Exact Inference: Variable Elimination

Complexity of Variable Elimination

Complexity

- Exponential size of the factors $\psi$ dominates the complexity
- If each variable has no more than $v$ values
- And a factor $\psi_i$ has a scope that contains $k_i$ variables, then the number of entries $N_i$ in $\psi_i$ is: $N_i \leq v^{k_i}$

Complexity

- Complexity of Variable Elimination depends on the structure of the graph
- Note: the VE algorithm does not care if the graph is directed, undirected, or partially directed
Complexity

• Let $\Phi$ be a set of factors. We define $\text{Scope}[\Phi] = \bigcup_{\phi \in \Phi} \text{Scope}[\phi]$ to be the set of all variables appearing in one of the factors in $\Phi$.

• We define $\mathcal{H}_\Phi$ to be the undirected graph whose nodes correspond to the variables in $\text{Scope}[\Phi]$ and where we have an edge $X_i \!-\! X_j \in \mathcal{H}_\Phi$ if and only if there exists a factor $\phi \in \Phi$ such that $X_i, X_j \in \text{Scope}[\phi]$.

(Informally) The undirected graph $\mathcal{H}_\Phi$ introduces a fully connected subgraph over the scope of each factor $\phi \in \Phi$, and hence is the minimal I-map for the distribution induced by $\Phi$.

eg. $\Phi = \{\phi_1(X_1, X_2, X_3), \phi_2(X_3, X_4), \phi_3(X_4, X_5, X_6, X_7)\}$

Complexity

• Proposition 9.1: Let $P$ be a distribution defined by multiplying the factors in $\Phi$ and normalizing to define a distribution. Letting $X = \text{Scope}[\Phi]$, $P(X) = \frac{1}{Z} \prod_{\phi \in \Phi} \phi$ where $Z = \sum_X \prod_{\phi \in \Phi} \phi$.

Then $\mathcal{H}_\Phi$ is the minimal Markov network I-map for $P$, and the factors $\Phi$ are a parameterization of this network that defines the distribution $P$.

• For a set of factors $\Phi$ defined by a Bayesian network $\mathcal{G}$, in the case without evidence, the undirected graph $\mathcal{H}_\Phi$ is the moralized graph of $\mathcal{G}$.

• The product of the factors is a normalized distribution and the partition function is simply 1.
Complexity

When variable $X$ is eliminated:
- Create a single factor $\psi$ that contains $X$ and all of the variables $Y$ with which it appears in factors.
- Eliminate $X$ from $\psi$, replacing it with a new factor $\tau$ that contains all of the variables $Y$ but does not contain $X$.
- Let $\Phi_X$ be the resulting set of factors.

Complexity

How does the graph $H_{\Phi_X}$ differ from $H_\Phi$?
- Constructing $\psi$ creates edges between all $Y \in Y$ (some were present in $H_\Phi$, others are fill edges, which are introduced in the elimination step).
- Eliminating $X$ from $\psi$ to construct $\tau$ has the effect of removing $X$ and all of its incident edges from the graph.

Complexity

Eliminating $C$

Complexity

Every factor that appears in one of the steps in the algorithm is reflected in the graph as a clique.
Complexity

Let $\Phi$ be a set of factors over $\mathcal{X} = \{X_1, \ldots, X_n\}$, and $< \text{ be an elimination ordering for some subset } \mathcal{X} \subseteq \mathcal{X}$.

The induced graph $I_{\Phi,<}$ is an undirected graph over $\mathcal{X}$, where $X_i$ and $X_j$ are connected by an edge if they both appear in some intermediate factor $\psi$ generated by the VE algorithm using $<$ as an elimination ordering.

Complexity

Let $I_{\Phi,<}$ be the induced graph for a set of factors $\Phi$ and some elimination ordering $<$. Then:

1. The scope of every factor generated during the variable elimination process is a clique in $I_{\Phi,<}$.
2. Every maximal clique in $I_{\Phi,<}$ is the scope of some intermediate factor in the computation.

(Proof omitted here)

Complexity

The induced graph for the student example. The edge G-S is the only fill edge introduced.

Clique tree in the induced graph

Complexity

- The width of an induced graph is defined as the number of nodes in the largest clique in the graph minus 1.
- The induced width $w_{\mathcal{K},<}$ of an ordering $<$ relative to a graph $\mathcal{K}$ (directed or undirected) is defined as the width of the graph $I_{\mathcal{K},<}$ induced by applying VE to $\mathcal{K}$ using the ordering $<$.
- The tree-width of a graph $\mathcal{K}$ to be its minimal induced width $w^*_{\mathcal{K}} = \min_{<} w(I_{\mathcal{K},<})$.
Complexity

The tree-width provides us a bound on the best performance we can hope for by applying VE to a probabilistic model that factorizes over $\mathcal{X}$.

Finding Elimination Orderings

Bad News:
- Determining whether there exists an elimination ordering achieving an induced width $\leq K$ (for some bound $K$) on a graph $H$ is NP-complete.
- Finding the optimal elimination order is NP-hard.

Even worse news:
- Even if we had the optimal elimination ordering, inference might require exponential time due to a large induced width.

Finding Elimination Orderings

NP-completeness? We remain unfazed!

How to find elimination orderings:
1. Graph theoretic approaches
2. Heuristic approaches
Finding Elimination Orderings

Graph-Theoretic Approaches
- Eliminate nodes such that you don’t produce fill edges
- Use the clique tree
  - Start eliminating from the leaves of the clique tree
- What if you don’t have the clique tree?
  - Use the Max-Cardinality algorithm (see pg 312 in book) on the original graph

Heuristic approaches use a greedy algorithm (could be done deterministically or stochastically)
- Requires a heuristic cost function.
- Examples of costs:
  - Min-neighbors: # of neighbors
  - Min-weight: domain cardinality of neighbors
  - Min-fill: # of fill edges added
  - Weighted min-fill: sum of weights of fill edges (weight = domain cardinality of vertices connected to the edge)

Heuristics work well in practice
- Min-fill and weighted min-fill tend to work the best