Lecture 17. PGM Probabilistic Inference PGM Statistical Inference

COMP90051 Statistical Machine Learning

Semester 2, 2019
Lecturer: Ben Rubinstein

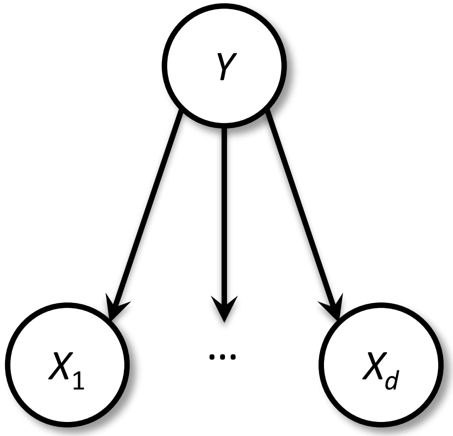
THE UNIVERSITY OF MELBOURNE

Probabilistic inference on PGMs

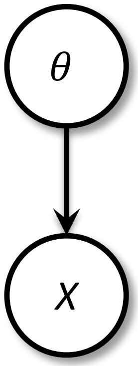
Computing marginal and conditional distributions from the joint of a PGM using Bayes rule and marginalisation.

This deck: how to do it efficiently.

Two familiar examples



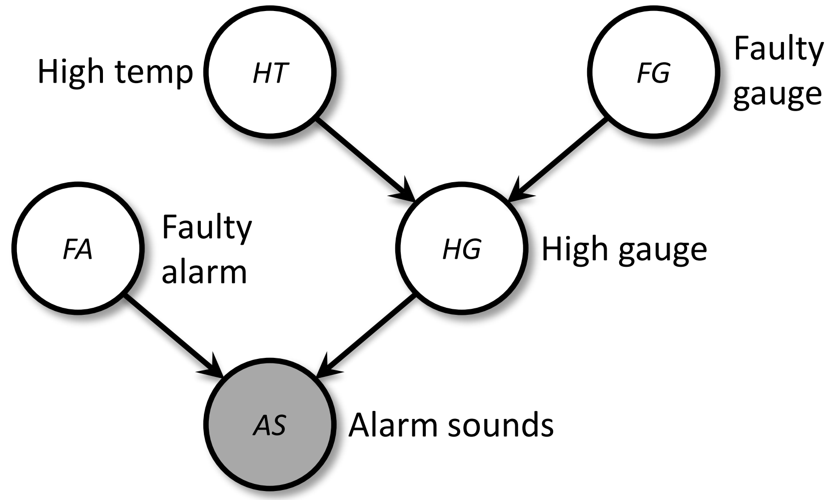
- Naïve Bayes (frequentist/Bayesian)
  - Chooses most likely class given data
  - $\Pr(Y|X_1,\ldots,X_d) = \frac{\Pr(Y,X_1,\ldots,X_d)}{\Pr(X_1,\ldots,X_d)} = \frac{\Pr(Y,X_1,\ldots,X_d)}{\sum_y \Pr(Y=y,X_1,\ldots,X_d)}$



- 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?!
- $\Pr(HT|AS=t) = \frac{\Pr(HT,AS{=}t)}{\Pr(AS{=}t)}$

$$= \frac{\sum_{FG,HG,FA} \Pr(AS{=}t,FA,HG,FG,HT)}{\sum_{FG,HG,FA,HT'} \Pr(AS{=}t,FA,HR,FG,HT')}$$

- Numerator (denominator similar)

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

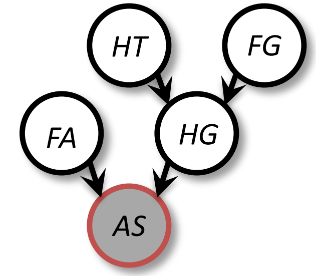
$$= \sum_{FG} \sum_{HG} \sum_{FA} \Pr(HT) \Pr(HG|HT,FG) \Pr(FG) \Pr(AS=t|FA,HG) \Pr(FA)$$

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

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

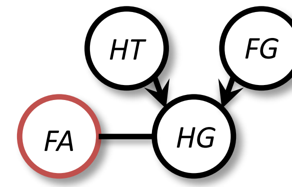
$f(X=a) = \sum_{x} f(X=x)\,\delta(x=a)$

Nuclear power plant (cont.)



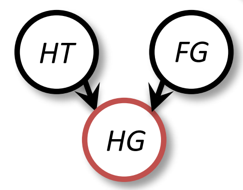
$= \Pr(HT) \sum_{FG} \Pr(FG) \sum_{HG} \Pr(HG|HT,FG) \sum_{FA} \Pr(FA) \Pr(AS = t|FA,HG)$

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



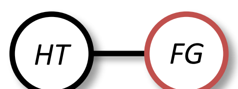
$= \Pr(HT) \sum_{FG} \Pr(FG) \sum_{HG} \Pr(HG|HT,FG) \sum_{FA} \Pr(FA)\, m_{AS}(FA,HG)$

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	*FA* t
0.6	0.4

X

	HG f	*HG* t
FA f	1.0	0
FA t	0.8	0.2

Elimination algorithm

$\{x_i : i \in F\}$

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 $\Pr(X_i \mid parents(X_i))$ to active

initialise

4. For each node X_i in E
   a) Append $\delta(X_i, x_i)$ to active

evidence

red in slide #5

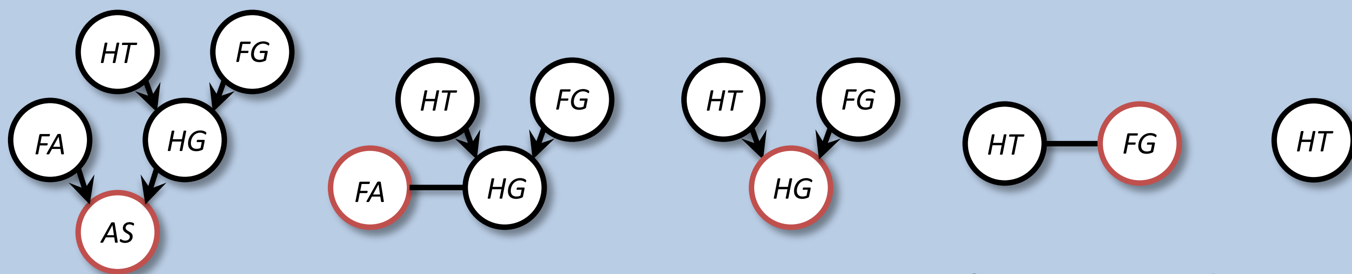
5. For each i in I
   a) potentials = Remove tables referencing X_i from active
   b) N_i = nodes other than X_i referenced by tables in potentials
   c) Table $\phi_i(X_i, X_{N_i})$ = product of tables in potentials
   d) Table $m_{i(X_{N_i})} = \sum_{X_i} \phi_i(X_i, X_{N_i})$
   e) Append $m_i(X_{N_i})$ to active

marginalise

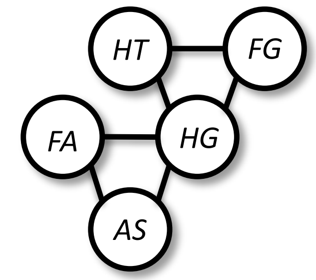
6. Return $\Pr(X_Q | X_E = x_E) = \phi_Q(X_Q) / \sum_{X_Q} \phi_Q(X_Q)$

normalise

Runtime of elimination algorithm



PGM after successive eliminations

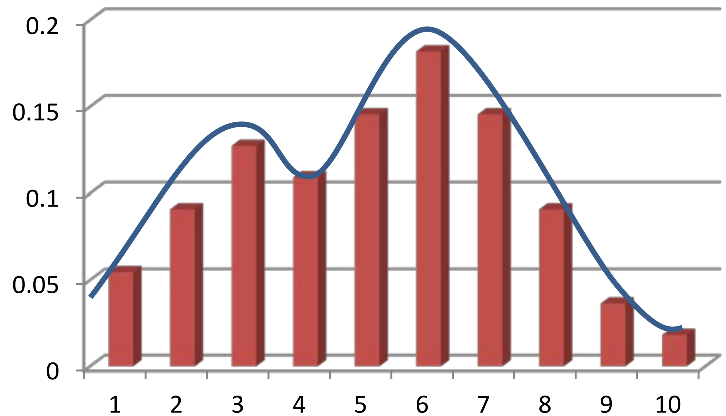


"reconstructed" graph
From process called ***moralisation***

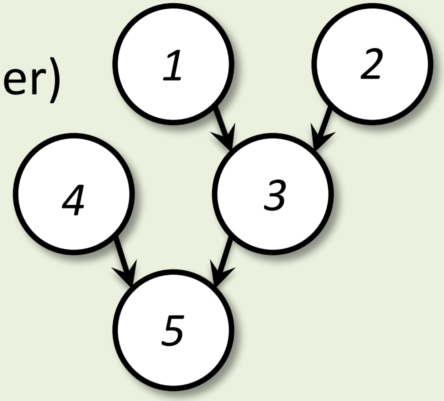
- Each step of elimination
  - Removes a node
  - Connects node's remaining neighbours
    → forms a clique in the "reconstructed" graph
    (cliques are exactly r.v.'s involved in each sum)
- Time complexity exponential in largest clique
- Different elimination orderings produce different cliques
 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

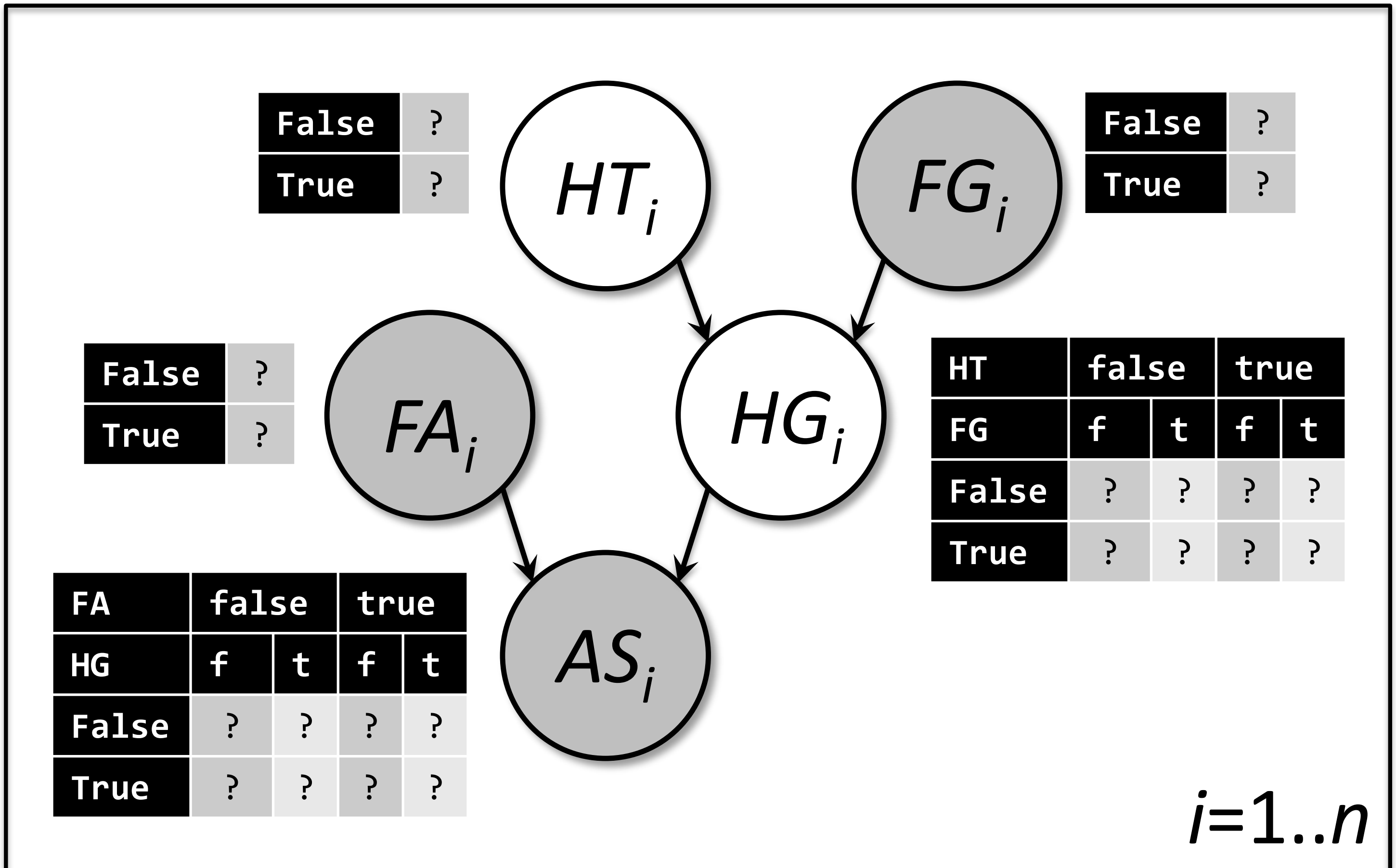
Statistical inference on PGMs

*Learning from data – fitting probability tables to observations (eg as a frequentist; a **Bayesian would just use probabilistic inference** to update prior to posterior)*

Where are we?

- Representation of joint distributions
  * PGMs encode conditional independence
- Independence, d-separation
- Probabilistic inference
  * Computing other distributions from joint
  * Elimination, sampling algorithms
- Statistical inference
  * Learn parameters from data

Have PGM, Some observations, No tables…



False	?
True	?

False	?
True	?

False	?
True	?

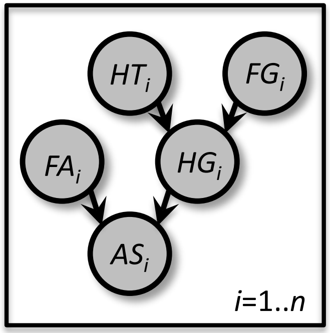
HT	false		true	
FG	f	t	f	t
False	?	?	?	?
True	?	?	?	?

FA	false		true	
HG	f	t	f	t
False	?	?	?	?
True	?	?	?	?

Fully-observed case is "easy"

- Max-Likelihood Estimator (MLE) says
  - If we observe *all* r.v.'s $\boldsymbol{X}$ in a PGM independently n times $\boldsymbol{x}_i$
  - Then maximise the *full* joint

$$\operatorname*{arg\,max}_{\theta\in\Theta} \prod_{i=1}^{n} \prod_{j} p\left(X^j = x_i^j | X^{parents(j)} = x_i{}^{parents(j)}\right)$$



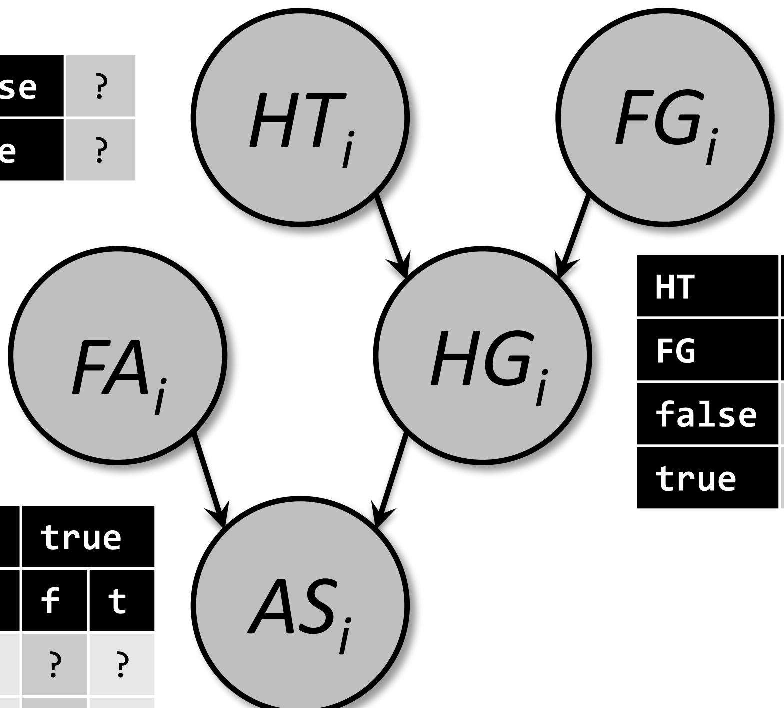
- Decomposes easily, leads to counts-based estimates
  - Maximise log-likelihood instead; becomes sum of logs

$$\operatorname*{arg\,max}_{\theta\in\Theta} \sum_{i=1}^{n} \sum_{j} \log p\left(X^j = x_i^j | X^{parents(j)} = x_i{}^{parents(j)}\right)$$

  - Big maximisation of all parameters together, decouples into small independent problems

- Example is training a naïve Bayes classifier

Example: Fully-observed case



false	?
true	?

false	?
true	?

false	?
true	?

FA	false		true	
HG	f	t	f	t
false	?	?	?	?
true	?	?	?	?

HT	false		true	
FG	f	t	f	t
false	?	?	?	?
true	?	?	?	?

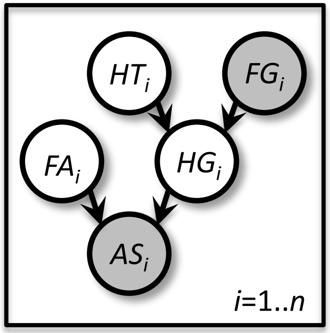
$$\frac{\#\{x_i|FG_i = false\}}{n}$$

$$\frac{\#\{x_i|FG_i = true\}}{n}$$

$$\frac{\#\{x_i|HG_i = true, HT_i = false, FG_i = false\}}{\#\{x_i|HT_i = false, FG_i = false\}}$$

Presence of unobserved variables trickier

- But most PGMs you'll encounter will have latent, or unobserved, variables



- What happens to the MLE?
  - Maximise likelihood of observed data only
  - Marginalise full joint to get to desired "partial" joint
  - $\underset{\theta\in\Theta}{\arg\max}\prod_{i=1}^{n}\sum_{\text{latent }j}\prod_{j}p\left(X^j = x_i{}^j | X^{parents(j)} = x_i{}^{parents(j)}\right)$
  - This won't decouple – oh-no's*!!*

→ Use EM algorithm!

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
- Statistical inference on PGMs
  - What is it and why do we care?
  - Straight MLE for fully-observed data
  - EM algorithm for mixed latent/observed data
- Next time: extra (some more on HMMs, message passing, etc.)