Exact Inference
What To Do With Bayesian/Markov Network?

• Compact representation of a complex model, but...

• Goal: efficient extraction of information
  – i.e., posing queries
Types of Queries

• Maximum a posterior (MAP) hypothesis
Types of Queries

• Maximum a posterior (MAP) hypothesis
  – Most probable explanation (MPE)
Types of Queries

• Maximum a posterior (MAP) hypothesis
  – Most probable explanation (MPE)

• Probability of evidence
  – Partition function
Types of Queries

- Maximum a posterior (MAP) hypothesis
  - Most probable explanation (MPE)
- Probability of evidence
  - Partition function
- Prior and/or posterior marginals

- Focus on probabilities
  - Will translate to hypotheses
How Difficult Is Answering Queries?

• Consider full probability table (assume n binary variables)

• Simple marginal
  – Find $P(X_i)$
  – Complexity: $O(2^{n-1})$

• Can we do this faster?
  – In the general case, no (more on this later)
  – What if the joint distribution has structure?
Rearranging Terms in the Sum

\[
P(L) = \sum_D \sum_I \sum_G \sum_S P(D, I, G, S, L)
\]

\[
= \sum_D \sum_I \sum_G \sum_S P(D) P(I) P(G|D, I) P(S|I) P(L|G)
\]

\[
= \sum_D P(D) \sum_I P(I) \sum_G P(G|D, I) P(L|G) \sum_S P(S|I)
\]

\[
= \sum_D P(D) \sum_I P(I) \sum_G P(G|D, I) P(L|G)
\]

\[
= \sum_D P(D) \sum_I P(I) \psi_1(D, I, L)
\]

\[
= \sum_D P(D) \psi_2(D, L)
\]

\[
= \psi_3(L)
\]
Rearranging Terms

• Complexity?
  – Exponential in the size of the largest factor
  – In turn, depends on the order of the summation
  – Also depends on whether all of the variables need to be summed over

• Bad news
  – Finding optimal order is NP-hard
All I Learned in This Course Was From Watching the Trees

• Graphs without loops
• Directed trees
  – Implies at most one parent for each node
• Polytrees
  – Possibly more than one parent for some nodes
Why Trees?

• Because efficient exact inference reduces to inference over trees
Chain: Tree, Only Simpler!

\[
Z = \sum_{X_1} \sum_{X_2} \sum_{X_3} \sum_{X_4} \sum_{X_5} \phi_1(x_1, x_2) \phi_2(x_2, x_3) \phi_3(x_3, x_4) \phi_4(x_4, x_5)
\]

\[
= \sum_{X_1} \sum_{X_2} \phi_1(x_1, x_2) \sum_{X_3} \phi_2(x_2, x_3) \sum_{X_4} \phi_3(x_3, x_4) \sum_{X_5} \phi_4(x_4, x_5)
\]

\[
= \sum_{X_1} \sum_{X_2} \phi_1(x_1, x_2) \sum_{X_3} \phi_2(x_2, x_3) \sum_{X_4} \phi_3(x_3, x_4) \psi_1(x_4)
\]

\[
= \sum_{X_1} \sum_{X_2} \phi_1(x_1, x_2) \sum_{X_3} \phi_2(x_2, x_3) \psi_2(x_3)
\]

\[
= \sum_{X_1} \sum_{X_2} \phi_1(x_1, x_2) \psi_3(x_2)
\]

\[
= \sum_{X_1} \psi_4(x_1)
\]

\[
P(X) = \frac{1}{Z} \phi_1(x_1, x_2) \phi_2(x_2, x_3) \phi_3(x_3, x_4) \phi_4(x_4, x_5)
\]

\[
\begin{array}{c}
\text{STAT 598L: Exact Inference}
\end{array}
\]

\[
\begin{array}{ccccc}
\text{x}_1 & - & \text{x}_2 & - & \text{x}_3 & - & \text{x}_4 & - & \text{x}_5
\end{array}
\]
Sum-Product Method for Chains

• Computational complexity?
  – $O(n |\text{Val}(X_{max})|^2)$

• Storage complexity?
  – Same
Sum-Product for Trees

\[ Z = \sum_{X_1} \sum_{X_2} \sum_{X_3} \sum_{X_4} \sum_{X_5} \phi_1(x_1, x_4) \phi_2(x_2, x_4) \phi_3(x_3, x_4) \phi_4(x_3, x_5) \]

\[ = \sum_{X_1} \sum_{X_4} \phi_1(x_1, x_4) \sum_{X_2} \phi_2(x_2, x_4) \sum_{X_3} \phi_3(x_3, x_4) \sum_{X_5} \phi_4(x_3, x_5) \]

\[ = \sum_{X_1} \sum_{X_4} \phi_1(x_1, x_4) \sum_{X_2} \phi_2(x_2, x_4) \sum_{X_4} \phi_3(x_3, x_4) \psi_1(x_3) \]

\[ = \sum_{X_1} \sum_{X_4} \phi_1(x_1, x_4) \sum_{X_4} \phi_2(x_2, x_4) \sum_{X_2} \phi_3(x_3, x_4) \psi_1(x_3) \]

\[ = \sum_{X_1} \phi_1(x_1, x_4) \sum_{X_2} \phi_2(x_2, x_4) \sum_{X_4} \phi_3(x_3, x_4) \psi_1(x_3) \]

\[ = \sum_{X_1} \phi_1(x_1, x_4) \psi_2(x_4) \psi_3(x_4) \psi_4(x_5) \]

\[ = \sum_{X_1} \psi_4(x_1) \]

\[ P(X) = \frac{1}{Z} \phi_1(x_1, x_4) \phi_2(x_2, x_4) \phi_3(x_3, x_4) \phi_4(x_3, x_5) \]

STAT 598L: Exact Inference
Sum-Product Method (Variable Elimination)

• Pick an order by selecting a node as root
• Start at the end of the ordering
  – Sum out the child and update (product) the factor for the parent
  – Continue until no variables left to sum
• Can be done in the opposite direction
  – Sum out the parents rather than children
What To Do for General Graphs?

• Think decomposable
  – Junction trees
Variable Elimination

• Pick ordering
  – Bayesian networks: topological ordering

• For each node in the ordering A
  – Compute the product of all factors involving A
  – Sum out A
    • Obtain a factor involving variables in all factors with A

• Complexity
  – Depends on the ordering
What Happens to the Network?

\[ P(L) = \sum_S \sum_G \sum_I \sum_D P(D, I, G, S, L) \]

\[ = \sum_S \sum_G \sum_I \sum_D P(D) P(I) P(G|D, I) P(S|I) P(L|G) \]

\[ = \sum_S \sum_G P(L|G) \sum_I P(I) P(S|I) \sum_D P(D) P(G|D, I) \]

\[ = \sum_S \sum_G P(L|G) \sum_I P(I) P(S|I) \psi_1(G, I) \]

\[ = \sum_S \sum_G P(L|G) \psi_2(G, S) \]

\[ = \sum_S \psi_3(S, L) \]

\[ = \psi_4(L) \]
Induced Graph

• Nodes = nodes of the original Bayesian/Markov network
• Edges = pairs of variables in the same factor at any stage of variable elimination
• Induced graph is chordal
• Every factor is a clique
• Every maximum clique is an intermediate factor
Induced Width

• Complexity of variable elimination
  – Exponential in the size of the largest clique of the induced graph

• **Induced width**: size of the largest clique of the induced graph – 1

• Optimization: minimize induced width
  – Original graph is **chordal** – easy!
  – General case – NP-hard!
Chordal Graphs

- Chordal graphs admit ordering that do not introduce extra edges.
- Proof: While there are vertices
  - Pick a simplicial vertex and eliminate the corresponding variable.

$H$ is induced graph $\iff$ $H$ is chordal graph.
Finding Orderings

• Chordal graphs: maximum cardinality
  – Pick a vertex with the largest number of neighbors
  – Eliminate it

• Does not add edges (homework)

• How well does it do for the general case?
General Greedy Strategies

• Current ordering $O=\emptyset$
• While $|O| \neq |H|$
  – Select a node $v$ according to a heuristic $h(H,O)$
  – Add the node to the ordering, $O=O\cup\{v\}$

• How to choose a heuristic $h$?
  – Problem is NP-hard, so none is optimal for all graphs
  – Examples:
    • Min-neighbors – number of neighbors in the current graph
    • Min-weight – domain cardinality of its neighbors
    • Min-fill – number of edges to add if eliminated
    • Weighted-min-fill – sum of weights of edges added if eliminated
Comparison of Ordering Heuristics
Conditioning

• What if some variables are known?
  – Don’t add to the induced width
• Idea: treat the values of some variables as known
  – Solve the elimination for each value
  – Aggregate
Local Probability Model

• What if a CPD has a compact representation
  – Noisy-or
  – Decision tree/graph
  – Rule-based
Noisy-or

- Can be made more efficient!

\[
P \left( Y = 0 \mid X_1, \ldots, X_k \right) = (1 - p_0) \prod_{i : X_i = 1} (1 - p_i)
\]

\[
P \left( Y = 1 \mid X_1, \ldots, X_k \right) = 1 - (1 - p_0) \prod_{i : X_i = 1} (1 - p_i)
\]