6.034 Final Examination

Fall 2002

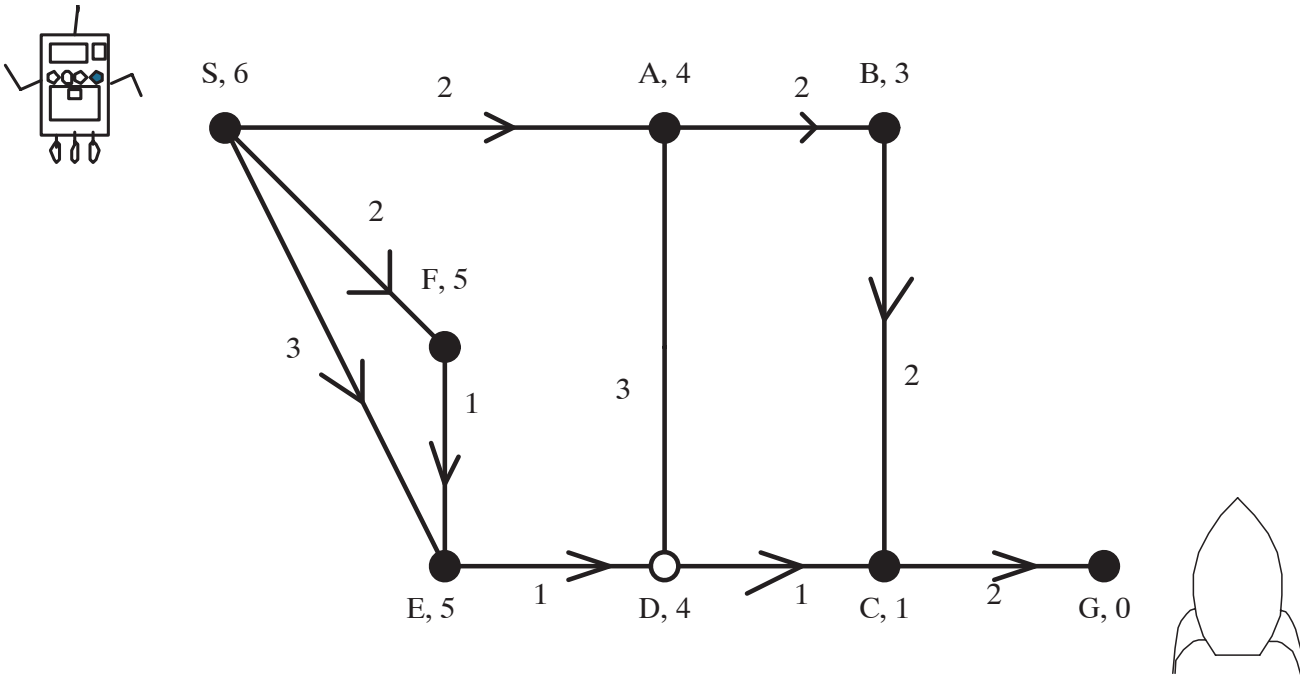
Name	
E-mail	

Caution: All the problems have easy parts. Several problems conclude with very challenging parts. Be sure to do the easy parts of all the problems.

Problem Number	Max	Score	Grader
Problem 1	16		phw rcb kk jb sl js as ou ml
Problem 2	22		phw rcb kk jb sl js as ou ml
Problem 3	14		phw rcb kk jb sl js as ou ml
Problem 4	21		phw rcb kk jb sl js as ou ml
Problem 5	13		phw rcb kk jb sl js as ou ml
Problem 6	14		phw rcb kk jb sl js as ou ml
Total	100		

Problem 1: Search (16 Points)

- Wallace, a robot, has finished his vacation on the moon and is about to head back to Earth in his rocket ship (located at position G below). Wallace must hurry to get to from position S to the rocket ship at position G. He has to navigate via the labeled landmarks. There is a blinking traffic light at D, which is relevant only to part G of this problem.



Note that the map is ***not drawn to scale***. All paths are one way, except the path connecting A to D. Distances between nodes are given by the numbers next to the links. Heuristics estimates of distance remaining appear next to the node names.

Assume the following for **every** search method:

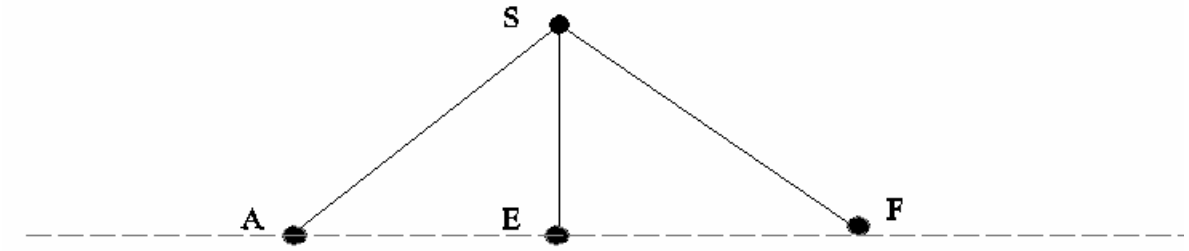
- None of the search methods generate paths with loops.
- Whenever a partial path is extended, the method checks to see if the path has already reached the goal, and if it does, search terminates. No other check is made to see if the goal has been reached.

Break ties as follows, as usual:

- First, alphabetically, by the head node in the path, the one furthest from the start node.
- Then, front-to-back order in queue.

Part A (2 Points)

Draw the complete, loop-free search tree represented by the graph shown above.



Part B (2 Points)

What is the order of node extension if the robot uses ***depth-first*** search with backup, but ***with neither an enqueued list nor an extended list***? What is the path that he will take using this method?

Extended nodes: Path:

Part C (2 Points)

Now using ***breadth-first*** search ***with an enqueued list***, in what order are nodes extended? What is the path taken?

Extended node sequence: Path:

Part D (2 Points)

Using ***branch and bound*** search (no use of an heuristic estimate of distance remaining), looking for ***the shortest path, with an extended list***, in what order are nodes extended? What is the path taken? Assume new paths are added to the front of the queue.

Extended node sequence: Path:

Part E (2 Points)

Using ***A**** search (with the heuristic estimates of distance remaining, as given) ***with an extended list***, looking for ***the shortest path,*** in what order are nodes extended? What is the path taken? Assume new paths are added to the front of the queue.

Extended node sequence: Path:

.

What is the length of the path found by A*? | |

Part F (2 Points)

What is the shortest path from S to G?

What is its length?

Part G (4 Points)

The robot notices that A* found the fastest path, but not the shortest one. He knows that the delay at the blinking light is the same time required to travel ***two units*** of length. Perhaps some of the heuristic estimates given on the diagram are inadmissible from the point of view of path length, but admissible from the point of view of fastest traversal time.

Circle all of the following which have admissible heuristic distances from the point of view of path length. Draw an x through the others.

- A
- B
- E
- No node
- All nodes

Circle all of the following which have admissible heuristic distances from the point of view of traversal time. Draw an x through the others.

- A
- B
- E
- No node
- All nodes

Problem 2: Constraint Propagation (22 Points)

After a semester of 6.034, you are a bit troubled that your perception has been permanently skewed. Wherever you go, you cannot help but see 6.034 ideas. You remember a recent example of this was at the MIT Dance Troupe concert, *Rhapsody*, just a few weeks ago…

(Flashback) While at the show, you notice that little skits are sometimes needed in between dances to take up time so dancers can change costumes. Under your breath, you mutter, "They shouldn't need to do this – finding an acceptable ordering of dances is just constraint propagation." Momentarily horrified, you quickly glance around to make sure no one heard you. Relieved to see the audience watching the skit, you decide to solve the ordering problem for a simplified concert -- one act with six dances. You label the slots for the dances {1, 2, 3, 4, 5, 6} and label the dances themselves {A, B, C, D, E, F}.

Part A: Pure Backtracking (8 points)

"First," you whisper to yourself, "each dance can only be in one slot." You realize this is a binary constraint across all pairs of slots.

First constraint:	No dance can be in more than one slot

"Second, the dances in the first and last slots need to be *showstoppers*." You realize this is a unary constraint for slots 1 and 6:

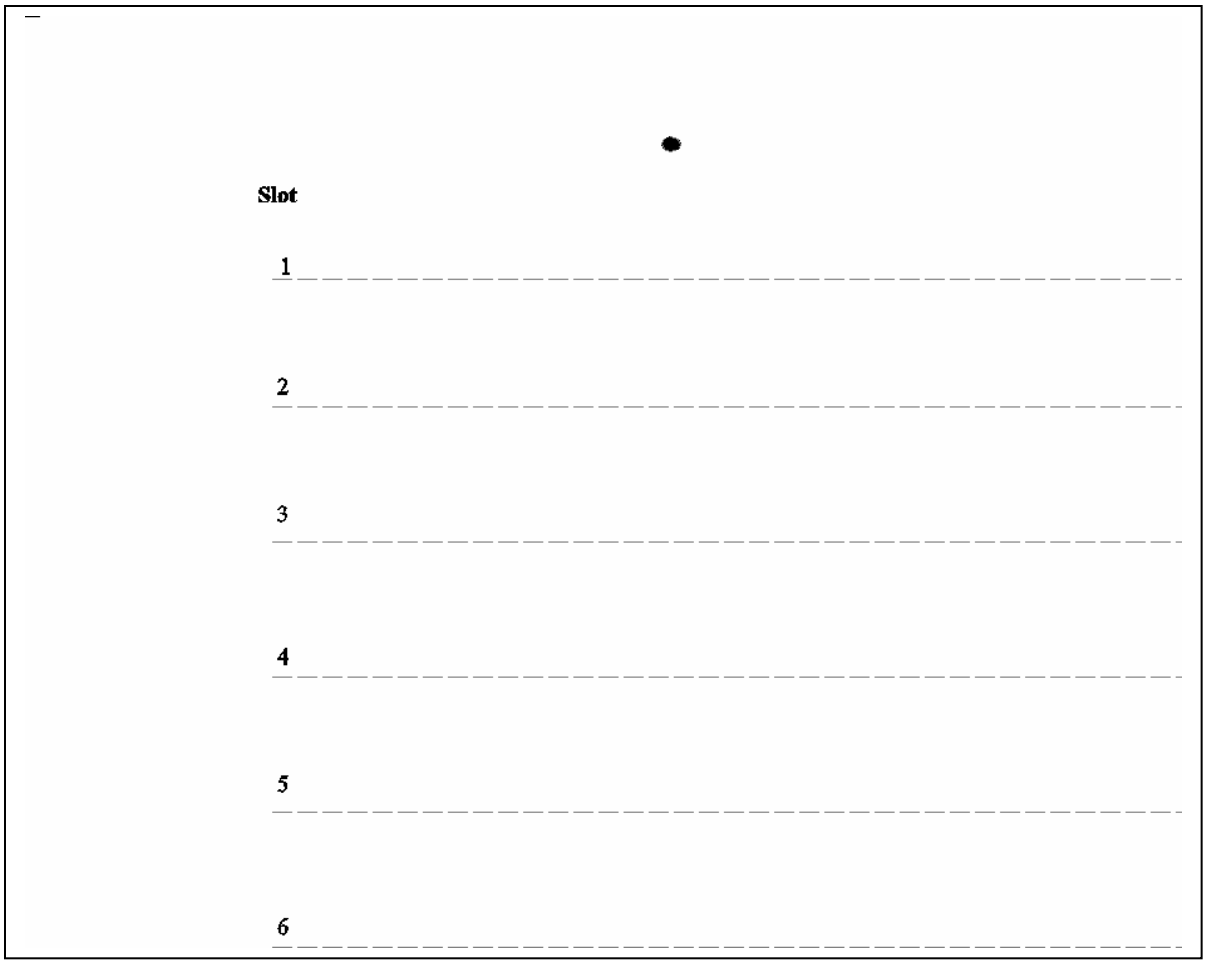
Second constraint:	Dances in slots 1 and 6 must be showstoppers

Now, you are ready to find an ordering for the dances. **Using the following showstopper data, do backtracking to find the first set of assignments that is consistent with these two constraints. (As usual, use alphabetical and numerical ordering to choose the order for variables and values.) Show your search tree.**

Dance	Showstopper?
A	Yes
B	
C	Yes
D	Yes
E	
F	

For your convenience, all constraints and tables are repeated on a tear off sheet at the end of the examination.

Search tree:



Solution:

Slot	1	2	3	4	5	6
Dance						

Part B: Backtracking with Forward Checking (8 points)

Pleased with your progress, you say, "Third, dances that share dancers should not follow each other." You realize this is a binary constraint across all pairs of consecutive dances.

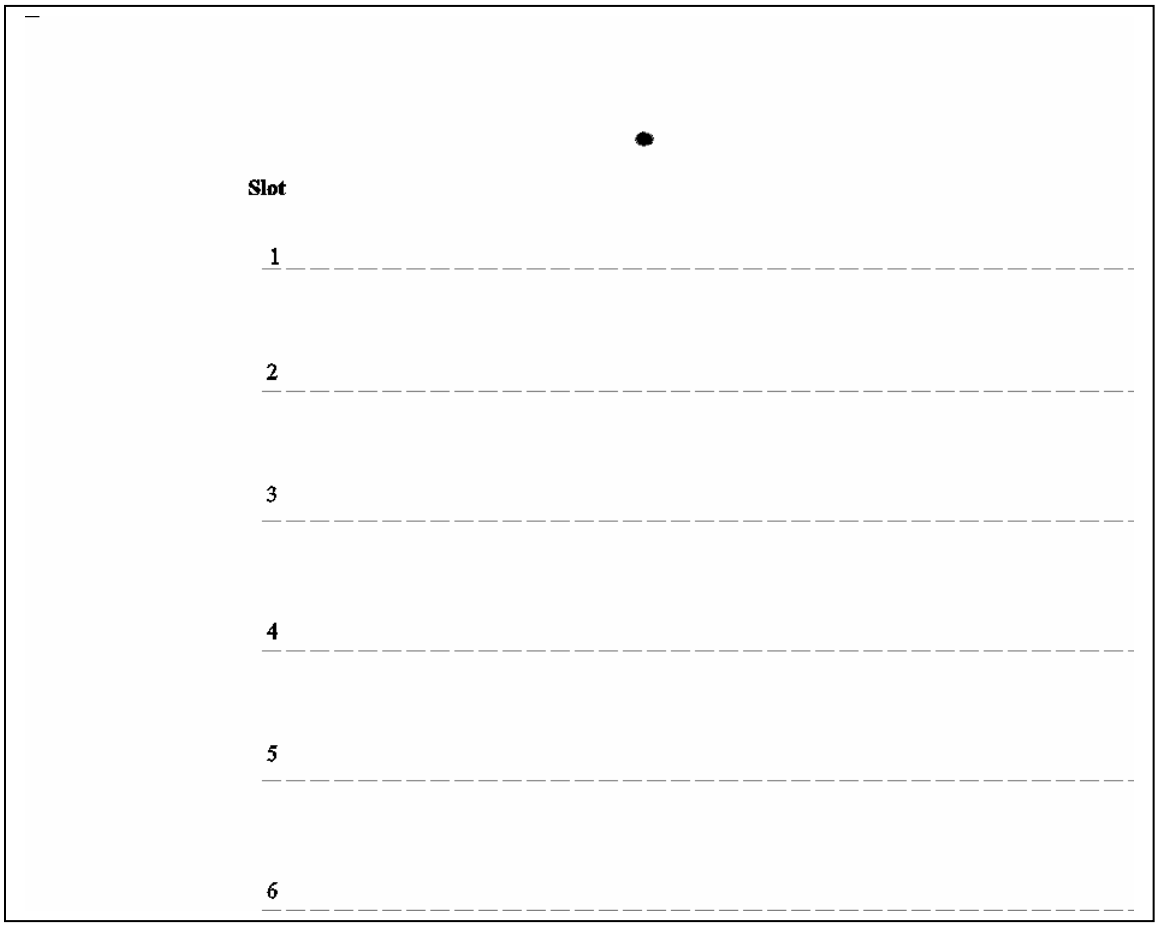
Third constraint:	Consecutive dances must not share dancers

You also know that the following dances share dancers:

	A	B	C	D	E	F
A			Shares			
B			Shares		Shares	
C	Shares	Shares		Shares		
D			Shares			Shares
E		Shares				Shares
F				Shares	Shares	

Using all three constraints, do backtracking with forward checking to find the first valid set of assignments. Show your search tree.

Search tree:



Solution:

Slot	1	2	3	4	5	6
Dance						

Part C: Optimal Constraint Propagation (6 points)

WARNING: ***Part C may be long and/or difficult; consider doing other problems first.***

Having accomplished your original goal, you clap out loud with excitement. Luckily, the rest of the audience is also clapping – for the skit performers. Dr. Koile, a dance enthusiast, happens to be sitting immediately behind you. Just as the next dance starts, she leans forward, gestures toward your notes, and whispers, "It is better for dances to alternate among the various dance genres." Looking at your show order, you realize it does not vary at all! How positively … engineered!

At that moment, your TA swaggers out on stage; you resolve to incorporate aesthetics into your calculations and find an optimal show order. "If only I had a heuristic for the goodness of genre transitions," you murmur. Dr. Koile hands you the following data:

The penalties genre transitions:

	Ballet	**Hip Hop**	**Jazz**	**Swing**
Ballet	10	5	1	2
Hip Hop	5	10	1	2
Jazz	1	1	10	3
Swing	2	2	3	10

Genres of the dances:

Dance	**A**	**B**	**C**	**D**	**E**	**F**
Genre	Hip Hop	Hip Hop	Jazz	Swing	Ballet	Jazz

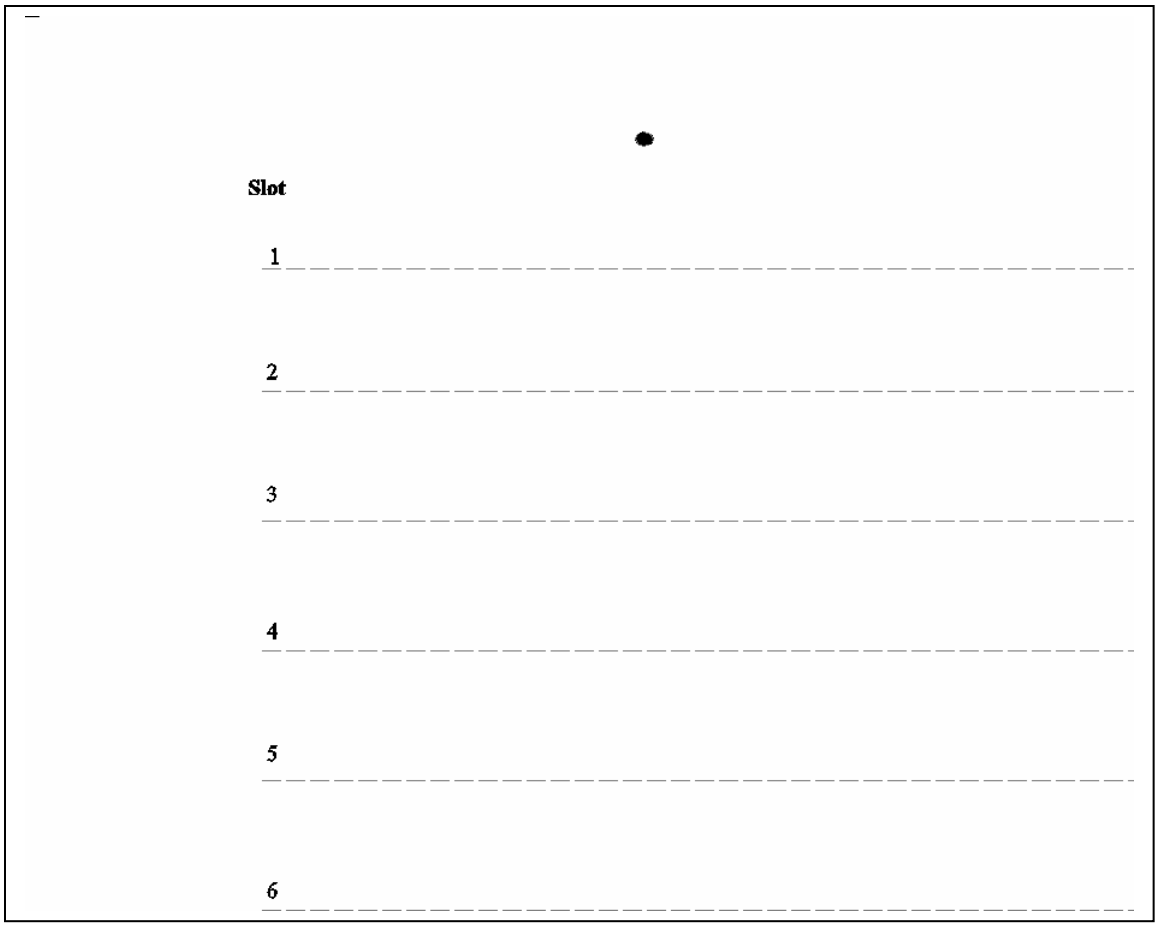
Dr. Koile also says that Justin is in dance A. This means dance A *has* to be first.

Using the data from these new tables, the fact that dance A has to be first, and the constraints from before, find the optimal ordering of dances (minimum total penalty). Show your search tree on the following page.

Hint: Consider using a different search algorithm than depth-first.

Please draw your search tree for Part C here. (The details of the previous parts of the problem are repeated on a tear-off sheet at the back of the exam.)

Search tree:



Solution:

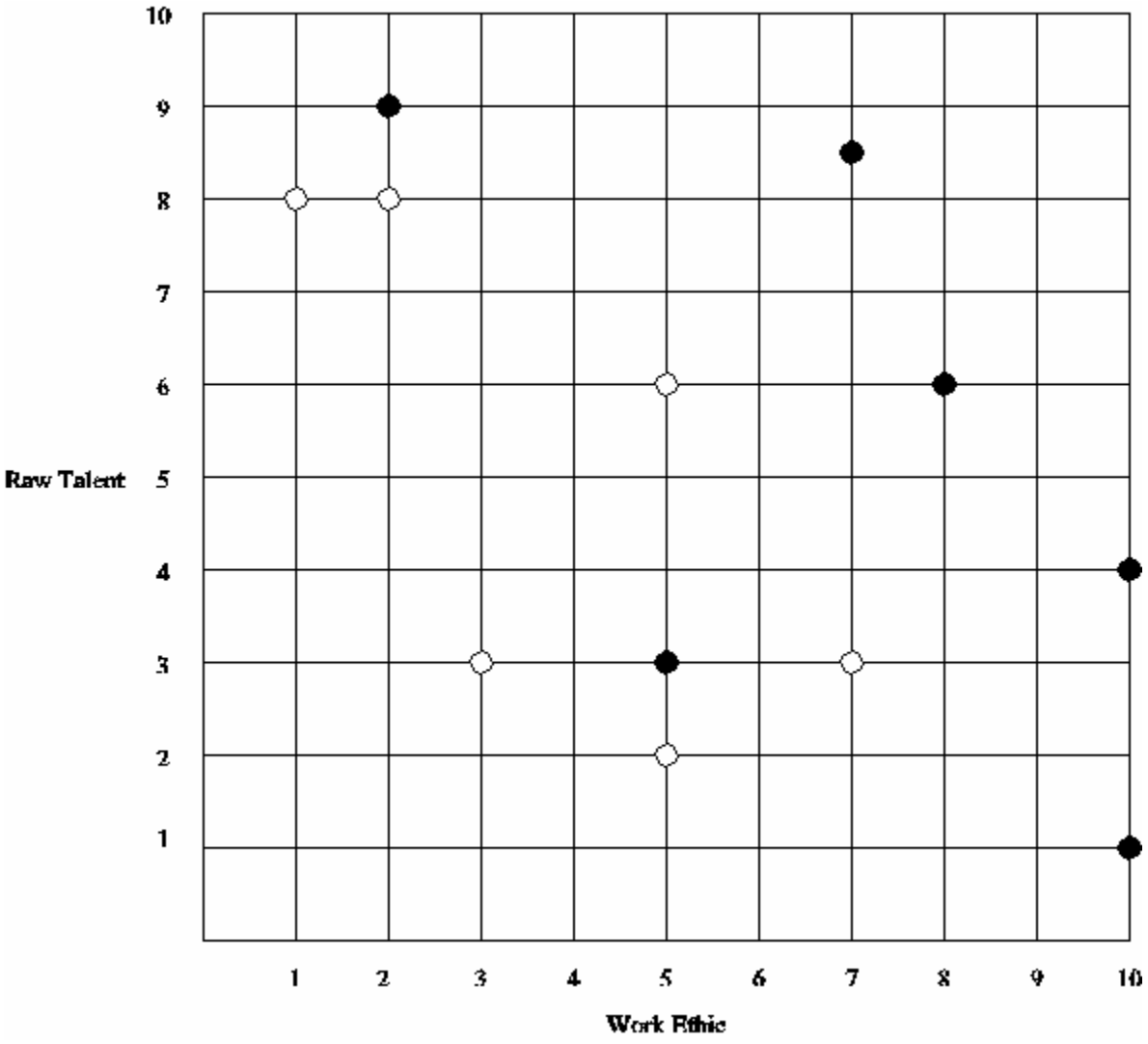
Slot	1	2	3	4	5	6
Dance						

Problem 3: Classification (14 Points)

Part A: Nearest Neighbors (6 Points)

The 6.034 staff has decided to launch a search for the newest AI superstar by hosting a television show that will make one aspiring student an *MIT Idol*. The staff has judged two criteria important in choosing successful candidates: work ethic (W) and raw talent (R). The staff will classify candidates into either potential superstar (black dot) or normal student (open circle) using a nearest-neighbors classifier.

On the graph below, draw the decision boundaries that a 1-nearest-neighbor classifier would find in the R-W plane.



Part B: Identification Trees (4 Points)

Part B1 (2 Points)

Now, leaving nearest neighbors behind, you decide to try an identification-tree approach. In the space below, you have two possible initial tests for the data. Calculate the average disorder for each test. Your answer may contain $\log_2$ expressions, but no variables. The graph is repeated below.

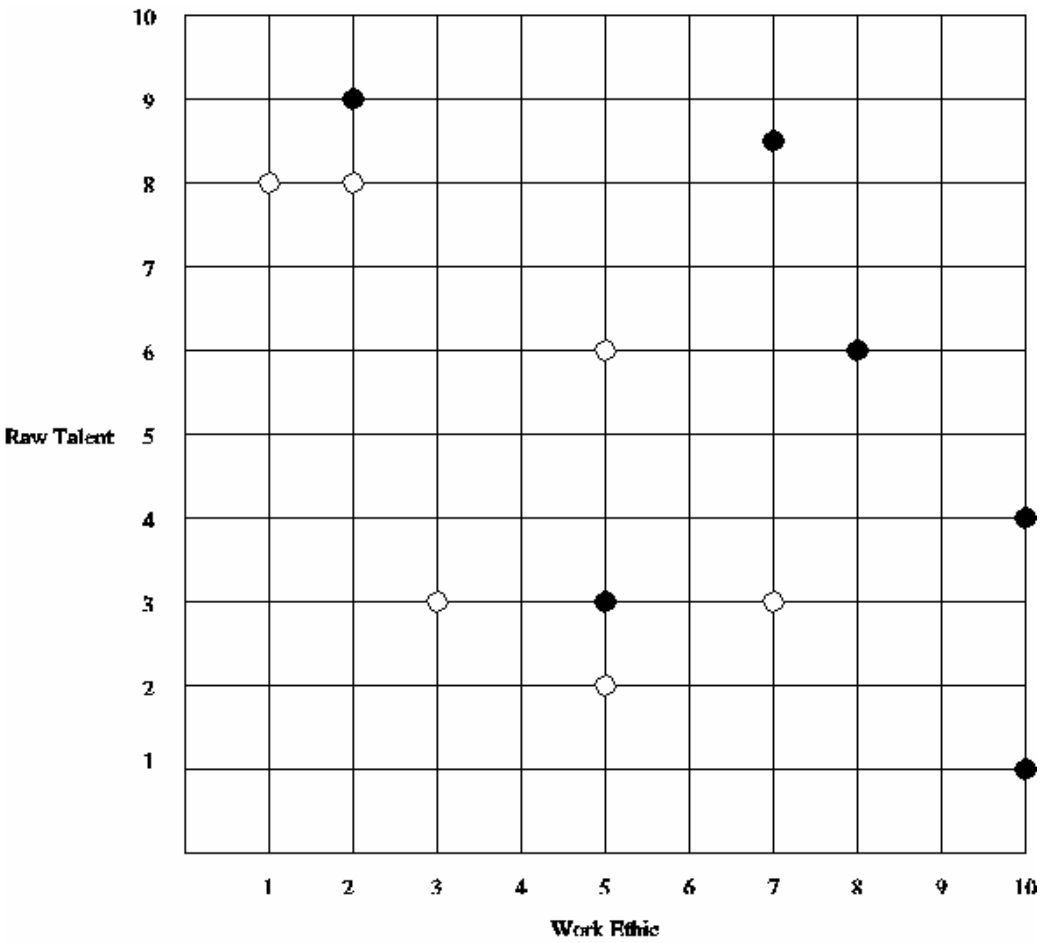
Test A: R > 5:

Test B: W > 6:

Part B2 (2 Points)

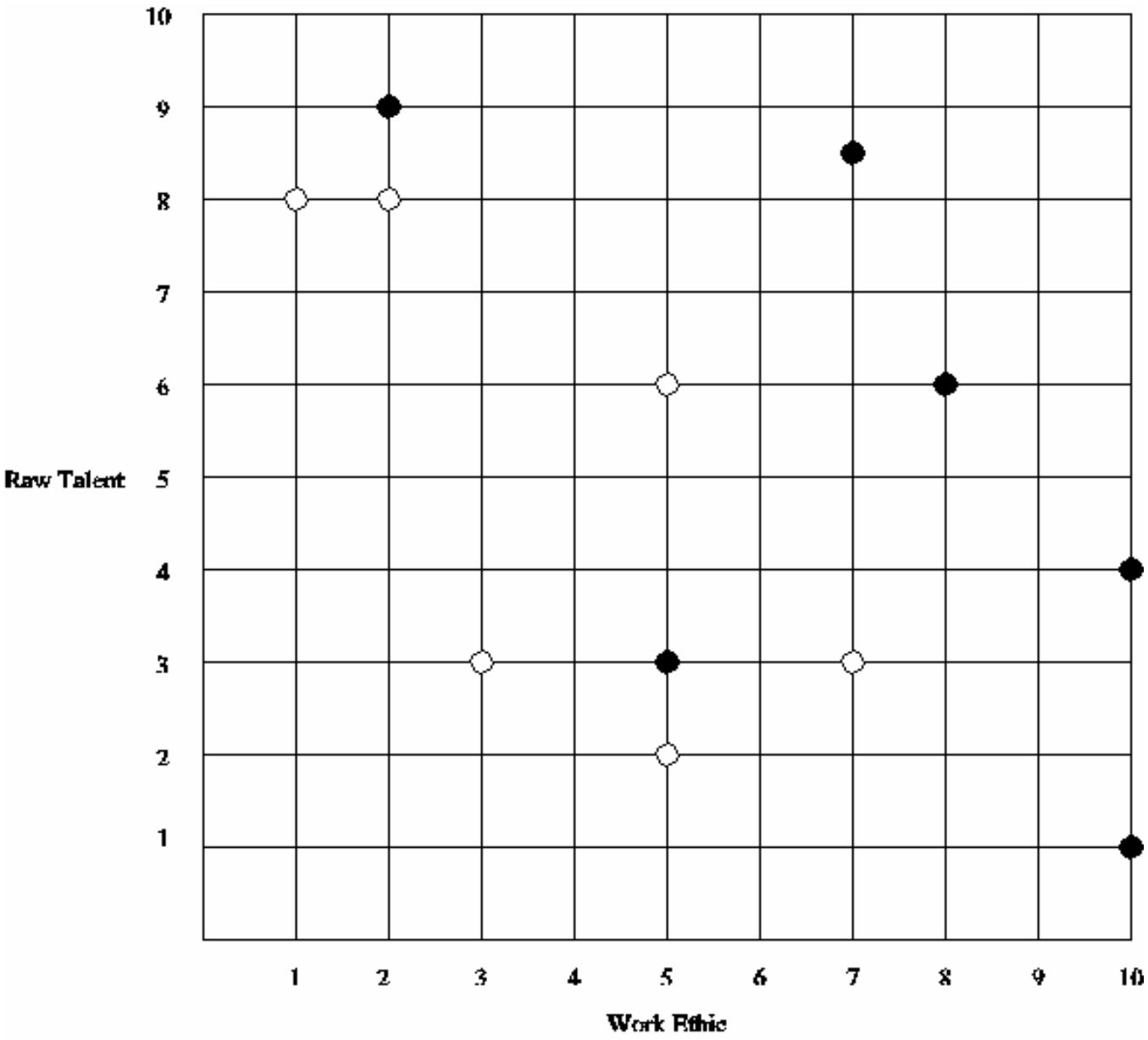
Now, indicate which of the two tests is chosen first by the greedy algorithm for building identification trees.

We include a copy of the graph below for your scratch work.



Part C: Identification Trees (4 Points)

Now, assume R > 5 is the first test selected by the identification-tree builder (which may or may not be correct). Then, draw in all the rest of the decision boundaries that would be placed (correctly) by the identification-tree builder:



Problem 4: Neural Networks (21 Points)

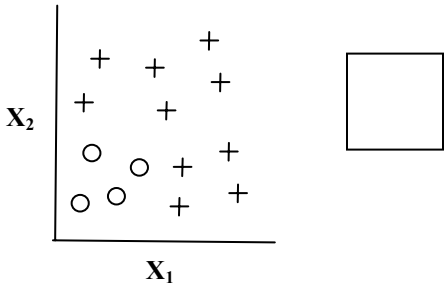
Part A: Perceptrons (11 Points)

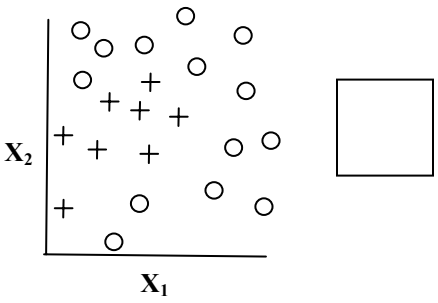
Part A1 (3 Points)

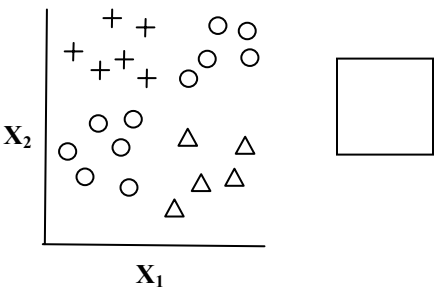
For each of the following data sets, draw the minimum number of decision boundaries that would completely classify the data using a perceptron network.

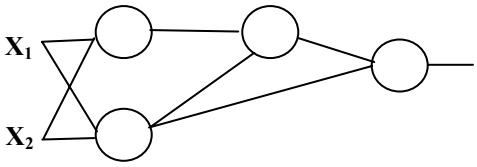
Part A2 (3 Points)

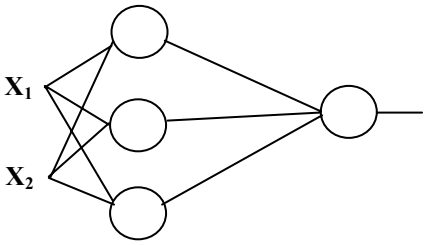
Recall that the output of a perceptron is 0 or 1. For each of the three following data sets, select the perceptron network with the fewest nodes that will separate the classes, and write the corresponding letter in the box. ***You can use the same network more than once.***

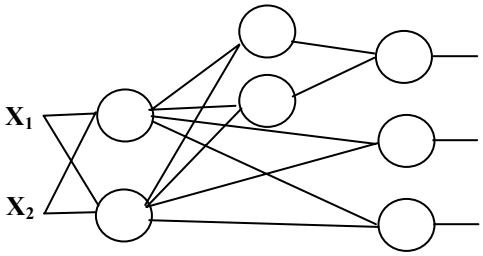


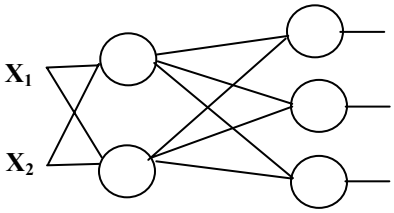


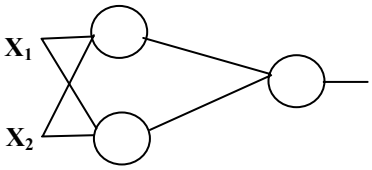


A 

B 

C 

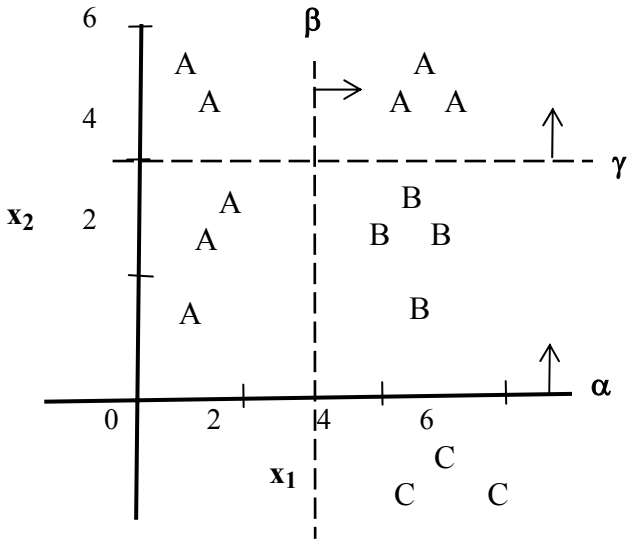
D 

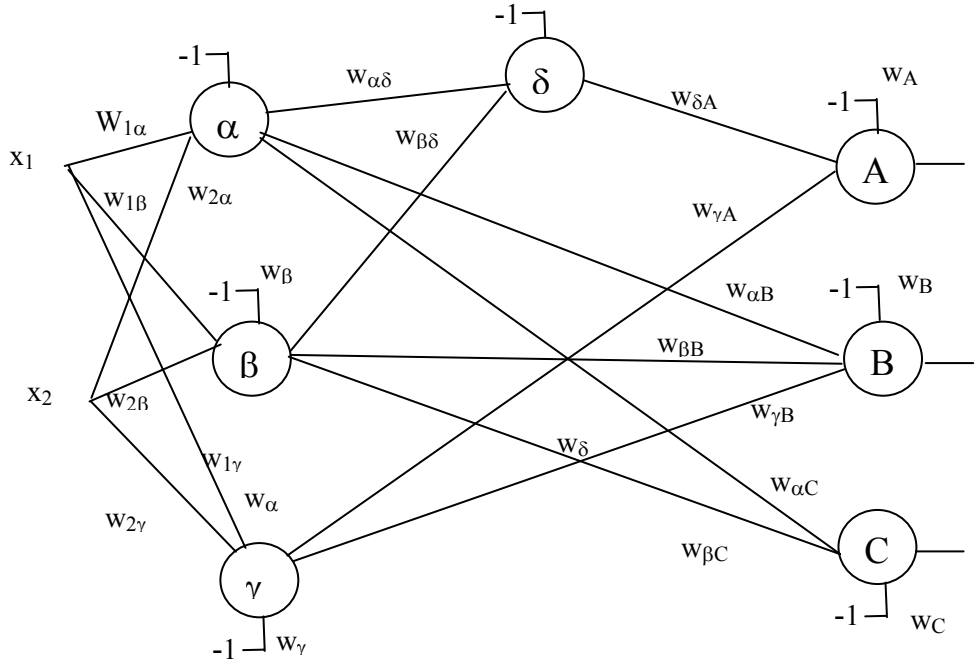
E 

Part A3 (5 Points)

Fill in the missing weights for each of the nodes in the perceptron network on **the next page**.
Make the following assumptions:

- Perceptrons output 0 or 1
- A, B, C are classes
- The lines labeled α (same as abscissa, the x_1 axis), β, γ represent decision boundaries
- The directions of the arrows shown on the graph represent the side of each boundary that causes a perceptron to output 1.





α	
$w_{1\alpha}$	
$w_{2\alpha}$	1
w_{α}	

β	
$w_{1\beta}$	1
$w_{2\beta}$	
w_{β}	

γ	
$w_{1\gamma}$	
$w_{2\gamma}$	1
w_{γ}	

δ	
$w_{\alpha\delta}$	1
$w_{\beta\delta}$	
w_{δ}	

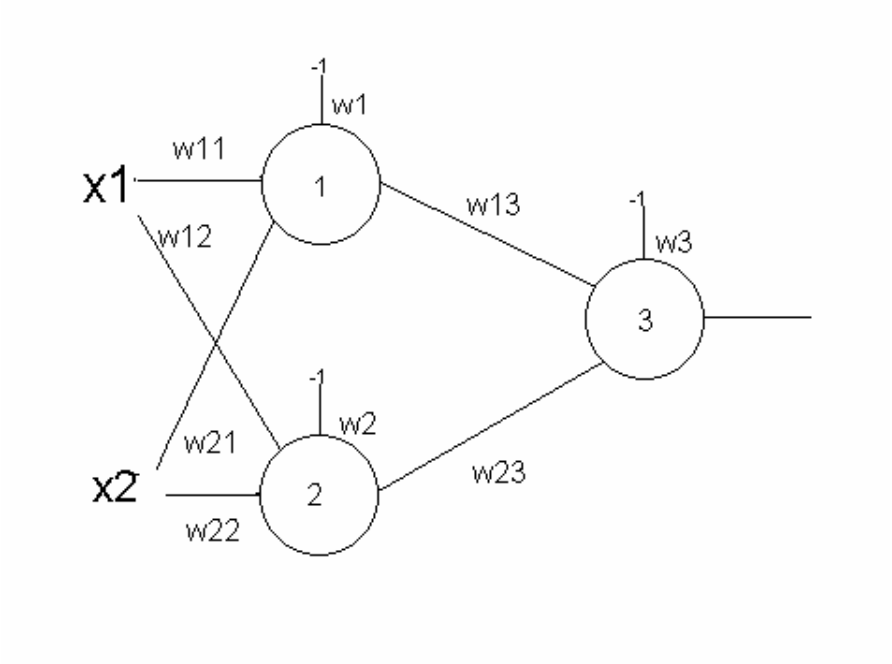
A	
$w_{\delta A}$	1
$w_{\gamma A}$	
w_{A}	

B	
$w_{\alpha B}$	1
$w_{\beta B}$	1
$w_{\gamma B}$	
w_{B}	

C	
$w_{\alpha C}$	
$w_{\beta C}$	1
w_{C}	

Part B: Negative sigmoids (10 Points)

The following sigmoid network has 3 units labeled 1, 2, and 3. All units are ***negative sigmoid*** units, meaning that their output is computed using the equation $n(z) = -\frac{1}{1+e^{-z}}$, which differs from the standard sigmoid by a minus sign. The equation for the derivative of $n(z)$ is: $\frac{dn(z)}{dz} = n(z)(1+n(z))$. Additionally, this network uses a ***non-standard error*** function $E = \frac{1}{2}(2y^* - 2y)^2$.



Part B1: Forward propagation (4 Points)

Using the initial weights provided below, and the input vector [x1, x2] = [2, 0.5], compute the output at each neuron after forward propagation. *Use the negative sigmoid values given in the table on the tear-off sheet at the end of the exam in your computation.*

Weight	w1	w11	w12	w13	w2	w21	w22	w23	w3
Value	0.5	1	1	0.5	0.5	1	1	0.25	0.5

y1 (output at neuron 1)

y2 (output at neuron 2)

y3 (output at neuron 3)

Part B2: Backward propagation (6 Points)

Using a **learning rate of 1**, and a **desired output of 0**, backpropagate the network by computing the δ values for nodes 2 and 3, and write the new values for the selected weights in the table below. Assume the initial values for the weights are as specified in Part B1, and assume the following values for the neuron outputs:

output at node 1, **y1 = -1.0**
output at node 2, **y2 = -1.0**
output at node 3, **y3 = -0.2**

Note: some helpful formulas appear on the tear-off sheet at the end of the examination.

Express δ_2 and δ_3 in terms of derivative-free expressions.

δ_3

$\delta 2$

Express the weights in terms of δs and numbers.

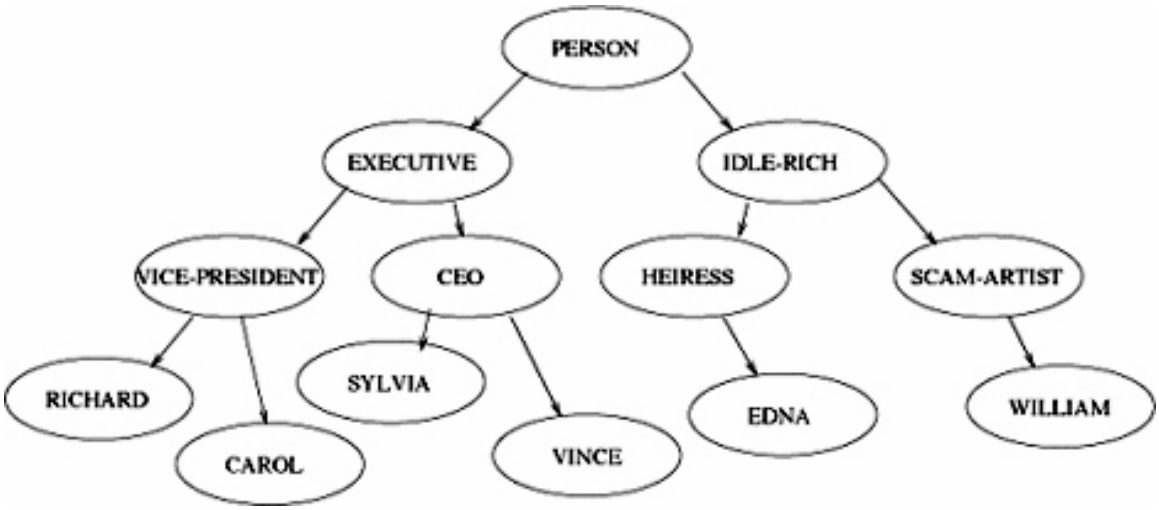
Weight	w22	w3
Value		

Problem 5: Near-Miss (13 Points)

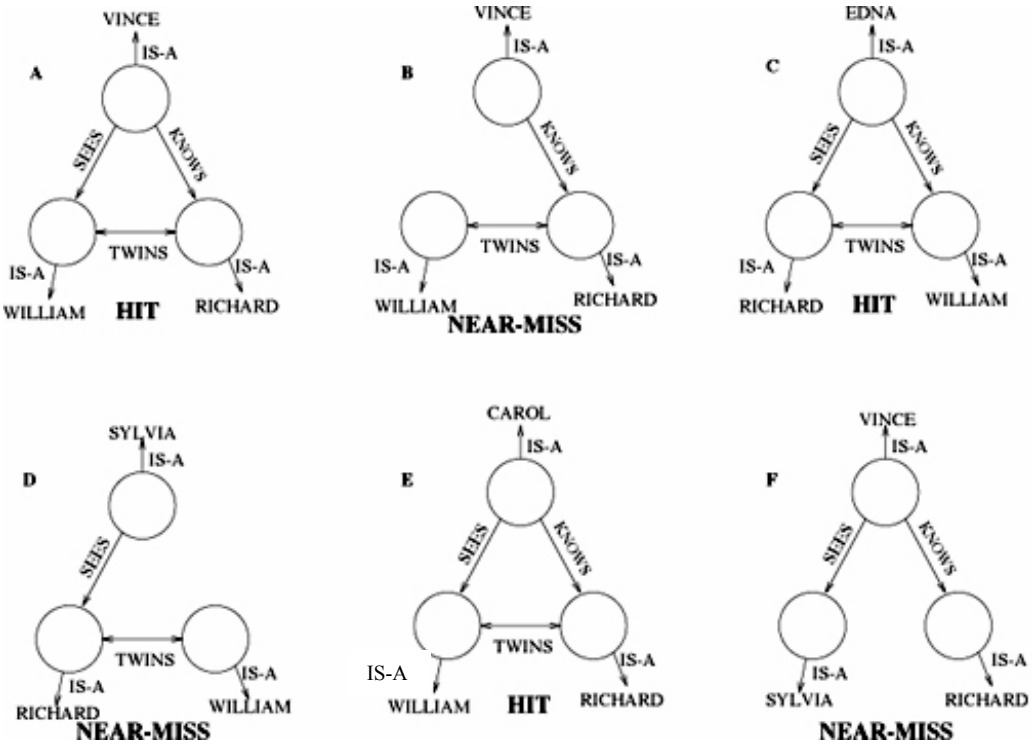
Part A: Mistaken Identity (9 Points)

Ben Bitdiddle, enterprising AI engineer, decides that the next hot product will be a system that summarizes soap operas, so that people don't have to spend time watching them to know what happens. As part of the system, he finds that he needs to recognize concepts based on relationships between people, and decides that the right way to do this is with a near-miss learning system.

Ben decides to learn about the sort of comedy that ensues when somebody mistakes one identical twin for another. He starts by studying the relationships between people in soap operas and determines that people tend to fall into a few predictable categories. He builds the following hierarchy to describe people and their occupations:

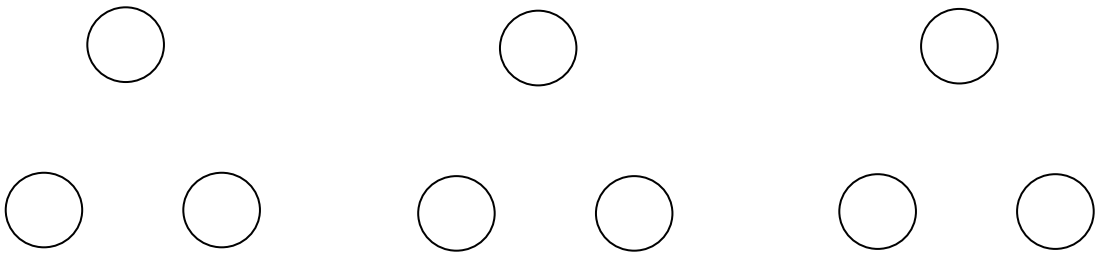


Ben then feeds the near-miss learning system six mistaken identity samples—which we have already translated for you from English into networks—in alphabetical order. These samples are shown on the next page.

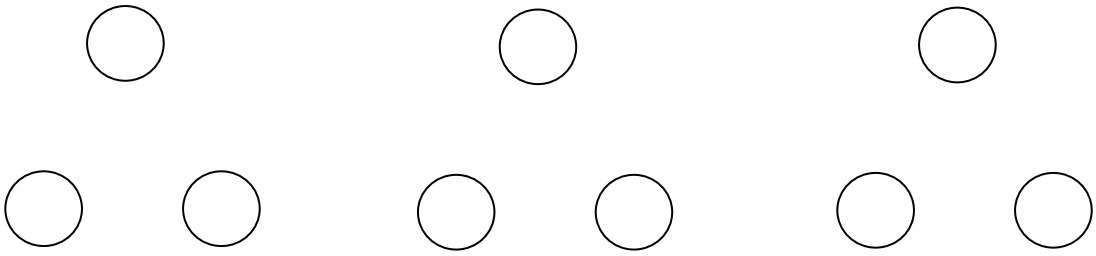


Part A1 (6 Points)

For each of the six samples above, draw the model Ben's system constructs from the data given thus far.



After A After B After C



After D After E After F

Part A2 (3 Points)

Now indicate for each of the six samples, whether the system is specializing (S), generalizing (G), both (B), or neither (N).

A

B.

C.

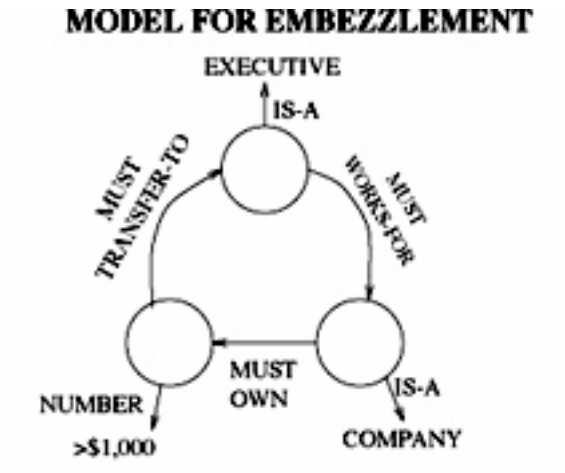
D.

E.

F.

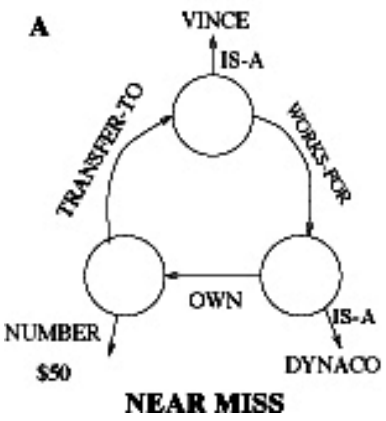
Part B: Embezzlement (4 Points)

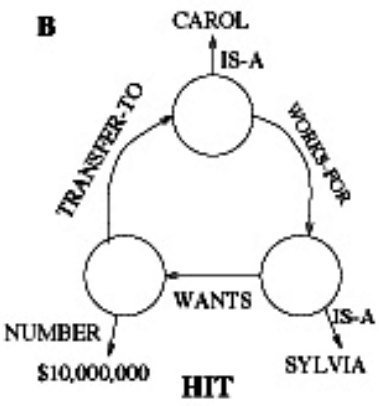
Pleased with his success, Ben decides to train the system for another model. Now he wants the system to recognize when embezzlement occurs. After a while, his model looks like this:

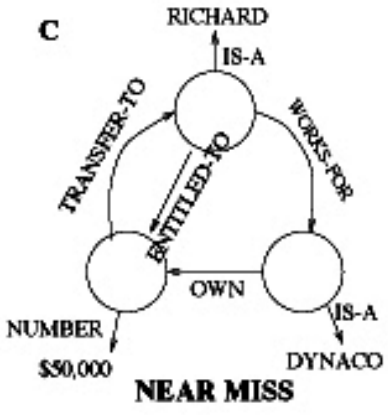


Part B1 (2 Points)

Noting that small amounts of money are often transferred to employees in repayment of entertainment expenses, which of the following three models should Ben choose for the next sample?

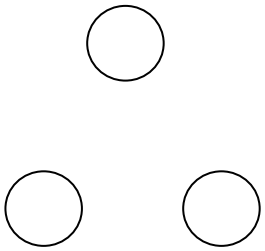






Part B2 (2 Points)

Draw the new model resulting from your choice.



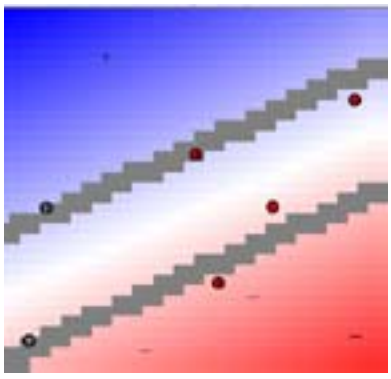
Problem 6: Support Vector Machines (14 Points)

Part A: (2 Points)

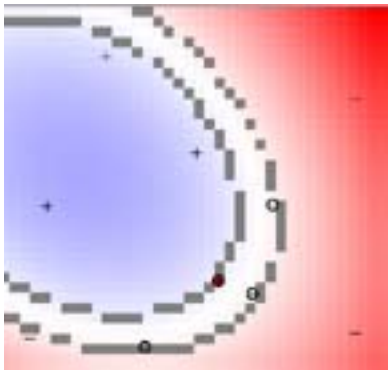
The following diagrams represent graphs of support vector machines trained to separate pluses (+) from minuses (-) for the same data set. The origin is at the lower left corner in all diagrams. Which represents the best classifier for the training data? ***See the separate color sheet for a clearer view of these diagrams.***

Indicate your choice here: ☐

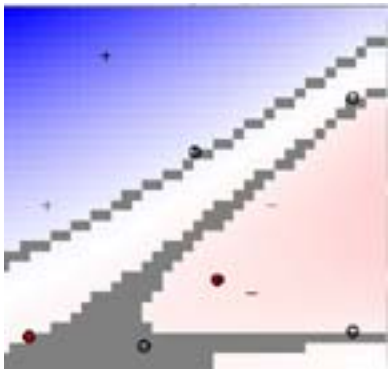
A.



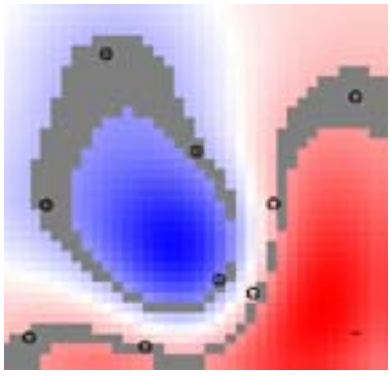
B.



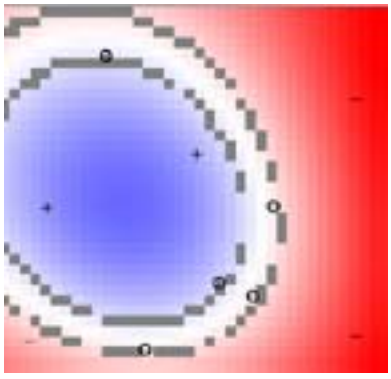
C.



D.



E.



Part B: (5 Points)

Match the diagrams in Part 1 with the following kernels:

Radial basis function, sigma .08 ☐

Radial basis function, sigma .5 ☐

Radial basis function, sigma 2.0 ☐

Linear ☐

Second order polynomial ☐

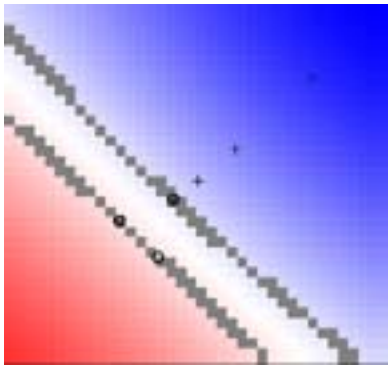
Part C: (3 Points)

Order the following diagrams from ***smallest*** support vector weights to ***largest*** support vector weights, assuming all diagrams are produced by the same mechanism using a linear kernel (that is, there is no transformation from the dot-product space).

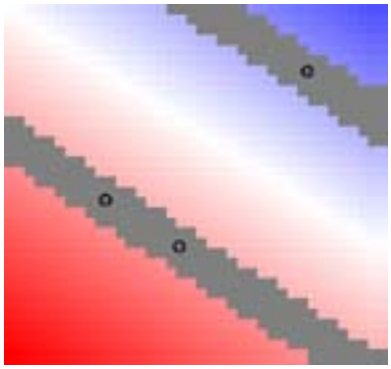
The origin is at the lower left corner in all diagrams. Support vector weights are also referred to as α_i values or LaGrangian multipliers. ***See the separate color sheet for a clearer view of these diagrams.***

Smallest ***Medium*** ***Largest***

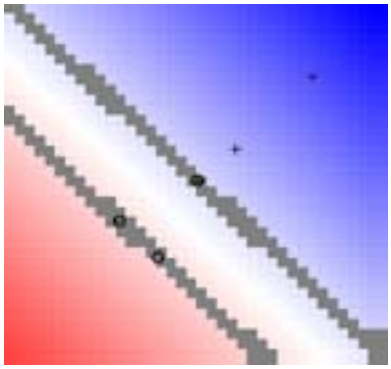
A.



B.



C.



Part D (4 Points)

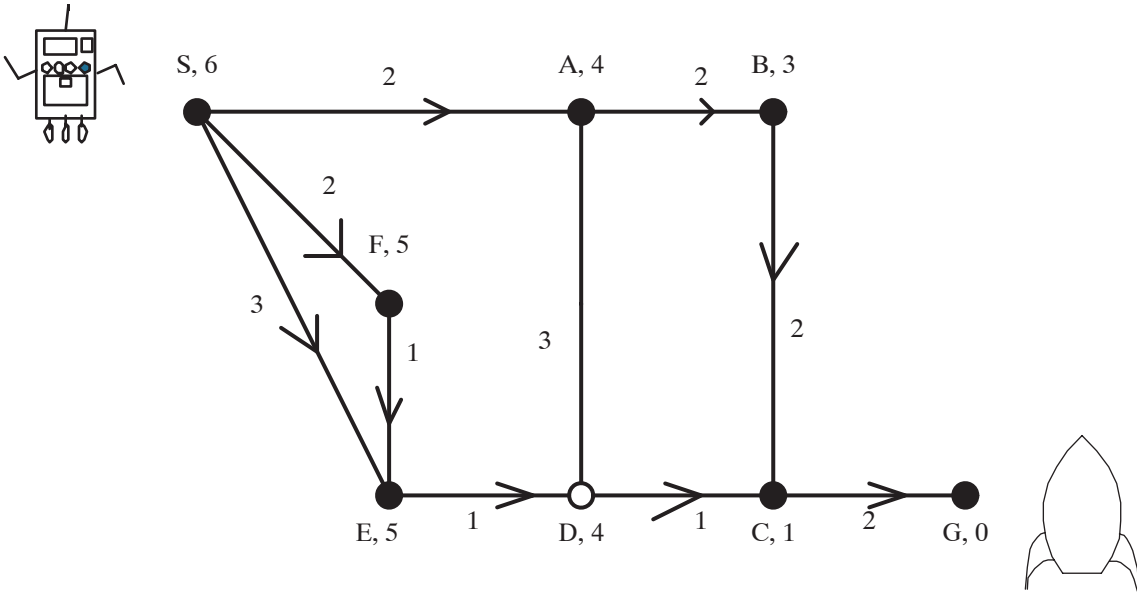
Suppose a support vector machine for separating pluses from minuses finds a plus support vector at the point $\mathbf{x_1}$= (1, 0), a minus support vector at $\mathbf{x_2}$= (0, 1).

You are to determine values for the classification vector ***w*** and the threshold value *b*. Your expression for ***w*** may contain $\mathbf{x_1}$ and $\mathbf{x_2}$ because those are vectors with known components, but you are not to include any α_i or y_i. Hint: think about the values produced by the decision rule for the support vectors, $\mathbf{x_1}$ and $\mathbf{x_2}$.

w

b

TEAR OFF PAGE FOR SEARCH



TEAR OFF PAGE FOR CONSTRAINT PROPAGATION

Constraints:

First constraint:	No dance can be in more than one slot
Second constraint:	Dances in slots 1 and 6 must be showstoppers
Third constraint:	Consecutive dances must not share dancers

Data figures:

Dances that share dancers:

	A	B	C	D	E	F
A			Shares			
B			Shares		Shares	
C	Shares	Shares		Shares		
D			Shares			Shares
E		Shares				Shares
F				Shares	Shares	

Genre to genre transition penalies:

Heuristics	**Ballet**	**Hip Hop**	**Jazz**	**Swing**
Ballet	10	5	1	2
Hip Hop	5	10	1	2
Jazz	1	1	10	3
Swing	2	2	3	10

Genres of the dances:

Dance	**A**	**B**	**C**	**D**	**E**	**F**
Genre	Hip Hop	Hip Hop	Jazz	Swing	Ballet	Jazz

Showstopper data:

Dance	**Showstopper?**
A	Yes
B	
C	Yes
D	Yes
E	
F	

TEAR OFF PAGE FOR BACKPROPAGATION

Negative Sigmoid Values

z	n(z)		z	n(z)
-3.00	-0.05		0.25	-0.56
-2.75	-0.06		0.50	-0.62
-2.50	-0.08		0.75	-0.68
-2.25	-0.10		1.00	-0.73
-2.00	-0.12		1.25	-0.78
-1.75	-0.15		1.50	-0.82
-1.50	-0.18		1.75	-0.85
-1.25	-0.22		2.00	-0.88
-1.00	-0.27		2.25	-0.90
-0.75	-0.32		2.50	-0.92
-0.50	-0.38		2.75	-0.94
-0.25	-0.44		3.00	-0.95
0.00	-0.50		3.25	-0.96

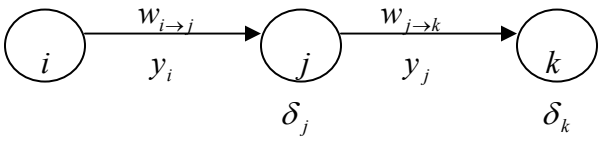
An efficient method of implementing gradient descent for neural networks.

The formulas below assume a regular sigmoid unit, $s(z) = \frac{1}{1+e^{-z}}$, and an error function of $E = \frac{1}{2}\sum(y^* - y)^2$.

Descent Rule $w_{i\to j} = w_{i\to j} - r\frac{dE}{dw_{i\to j}} = w_{i\to j} - r\delta_j y_i$

Backprop rule $\delta_j = \frac{ds(z_j)}{dz_j}\sum_k \delta_k w_{j\to k}$

1. Initialize weights to small random values
2. Choose a random sample input feature vector
3. Compute total input (z_j) and output (y_j) for each unit (forward prop)
4. Compute δ_n for output layer $\delta_n = \frac{ds(z_n)}{dz_n}(y_n - y_n^*) = y_n(1-y_n)(y_n - y_n^*)$
5. Compute δ_j for preceding layer by backprop rule (repeat for all layers)
6. Compute weight change by descent rule (repeat for all weights)



y_i is x_i for input layer