Royal Institute of Technology

STAT. METH. IN CS – JUNCTION TREES, PREP. FOR CH 20, 19

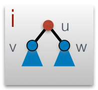
Lecture 7

EXACT ALGORITHMS FOR GRAPHICAL MODELS

- Many problems are NP-hard (marginalization etc.)
- For trees they are 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
- Today assuming the representation is given.
- Independent set as exercise.

ALGORITHM - MARGINALIZATION TREE DGM

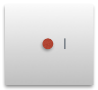
- Given Bernoulli DGM with G binary directed tree and evidence $x_{e,}$ for evidence set e
- Subproblem, subsolution



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

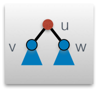
ALGORITHM - MARGINALIZATION TREE DGM

- Given Bernoulli DGM with G binary directed tree and evidence x_e for evidence set e
- Visit the vertices of G from leaves to root
  - when at leaf l



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

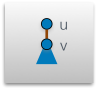
  - when at vertex u with children v and w



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

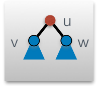
ZOOM IN

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



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

  - when at vertex u with children v and w

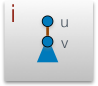


$$\cancel{s(u,i) = \begin{cases} 0 \text{ if } u \in e \text{ and } x_u \neq i \text{ otherwise case below} \\ \left(\sum_{j\in\{0,1\}} P(X_v = j|X_u = i)s(u,i)\right)\left(\sum_{j\in\{0,1\}} P(X_w = j|X_u = i)s(u,i)\right) \end{cases}}$$

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

ALGORITHM - MARGINALIZATION TREE DGM

- Subproblem, subsolution

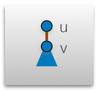


$$s(uv,i) = P(X_{V(T_v)\setminus e}, x_{V(T_v)\cap e} | X_u = i)$$

$$\begin{aligned} s(uv,i) =& P(X_{V(T_v)\setminus e}, x_{V(T_v)\cap e} | X_u = i) \\ =& \sum_{j\in\{0,1\}} P(X_{V(T_v)\setminus e}, x_{V(T_v)\cap e}, X_v = j | X_u = i) \\ =& \sum_{j\in\{0,1\}} P(X_v = j|X_u = i) P(X_{V(T_v)\setminus e}, x_{V(T_v)\cap e}, | X_v = j, X_u = i) \\ =& \sum_{j\in\{0,1\}} P(X_v = j|X_u = i) P(X_{V(T_v)\setminus e}, x_{V(T_v)\cap e}, | X_v = j) \\ =& \sum_{j\in\{0,1\}} P(X_v = j|X_u = i) s(v,j) \end{aligned}$$

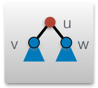
ZOOM IN

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



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

  - when at vertex u with children v and w



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

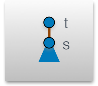
$$s(u,i) = \begin{cases} 0 \text{ if } u \in e \text{ and } x_u \neq i \\ s(uv,i)s(uw,i) \text{ otherwise} \end{cases}$$

SUMMARY - IND. SET ALGORITHM

- Given junction-tree (T,B) for G. Make T binary, let r be the root of T
  - Visit the vertices and edge of T from leaves to root
    - when at leaf t

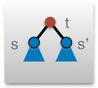
$$I_t(S) = \begin{cases} -\infty \text{ if } S \text{ not ind. set} \\ |S| \text{ if } S \text{ is ind. set} \end{cases}$$

    - at edge st



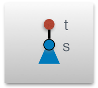
$$I_{st}(S) = \max_{S' \in B(s): S\cap B(s) = S'} I(S')$$

    - if two children s and s'



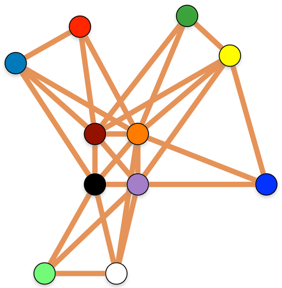
$$I_t(S) = \begin{cases} -\infty \text{ if } S \text{ not ind. set and otherwise the case below} \\ I_{st}(S\cap B(s)) + I_{s't}(S \cap B(s')) + |S \setminus (B(s)\cup B(s'))| - |S\cap B(s) \cap B(S') \end{cases}$$

    - if single child s

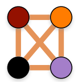


$$I_t(S) = \begin{cases} -\infty \text{ if } S \text{ not ind. set} \\ I(S\cap B(S)) + |S\setminus S'| \text{ if } S \text{ is ind. set} \end{cases}$$

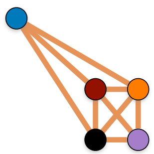
K-TREES (HERE K=3)



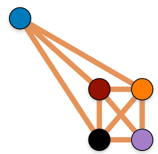
3-TREE

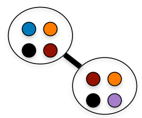


3-TREE



3-TREE



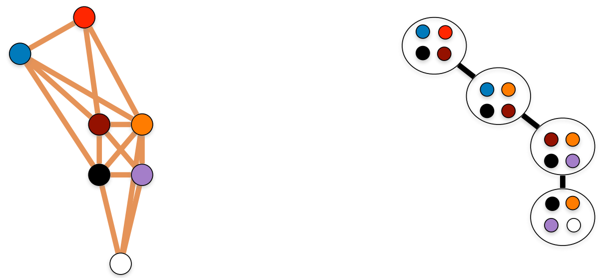


3-TREE

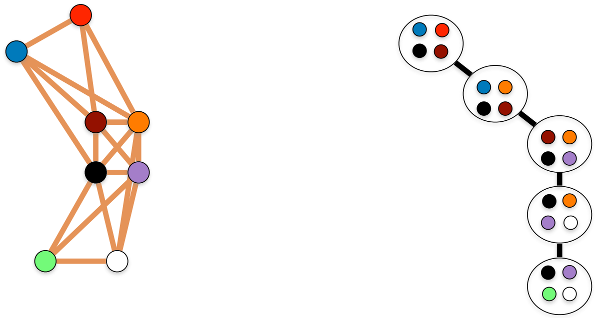
3-TREE

3-TREE

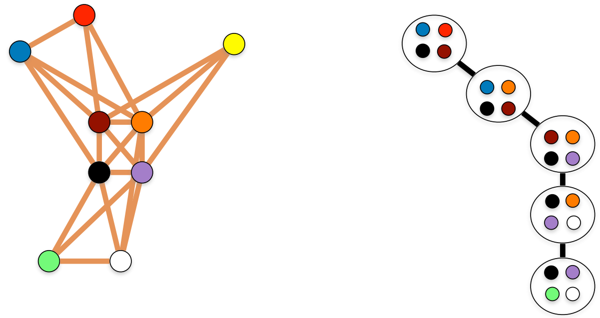
3-TREE



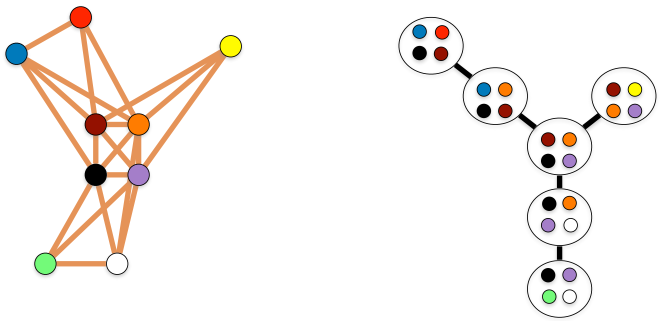
3-TREE



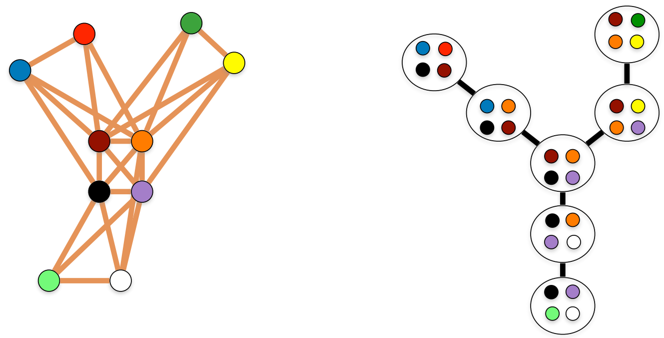
3-TREE



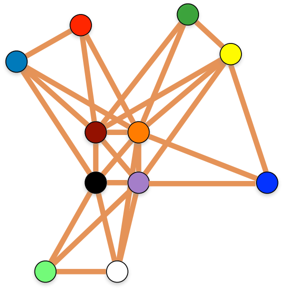
3-TREE

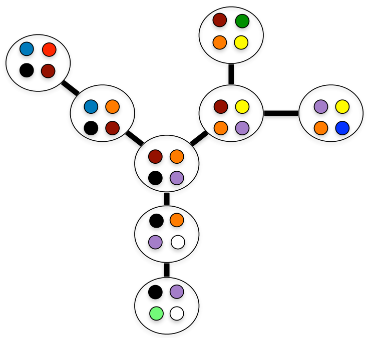


3-TREE



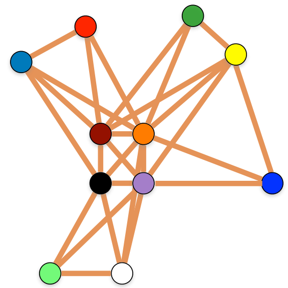
3-TREE

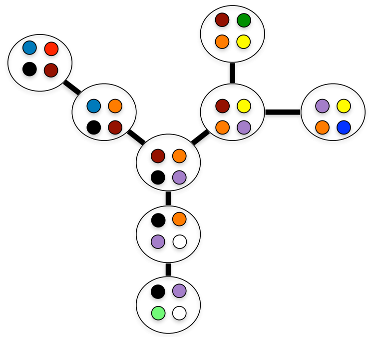




JUNCTION TREE

Graph G	T,B Junction tree

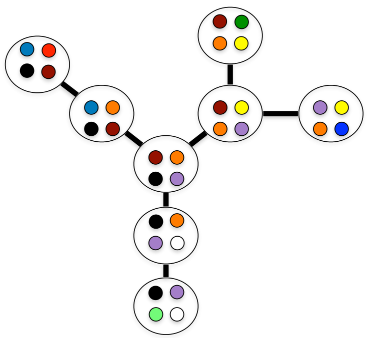




JUNCTION TREE G

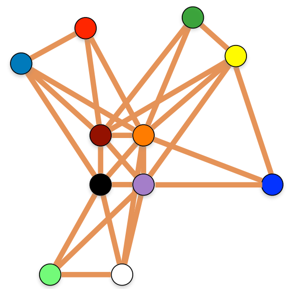
Definition Junction tree	T,B Junction tree

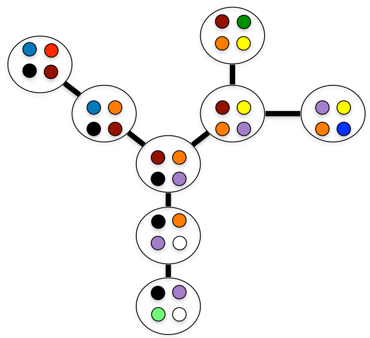
* Each vertex of G in some bag



JUNCTION TREE G

Graph G	T,B Junction tree



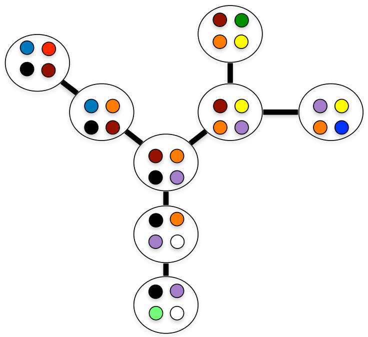


JUNCTION TREE

Definition Junction tree

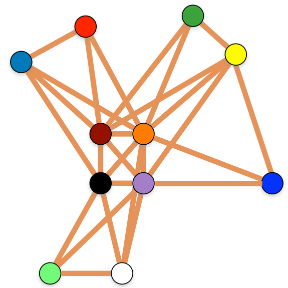
T,B Junction tree

* Each vertex of G in some bag
* Each edge of G in some bag

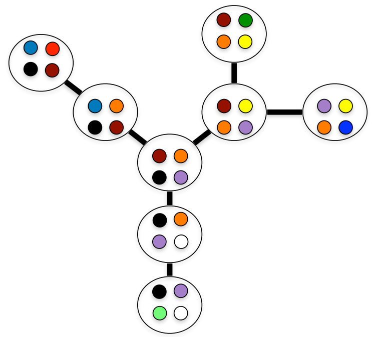


JUNCTION TREE

Graph G



T,B Junction tree

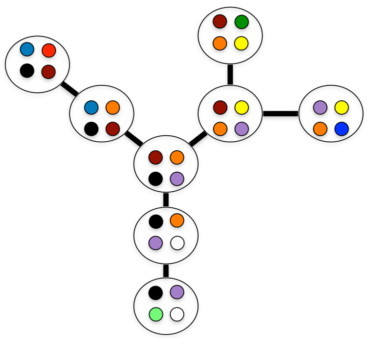


JUNCTION TREE

Definition Junction tree

T,B Junction tree

* Each vertex of G in some bag
* Each edge of G in some bag
* Each vertex of G induce a subtree (running intersection property)

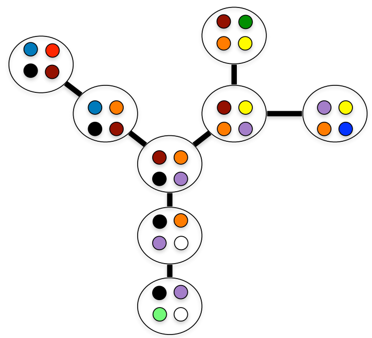


JUNCTION TREE

Definition Junction tree

T,B Junction tree

* Each vertex of G in some bag
* Each edge of G in some bag
* Each vertex of G induce a subtree (running intersection property)

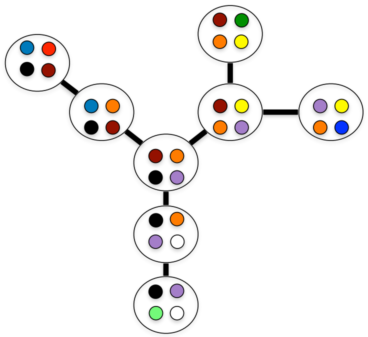


JUNCTION TREE

Definition Junction tree

T,B Junction tree

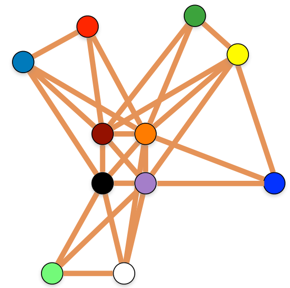
- Each vertex of G in some bag
- Each edge of G in some bag
- Each vertex of G induce a subtree (running intersection property)

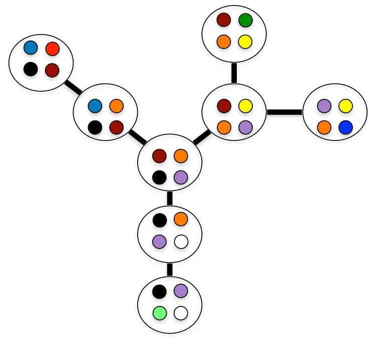


JUNCTION TREE

Graph G

T,B Junction tree



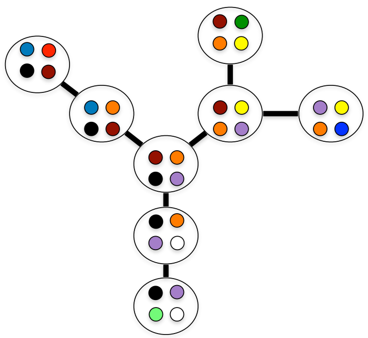


(TREE) WIDTH

Width of junction tree

T,B Junction tree

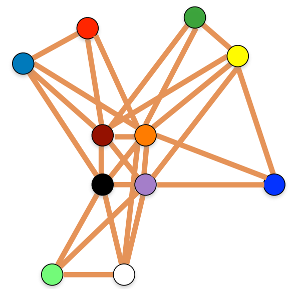
- (Size of largest bag)- 1

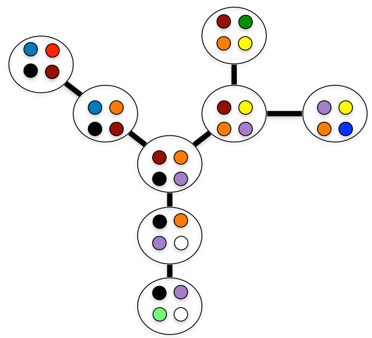


CLIQUES IN G

Graph G

T,B Junction tree

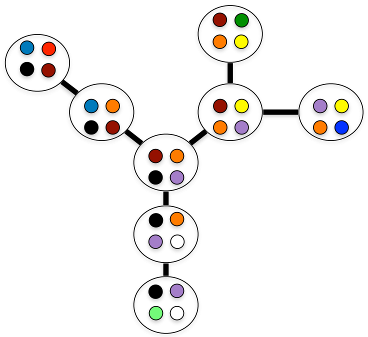




JUNCTION TREE - 2 IMPORTANT PROPERTIES

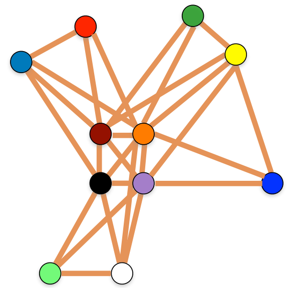
T,B Junction tree

* Every clique can be found in some bag
* The intersection of 2 neighboring bags is a separator

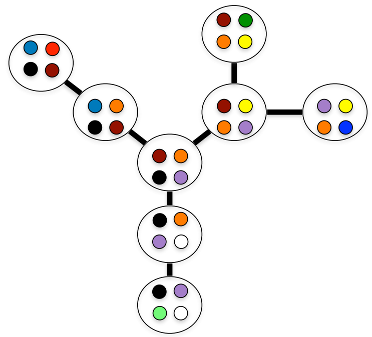


CLIQUES AND SEPARATORS IN G

Graph G

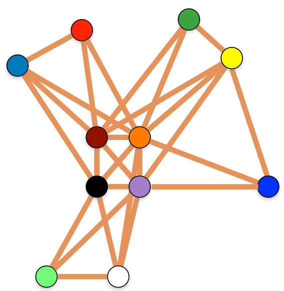


T,B Junction tree

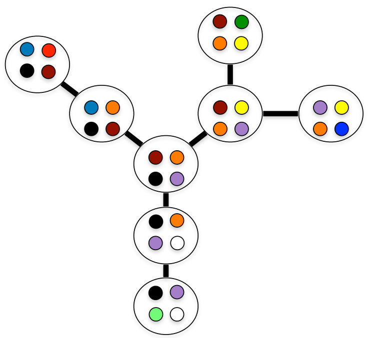


JUNCTION TREE

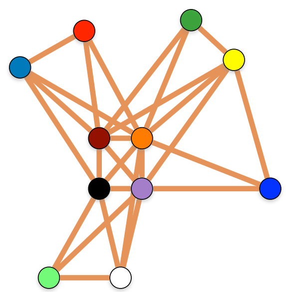
Graph G



T,B Junction tree

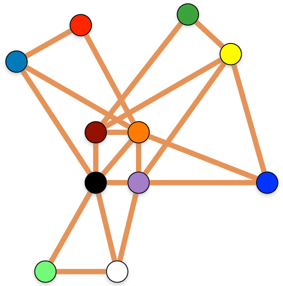


PARTIAL K-TREE



* Removing edges from k-tree gives partial k-tree

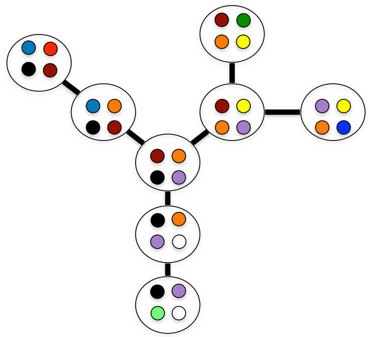
PARTIAL K-TREE



- ⋆Removing edges from k-tree gives partial k-tree

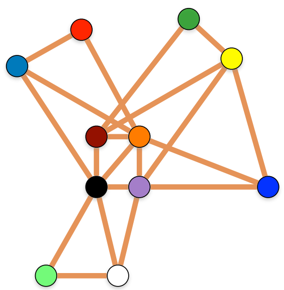
STILL TRUE FOR PARTIAL K-TREE

T,B Junction tree



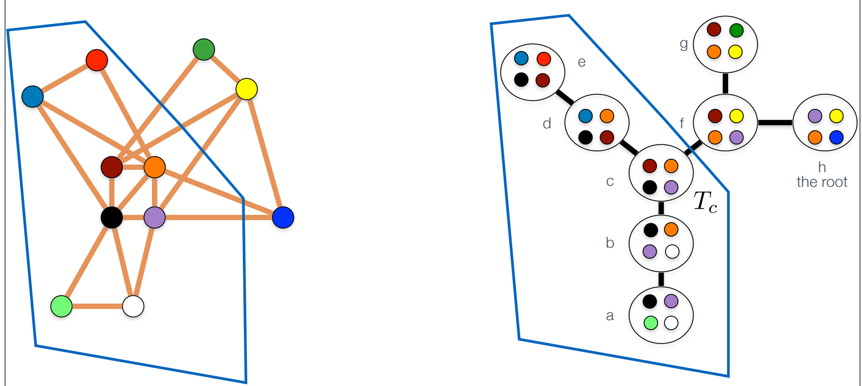
- ⋆ Every clique can be found in some bag
- ⋆ The intersection of 2 neighboring bags is a separator

JUNCTION TREE GUIDED DP - INDEPENDENT SET



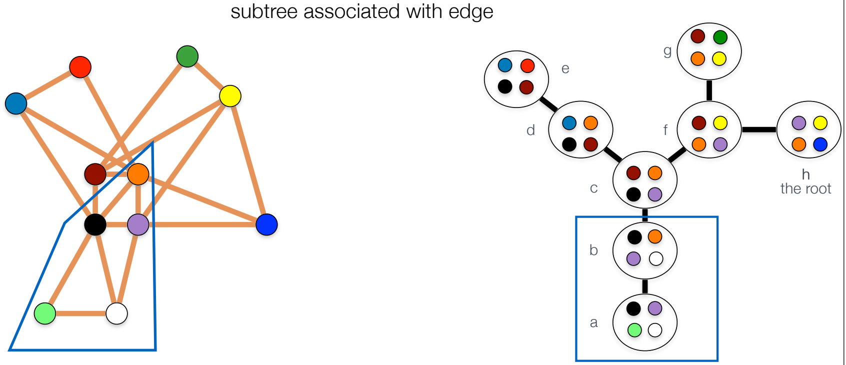
JUNCTION TREE GUIDED DP - SUBPROBLEMS

Subproblems
subtree rooted at c



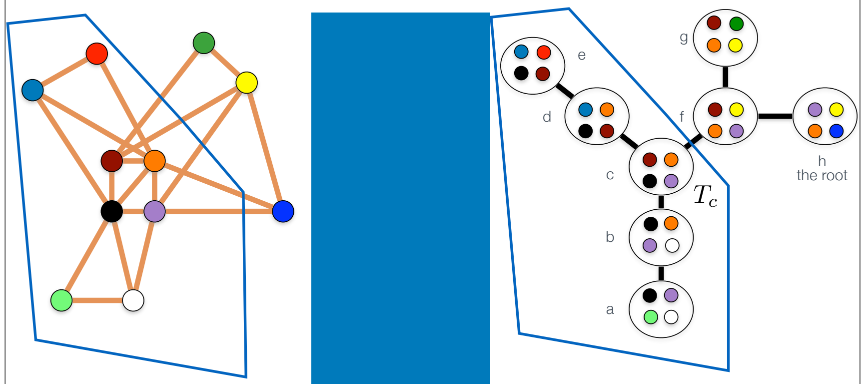
JUNCTION TREE GUIDED DP - SUBPROBLEMS

Subproblems
- subtree rooted at v
- subtree associated with edge



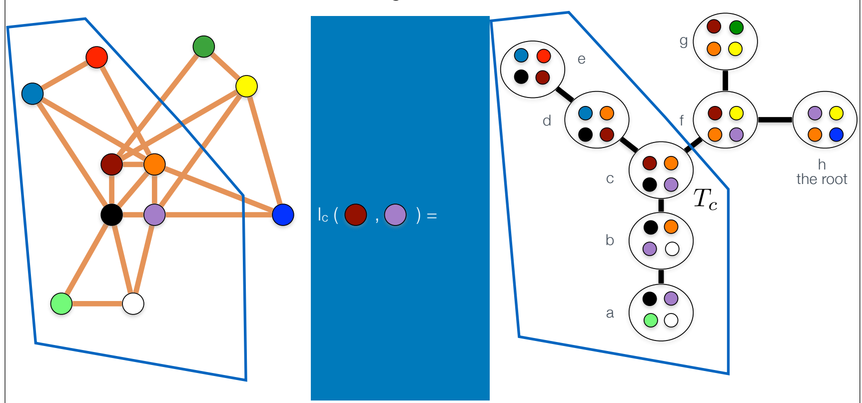
JUNCTION TREE GUIDED DP - SUBSOLUTIONS

I_c table/function with sizes of maximum independent sets in subtree rooted at c based on intersection with c:s bag



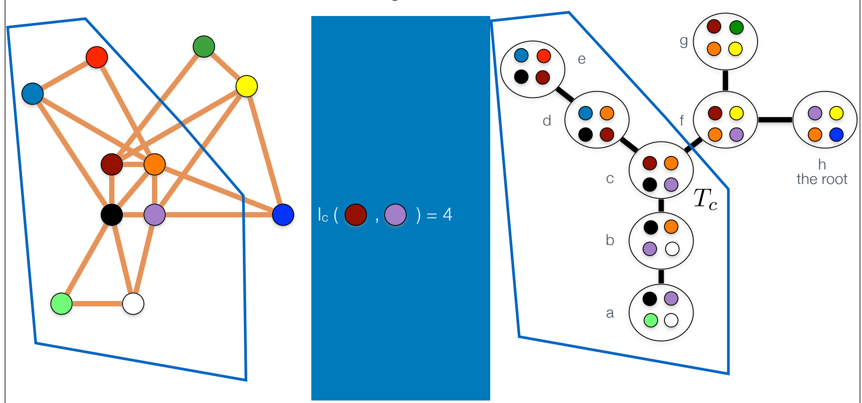
JUNCTION TREE GUIDED DP - SUBSOLUTIONS

I_c table/function with sizes of maximum independent sets in subtree rooted at c based on intersection with c:s bag



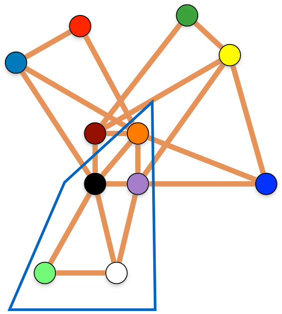
JUNCTION TREE GUIDED DP - SUBSOLUTIONS

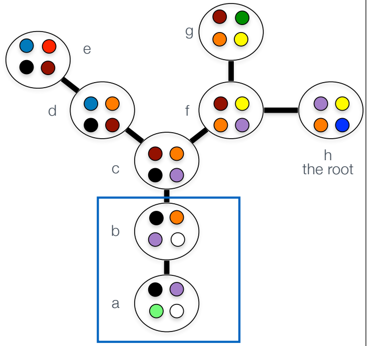
I_v table/function with sizes of maximum independent sets in subtree rooted at v based on intersection with v:s bag



JUNCTION TREE GUIDED DP - SUBSOLUTIONS

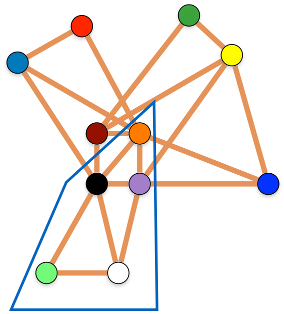
$I_{c\cap b}$ table/function with sizes of maximum independent sets in subtree below c and b based on intersection with the sepset c∩b

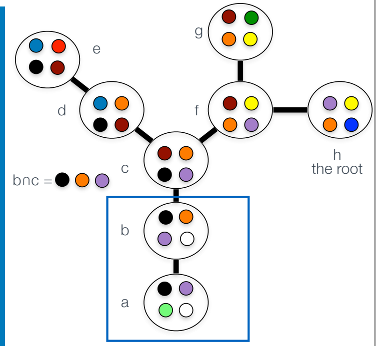




JUNCTION TREE GUIDED DP - SUBSOLUTIONS

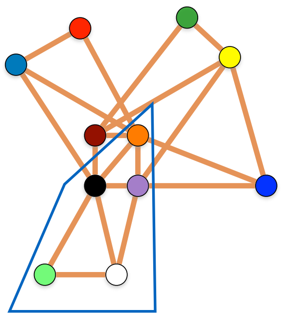
$I_{c\cap b}$ table/function with sizes of maximum independent sets in subtree below c and b based on intersection with the sepset c∩b



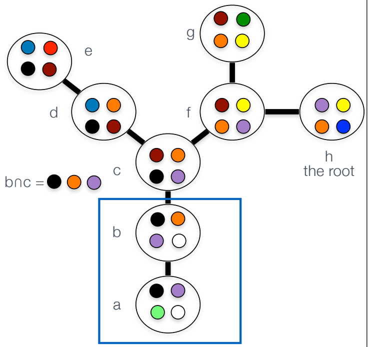


JUNCTION TREE GUIDED DP - SUBSOLUTIONS

$I_{c\cap b}$ table/function with sizes of maximum independent sets in subtree below c and b based on intersection with the sepset c∩b

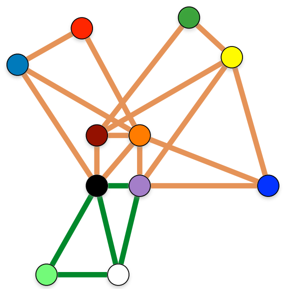


$I_{b\cap c}$ ( ) = 2



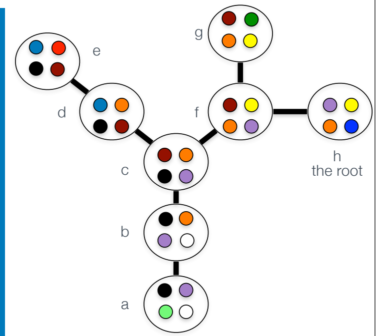
JUNCTION TREE GUIDED DP - COMPUTING

I_v table/function with sizes of maximum independent sets in subtree rooted at v based on intersection with v:s bag



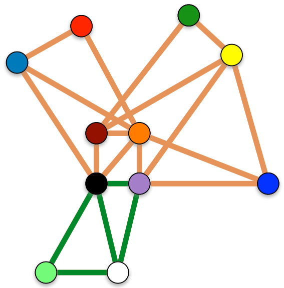
I_a =

				s
1	0	0	0	1
0	1	0	0	1
0	0	1	0	1
0	0	0	1	1
0	1	0	1	2
0	0	0	0	0



JUNCTION TREE GUIDED DP - COMPUTING

$I_{a \cap b}$ table/function with sizes of maximum independent sets in subtree below a and b based on intersection with the sepset $a \cap b$

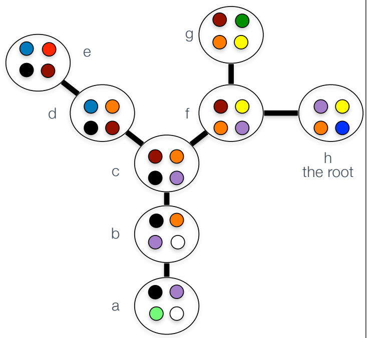


$I_{a \cap b}$ =

			s
1	0	0	1
0	1	0	2
0	0	1	1
0	0	0	1

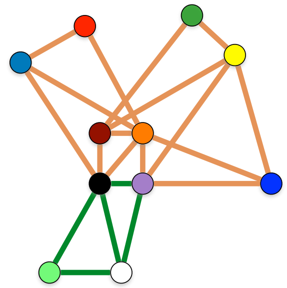
I_a =

				s
1	0	0	0	1
0	1	0	0	1
0	0	1	0	1
0	0	0	1	1
0	1	0	1	2
0	0	0	0	0



JUNCTION TREE GUIDED DP - COMPUTING

$I_{a \cap b}$ table/function with sizes of maximum independent sets in subtree below a and b based on intersection with the sepset $a \cap b$

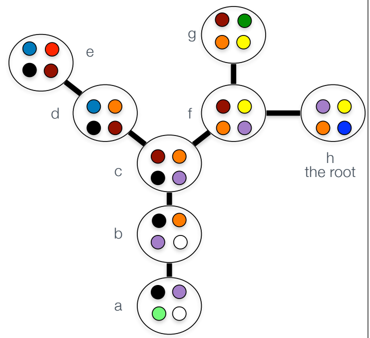


$I_{a \cap b}$ =

			s
1	0	0	1
0	1	0	2
0	0	1	1
0	0	0	1

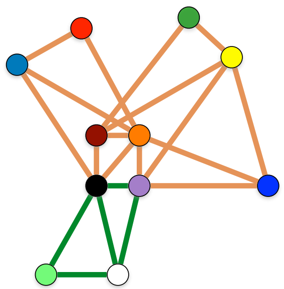
I_a =

				s
1	0	0	0	1
0	1	0	0	1
0	0	1	0	1
0	0	0	1	1
0	1	0	1	2
0	0	0	0	0



JUNCTION TREE GUIDED DP - COMPUTING

$I_{a \cap b}$ table/function with sizes of maximum independent sets in subtree below a and b based on intersection with the sepset $a \cap b$

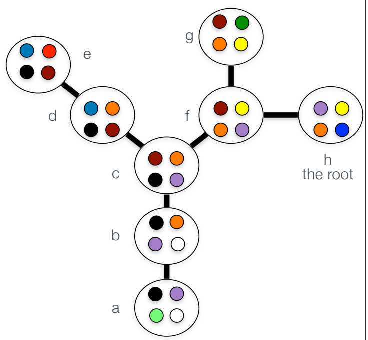


$I_{a \cap b}$ =

			s
1	0	0	1
0	1	0	2
0	0	1	1
0	0	0	1

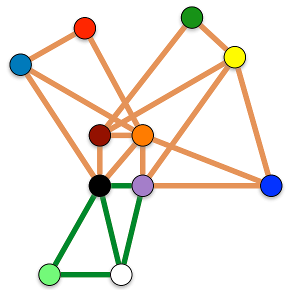
I_a =

				s
1	0	0	0	1
0	1	0	0	1
0	0	1	0	1
0	0	0	1	1
0	1	0	1	2
0	0	0	0	0



JUNCTION TREE GUIDED DP - COMPUTING

$I_{a \cap b}$ table/function with sizes of maximum independent sets in subtree below a and b based on intersection with the sepset $a \cap b$

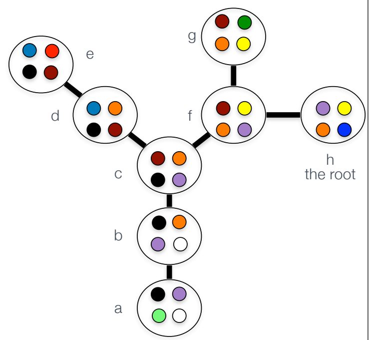


$I_{a \cap b}$ =

			s
1	0	0	1
0	1	0	2
0	0	1	1
0	0	0	1

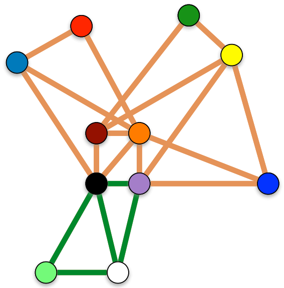
I_a =

				s
1	0	0	0	1
0	1	0	0	1
0	0	1	0	1
0	0	0	1	1
0	1	0	1	2
0	0	0	0	0



JUNCTION TREE GUIDED DP - COMPUTING

$I_{a \cap b}$ table/function with sizes of maximum independent sets in subtree below a and b based on intersection with the sepset a∩b

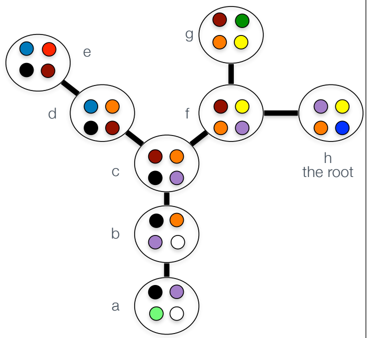


$I_{a \cap b}$ =

			s
1	0	0	1
0	1	0	2
0	0	1	1
0	0	0	1

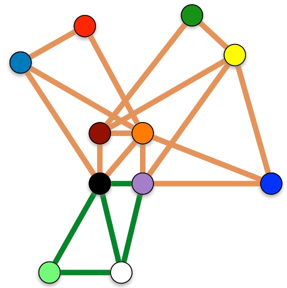
I_a =

				s
1	0	0	0	1
0	1	0	0	1
0	0	1	0	1
0	0	0	1	1
0	1	0	1	2
0	0	0	0	0



JUNCTION TREE GUIDED DP - COMPUTING

B the bag and s below t, $I_{st}(S) = \max_{S' \in B(s) : S' \cap B(t) = S} I(S')$

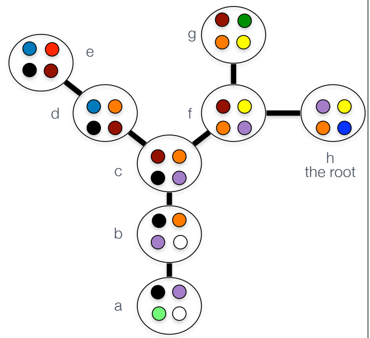


$I_{a \cap b}$ =

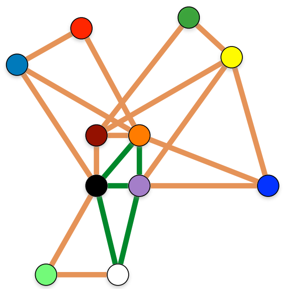
			s
1	0	0	1
0	1	0	2
0	0	1	1
0	0	0	1

I_a =

				s
1	0	0	0	1
0	1	0	0	1
0	0	1	0	1
0	0	0	1	1
0	1	0	1	2
0	0	0	0	0



JUNCTION TREE GUIDED DP - COMPUTING

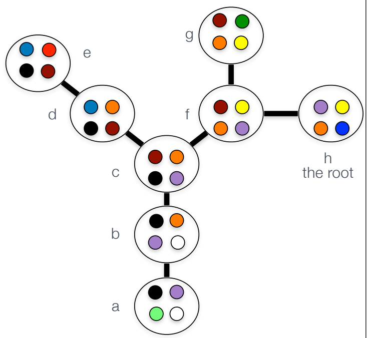


$I_{a \cap b}$ =

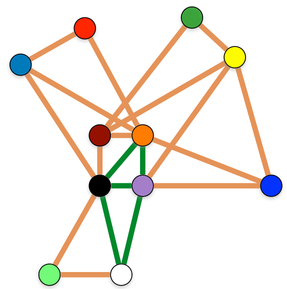
			s
1	0	0	1
0	1	0	2
0	0	1	1
0	0	0	1

I_b =

				s
1	0	0	0	1
0	1	0	0	2
0	0	1	0	1
0	0	0	1	2
0	0	1	1	2



JUNCTION TREE GUIDED DP - COMPUTING

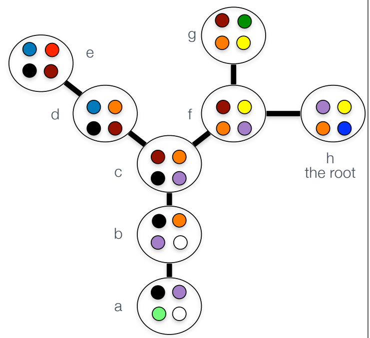


$I_{b \cap c}$ =

			s
1	0	0	1
0	0	1	2
0	0	0	1
0	1	0	2

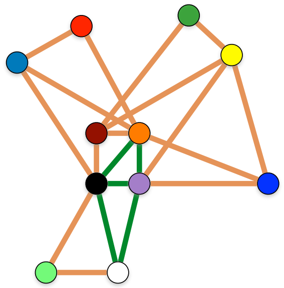
I_b =

				s
1	0	0	0	1
0	1	0	0	2
0	0	1	0	1
0	0	0	1	2
0	0	1	1	2



JUNCTION TREE GUIDED DP - COMPUTING

B the bag and s only child of t, $I_t(S) = \begin{cases} I_{st}(S \cap B(s)) + |S \setminus B(s)| & \text{if } S \text{ independent set } \subseteq B(t) \\ -\infty & \text{otherwise} \end{cases}$

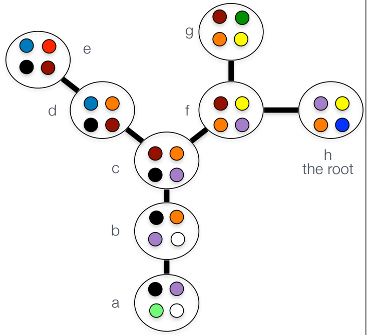


$I_{b \cap c} =$

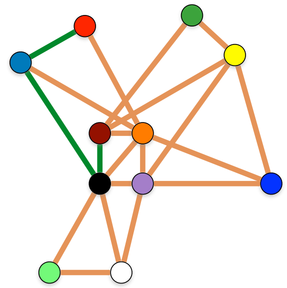
●	●	●	s
1	0	0	1
0	0	1	2
0	0	0	1
0	1	0	2

$I_b =$

●	●	○	●	s
1	0	0	0	1
0	1	0	0	2
0	0	1	0	1
0	0	0	1	2
0	0	1	1	2

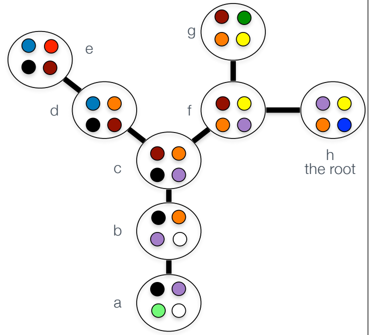


JUNCTION TREE GUIDED DP - COMPUTING



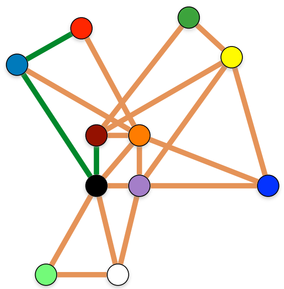
$I_e =$

●	●	○	●	s
0	1	1	0	2
0	1	0	1	2
1	0	0	1	2
1	0	0	0	1
0	1	0	0	1
⋮	⋮	⋯	⋯	⋮



JUNCTION TREE GUIDED DP - COMPUTING

$$B \text{ the bag and } s \text{ below } t, \; I_{st}(S) = \max_{S' \in B(s): S' \cap B(t) = S} I(S')$$

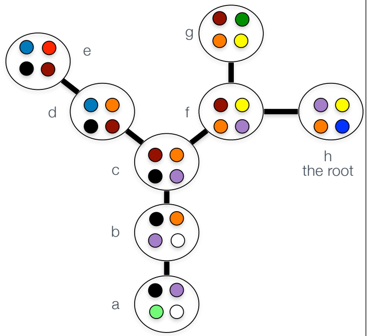


$I_{end} =$

●	●	○	s
0	1	1	2
1	0	0	2
0	1	0	2
0	0	1	1
0	0	0	1

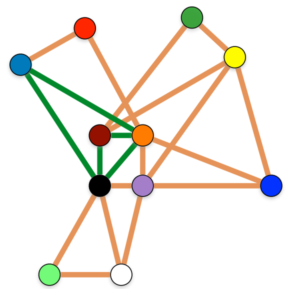
$I_e =$

●	●	○	●	s
0	1	1	0	2
0	1	0	1	2
1	0	0	1	2
1	0	0	0	1
0	1	0	0	1
⋮	⋮	⋯	⋯	⋮



JUNCTION TREE GUIDED DP - COMPUTING

B the bag and s only child of t, $I_t(S) = \begin{cases} I_{st}(S \cap B(s)) + |S \setminus B(s)| & \text{if } S \text{ independent set } \subseteq B(t) \\ -\infty & \text{otherwise} \end{cases}$

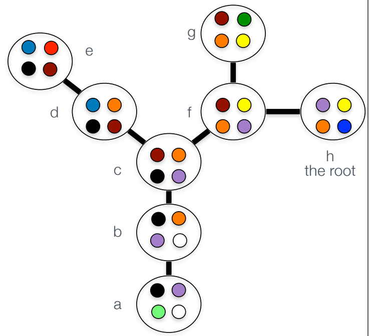


$I_{end} =$

●	●	○	s
0	1	1	2
1	0	0	2
0	1	0	2
0	0	1	1
0	0	0	1

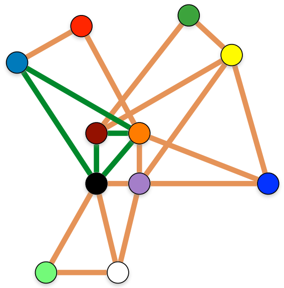
$I_d =$

●	●	○	●	s
1	0	0	0	2
0	1	0	0	2
0	0	1	0	1
0	0	0	1	2
0	1	1	0	2
0	0	0	0	1



JUNCTION TREE GUIDED DP - COMPUTING

B the bag and s below t, $I_{st}(S) = \max_{S' \in B(s): S \cap B(s) = S'} I(S')$

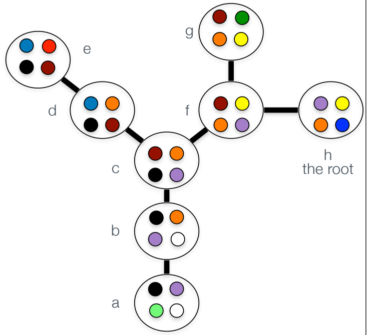


I_{cnd} =

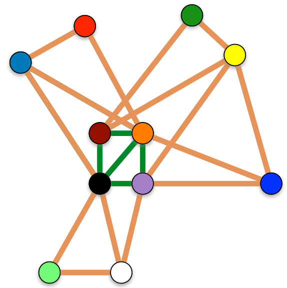
			s
1	0	0	2
0	1	0	2
0	0	1	2
0	0	0	1

I_d =

				s
1	0	0	0	2
0	1	0	0	2
0	0	1	0	1
0	0	0	1	2
0	1	1	0	2
0	0	0	0	1



JUNCTION TREE GUIDED DP - COMPUTING



I_{cnd} =

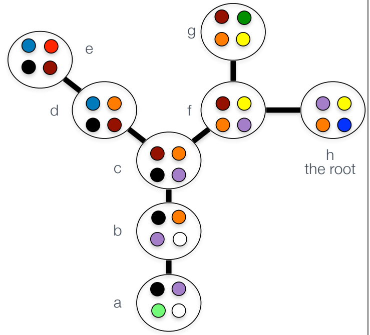
			s
1	0	0	2
0	1	0	2
0	0	1	2
0	0	0	1

I_{bnc} =

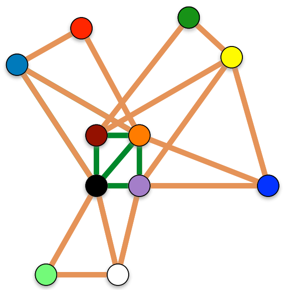
			s
1	0	0	1
0	0	1	2
0	0	0	1
0	1	0	2

I_c =

				s
1	0	0	0	2
0	1	0	0	3
0	0	1	0	3
0	1	1	0	4
0	0	0	1	3



JUNCTION TREE GUIDED DP - COMPUTING



I_{cnd} =

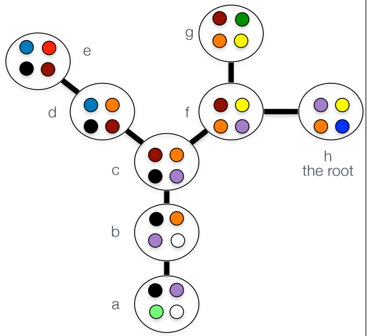
			s
1	0	0	2
0	1	0	2
0	0	1	2
0	0	0	1

I_{bnc} =

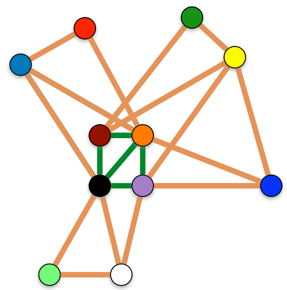
			s
1	0	0	1
0	0	1	2
0	0	0	1
0	1	0	2

I_c =

				s
1	0	0	0	2
0	1	0	0	3
0	0	1	0	3
0	1	1	0	4
0	0	0	1	3



JUNCTION TREE GUIDED DP - COMPUTING



I_{cnd} =

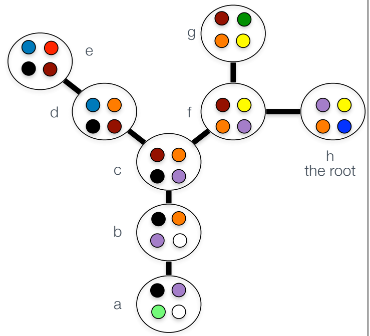
			s
1	0	0	2
0	1	0	2
0	0	1	2
0	0	0	1

I_{bnc} =

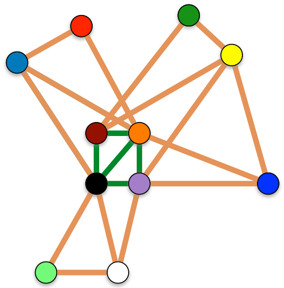
			s
1	0	0	1
0	0	1	2
0	0	0	1
0	1	0	2

I_c =

				s
1	0	0	0	2
0	1	0	0	3
0	0	1	0	3
0	1	1	0	4
0	0	0	1	3



JUNCTION TREE GUIDED DP - COMPUTING



I_{cnd} =

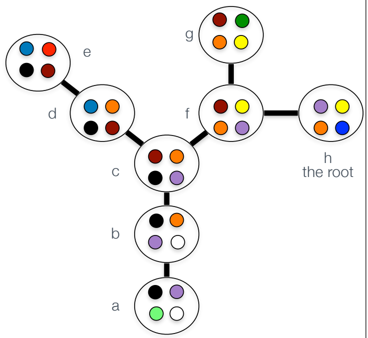
●	●	●	s
1	0	0	2
0	1	0	2
0	0	1	2
0	0	0	1

I_{bnc} =

●	●	●	s
1	0	0	1
0	0	1	2
0	0	0	1
0	1	0	2

I_c =

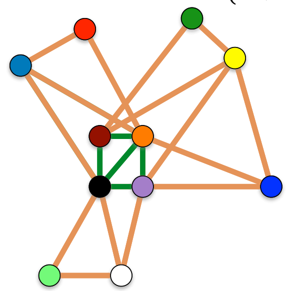
●	●	●	●	s
1	0	0	0	2
0	1	0	0	3
0	0	1	0	3
0	1	1	0	4
0	0	0	1	3



JUNCTION TREE GUIDED DP - COMPUTING

s, s' the 2 only children of t

$$I_t(S) = \begin{cases} -\infty \text{ if } S \text{ not ind. set and otherwise the case below} \\ I_{st}(S \cap B(s)) + I_{s't}(S \cap B(s')) + |S \setminus (B(s) \cup B(s'))| - |S \cap B(s) \cap B(S')| \end{cases}$$

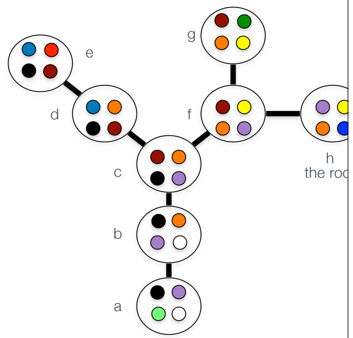


I_{bnc} =

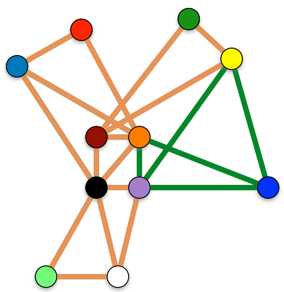
●	●	●	s
1	0	0	1
0	0	1	2
0	0	0	1
0	1	0	2

I_c =

●	●	●	●	s
1	0	0	0	2
0	1	0	0	3
0	0	1	0	3
0	1	1	0	4
0	0	0	1	3

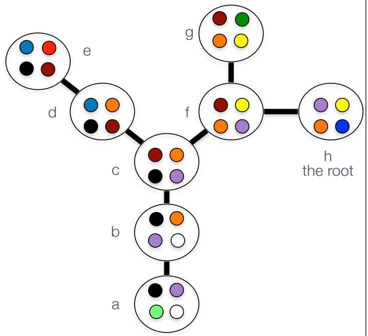


JUNCTION TREE GUIDED DP - COMPUTING



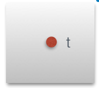
I_h =

●	●	●	●	s
1	0	0	0	3
0	1	0	0	4
0	0	1	0	3
0	0	0	1	4
1	0	0	1	4



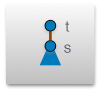
SUMMARY - IND. SET ALGORITHM

* Given junction-tree (T,B) for G. Make T binary, let r be the root of T
  * Visit the vertices and edge of T from leaves to root



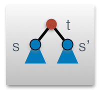
* when at leaf t

$$I_t(S) = \begin{cases} -\infty \text{ if } S \text{ not ind. set} \\ |S| \text{ if } S \text{ is ind. set} \end{cases}$$



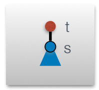
* at edge st

$$I_{st}(S) = \max_{S' \in B(s): S \cap B(s) = S'} I(S')$$



* if two children s and s'

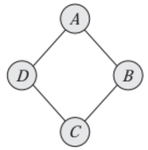
$$I_t(S) = \begin{cases} -\infty \text{ if } S \text{ not ind. set and otherwise the case below} \\ I_{st}(S \cap B(s)) + I_{s't}(S \cap B(s')) + |S \setminus (B(s) \cup B(s'))| - |S \cap B(s) \cap B(S')| \end{cases}$$



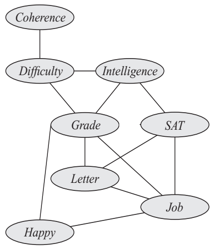
* if single child s

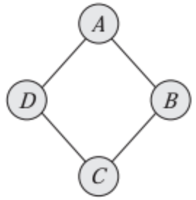
$$I_t(S) = \begin{cases} -\infty \text{ if } S \text{ not ind. set} \\ I(S \cap B(S)) + |S \setminus S'| \text{ if } S \text{ is ind. set} \end{cases}$$

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





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)

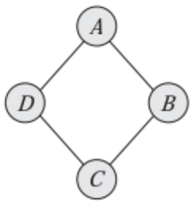
Factors – misconception example

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

PROBABILISTIC INTERPRETATION

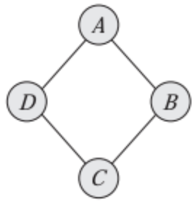
$\phi_1(A,B)$				$\phi_2(B,C)$				$\phi_3(C,D)$				$\phi_4(D,A)$		
a^0	b^0	30		b^0	c^0	100		c^0	d^0	1		d^0	a^0	100
a^0	b^1	5		b^0	c^1	1		c^0	d^1	100		d^0	a^1	1
a^1	b^0	1		b^1	c^0	1		c^1	d^0	100		d^1	a^0	1
a^1	b^1	10		b^1	c^1	100		c^1	d^1	1		d^1	a^1	100

(a) (b) (c) (d)



Assignment				Unnormalized	Normalized
a^1	b^0	c^0	d^1	$1,000,000$	0.14
a^1	b^0	c^1	d^0	100	$1.4 \cdot 10^{-5}$
a^1	b^0	c^1	d^1	100	$1.4 \cdot 10^{-5}$
a^1	b^1	c^0	d^0	10	$1.4 \cdot 10^{-6}$
a^1	b^1	c^0	d^1	$100,000$	0.014
a^1	b^1	c^1	d^0	$100,000$	0.014
a^1	b^1	c^1	d^1	$100,000$	0.014

UGMS



$\phi_1(A,B)$				$\phi_2(B,C)$				$\phi_3(C,D)$				$\phi_4(D,A)$		
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

-

$$\begin{aligned}\phi_1(A=1,B=1)\phi_2(B=1,C=0)\phi_3(C=0,D=1)\phi_4(D=1,A=1) &\\ = 10\cdot 1\cdot 100\cdot 100 &\\ = 100000 &\end{aligned}$$

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

A CONCRETE FACTOR PRODUCT

a^1	b^1	0.5
a^1	b^2	0.8
a^2	b^1	0.1
a^2	b^2	0
a^3	b^1	0.3
a^3	b^2	0.9

b^1	c^1	0.5
b^1	c^2	0.7
b^2	c^1	0.1
b^2	c^2	0.2

a^1	b^1	c^1	$0.5 \cdot 0.5 = 0.25$
a^1	b^1	c^2	$0.5 \cdot 0.7 = 0.35$
a^1	b^2	c^1	$0.8 \cdot 0.1 = 0.08$
a^1	b^2	c^2	$0.8 \cdot 0.2 = 0.16$
a^2	b^1	c^1	$0.1 \cdot 0.5 = 0.05$
a^2	b^1	c^2	$0.1 \cdot 0.7 = 0.07$
a^2	b^2	c^1	$0 \cdot 0.1 = 0$
a^2	b^2	c^2	$0 \cdot 0.2 = 0$
a^3	b^1	c^1	$0.3 \cdot 0.5 = 0.15$
a^3	b^1	c^2	$0.3 \cdot 0.7 = 0.21$
a^3	b^2	c^1	$0.9 \cdot 0.1 = 0.09$
a^3	b^2	c^2	$0.9 \cdot 0.2 = 0.18$

A CONCRETE FACTOR PRODUCT

MARGINALIZE

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

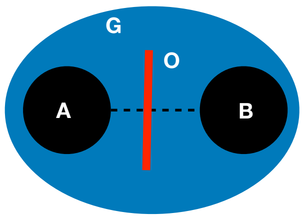
- The denominator contains a marginal likelihood
- Summing out V binary hidden variables – $O(2^V)$
- K values – $O(K^V)$

EQUIVALENCE I-MAP AND FACTORIZATION

- For positive distributions p (i.e., ∀y, p(y)>0),

  I(G) ⊆ I(p) ⇔ p can be expressed as a normalised product over factors of G (as below)

$$p(\mathbf{y}|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \prod_{c \in \mathcal{C}} \psi_c(\mathbf{y}_c|\boldsymbol{\theta}_c)$$



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

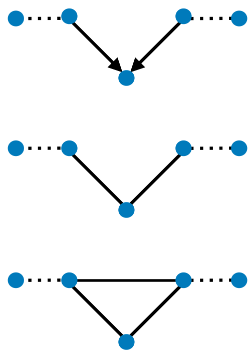
★ In a graph G,

⇕

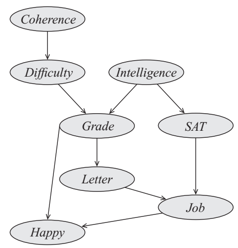
A is separated from B given O

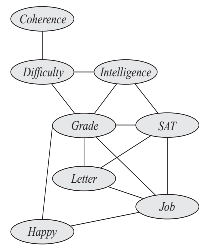
DEF I-MAP

- G is an I-map for p if all independence relation in G hold for p, i.e., I(G)⊆I(p)
- Moralize 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

THE END