CS181 Lecture 22: More Bayesian networks

Overview

- Inference in Bayesian networks
  - exact: variable elimination
    - example
    - in general
    - cost of variable elimination
  - approximate: sampling-based methods
- Learning in Bayesian networks

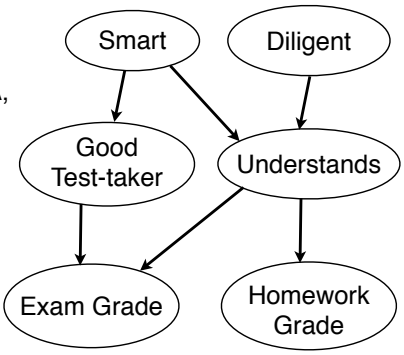
What can we do with a Bayesian net?

- We can compute the probability of any variable we are interested in, given any other variable
  - e.g. P(Accident | Age = 24, SpeedingTicketLastYear)
- We will now see a general algorithm to perform such computations
- The algorithm is called **variable elimination**

Worked out example

Want to know
P(Understands | Exam Grade = A, HW Grade = C)

P(U | E = A, H = C)



Note

$$P(U = u \mid E = A, H = C) = \frac{P(U = u, E = A, H = C)}{P(E = A, H = C)}$$

- Denominator is normalizing factor
- We can compute $P(U=u, E=A, H=C)$ for all values of U, and normalize

What we want

- We want

  $P(U = u, E = A, H = C)$
- Bayesian network gives us joint distribution

  $P(S = s, D = d, G = g, U = u, E = A, H = C)$
- We don't know (or care about) Smart, Diligent, or Good test-taker
- Compute joint and sum out unknown variables

  $$P(U = u, E = A, H = C) = \sum_{s,d,g} P(S = s, D = d, G = g, U = u, E = A, H = C)$$
- Problem: exponential number summation terms

Bayesian network distribution

- Use joint distribution from network

$$P(U=u,E=A,H=C)$$
$$=\sum_{s,d,g} P(S=s,D=d,G=g,U=u,E=A,H=C)$$
$$=\sum_{s,d,g} P(S=s)P(D=d)P(G=g \mid S=s)P(U=u \mid S=s,D=d)* P(E=A \mid G=g,U=u)P(H=C \mid U=u)$$

- Sum-of-products expression
- Solved using variable elimination

Idea: exploiting common factors

- Suppose we want to compute

$$xy + xw + xz + xu$$

- Number of operations

  - 4 multiplications + 3 additions = 7

- Instead, compute $x(y + w + z + u)$
- Number of operations

  - 3 additions, 1 multiplication: 4

Factors

- Each product term is a function from variable values to real numbers

  - $P(G = g \mid S = s)$ is a function from g and s to real numbers

  - $P(H = C \mid U = u)$ is a function of u, but not of the value of H, which is constant

- Each such term is called a **factor**
- We say that the factor **mentions** variables

  - e.g. $P(G = g \mid S = s)$ mentions G and S

Example factors

P(G | S)

S	G	
false	false	0.75
false	true	0.25
true	false	0.25
true	true	0.75

P(H = C | U)

U	
false	0.4
true	0.03

Giving the factors names

- $\phi_1 = P(S)$
- $\phi_2 = P(D)$
- $\phi_3 = P(G \mid S)$
- $\phi_4 = P(U \mid S,D)$
- $\phi_5 = P(E = A \mid G,U)$
- $\phi_6 = P(H = C \mid U)$

$$P(U=u,E=A,H=C) = \sum_{s,d,g} \phi_1(s)\phi_2(d)\phi_3(s,g)\phi_4(s,d,u)\phi_5(g,u)\phi_6(u)$$

Variable elimination

$$P(U=u,E=A,H=C) = \sum_{s,d,g} \phi_1(s)\phi_2(d)\phi_3(s,g)\phi_4(s,d,u)\phi_5(g,u)\phi_6(u)$$

- Eliminates summation variables one at a time
- At the end, we have a factor over the remaining variables
- Need elimination order

  - choose g,s,d

Eliminating good test-taker

$$P(U=u,E=A,H=C)$$
$$= \sum_{s,d,g} \phi_1(s)\phi_2(d)\phi_3(s,g)\phi_4(s,d,u)\phi_5(g,u)\phi_6(u)$$
$$= \sum_{s,d} \sum_{g} \phi_1(s)\phi_2(d)\phi_3(s,g)\phi_4(s,d,u)\phi_5(g,u)\phi_6(u)$$

Eliminating good test-taker

$$P(U=u,E=A,H=C)$$
$$= \sum_{s,d,g} \phi_1(s)\phi_2(d)\phi_3(s,g)\phi_4(s,d,u)\phi_5(g,u)\phi_6(u)$$
$$= \sum_{s,d} \sum_{g} \phi_1(s)\phi_2(d)\phi_3(s,g)\phi_4(s,d,u)\phi_5(g,u)\phi_6(u)$$
$$= \sum_{s,d} \sum_{g} \phi_1(s)\phi_2(d)\phi_4(s,d,u)\phi_6(u)\phi_3(s,g)\phi_5(g,u)$$

Eliminating good test-taker

$$P(U=u,E=A,H=C)$$
$$= \sum_{s,d,g} \phi_1(s)\phi_2(d)\phi_3(s,g)\phi_4(s,d,u)\phi_5(g,u)\phi_6(u)$$
$$= \sum_{s,d} \sum_{g} \phi_1(s)\phi_2(d)\phi_3(s,g)\phi_4(s,d,u)\phi_5(g,u)\phi_6(u)$$
$$= \sum_{s,d} \sum_{g} \phi_1(s)\phi_2(d)\phi_4(s,d,u)\phi_6(u)\phi_3(s,g)\phi_5(g,u)$$
$$= \sum_{s,d} \phi_1(s)\phi_2(d)\phi_4(s,d,u)\phi_6(u)\sum_{g} \phi_3(s,g)\phi_5(g,u)$$

Evaluating

$$\sum_{g} \phi_3(s,g)\phi_5(g,u)$$

- This is a sum of products expression that cannot be further decomposed
- We must compute it explicitly, using the definitions of the factors

Definitions of the factors

$$\sum_{g} \phi_3(s,g)\phi_5(g,u)$$

$\phi_3(s,g)$ = P(G | S) =

S	G	ϕ_3
false	false	0.75
false	true	0.25
true	false	0.25
true	true	0.75

$\phi_5(g,u)$ = P(E=A | G, U) =

G	U	ϕ_5
false	false	0.05
false	true	0.4
true	false	0.3
true	true	0.7

Computing $\phi_3 * \phi_5$

S	G	ϕ_3
false	false	0.75
false	true	0.25
true	false	0.25
true	true	0.75

*

G	U	ϕ_5
false	false	0.05
false	true	0.4
true	false	0.3
true	true	0.7

=

S	G	U	$\phi_3 * \phi_5$
false	false	false	
false	false	true	
false	true	false	
false	true	true	
true	false	false	
true	false	true	
true	true	false	
true	true	true	

Computing $\phi_3 * \phi_5$

S	G	ϕ_3
false	false	0.75
false	true	0.25
true	false	0.25
true	true	0.75

*

G	U	ϕ_5
false	false	0.05
false	true	0.4
true	false	0.3
true	true	0.7

=

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.75
false	false	true	
false	true	false	
false	true	true	
true	false	false	
true	false	true	
true	true	false	
true	true	true	

Computing $\phi_3 * \phi_5$

S	G	ϕ_3
false	false	0.75
false	true	0.25
true	false	0.25
true	true	0.75

*

G	U	ϕ_5
false	false	0.05
false	true	0.4
true	false	0.3
true	true	0.7

=

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.75*0.05
false	false	true	
false	true	false	
false	true	true	
true	false	false	
true	false	true	
true	true	false	
true	true	true	

Computing $\phi_3 * \phi_5$

S	G	ϕ_3
false	false	0.75
false	true	0.25
true	false	0.25
true	true	0.75

*

G	U	ϕ_5
false	false	0.05
false	true	0.4
true	false	0.3
true	true	0.7

=

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.75*0.05 = 0.0375
false	false	true	
false	true	false	
false	true	true	
true	false	false	
true	false	true	
true	true	false	
true	true	true	

Computing $\phi_3 * \phi_5$

S	G	ϕ_3
false	false	0.75
false	true	0.25
true	false	0.25
true	true	0.75

*

G	U	ϕ_5
false	false	0.05
false	true	0.4
true	false	0.3
true	true	0.7

=

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	
false	true	false	
false	true	true	
true	false	false	
true	false	true	0.25
true	true	false	
true	true	true	

Computing $\phi_3 * \phi_5$

S	G	ϕ_3
false	false	0.75
false	true	0.25
true	false	0.25
true	true	0.75

*

G	U	ϕ_5
false	false	0.05
false	true	0.4
true	false	0.3
true	true	0.7

=

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	
false	true	false	
false	true	true	
true	false	false	
true	false	true	
true	true	false	
true	true	true	

Computing $\phi_3 * \phi_5$

S	G	ϕ_3
false	false	0.75
false	true	0.25
true	false	0.25
true	true	0.75

*

G	U	ϕ_5
false	false	0.05
false	true	0.4
true	false	0.3
true	true	0.7

=

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	
false	true	false	
false	true	true	
true	false	false	
true	false	true	0.25*0.4 = 0.1
true	true	false	
true	true	true	

Computing $\phi_3 * \phi_5$

S	G	ϕ_3
false	false	0.75
false	true	0.25
true	false	0.25
true	true	0.75

*

G	U	ϕ_5
false	false	0.05
false	true	0.4
true	false	0.3
true	true	0.7

=

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	0.75*0.4 = 0.3
false	true	false	0.25*0.3 = 0.075
false	true	true	0.25*0.7 = 0.175
true	false	false	0.25*0.05 = 0.0125
true	false	true	0.1
true	true	false	0.75*0.3 = 0.225
true	true	true	0.75*0.7 = 0.525

Computing $\Sigma_g \phi_3 * \phi_5$

$\sum_g$

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	0.3
false	true	false	0.075
false	true	true	0.175
true	false	false	0.0125
true	false	true	0.1
true	true	false	0.225
true	true	true	0.525

=

S	U	$\Sigma_g \phi_3 * \phi_5$
false	false	
false	true	
true	false	
true	true	

Computing $\Sigma_g \phi_3 * \phi_5$

$\sum_g$

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	0.3
false	true	false	0.075
false	true	true	0.175
true	false	false	0.0125
true	false	true	0.1
true	true	false	0.225
true	true	true	0.525

=

S	U	$\Sigma_g \phi_3 * \phi_5$
false	false	0.0375
false	true	
true	false	
true	true	

Computing $\Sigma_g \phi_3 * \phi_5$

$\sum_g$

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	0.3
false	true	false	0.075
false	true	true	0.175
true	false	false	0.0125
true	false	true	0.1
true	true	false	0.225
true	true	true	0.525

=

S	U	$\Sigma_g \phi_3 * \phi_5$
false	false	0.0375 + 0.075
false	true	
true	false	
true	true	

Computing $\Sigma_g \phi_3 * \phi_5$

$\sum_g$

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	0.3
false	true	false	0.075
false	true	true	0.175
true	false	false	0.0125
true	false	true	0.1
true	true	false	0.225
true	true	true	0.525

=

S	U	$\Sigma_g \phi_3 * \phi_5$
false	false	0.0375 + 0.075 = 0.1125
false	true	
true	false	
true	true	

Computing $\Sigma_g \phi_3 * \phi_5$

$\sum_g$

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	0.3
false	true	false	0.075
false	true	true	0.175
true	false	false	0.0125
true	false	true	0.1
true	true	false	0.225
true	true	true	0.525

=

S	U	$\Sigma_g \phi_3 * \phi_5$
false	false	0.1125
false	true	
true	false	
true	true	0.1

Computing $\Sigma_g\ \phi_3 * \phi_5$

$\sum_g$

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	0.3
false	true	false	0.075
false	true	true	0.175
true	false	false	0.0125
true	false	true	0.1
true	true	false	0.225
true	true	true	0.525

=

S	U	$\Sigma_g\ \phi_3 * \phi_5$
false	false	0.1125
false	true	
true	false	
true	true	0.1 + 0.525

Computing $\Sigma_g\ \phi_3 * \phi_5$

$\sum_g$

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	0.3
false	true	false	0.075
false	true	true	0.175
true	false	false	0.0125
true	false	true	0.1
true	true	false	0.225
true	true	true	0.525

=

S	U	$\Sigma_g\ \phi_3 * \phi_5$
false	false	0.1125
false	true	
true	false	
true	true	0.1 + 0.525 = 0.625

Computing $\Sigma_g\ \phi_3 * \phi_5$

$\sum_g$

S	G	U	$\phi_3 * \phi_5$
false	false	false	0.0375
false	false	true	0.3
false	true	false	0.075
false	true	true	0.175
true	false	false	0.0125
true	false	true	0.1
true	true	false	0.225
true	true	true	0.525

=

S	U	$\Sigma_g\ \phi_3 * \phi_5$
false	false	0.1125
false	true	0.3 + 0.175 = 0.475
true	false	0.013 + 0.225 = 0.2375
true	true	0.625

Eliminating good test-taker

$\Phi_G(s,u) =$

S	U	$\Sigma_g\ \phi_3 * \phi_5$
false	false	0.1125
false	true	0.475
true	false	0.2375
true	true	0.625

- Replace $\sum_g \phi_3(s,g)\phi_5(g,u)$ with $\Phi_G(s,u)$
- New expression:

$$\sum_{s,d} \phi_1(s)\phi_2(d)\phi_4(s,d,u)\phi_6(u)\Phi_G(s,u)$$

Eliminating smart

$$P(U=u, E=A, H=C)$$

$$= \sum_{s,d} \phi_1(s)\phi_2(d)\phi_4(s,d,u)\phi_6(u)\Phi_G(s,u)$$

$$= \sum_{d} \phi_2(d)\phi_6(u) \sum_{s} \phi_1(s)\phi_4(s,d,u)\Phi_G(s,u)$$

Computing $\phi_1 * \phi_4 * \Phi_G$

$\phi_1(s) = P(S)$

S	ϕ_1
false	0.5
true	0.5

*

$\phi_4(s,d,u) = P(U \mid S,D)$

S	D	U	ϕ_4
false	false	false	0.8
false	false	true	0.2
false	true	false	0.4
false	true	true	0.6
true	false	false	0.4
true	false	true	0.6
true	true	false	0.05
true	true	true	0.95

*

$\Phi_G(s,u)$

S	U	Φ_G
false	false	0.1125
false	true	0.475
true	false	0.2375
true	true	0.625

S	D	U	$\phi_1 * \phi_4 * \Phi_G(S,U)$
false	false	false	0.5*0.8*0.1125
false	false	true	
...			
true	true	true	

Computing $\phi_1 * \phi_4 * \Phi_G$

$\phi_1(s) = P(S)$

S	ϕ_1
false	0.5
true	0.5

*

$\phi_4(s,d,u) = P(U \mid S,D)$

S	D	U	ϕ_4
false	false	false	0.8
false	false	true	0.2
false	true	false	0.4
false	true	true	0.6
true	false	false	0.4
true	false	true	0.6
true	true	false	0.05
true	true	true	0.95

*

$\Phi_G(s,u)$

S	U	Φ_G
false	false	0.1125
false	true	0.475
true	false	0.2375
true	true	0.625

S	D	U	$\phi_1 * \phi_4 * \Phi_G(S,U)$
false	false	false	0.045
false	false	true	0.5*0.2*0.475
true	true	true	

Computing $\phi_1 * \phi_4 * \Phi_G$

$\phi_1(s) = P(S)$

S	ϕ_1
false	0.5
true	0.5

*

$\phi_4(s,d,u) = P(U \mid S,D)$

S	D	U	ϕ_4
false	false	false	0.8
false	false	true	0.2
false	true	false	0.4
false	true	true	0.6
true	false	false	0.4
true	false	true	0.6
true	true	false	0.05
true	true	true	0.95

*

$\Phi_G(s,u)$

S	U	Φ_G
false	false	0.1125
false	true	0.475
true	false	0.2375
true	true	0.625

S	D	U	$\phi_1 * \phi_4 * \Phi_G(S,U)$
false	false	false	0.045
false	false	true	0.0475
true	true	true	0.5*0.95*0.625

Computing $\Sigma_s \phi_1 * \phi_4 * \Phi_G$

$\sum_s$

S	D	U	$\phi_1 * \phi_4 * \Phi_G(S,U)$
false	false	false	0.045
false	false	true	0.0475
false	true	false	0.0225
false	true	true	0.1425
true	false	false	0.0475
true	false	true	0.1875
true	true	false	0.00594
true	true	true	0.2969

=

D	U	$\Sigma_s \phi_1 * \phi_4 * \Phi_G(S,U)$
false	false	0.0925
false	true	0.235
true	false	0.02844
true	true	0.4394

Eliminating smart

$$P(U \mid E = A, H = C)$$

$$= \sum_{s,d} \phi_1(s)\phi_2(d)\phi_4(s,d,u)\phi_6(u)\Phi_G(s,u)$$

$$= \sum_{d} \phi_2(d)\phi_6(u) \sum_{s} \phi_1(s)\phi_4(s,d,u)\Phi_G(s,u)$$

$$= \sum_{d} \phi_2(d)\phi_6(u)\Phi_S(d,u)$$

$\Phi_S(d,u) =$

D	U	$\Sigma_s \phi_1 * \phi_4 * \Phi_G(S,U)$
false	false	0.0925
false	true	0.235
true	false	0.02844
true	true	0.4394

Eliminating diligent

$$P(U = u, E = A, H = C)$$

$$= \phi_6(u) \sum_{d} \phi_2(d)\Phi_S(d,u)$$

$\sum_d$

$\phi_2(d) = P(D)$

D	ϕ_2
false	0.3
true	0.7

*

$\Phi_S(d,u)$

D	U	$\Sigma_s \phi_1 * \phi_4 * \Phi_G(s,u)$
false	false	0.0925
false	true	0.235
true	false	0.02844
true	true	0.4394

=

U	$\Phi_D(u)$
false	0.3*0.0925+0.7*0.02844 = 0.04766
true	0.3*0.235+0.7*0.4394 = 0.3781

Finishing up

$$P(U = u, E = A, H = C)$$

$$= \phi_6(u)\Phi_D(u)$$

$\phi_6(u) = P(H=C \mid U)$

U	ϕ_6
false	0.4
true	0.03

*

$\Phi_D(u)$

U	$\Phi_D(u)$
false	0.04766
true	0.3781

=

U	P(U=u,E=A,H=C)
false	0.019
true	0.0113

Renormalizing

$$P(U = u \mid E = A, H = C) = \frac{P(U = u, E = A, H = C)}{P(E = A, H = C)} = \frac{P(U = u, E = A, H = C)}{\sum_u P(U = u, E = A, H = C)}$$

U	P(U=u,E=A,H=C)
false	0.019
true	0.0113

normalize

U	P(U=u | E=A,H=C)
false	0.627
true	0.373

What was the point?

- What did we gain over enumerating all the summation terms explicitly?
- When we eliminate a variable, how many rows does the table have?
  - exponential in number of variables co-mentioned with that variable
- Without variable elimination, exponential in number of summation variables

Overview

- Inference in Bayesian networks
  - exact: variable elimination
    - example
    - in general
    - cost of variable elimination
  - approximate: sampling-based methods
- Learning in Bayesian networks

The task

- Given:
  - a Bayesian network
  - evidence variables $E_1,\ldots,E_m$
  - evidence values $e_1,\ldots,e_m$
  - query variables $Q_1,\ldots,Q_k$
- Goal:
  - compute $P(Q_1,\ldots,Q_k \mid E_1 = e_1, \ldots, E_m = e_m)$

Notation

- Order the variables of the BN so that the query variables come first, then the evidence variables, then the remaining variables
- Let $X_1,\ldots,X_m$ be the query variables
- Let $X_{m+1},\ldots,X_k$ be the evidence variables
- Let $X_{k+1},\ldots,X_n$ be the remaining variables

What we want

- We want

$$P(X_1,\ldots,X_m,x_{m+1},\ldots,x_k) = \sum_{x_{k+1}\in Dom[X_{k+1}]} \ldots \sum_{x_n\in Dom[X_n]} P(X_1,\ldots,X_m,x_{m+1},\ldots,x_k,x_{k+1},\ldots,x_n)$$

- $P(X_1,\ldots,X_m,x_{m+1},\ldots,x_k,x_{k+1},\ldots,x_n)$ is defined by the BN
- In principle, we could compute this for all values of $X_{k+1},\ldots,X_n$
- Problem: number of summation terms is exponential in $n-k$

Better approach

- Use

$$P(X_1,...,X_m,x_{m+1},...,x_k) = \sum_{x_{k+1}} ... \sum_{x_n} \prod_{i=1}^{n} P(x_i \mid Pa[x_i])$$

- This is a sum-of-products expression
- Solve using variable elimination

Variable elimination

Repeat until all variables have been eliminated:

Choose a variable X_i to eliminate.

Move Σ_{Xi} to the right of other summations.

Rearrange factors so that all factors mentioning X_i are to the right.

Using the distributive law, move Σ_{Xi} to the right of all factors not mentioning X_i.

$\psi_{Xi} = \Sigma_{Xi}\ \psi_1 \cdots \psi_k$ mentioning all variables mentioned in $\psi_1 \cdots \psi_k$ (except X_i)

Multiply the remaining factors to get $P(X_1,\ldots,X_l,x_{l+1},\ldots,x_k)$

Normalize to get $P(X_1,\ldots,X_l \mid x_{l+1},\ldots,x_k)$

The point

- When we eliminate a variable, it will only enclose terms involving a few other variables, so computing the product and sum is feasible
- But how much do we actually save?
- Is there a way to know in advance of doing the computation how much it will cost?

Overview

- Inference in Bayesian networks
  - exact: variable elimination
    - example
    - in general
    - cost of variable elimination
  - approximate: sampling-based methods
- Learning in Bayesian networks

Basic cost

- Let ϕ_i be the intermediate factor produced while eliminating X_i
- Let $|\phi_i|$ be the number of rows in ϕ_i
- Cost of eliminating X_i: $O(|\phi_i|)$
- $|\phi_i|$ is exponential in number of variables mentioned by ϕ_i
- **Therefore cost of VE is exponential in maximum number of variables mentioned by any intermediate ϕ_i**

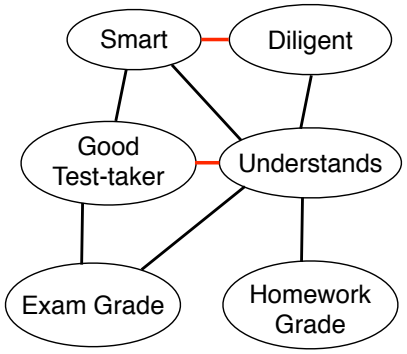
Understanding the cost

- What is this maximum number?
- It is hard to read off the graph
- Given an elimination ordering, we can find it

Moral graph

- The **moral graph** of a BN is an **undirected graph** in which:
  - the nodes are the nodes of the BN
  - there is an edge between X and Y if
    - X is a parent of Y
    - Y is a parent of X, or
    - X and Y are both parents of the same variable

Moral graph example



Why the moral graph matters

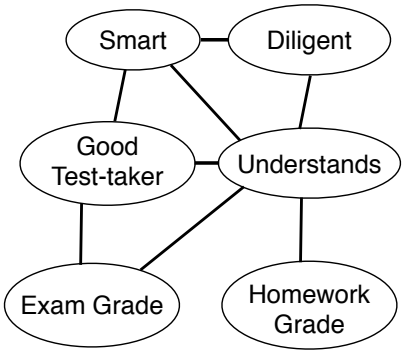
- There is an edge between two nodes in the moral graph iff they appear together in the same CPD
  - parent and child both appear in the child's CPD
  - two parents of the same child both appear in the child's CPD

Induced graph for elim. order

1. Begin with the moral graph
2. Delete all evidence variables and the edges coming out of them
3. For each eliminated variable, in order:
   1. Connect all its uneliminated neighbors
   2. Mark the node as eliminated

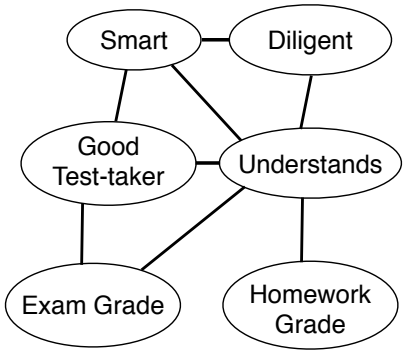
Induced graph for order S,D,G

Begin with moral graph



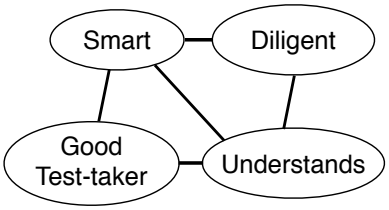
Induced graph for order S,D,G

Delete evidence variables

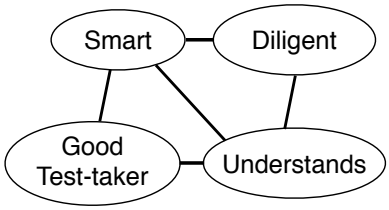


Induced graph for order S,D,G

Delete evidence variables

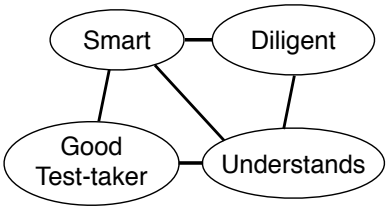


Induced graph for order S,D,G



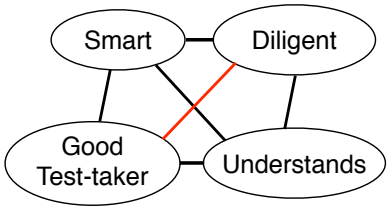
Eliminate S

Induced graph for order S,D,G



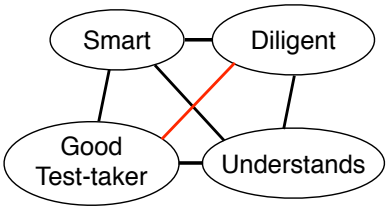
Eliminate S
Connect uneliminated neighbors

Induced graph for order S,D,G



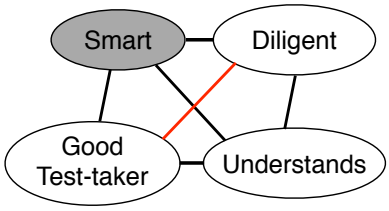
Eliminate S
Connect uneliminated neighbors

Induced graph for order S,D,G



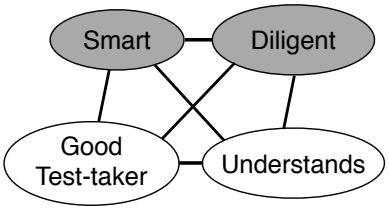
Eliminate S
Connect uneliminated neighbors
Mark S as eliminated

Induced graph for order S,D,G



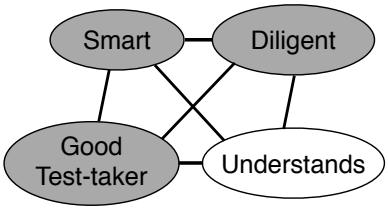
Eliminate S
Connect uneliminated neighbors
Mark S as eliminated

Induced graph for order S,D,G



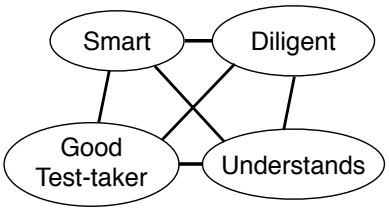
Eliminate D
Connect uneliminated neighbors
Mark D as eliminated

Induced graph for order S,D,G



Eliminate G
Connect uneliminated neighbors
Mark G as eliminated

Final induced graph for order S,D,G



Clique

- A **complete** set of nodes in a graph is a set of nodes that are all connected to each other
- A **clique** is a maximal complete set
  - maximal: adding any other node would make the set no longer complete

Claims (proved in notes)

- Claim 1: For any clique in the induced graph, there is a factor used during the VE process that contains all the variables in the clique
- Claim 2: If a factor used during the VE process mentions a set of variables, the variables form a complete set in the induced graph

Conclusion

- The number of variables mentioned by the largest factor is the size of the largest clique in the induced graph
- Therefore the cost of variable elimination is **exponential** in the size of the largest clique in the induced graph

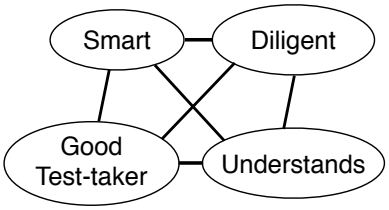
Complexity

- In general, the task of computing probabilities in a BN is #P hard
- This is because the maximal clique in the induced graph may be large, even for simple BNs
- But for many BNs, the size of the maximal clique is small
- Thus inference in BNs is often tractable

Elimination order is crucial

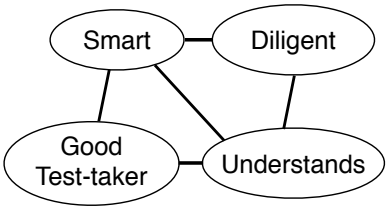
- The induced graph is defined relative to an elimination order
- If we have two different orders, we may get two very different induced graphs
  - different size of largest clique
- Therefore the cost of VE may be very different for two different orders

Example: order S,D,G



Largest clique size is 4

Example: order D,S,G



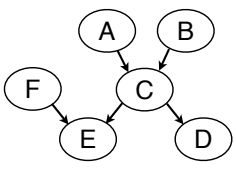
Largest clique size is 3

Finding an elimination order

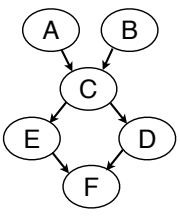
- Finding the optimal elimination order in a BN is NP-hard
- But simple heuristics work well and can quickly find good elimination orders in many cases

Polytrees

A **polytree** is a BN with no undirected cycles



polytree



not polytree

Polytrees are easy

- For a polytree, there is an elimination order that adds no edges to the moral graph
- Therefore a polytree can be solved in time linear in the size of the network
- Algorithm:
  - Root the polytree at the query node
  - Eliminate from the leaves upwards

Overview

- Inference
  - exact: variable elimination
  - approximate: sampling-based methods
- Learning

Need approximation

- Exact inference methods are computationally infeasible in the general case
- We need to sacrifice exactness for computational tractability

Generative model

- Recall that Bayesian networks define a generative process:
  - For each node X, **in topological order**
    - Let $\mathbf{u}$ be the previously generated values for $\mathrm{Pa}[X]$
    - Sample a value x for X according to the distribution $P(X \mid \mathbf{u})$

Sample distribution

- Distribution defined by Bayesian network is distribution of samples generated in this way
- N = number of samples
- $N(X_1=x_1,...,X_n=x_n)$ = number of samples with attribute values $x_1,...x_n$

$$\lim_{N\to\infty} \frac{N(X_1 = x_1,...,X_n = x_n)}{N} = P(X_1 = x_1,...,X_n = x_n)$$

Rejection sampling

- We want P(query | evidence)
- Generate N samples according to BN
- Reject all samples inconsistent with evidence
- Estimate probability with remaining samples

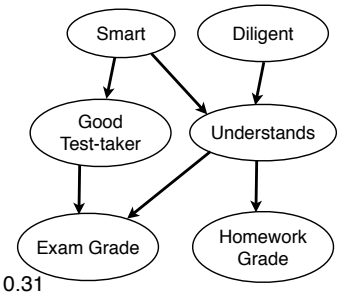
$$P(\text{query}) \approx \frac{N(\text{query},\text{evidence})}{N(\text{evidence})}$$

Example

Want to know
P(U | E = A, H = C)

Generate 1000 samples.

29 consistent with evidence
9 have U = true
20 have U = false



P(U = true | E=A, H=C) ≈ 9/29 = 0.31
P(U = false | E=A, H=C) ≈ 20/29 = 0.69

Exact distribution from variable elimination:
P(U = true) = 0.373
P(U = false) = 0.627

Rejection sampling

- What's wrong with this technique?
- We reject a lot of samples!
  - the more evidence we have, the more samples we reject
  - the less likely our evidence, the more samples we reject

Likelihood weighting

- Instead, weight each sample by the likelihood of the evidence given the rest of the sample
- Use weighted counts over **all** samples to estimate probabilities

Example

- Generate non-evidence according to BN:
  - [S = true, D = true, G = true, U = true, E = a, H = c]
- Compute likelihood of evidence (E = a, H = c):
  - P(E = a | G = true, U = true) = 0.7
  - P(H = c | U = true) = 0.03
  - weight = 0.7*0.25 = 0.021
- Add weight to count for U = true

Correctness of likelihood weighting

- E = evidence variables
- Z = remaining variables
- Sampling distribution for non-evidence variables

$$S_{WS}(\mathbf{z},\mathbf{e}) = \prod_{i=1}^{k} P(z_i \mid Pa[Z_i])$$

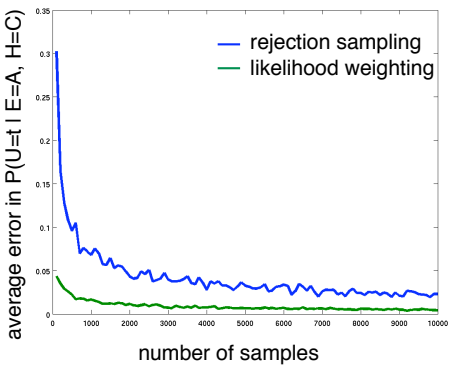
- Weight for each sample

$$w(\mathbf{z},\mathbf{e}) = \prod_{i=1}^{m} P(e_i \mid Pa[E_i])$$

- Weighted probability of a sample

$$S_{WS}(\mathbf{z},\mathbf{e})w(\mathbf{z},\mathbf{e}) = \prod_{i=1}^{k} P(z_i \mid Pa[Z_i]) \prod_{i=1}^{m} P(e_i \mid Pa[E_i]) = P(\mathbf{z},\mathbf{e})$$

Sampling comparison



Weaknesses of likelihood weighting

- Performance degrades as number of evidence variables goes up
  - most samples have very low weight
- Worse when evidence variables appear late in variables ordering
  - evidence is not reflected in sampling

Sampling summary

- Exact inference is intractable in large complex (realistic) networks
- Sampling is a powerful technique for computing approximations in probabilistic models
- More sophisticated sampling techniques increase sampling efficiency
  - e.g. Markov chain Monte Carlo (MCMC)

Overview

- Inference
  - exact: variable elimination
  - approximate: sampling-based
- Learning

Learning Bayesian networks

- Given a model, inference allows us to reason about uncertainty
- But how can we generate the model from data?

Known structure, complete data

- Given: graph structure of Bayesian network
- Input: instances of values for each variable

$$\langle X_1 = x_1^1, \ldots, X_n = x_n^1 \rangle$$
$$\langle X_1 = x_1^2, \ldots, X_n = x_n^2 \rangle$$
$$\ldots$$
$$\langle X_N = x_1^N, \ldots, X_n = x_n^N \rangle$$

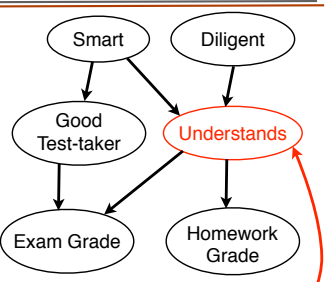
- Goal: learn CPD for each variable

Maximum likelihood

- Compute counts
  - N_i^{μ} = number of times $Pa[X_i]$ had value μ
  - $N_{i,x}^{\mu}$ = number of times variable X_i had value x, $Pa[X_i]$ had value μ
- Compute parameters
  - $\theta_{i,x}^{\mu} = P(X_i = x_i \mid Pa[X_i] = \mu)$

Example

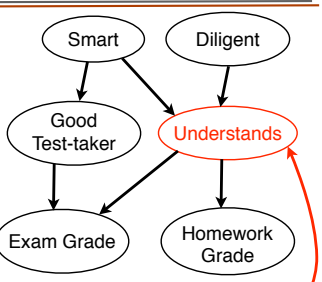
S	D	G	U	E	H
t	t	t	t	a	a
f	t	f	f	b	b
t	f	f	t	d	a
f	t	f	f	b	b
t	t	f	t	b	b
t	t	t	t	b	b
f	t	f	t	a	a
f	f	f	t	a	a
t	t	t	t	a	a
f	t	t	t	a	a
f	f	f	f	f	b
t	t	t	t	a	b
f	t	f	t	b	a
t	f	t	f	b	c



S	D	P(U I s,d)	
f	f		
f	t		
t	f		
t	t		

Example

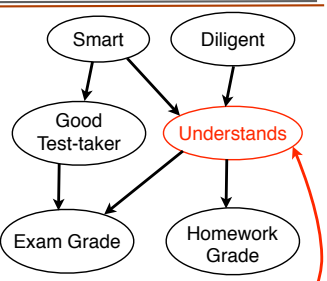
S	D	G	U	E	H
t	t	t	t	a	a
f	t	f	f	b	b
t	f	f	t	d	a
f	t	f	f	b	b
t	t	f	t	b	b
t	t	t	t	b	b
f	t	f	t	a	a
f	f	f	t	a	a
t	t	t	t	a	a
f	t	t	t	a	a
f	f	f	f	f	b
t	t	t	t	a	b
f	t	f	t	b	a
t	f	t	f	b	c



S	D	P(U I s,d)	
f	f		
f	t		
t	f		
t	t		

Example

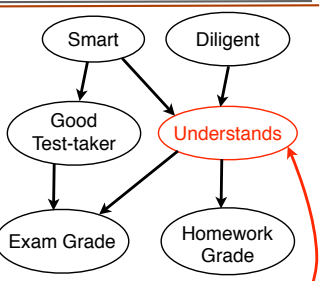
S	D	G	U	E	H
t	t	t	t	a	a
f	t	f	f	b	b
t	f	f	t	d	a
f	t	f	f	b	b
t	t	f	t	b	b
t	t	t	t	b	b
f	t	f	t	a	a
f	f	f	t	a	a
t	t	t	t	a	a
f	t	t	t	a	a
f	f	f	f	f	b
t	t	t	t	a	b
f	t	f	t	b	a
t	f	t	f	b	c



S	D	P(U I s,d)	
f	f	0.5	0.5
f	t		
t	f		
t	t		

Example

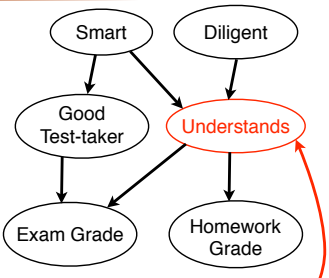
S	D	G	U	E	H
t	t	t	t	a	a
f	t	f	f	b	b
t	f	f	t	d	a
f	t	f	f	b	b
t	t	f	t	b	b
t	t	t	t	b	b
f	t	f	t	a	a
f	f	f	t	a	a
t	t	t	t	a	a
f	t	t	t	a	a
f	f	f	f	f	b
t	t	t	t	a	b
f	t	f	t	b	a
t	f	t	f	b	c



S	D	P(U I s,d)	
f	f	0.5	0.5
f	t	0.6	0.4
t	f		
t	t		

Example

S	D	G	U	E	H
t	t	t	t	a	a
f	t	f	f	b	b
t	f	f	t	d	a
f	t	f	f	b	b
t	t	f	t	b	b
t	t	t	t	b	b
f	t	f	t	a	a
f	f	f	t	a	a
t	t	t	t	a	a
f	t	t	t	a	a
f	f	f	f	f	b
t	t	t	t	a	b
f	t	f	t	b	a
t	f	t	f	b	c



S	D	P(U I s,d)	
f	f	0.5	0.5
f	t	0.6	0.4
t	f	0.5	0.5
t	t	1.0	0.0

Known structure, incomplete data

- Given: graph structure of Bayesian network
- Input: instances of values for **some** variables

$$\langle X_1 = x_1^1, X_2 = ?, ..., X_n = x_n^1 \rangle$$

$$\langle X_1 = ?, X_2 = x_2^1 ..., X_n = ? \rangle$$

...

$$\langle X_N = x_1^N, .X_2 = x_2^N .., X_n = x_n^N \rangle$$

- Goal: learn CPD for each variable

Familiar problem?

- Given HMM structure
- Input: observation sequences

$\langle o_1^1, ..., o_{n^1}^1 \rangle$

$\langle o_1^2, ..., o_{n^2}^2 \rangle$

...

$\langle o_1^N, ..., o_{n^N}^N \rangle$

- sequences may be different lengths

- Goal: Learn HMM parameters

Expectation maximization

- HMM is special case of Bayesian network
- Like Autoclass and Baum-Welch
  - expectation step computes expected sufficient statistics
    - uses inference techniques
  - maximization step computes maximum likelihood parameters
    - conditional probabilities for each variable

Unknown structure, complete data

- Input: instances of values for all variables

$\langle X_1 = x_1^1, ..., X_n = x_n^1 \rangle$

$\langle X_1 = x_1^2, ..., X_n = x_n^2 \rangle$

...

$\langle X_N = x_1^N, ..., X_n = x_n^N \rangle$

- Goal: learn network structure and CPDs for all variables

Structure score

- Complete data: ML parameters can be learned quickly by counting
- Likelihood of data given parameters provides a score for the structure
  - for a structure G

$$\text{Score}(G) = \sum_{i=1}^{N} \log P(\mathbf{x}^i \mid G)$$

- What is the problem with this approach?

Overfitting

- Maximizing likelihood of the data tends to overfit
  - complex models, such as fully connected, can fit training data very well, but generalize poorly
- Use Bayesian score instead

$$\text{Score}(G) = \log \int P(D \mid G, \theta_G) P(\theta_G \mid G) d\theta$$

- Difficult integral: Bayesian information criterion (BIC) is an approximation

$$\text{Score}(G) = \log P(D \mid G, \theta_G) - \frac{|\theta_G|}{2} \log N$$

- BIC score helps combat overfitting

Structure search

- Ideally, choose structure with best score
  - too many structures to consider them all
- Instead, do greedy local search
  - consider adding, removing, reversing edges
  - compute score, take step if score improved

Hidden variables

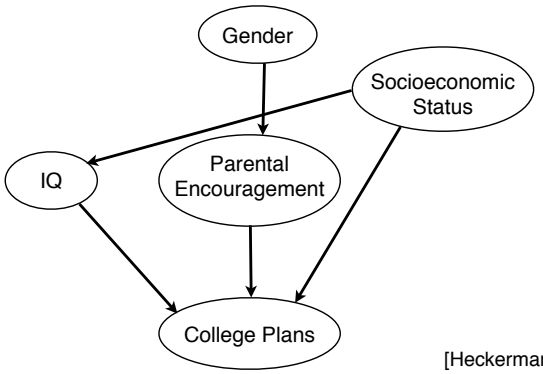
- We may be better able to model the data by hypothesizing hidden variables
  - e.g. cause of multiple variables
- Structure search can include steps to add hidden variables

Example: college plans

- Gender
- Socioeconomic status: low, lower middle, upper middle, high
- IQ: low, lower middle, upper middle, high
- Parental encouragement: low, high
- College plans: yes, no

Example: college plans

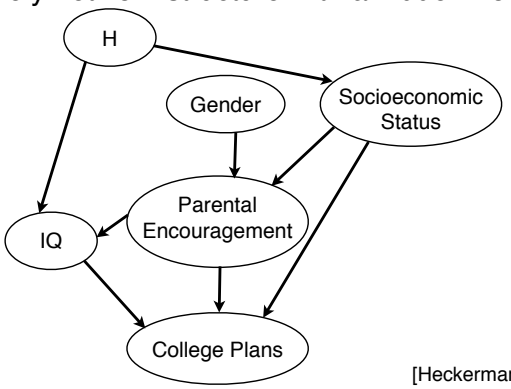
Most likely network structure



[Heckerman, 2008]

Example: college plans

Most likely network structure with a hidden node



[Heckerman, 2008]