Review

* d-separation

(i) intermediate cause
(ii) common cause
(iii) "no observed common effect"

* polytree algorithm

* evidence \( E = E^+ \cup E^-
\)

* Bayes rule

\[
P(C(X|E)) = \frac{P(E^{-} | X) P(C(X|E^+))}{P(E^+ | E^-)}
\]

* "Upstream" recursion

\[
P(X | E^+) = \sum_{U} P(X | U = \bar{U}) \prod_{i \in U} P(U_i = U_i | E \cup \bar{U})
\]

* Downstream recursion:

\[
P(E^{-} | X) = \prod_{j} P(E_{j|X} | X) \quad \text{d-separation case II}
\]

* Stated without proof:

\[
P(E_{j|X} | X = x) = \left( \text{constant factor independent of } X \right) \sum_{Z_j} P(E_{j|X} | Y_j) \sum_{Z_j} P(Y_j | Z_{j|x}) \times
\]

\[
\prod_{k} P(Z_{j|k} | E_{j|k} \setminus Y_j)
\]

* 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"

\[ P(X_1, X_2, \ldots, X_n) = \prod_i P(X_i | \text{pa}(X_i)) \]

Easy to draw samples from joint distribution.

Harder to draw samples from posterior distribution.

E = evidence nodes
Q = query nodes

How to estimate \( P(Q | E) \)?

* Rejection sampling

To estimate \( P(Q = q | E = e) \)?
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 \( P(Q = q | E = e) \sim N(q, e) / N(e) \) with \( N(q, e) \leq N(e) \leq N \)
Convexes 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 \( P(Q = q | E = e) \):

- Draw samples \( x_1, y_1, q_1 \sim N \)
- Sample \( x_i \) from \( P(X) \)
- Sample \( y_i \) from \( P(Y | X = x_i) \)
- Fix \( E = e \)
- Sample \( q_i \) from \( P(Q | Y = y_i, E = e) \)

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

** Estimate

\[ P(Q = q | E = e) \sim \frac{\sum_{i=1}^{N} I(q, q_i) P(E = e | X = x_i)}{\sum_{i=1}^{N} P(E = e | X = x_i)} \]
*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.