Exact Inference: Variable Elimination 3

1

Complexity of Variable Elimination

2

Complexity

Exponential size of the factors ψ dominates the complexity

- If each variable has no more than *v* values
- And a factor ψ_i has a scope that contains k_i variables, then the number of entries N_i in ψ_i is: $N_i \leq v^{k_i}$

3

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

4

Complexity

- Let Φ be a set of factors. We define

  $$Scope[\Phi] = \bigcup_{\phi \in \Phi} Scope[\phi]$$

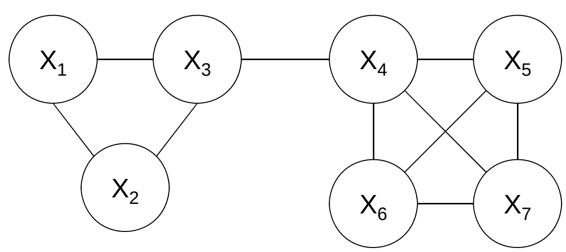
  to be the set of all variables appearing in one of the factors in Φ.
- We define $\mathcal{H}_\Phi$ to be the undirected graph whose nodes correspond to the variables in *Scope[Φ]* and where we have an edge X_i—$X_j \in \mathcal{H}_\Phi$ if and only if there exists a factor $\phi \in \Phi$ such that $X_i, X_j \in Scope[\phi]$

5

Complexity

(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 Φ

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



6

Complexity

- Proposition 9.1: Let *P* be a distribution defined by multiplying the factors in Φ and normalizing to define a distribution. Letting $\boldsymbol{X}$ = *Scope[*Φ*]*,

$$P(\boldsymbol{X}) = \frac{1}{Z}\prod_{\phi\in\Phi}\phi \quad \text{where } Z = \sum_{\boldsymbol{X}}\prod_{\phi\in\Phi}\phi$$

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

7

Complexity

- For a set of factors Φ 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

8

Complexity

When variable X is eliminated:

- Create a single factor ψ that contains X and all of the variables $\boldsymbol{Y}$ with which it appears in factors
- Eliminate X from ψ, replacing it with a new factor τ that contains all of the variables $\boldsymbol{Y}$ but does not contain X.
- Let Φ_X be the resulting set of factors

9

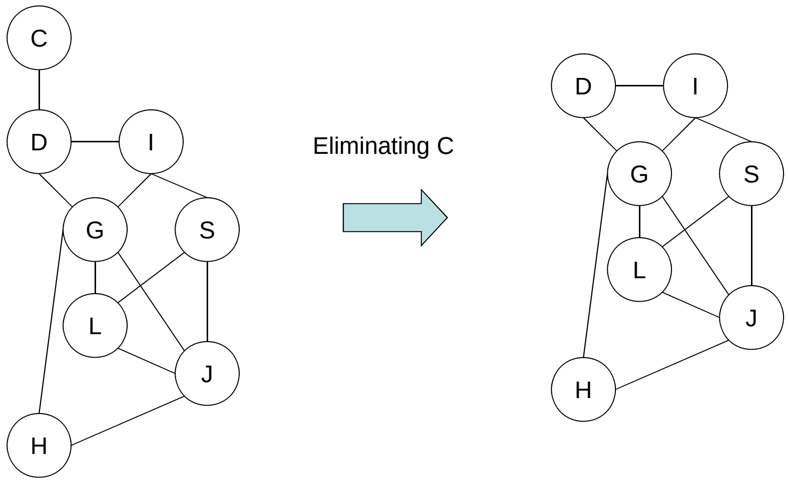
Complexity

How does the graph $\mathcal{H}_{\Phi_X}$ differ from $\mathcal{H}_{\Phi}$?

- Constructing ψ creates edges between all $Y \in \boldsymbol{Y}$ (some were present in $\mathcal{H}_{\Phi}$, others are fill edges, which are introduced in the elimination step)
- Eliminating X from ψ to construct τ has the effect of removing X and all of its incident edges from the graph

10

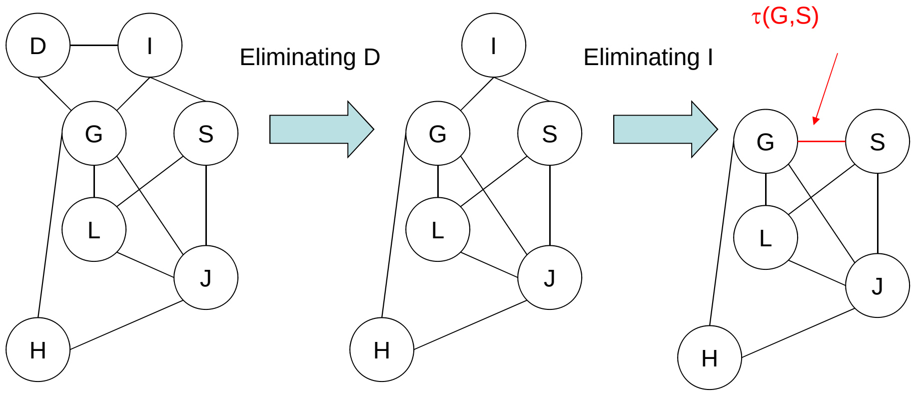
Complexity



11

Complexity

Fill in edge
for new
factor
$\tau(G,S)$



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

12

Complexity

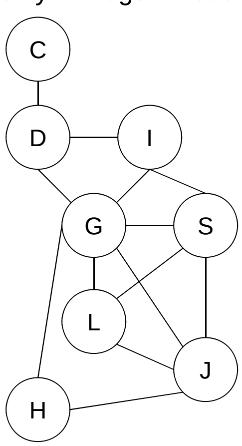
Let Φ be a set of factors over $\mathcal{X} = \{X_1, \ldots, X_n\}$, and $<$ be an elimination ordering for some subset $\boldsymbol{X} \subseteq \mathcal{X}$.

The induced graph $\mathcal{I}_{\Phi,<}$ 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 ψ generated by the VE algorithm using $<$ as an elimination ordering

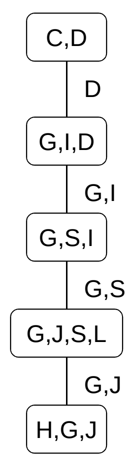
13

Complexity

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



Clique tree in the induced graph



14

Complexity

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

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

(Proof omitted here)

15

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 $\mathcal{I}_{\mathcal{K},<}$ induced by applying VE to $\mathcal{K}$ using the ordering <.
- The tree-width of a graph $\mathcal{K}$ to be its minimal induced width $w^*_{\mathcal{K}} = \min_< w(\mathcal{I}_{\mathcal{K},<})$

16

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 $\mathcal{K}$

17

Finding Elimination Orderings

18

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

19

Finding Elimination Orderings

NP-completeness? We remain unfazed!

How to find elimination orderings:

1. Graph theoretic approaches
2. Heuristic approaches

20

Finding Elimination Orderings

Graph-Theoretic Approaches

- Eliminate nodes such that you don't produce fill edges
- Use the clique tree
  - Start eliminating from the leafs 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

21

Finding Elimination Orderings

- 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)

22

Finding Elimination Orderings

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

23