Royal Institute of Technology

STATISTICAL METHODS IN CS, CH 10, 19, 20

Lecture 6

LAST LECTURE

- DGM
- Factorization of likelihood
- MLE estimates for categorical CPDs
- Factorization of posterior for decomposable prior
- Posterior for categorical CPDs
- Semantics of the DAG, Independence

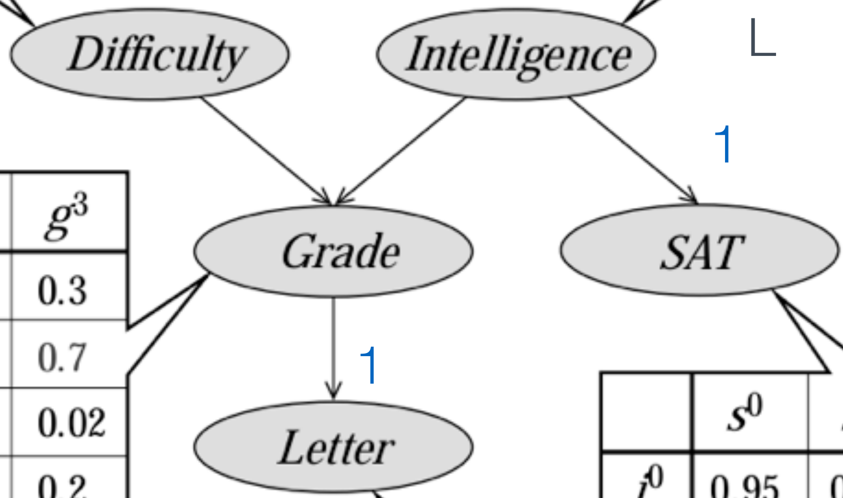
THIS LECTURE

- DP for trees
  - independent set
  - marginalization
- UGMs
- Converting a DGM to a UGM

EXTENDED STUDENT EXAMPLE

d^0	d^1
0.6	0.4
L	H

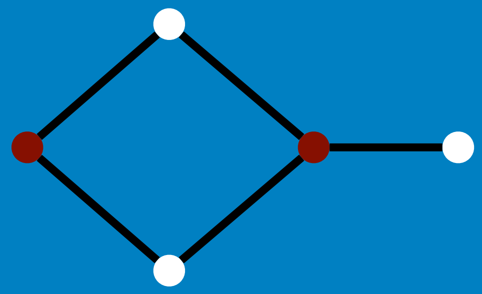
i^0	i^1
0.7	0.3
L	H



	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^1, d^0	0.9	0.08	0.02
i^1, d^1	0.5	0.3	0.2
	B		L

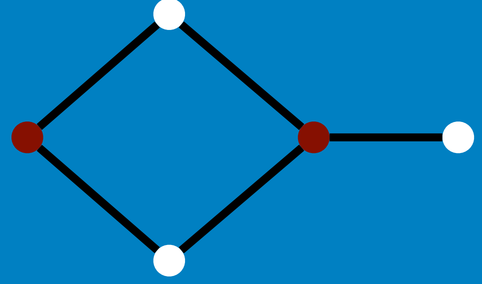
	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8
	L	H

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^3	0.99	0.01
	L	B



INDEPENDENT SET

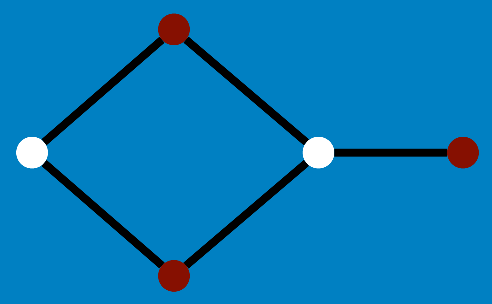
- $I \subseteq V(G)$ is and independent set if there is no edge between any pair of vertices of $V(G)$



MAXIMUM INDEPENDENT SET

- An independent set is a maximal independent set if it is *not a proper subset* of an independent set

MAXIMUM INDEPENDENT SET

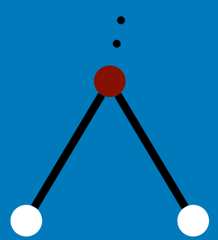


- An independent set is a maximum independent set if there is no *larger* independent set

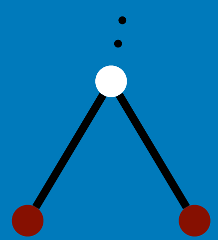
COMPUTATIONAL PROBLEM

- <u>Maximum independent set</u>
  - Input: graph G
  - Output: a maximum independent set I of G
- NP-complete in general (etc)
- Easy for trees
  - Since including a leaf is always at least as beneficial as including its parent, for cherry better.

If you give me



I replace it by

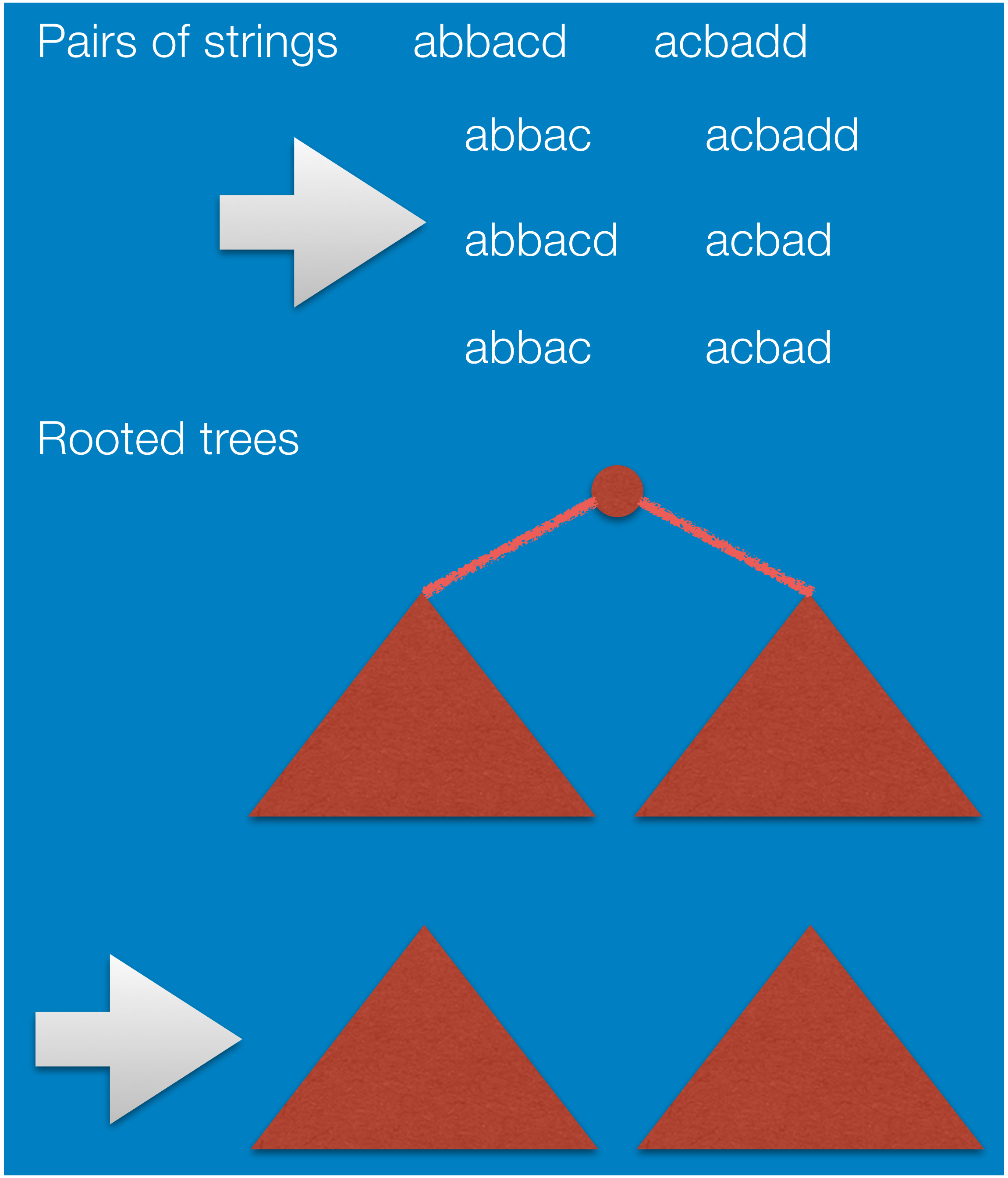


which is better

COMPUTATIONAL PROBLEM

- Maximum weight independent set
  - Input: graph G, and weight function w: V(G) → R
  - Output: independent set I of G maximizing

$$w(I) = \sum_{v \in I}$$



DP

- What is a subproblem?
- What is a subsolution?
- How do we decompose into smaller subproblems?
- How do we combine subsolutions into larger?
- How do we enumerate?
- How many and what time?

SUBPROBLEMS

For every vertex v, 2 subproblems

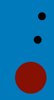
Solution: weight of max weight independent set in the rooted subtree with v in or out

- Denote solutions s(v,in) and s(v,out)
  - i.e., the weight of the maximum weight independent set
- In the end the solution is max(w(root,in), w(root,out))
- Tracing back-pointers gives the actual set

SUBSOLUTIONS FOR LEAVES

For leaf l

l out 

l in 

Solution: weight of max weight independent set in the rooted subtree with v in or out

- s(l,out) := 0
- s(l,in) := w(l)

SUBSOLUTIONS FOR INTERNAL

For leaf internal vertex v

SUBSOLUTIONS FOR LEAVES

For leaf internal vertex v

SUBSOLUTIONS FOR LEAVES

For leaf internal vertex v

SUBSOLUTIONS FOR LEAVES

For leaf internal vertex v

SUBSOLUTIONS FOR LEAVES

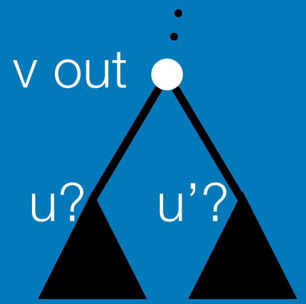
For leaf internal vertex v

SUBSOLUTIONS FOR LEAVES

For leaf internal vertex v

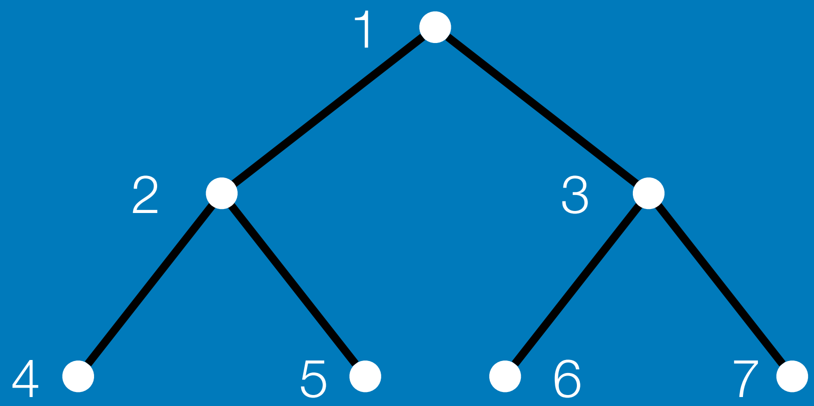
SUBSOLUTIONS FOR LEAVES

For leaf internal vertex v



- $s(v,in) := w(v) + s(u,out) + s(u',out)$
- $s(v,out) := \max_{a,b \in \{in,\, out\}} \{s(u,a) + s(u',b)\}$

EXAMPLE



Vertex	1	2	3	4	5	6	7
w	5	14	2	3	4	8	9
in	29	14	2	3	4	8	9
out	31	7	17	0	0	0	0

MARGINALIZE WITH EVIDENCE

$$p(\boldsymbol{X}_m | \boldsymbol{x}_e) = \frac{\sum_{\boldsymbol{x}_{V\setminus(m\cup e)}} p(\boldsymbol{X}_m, \boldsymbol{x}_{V\setminus(m\cup e)}, \boldsymbol{x}_e)}{\sum_{\boldsymbol{x}_{V\setminus e}} p(\boldsymbol{x}_{V\setminus e}, \boldsymbol{x}_e)}$$

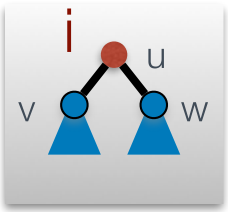
- Boils down to marginalisation, i.e., summing out
- Summing out V binary hidden variables – $O(2^V)$
- K values – $O(K^V)$

ALGORITHM - MARGINALIZATION TREE DGM

- Given DGM with
  - G=T binary directed tree
  - Bernoulli CPDs
  - not summed out x_e,
- Compute

$$\sum_{\boldsymbol{x}_{V \setminus e}} p(\boldsymbol{x}_{V \setminus e}, \boldsymbol{x}_e)$$

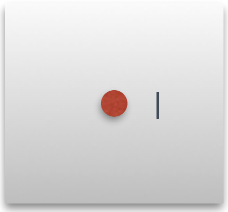
- Subproblem, subsolution



$$s(u,i) = \sum_{x_{V(T_u)\setminus e}} P(x_{V(T_u)\setminus(e \cup \{v\})}, x_{V(T_u)\cap e} | X_u = i)$$

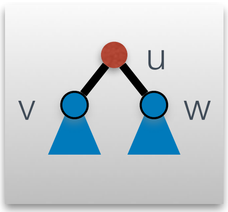
ALGORITHM - MARGINALIZATION TREE DGM

- Visit the vertices of T from leaves to root
  - when at leaf l



$$s(l,i) = \begin{cases} 0 \text{ if } l \in e \text{ and } x_l \neq i \\ 1 \text{ otherwise} \end{cases}$$

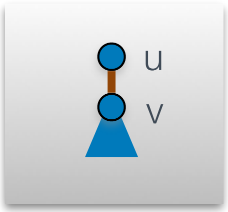
  - when at vertex u with children v and w



$$s(u,i) = \begin{cases} 0 \text{ if } u \in e \text{ and } x_u \neq i \text{ otherwise case below} \\ \left(\sum_{j\in\{0,1\}} P(X_v = j|X_u = i)s(v,j)\right)\left(\sum_{j\in\{0,1\}} P(X_w = j|X_u = i)s(w,j)\right) \end{cases}$$

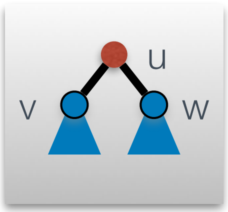
INTRODUCING AN "EDGE" PROBABILITY

- Visit the vertices and edges of T from leaves to root
  - when at edge uv (u has another child too)



$$s(uv,i) = \begin{cases} 0 \text{ if } u \in e \text{ and } x_u \neq i \text{ otherwise case below} \\ \sum_{j\in\{0,1\}} P(X_v = j | X_u = i) s(v,j) \end{cases}$$

  - when at vertex u with children v and w

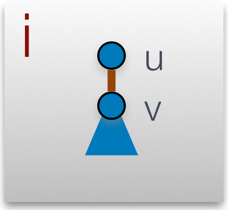


$$\xcancel{s(u,i) = \begin{cases} 0 \text{ if } u \in e \text{ and } x_u \neq i \text{ otherwise case below} \\ \left(\sum_{j\in\{0,1\}} P(X_v = j | X_u = i) s(u,i)\right)\left(\sum_{j\in\{0,1\}} P(X_w = j | X_u = i) s(u,i)\right) \end{cases}}$$

$$s(u,i) = \begin{cases} 0 \text{ if } u \in e \text{ and } x_u \neq i \\ s(uv,i)s(uw,i) \text{ otherwise} \end{cases}$$

ALGORITHM - MARGINALIZATION TREE DGM

- Subproblem, subsolution

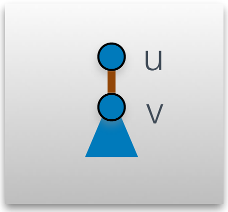


$$s(uv, i) = \sum_{x_{V(T_v)\setminus e}} P(x_{V(T_v)\setminus e}, x_{V(T_v)\cap e} | X_u = i)$$

$$s(uv, i) = \sum_{j \in \{0,1\}} P(X_v = j | X_u = i) s(v, j)$$

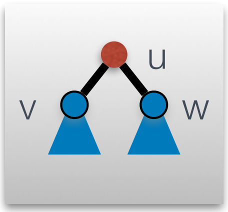
ZOOM IN ON NON-LEAVES

- Visit the vertices and edges of T from leaves to root
  - when at edge uv (u has another child too)



$$s(uv, i) = \begin{cases} 0 \text{ if } u \in e \text{ and } x_u \neq i \text{ otherwise case below} \\ \sum_{j \in \{0,1\}} P(X_v = j | X_u = i) s(v, j) \end{cases}$$

  - when at vertex u with children v and w

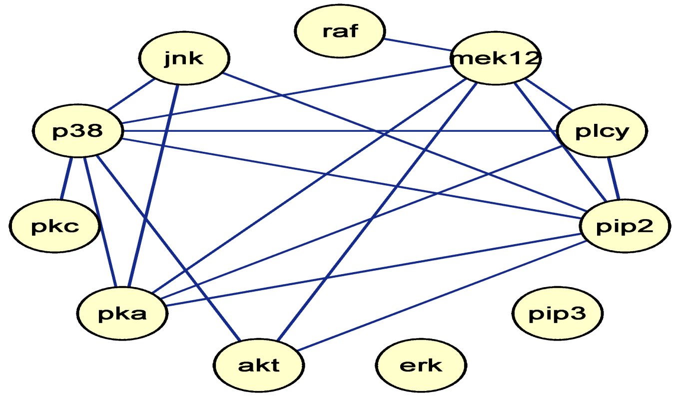


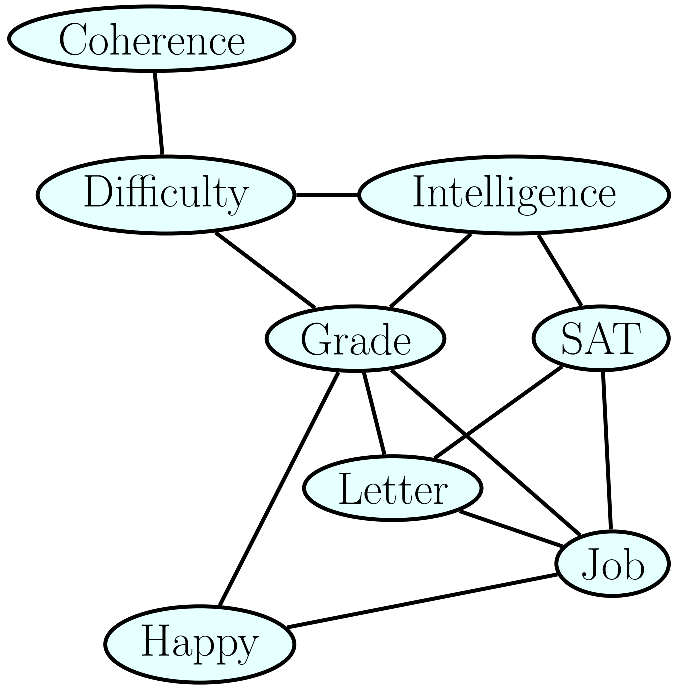
$$s(u, i) = \begin{cases} 0 \text{ if } u \in e \text{ and } x_u \neq i \text{ otherwise case below} \\ \left(\sum_{j \in \{0,1\}} P(X_v = j | X_u = i) s(u, i)\right)\left(\sum_{j \in \{0,1\}} P(X_w = j | X_u = i) s(u, i)\right) \end{cases}$$

$$s(u, i) = \begin{cases} 0 \text{ if } u \in e \text{ and } x_u \neq i \\ s(uv, i) s(uw, i) \text{ otherwise} \end{cases}$$

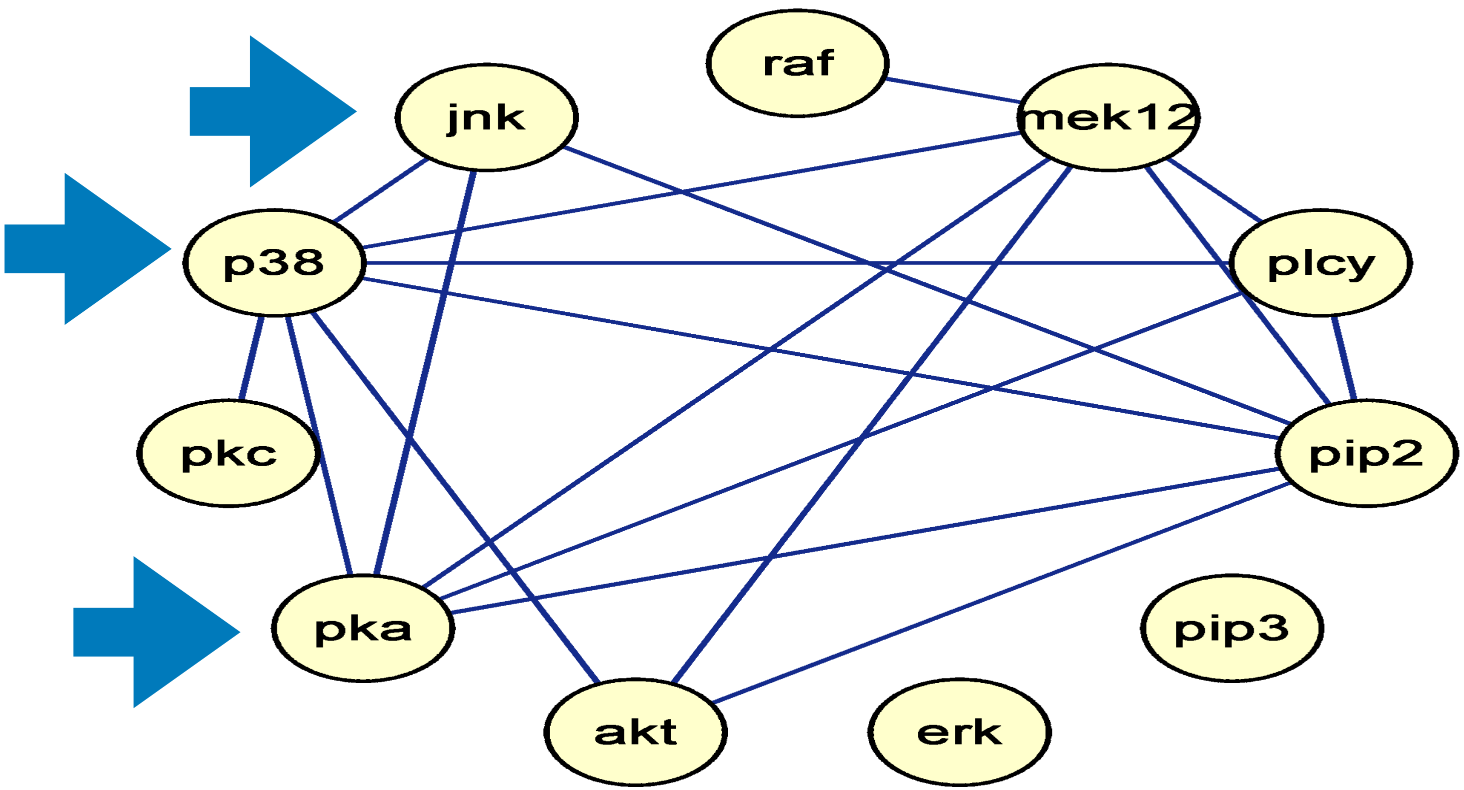
UGM

- ⋆ UGMs - Undirected graphical models
- ⋆ What is the direction between 2 pixels, 2 proteins?
- ⋆ Probabilistic interpretation?
- ⋆ p factorizes over G – can be expressed as normalized product over factors associated with cliques

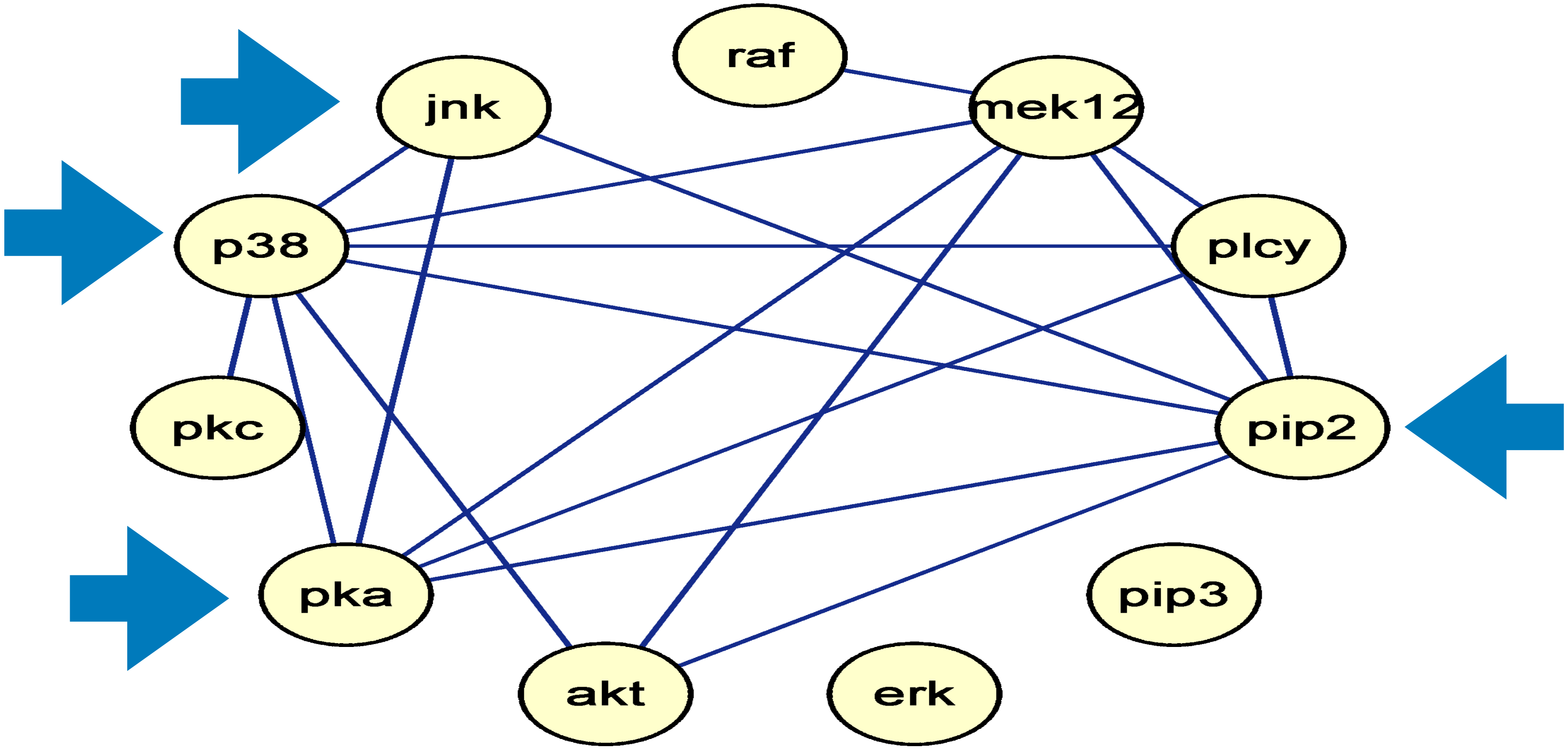




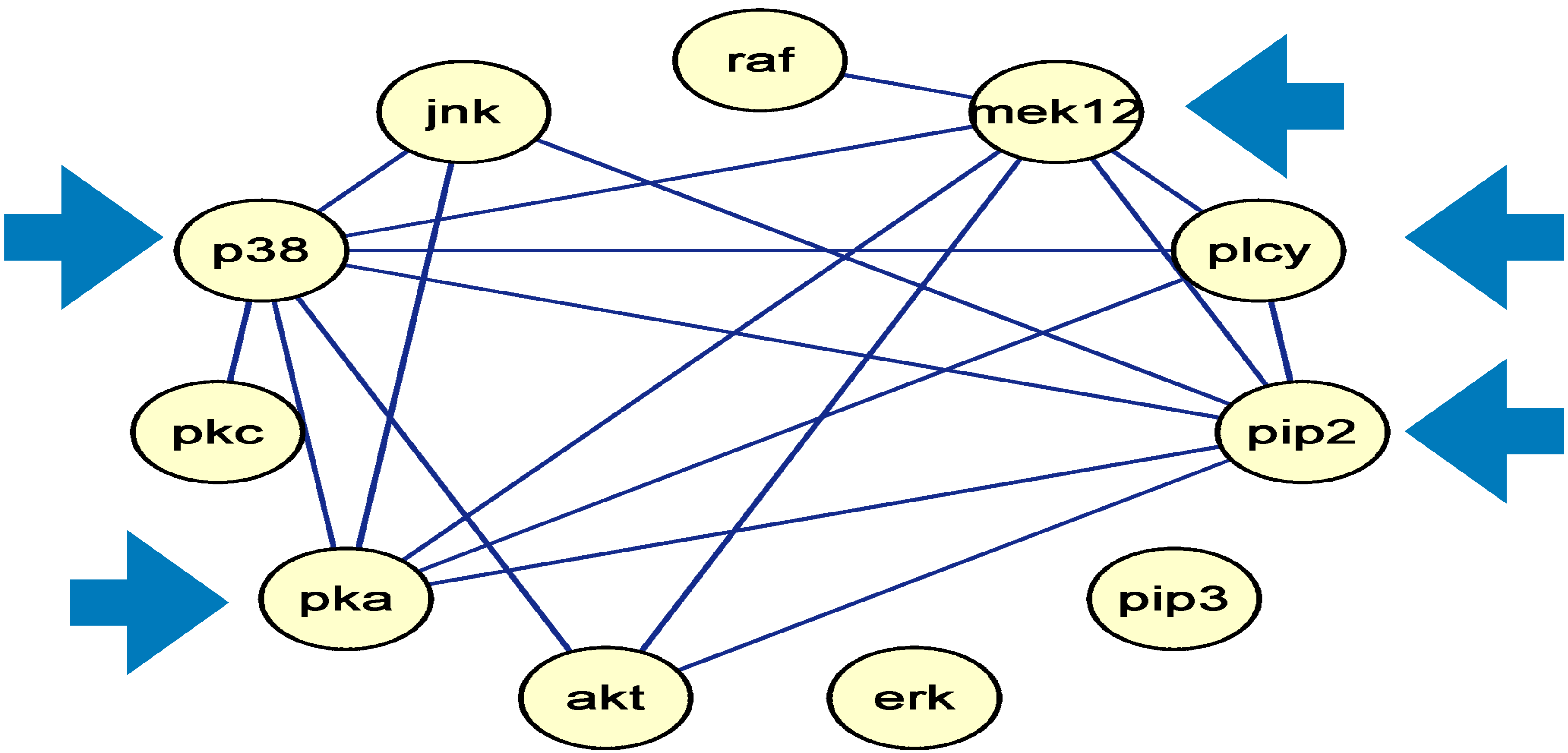
EXAMPLE CLIQUE



EXAMPLE MAXIMAL CLIQUE



EXAMPLE MAXIMUM CLIQUE

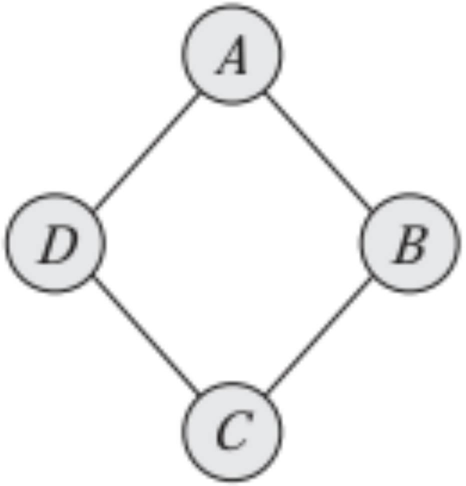


UGM

- An undirected graph G with so-called factors associated with its maximal cliques $C(G)$, for $C \in C(G)$ factor ψ_C
- ψ_C is a function from the clique's variables (the scope) to non-neg real numbers

$$p(x_1, \ldots, x_V) = \frac{1}{Z} \prod_{C \in C(G)} \psi_C(x_C)$$

$$Z = \sum_{x_1, \ldots, x_V} \prod_{C \in C(G)} \psi_C(x_C)$$



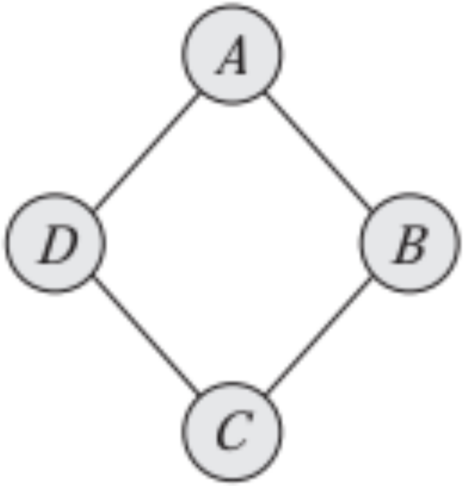
Scope	A,B			B,C			C,D			D,A		
	$\phi_1(A,B)$			$\phi_2(B,C)$			$\phi_3(C,D)$			$\phi_4(D,A)$		
	a^0	b^0	30	b^0	c^0	100	c^0	d^0	1	d^0	a^0	100
	a^0	b^1	5	b^0	c^1	1	c^0	d^1	100	d^0	a^1	1
	a^1	b^0	1	b^1	c^0	1	c^1	d^0	100	d^1	a^0	1
	a^1	b^1	10	b^1	c^1	100	c^1	d^1	1	d^1	a^1	100
	(a)			(b)			(c)			(d)		

Factors – misconception example

- $$P(A,B,C,D) = \frac{1}{Z}\phi_1(A,B)\phi_2(B,C)\phi_3(C,D)\phi_4(D,A)$$

- $$Z = \sum_{a,b,c,d} \phi_1(a,b)\phi_2(b,c)\phi_3(c,d)\phi_4(d,a)$$

PROBABILISTIC INTERPRETATION



$\phi_1(A,B)$			$\phi_2(B,C)$			$\phi_3(C,D)$			$\phi_4(D,A)$		
a^0	b^0	30	b^0	c^0	100	c^0	d^0	1	d^0	a^0	100
a^0	b^1	5	b^0	c^1	1	c^0	d^1	100	d^0	a^1	1
a^1	b^0	1	b^1	c^0	1	c^1	d^0	100	d^1	a^0	1
a^1	b^1	10	b^1	c^1	100	c^1	d^1	1	d^1	a^1	100
(a)			(b)			(c)			(d)		

Misconception

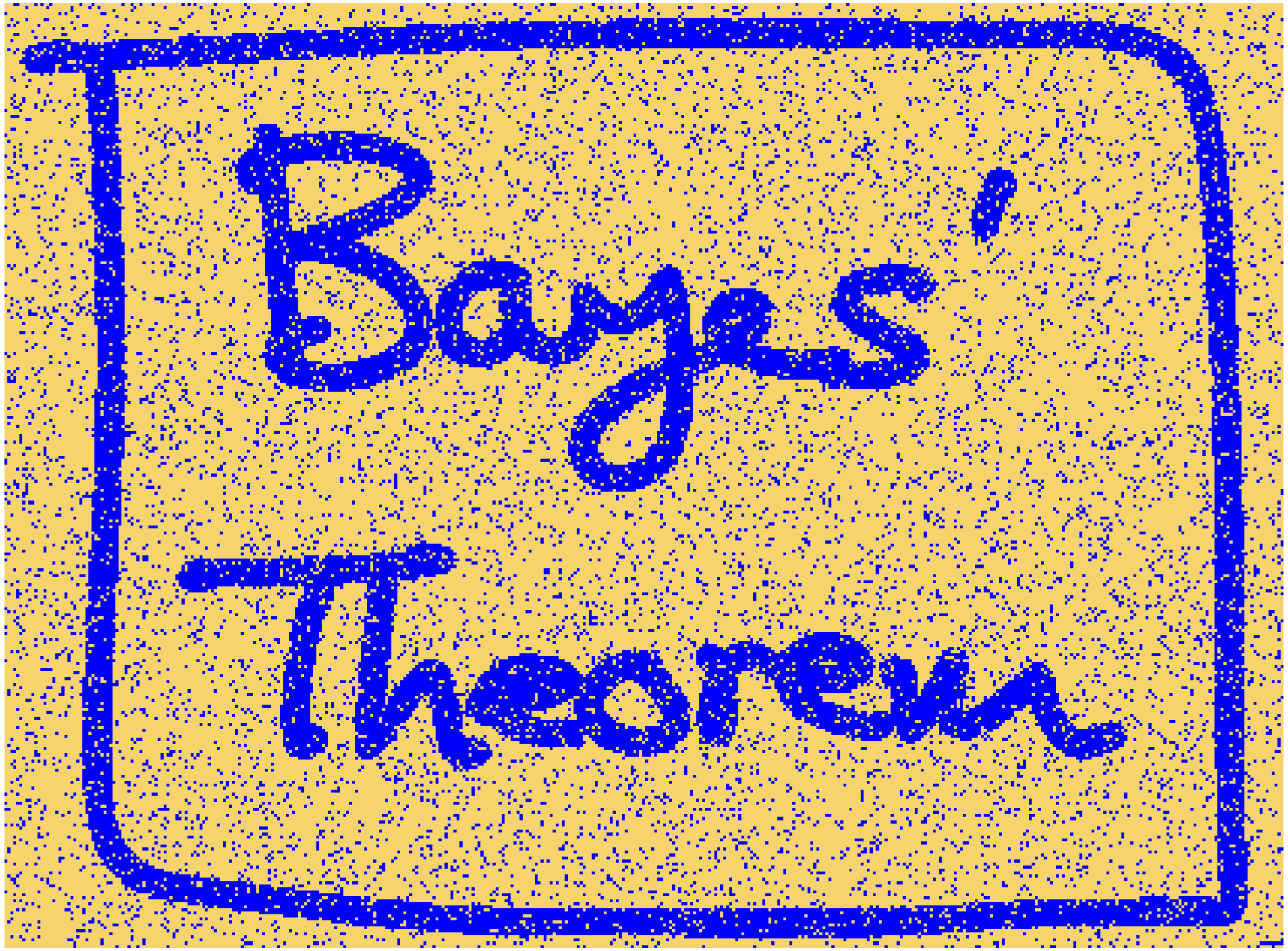
-

$$\phi_1(A=1,B=1)\phi_2(B=1,C=0)\phi_3(C=0,D=1)\phi_4(D=1,A=1)$$
$$= 10 \cdot 1 \cdot 100 \cdot 100$$
$$= 100000$$

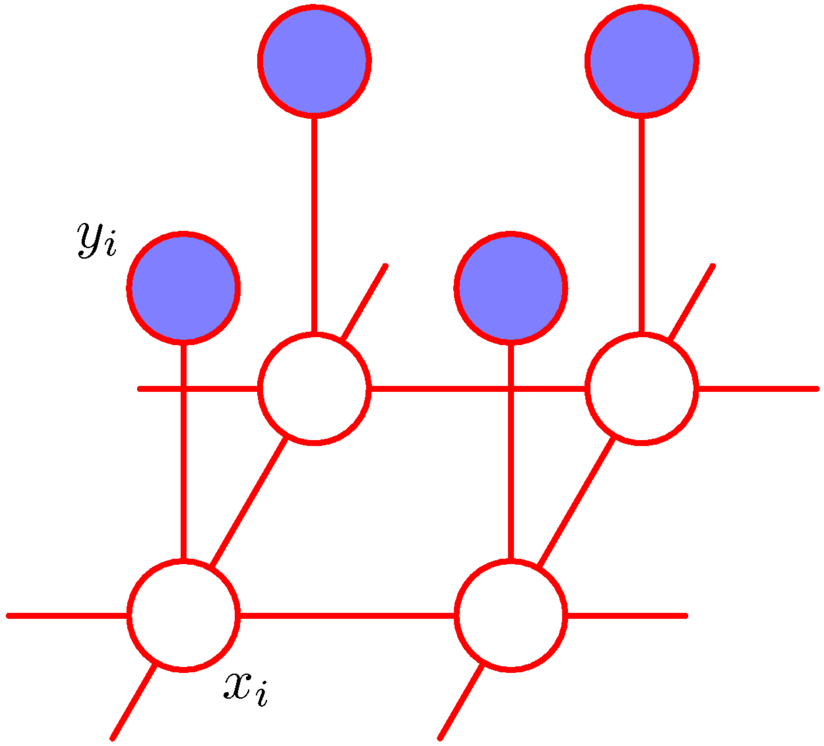
$$Z = \sum_{a,b,c,d} \phi_1(a,b)\phi_2(b,c)\phi_3(c,d)\phi_4(d,a)$$

A FACTOR PRODUCT

DE-NOISING



ISING MODEL-DE-NOISING



Values -1,1

Factors of form

$$e^{\beta x_i x_j}$$

and

$$e^{\eta x_i y_i}$$

ISING MODEL-DE-NOISING

Values -1,1

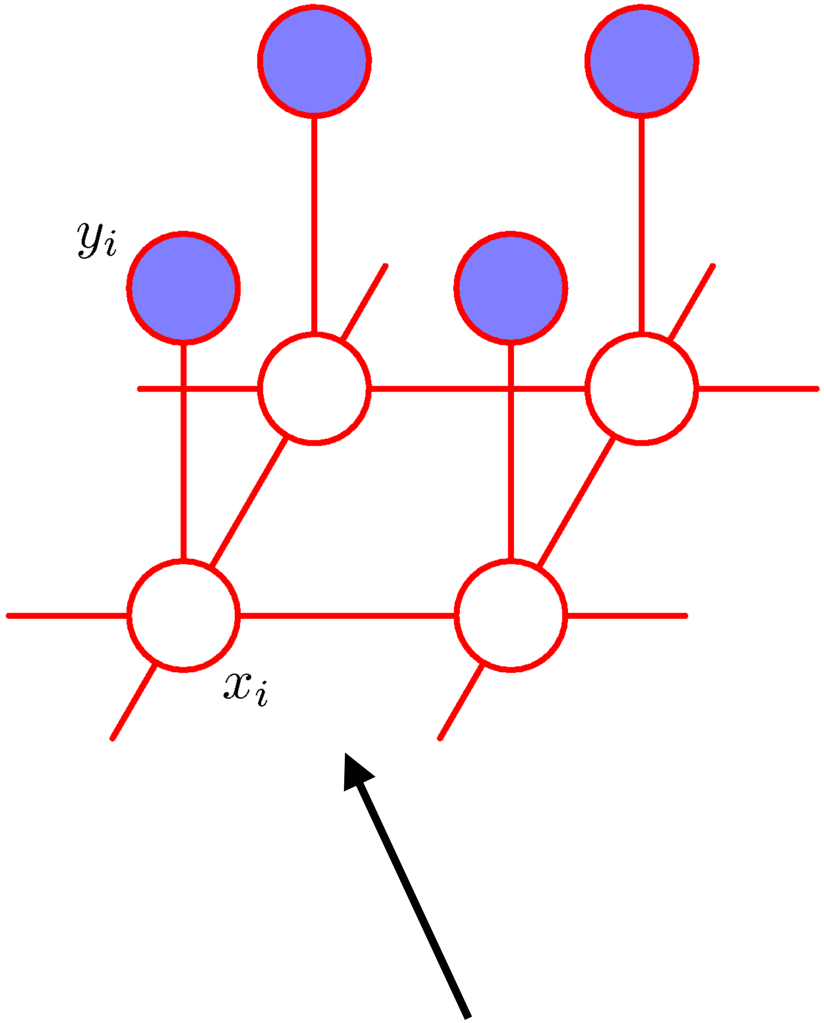
Factors of form

$$e^{\beta x_i x_j}$$

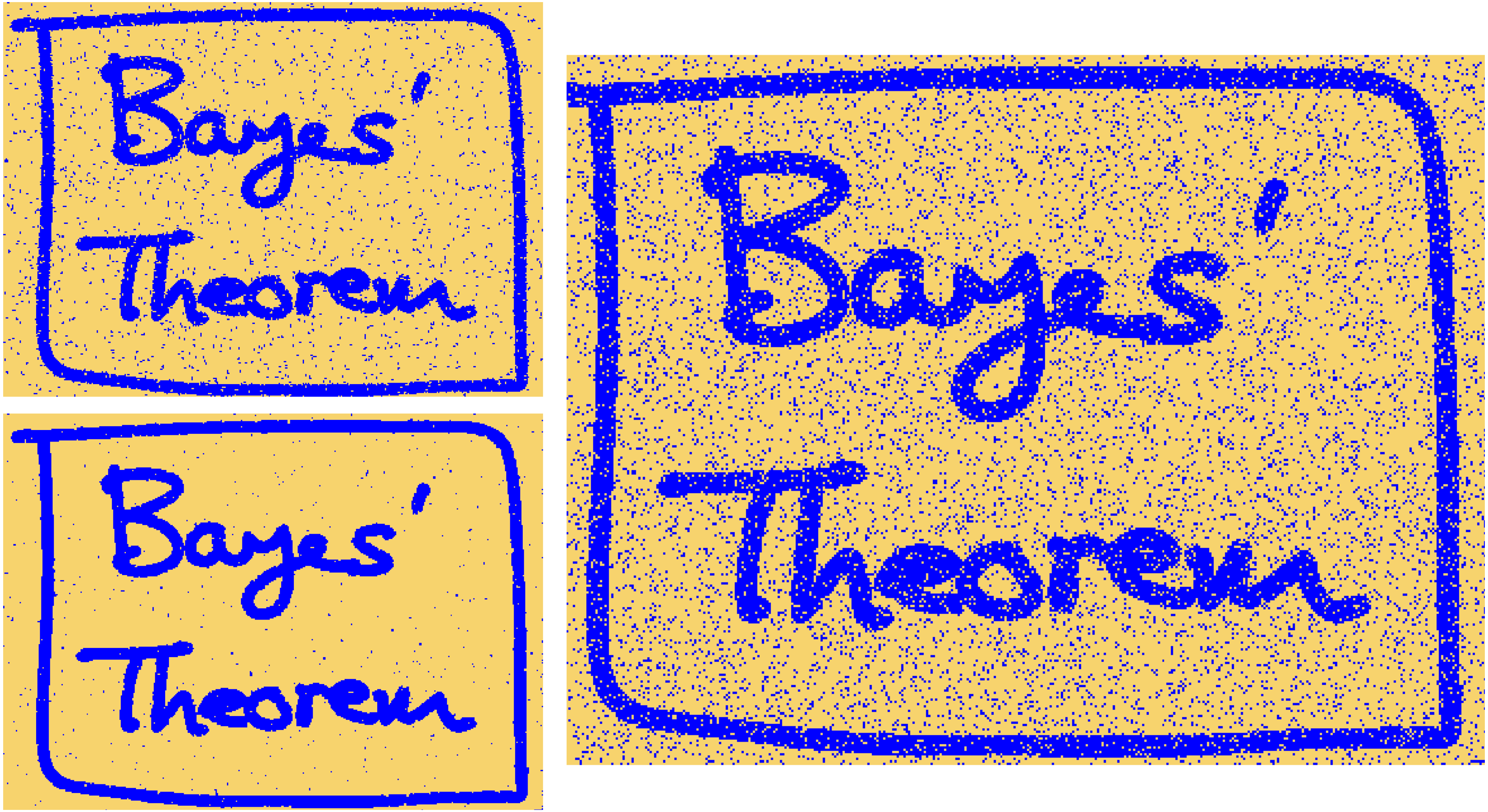
and

$$e^{\eta x_i y_i}$$

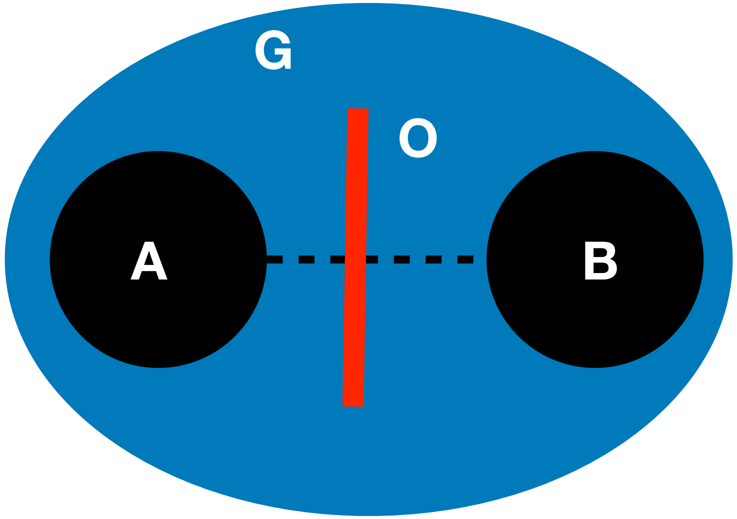
p(y | x) ex Gaussian



- Bipartite graph
- Suggests iterative procedu



- Large is the noisy image; upper, UGM de-noised; and lower, graph cut de-noised



SEPARATION AND CI OF UGM

★ A is separated from B given O in G if there is no path between A and B in G\O

★ In a graph G,

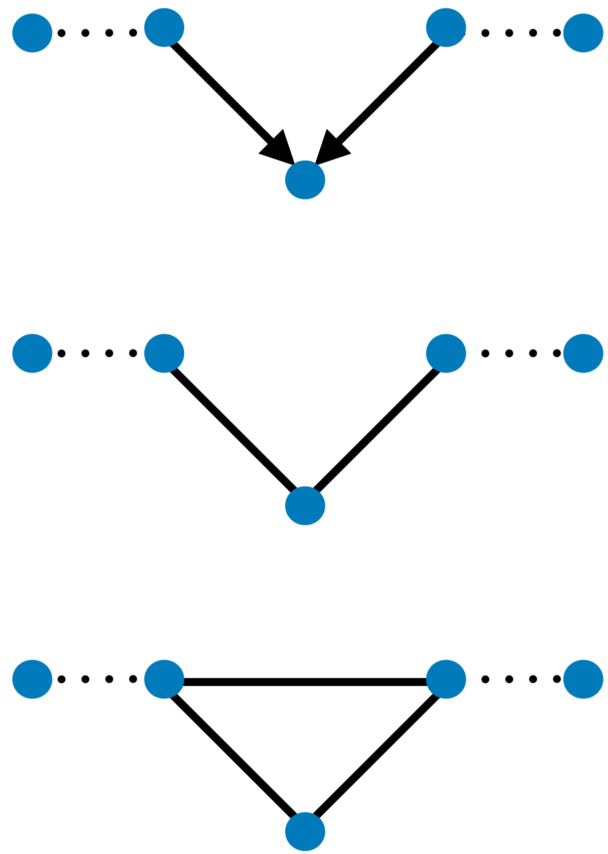
$$\boldsymbol{x}_A \perp_G \boldsymbol{x}_B | \boldsymbol{x}_O$$

$$\Updownarrow$$

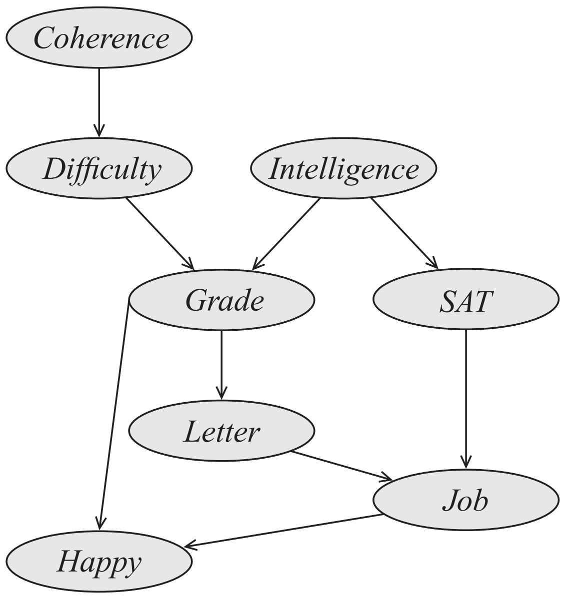
A is separated from B given O

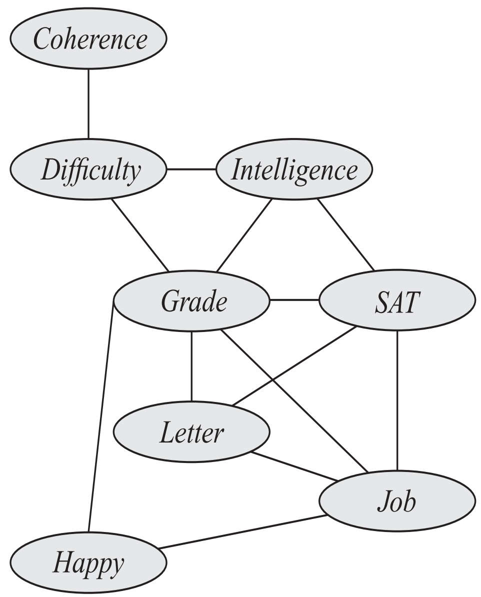
MORALIZATION

- Moralization - add edge between any two parents
- We can moralize a DGM and get a UGM having no more independence relations
- Each family is a clique in the moralized UGM



CONVERTING DGM TO UGM





- Moralize and remove directions
  - does not introduce new independencies!
- Use CPDs as factors

EXACT ALGORITHMS FOR GRAPHICAL MODELS

- Many problems are NP-hard (marginalization etc.)
- For trees, many of them can be solved by DP
- When the graph is “tree-like” find a representation of the “tree-likness” and use it to guide DP
- Unfortunately, finding the representation is not always easy
- Next time assuming the representation is given.
- Independent set as exercise.
- Then marginalization.

THE END