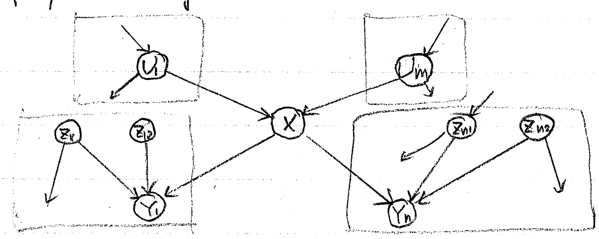
10/17

Review

* d-separation
  - (i) intermediate cause
  - (ii) common cause
  - (iii) "no observed common effect"
* polytree algorithm



* evidence $E = \underset{\substack{\uparrow \\ \text{above } X}}{E_X^+} \cup \underset{\substack{\nwarrow \\ \text{below } X}}{E_X^-}$

* Bayes rule

$$P(X|E) = \frac{P(E_X^- | X)\, P(X|E_X^+)}{P(E_X^- | E_X^+)}$$

$\vec{U} = (U_1, U_2, \cdots U_m)$
$\vec{u} = (u_1, u_2, \cdots, u_m)$

* "Upstream" recursion

evidence connected to U_i not via X.

$$P(X|E_X^+) = \sum_{\vec{u}} \underbrace{P(X|\vec{U}=\vec{u})}_{\text{CPT}} \prod_{i=1}^{m} \underbrace{P(U_i = u_i | E_{U_i \setminus X})}_{\text{recurse on parents}}$$

* Downstream recursion:

$$P(E_X^- | X) = \prod_j P(E_{Y_j \setminus X} | X) \quad \text{d-separation case II.}$$

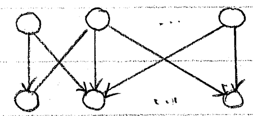
* Stated without proof:

$$P(E_{Y_j \setminus X} | X = x) = \left(\begin{matrix}\text{constant factor} \\ \text{independent of } X\end{matrix}\right) \sum_{Y_j} \overbrace{P(E_{Y_j}^- | Y_j)}^{\text{Recursion}} \sum_{\substack{\vec{Z}_j \\ \text{spouses}}} \overbrace{P(Y_j | \vec{Z}_j, X = x)}^{\text{CPT}} \times \prod_k \underbrace{P(Z_{jk} | E_{Z_{jk} \setminus Y_j})}_{\text{Recursion}}$$

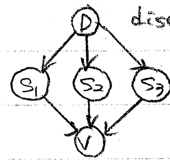
* Termination conditions
  - root node (no parents)
  - leaf node (no children)
  - evidence node (trivial)
* Running time
  - linear # nodes and size of CPTs.

Loopy networks

Ex: medical diagnosis
two-layer network



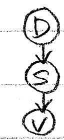
Ex: simpler example



* Exact inference

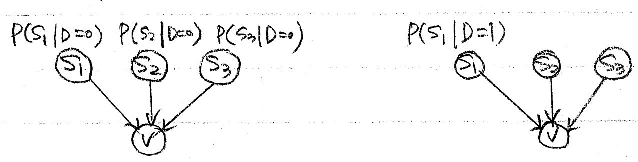
How to turn a loopy network into a polytree?

(1) Node clustering



- merge nodes to form polytree.
  ex. merge S_1, S_2, S_3 into one node S
- merge CPTs
  ex. merge $P(S_1|D)$, $P(S_2|D)$, $P(S_3|D)$ into mega-CPT $P(S|D)$.
- apply polytree algorithm
  size of mega node : 2^3
  size of mega CPT : 2^4
  polytree algorithm linear in CPT size
  CPT size grows exponentially with clustering.
  How to choose optimal clustering of nodes? Hard problem.

(2) Cutset conditioning
- Instantiate nodes so that remaining nodes form a polytree
  ex. instantiate $D=0$ or $D=1$



- Apply polytree algorithm on each sub-network separately then compute weighted average using $P(D=0)$ and $P(D=1)$ from original BN.
- Set of instantiated nodes : cut-set

* Approximate inference

Exact inference is NP-hard.

Approximate methods best choice for loopy BNs.

Stochastic simulation

* Belief network as "generative model"

$$P(X_1, X_2, \cdots, X_n) = \prod_i P(X_i \mid pa(X_i))$$

Easy to draw samples from joint distribution.

Harder to draw samples from posterior distribution.

E = evidence nodes

Q = query nodes

How to estimate $P(Q \mid E)$?

* Rejection sampling

To estimate $P(Q=q \mid E=e)$?

Generate N samples from joint distribution of BN.

Count # samples $N(e)$ where $E=e$

Count # samples $N(q,e)$ where $E=e$ and $Q=q$.

Estimate $P(Q=q \mid E=e) \approx N(q,e)/N(e)$ with $N(q,e) \le N(e) \le N$

Converges as $N \to \infty$.

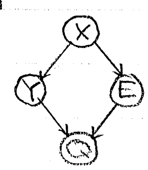
Inefficient!

- takes many samples for rare evidence and queries.
- discards samples without $E=e$

* Likelihood weighting

- Instantiate evidence nodes instead of sampling them.
- Weight each sample using CPTs at evidence nodes

Ex:



To estimate $P(Q=q \mid E=e)$:

- draw samples $\{x_i, y_i, q_i\}_{i=1}^N$
- sample x_i from $P(X)$
- sample y_i from $P(Y \mid X=x_i)$
- fix $E=e$
- sample q_i from $P(Q \mid Y=y_i, E=e)$

* Define "indicator" function : $I(q,q') = \begin{cases} 0 & \text{otherwise} \\ 1 & \text{if } q=q' \end{cases}$

* Estimate

$$P(Q=q \mid E=e) \approx \frac{\sum_{i=1}^N I(q,q_i) \overbrace{P(E=e \mid X=x_i)}^{\text{likelihood weight}}}{\sum_{i=1}^N P(E=e \mid X=x_i)}$$

6

* Much faster than rejection sampling :
- uses all samples with instantiated evidence
- converges in limit $N \to \infty$ to correct answer
- still slow for rare events

  Suppose $P(Q=q \mid E=e) \sim 10^{-20}$

  Need roughly 10^{20} samples.