Royal Institute of Technology

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

Lecture 7

LAST LECTURE

- DP for trees
  - independent set
  - marginalization
- UGMs
- Converting a DGM to a UGM
- k-trees, tree-width, and junction-trees

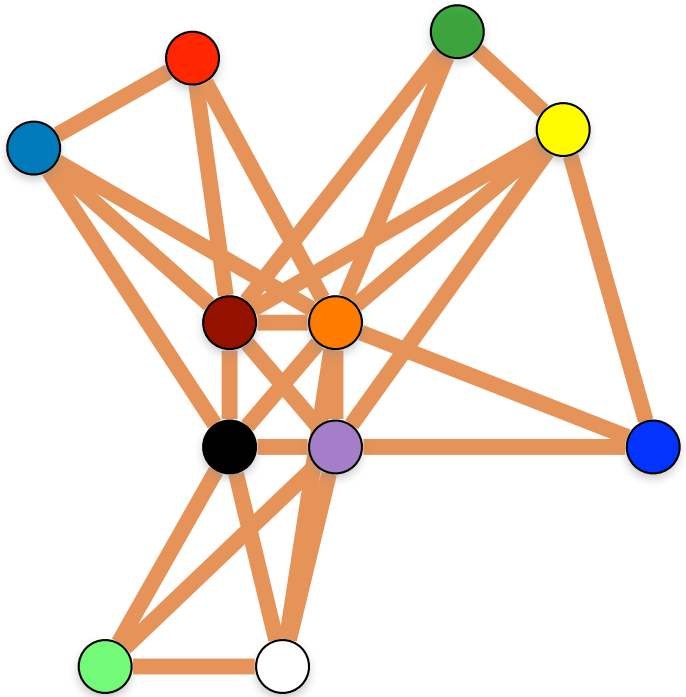
THIS LECTURE

- k-trees and junction-trees
- Factor products
- Summing out
- Elimination order & complexity of summing out
- Junction tree guided DP for independent set
- Possibly: marginalization for each bag

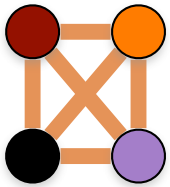
EXACT ALGORITHMS FOR GRAPHICAL MODELS

- 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.

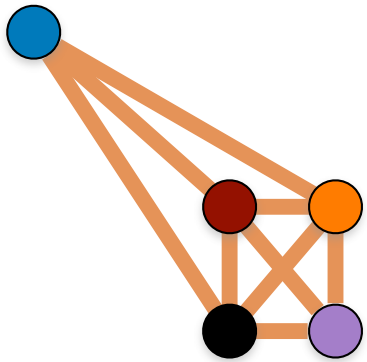
K-TREES (HERE K=3)



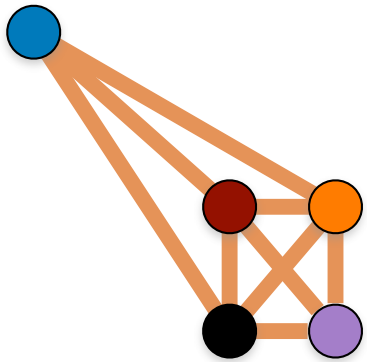
3-TREE

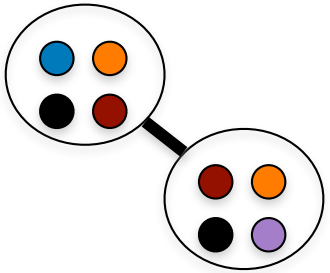


3-TREE

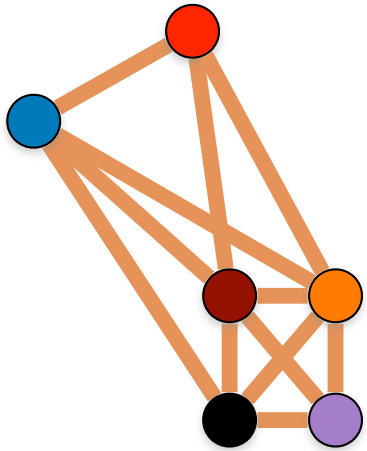


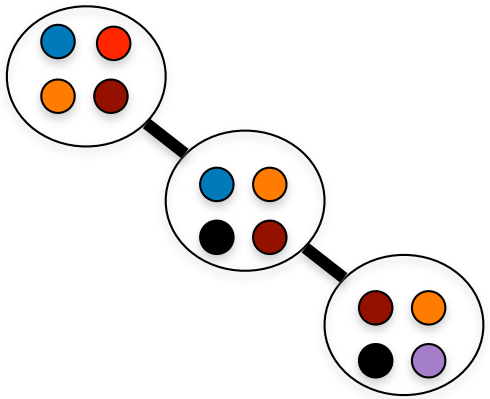
3-TREE



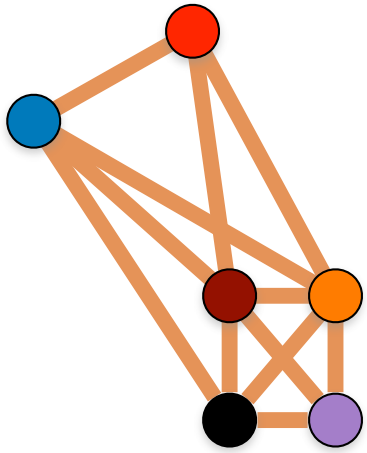


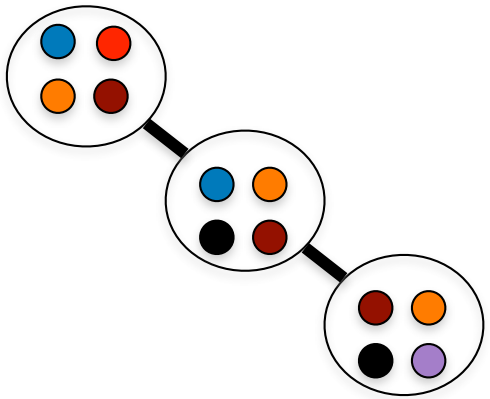
3-TREE



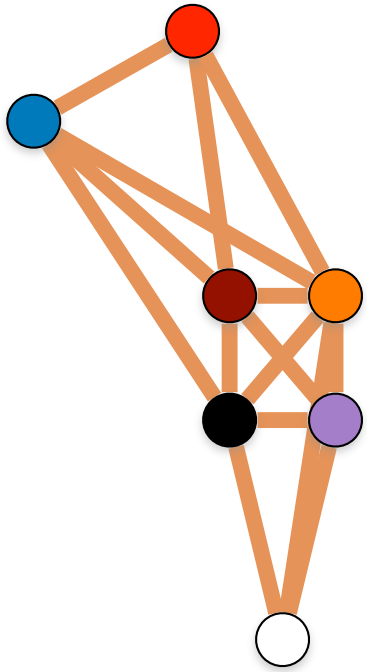


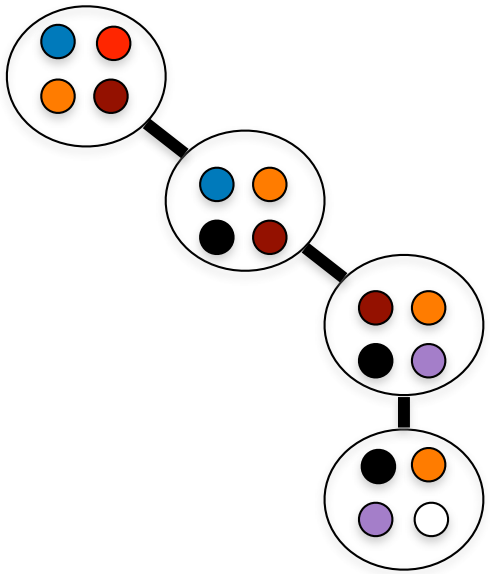
3-TREE



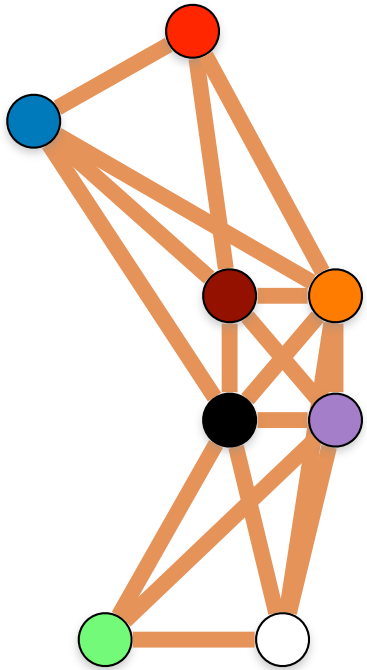


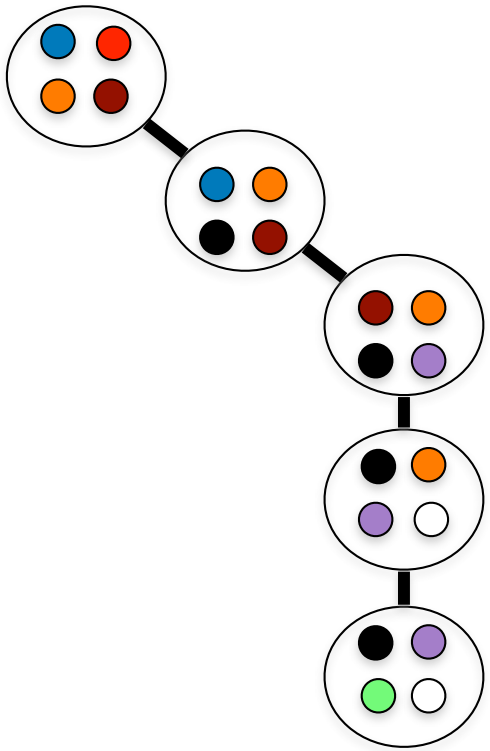
3-TREE



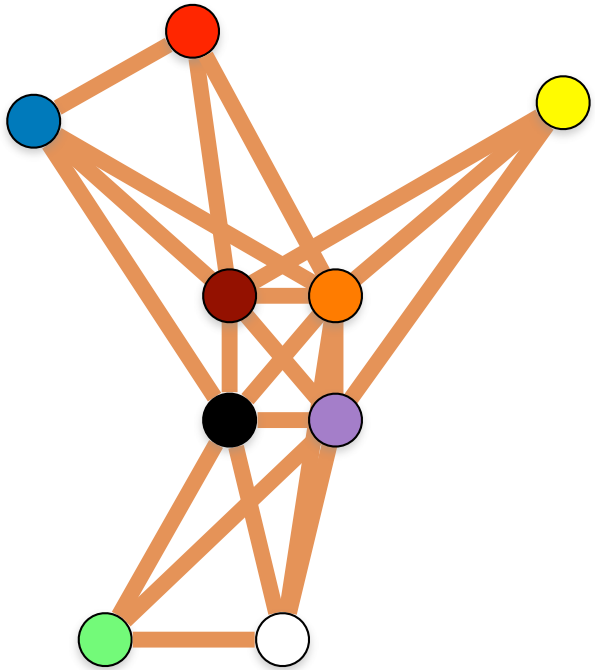


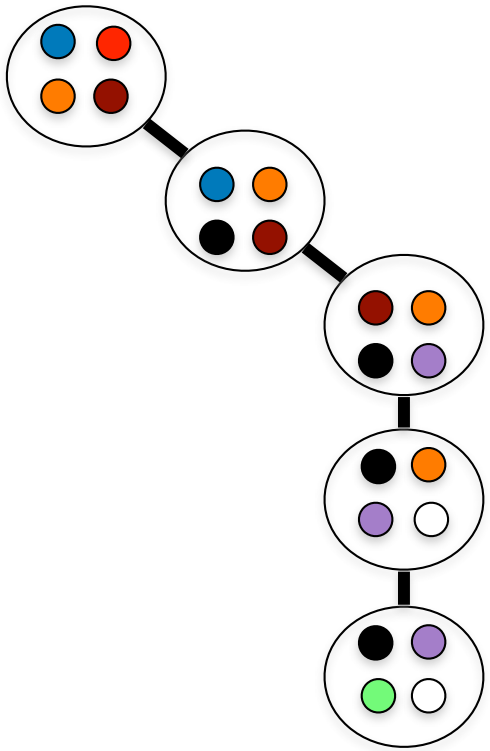
3-TREE



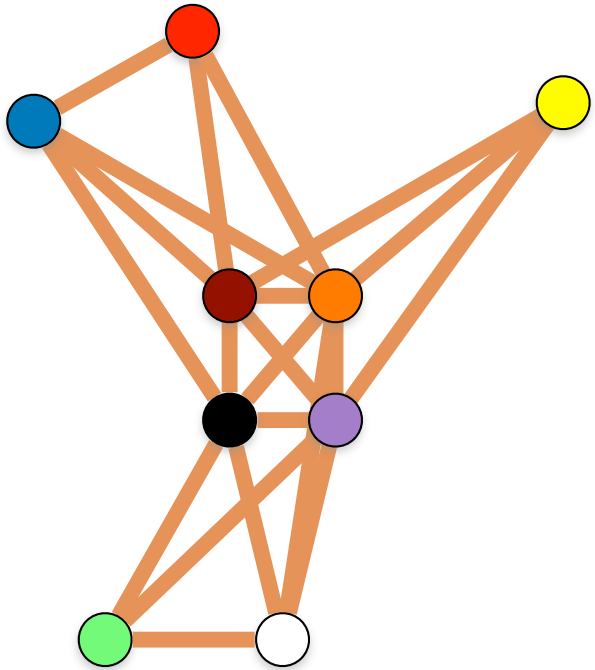


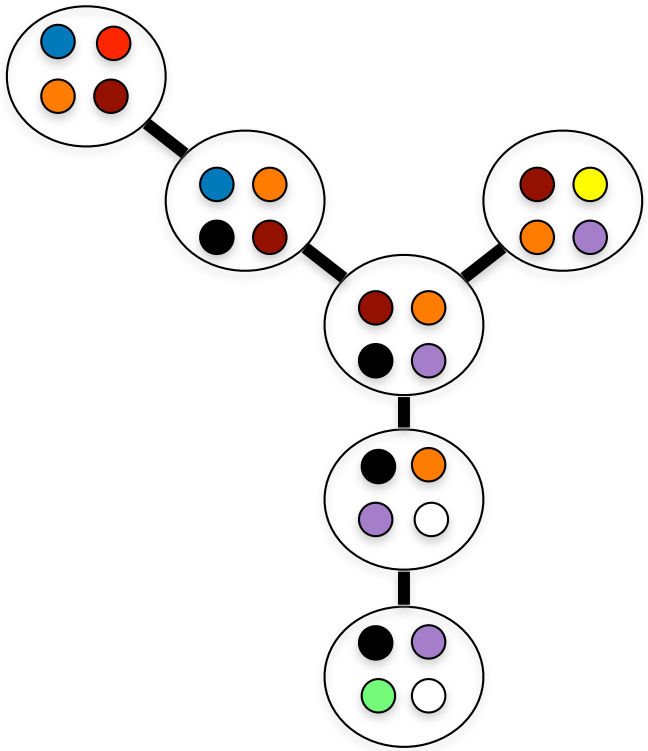
3-TREE



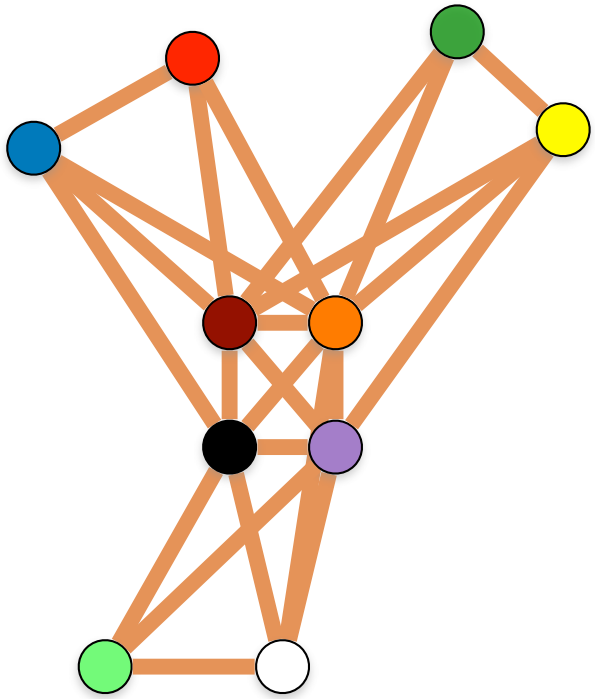


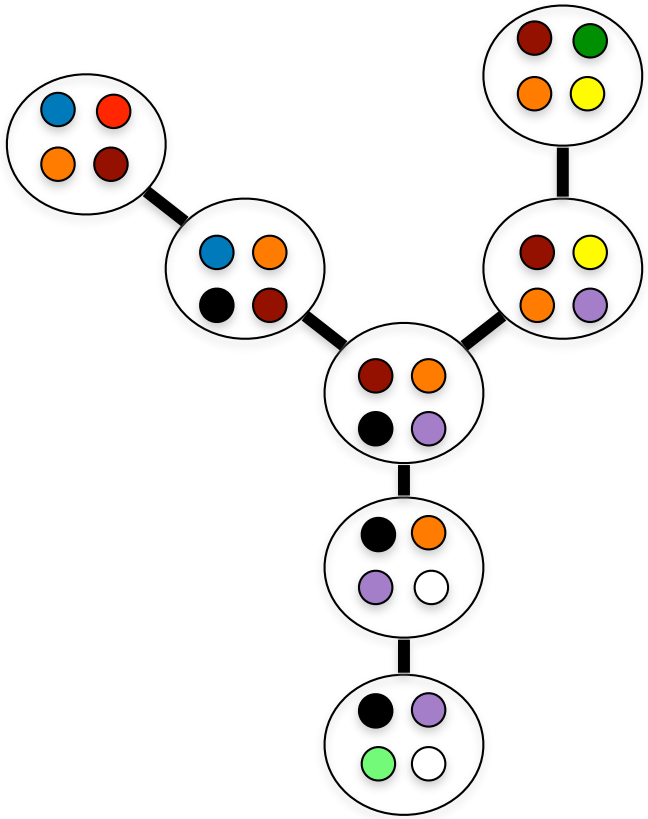
3-TREE



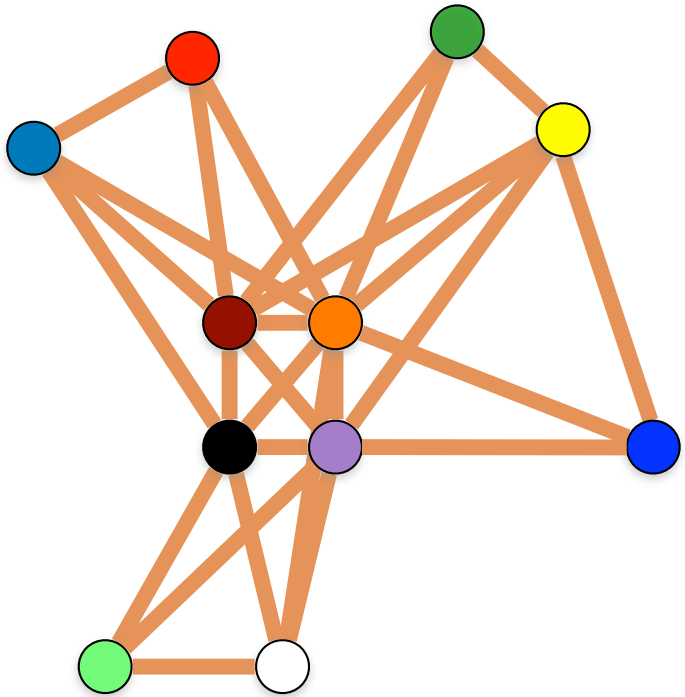


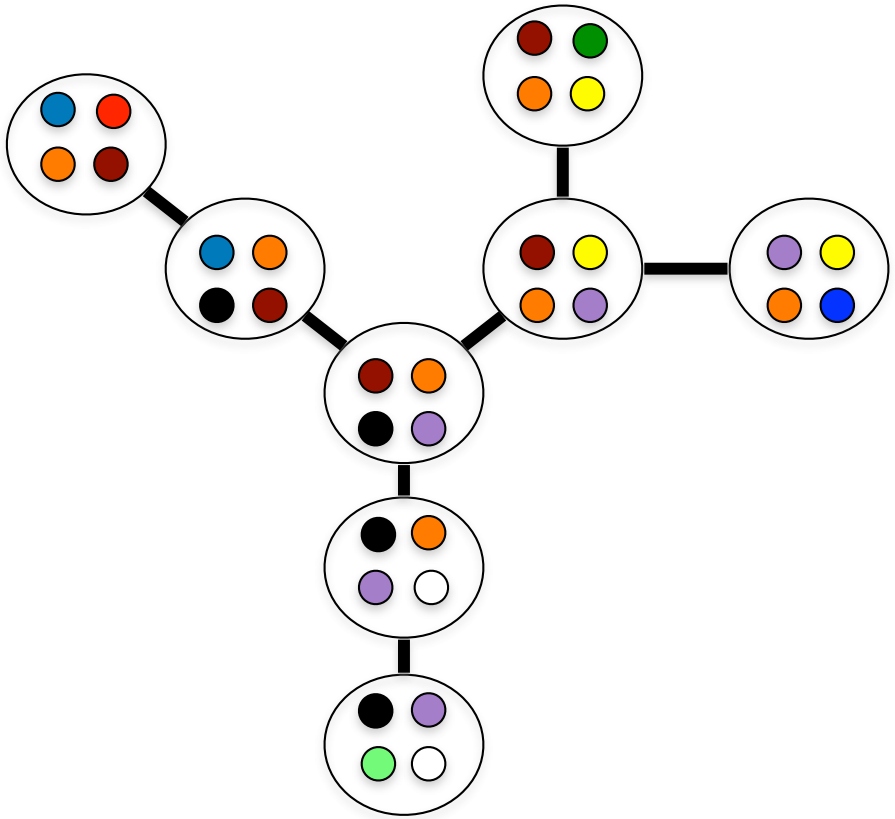
3-TREE





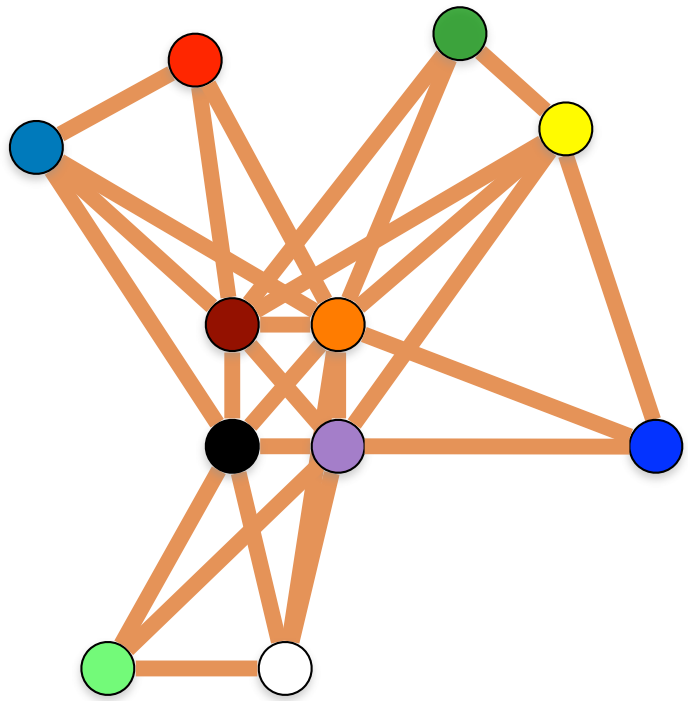
3-TREE



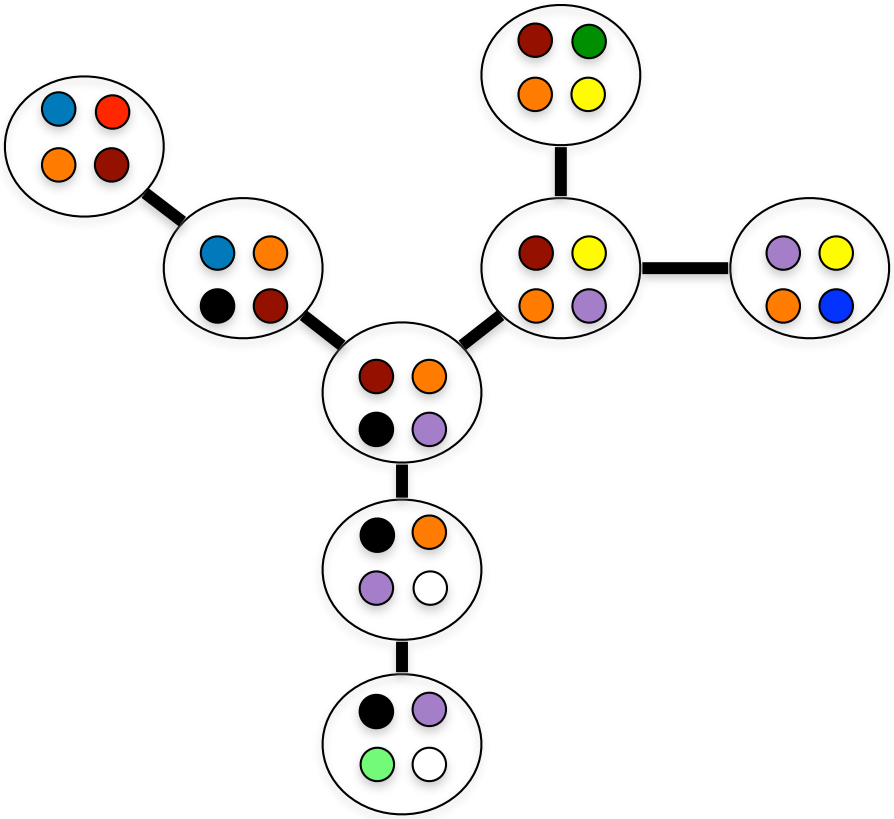


JUNCTION TREE

Graph G



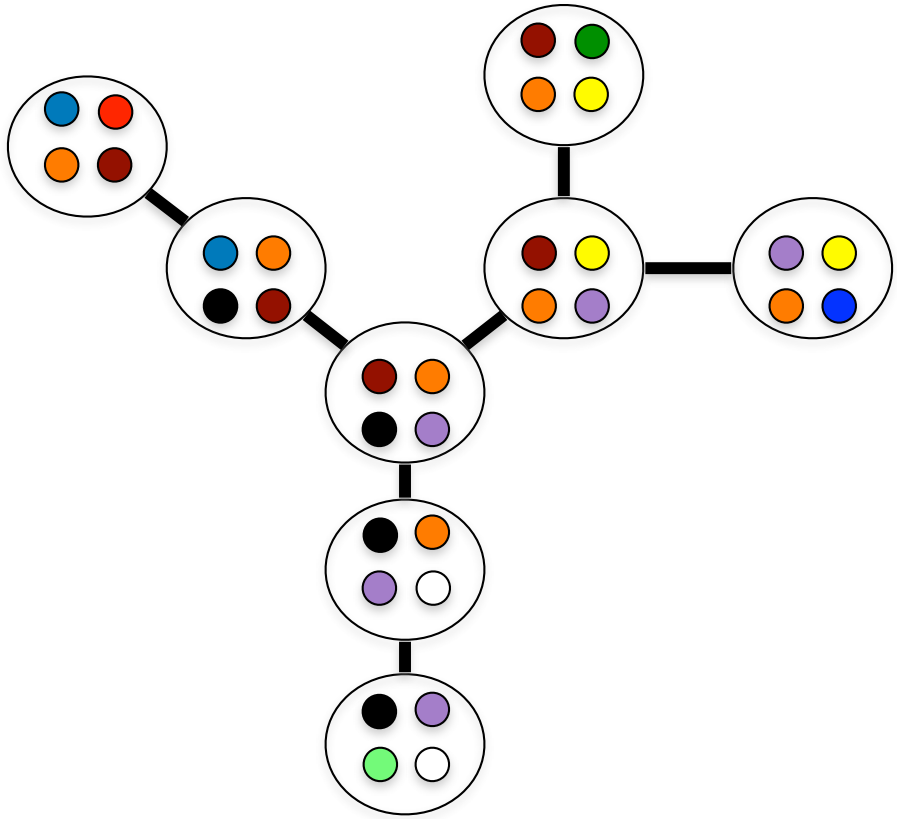
T,B Junction tree



JUNCTION TREE G

Definition Junction tree

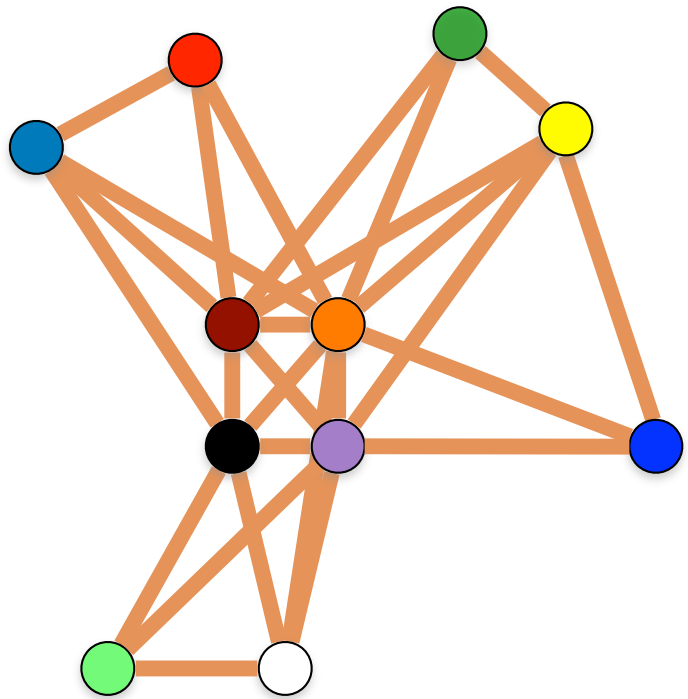
T,B Junction tree



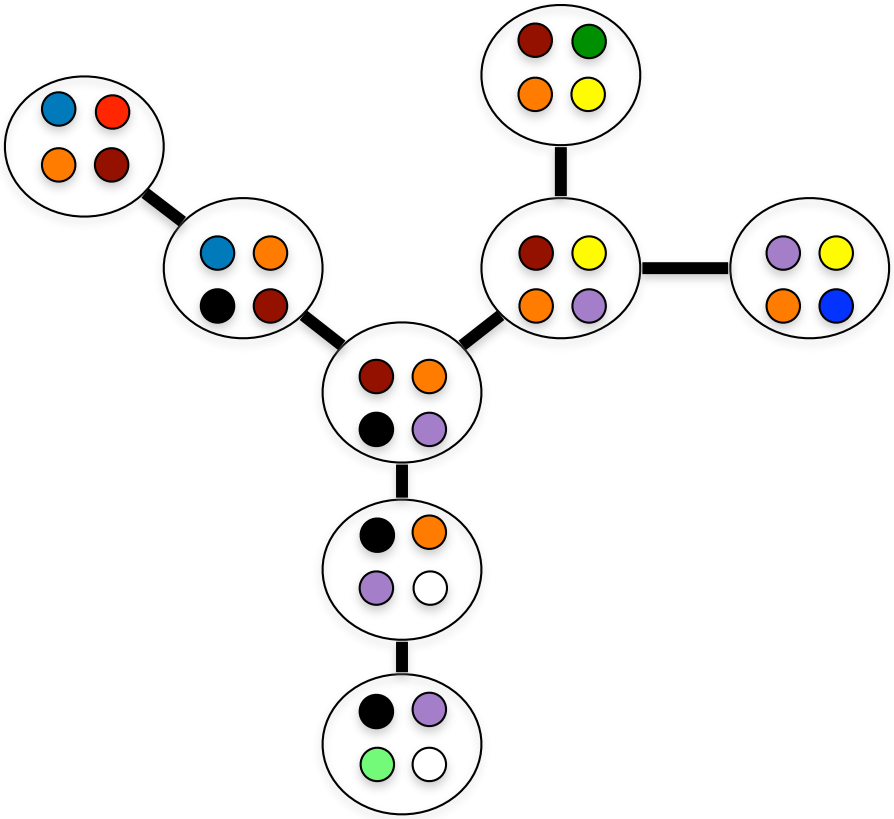
- Each vertex of G in some bag

JUNCTION TREE G

Graph G



T,B Junction tree

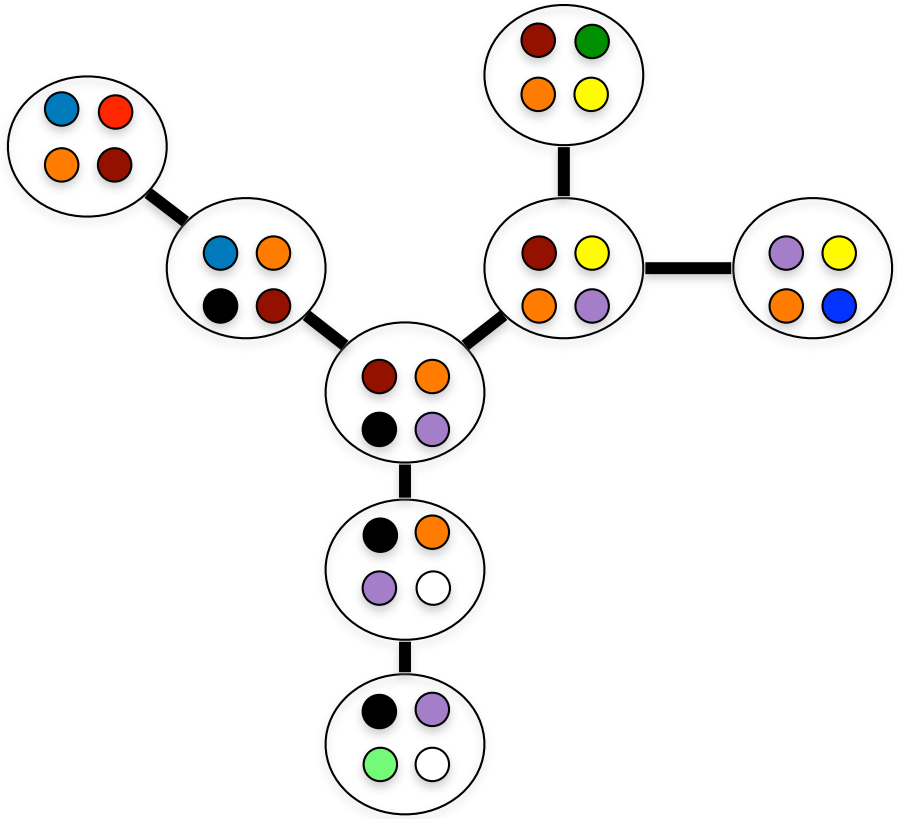


DEF JUNCTION TREE

Definition Junction tree

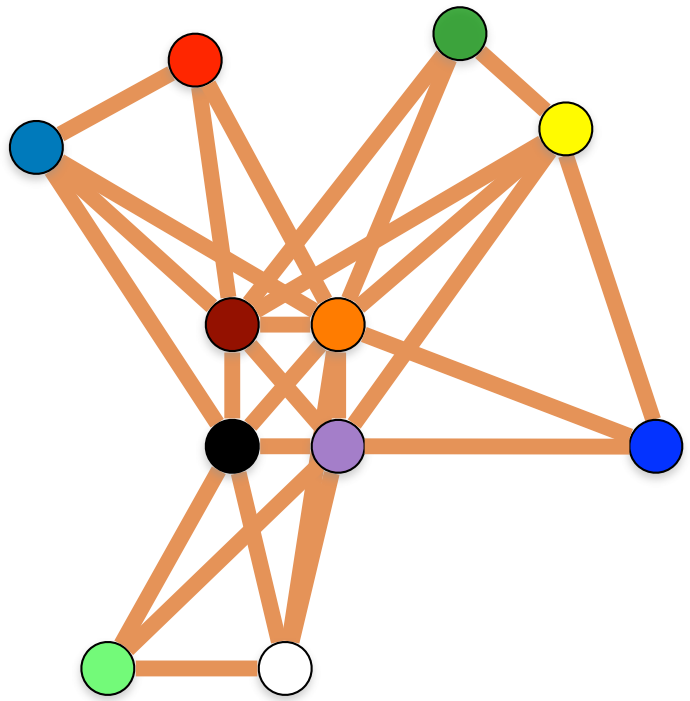
T,B Junction tree

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

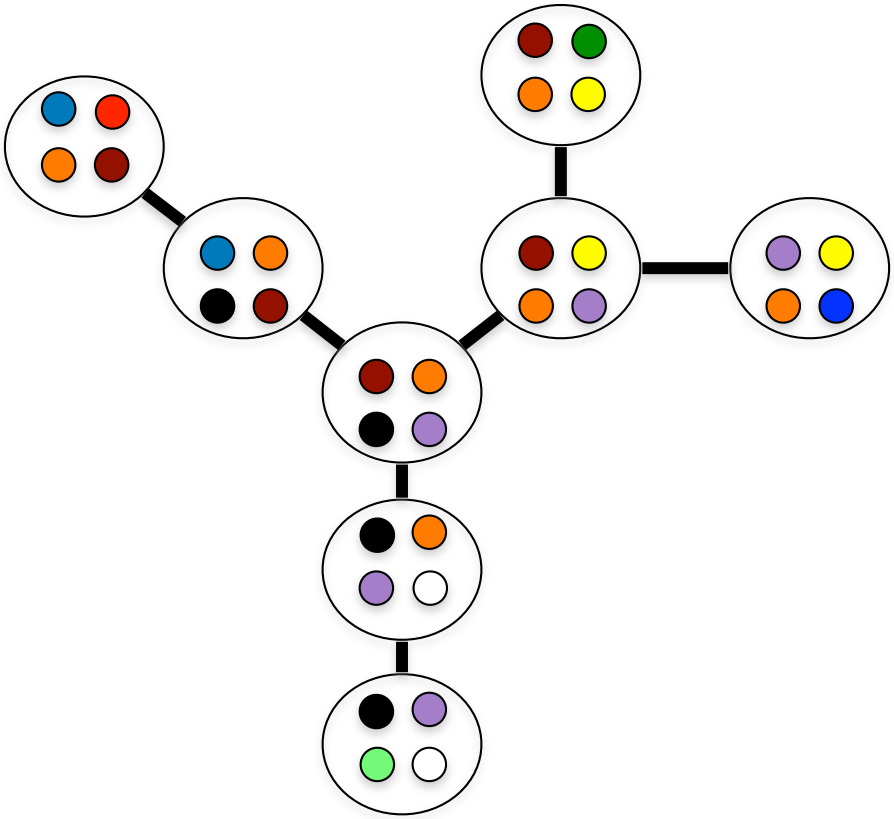


VERTEX & EDGE CONDITIONS

Graph G



T,B Junction tree

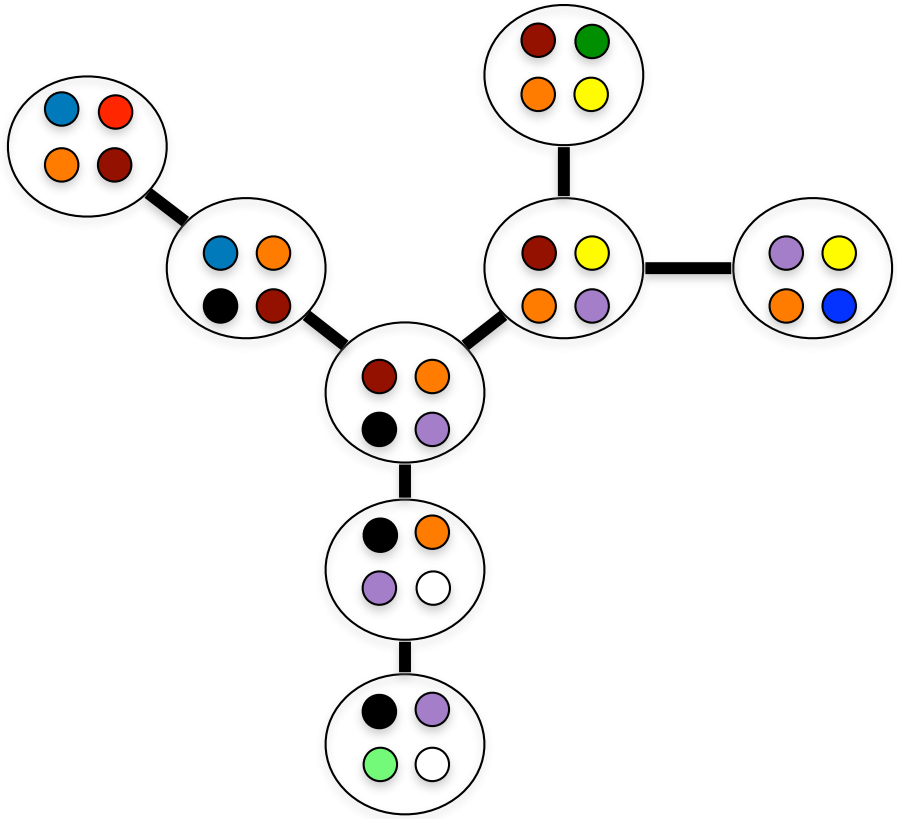


RUNNING INTERSECTION

Definition Junction tree

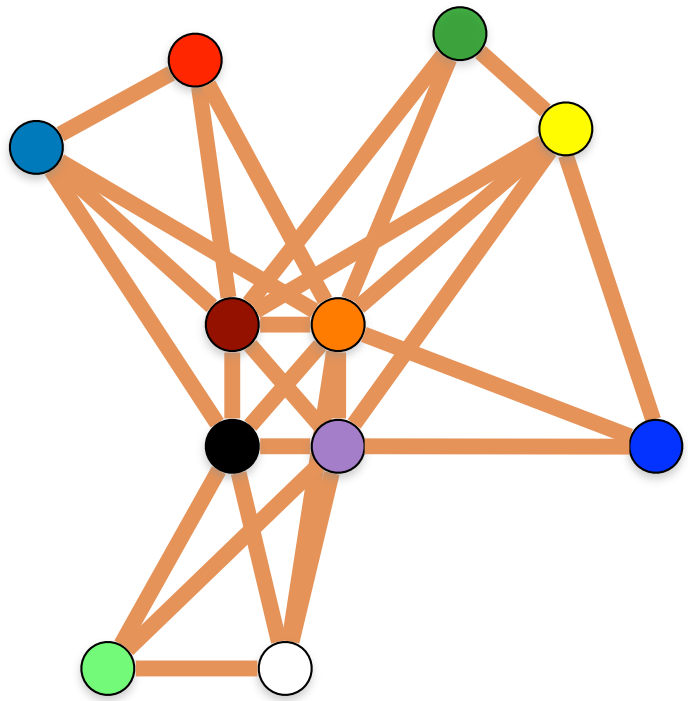
T,B Junction tree

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

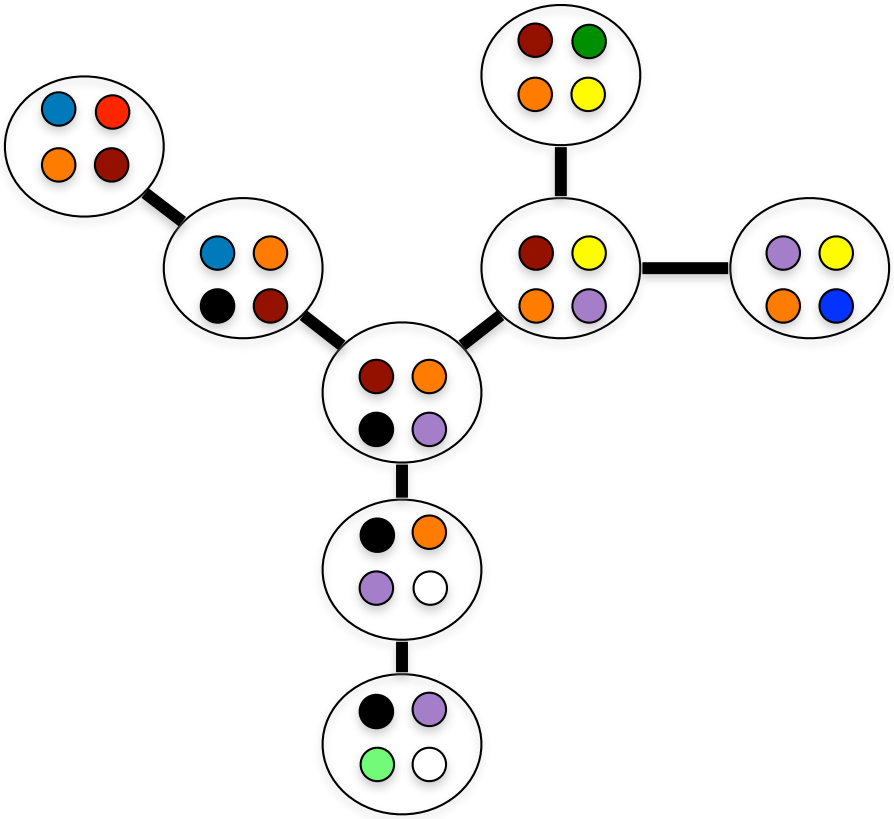


JUNCTION TREE

Graph G



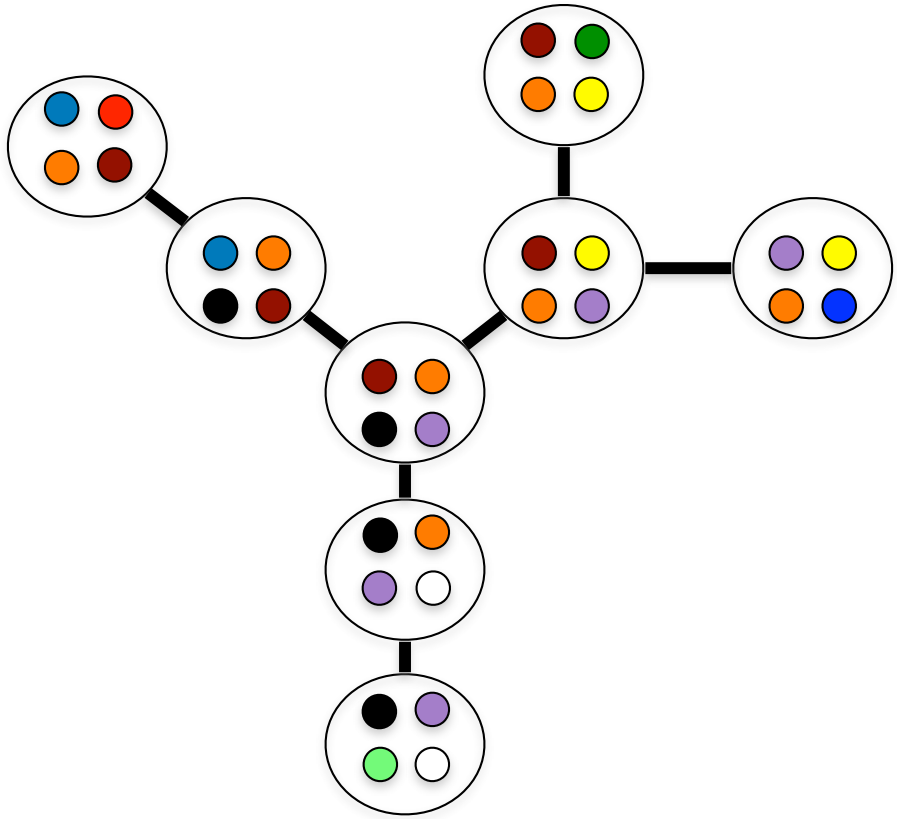
T,B Junction tree



(TREE) WIDTH - CLIQUE SIZE

Width of junction tree

T,B Junction tree

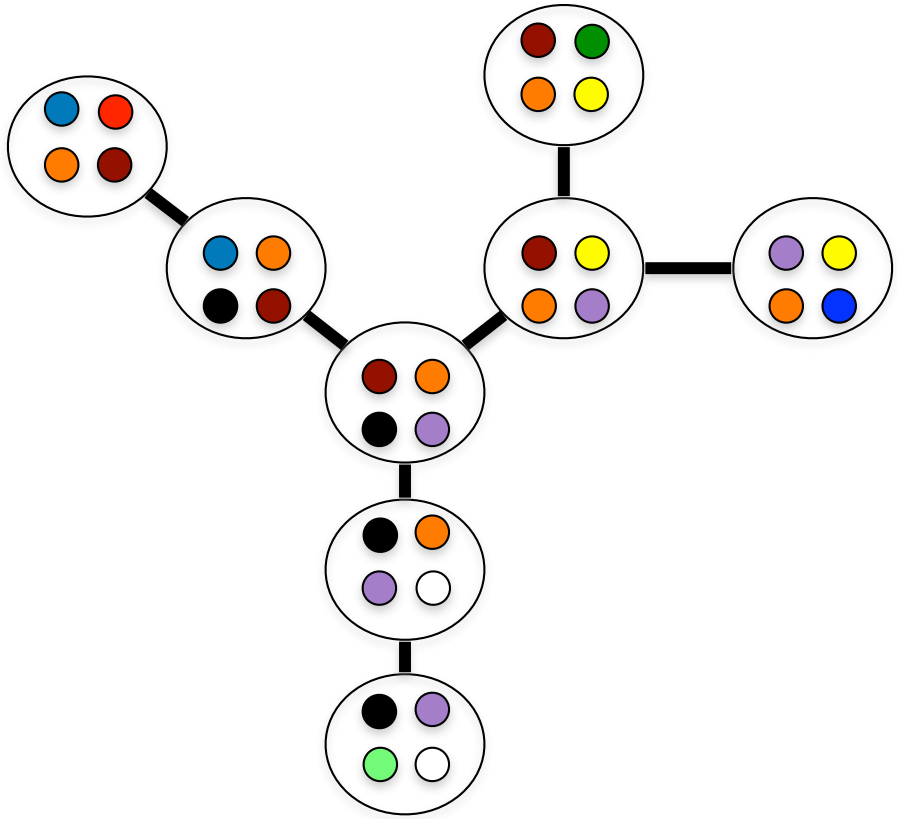


- (Size of largest bag)- 1

JUNCTION TREE - 2 IMPORTANT PROPERTIES

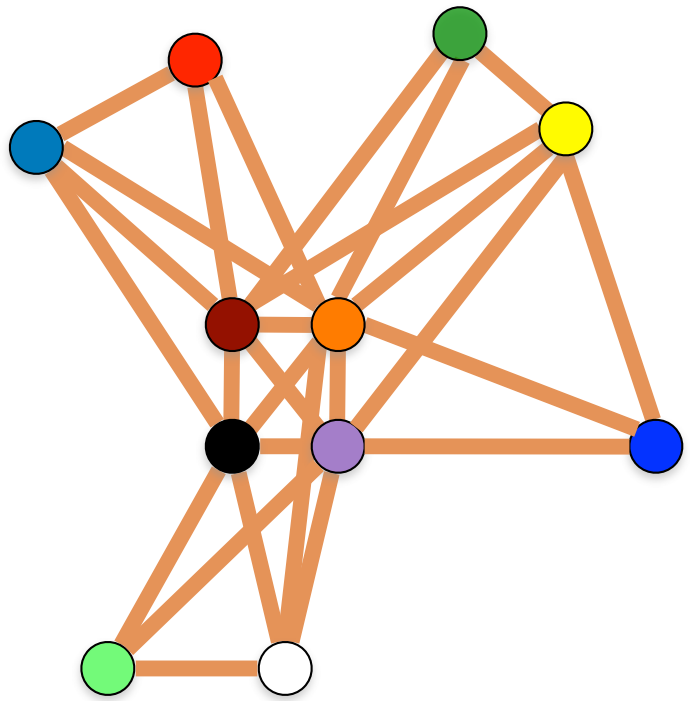
- Every clique can be found in some bag
- The intersection of 2 neighboring bags is a separator

T,B Junction tree

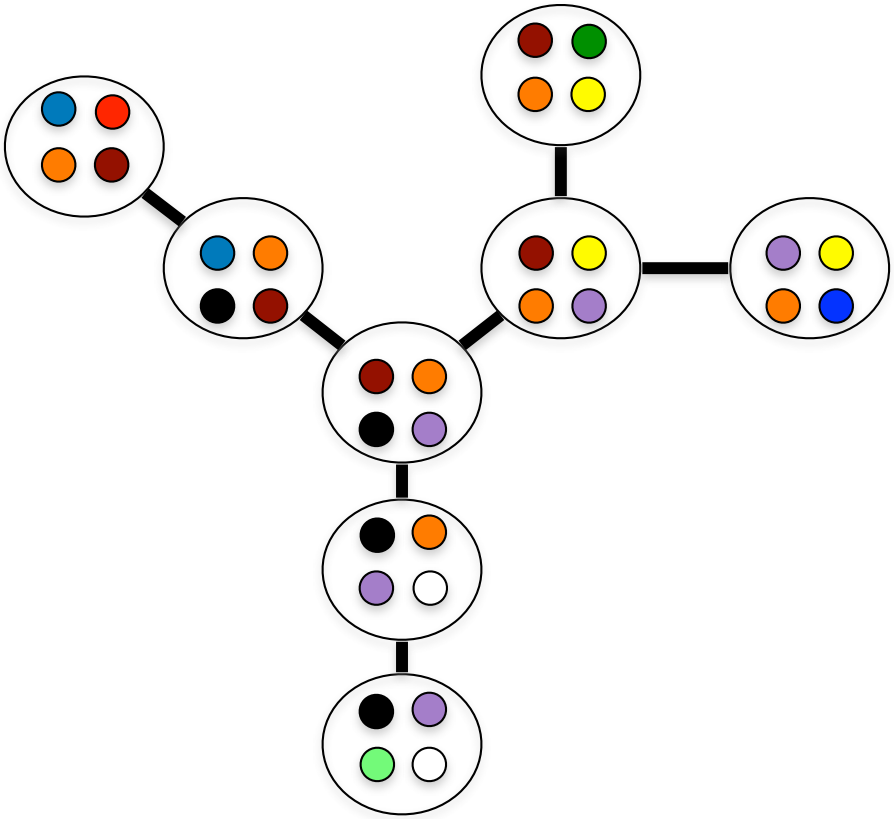


CLIQUES AND SEPARATORS IN G

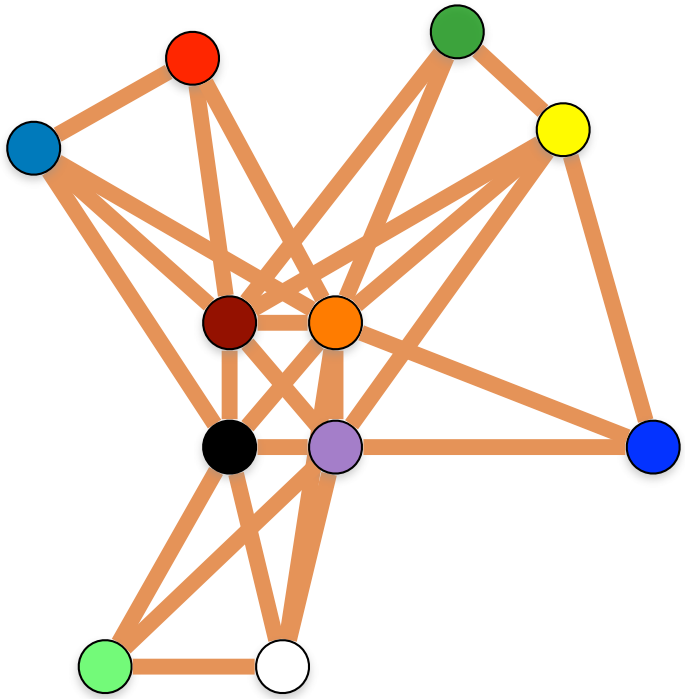
Graph G



T,B Junction tree

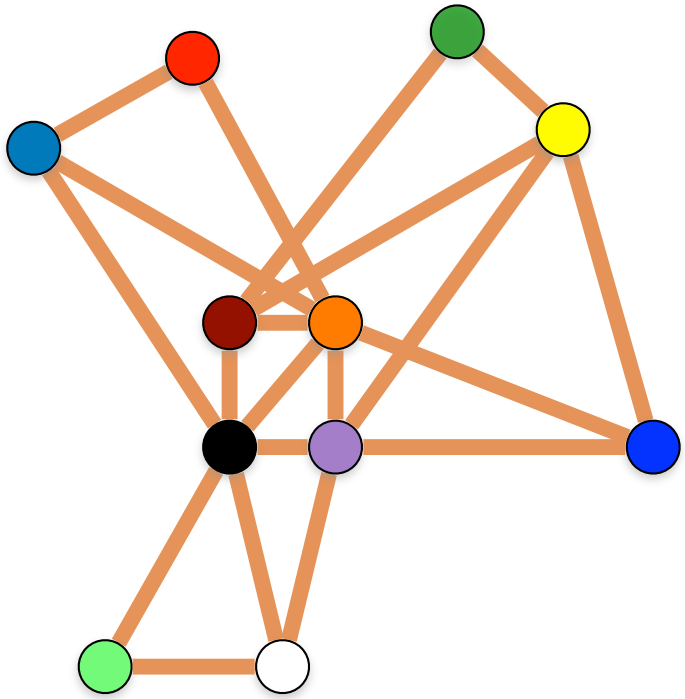


PARTIAL K-TREE



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

PARTIAL K-TREE

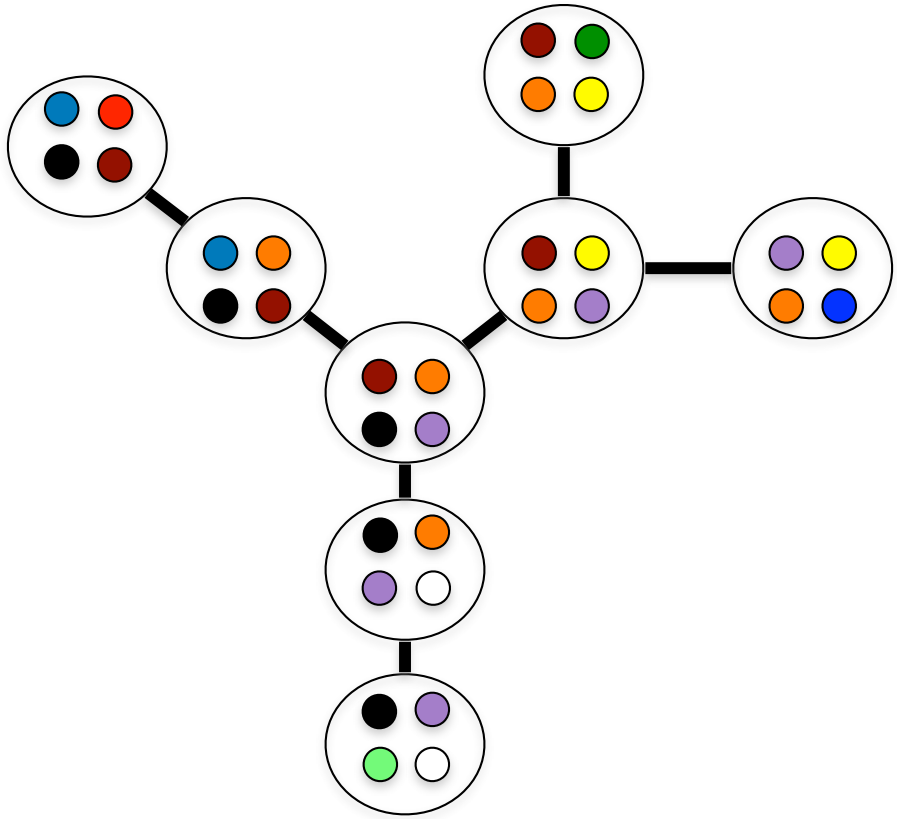


- Removing edges from k-tree gives partial k-tree

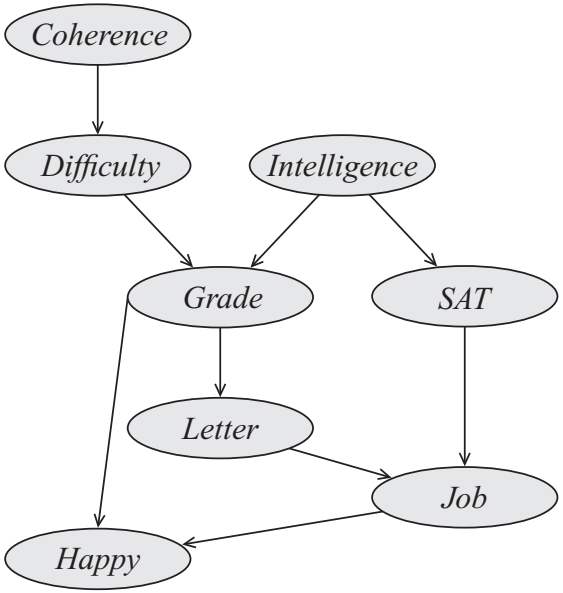
JUNCTION TREE REMAINS JUNCTION TREE WHEN EDGES DELETED

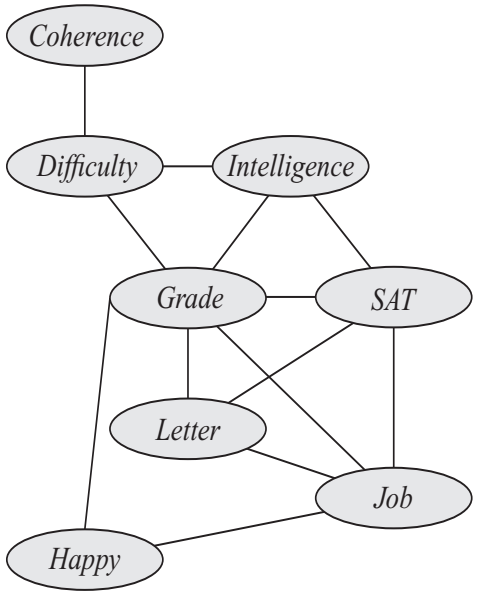
- Each vertex of G in some bag
- Each edge of G in some bag
- Each vertex of G induce a subtree (running intersection property)

T,B Junction tree



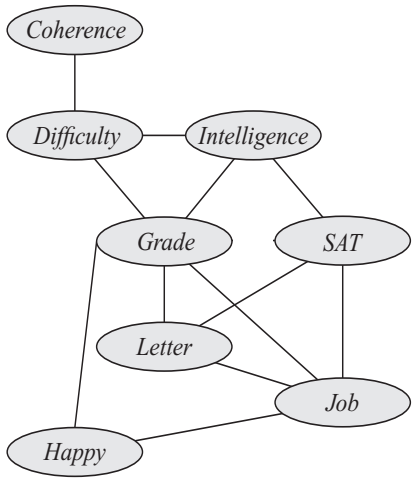
CONVERTING DGM TO UGM





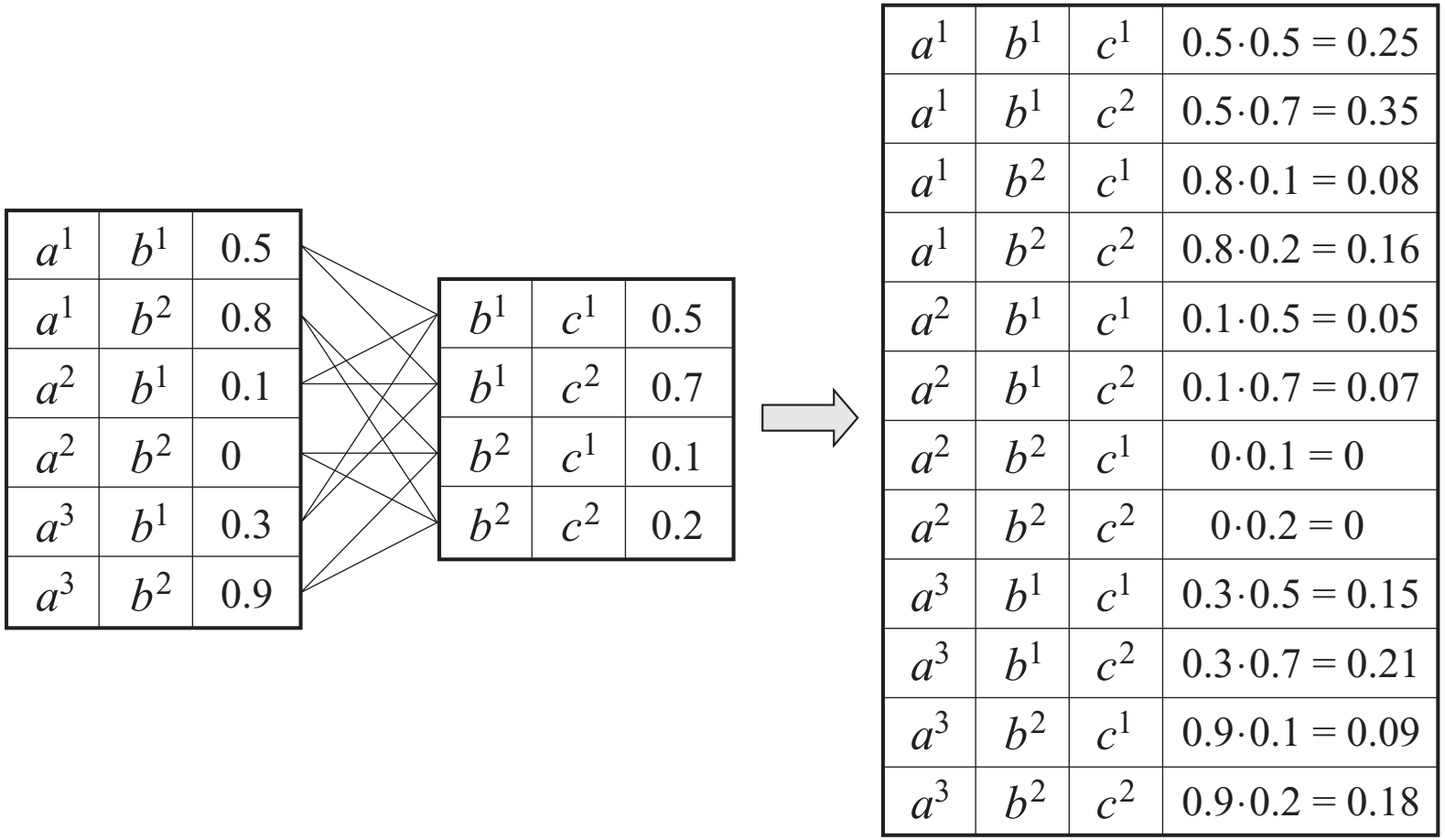
- Moralize and remove directions
  - does not introduce new independencies!
- Use CPDs as factors

JOINT AS FACTOR PRODUCT



- $$P(C, D, I, G, S, L, J, H)$$
$$= P(C)P(D|C)P(I)P(G|I, D)P(S|I)P(L|G)P(J|L, S)P(H|G, J)$$

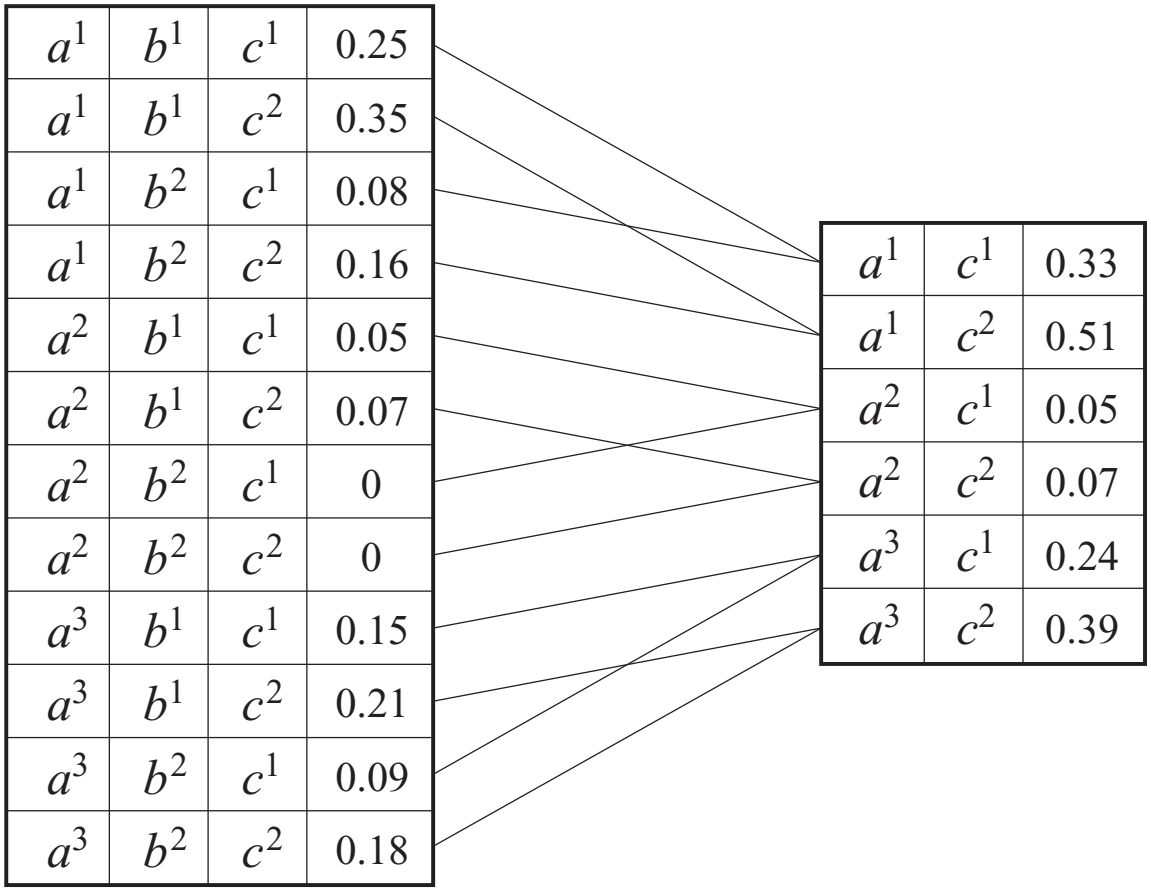
- $$p(C, D, I, G, S, L, J, H)$$
$$= \psi_C(C)\psi_D(D, C)\psi_I(I)\psi_G(G, I, D)\psi_S(S, I)\psi_L(L, G)\psi_J(J, L, S)\psi_H(H, G, J)$$



A FACTOR PRODUCT - RESULT SIZE IS EXP IN #R.V.S

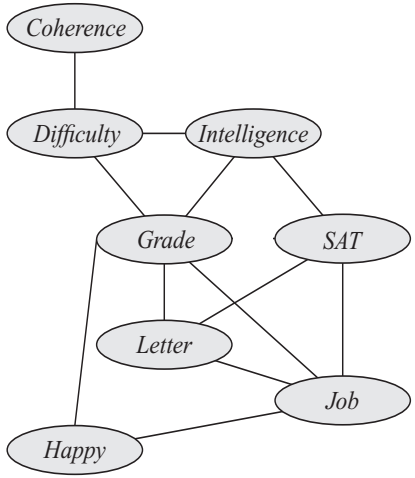
SUMMING OUT - IN FACTOR

a^1	b^1	c^1	0.25
a^1	b^1	c^2	0.35
a^1	b^2	c^1	0.08
a^1	b^2	c^2	0.16
a^2	b^1	c^1	0.05
a^2	b^1	c^2	0.07
a^2	b^2	c^1	0
a^2	b^2	c^2	0
a^3	b^1	c^1	0.15
a^3	b^1	c^2	0.21
a^3	b^2	c^1	0.09
a^3	b^2	c^2	0.18



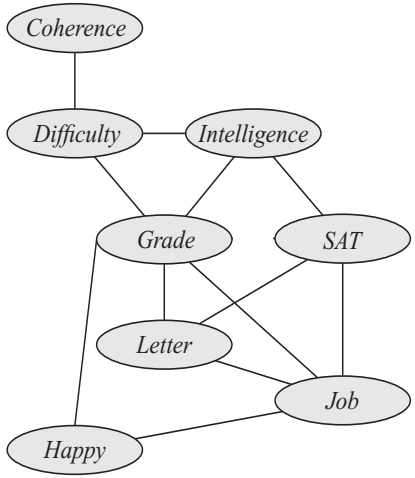
a^1	c^1	0.33
a^1	c^2	0.51
a^2	c^1	0.05
a^2	c^2	0.07
a^3	c^1	0.24
a^3	c^2	0.39

SUMMING OUT



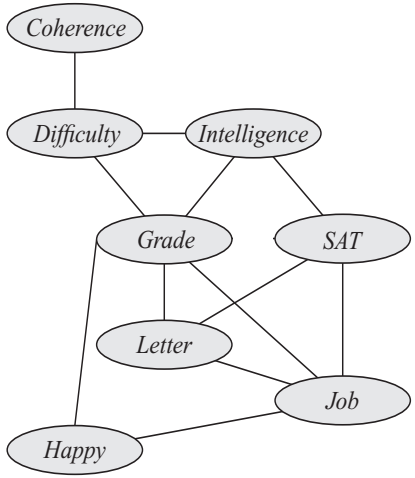
- $$p(J) = \sum_L \sum_S \sum_G \sum_H \sum_I \sum_D \sum_C p(C, D, I, G, S, L, J, H)$$

SUMMING OUT



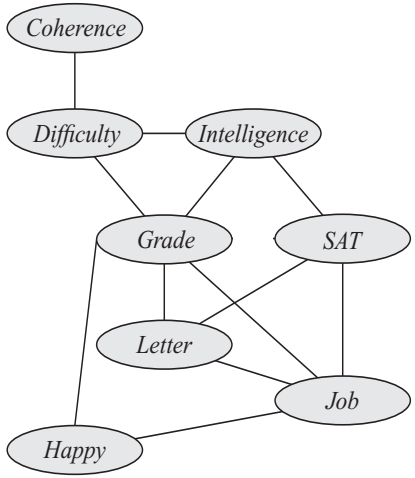
$$
\begin{aligned}
p(J) &= \sum_{L,S,G,H,I,D,C} p(C,D,I,G,S,L,J,H) \\
&= \sum_{L,S,G,H,I,D,C} \psi_C(C)\psi_D(D,C)\psi_I(I)\psi_G(G,I,D)\psi_S(S,I)\psi_L(L,G) \\
&\quad \times \psi_J(J,L,S)\psi_H(H,G,J) \\
&= \sum_{L,S} \psi_J(J,L,S) \sum_{G} \psi_L(L,G) \sum_{H} \psi_H(H,G,J) \sum_{I} \psi_S(S,I)\psi_I(I) \\
&\quad \times \sum_{D} \psi_G(G,I,D) \sum_{C} \psi_C(C)\psi_D(D,C)
\end{aligned}
$$

SUMMING OUT



- $$p(J) = \sum_{L,S} \psi_J(J,L,S) \sum_G \psi_L(L,G) \sum_H \psi_H(H,G,J) \sum_I \psi_S(S,I)\psi_I(I)$$
$$\times \sum_D \psi_G(G,I,D) \sum_C \psi_C(C)\psi_D(D,C)$$

SUMMING OUT



$$p(J) = \sum_{L,S} \psi_J(J,L,S) \sum_G \psi_L(L,G) \sum_H \psi_H(H,G,J) \sum_I \psi_S(S,I)\psi_I(I)$$
$$\times \sum_D \psi_G(G,I,D) \sum_C \psi_C(C)\psi_D(D,C)$$

Temporary factors

$$\tau_1'(C,D) = \psi_C(C)\psi_D(D,C)$$

$$\tau_1(D) = \sum_C \tau_1'(C,D)$$

$$\sum_L \sum_S \psi_J(J,L,S) \sum_G \psi_L(L,G) \sum_H \psi_H(H,G,J) \sum_I \psi_S(S,I)\psi_I(I) \sum_D \psi_G(G,I,D) \underbrace{\sum_C \psi_C(C)\psi_D(D,C)}_{\tau_1(D)}$$

$$\sum_L \sum_S \psi_J(J,L,S) \sum_G \psi_L(L,G) \sum_H \psi_H(H,G,J) \sum_I \psi_S(S,I)\psi_I(I) \underbrace{\sum_D \psi_G(G,I,D)\tau_1(D)}_{\tau_2(G,I)}$$

$$\sum_L \sum_S \psi_J(J,L,S) \sum_G \psi_L(L,G) \sum_H \psi_H(H,G,J) \underbrace{\sum_I \psi_S(S,I)\psi_I(I)\tau_2(G,I)}_{\tau_3(G,S)}$$

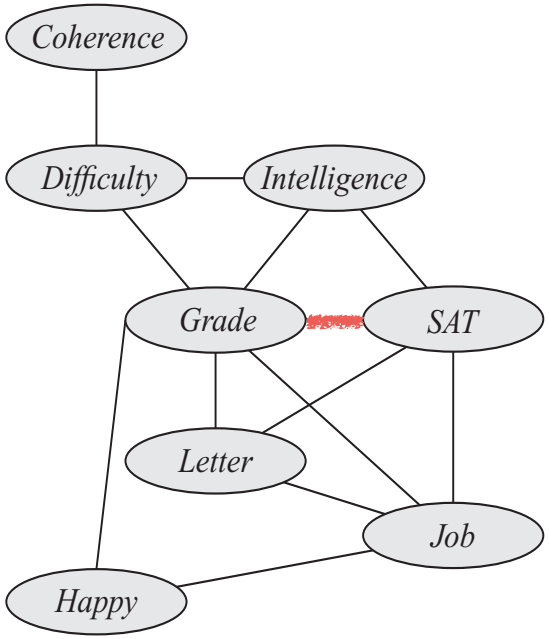
$$\sum_L \sum_S \psi_J(J,L,S) \sum_G \psi_L(L,G) \underbrace{\sum_H \psi_H(H,G,J)}_{\tau_4(G,J)} \tau_3(G,S)$$

$$\sum_L \sum_S \psi_J(J,L,S) \underbrace{\sum_G \psi_L(L,G)\tau_4(G,J)\tau_3(G,S)}_{\tau_5(J,L,S)}$$

$$\sum_L \underbrace{\sum_S \psi_J(J,L,S)\tau_5(J,L,S)}_{\tau_6(J,L)}$$

$$\underbrace{\sum_L \tau_6(J,L)}_{\tau_7(J)}$$

THE ENTIRE SEQUENCE



$$\sum_L \sum_S \psi_J(J,L,S) \sum_G \psi_L(L,G) \sum_H \psi_H(H,G,J) \underbrace{\sum_I \psi_S(S,I)\psi_I(I)\tau_2(G,I)}_{\tau_3(G,S)}$$

$$\sum_L \sum_S \psi_J(J,L,S) \sum_G \psi_L(L,G) \underbrace{\sum_H \psi_H(H,G,J)}_{\tau_4(G,J)} \tau_3(G,S)$$

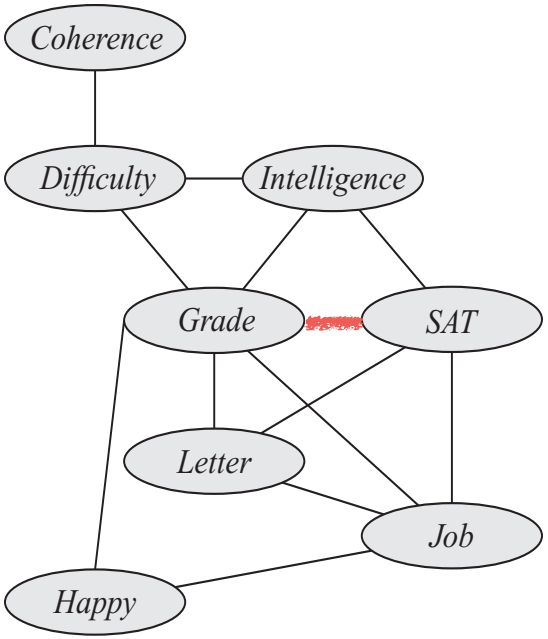
$$\sum_L \sum_S \psi_J(J,L,S) \underbrace{\sum_G \psi_L(L,G)\tau_4(G,J)\tau_3(G,S)}_{\tau_5(J,L,S)}$$

$$\sum_L \underbrace{\sum_S \psi_J(J,L,S)\tau_5(J,L,S)}_{\tau_6(J,L)}$$

$$\underbrace{\sum_L \tau_6(J,L)}_{\tau_7(J)}$$

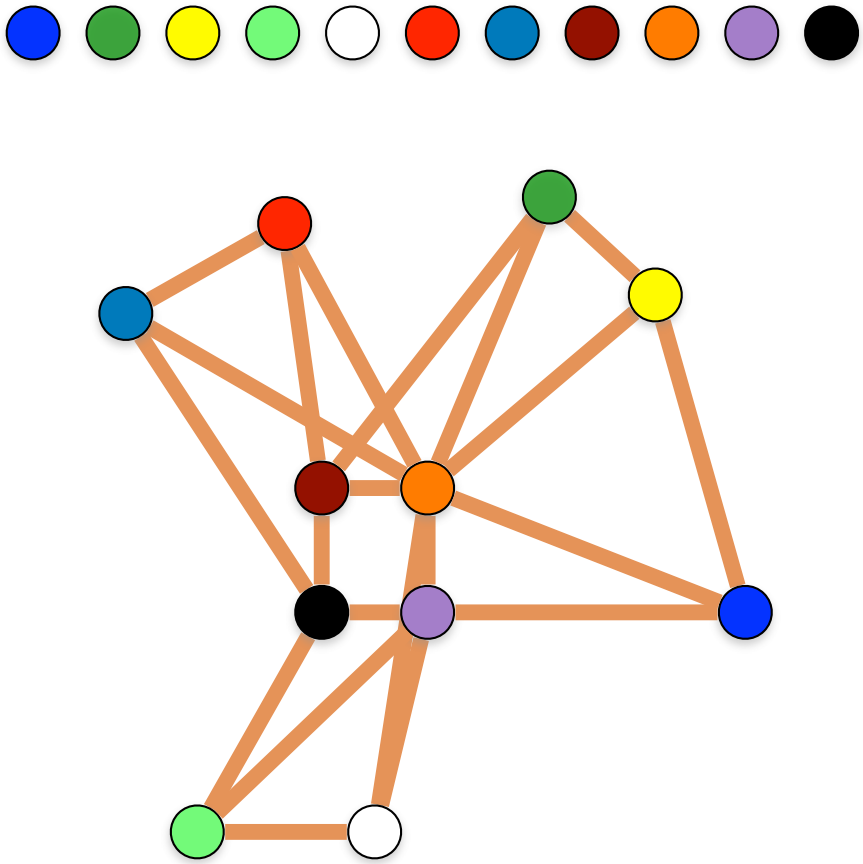
Fill in edge between two vertices in a “temporary factor”

A FILL-IN EDGE



- The fill in edges depends on the elimination order.
- The complexity depends on the maximum clique size (in G with fill in added)

GETTING GOOD FILL-IN EDGES



ALTERNATIVE DEFINITION OF WIDTH

Elimination order of G – ordering of V(G)

$v_1,\ldots,v_n$

It defines a sequence of graphs $G=G_0,\ldots,G_n$

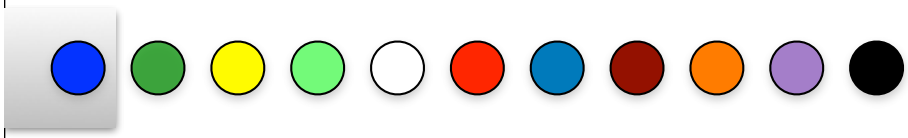
where $G_i := G_{i-1} \setminus v_i \cup \{(u,w): u,w \in N_{G_{i-1}}(v_i)\}$

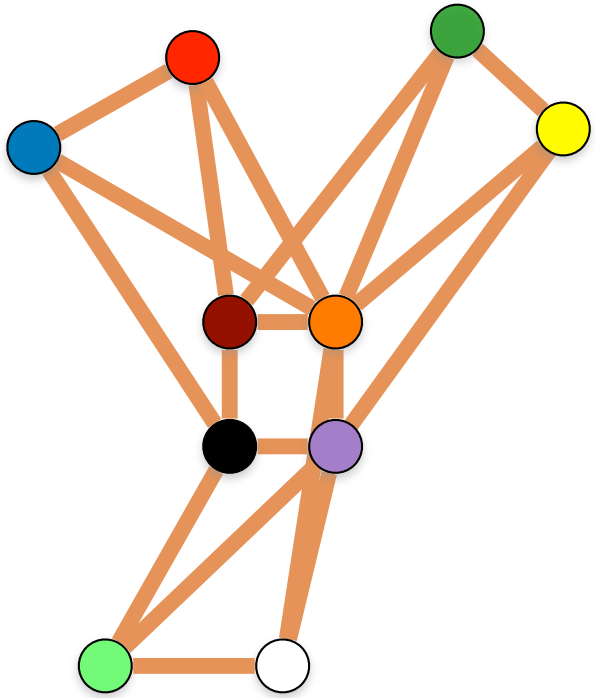
(i.e., we add edges between all neighbours)

width of order = (max clique size in $\cup_i G_i$) -1

= $\max_i$ degree of v_i in G_i

width of G = min width of order





ALTERNATIVE DEFINITION OF WIDTH

Elimination order of G – ordering of V(G)

$v_1,\ldots,v_n$

It defines a sequence of graphs $G=G_1,\ldots,G_n$

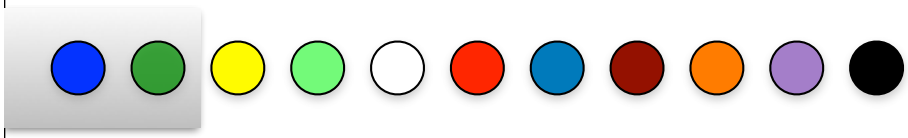
where $G_i := G_{i-1} \setminus v_i \cup \{(u,w): u,w \in N_{G_{i-1}}(v_i)\}$

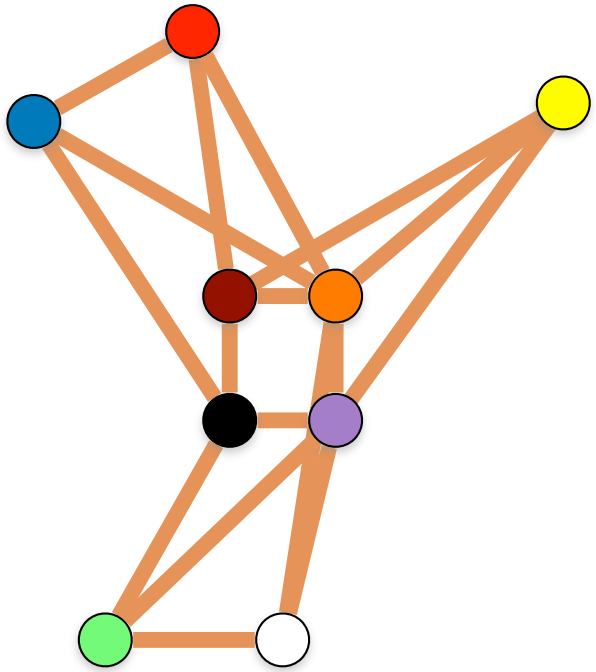
(i.e., we add edges between all neighbours)

width of order = (max clique size in $\cup_i G_i$) -1

= $\max_i$ degree of v_i in G_i

width of G = min width of order





ALTERNATIVE DEFINITION OF WIDTH

Elimination order of G – ordering of V(G)

$v_1,\ldots,v_n$

It defines a sequence of graphs $G=G_1,\ldots,G_n$

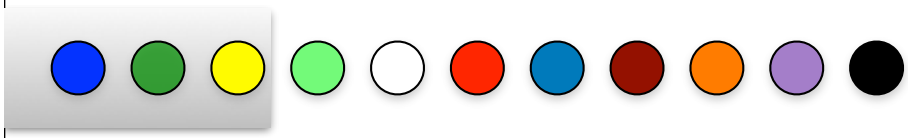
where $G_i := G_{i-1} \setminus v_i \cup \{(u,w): u,w \in N_{G_{i-1}}(v_i)\}$

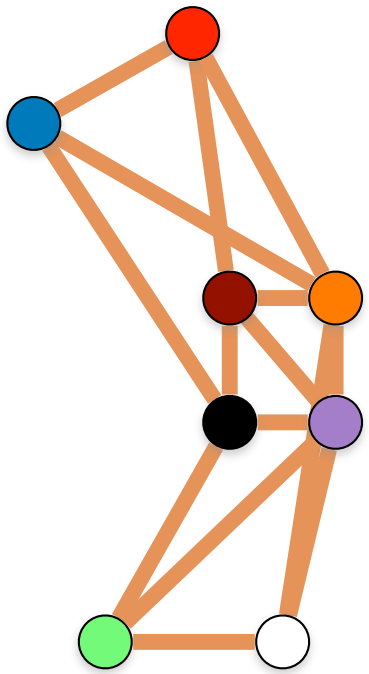
(i.e., we add edges between all neighbours)

width of order = (max clique size in $\cup_i G_i$) -1

= $\max_i$ degree of v_i in G_i

width of G = min width of order





ALTERNATIVE DEFINITION OF WIDTH

Elimination order of G – ordering of V(G)

$v_1,\ldots,v_n$

It defines a sequence of graphs $G=G_1,\ldots,G_n$

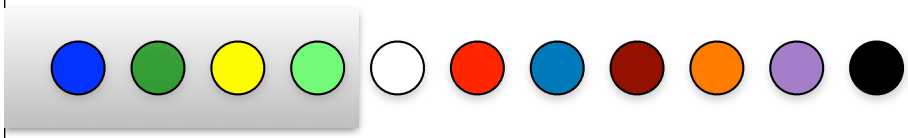
where $G_i := G_{i-1} \setminus v_i \cup \{(u,w): u,w \in N_{G_{i-1}}(v_i)\}$

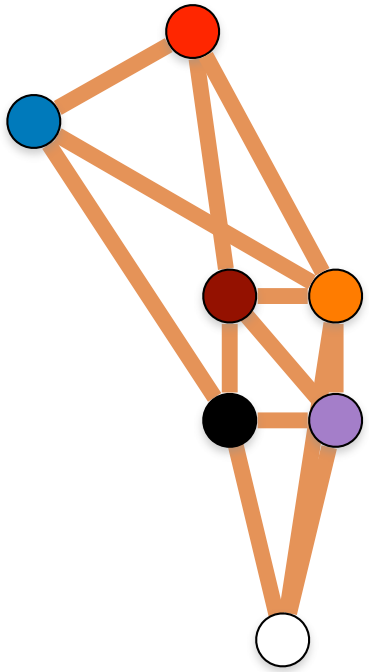
(i.e., we add edges between all neighbours)

width of order = (max clique size in $\cup_i G_i$) -1

= $\max_i$ degree of v_i in G_i

width of G = min width of order





ALTERNATIVE DEFINITION OF WIDTH

Elimination order of G – ordering of V(G)

$v_1,\ldots,v_n$

It defines a sequence of graphs $G=G_1,\ldots,G_n$

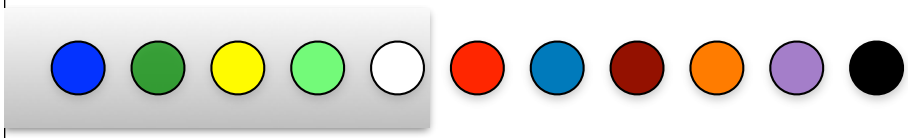
where $G_i := G_{i-1} \setminus v_i \cup \{(u,w): u,w \in N_{G_{i-1}}(v_i)\}$

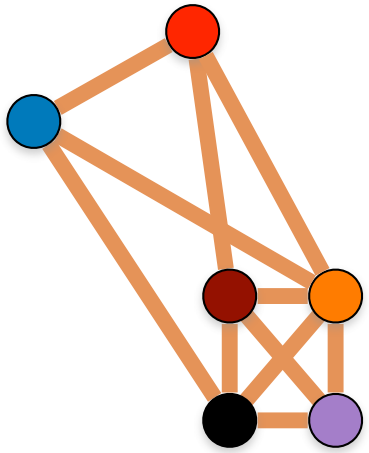
(i.e., we add edges between all neighbours)

width of order = (max clique size in $\cup_i G_i$) -1

= $\max_i$ degree of v_i in G_i

width of G = min width of order





ALTERNATIVE DEFINITION OF WIDTH

Elimination order of G – ordering of V(G)

$v_1,\ldots,v_n$

It defines a sequence of graphs $G=G_1,\ldots,G_n$

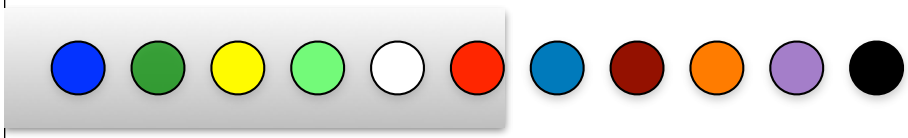
where $G_i := G_{i-1} \setminus v_i \cup \{(u,w): u,w \in N_{G_{i-1}}(v_i)\}$

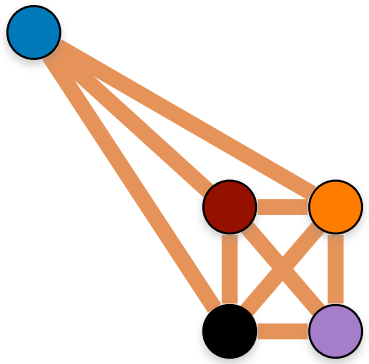
(i.e., we add edges between all neighbours)

width of order = (max clique size in $\cup_i G_i$) -1

= $\max_i$ degree of v_i in G_i

width of G = min width of order





ALTERNATIVE DEFINITION OF WIDTH

Elimination order of G – ordering of V(G)

$v_1,\ldots,v_n$

It defines a sequence of graphs $G=G_1,\ldots,G_n$

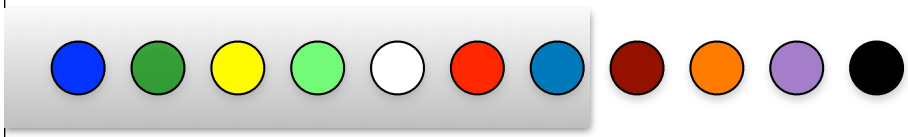
where $G_i := G_{i-1} \setminus v_i \cup \{(u,w): u,w \in N_{G_{i-1}}(v_i)\}$

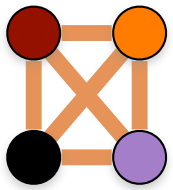
(i.e., we add edges between all neighbours)

width of order = (max clique size in $\cup_i G_i$) -1

= $\max_i$ degree of v_i in G_i

width of G = min width of order





ALTERNATIVE DEFINITION OF WIDTH

Elimination order of G – ordering of V(G)

$v_1,\ldots,v_n$

It defines a sequence of graphs $G=G_1,\ldots,G_n$

where $G_i := G_{i-1} \setminus v_i \cup \{(u,w): u,w \in N_{G_{i-1}}(v_i)\}$

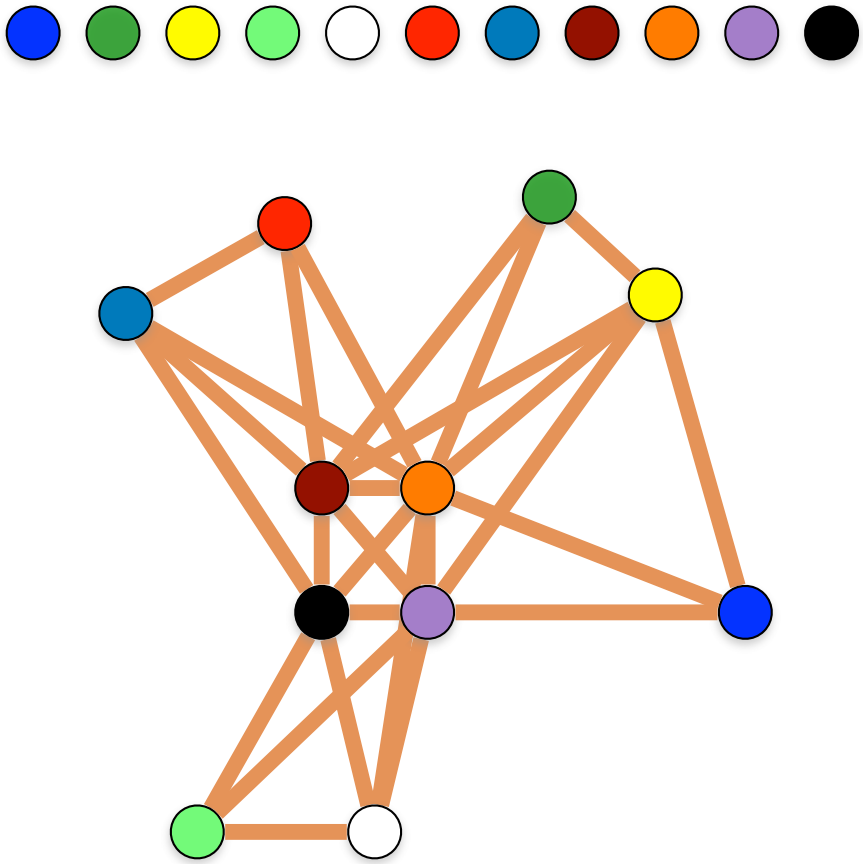
(i.e., we add edges between all neighbours)

width of order = (max clique size in $\cup_i G_i$) -1

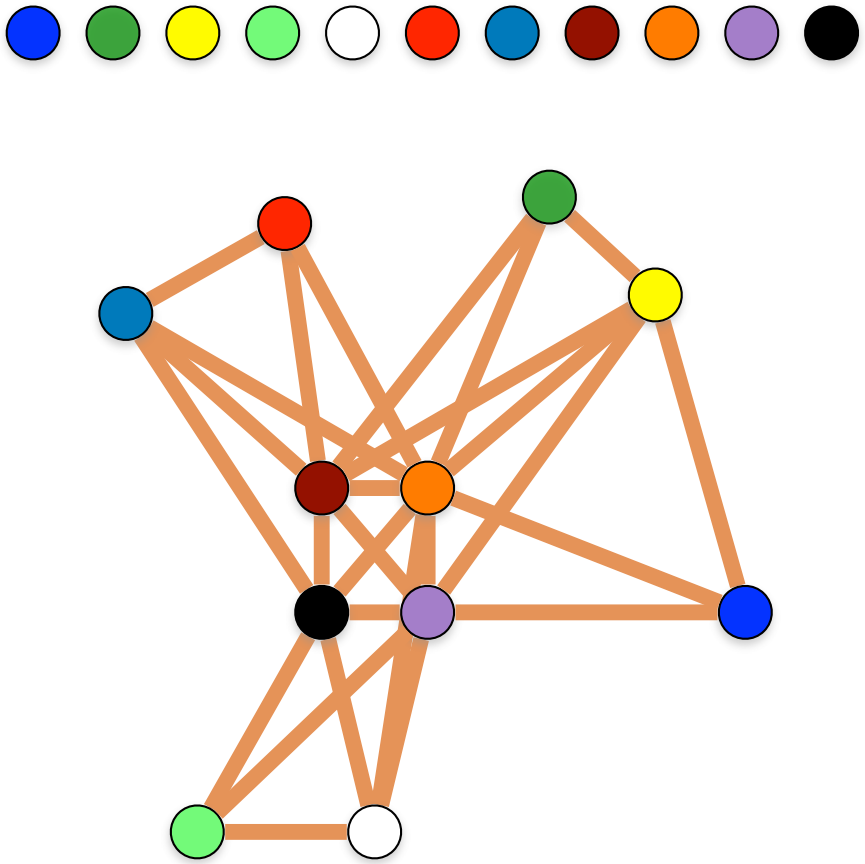
= $\max_i$ degree of v_i in G_i

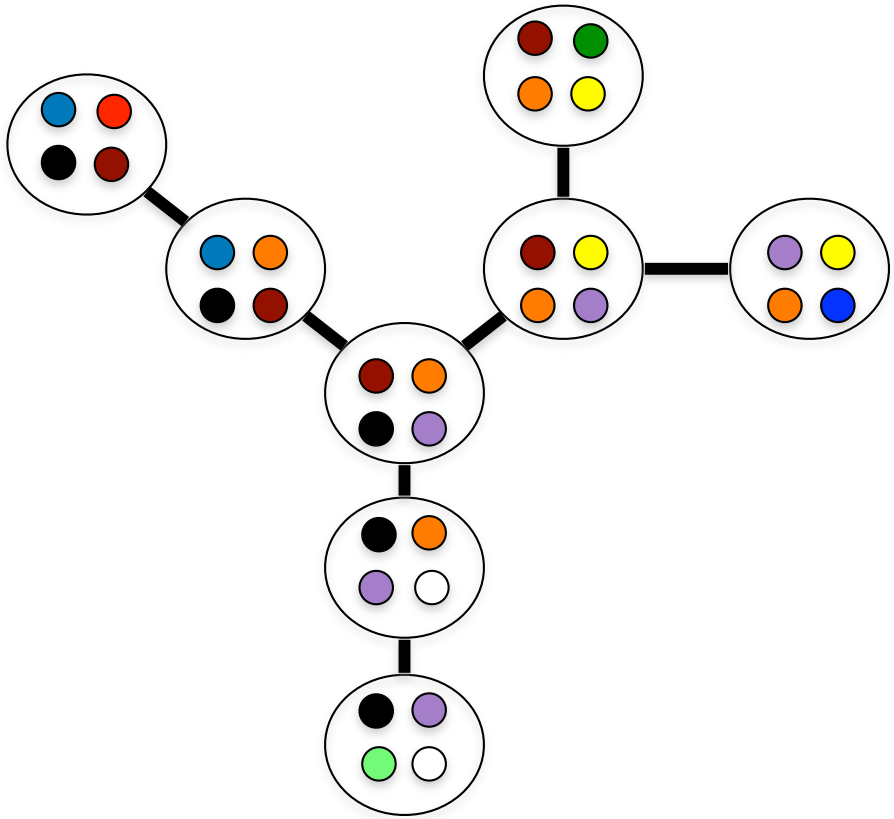
width of G = min width of order

ALTERNATIVE DEFINITION OF WIDTH

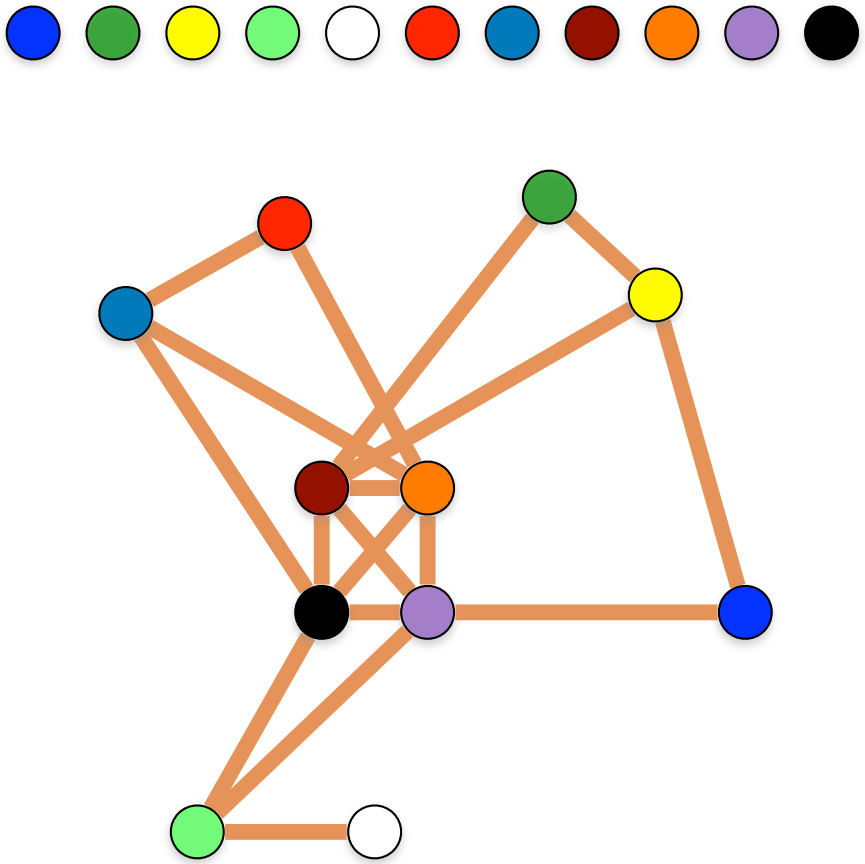


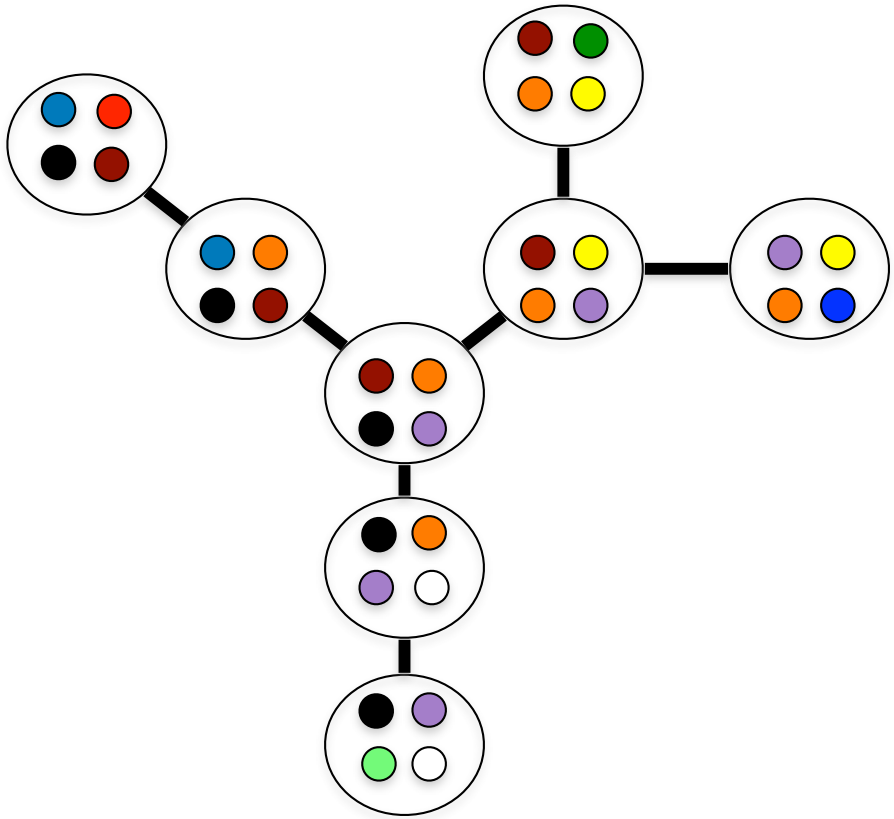
A graph has a junction tree of width k if and only if it has a width k elimination order





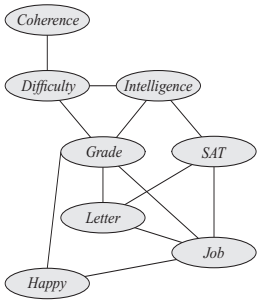
JUNCTION TREE AND ELIMINATION ORDER





JUNCTION TREE AND ELIMINATION ORDER

COMPLEXITY OF VERTEX ELIMINATION



$$\sum_L \sum_S \psi_J(J,L,S) \sum_G \psi_L(L,G) \sum_H \psi_H(H,G,J) \underbrace{\sum_I \psi_S(S,I)\psi_I(I)\tau_2(G,I)}_{\tau_3(G,S)}$$

$$\sum_L \sum_S \psi_J(J,L,S) \sum_G \psi_L(L,G) \underbrace{\sum_H \psi_H(H,G,J)\,\tau_3(G,S)}_{\tau_4(G,J)}$$

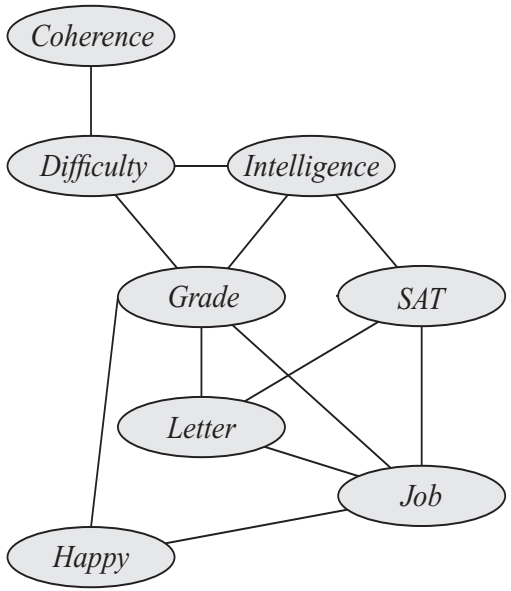
$$\sum_L \sum_S \psi_J(J,L,S) \underbrace{\sum_G \psi_L(L,G)\tau_4(G,J)\tau_3(G,S)}_{\tau_5(J,L,S)}$$

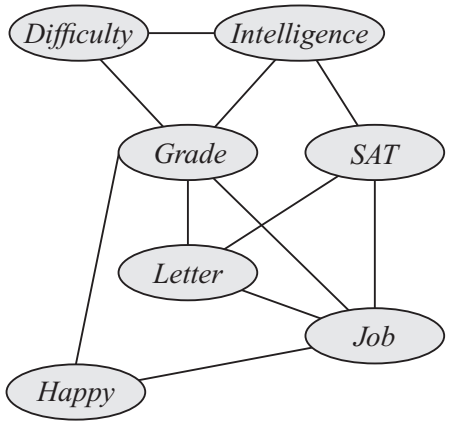
$$\underbrace{\sum_L \sum_S \psi_J(J,L,S)\tau_5(J,L,S)}_{\tau_6(J,L)}$$

$$\underbrace{\sum_L \tau_6(J,L)}_{\tau_7(J)}$$

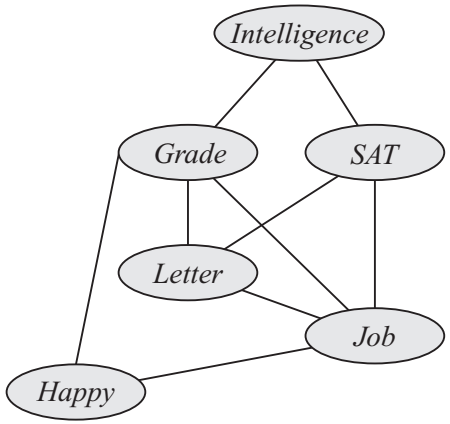
- If the width (of G ∪ fill in) is w
- All factors will have ≤ w+1 r.v.s
- Time $O(|V(G)|\ 2^w)$ for binary
- For categorical with C classes, $O(|V(G)|\ C^w)$

ELIMINATION ACCORDING TO THE ORDER C,D,I,H,G,S,L

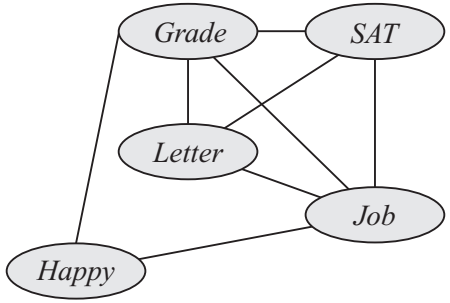




(a)

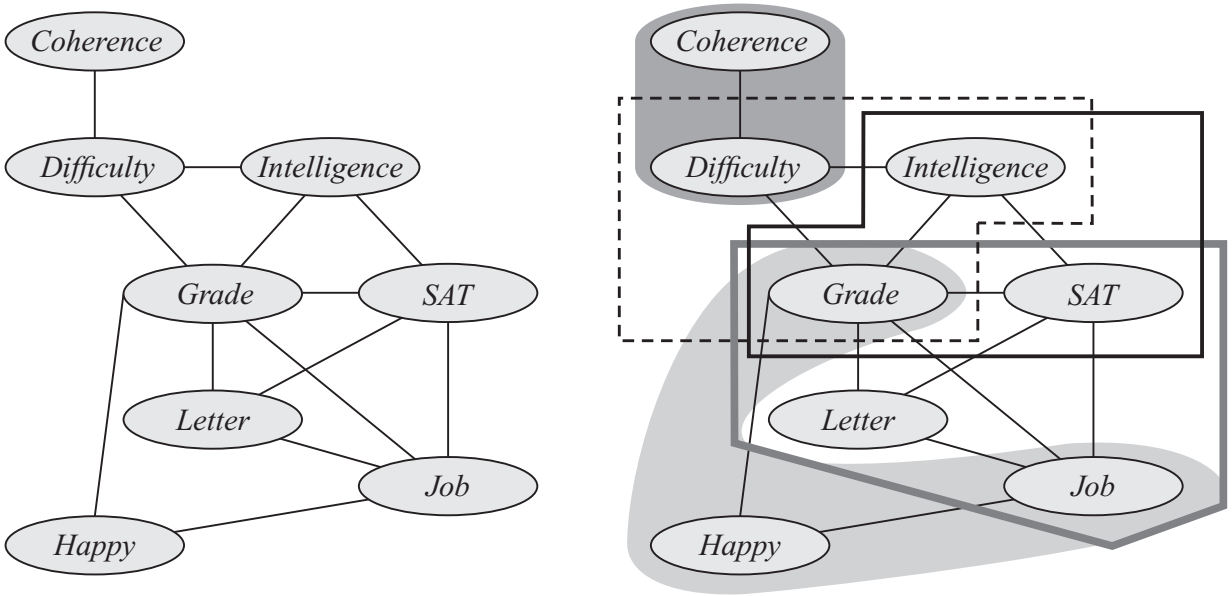


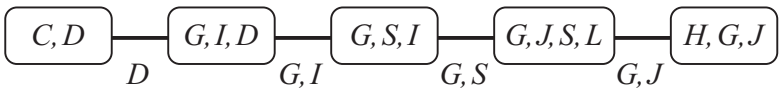
(b)



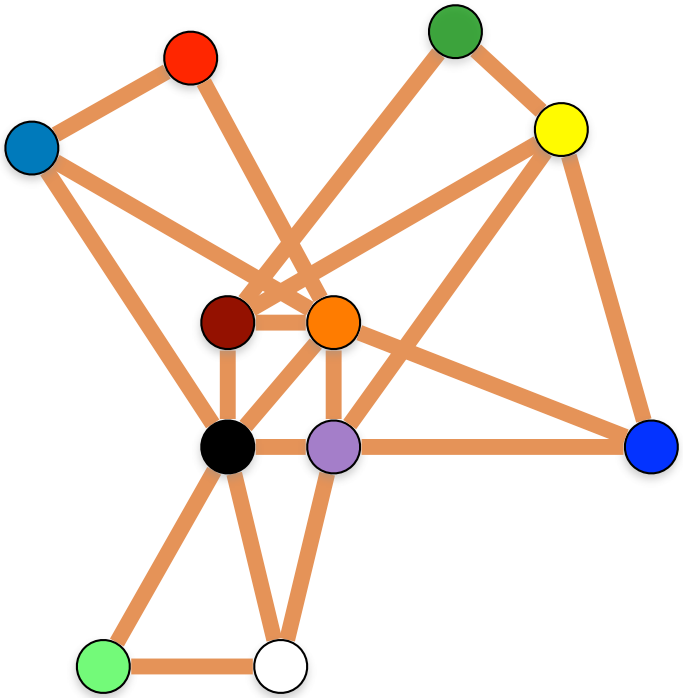
(c)

INDUCED GRAPH AND JUNCTION TREE FOR C,D,I,H,G,S,L





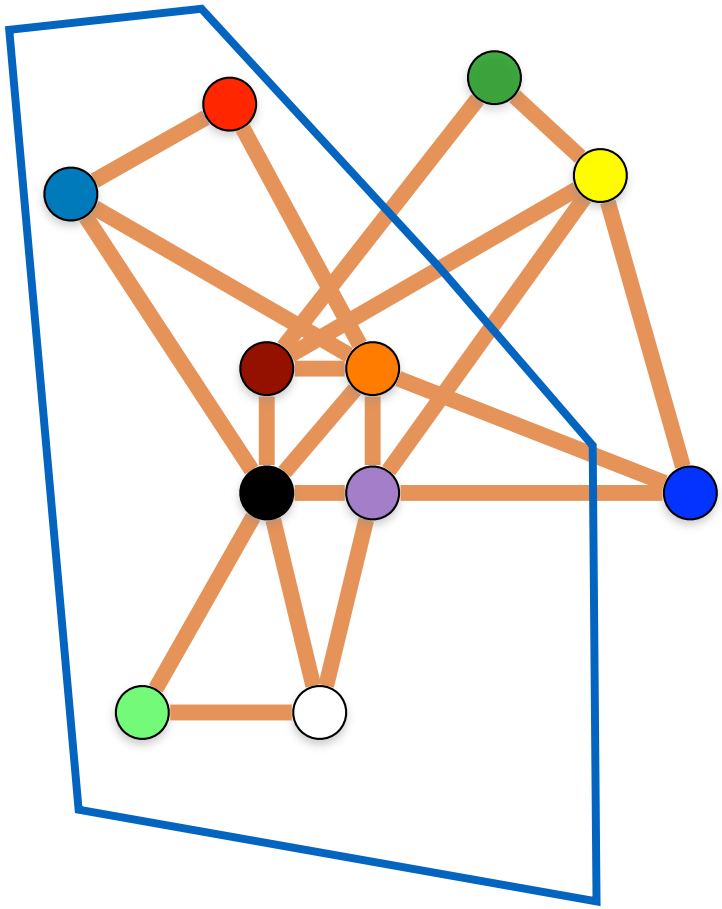
JUNCTION TREE GUIDED DP - INDEPENDENT SET

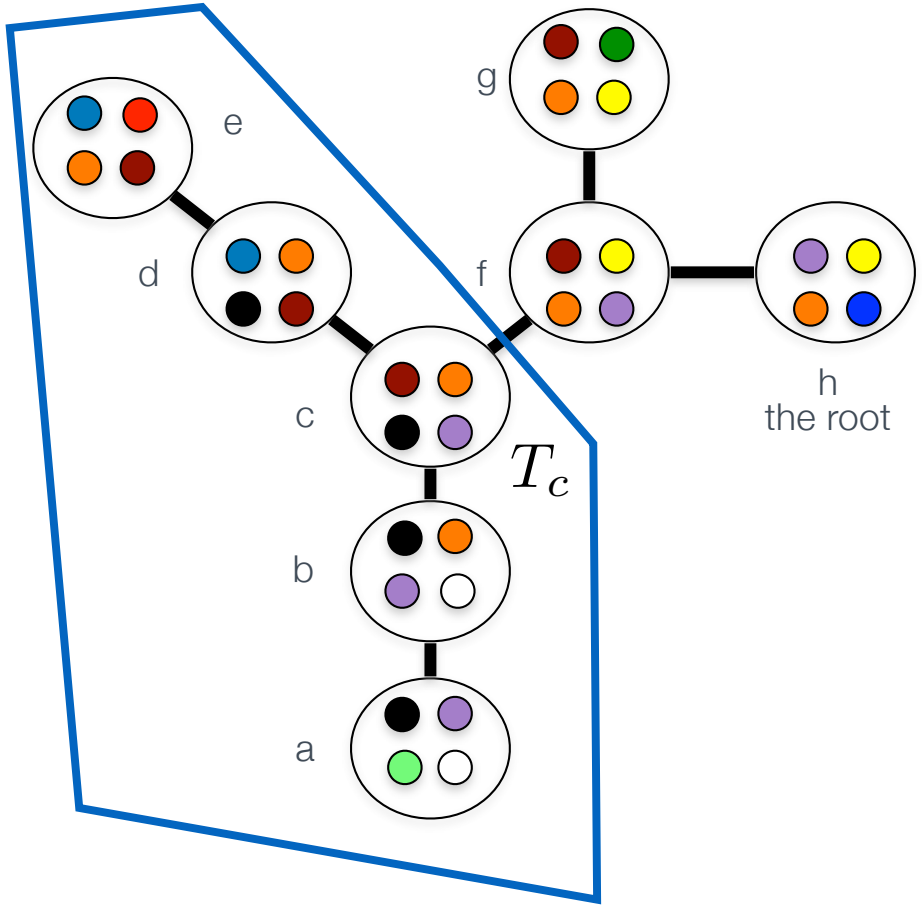


JUNCTION TREE GUIDED DP - SUBPROBLEMS

Subproblems

subtree rooted at c



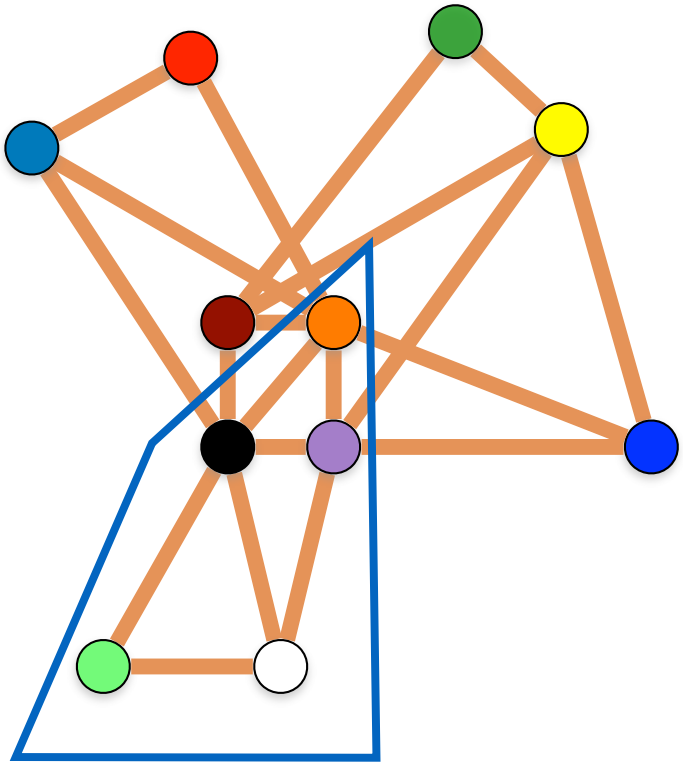


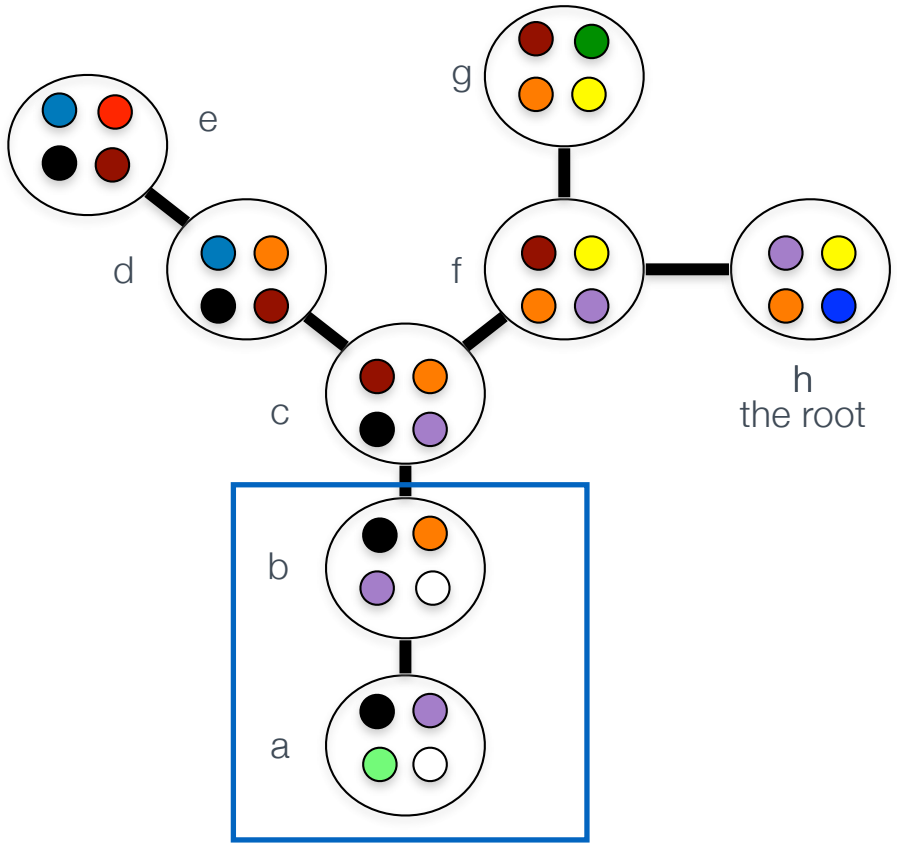
JUNCTION TREE GUIDED DP - SUBPROBLEMS

Subproblems

subtree rooted at v

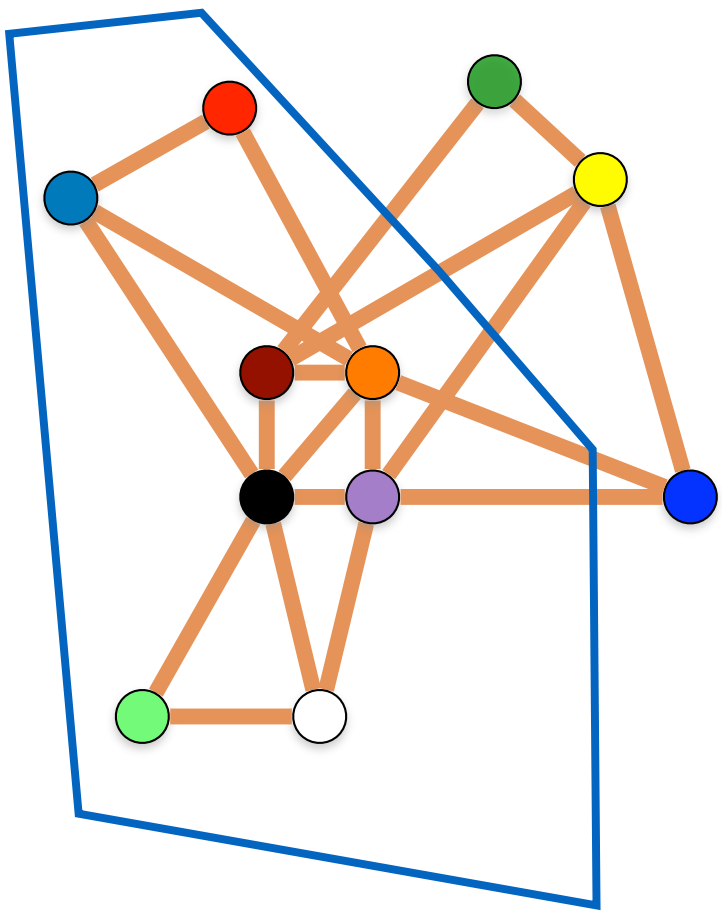
subtree associated with edge

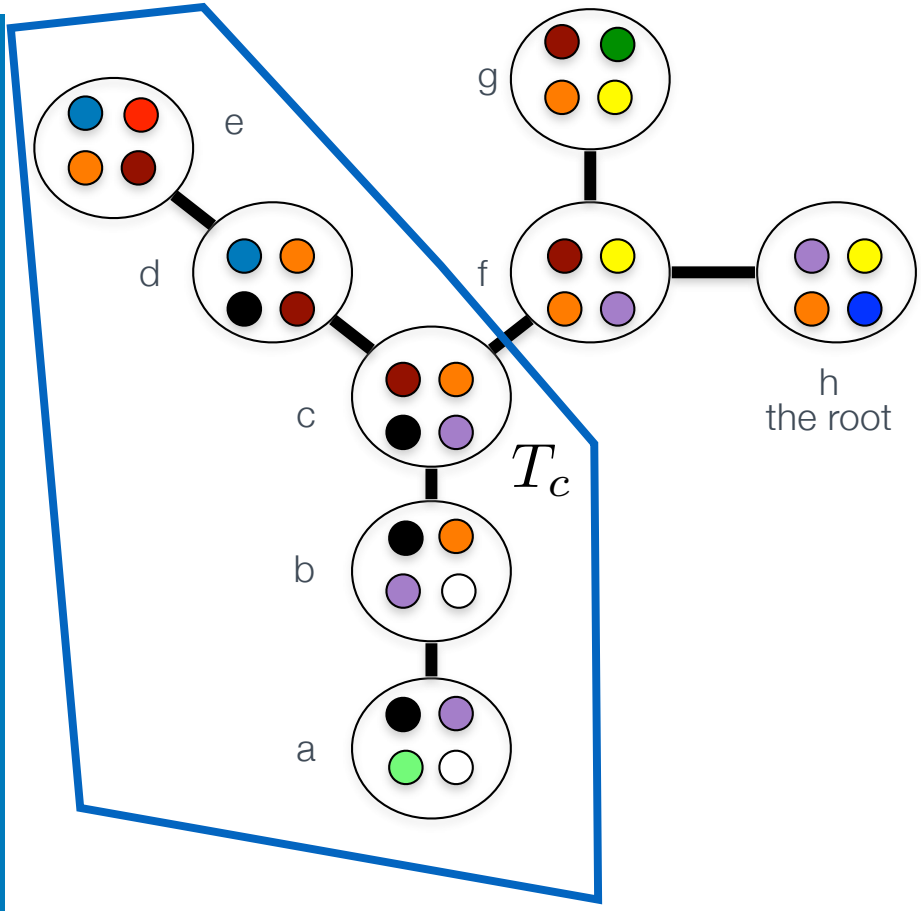




JUNCTION TREE GUIDED DP - SUBSOLUTIONS

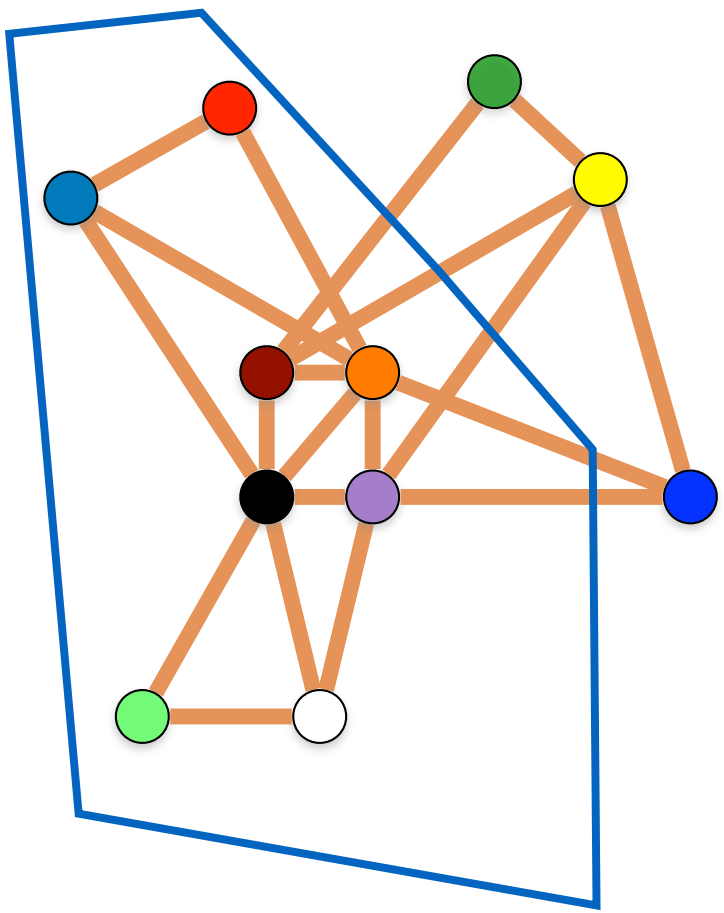
I_c table/function with sizes of maximum independent sets in subtree rooted at c
constrained by intersection with c:s bag



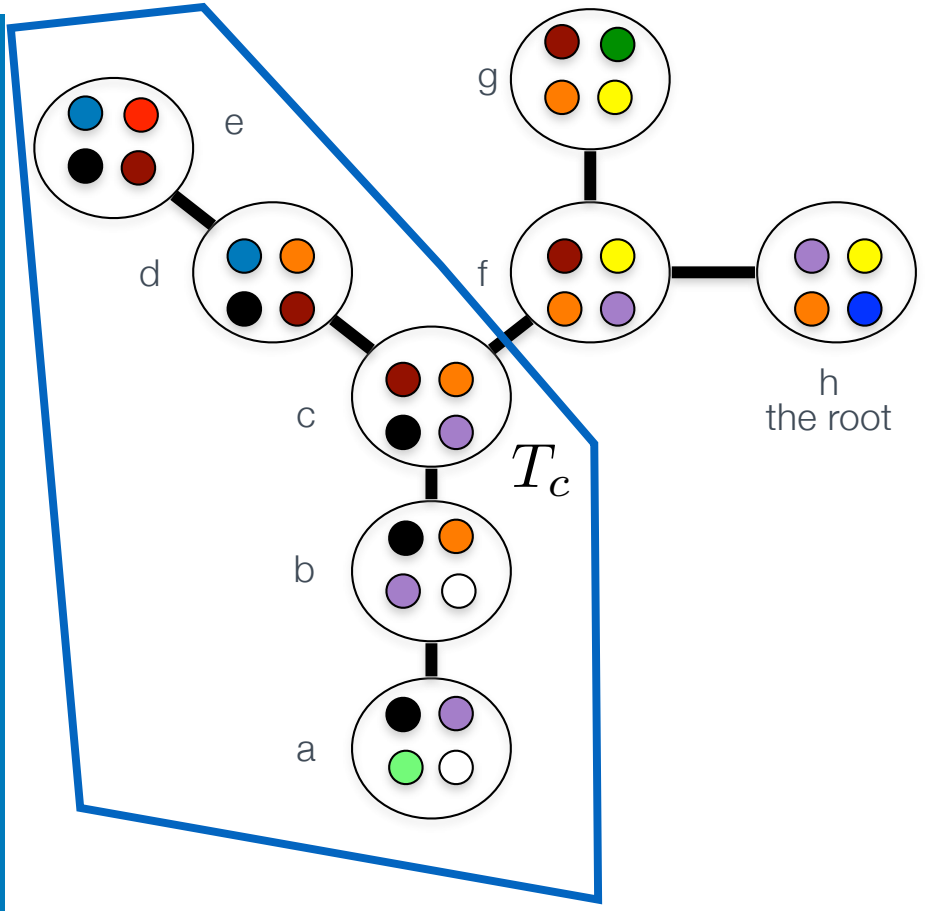


JUNCTION TREE GUIDED DP - SUBSOLUTIONS

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

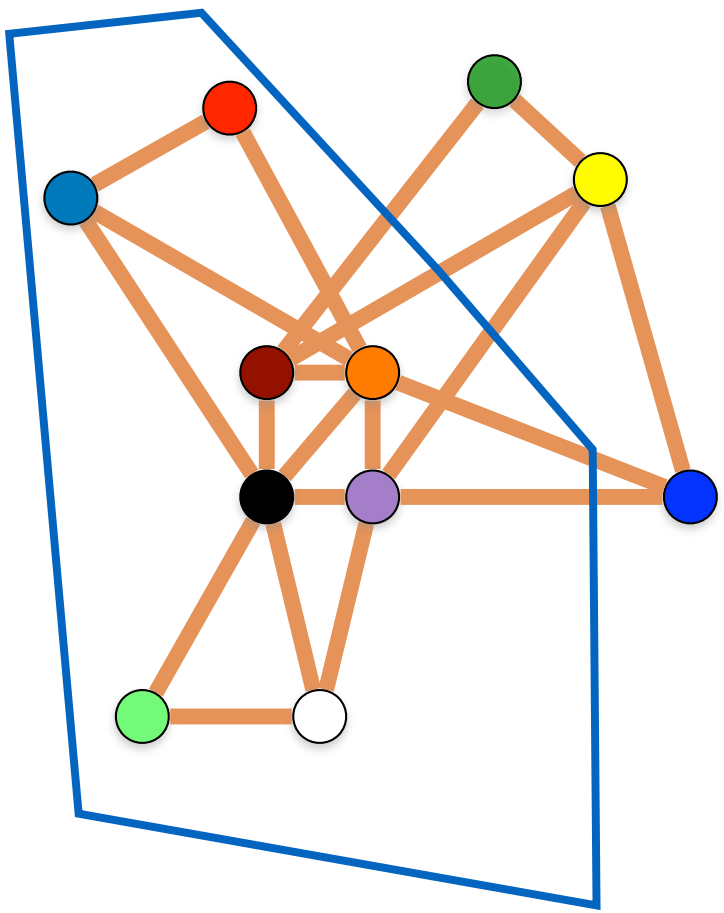


l_c (,) =

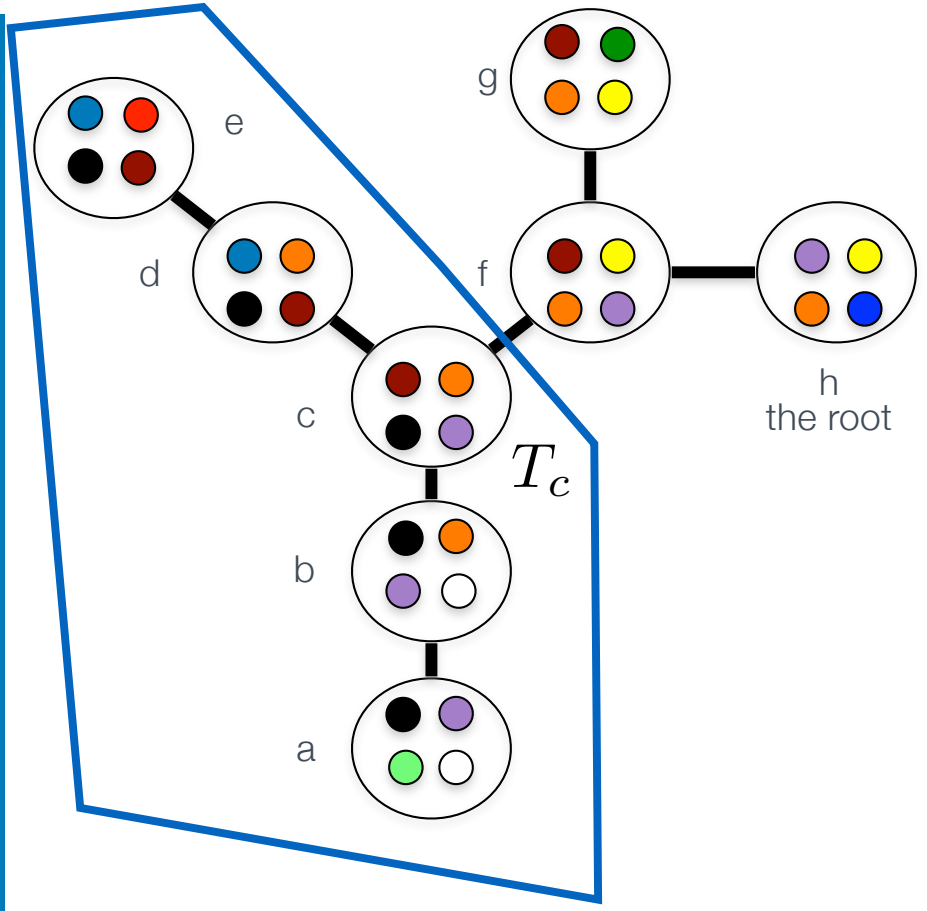


JUNCTION TREE GUIDED DP - SUBSOLUTIONS

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

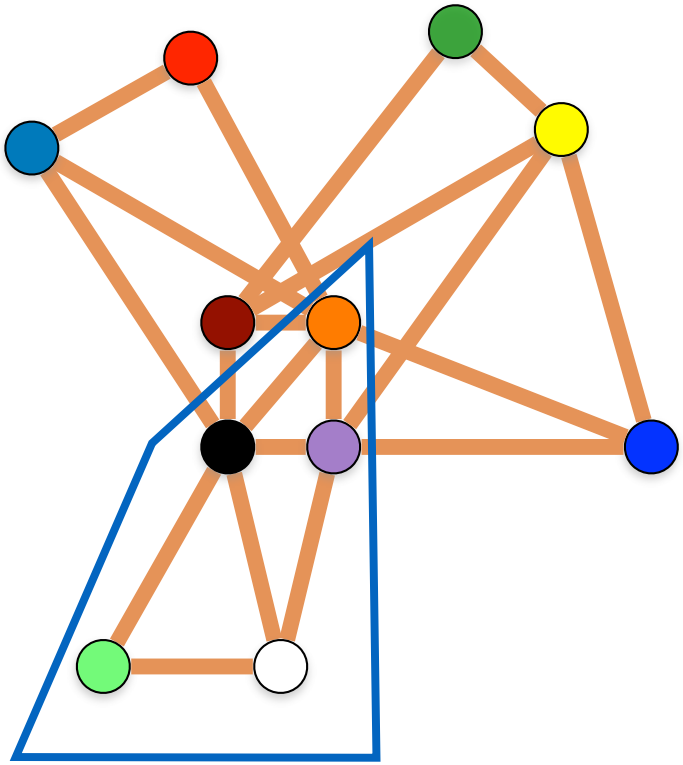


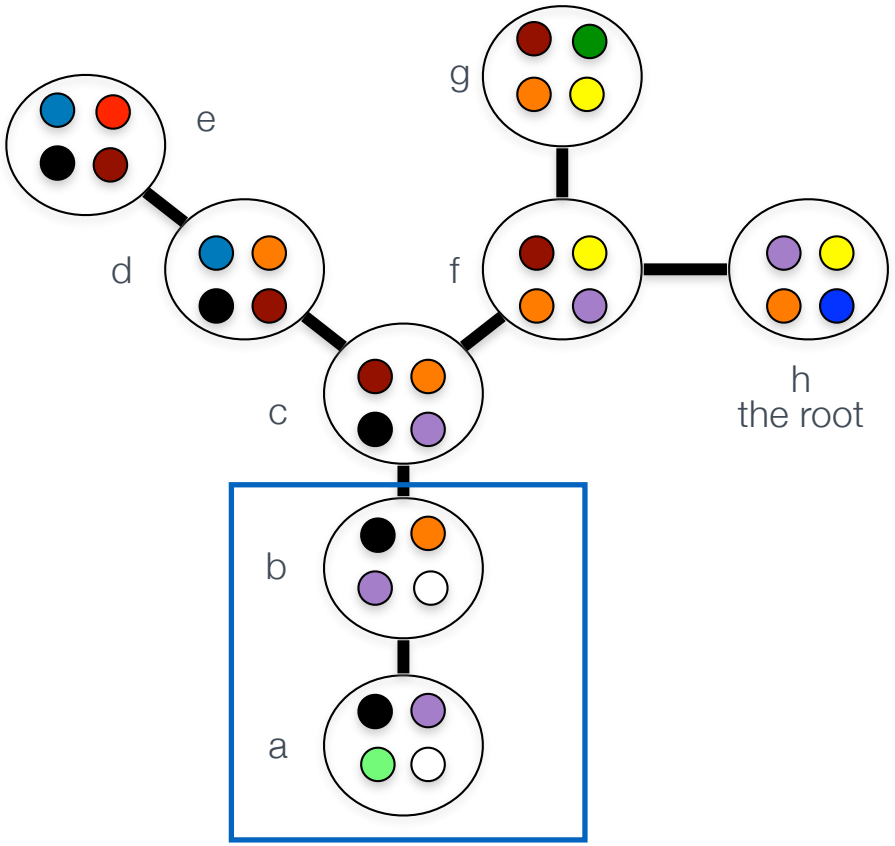
I_c (,) = 4



JUNCTION TREE GUIDED DP - SUBSOLUTIONS

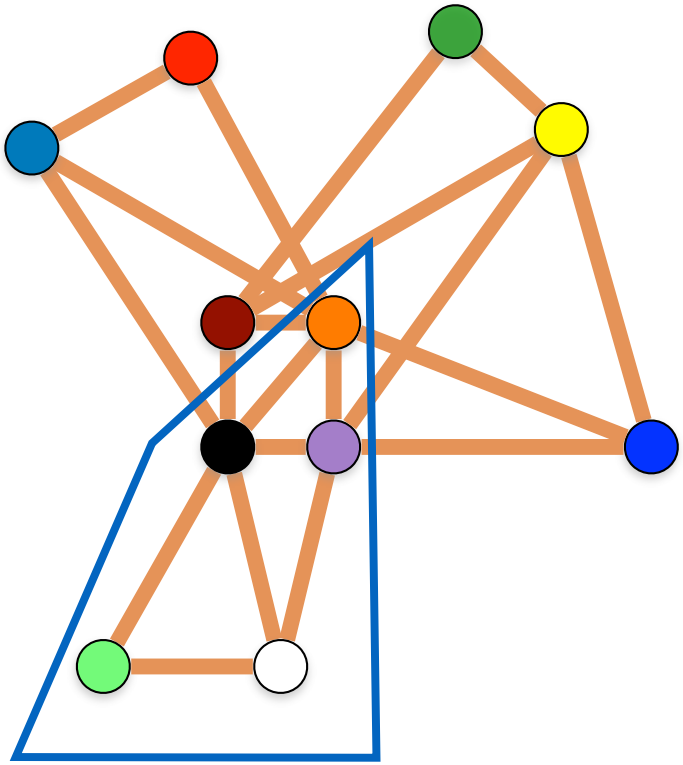
$I_{c \cap b}$ table/function with sizes of maximum independent sets in subtree below c and b based on intersection with the sepset $c \cap b$

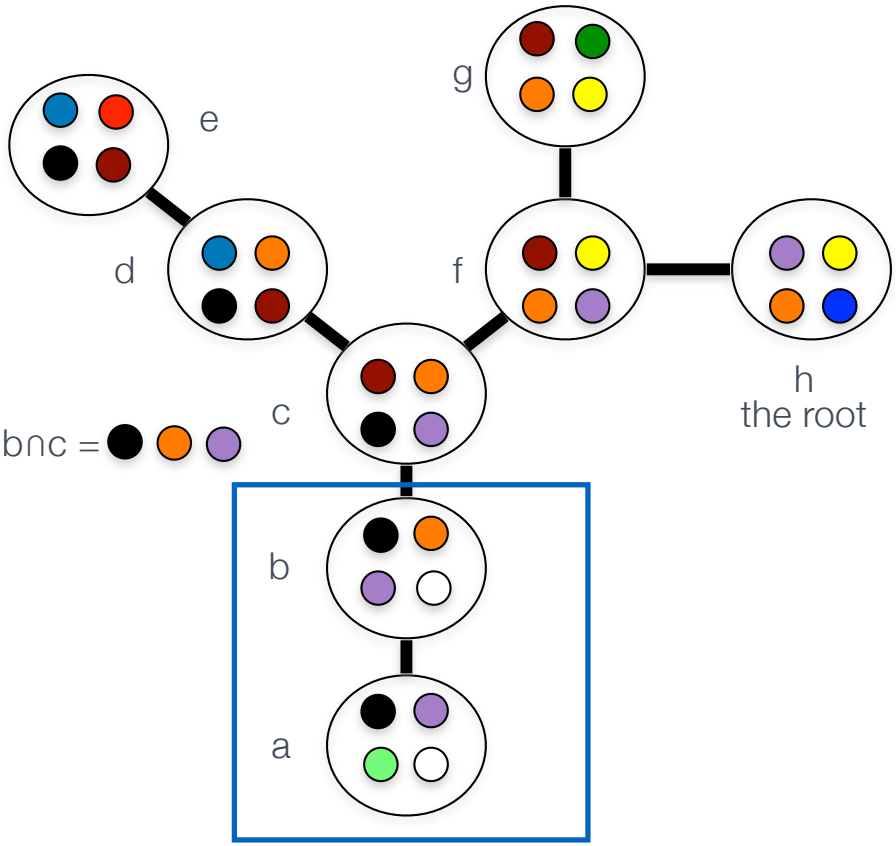




JUNCTION TREE GUIDED DP - SUBSOLUTIONS

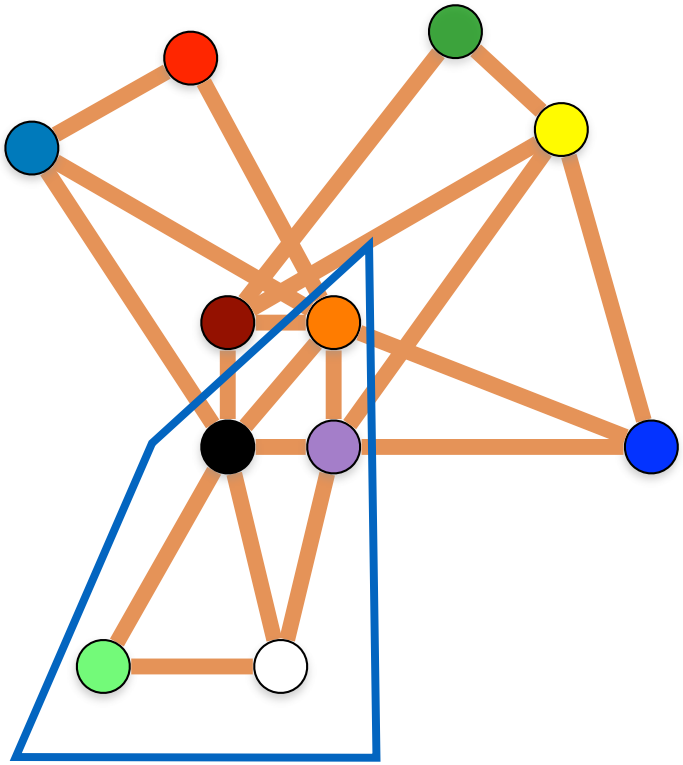
$I_{c \cap b}$ table/function with sizes of maximum independent sets in subtree below c and b based on intersection with the sepset $c \cap b$



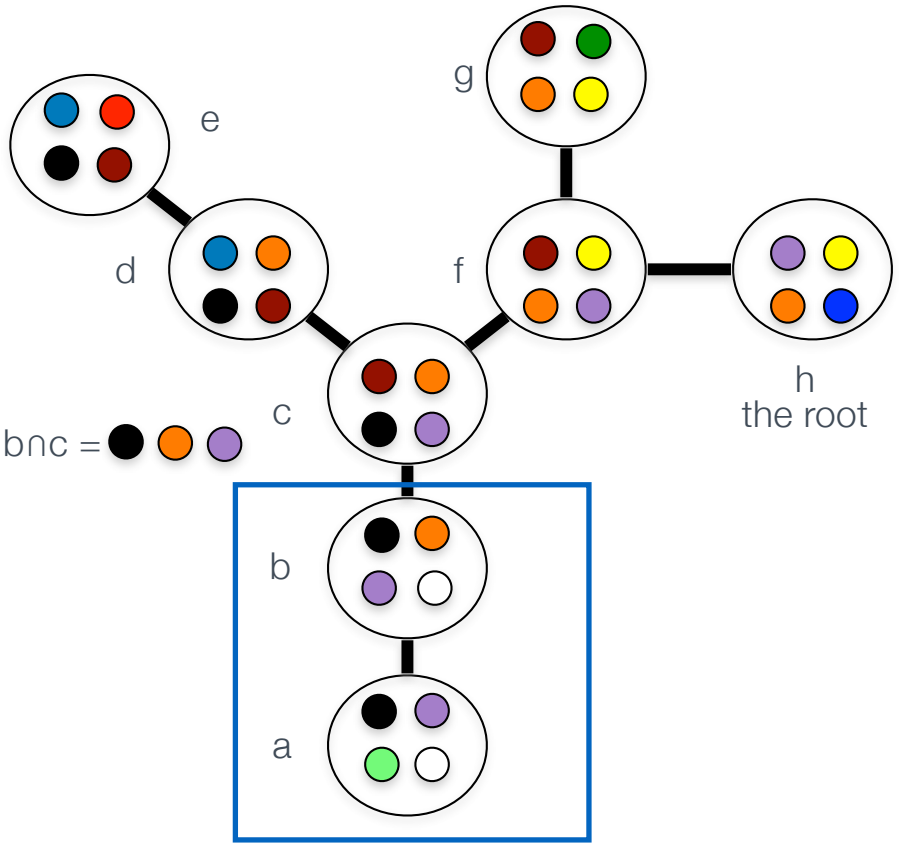


JUNCTION TREE GUIDED DP - SUBSOLUTIONS

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

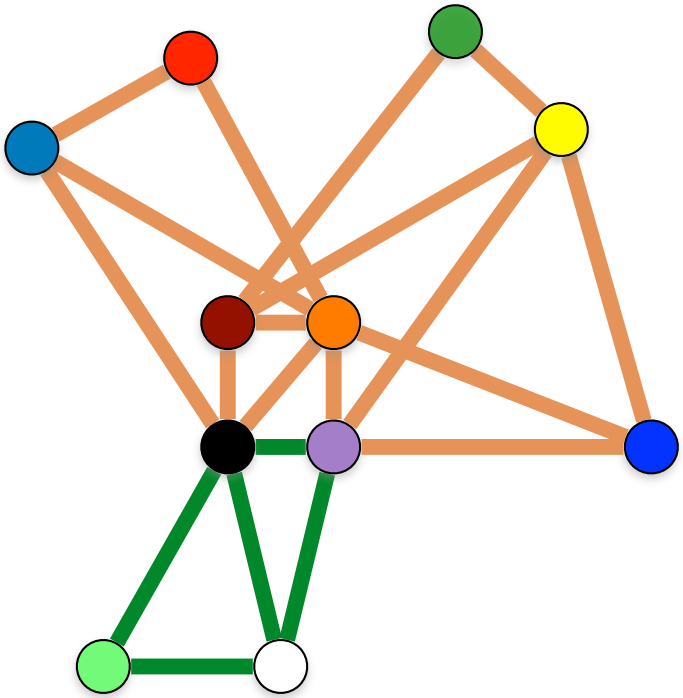


$I_{b \cap c}$ (●) = 2



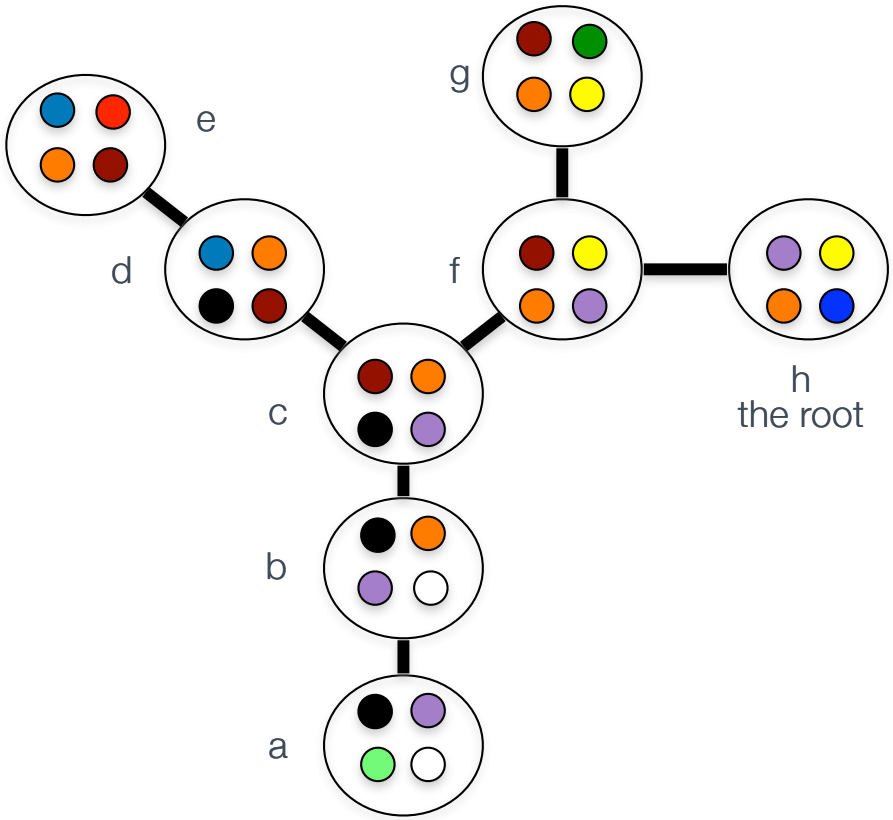
JUNCTION TREE GUIDED DP - COMPUTING

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



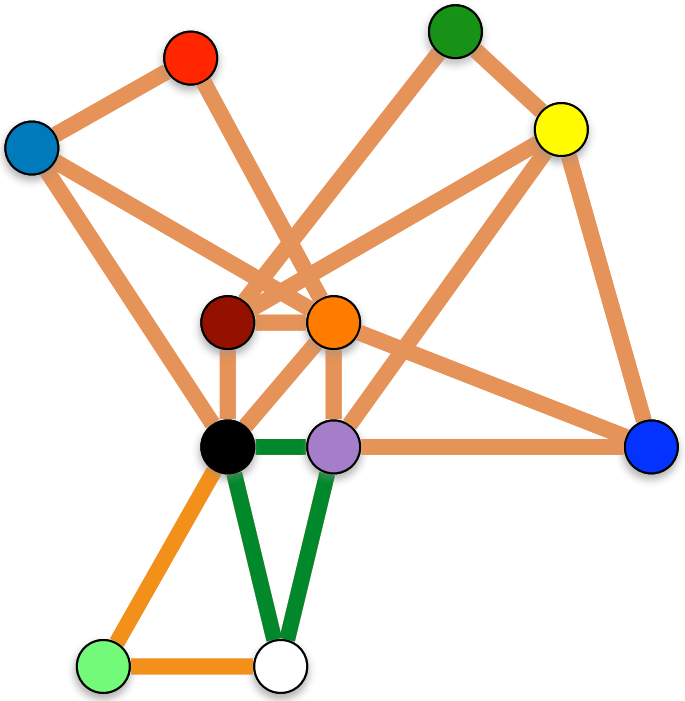
I_a =

●	●	○	●	s
0	1	0	1	2
1	0	0	0	1
0	1	0	0	1
0	0	1	0	1
0	0	0	1	1
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,b constrained by intersection with the sepset a∩b

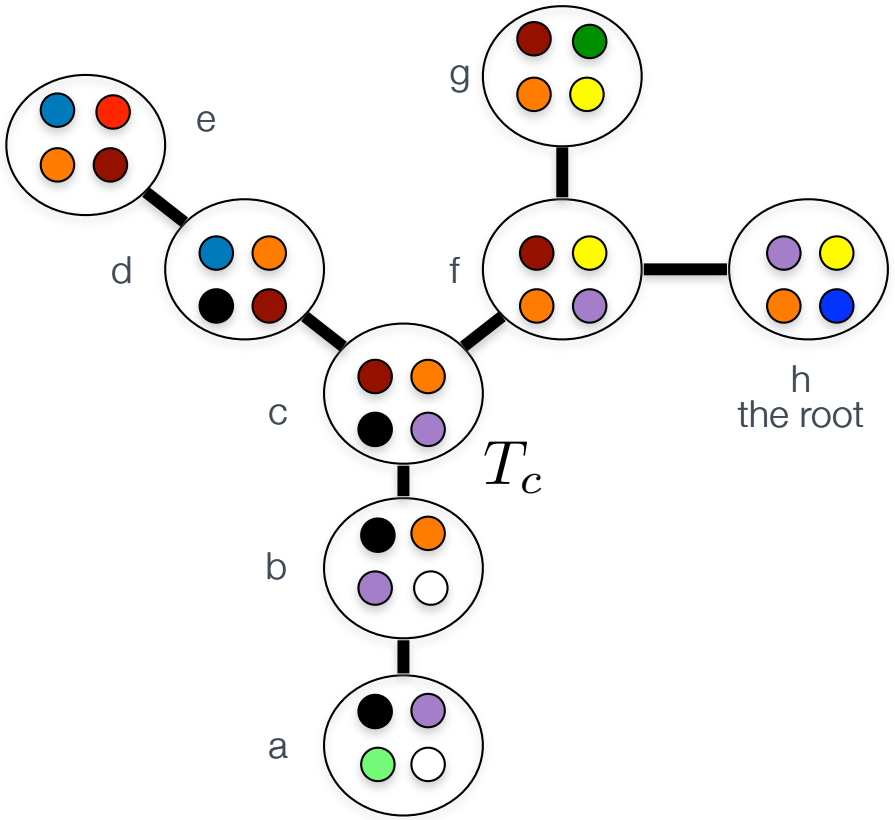


$I_{a \cap b}$ =

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

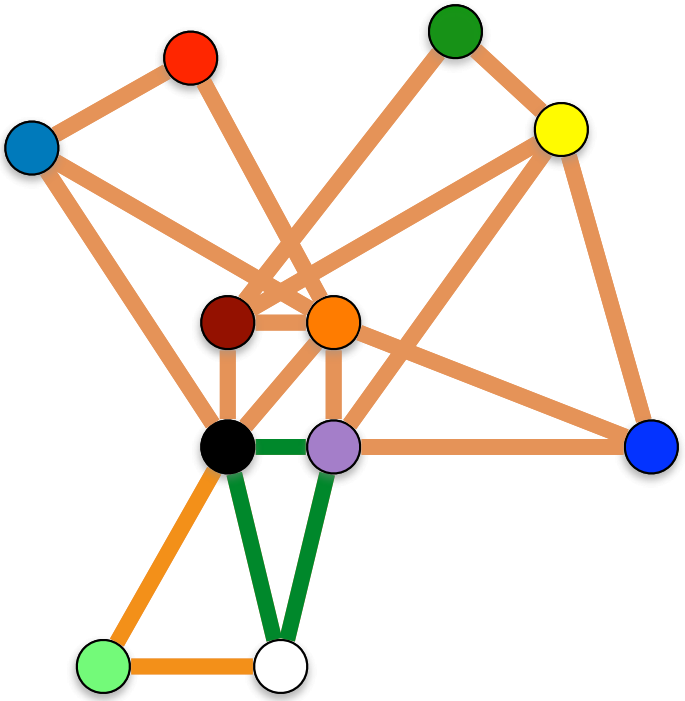
I_a =

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



JUNCTION TREE GUIDED DP - COMPUTING

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

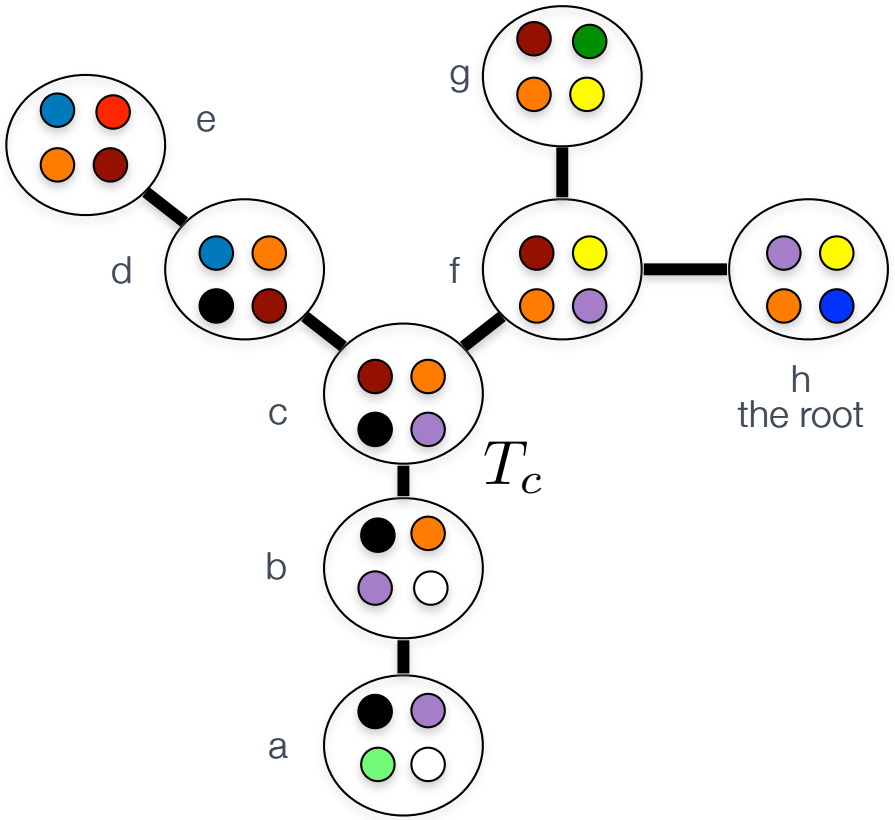


$I_{a \cap b} =$

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

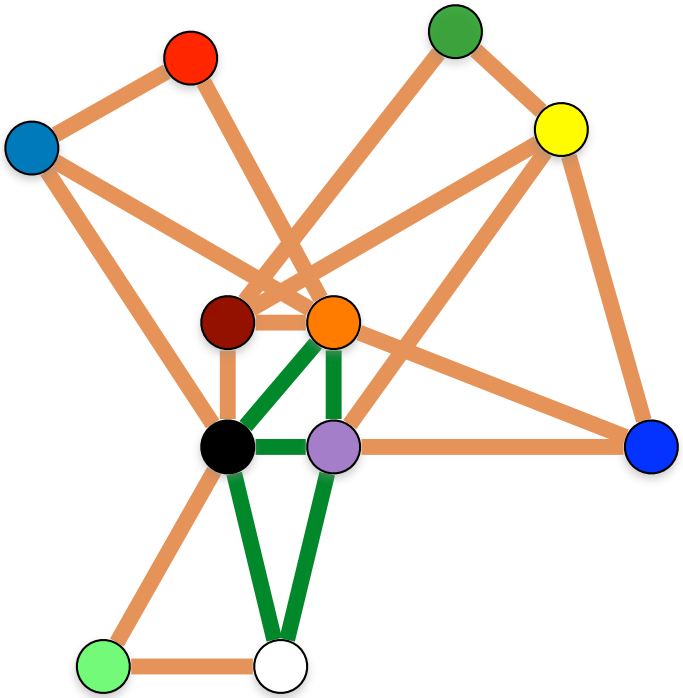
$I_a =$

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



JUNCTION TREE GUIDED DP - COMPUTING

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

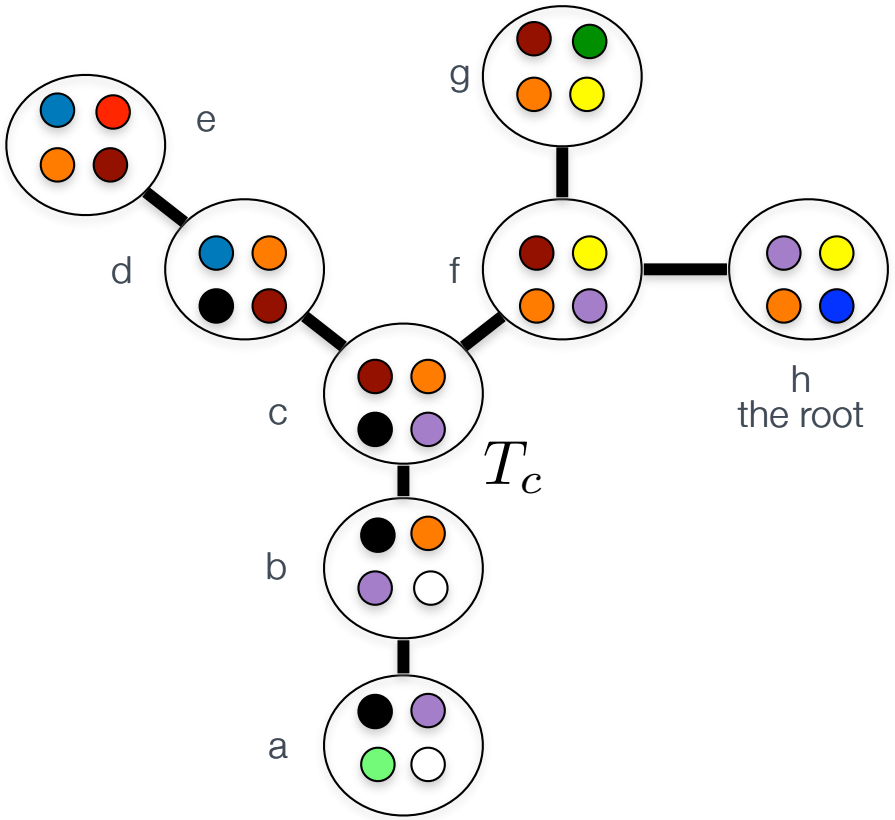


$I_{a \cap b} =$

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

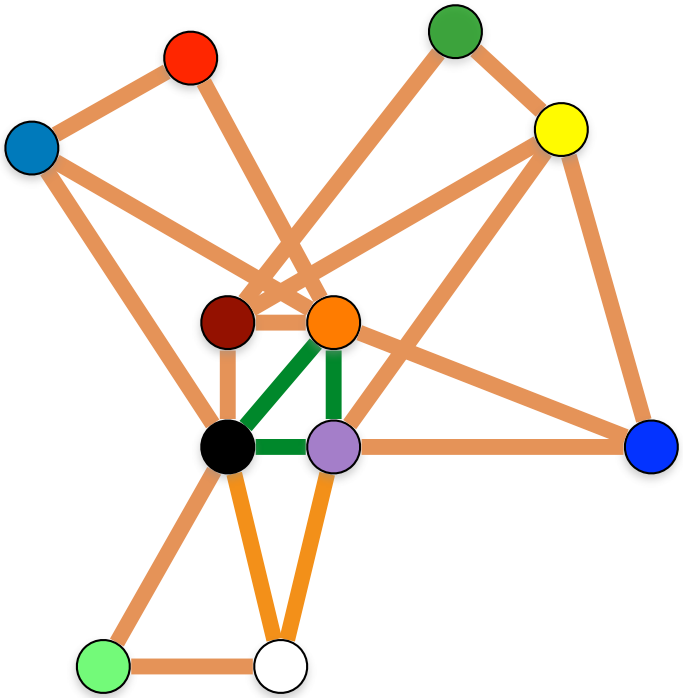
$I_b =$

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



JUNCTION TREE GUIDED DP - COMPUTING

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

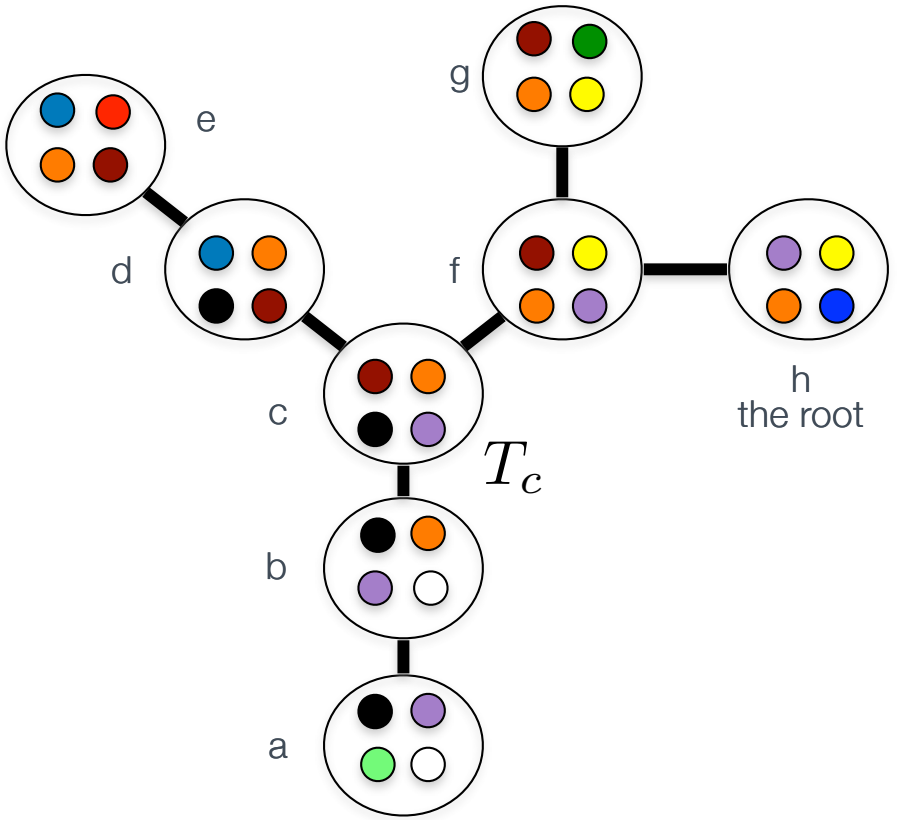


I_{bnc} =

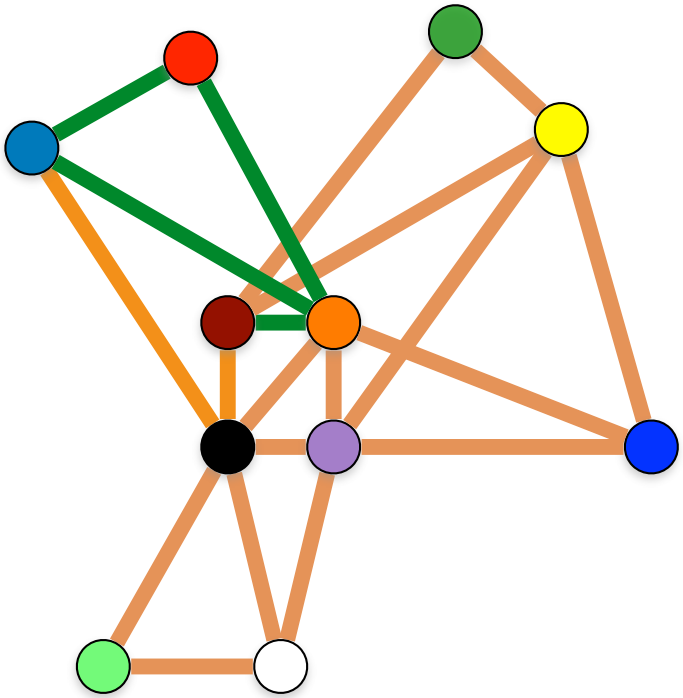
			s
1	0	0	1
0	1	0	2
0	0	1	2
0	0	0	1

I_b =

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

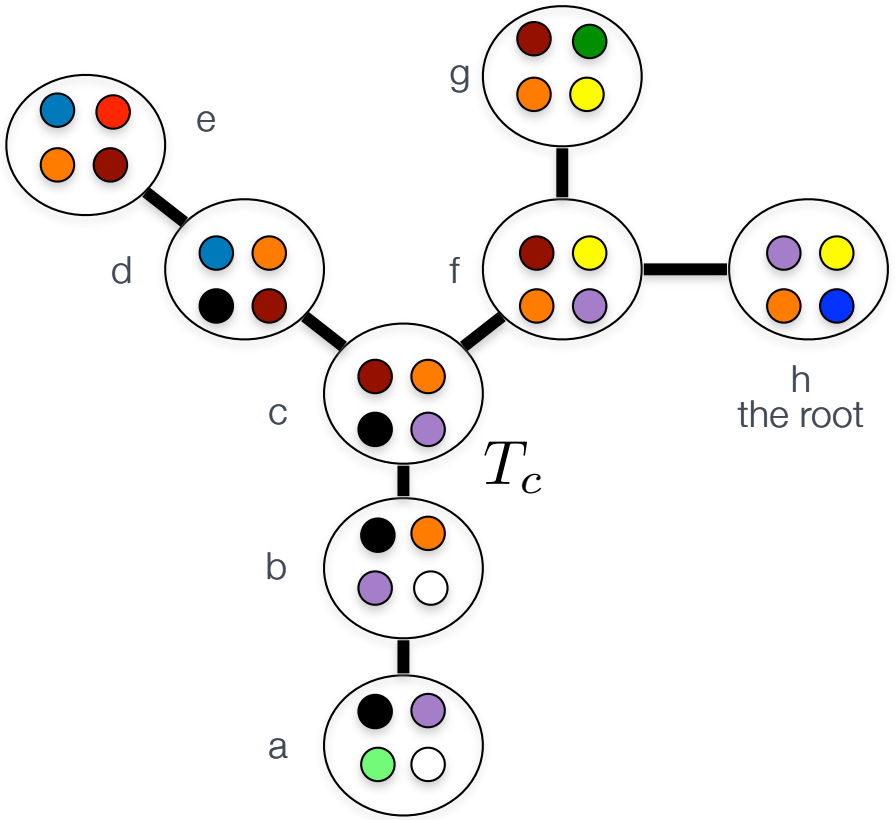


JUNCTION TREE GUIDED DP - COMPUTING



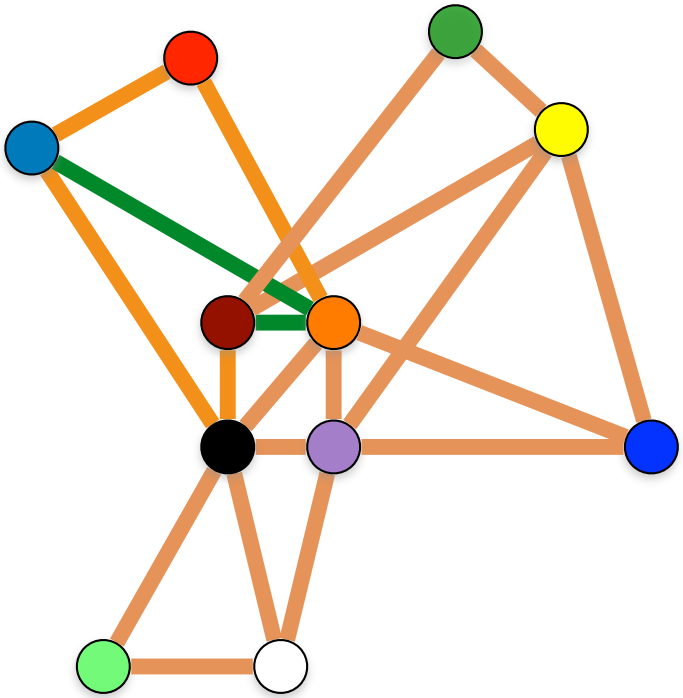
$I_e =$

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



JUNCTION TREE GUIDED DP - COMPUTING

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

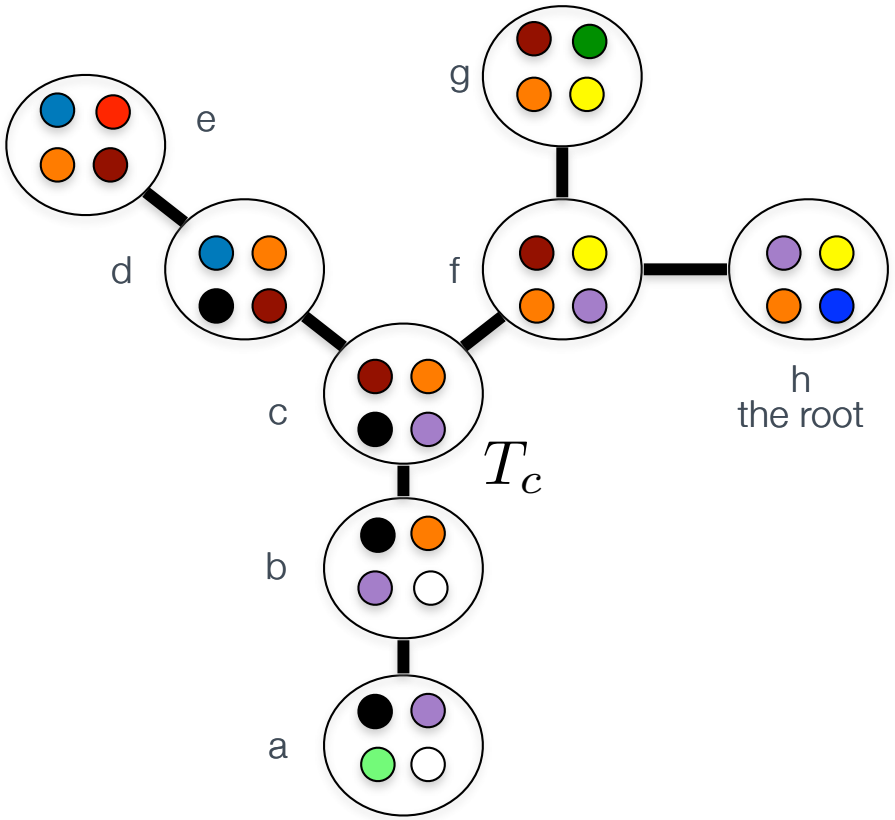


I_{end} =

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

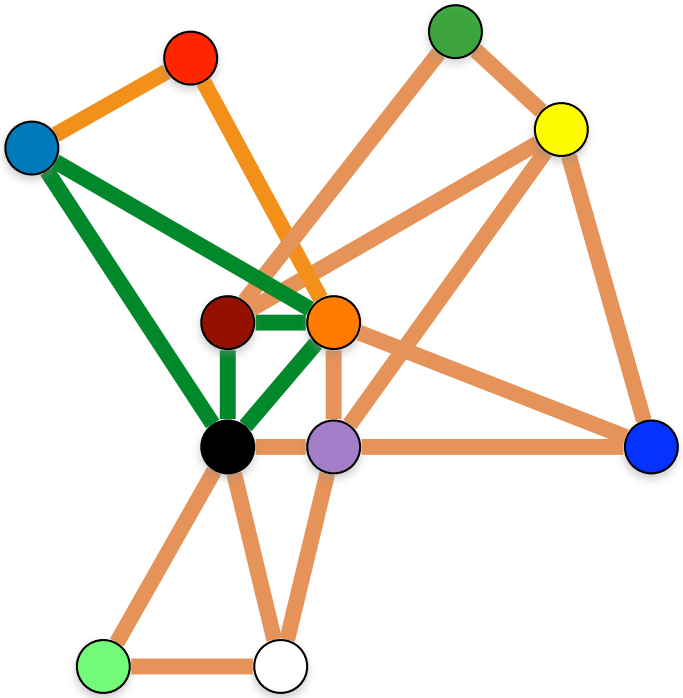
I_e =

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



JUNCTION TREE GUIDED DP - COMPUTING

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

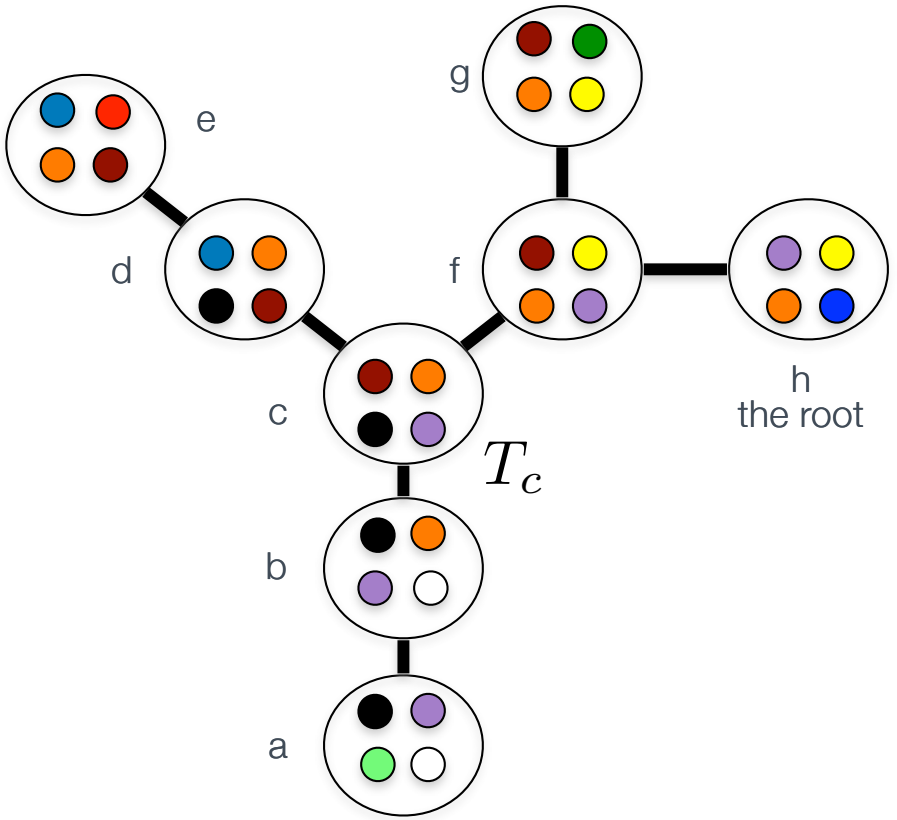


I_{end} =

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

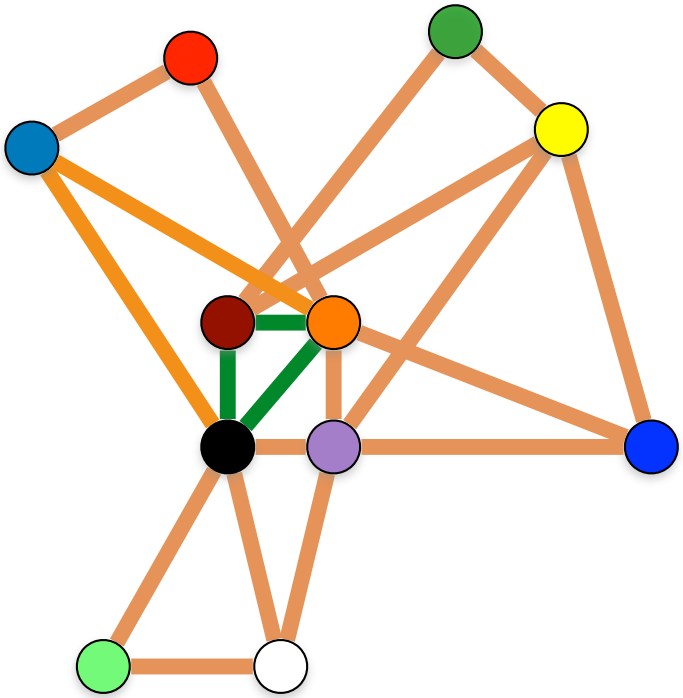
I_d =

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



JUNCTION TREE GUIDED DP - COMPUTING

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

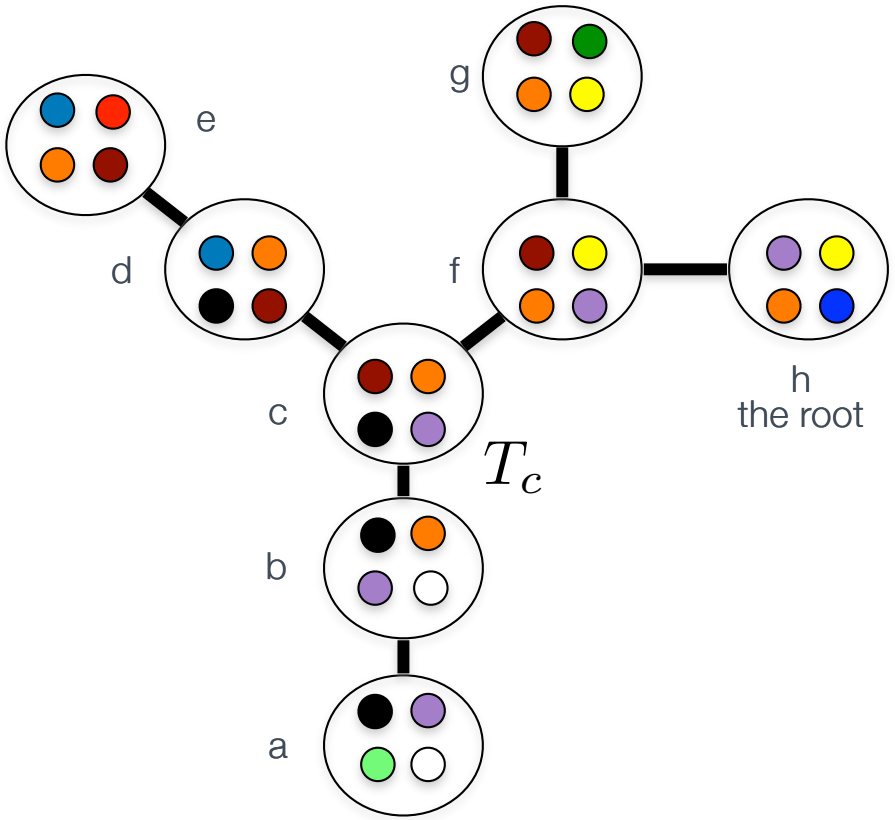


I_{cnd} =

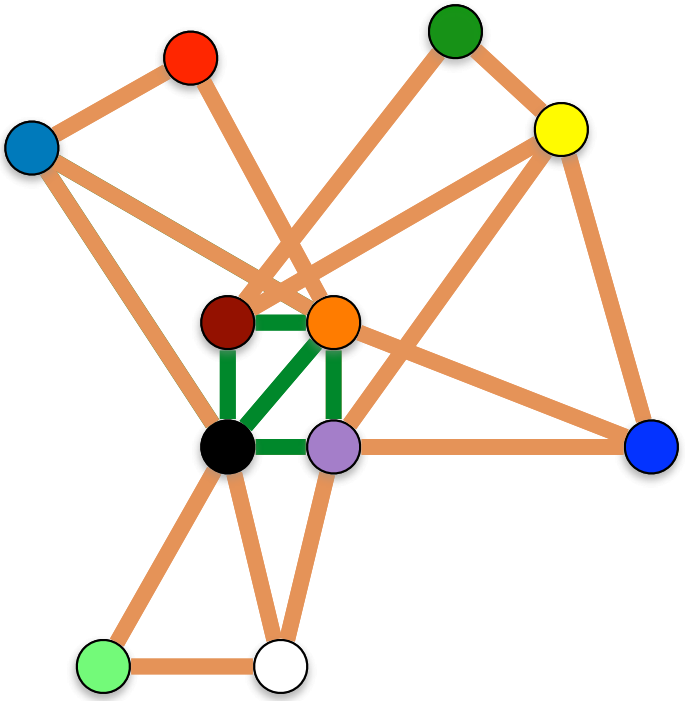
			s
1	0	0	2
0	1	0	1
0	0	1	2
0	0	0	1

I_d =

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



JUNCTION TREE GUIDED DP - COMPUTING



I_{cnd} =

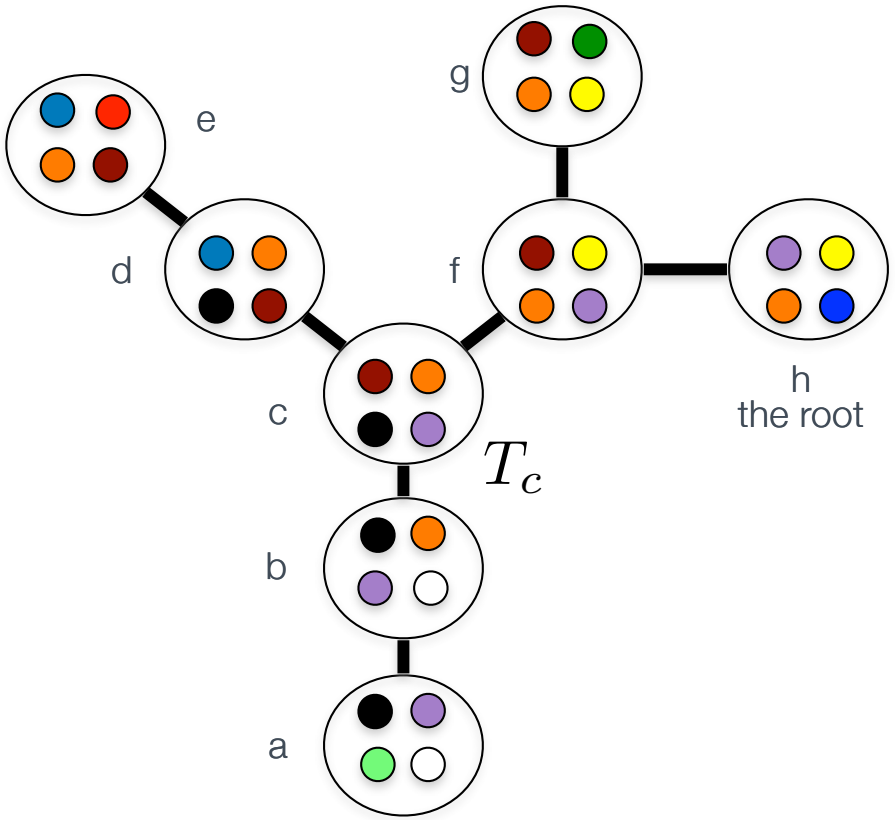
			s
1	0	0	2
0	1	0	1
0	0	1	2
0	0	0	1

I_{bnc} =

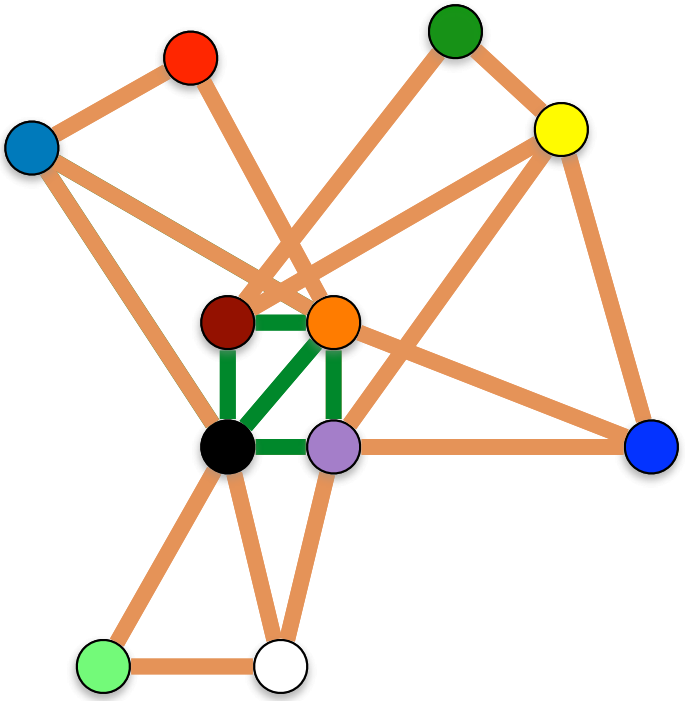
			s
1	0	0	1
0	1	0	2
0	0	1	2
0	0	0	1

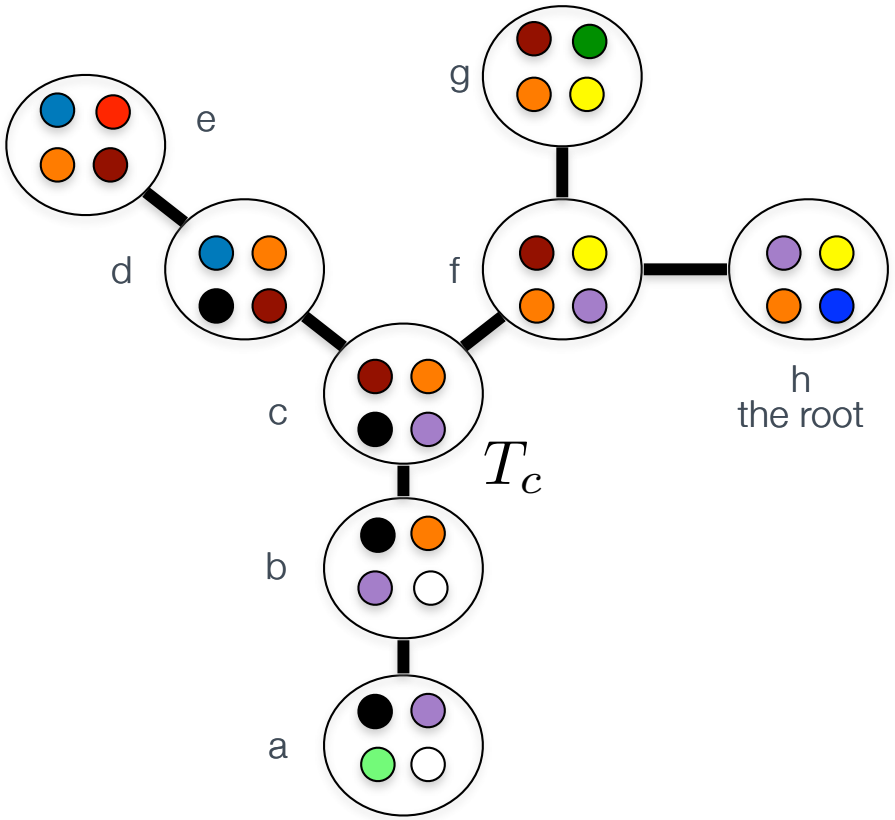
I_c =

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



JUNCTION TREE GUIDED DP - COMPUTING

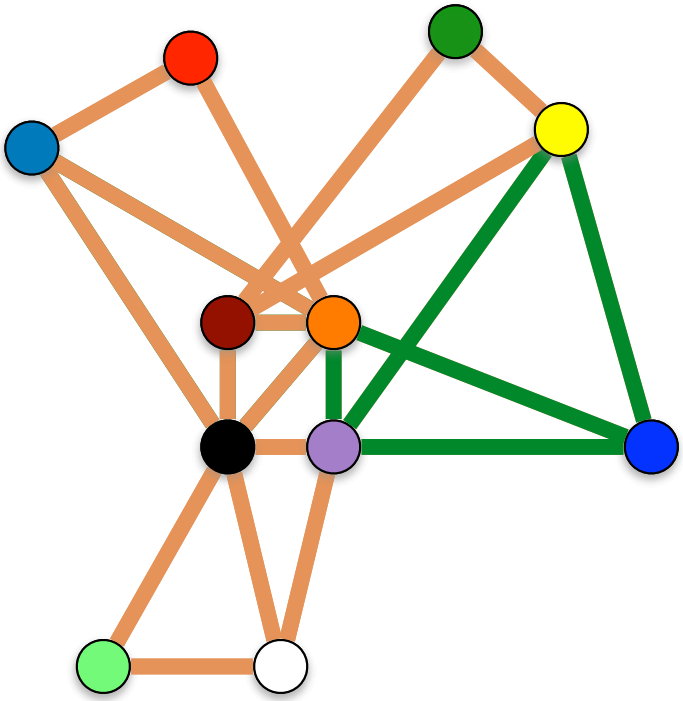




s, s' the 2 only children of t

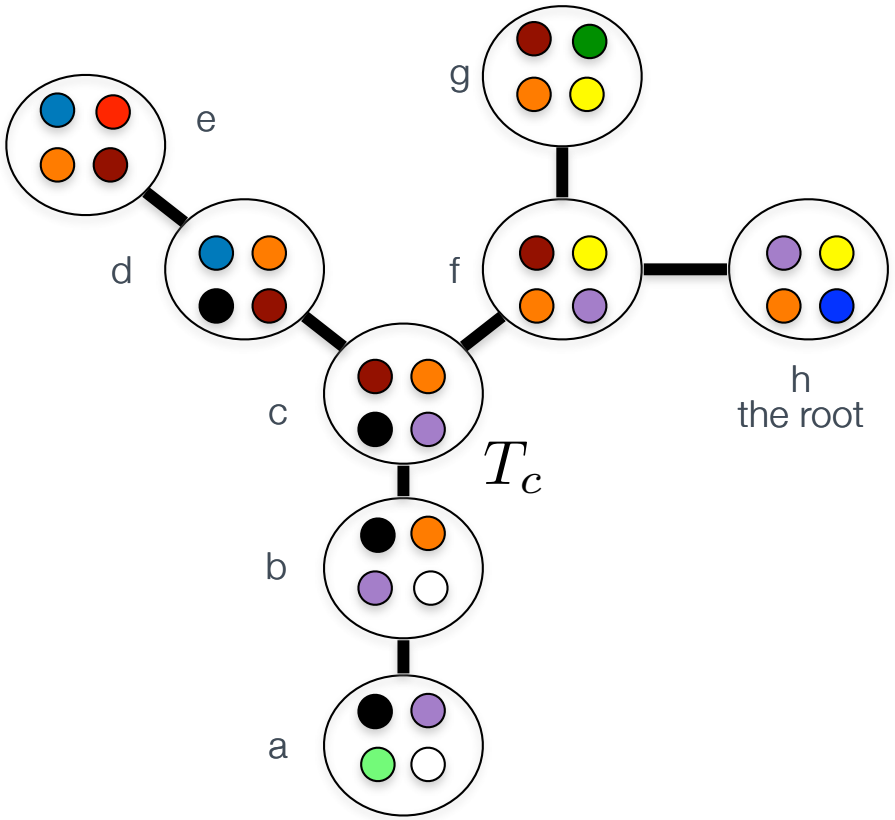
$$I_t(S) = \begin{cases} -\infty \text{ if } S \text{ not ind. set and otherwise the case below} \\ I_{st}(S \cap B(s)) + I_{s't}(S \cap B(s')) + |S \setminus (B(s) \cup B(s'))| - |S \cap B(s) \cap B(S')| \end{cases}$$

FINALLY TAKE MAXIMUM AT ROOT



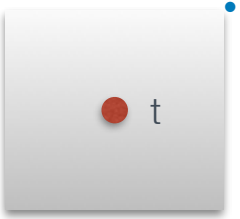
I_h =

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



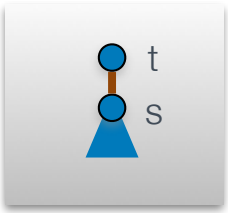
SUMMARY - IND. SET ALGORITHM

- 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



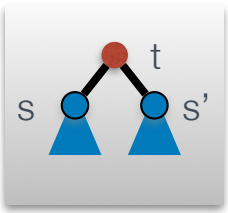
  - when at leaf t

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



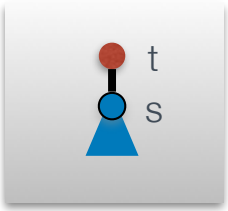
  - at edge st

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



  - 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

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

THE END