COMP90051 **Statistical Machine Learning**

Semester 2, 2015

Lecturer: Ben Rubinstein

8. PGM Probabilistic Inference

THE UNIVERSITY OF MELBOURNE

Probabilistic inference on PGMs

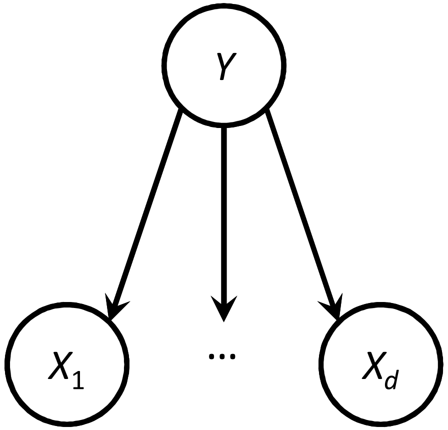
Computing marginal and conditional distributions from the joint of a PGM using Bayes rule and marginalisation.

This deck: how to do it efficiently.

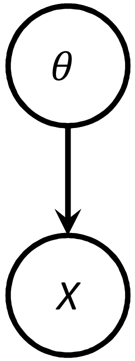
Based on Andrew Moore's tutorial slides

Two familiar examples

- Naïve Bayes (frequentist/Bayesian)
  - Chooses most likely class given data
  - $\Pr(Y|X_1,\ldots,X_d) = \frac{\Pr(Y,X_1,\ldots,X_d)}{\Pr(X_1,\ldots,X_d)} = \frac{\Pr(Y,X_1,\ldots,X_d)}{\sum_y \Pr(Y=y,X_1,\ldots,X_d)}$



- Data $X|\theta \sim N(\theta, 1)$ with prior $\theta \sim N(0,1)$ (Bayesian)
  - Given observation $X = x$ update posterior
  - $\Pr(\theta|X) = \frac{\Pr(\theta,X)}{\Pr(X)} = \frac{\Pr(\theta,X)}{\sum_\theta \Pr(\theta,X)}$

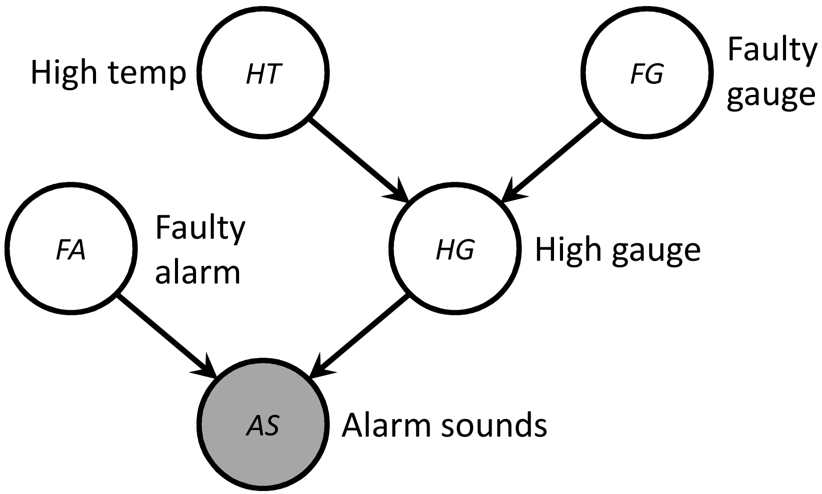


- Joint + Bayes rule + marginalisation → anything

Nuclear power plant

- **Alarm sounds**; meltdown?!
- $\Pr(HT|AS = t) = \frac{\Pr(HT, AS{=}t)}{\Pr(AS{=}t)}$

$$= \frac{\sum_{FG,\,HG,\,FA} \Pr(AS{=}t, FA, HG, FG, HT)}{\sum_{FG,\,HG,\,FA,\,HC} \Pr(AS{=}t, FA, HR, FG, HT)}$$



- Numerator (denominator similar)

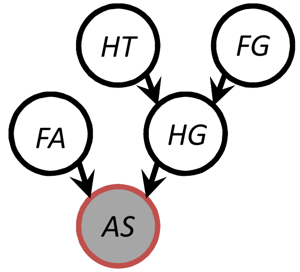
expanding out sums, joint *summing once over 2^5 table*

$$= \sum_{FG} \sum_{HG} \sum_{FA} \Pr(HT) \Pr(HG|HT, FG) \Pr(FG) \Pr(AS = t|FA, HG) \Pr(FA)$$

distributing the sums as far down as possible *summing over several smaller tables*

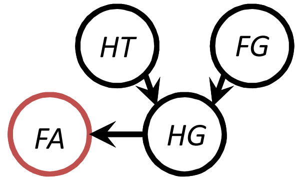
$$= \Pr(HT) \sum_{FG} \Pr(FG) \sum_{HG} \Pr(HG|HT, FG) \sum_{FA} \Pr(FA) \sum_{AS} \Pr(AS = t|FA, HG)$$

Nuclear power plant (cont.)



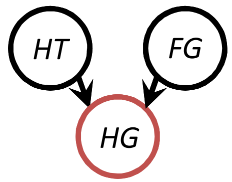
$= \Pr(HT)\sum_{FG}\Pr(FG)\sum_{HG}\Pr(HG|HT,FG)\sum_{FA}\Pr(FA)\sum_{AS}\Pr(AS=t|FA,HG)$

eliminate *AS*: since *AS* observed, really a no-op



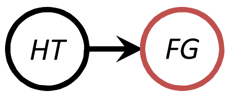
$= \Pr(HT)\sum_{FG}\Pr(FG)\sum_{HG}\Pr(HG|HT,FG)\sum_{FA}\Pr(FA)\, m_{AS}(FA,HG)$

eliminate *FA*: multiplying 1x2 by 2x2



$= \Pr(HT)\sum_{FG}\Pr(FG)\sum_{HG}\Pr(HG|HT,FG)\, m_{FA}(HG)$

eliminate *HG*: multiplying 2x2x2 by 2x1



$= \Pr(HT)\sum_{FG}\Pr(FG)\, m_{HG}(HT,FG)$

eliminate *FG*: multiplying 1x2 by 2x2

$= \Pr(HT)\, m_{FG}(HT)$

Multiplication of tables, followed by summing, is actually matrix multiplication

$m_{FA}(HG)$=

FA	
f	t
0.6	0.4

X

		HG	
		f	t
FA	f	1.0	0
	t	0.8	0.2

Elimination algorithm

Orange background = Slide just for fun!

Eliminate (Graph G, Evidence nodes E, Query nodes Q)

1. Choose node ordering I such that Q appears last
2. Initialise empty list active
3. For each node X_i in G
   a) Append $\Pr(X_i | parents(X_i))$ to active
4. For each node X_i in E
   a) Append $\delta(X_i, x_i)$ to active
5. For each i in I
   a) potentials = Remove tables referencing X_i from active
   b) N_i = nodes other than X_i referenced by tables
   c) Table $\varphi_i(X_i, X_{N_i})$ = product of tables
   d) Table $m_{i(X_{N_i})} = \sum_{X_i} \varphi_i(X_i, X_{N_i})$
   e) Append $m_i(X_{N_i})$ to active
6. Return $\Pr(X_Q | X_E = x_E) = \varphi_Q(X_Q) / \sum_{X_Q} \varphi_Q(X_Q)$

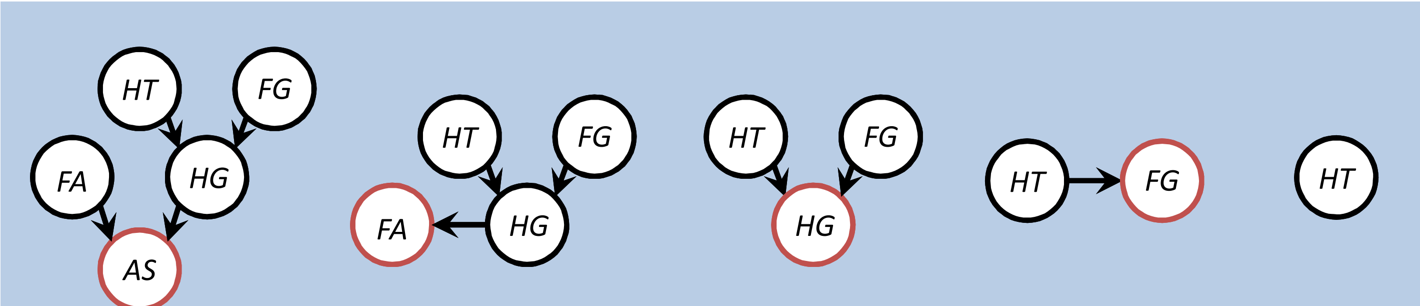
$\left.\begin{array}{l}1.\text{–}3.\end{array}\right\}$ initialise

$\left.\begin{array}{l}4.\end{array}\right\}$ evidence

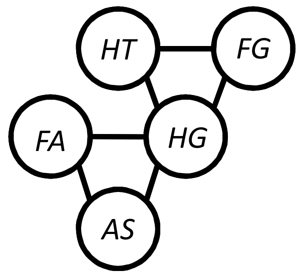
$\left.\begin{array}{l}5.\end{array}\right\}$ marginalise

$\left.\begin{array}{l}6.\end{array}\right\}$ normalise

Runtime of elimination algorithm



PGM after successive eliminations

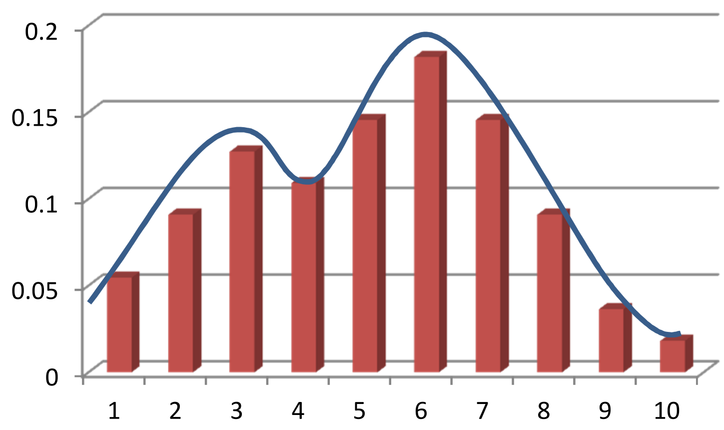


"reconstructed" graph
From process called ***moralisation***

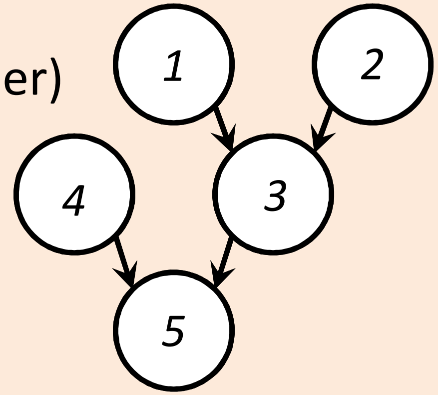
- Each step of elimination
  - Removes a node
  - Connects node's remaining neighbours
    → forms a clique in the "reconstructed" graph
    (cliques are exactly r.v.'s involved in each sum)
- Time complexity exponential in largest clique
- Different elimination orderings produce different cliques
  - Treewidth: minimum over orderings of the largest clique
  - Best possible time complexity is exponential in the treewidth

Probabilistic inference by simulation

- Exact probabilistic inference can be expensive/impossible
- Can we approximate numerically?
- Idea: sampling methods
  - Cheaply sample from desired distribution
  - Approximate **distribution** by **histogram of samples**



Monte Carlo approx probabilistic inference



- Algorithm: sample once from joint
  1. Order nodes' parents before children (topological order)
  2. Repeat
     a) For each node X_i
        i. Index into $\Pr(X_i | parents(X_i))$ with parents' values
        ii. Sample X_i from this distribution
     b) Together $\boldsymbol{X} = (X_1, \ldots, X_d)$ is a sample from the joint
- Algorithm: sampling from $\Pr(X_Q | X_E = x_E)$
  1. Order nodes' parents before children
  2. Initialise set S empty; Repeat
     1. Sample $\boldsymbol{X}$ from joint
     2. If $X_E = x_E$ then add X_Q to S
  3. Return: Histogram of S, normalising counts via divide by $|S|$
- Sampling++: Importance weighting, Gibbs, Metropolis-Hastings

Alternate forms of probabilistic inference

- Elimination algorithm produces single marginal
- Sum-product algorithm on trees
  - 2x cost, supplies all marginals
  - Name: Marginalisation is just sum of product of tables
  - "Identical" variants: Max-product, for MAP estimation
- In general these are message-passing algorithms
  - Can generalise beyond trees (beyond scope): junction tree algorithm, loopy belief propagation
- Variational Bayes: approximation via optimisation

Summary

- Probabilistic inference on PGMs
  - What is it and why do we care?
  - Elimination algorithm; complexity via cliques
  - Monte Carlo approaches as alternate to exact integration