x_l) \right\}$$

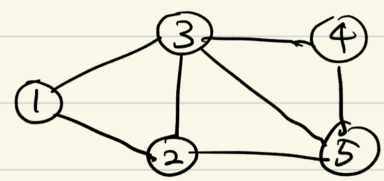
* Junction Tree Algorithm : Elimination on Junction $\boxed{\text{Tree}}$

└ exact.

- Once the tree is constructed, inference is easy using any method.
- Conceptually the approach is closer to Elimination algo [exact ordering.
- finding the right data structure for elimination to be efficient

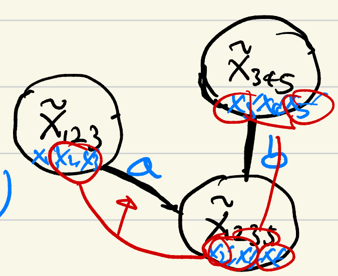
*Given a MRF $G=(V,E)$ we can construct clique Tree (none unique).

$$\text{MRF: } P(x) = \frac{1}{Z} f_{123}(x_1,x_2,x_3)\, f_{235}(x_2,x_3,x_5)\, f_{345}(x_3,x_4,x_5)$$



one example of a clique Tree for the MRF :

- Create a (giant) node for each clique
  Variable $\tilde{x}_c \in \mathcal{X}^{|c|}$
  each node has a local copy of a variable
  $\tilde{x}_{123} = (x_1, x_2, x_3)$
  $\tilde{x}_{235} = (x_2', x_3', x_5')$, $\tilde{x}_{345} = (x_3'', x_4'', x_5'')$
- assign edges so that it forms a tree



clique Tree CT_1

edges in C.T. ensure consistency of the local copies of variables.

$$\tilde{P}(\tilde{x}_{123}, \tilde{x}_{235}, \tilde{x}_{345}) = \frac{1}{Z} f_{123}(\tilde{x}_{123}) \cdot f_{235}(\tilde{x}_{235}) \cdot f_{345}(\tilde{x}_{345}) \cdot \mathbb{I}([\tilde{x}_{123}]_2 = [\tilde{x}_{235}]_1)$$

$$\mathbb{I}([\tilde{x}_{235}]_3 = [\tilde{x}_{345}]_3) \cdot \mathbb{I}(\underset{x_3'}{[\tilde{x}_{235}]_2} = \underset{x_3''}{[\tilde{x}_{345}]_1}) \qquad \mathbb{I}([\tilde{x}_{123}]_3 = [\tilde{x}_{235}]_2)$$

$f_b(\tilde{x}_{345}, \tilde{x}_{235})$ $\qquad$ $f_a(\tilde{x}_{123}, \tilde{x}_{235})$

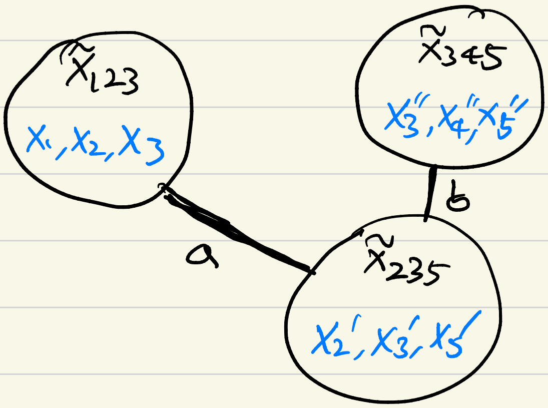
* add consistency for all shared variables

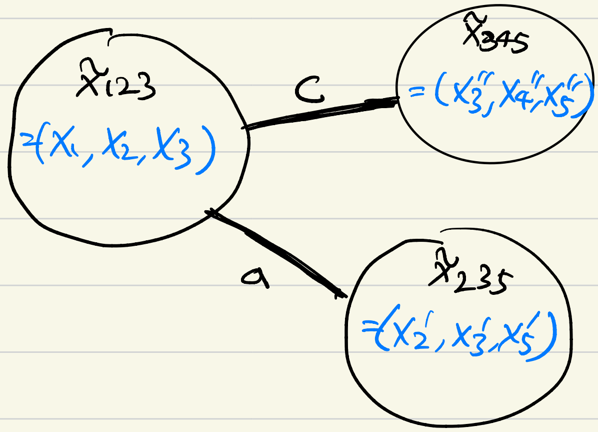
$$= \frac{1}{Z} f_a(\tilde{x}_{123}, \tilde{x}_{235}) \cdot f_b(\tilde{x}_{345}, \tilde{x}_{235}).$$

*If all local copies can be made consistent, then we have global consistency

$$P(x_1) = \sum_{x_2, x_3} \tilde{P}(\tilde{x}_{123})$$

example when Global consistency is not achievable. (depending on the tree)





we cannot ensure $X_5'' = X_5'$
with factors $f_c(\tilde{X}_{345}, \tilde{X}_{123})$
$f_a(\tilde{X}_{123}, \tilde{X}_{235})$
because this nodes are separated by a node that does not contain a local copy of X_5

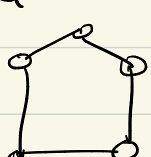
Q. When is a clique tree globally consistent?

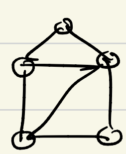
Def. Junction Tree Property.

Given a MRF $G=(V,E)$ with $\mathcal{C}$ set of maximal cliques,
for a tree T over $\mathcal{C}$, a node $i \in V$ satisfy J.T.P if
all cliques containing i form a connected subtree in T.

Def. T is a Junction Tree for G if all $i \in V$ has J.T.P.

[existence]
Q. - When does G have a Junction Tree?
If the graph G is chordal ($\triangleq$ any loop ≥ 4 has a chord).


4 loop without chord.


chordal

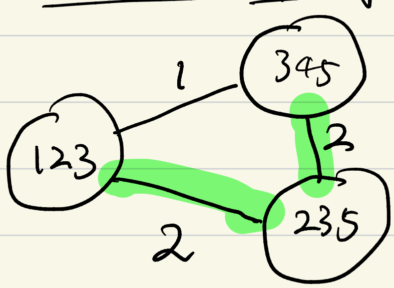
[Construction]

Q. How do we find a Junction tree? ①

Q. If the graph is not chordal, what can we do? ②

① Algorithm sketch to find J.T. given $G=(V,E)$ with $\mathcal{C}$: set of max cliques.

- consider a complete graph on $\mathcal{C}$
- assign weight on edges as # of shared variables between the two cliques
- find the max weight spanning tree. ($\triangleq$ a tree whose sum of edge weight is maximized).



can be easily found by Kruskal's algorithm.

Claim. [Correctness] A clique tree T is a Junction tree if and only if it is a maximal spanning tree.

$$w(T) = \sum_{(C_l, C_r)\in T} |C_l \cap C_r|$$

for any tree

$$= \sum_{(C_l, C_r)\in T} \sum_{i\in V} \mathbb{I}(i\in C_l \,\&\, i\in C_r)$$

$$= \sum_{i\in V} \boxed{\sum_{(C_l, C_r)\in T} \mathbb{I}(i\in C_l \,\&\, i\in C_r)} \leq m_i - 1$$

because cliques containing node i is a sub-tree of size m_i

m_i : # cliques containing node i

$$\leq \sum_{i\in V} (m_i - 1)$$

equal if and only if all sub~~tree~~graph containing a node i ~~form~~ is connected.

$\therefore$ Junction Tree T $\longrightarrow$ achieve max-weight

② If you are given a graph $G=(V,E)$ that is not chordal.

sketch of algorithm

- choose an elimination ordering
- find the reconstituted graph (which is chordal)
- construct a complete clique graph
- Find max-weight spanning tree

$\Rightarrow$ Junction Tree

we can run any algorithm we learned on this junction tree for exact inference

* Different elimination ordering gives different complexity
max-clique size

at best we get $O(|\mathcal{X}|^{\text{treewidth}})$.

Q. How good is belief propagation for $|\mathcal{X}| < \infty$?

- If G is tree $\Rightarrow$ it converges and exact
- If $\exists$ only 1 loop $\Rightarrow$ Converges but could be wrong [Weiss 2000]
- Maximum Weight Matching Problem $\Rightarrow$ Exact, Converges & [Bayati, Shah, Sharma 2005]
- In the limit of large graph, <u>Density Evolution</u> provides an asymptotic performance estimate.

  - Analysis of compressed sensing

$$\min_{x \in \mathbb{R}^d} \|Ax - b\|^2 + \lambda \cdot \|x\|_{L1}$$

A: fat = underdetermined system

$\|x\|_{L1} = \sum_{i=1}^{d} |x_i|$

A, x: sparse.

  - Analysis of LDPC decoders
    $\Rightarrow$ Design of the first provably capacity achieving codes [Luby 1998]
  - Analysis of near-optimal crowdsourcing algorithm [KargerOhShah 2011].
  - Community detection for stochastic block models.

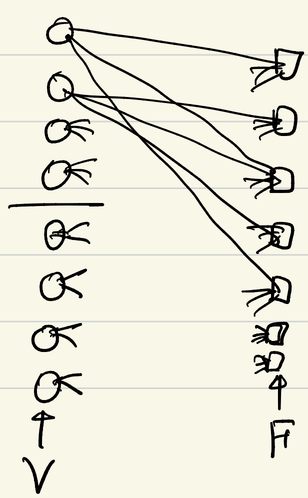
* Assumption: Factor Graph model on Random Graphs.

Def. Random Graph $(n, m, l = [l_1, l_2, l_3 \cdots, l_{d_1}], r = [r_1, r_2, r_3, r_4, r_{d_2}])$

$n = |V|$, $m = |F|$, $l_1 = 0$, $l_2 = .5$, $l_3 = .5$, $l_{d_1} = 0$, $r_1 = 0$, $r_2 = 0$, $r_3 = 0$, $r_4 = 1$.

s.t. $n \cdot \bar{l} = m \cdot \bar{r} = |E|$.

$\bar{l} = \sum_i l_i \cdot i$, $\quad \bar{r} = \sum_j r_j \cdot j$



G is drawn from Random Graph (n, m, l, r) if it is chosen uniformly over all graphs satisfying the (l, r) degree distribution.

• given (ℓ, r). how do you generate such graphs?

Def. Configuration model. $RG(n, m, \ell, r)$

$$\begin{array}{cc} \ell_1 \\ \ell_2 \\ \ell_3 \\ \ell_4 \end{array} \begin{bmatrix} 0 \\ 0 \\ .5 \\ .5 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \begin{array}{c} r_1 \\ r_2 \\ r_3 \\ r_4 \end{array}$$

$\bar{\ell} = 3.5 \quad \bar{r} = 4$

$n = 6 \quad m = 7 \qquad |E| = 21$

① draw a Random Permutation π from $[|E|] \to [|E|]$
$\{1, 2, \cdots, 21\} \quad \{1, 2, \cdots, 21\}$

② and match left half edges to the right w.r.t π.

③ locally fix double-edges.

Density Evolution for Random Graphs.

ex) LDPC codes.

Factor Graph Model: $X_i \in \{0, 1\}$, $Y_i \in \{0, 1, \overset{\text{erasure.}}{*}\}$

$\mathbb{P}(Y_i = * \mid X_i) = \varepsilon$

$\mathbb{P}(Y_i = X_i \mid X_i) = 1 - \varepsilon.$

$$P_Y(x) = \frac{1}{Z} \cdot \underbrace{\prod_{i=1}^{n} \mathbb{P}(Y_i \mid X_i)}_{\text{channel}} \cdot \underbrace{\prod_{a \in F} \mathbb{I}(\bigoplus X_a = 0)}_{\text{LDPC Parity.}}$$

XOR of all binary variables.

Q. what is the error achieved by BP to get $\hat{P}_Y(X_i)$'s.

* Belief Propagation on LDPC codes.

$$m_{i\to a}(x_i) = \mathbb{P}(y_i, x_i)\cdot \prod_{b\in\partial i\setminus\{a\}} \hat{m}_{b\to i}(x_i) \qquad \mathbb{P}(y_i,x_i) = \begin{cases} \varepsilon & y_i = * \\ 1-\varepsilon & y_i = x_i \\ 0 & y_i = \text{not } x_i \end{cases}$$

binary operation

$$\hat{m}_{a\to i}(x_i) = \sum_{x_{\partial a\setminus\{i\}}} \left\{ \prod_{j\in\partial a\setminus\{i\}} m_{j\to a}(x_j) \right\} \mathbb{I}(\oplus x_{\partial a}=0)$$

claim: $m_{i\to a}(x_i) \in \left\{ \begin{bmatrix}1\\0\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}\frac{1}{2}\\\frac{1}{2}\end{bmatrix} \right\}$ ← The messages are Discrete.

Def. $x_{i\to a}=0$ $\quad x_{i\to a}=1$ $\quad x_{i\to a}=*$

proof: by induction

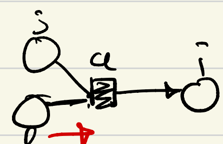
Init: $\hat{m}^{(0)}_{b\to i}(x_i)$ is initialized as $\begin{bmatrix}\frac{1}{2}\\\frac{1}{2}\end{bmatrix}$,

$$m^{(1)}_{i\to a}(x_i) = \begin{cases} \begin{bmatrix}\varepsilon\\\varepsilon\end{bmatrix} \propto \begin{bmatrix}\frac{1}{2}\\\frac{1}{2}\end{bmatrix} & \text{if } y_i = * \\ \begin{bmatrix}1-\varepsilon\\0\end{bmatrix} \propto \begin{bmatrix}1\\0\end{bmatrix} & y_i = 0 \\ \begin{bmatrix}0\\1\end{bmatrix} & y_i = 1 \end{cases}$$

Induction: ① $\hat{m}^{(t)}_{a\to i}$:

$m_{l\to a}(x_l) = \begin{bmatrix}1\\0\end{bmatrix}$ $x_{l\to a}=0$

Case 1: all incoming messages are $m_{j\to a}(x_j) = \begin{bmatrix}1\\0\end{bmatrix}$ $x_{j\to a}=1$



then
$$\hat{m}^{(t)}_{a\to i}(x_i) = \begin{bmatrix} m_{\to}(x_j=0)\, m_{\to}(x_l=0) + \cancel{m(x_j=1)m(x_l=1)} \\ \cancel{m(x_j=0)m(x_l=1)} + \cancel{m(x_j=1)m(x_l=0)} \end{bmatrix}$$
$$= \begin{bmatrix}1\\0\end{bmatrix}$$

$\{x_l=0$ or $x_l=1$ or $x_l=*$

* If incoming messages are all $\begin{bmatrix}0\\1\end{bmatrix}$ or $\begin{bmatrix}1\\0\end{bmatrix}$

then $X^{(t)}_{a\to i} = \bigoplus_{j\in\partial a\setminus\{i\}} X_{j\to a}$

Case 2: at least one incoming message is $m_{i\to a}(x_i) = \begin{bmatrix}\frac{1}{2}\\\frac{1}{2}\end{bmatrix}$, $x_{i\to a}=*$

then
$$\hat{m}^{(t)}_{a\to i}(x_i) = \begin{bmatrix} \underbrace{m(x_j=0)}_{\frac{1}{2}}\, \underbrace{m(x_l=0)}_{1} + m(x_j=1)\cancel{m(x_l=1)} \\ m(x_j=0)\cancel{m(x_l=1)} + \underbrace{m(x_j=1)}_{\frac{1}{2}}\underbrace{m(x_l=0)}_{1} \end{bmatrix}$$
$$= \begin{bmatrix}\frac{1}{2}\\\frac{1}{2}\end{bmatrix}$$

* If incoming message has at least one $\begin{bmatrix}\frac{1}{2}\\\frac{1}{2}\end{bmatrix}$, then $\hat{m}^{(t)}_{a\to i} = \begin{bmatrix}\frac{1}{2}\\\frac{1}{2}\end{bmatrix}$

$* \quad \tilde{m}^{(t)}_{a\to i}(x_i) \in \begin{bmatrix}1\\0\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}1/2\\1/2\end{bmatrix}$

$\tilde{x}^{(t)}_{a\to i} = 0 \ , \ 1 \ , \ *$

② $$m^{(t)}_{i\to a}(x_i) = \prod_{b\in \partial i\setminus\{a\}} \hat{m}^{(t)}_{b\to i}(x_i)$$

Case 1: if all incoming is $\tilde{m}_{b\to i} = \begin{bmatrix}1/2\\1/2\end{bmatrix}$, $\tilde{x}_{b\to i} = *$

then $m_{a\to i} = \begin{bmatrix}1/2\\1/2\end{bmatrix}$, $x_{a\to i} = *$

Case 2: if at least one incoming is $\tilde{m}_{b\to i} = \begin{bmatrix}0\\1\end{bmatrix}$ or $\begin{bmatrix}1\\0\end{bmatrix}$

$\tilde{x}_{b\to i} = 1 \quad 0$

then $m_{a\to i} = \tilde{m}_{b\to i} \in \begin{bmatrix}0\\1\end{bmatrix}$ or $\begin{bmatrix}1\\0\end{bmatrix}$.

With this claim, we can simplify the BP update.

$x^{(t)}_{i\to a} \in \{0,1,*\}$, $\tilde{x}^{(t)}_{a\to i} \in \{0,1,*\}$

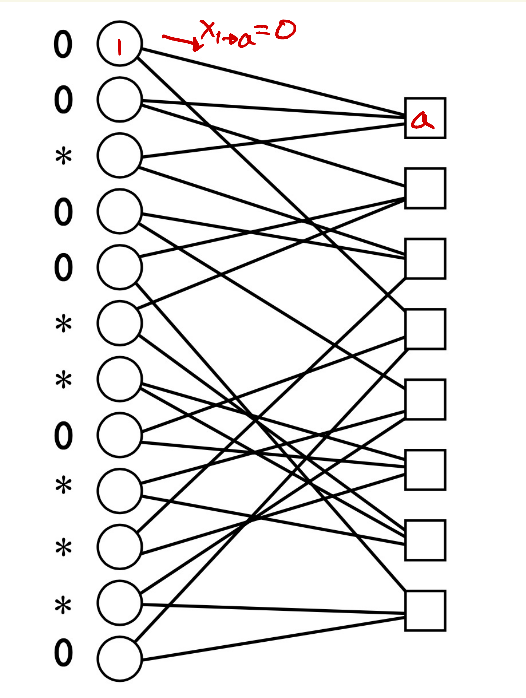
initialize $x_{i\to a} = y_i$

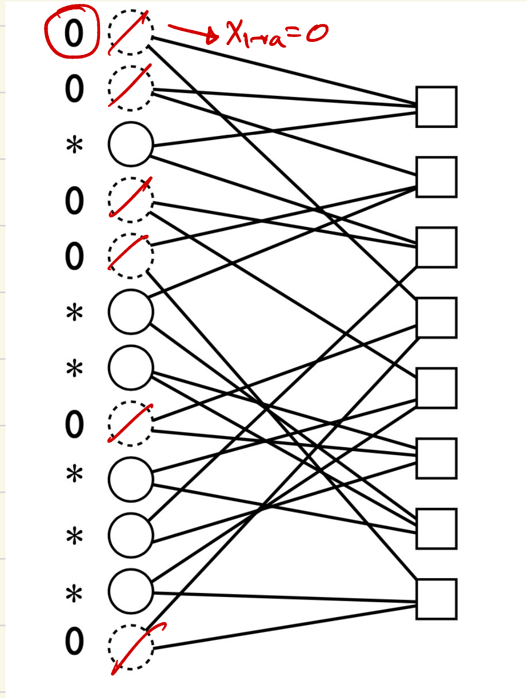
update $\tilde{x}_{a\to i} = \begin{cases} * & \text{, at least one incoming is } *. \\ \oplus x_{\partial a\setminus i\to i}, & \end{cases}$

$x_{i\to a} = \begin{cases} * & \text{, all incoming is } * \\ \tilde{x}_{b\to i} & , \end{cases}$

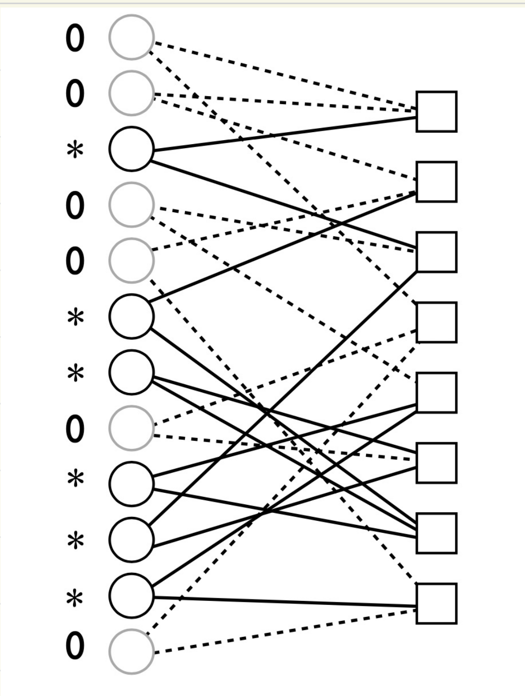
Claim: this is equivalent to the following <u>peeling algorithm</u>.

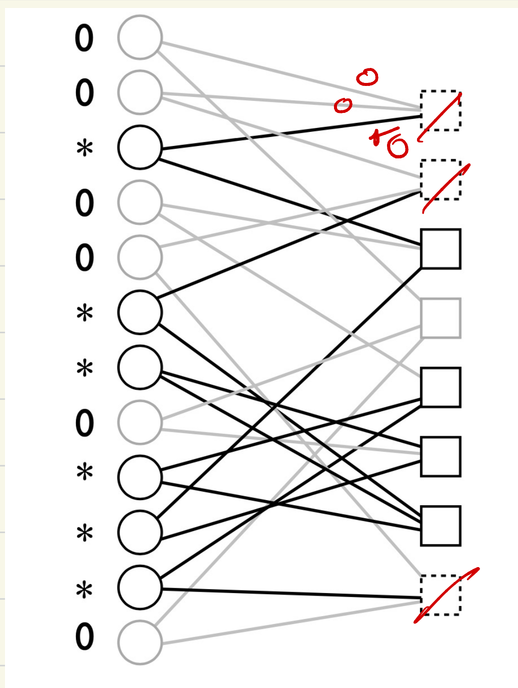
Without loss of generality, suppose all 0's were sent.



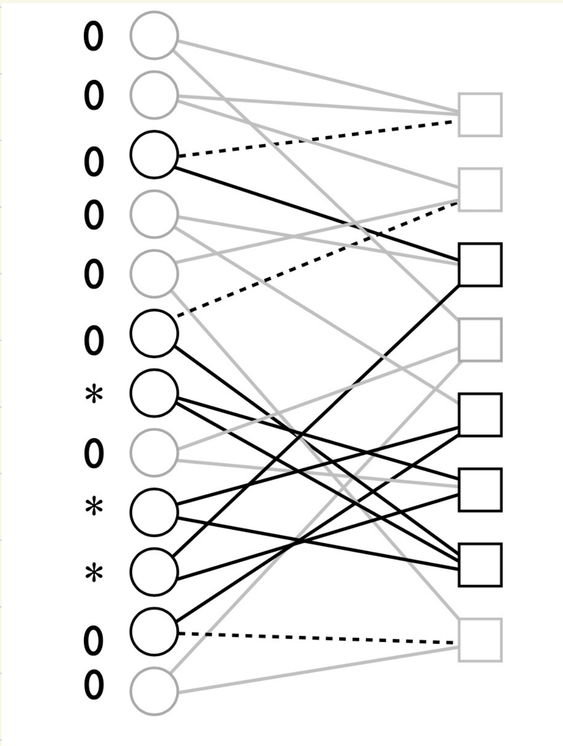


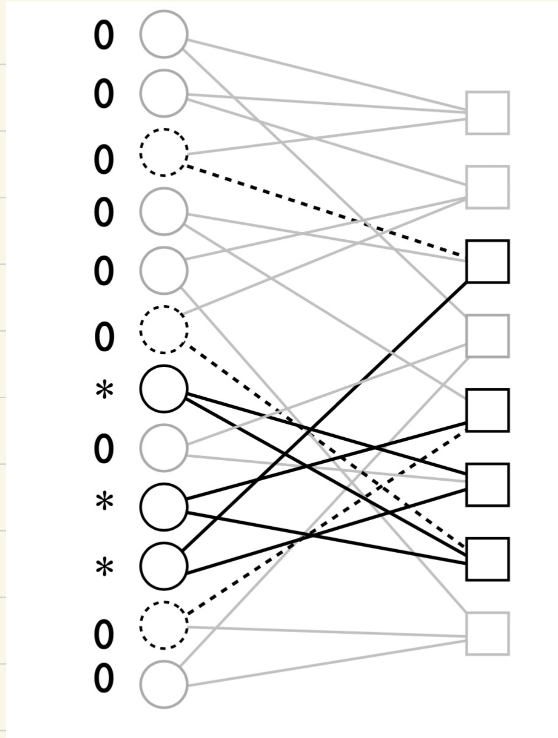
$x_1=0$

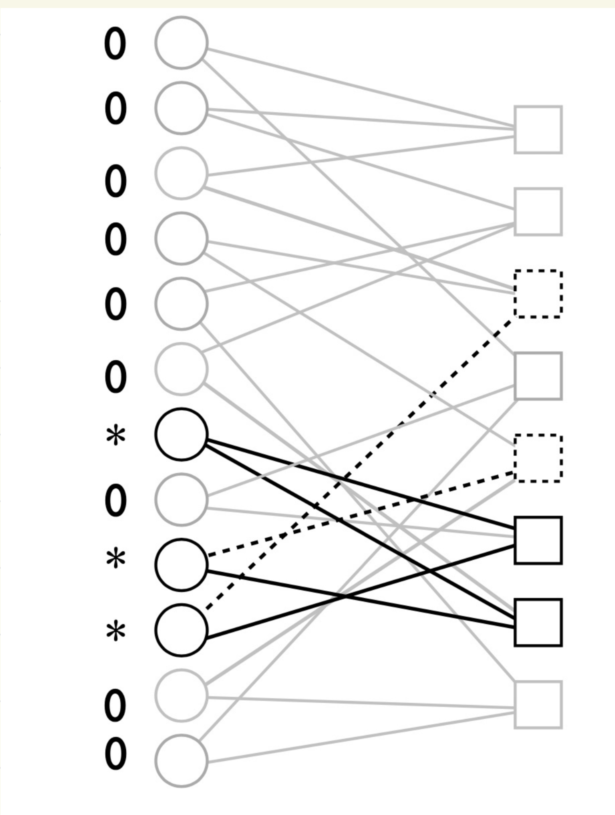


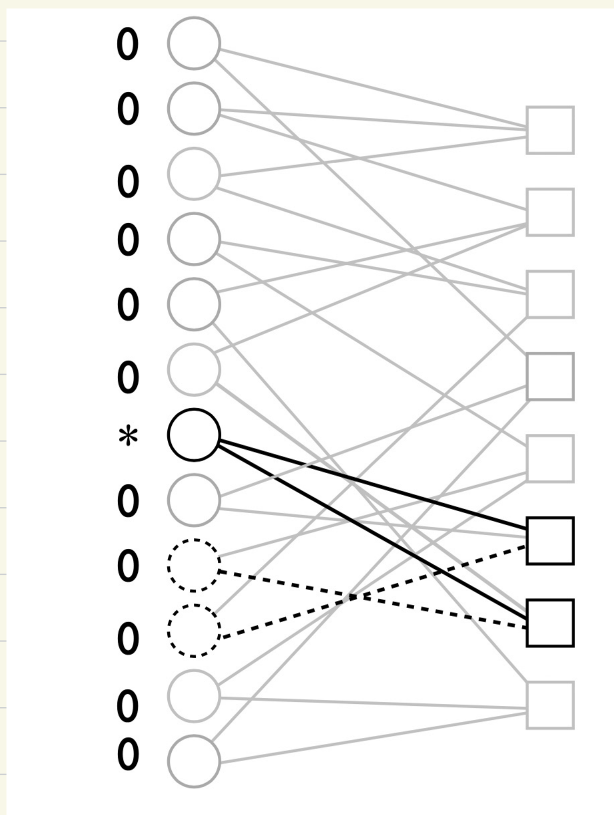


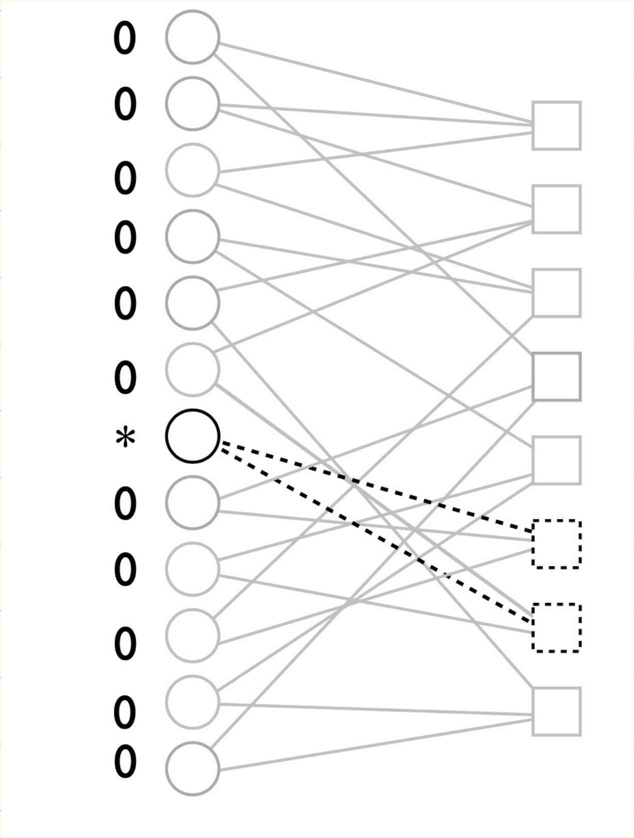
* factor nodes with 1 remaining edge can be decoded.


0
0
0
0
0
0
*
0
*
*
0
0


0
0
0
0
0
0
*
0
*
*
0
0


0
0
0
0
0
0
*
0
*
*
0
0


0
0
0
0
0
0
*
0
0
0
0
0


0
0
0
0
0
0
*
0
0
0
0
0