Stat 521A
Lecture 1
Introduction; directed graphical models

Outline

- Administrivia
- Overview
- Local markov property, factorization (3.2)
- Global markov property (3.3)
- Deriving graphs from distributions (3.4)

Administrivia

- Class web page

  www.cs.ubc.ca/~murphyk/Teaching/Stat521A-spring08
- Join groups.google.com/group/stat521a-spring09
- Office hours: Fri 10-11 am
- Final project due Fri Apr 24th
- Weekly homeworks
- Grading
  - Final project: 60%
  - Weekly Assignments: 40%

Auditing

- If you want to 'sit in' on the class, please register for it as 'pass/fail'; you will automatically pass as long as you show up for (most of) the class (no other requirements!)
- If you take it for real credit, you will likely learn more…

Homeworks

Weekly homeworks, out on Tue, due back on Tue

- Collaboration policy:
  - You can collaborate on homeworks if you write the name of your collaborators on what you hand in; however, you must understand everything you write, and be able to do it on your own
- Sickness policy:
  - If you cannot do an assignment, you must come see me in person; a doctor's note (or equivalent) will be required.

Workload

- This class will be quite time consuming.
- Attending lectures: 3h.
- Weekly homeworks: about 3h.
- Weekly reading: about 10h.
- Total: 16h/week.

Pre-requisites

- You should know
  - Basic applied math (calculus, linear algebra)
  - Basic probability/ statistics e.g. what is a covariance matrix, linear/logistic regression, PCA, etc
  - Basic data structures and algorithms (e.g., trees, lists, sorting, dynamic programming, etc)
  - Prior exposure to machine learning (eg CS540) and/or multivariate statistics is strongly recommended

Textbooks

- “Probabilistic graphical models: principles and techniques”, Daphne Koller and Nir Friedman (MIT Press 2009, in press).
- We will endeavour to cover the first 900 (of 1100) pages!
- Copies available at Copiesmart copy center in the village (next to McDonalds) from Thursday
- I may hand out some chapters from Michael Jordan’s draft book, “Probabilistic graphical models”
- I am writing my own book “Machine learning: a probabilistic approach”; I may hand out some chapters from this during the semester.

Matlab

- Matlab is a mathematical scripting language widely used for machine learning (and engineering and numerical computation in general).
- Everyone should have access to Matlab via their CS or Stats account.
- You can buy a student version for $170 from the UBC bookstore. Please make sure it has the Stats toolbox.
- Matt Dunham has written an excellent Matlab tutorial which is on the class web site – please study it carefully!

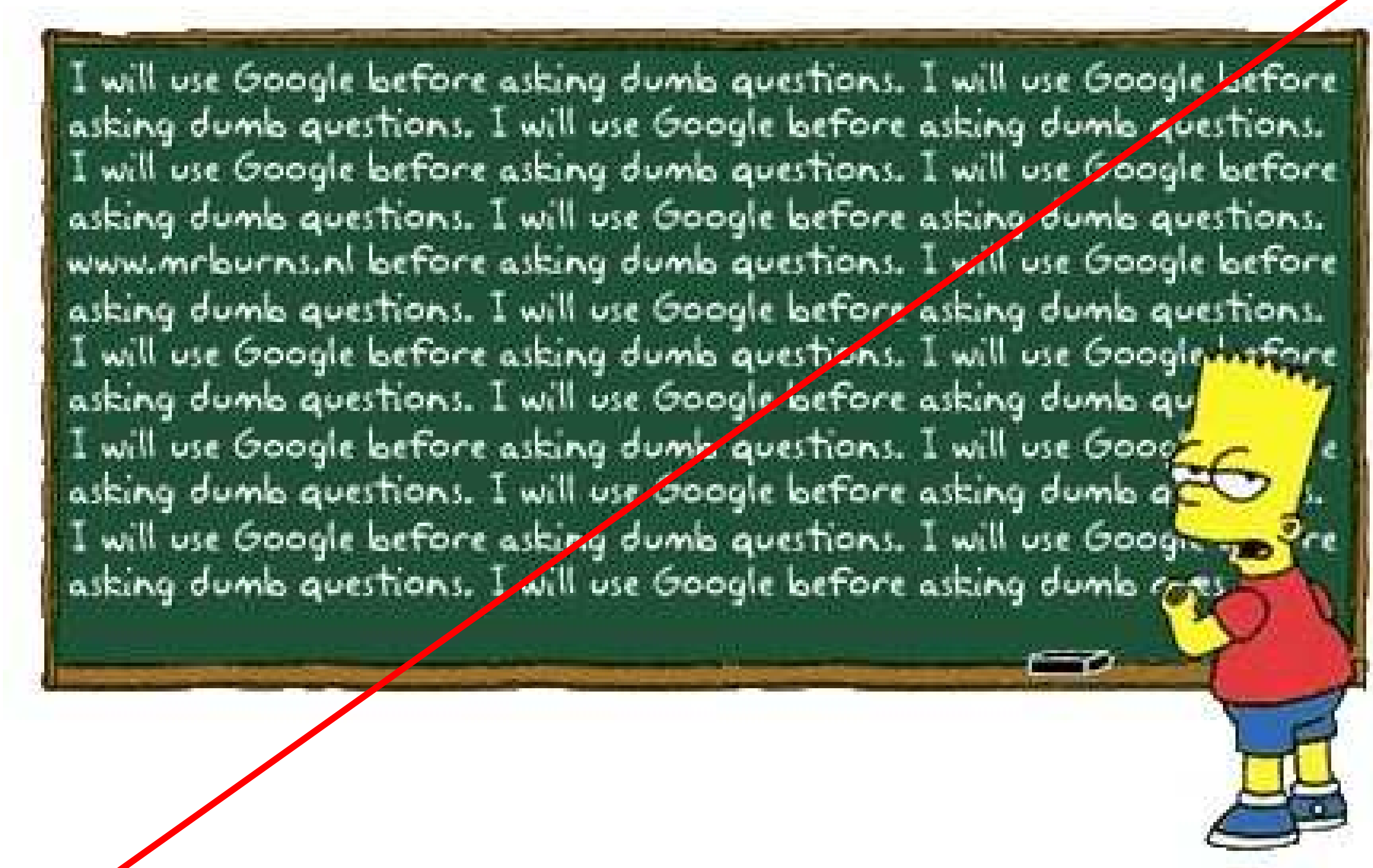
PMTK

- Probabilistic Modeling Toolkit is a Matlab package I am currently developing to go along with my book.
- It uses the latest object oriented features of Matlab 2008a and will not run on older versions.
- It is designed to replace my earlier ‘Bayes net toolbox’.
- PMTK will form the basis of some of the homeworks, and may also be useful for projects. (Currently support for GMs is very limited.)
- http://www.cs.ubc.ca/~murphyk/pmtk/

Learning objectives

- By the end of this class, you should be able to
  - Understand basic principles and techniques of probabilistic graphical models
  - Create suitable models for any given problem
  - Derive the algorithm (equations, data structures etc) needed to apply the model to data
  - Implement the algorithm in reasonably efficient Matlab
  - Demonstrate your skills by doing a reasonably challenging project

Ask questions early and often!

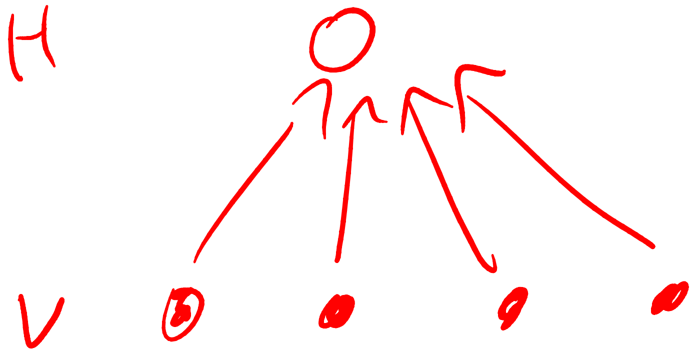


Outline

- Administrivia
- Overview
- Local markov property, factorization (3.2)
- Global markov property (3.3)
- Deriving graphs from distributions (3.4)

Supervised learning

- Predict output given inputs, ie compute p(h|v)
- Regression: h in R
- Classification: h in {1,…,C}



Structured output learning

- Model *joint* density of p(h,v) (or maybe p(h|v))
- Then infer p(h|v) - state estimation
- MAP estimation (posterior mode)

$$\mathbf{h}^* = \arg\max_{h_1}, \ldots, \arg\max_{h_n} p(\mathbf{h}|\mathbf{v}, \boldsymbol{\theta})$$

- Posterior marginals

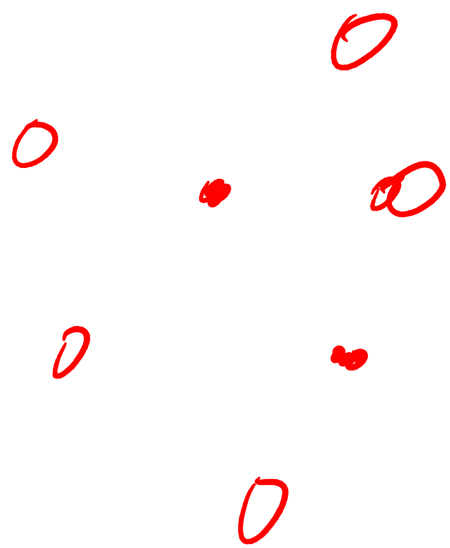
$$h_1^* = \sum_{h_2} \ldots, \sum_{h_n} p(\mathbf{h}|\mathbf{v}, \boldsymbol{\theta})$$

- Also need to estimate parameters and structure

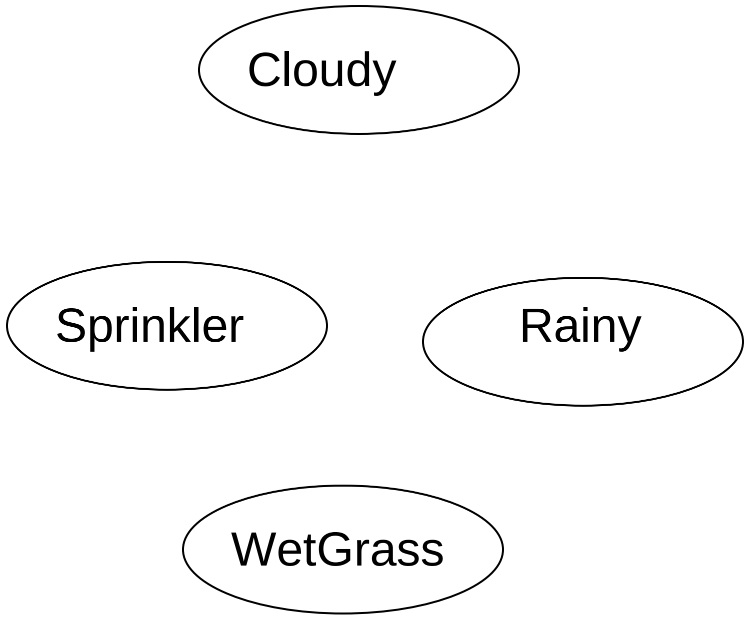
	supervised		unsupervised	
H	Labels	Diseases	Genotype	Low-dim rep
V	Pixels	Symptoms	Phenotype	Features

Density estimation

- Model joint density of all variables
- No distinction between inputs and outputs: different subsets of variables can be observed at different times (eg for missing data imputation)
- Can run model in any ‘direction’



Water sprinkler joint distribution



$$p(C, S, R, W)$$

c	s	r	w	prob
0	0	0	0	0.200
0	0	0	1	0.000
0	0	1	0	0.005
0	0	1	1	0.045
0	1	0	0	0.020
0	1	0	1	0.180
0	1	1	0	0.001
0	1	1	1	0.050
1	0	0	0	0.090
1	0	0	1	0.000
1	0	1	0	0.036
1	0	1	1	0.324
1	1	0	0	0.001
1	1	0	1	0.009
1	1	1	0	0.000
1	1	1	1	0.040

Inference

- Prior that sprinkler is on

$$p(S=1) = \sum_{c=0}^{1}\sum_{r=0}^{1}\sum_{w=0}^{1} p(C=c, S=1, R=r, W=w) = 0.3$$

- Posterior that sprinkler is on given that grass is wet

$$p(S=1|W=1) = \frac{p(S=1, W=1)}{p(W=1)} = 0.43$$

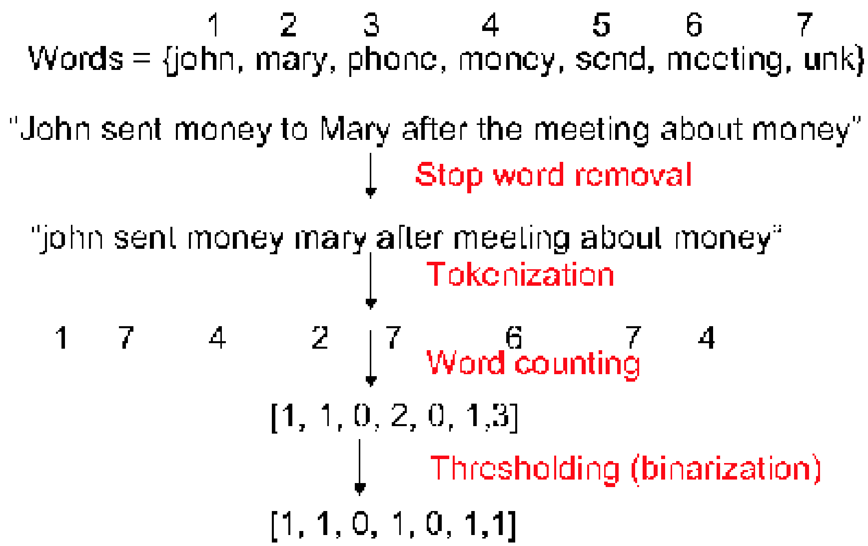
- Posterior that sprinkler is on given that grass is wet and it is raining

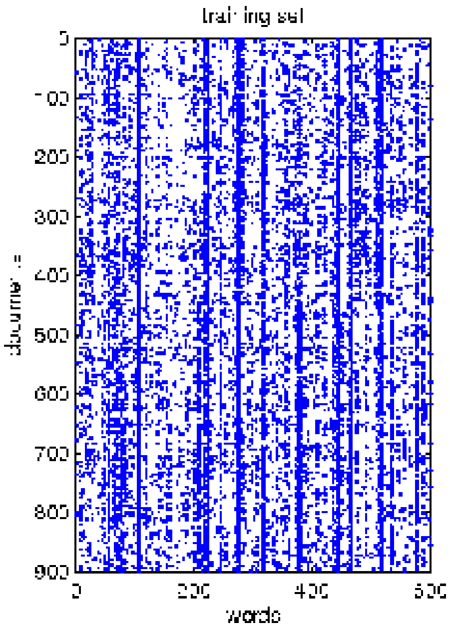
$$p(S=1|W=1, R=1) = \frac{p(S=1, W=1, R=1)}{p(W=1, R=1)} = 0.19$$

Explaining away

Bag of words model

- bag-of-words representation of text documents
- Xi=1 iff word i occurs in document
- Define a joint distribution over bit vectors, p(x1,…,xn)





Inference

- Given word Xi occurs, which other words are likely to co-occur?
- What is the probability of any particular bit vector?
- Sample (generate) documents from joint p(x)

Bayesian classifiers

- Define joint p(y,x) = p(x|y) p(y) on document class label and bit vectors
- Can infer class label using Bayes rule

Class posterior

Class-conditional density

Class prior

$$p(y=c|x) = \frac{p(x|y=c)p(y=c)}{\sum_{c'} p(x|y=c')p(y=c')}$$

Normalization constant

- If y is hidden, we can use this to cluster documents.
- In both cases, we need to define p(x|y=c)

Naïve Bayes assumption

- The simplest approach is to assume each feature is conditionally independent given the class/cluster Y

$$X_i \perp X_j | Y = c$$

- In this case, we can write

$$p(\mathbf{x}|y = c) = \prod_{j=1}^{d} p(x_j|y = c)$$

- The number of parameters is reduced from $O(C\ K^d)$ to $O(C\ K)$, assuming C classes and K-ary features

Conditional independence

- In general, making CI assumptions is one of the most useful tools in representing joint probability distributions in terms of low-dimensional quantities, which are easier to estimate from data
- **Graphical models are a way to represent CI assumptions using graphs**
- The graphs provide an intuitive representation, and enable the derivation of efficient algorithms

Graphical models

- There are many kinds of graphical models
- Directed Acyclic graphs – “Bayesian networks”
- Undirected graphs – “Markov networks”
- Directed cyclic graphs – “dependency networks”
- Partially directed acyclic graphs (PDAGs) – “chain graphs”
- Factor graphs
- Mixed ancestral graphs
- Etc
- Today we will focus on DAG models

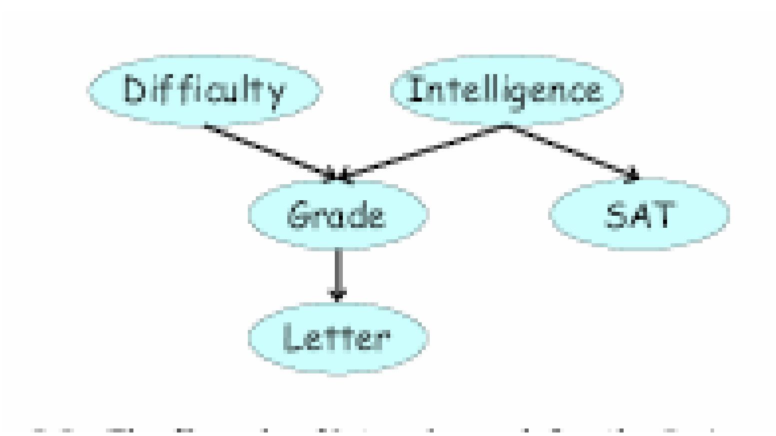
Outline

- Administrivia
- Overview
- Local markov property, factorization (3.2)
- Global markov property (3.3)
- Deriving graphs from distributions (3.4)

CI properties of DAGs

- Defn 3.2.1. A BN *structure* G is a DAG whose nodes represent rvs $X_1,\ldots,X_n$. Let $Pa(X_i)$ be the parents of X_i, and $Nd(X_i)$ be the non-descendants of Xi. Then G encodes the following local Markov assumptions:

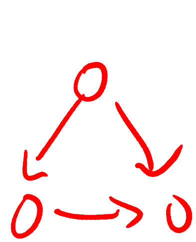
$$I_\ell(G) = \{X_i \perp Nd(X_i) | Pa(X_i)\}$$

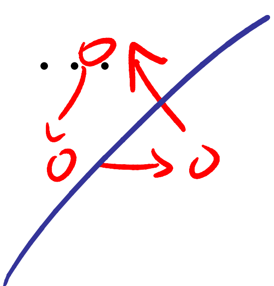


Student network

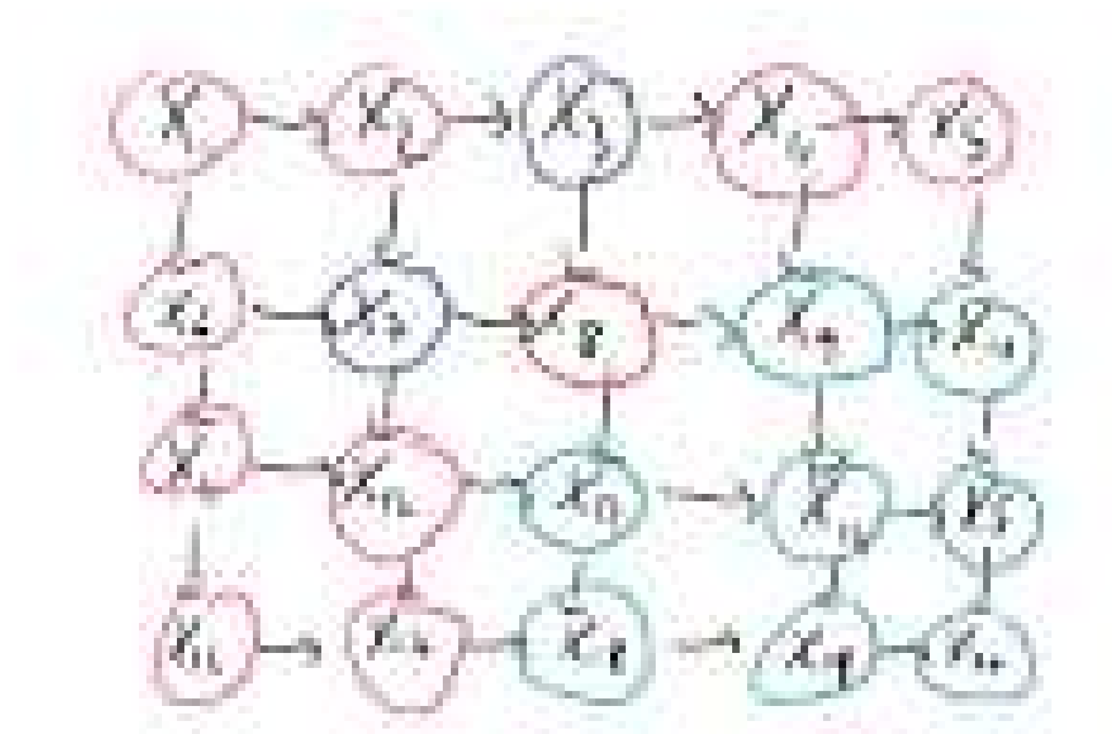
$$G \perp S | D, I$$

$$I \perp D$$





Another Example



Red (X8) ⊥ pink | bue

I-maps

- Def 3.2.2. Let I(P) be the set of independence assertions of the form X $\perp$ Y | Z that hold in P

$$P \models X \perp Y | Z$$

- Def 3.2.3. We say G is an I-map for set I if I(G) $\subseteq$ I

  (hence the graph does not make any false independence assumptions)

I-maps: examples

- Examples 3.2.4, 3.2.5

X	Y	$P(X,Y)$
x^0	y^0	0.08
x^0	y^1	0.32
x^1	y^0	0.12
x^1	y^1	0.48

X	Y	$P(X,Y)$
x^0	y^0	0.4
x^0	y^1	0.3
x^1	y^0	0.2
x^1	y^1	0.1

$$\begin{array}{cc} & \quad Y \\ & \;\; 0 \qquad\quad 1 \\ X \begin{array}{c} 0 \\ 1 \end{array} & \begin{pmatrix} 0.08 & 0.32 \\ 0.12 & 0.43 \end{pmatrix} \end{array} = \overset{X}{\begin{pmatrix} 0.4 \\ 0.6 \end{pmatrix}} \overset{Y}{\begin{pmatrix} 0.2 & 0.8 \end{pmatrix}}$$

$P(X,Y) = P(X)\,P(Y)$

$P \models X \perp Y$

$P \not\models X \perp Y$

Imaps = X Y, X -> Y, X <- Y

Imaps = X -> Y, X <- Y

I-map to factorization

- Def 3.2.5. A distribution P factorizes over a DAG G if it can be written in the form

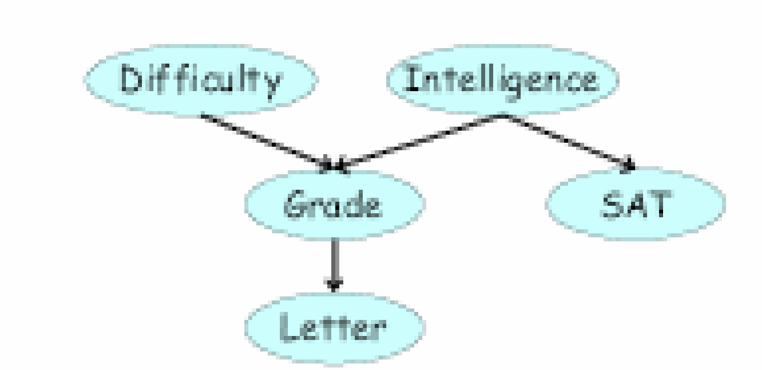
$$p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i | Pa(X_i))$$

- Thm 3.2.7. If G is an I-map for P, then P factorizes according to G.
- Proof: by the chain rule, we can always write

$$p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i | X_{1:i-1})$$

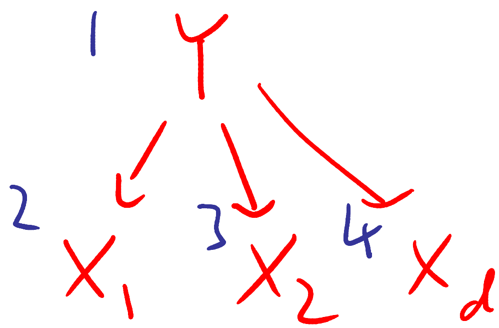
- By the local markov assumption, we can drop all the ancestors except the parents. QED.

Student network



$$
\begin{aligned}
p(I, D, G, S, L) &= \\
&\quad p(I)p(D|\cancel{I})p(G|I, D)p(S|I, \cancel{D}, \cancel{G})p(L|\cancel{I}, \cancel{D}, \cancel{G}, S) \\
&= \quad p(I)p(D|)p(G|I, D)p(S|I)p(L|S)
\end{aligned}
$$

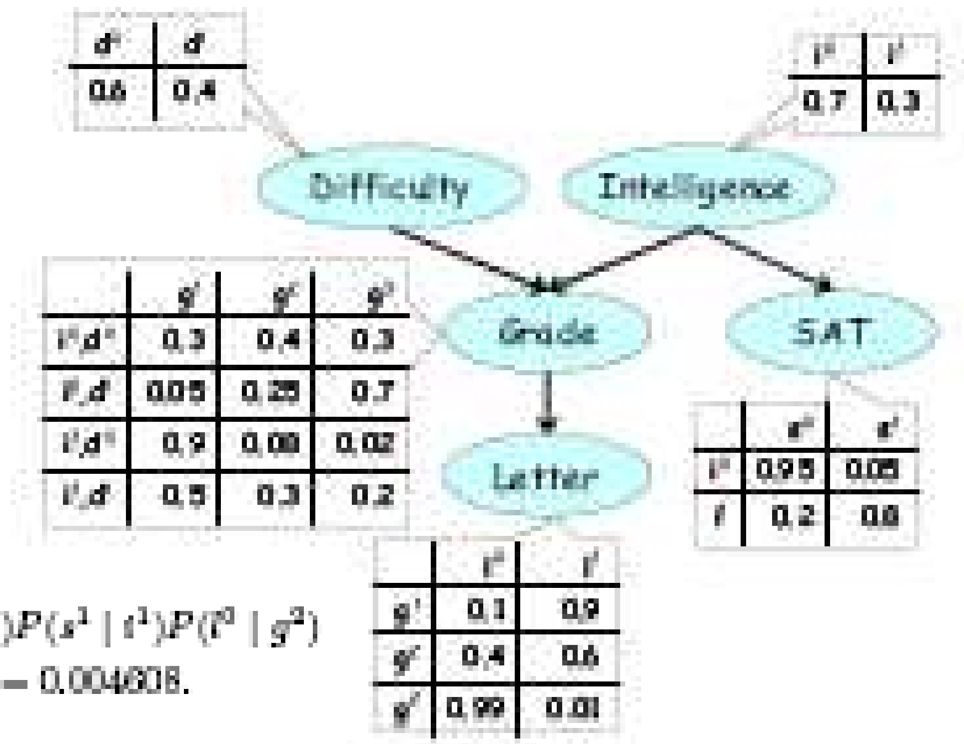
Naïve Bayes classifier



$$p(y, \mathbf{x}) = p(y) \prod_{j=1}^{d} p(x_j | y)$$

Bayes net = DAG + CPD

- A DAG defines a family of distributions, namely all those that factorize in the specified way.
- Def 3.2.6. A Bayes net is a DAG G together with a set of local Conditional Probability Distributions p(X_i|Pa(X_i)).

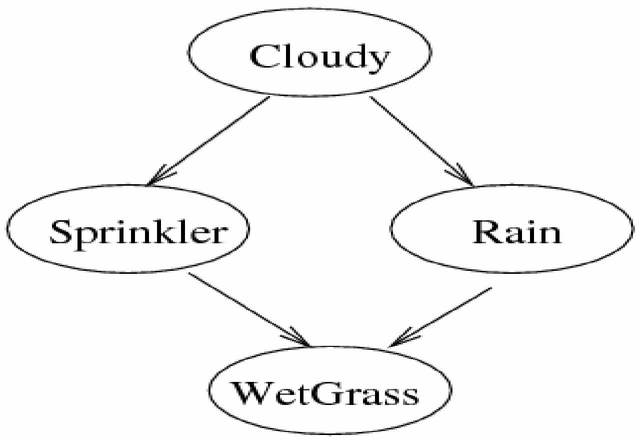


CPTs:
Each row is a different multinomial distribution,
One per parent combination

$$P(i^1, d^0, g^2, s^1, l^0) = P(i^1)P(d^0)P(g^2 \mid i^1, d^0)P(s^1 \mid i^1)P(l^0 \mid g^2)$$
$$= 0.3 \cdot 0.6 \cdot 0.08 \cdot 0.8 \cdot 0.4 = 0.004608.$$

Water sprinkler BN

P(C=F)	P(C=T)
0.5	0.5



C	P(S=F)	P(S=T)
F	0.5	0.5
T	0.9	0.1

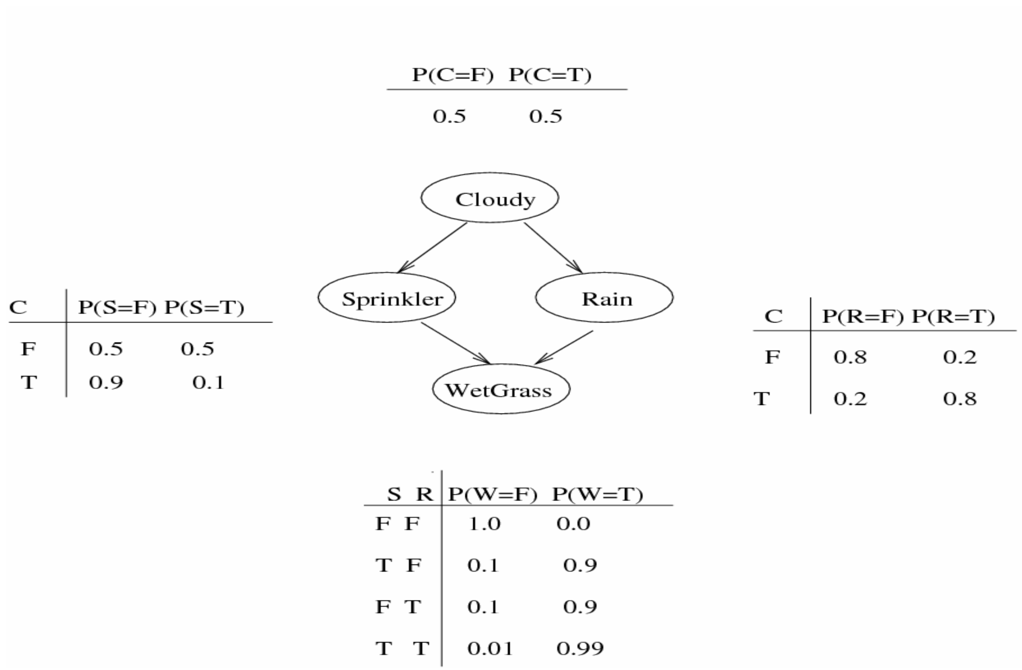
C	P(R=F)	P(R=T)
F	0.8	0.2
T	0.2	0.8

S	R	P(W=F)	P(W=T)
F	F	1.0	0.0
T	F	0.1	0.9
F	T	0.1	0.9
T	T	0.01	0.99

$$p(C,S,R,W) = p(C)p(S|C)p(R|C)p(W|S,R)$$

Joint distribution for sprinkler network

$$p(C, S, R, W) = p(C)p(S|C)p(R|C)p(W|S, R)$$



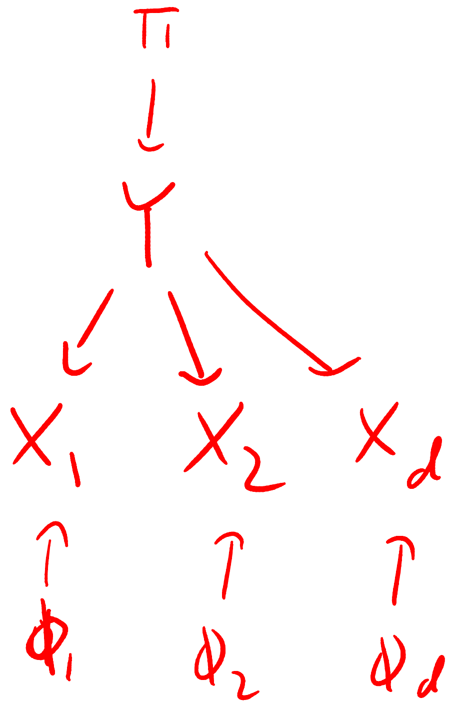
c	s	r	w	prob
0	0	0	0	0.200
0	0	0	1	0.000
0	0	1	0	0.005
0	0	1	1	0.045
0	1	0	0	0.020
0	1	0	1	0.180
0	1	1	0	0.001
0	1	1	1	0.050
1	0	0	0	0.090
1	0	0	1	0.000
1	0	1	0	0.036
1	0	1	1	0.324
1	1	0	0	0.001
1	1	0	1	0.009
1	1	1	0	0.000
1	1	1	1	0.040

CPDs

- CPDs can be any conditional distribution p(X_i|Pa(X_i))
- If Xi has no parents, this is an unconditional distribution
- For discrete variables, it is common to use tables (conditional multinomials)
- However, CPTs have O($K^{|pa|}$) parameters; we will consider more parsimonious representations (such as logistic regression) – see ch 5
- For continuous variables, it is common to use linear regression to define CPDs (see ch 7)

$$p(X_i|Pa(X_i) = \mathbf{u}, \boldsymbol{\theta}_i) = \mathcal{N}(X_i|\mathbf{u}^T\boldsymbol{\theta}_i, \sigma_i^2)$$

Representing parameters as nodes



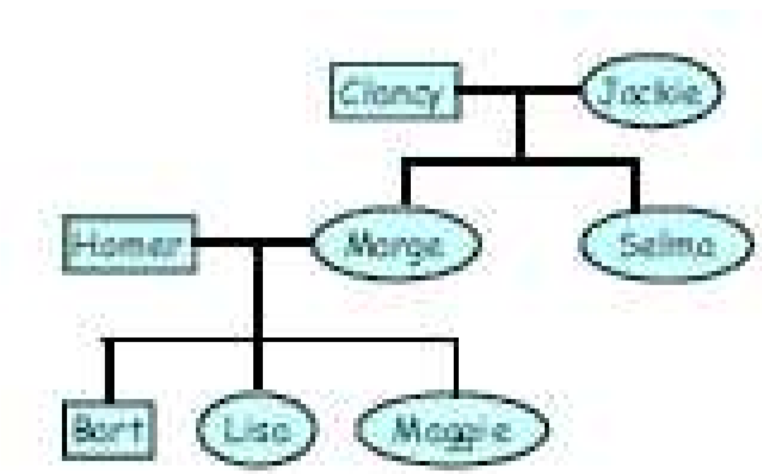
$$p(y, \mathbf{x}, \boldsymbol{\theta}) = p(y|\boldsymbol{\pi})p(\boldsymbol{\pi}) \prod_{j=1}^{d} p(x_j|y, \boldsymbol{\phi}_j)p(\boldsymbol{\phi}_j)$$

We will return to this representation when we discuss parameter estimation

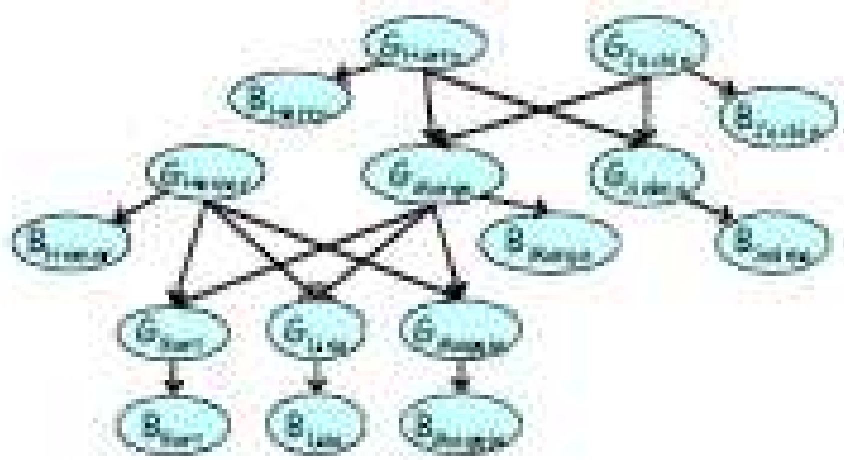
DAGs are widely used for Hierarchical Bayesian models

Genetic inheritance

- G(x) = genotype (allele) of person x at given locus, say {A,B,O} x {A,B,O}
- B(x) = phenotype (blood group) in {A,B,O}
- P(B(c)|G(c)) = penetrance model
- P(G(c)|G(p),G(m)) = transmission model
- P(G(c)) = priors for founder nodes



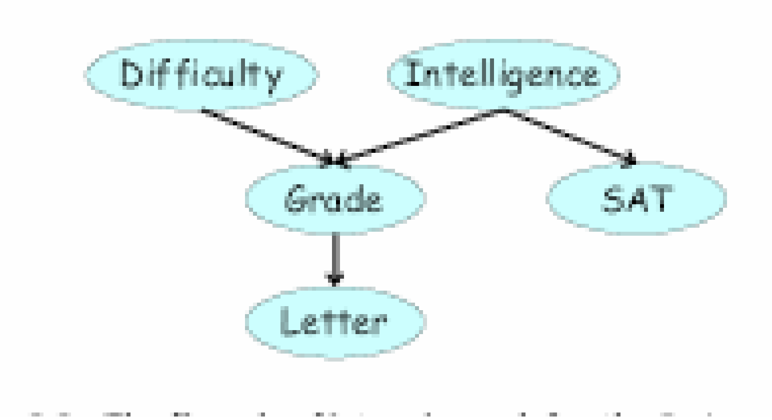
(a)



(b)

Factorization to I-map

- Thm 3.2.9. If P factorizes over G, then G is an I-map for P.
- Proof (by example)
- We need to show all the local Markov properties hold in P eg. RTP



$$p(S|I,D,G,L) = p(S|I)$$

- By factorization and elementary probability,

$$\begin{aligned} p(S|I,D,G,L) &= \frac{p(S,I,D,G,L)}{p(I,D,G,L)} \\ &= \frac{p(I)p(D)p(G|I,D)p(L|G)p(S|I)}{p(I)p(D)p(G|I,D)p(L|G)} = p(S|I) \end{aligned}$$

Outline

- Administrivia
- Overview
- Local markov property, factorization (3.2)
- Global markov property (3.3)
- Deriving graphs from distributions (3.4)

Global Markov properties

- The DAG defines local markov properties

$$I_\ell(G) = \{X_i \perp Nd(X_i) | Pa(X_i)\}$$

- We would like to be able to determine global markov properties, i.e., statements of the form

$$I(G) = \{X \perp Y | Z : f(X, Y, Z, G)\}$$

for some function f.

- There are several equivalent ways to define f:
- Bayes ball
- d-separation
- Ancestral separation (ch 4)

Chains

- Consider the chain

$X \to Y \to Z$

$$p(x,y,z) = p(x)p(y|x)p(z|y)$$

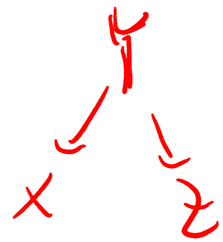
- If we condition and y, x and z are independent

$$\begin{aligned} p(x,z|y) &= \frac{p(x)p(y|x)p(z|y)}{p(y)} \\ &= \frac{p(x,y)p(z|y)}{p(y)} \\ &= p(x|y)p(z|y) \end{aligned}$$

$X \to Y \to Z$

Common cause

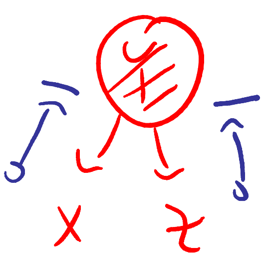
- Consider the “tent”



$$p(x, y, z) = p(y)p(x|y)p(z|y)$$

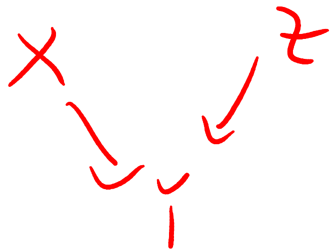
- Conditioning on Y makes X and Z independent

$$
\begin{aligned}
p(x, z|y) &= \frac{p(x, y, z)}{p(y)} \\
&= \frac{p(y)p(x|y)p(z|y)}{p(y)} = p(x|y)p(z|y)
\end{aligned}
$$



V-structure (common effect)

- Consider the v-structure



$$p(x, y, z) = p(x)p(z)p(y|x, z)$$

- X and Z are unconditionally independent

$$p(x, z) = \sum_y p(x, y, z) = \sum_y p(x)p(z)p(y|x, z) = p(x)p(z)$$

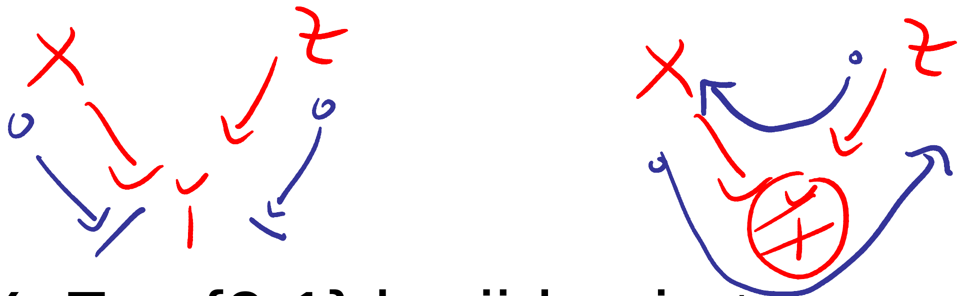
but are conditionally dependent

$$p(x, z|y) = \frac{p(x)p(z)p(y|x, z)}{p(y)} \neq f(x)g(z)$$

Explaining away

- Consider the v-structure

$X \perp Z$



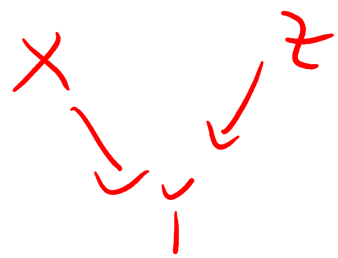
$X \not\perp Z \mid Y$

- Let X, Z ∈ {0,1} be iid coin tosses.
- Let Y = X + Z.
- If we observe Y, X and Z are coupled.

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	2

Explaining away

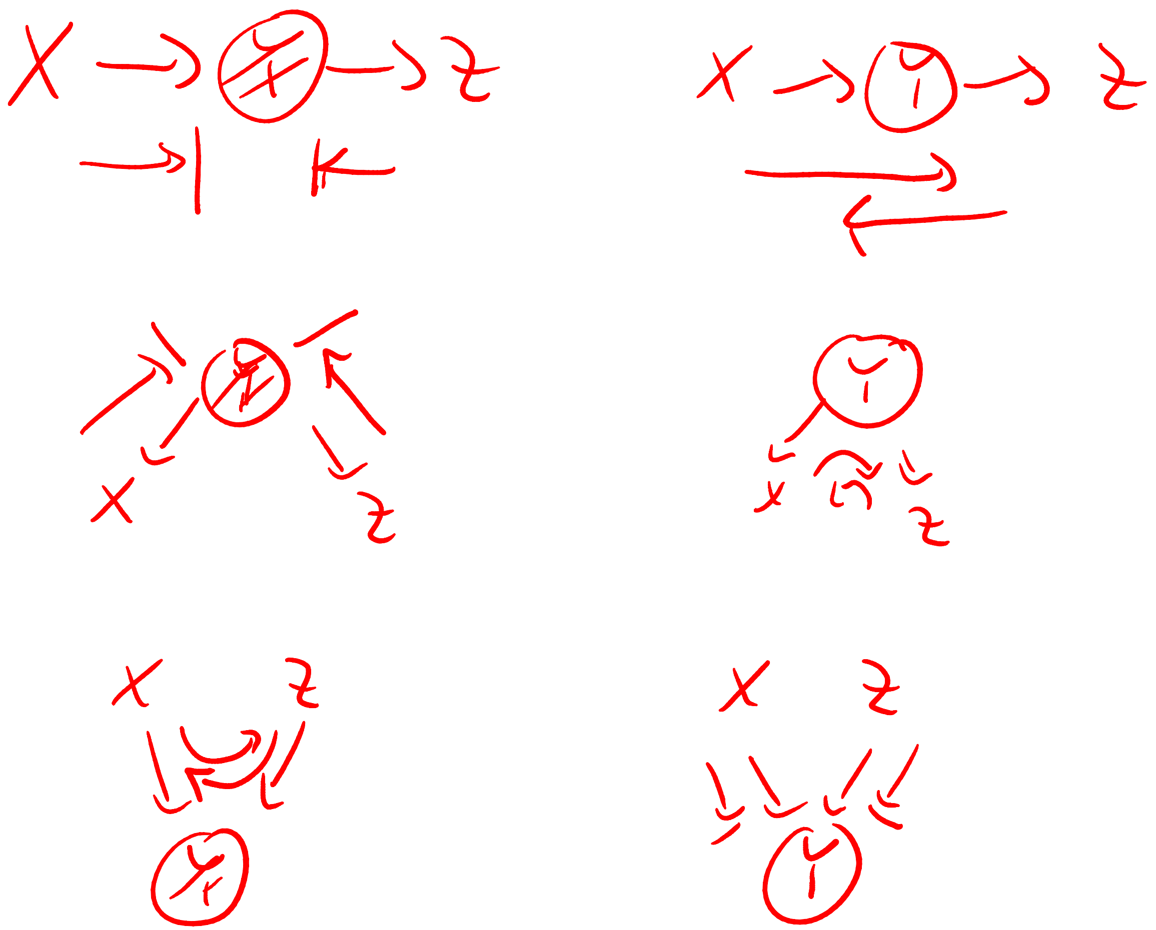
- Let Y = 1 iff burglar alarm goes off,
- X=1 iff burglar breaks in
- Z=1 iff earthquake occurred



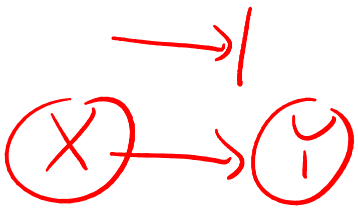
- X and Z compete to explain Y, and hence become dependent
- Intuitively, $p(X=1|Y=1) > p(X=1|Y=1,Z=1)$

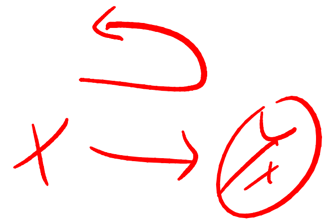
Bayes Ball Algorithm

- $X_A \perp X_B \mid X_C$ if we cannot get a ball from any node in A to any node in B when we shade the variables in C. Balls can get blocked as follows.

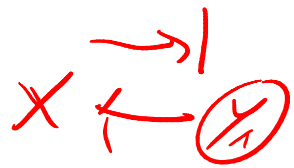


Boundary conditions (source X = destn Z)



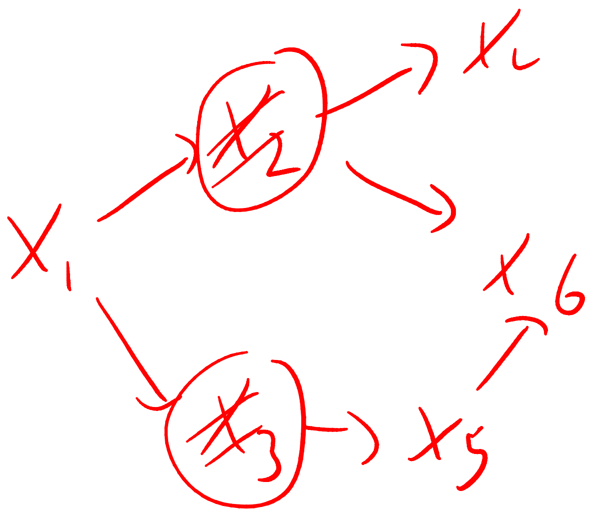


V-structure
First X->Y then Y <- Z



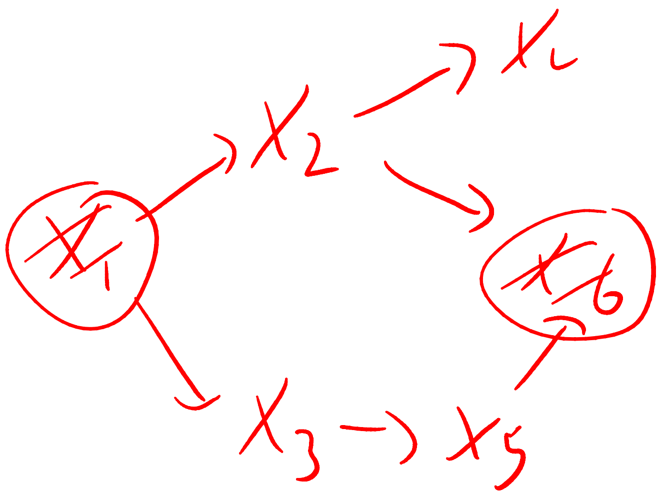
Tent
First X <- Y then Y -> Z

Example



$X_1 \perp X_6 \mid X_2, X_3$?

Example



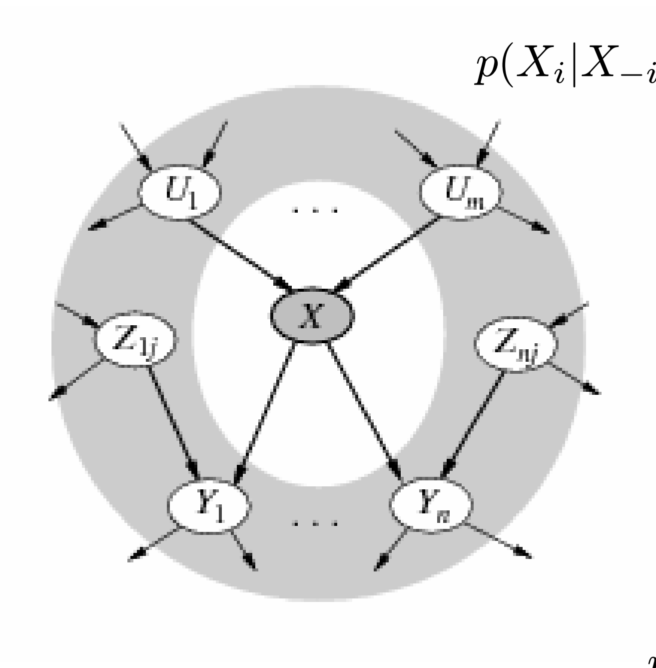
$X_2 \perp X_3 \mid X_1, X_6$?

Markov blankets for DAGs

- The Markov blanket of a node is the set that renders it independent of the rest of the graph.

$$MB(X) = \text{minimal set } U s.t. X \perp \mathcal{X} \setminus \{X\} \setminus U | U$$

- This is the parents, children and co-parents.

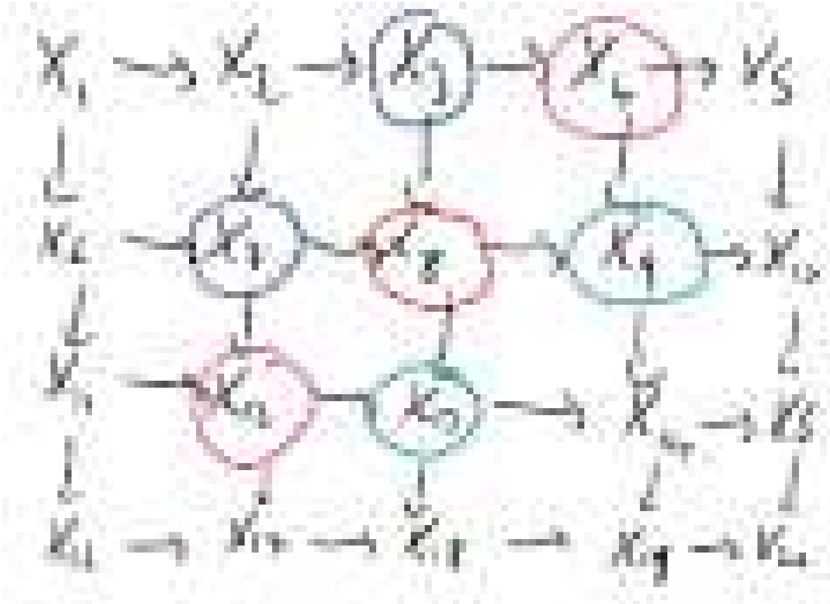


$$
\begin{aligned}
p(X_i|X_{-i}) &= \frac{p(X_i, X_{-i})}{\sum_x p(X_i, X_{-i})} \\
&= \frac{p(X_i, U_{1:n}, Y_{1:m}, Z_{1:m}, R)}{\sum_x p(x, U_{1:n}, Y_{1:m}, Z_{1:m}, R)} \\
&= \frac{p(X_i|U_{1:n})[\prod_j p(Y_j|X_i, Z_j)]P(U_{1:n}, Z_{1:m}, R)}{\sum_x p(X_i = x|U_{1:n})[\prod_j p(Y_j|X_i = x, Z_j)]P(U_{1:n}, Z_{1:m}, R)} \\
&= \frac{p(X_i|U_{1:n})[\prod_j p(Y_j|X_i, Z_j)]}{\sum_x p(X_i = x|U_{1:n})[\prod_j p(Y_j|X_i = x, Z_j)]}
\end{aligned}
$$

$$p(X_i|X_{-i}) \propto p(X_i|Pa(X_i)) \prod_{Y_j \in ch(X_i)} p(Y_j|Pa(Y_j))$$

Useful for Gibbs sampling

Another example



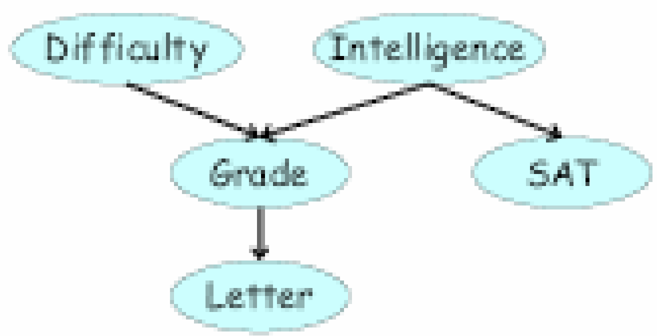
Red node (X8) indep of rest (black) given MB (blue parents, green children, pink co-parents)

Active trails

- Whenever influence can flow from to Y via Z, we say that the trail X <-> Y <-> Z is active.
- Causal trail: X -> Z -> Y. Active iff Z not obs.
- Evidential trail: X <- Z <- Y. Active iff Z not obs
- Common cause: X <- Z -> Y. Active iff Z not obs
- Common effect; X -> Z <- Y. Active iff either Z or one of its descendants is observed.
- Def 3.3.1. Let G be a BN structure, and X1 <-> … <-> Xn be a trail in G. Let E be a subset of nodes. The trail is active given E if
- Whenever we have a v-structure X_{i-1} -> X_i <- X_{i+1}, then X_i or one of its desc is in E
- No other node along the trail is in E

Example

- D-> G <- I ->S not active for E={}
- D-> G <- I ->S is active for E={L}
- D-> G <- I ->S not active for E={L,I}
- Non-monotonic



d-separation

- Def 3.3.2, We say X and Y are d-separated given Z, denoted d-sep_G(X;Y|Z), if there is no active trail between any node in X to any node in Y, given Z. The set of such independencies is denoted

$$I(G) = \{X \perp Y | Z : \text{dsep}_G(X;Y|Z)\}$$

- Thm 3.3.3. (Soundness of dsep). If P factorizes according to G, then I(G) $\subseteq$ I(P).

- False thm (completeness of dsep). For any P that factorizes according to G, if X $\perp$ Y | Z in I(P), then desp_G(X;Y|Z) (i.e., P is faithful to G)

Faithfulness

- Def 3.3.4. A distribution P is faithful to G if, whenever X ⊥ Y | Z in I(P), we have dsep_G(X;Y|Z) i.e., there are no “non-graphical” independencies buried in the parameters
- A simple unfaithful distribution, with Imap A->B:

	b^0	b^1
a^0	0.4	0.6
a^1	0.4	0.6

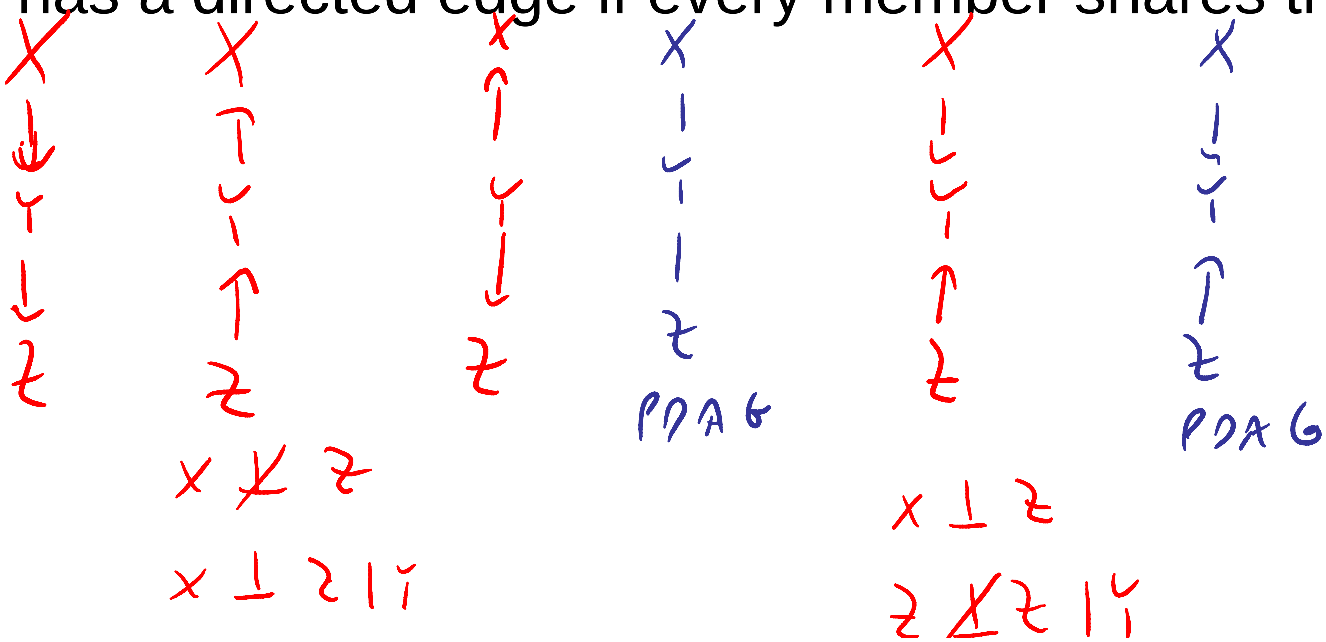
-

Such distributions are “rare”

- Thm 3.3.7. For almost all distributions P that factorize over G (ie except for a set of measure zero in the space of CPD parameterizations), we have that I(P)=I(G)

Markov equivalence

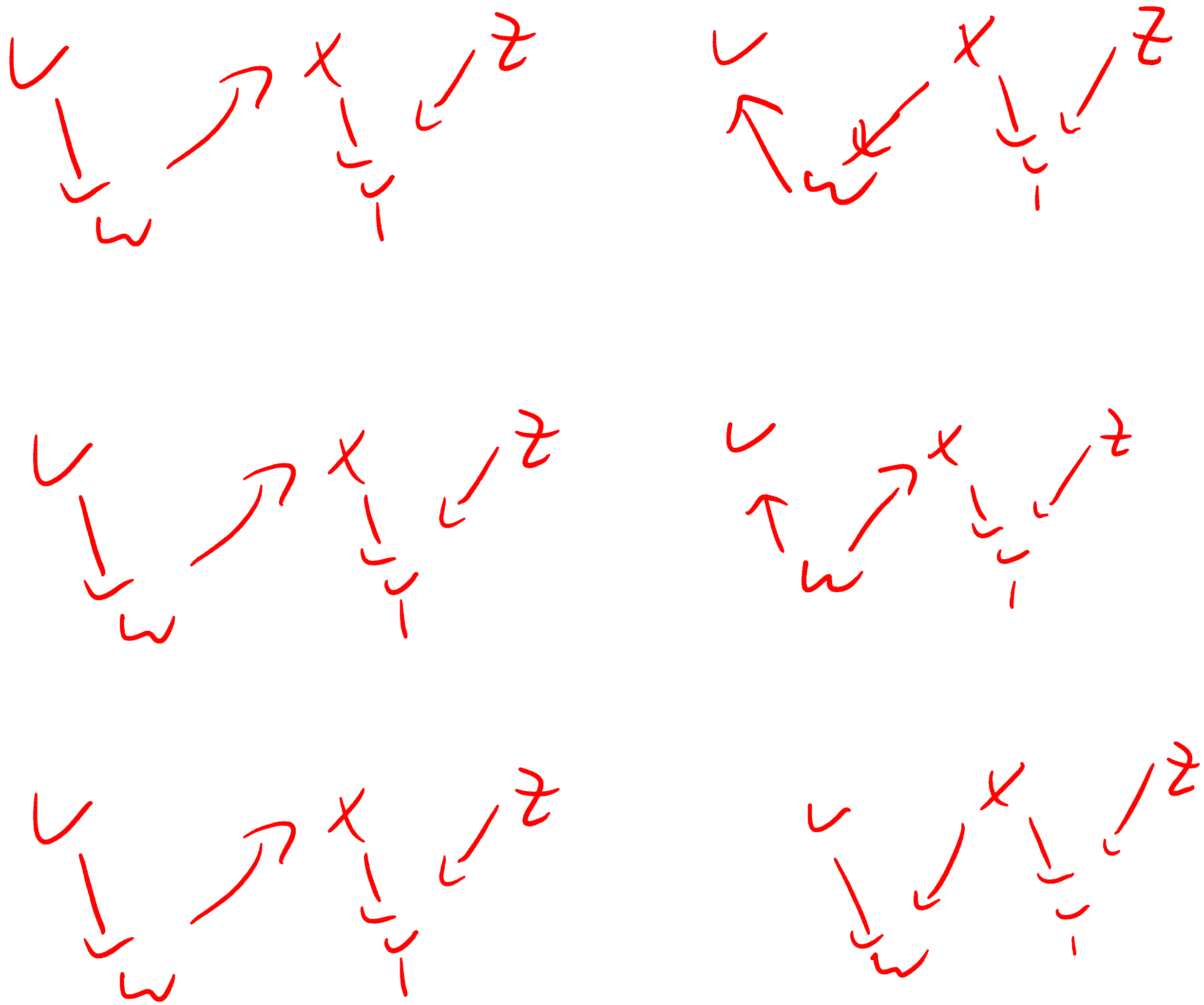
- A DAG defines a set of distributions. Different DAGs may encode the same set and hence are indistinguishable given observational data.
- Def 3.3.10. DAGs G1 and G2 are I-equivalent if I(G1)=I(G2). The set of all DAGs can be partitioned into I-equivalence classes.
- Def 3.4.11. Each can be represented by a class PDAG: only has a directed edge if every member shares that edge.



Identifying I-equivalence

- Def 3.3.11. The skeleton of a DAG is an undirected graph obtained by dropping the arrows.
- Thm 3.3.12. If G1 and G2 have the same skeleton and the same v-structures, they are I-equivalent.
- However, there are structures that are I-equiv but do not have same v-structures (eg fully connected DAG).
- Def 3.3.13. A v-structure X->Z<-Y is an immorality if there is no edge between X and Y (unmarried parents who have a child)
- Thm 3.3.14. G1 and G2 have the same skeleton and set of immoralities iff they are I-equiv.

Examples



Markov properties of DAGs

- DF: F factorizes over G
- DG: $I(G) \subseteq I(P)$
- DL: $I_l(G) \subseteq I(P)$

$$\begin{tikzcd}
DF \arrow[r, Rightarrow, "\substack{3.3.3 \\ \text{(soundness} \\ \text{dsep)}}"'] \arrow[rr, bend right=50, Rightarrow, "3.2.9"'] & DG \arrow[r, Rightarrow, "\text{trivial}"'] & DL \arrow[ll, bend right=50, Rightarrow, "3.2.7"']
\end{tikzcd}$$

Based on Jordan ch 4

Outline

- Administrivia
- Overview
- Local markov property, factorization (3.2)
- Global markov property (3.3)
- Deriving graphs from distributions (3.4)

Deriving graphs from distributions

- So far, we have discussed how to derive distributions from graphs.
- But how do we get the DAG?
- Assume we have access to the true distribution P, and can answer questions of the form

$$P \models X \perp Y | Z$$

- For finite data samples, we can approximate this oracle with a CI test – the frequentist approach to graph structure learning (see ch 18)
- What DAG can be used to represent P?

Minimal I-map

- The complete DAG is an I-map for any distribution (since it encodes no CI relations)
- Def 3.4.1. A graph K is a minimal I-map for a set of independencies I if it is an I-map for I, and if the removal of even a single edge from K renders it not an I-map.
- To derive a minimal I-map, we pick an arbitrary node ordering, and then find some minimal subset U to be X_i's parents, where

$$X_i \perp \{X_1, \ldots, X_{i-1}\} \setminus U | U$$

- (K2 algorithm replace this CI test with a Bayesian scoring metric: sec 18.4.2).

Effect of node ordering

- “Bad” node orderings can result in dense, unintuitive graphs.
- Eg L,S,G,I,D. Add L. Add S: must add L as parent, since $P \not\models L \perp S$ Add G: must add L,S as parents.

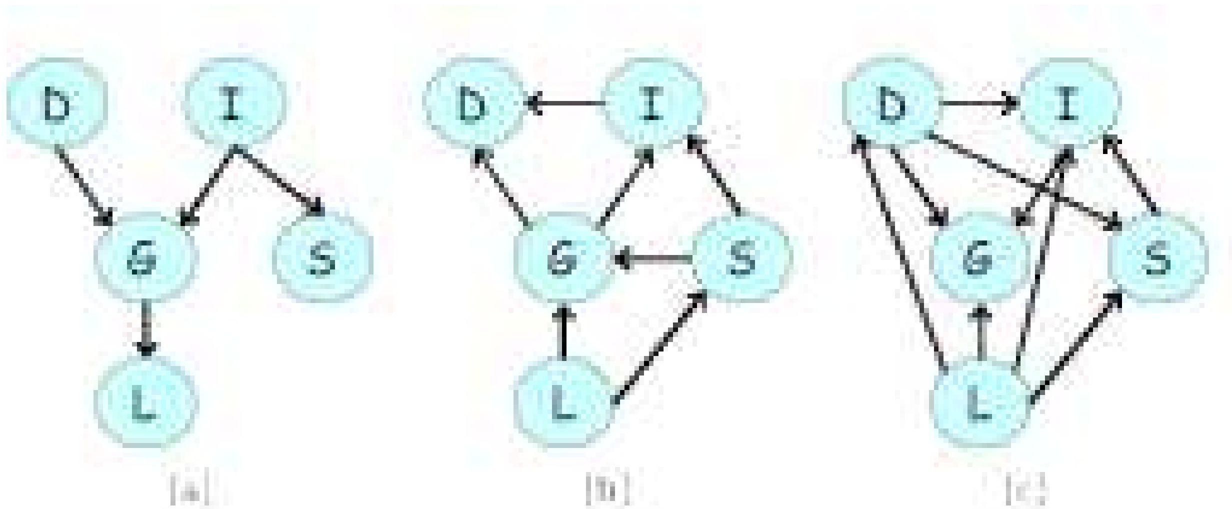
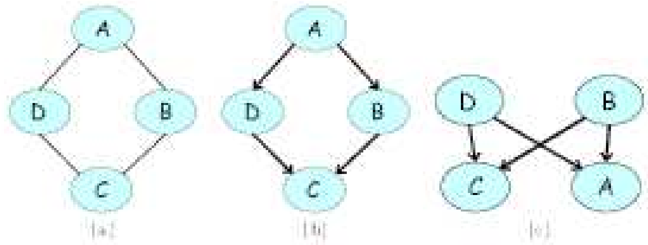


Figure 3.8: Three minimal I-maps for $P_{student}$, induced by different orderings: (a) D, I, S, G, L (b) L, S, G, I, D (c) L, D, S, I, G

Perfect maps

- Minimal I-maps can have superfluous edges.
- Def 3.4.2. Graph K is a perfect map for a set of independencies I if I(K)=I. K is a perfect map for P if I(K)=I(P).
- Not all distributions can be perfectly represented by a DAG.
- Eg let Z = xor(X,Y) and use some independent prior on X, Y. Minimal I-map is X -> Z <- Y. However, X $\perp$ Z in I(P), but not in I(G).
- Eg. A $\perp$ C | {B,D} and B $\perp$ D | {A,C}



Finding perfect maps

- If P has a perfect map, we can find it in polynomial time, using an oracle for the CI tests.
- We can only identify the graph up to I-equivalence, so we return the PDAG that represents the corresponding equivalence class.
- The method* has 3 steps (see sec 3.4.3)
  - Identify undirected skeleton
  - Identify immoralities
  - Compute eclass (compelled edges)
- This algorithm has been used to claim one can infer causal models from observational data, but this claim is controversial

Algorithm due to Verma & Pearl 1991, Spirtes, Glymour, Scheines 1993, Meek 1995