6.034 Final Examination
Tuesday, December 20, 2005

Name	MARVIN MINSKY
Email	

Note that problems are given roughly in the order the material was covered, not in the order of difficulty or expected time to complete. The total number of points is 300, 100 for each of the three sections, each of which corresponds to approximately 1/3 of the subject.

Problem number	Maximum	Score	Grader
I-1	50	50	PHW
I-2	50	50	PHW
II-1	50	50	PHW
II-2	50	50	PHW
III-1	10	10	PHW
III-2	34	34	PHW
III-3	20	20	PHW
III-4	36	36	PHW
Total	300	300	

Section I, Problem 1, Rules (50 points)

Important note: This problem resembles a problem on the first quiz, but is sufficiently different that you must not think to solve it by copy and pasting the solution to that corresponding problem.

Sam Urai decides to use his 6.034 knowledge to defend MIT. To this end, he wants to identify ninjas. He would like to identify less powerful ninja first, to allow him to level up before the boss battle.

He produces the following rule set: **(Reproduced in supplement)**

```
(P0
(IF
THEN (evaluate-gradstudents)))

(P1
(IF (evaluate-gradstudents)
(?person is gradstudent)
(?person researches ?topic)
(?topic national-secret)
THEN (?person ninja)))

(P2
(IF (evaluate-gradstudents)
DELETE (evaluate-gradstudents)
ADD (evaluate-professors)))

(P3
(IF (?x is professor)
(?y is student of ?x)
(?y ninja)
THEN (?x ninja)))
```

He then debugs it on the following fabricated data set:

```
A1: (Cat researches ethics)
A2: (Tanis researches finance)
A3: (Jennifer researches ColdFusion)

A4: (ethics national-secret)
A5: (ColdFusion national-secret)

A6: (Cat is gradstudent)
A7: (Tanis is gradstudent)
A8: (Jennifer is gradstudent)

A9: (Winston is professor)

A10: (Cat is student of Winston)
A11: (Tanis is student of Winston)
```

Part A, Forward Chaining (25 points)

Part A1 (15 points)

Sam calls (chain 6). Fill in the table with what the computer does, assuming:

1 Ties are broken by rule order

2 Ties between rules are broken by assertion order.

Note that only the far-right column will be graded; the other columns are for whatever scratch work you choose to do.

Matched	Triggered	Fired	Changed	
			(EVALUATE-GRAD STUDENTS)	+
			(CAT NINJA)	+
			(JENNIFER NINJA)	+
			(EVALUATE-PROFESSORS) (EVALUATE-GRADSTUDENTS)	+ –
			(EVALUAT-GRADSTUDENTS	+
			(EVALUATE-GRAD STUDENTS)	–

Part A2 (5 points)

Who is a identified as a ninja after (chain 6)?

CAT, JENNIFER

Part A3 (5 points)

Who is identified as a ninja after (chain 100)?

CAT, JENNIFER

Part B, Backward Chaining (25 points)

Now, Sam calls a backward chainer on the assertion (Winston is Ninja)

Part B1 (20 points)

Fill in the table with the assertion that the backward chainer is trying to match, in the order that the backward chainer tries to match them. Assume, as in quiz 1:

1 If the system asks the user if an assertion is true, the answer is no.

2 The backward chainer does not modify the assertion base, so it can derive the same assertion multiple times.

3 Conflict Resolution is by rule order

4 If a rule can have multiple bindings, then conflict resolution is by assertion order.

(Winston ninja)
(EVALUATE-GRADSTUDENTS)
(WINSTON IS GRADSTUDENT)
(WINSTON IS PROFESSOR)
(?V IS STUDENT OF WINSTON)
(CAT NINJA)
(EVALUATE - GRADSTUDENTS)
(CAT IS GRADSTUDENT)
(CAT RESEARCHES ETHICS)
(ETHICS NATIONAL-SECRET)

Part B2 (5 points)

Is Winston identified as a Ninja?

YES

Section I, Problem 2, Search (50 points)

Part A (20 points)

The following are some multiple choice questions about the searches we have learned in class. Answer them, considering all of the search functions we covered in class. Use the convention that S is the start node, and G is the goal. Circle only 1 answer.

If you are more comfortable with the tree method, recall that the paths in the queue are the same as the paths in the tree that can be extended.

Part A1 (5 points)

Could the path (S A B C A) appear on the search queue?

A Yes, it is possible for it to appear on the search queue of any of the search functions we learned about in class.

B Yes, it can appear on the search queue, but only if you do not use an extended list.

[✓ C] No, it cannot appear on the search queue, because the path contains a loop.

D No, it cannot appear in the search queue because it is not a valid path.

Part A2 (5 points)

Could the paths (S A C D B) and (S A B D C) both appear, *at some point in the search*, on the same search queue?

A Yes, it is possible for them to appear on the search queue of any of the search functions we learned about in class.

[✓ B] Yes, they can appear on the search queue, but only if you do not use an extended list.

C No, they cannot both appear on the search queue, because the two paths contain one node multiple times.

D No, they cannot appear on the search queue, because one/both of them are not valid paths.

Part A3 (5 points)

Could the paths (S A C D) and (S A) both appear, *simultaneously* on the same search queue?

A Yes, it is possible for them to appear on the search queue of any of the search functions we learned about in class.

B Yes, they can appear on the search queue, but only if you do not use an extended list.

C No, they cannot be on the search queue at the same time, because (S A) is shorter than (S A C D)

[✓ D] No, they cannot appear on the search queue, because (S A) must be extended before (S A C D) can be put on the search queue.

Part A4 (5 points)

What is an extended list?

[✓ A] A list of nodes, which have been at the end of a path which was **removed** from the search queue.

B A list of nodes, which have been at the end of a path which was **put on** the search queue.

Part B (10 points)

Search queues are characterized by the operations that place new paths on the queue and that provide a path when asked for one. Here are the options available to you:

Queue A:
1. Puts new paths on the **back** of the Queue, in lexical order.
2. Takes a path off the front of the Queue.

Queue B:
1. Puts new paths on the **front** of the Queue, in lexical order.
2. Takes a path off the front of the Queue.

Queue C:
1. Puts new paths on the **back** of the Queue, ordered by their **heuristic values**.
2. Takes a path off the front of the Queue.

Queue D:
1. Puts new elements on the **front** of the Queue, ordered by their **heuristic values**.
2. Take a path off the front of the Queue.

Queue E:
1. Puts new elements into the Queue.
2. Take the path in the entire Queue with the best **heuristic value**.

Queue F:
1. Puts new elements into the Queue.
2. Take the path in the entire Queue with the **least path length**.

Queue G:
1. Puts new elements into the Queue.
2. Take the path in the entire Queue with the **least (consistent estimate of distance remaining + path length)**.

For each search below, circle the letter that identifies the queue type, from the list above, you would use to implement the search. If you are more comfortable with the tree method, recall that the queue determines the order that paths are selected to be extended.

Depth	A [✓ B] C D E F G
Breadth	[✓ A] B C D E F G
Hillclimbing	A B C [✓ D] E F G
Best First	A B C D [✓ E] F G
Branch and Bound	A B C D E [✓ F] G
A*	A B C D E F [✓ G]

Part C (10 points)

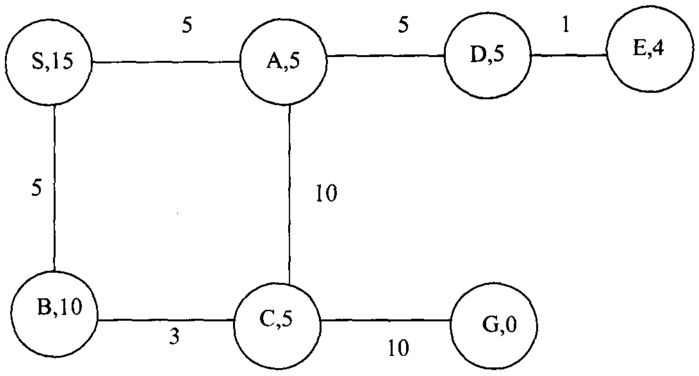
In the final section of this problem, we ask you to create the search tree for Mystery Search, which is not the same as any search you have studied so far. Mystery Search has the following characteristics:

1 When you finish extending a path, put the new paths on to the *front* of the search queue, in (heuristic + path) order, such that the new path with the least (heuristic + path) should end up at the *front* of the queue.

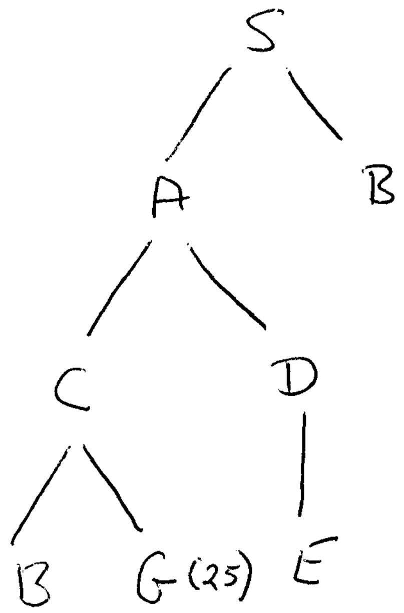
2 When you want to extend another path, take a path off the *front* of the queue.

3 Stop when a path with the goal comes off the queue.

4 Mystery search **uses backtracking and an extended list**. S is the start and G is the goal.



Part C1: Draw the tree produced by Mystery search.



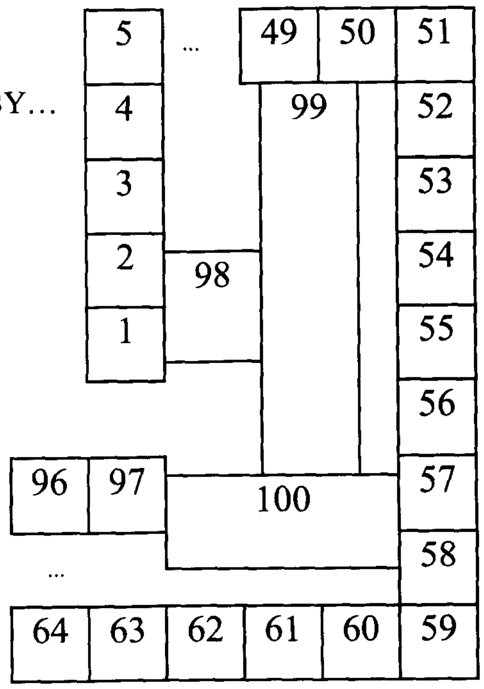
Part C2: Does Mystery search find the shortest path in the example? Yes [✓ No]
Part C3: Is the heuristic admissible? [✓ Yes] No

Section II, Question 1: Constraint Propagation (50 points)

Part A: Propagating through a network (10 points)

How long will it take to color this map of 100 blocks with <u>4</u> colors using **backtracking with forward checking <u>AND propagating through singleton domains</u>**, assuming:

(1) we start coloring from node 1 and go in sequence
(2) each coloring operation will take 1 second
(3) variables are assigned, whenever possible, RGBYRGBY…
(4) No two touching squares should have the same color.



Note that there are 43 blocks between 5 and 49, and 31 blocks between 96 and 64; Each of these blocks have exactly two blocks touching them.

Fill in the blank below:

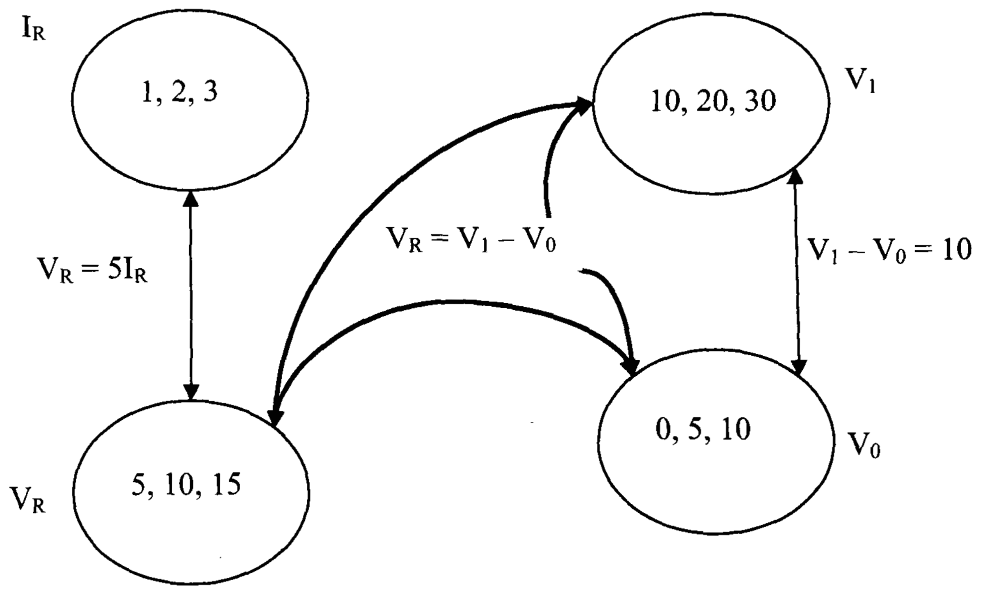
It will take approximately 100 **seconds.**

Only *approximate* answers are required.

Explain your answer in the space below:

Part B: Propagating through a network (40 points)

After a bit of tinkering with an electrical circuit containing a voltage supply and a resistor, you develop the following constraint network with the integer domains shown: **(Reproduced in supplement)**



Notice the three curvy bold lines represent a single constraint between three variables.

When you are propagating a constraint, you have to try every possible combination of values and eliminate those values that were not part of any feasible combination.

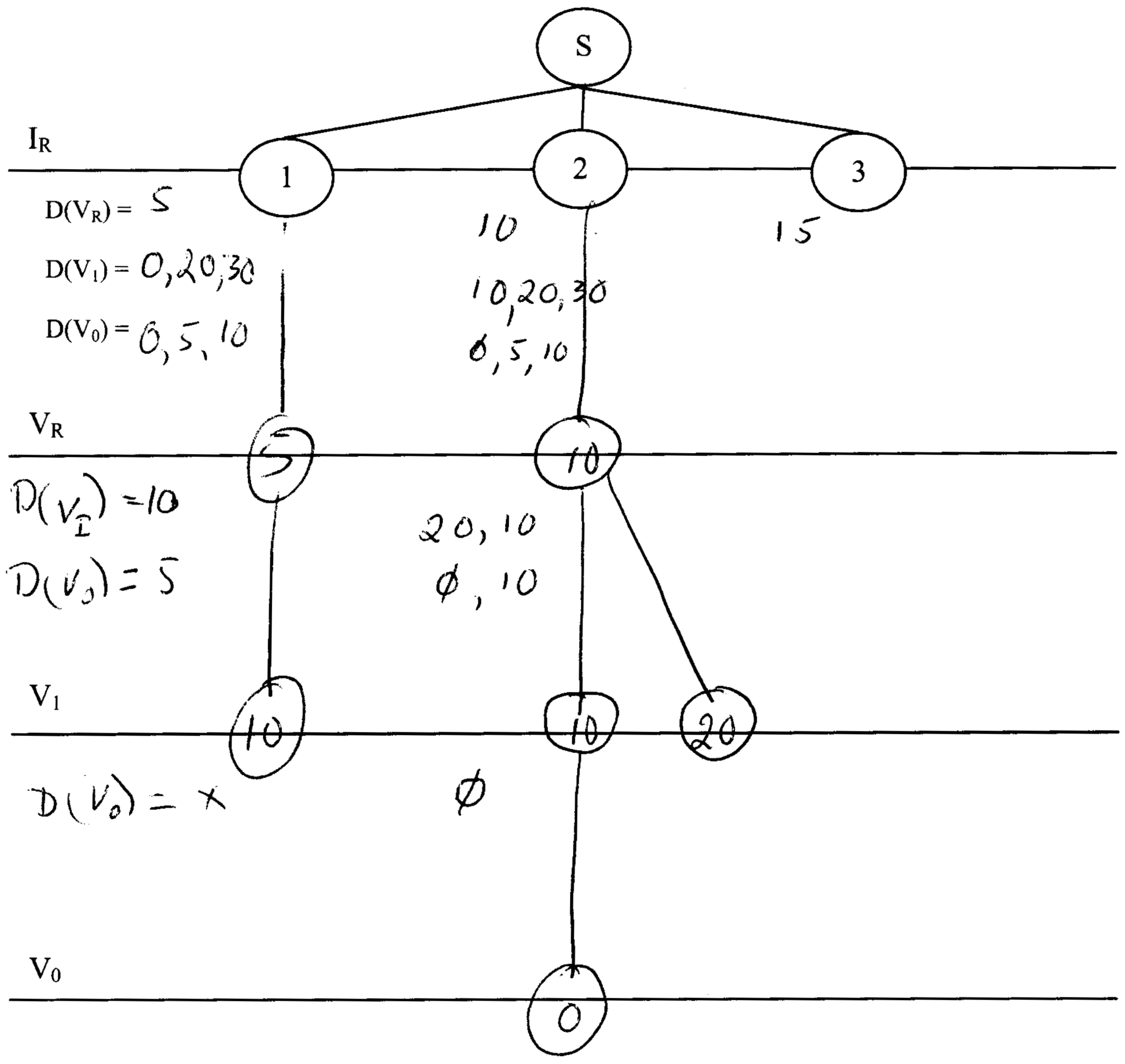
PART B1: (20 pts)

Draw your search tree for using **backtracking with forward checking** (no propagation beyond neighbors of just-assigned variable)

In addition:

1. for every node in the tree draw *only* the valid descendants at that point *and*
2. for every node in the tree draw the domains at that point.

A portion of the tree is drawn for you.



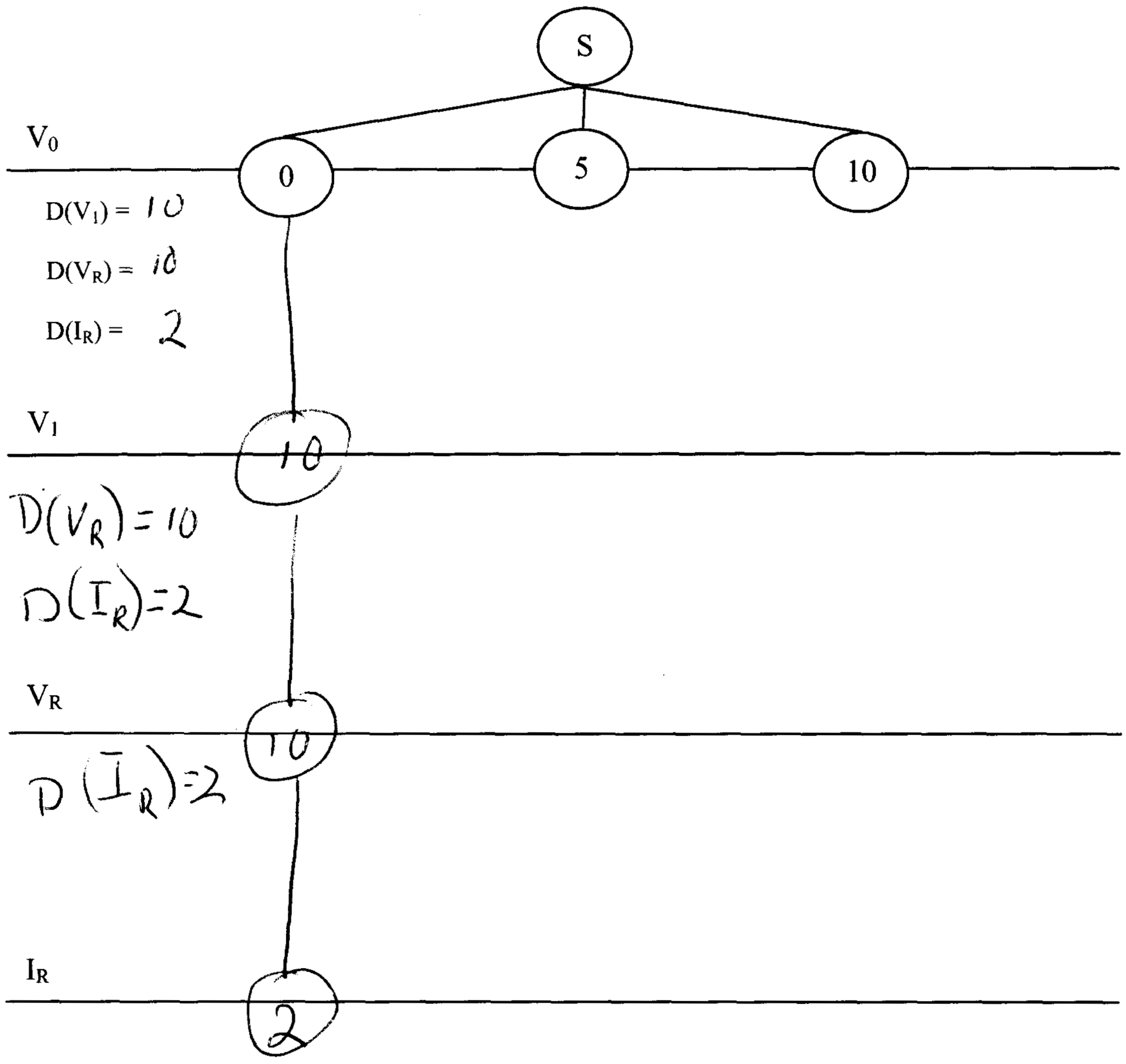
PART B2: (20 pts)

Draw your search tree for using **backtracking with forward checking AND propagating through domains that are reduced to size 1 [singleton domains]**.

In addition:

1. for every node in the tree draw *only* the valid descendants at that point *and*
2. for every node in the tree draw the domains at that point.

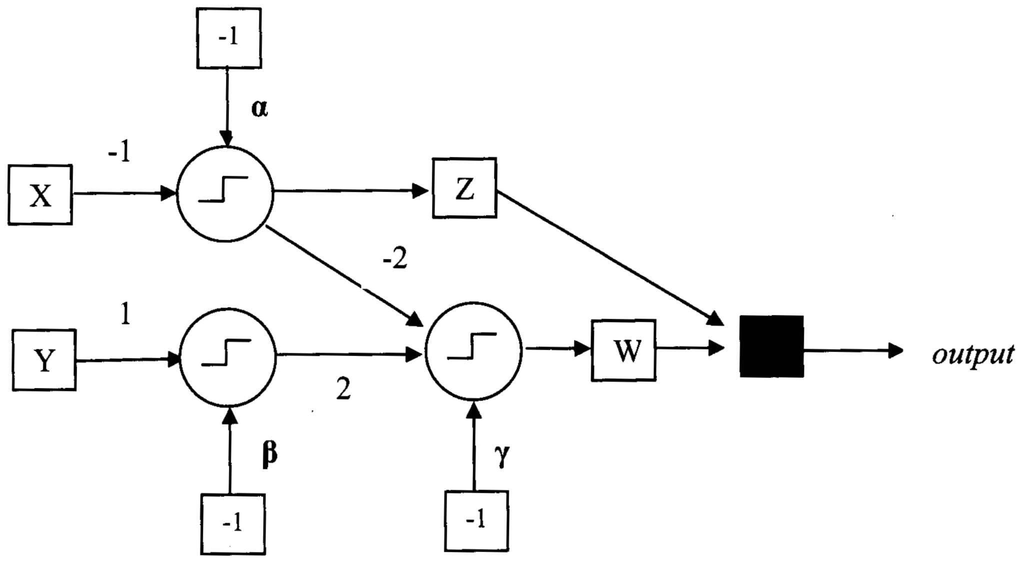
A portion of the tree is drawn for you. **Note that the order of variable assignment is not the same as the order specified in Part B1.**



Section II, Question 2: Neural Networks (50 points)

Part A: Calculation of weights (30 points)

For this part, consider the following network:

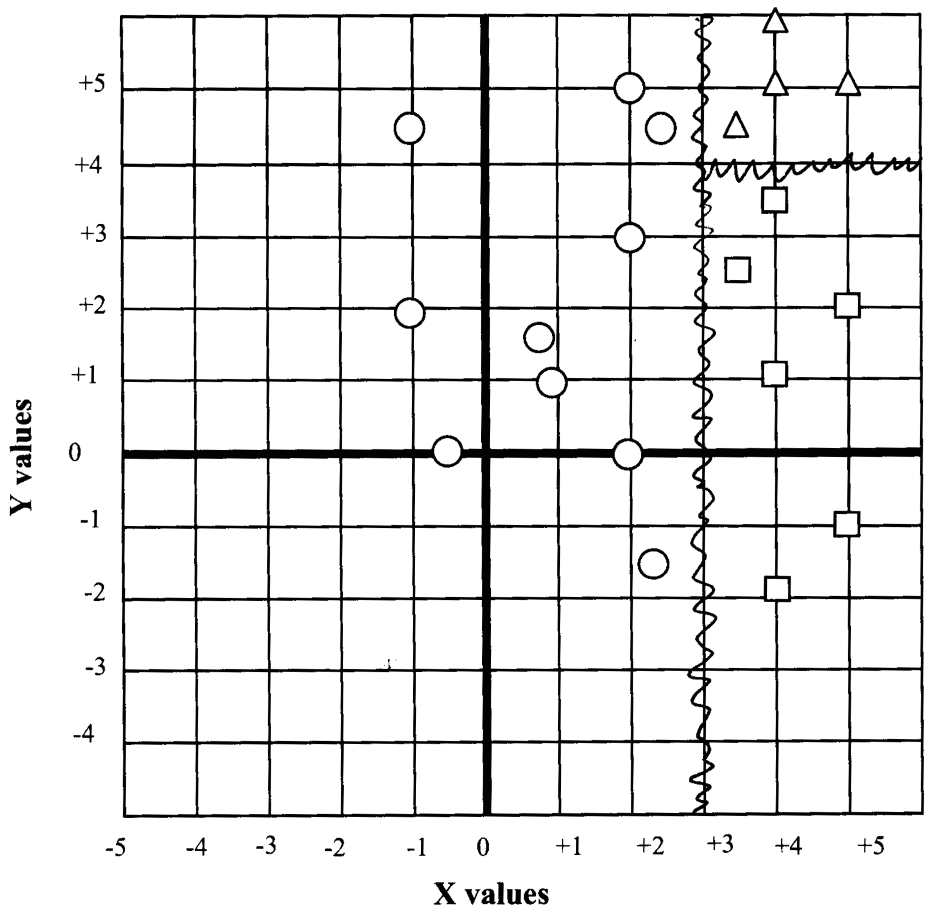


All units have step function thresholds, not sigmoids!!
That is, output = 1 for input greater than 0; output = 0 for input equal to or less than 0.

The black box in the diagram is a logic box that converts its inputs into outputs according to the following table:

Z	W	*Output*
1	1	○
1	0	○
0	1	△
0	0	□

You want to use the network on the previous page to recognize this data:



Find the appropriate weights for α, β, and γ so that this network correctly identifies all the points on the data set. You may only use integral numbers as your answer.

$\alpha =$ $\boxed{-3}$

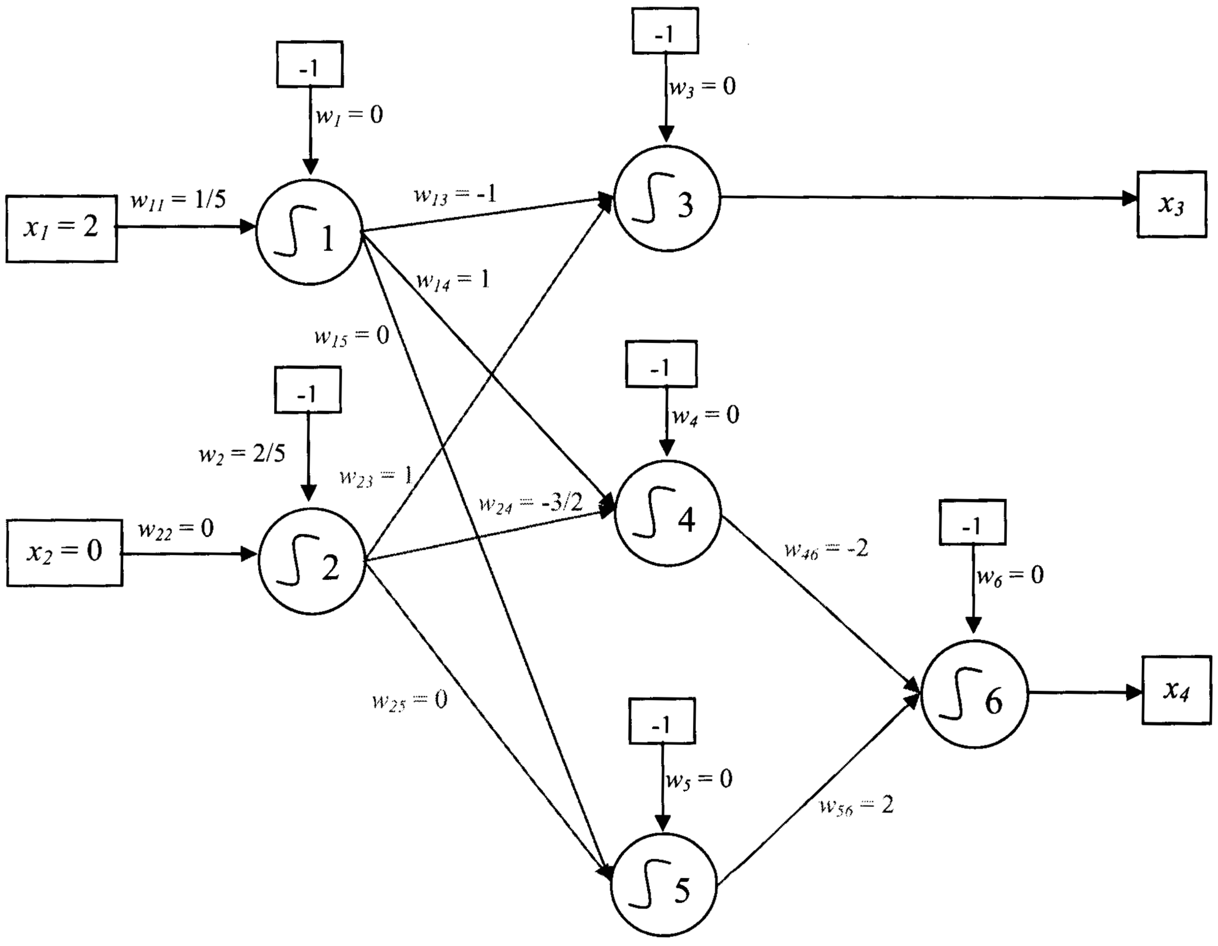
$\beta =$ $\boxed{4}$

$\gamma =$ $\boxed{1 \text{ or } \phi}$

Part B: Forward Propagation (20 points)

The following network has 6 units labeled 1, 2, 3, 4, 5, 6. All units are **sigmoid** units, meaning that their output s(z) is computed using the sum of their weighted inputs (z):

$$s(z) = \frac{1}{1+e^{-z}}$$



Below are the initial weights, repeated in this table for your convenience. Note that w_{11} is the multiplier of the value x_1, not the value from sigmoid unit 1.

w_1	w_{11}	w_{13}	w_{14}	w_{15}	w_2	w_{22}	w_{23}	w_{24}	w_{25}	w_3	w_4	w_{46}	w_5	w_{56}	w_6
0	$1/5$	-1	1	0	$2/5$	1	1	$-3/2$	0	0	0	-2	0	2	0

Using these weights, and the input vector $(x_1,x_2) = [2, 0]$, compute the output of each unit after forward propagation.

A table of sigmoid values is included for you at the bottom of this page. Where appropriate, use the values on the table to estimate the s(z) values.

y_1 (output of unit 1) = 0.6

y_2 (output of unit 2) = 0.4

y_4 (output of unit 4) = 0.5

y_5 (output of unit 5) = 0.5

Sigmoid Function S(z)

z	-0.8	-0.6	-0.4	-0.2	0	0.2	0.4	0.6	0.8
s(z)	0.31	0.35	0.40	0.45	0.5	0.54	0.60	0.64	0.69

Section III, Problem 1, Miscellaneous (10 pointss)

In all cases, circle the correct answer

Part A

Learning with the arch-learning heuristics always leads to:

A. Generalizations of the first example, never specializations.
B. Specializations of the first example, never generaizations.
[✓ C.] Either generalizations or specializations or both.

Part B

An fMRI machine:

A. Is used to clean test tubes
[✓ B.] Detects small changes in blood flow
C. Measures gamma-ray emmissions consequent to neutron bombardment

Part C

Brain areas have been found that are especially sensitive to pictures of:

[✓ A.] Body parts
B. Snakes and insects
C. Cups and bowls

Part D

Noam Chomsky has indicated that he believes human intelligence was enabled 50,000 years ago by the emergence of :

A. The opposable thumb
B. Movement to carnivorous diet caused by climate change
[✓ C.] The ability to join concepts together without limit

Part E

The principle virtue of high-dimensional spaces is that:

A. It simplifies optimization by making it possible to introduce LaGrange multipliers
[✓ B.] It is easier to find a decision boundary that appropriately divides a sample set
C. Overfitting is no longer possible

Section III, Problem 2, Support Vector Machines (34 points)

Part A (10 points)

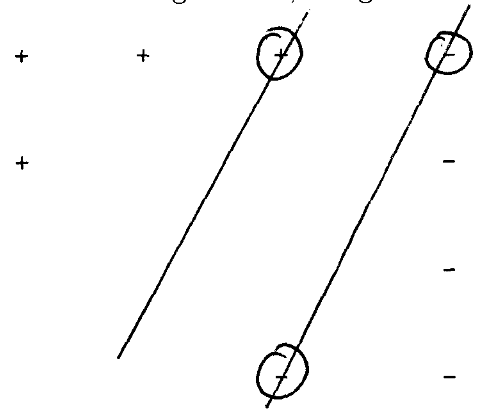
Suppose you train a Support Vector Machine of the form

$$h(\bar{x}) = \text{sign}\left(\sum_i \alpha_i y_i K(\bar{x}_i, \bar{x}) + b\right)$$

where

$$\text{sign}(z) = \begin{cases} +1 & \text{if } z \geq 0 \\ -1 & \text{if } z < 0 \end{cases}$$

on the following dataset, using a **linear** kernel ($K(x_i, x_j) = \bar{x}_i \cdot \bar{x}_j$).



1. draw (directly on the diagram above) the *street (not the decision boundary)* that the SVM classifier identifies
2. *circle* which points will have *non-zero* support vector weights

How many support vectors did you circle? (Circle your answer below.)

1 2 [✓ 3] 4 5 6 7 8

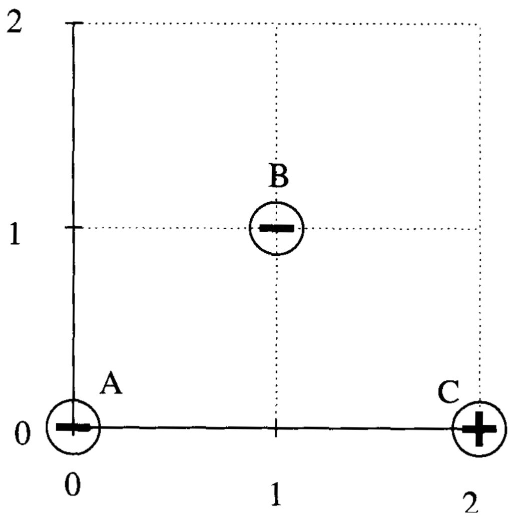
Note: Do this by visual inspection only!

Part B (24 points)

In this problem, we explore how an SVM classifier

$$h(\bar{x}) = \text{sign}\left(\sum_i \alpha_i y_i K(\bar{x}_i, \bar{x}) + b\right)$$

with a **linear** kernel ($K(\bar{x}_i, \bar{x}_j) = \bar{x}_i \cdot \bar{x}_j$) treats the following data set. There are two negative points (-), at (0, 0) and (1, 1), labeled A and B, and one postive (+) point, at (2, 0), labeled C.



Part B1 (5 points)

Compute the 9 kernel values:

$K(\bar{x}_A, \bar{x}_A) = 0$	$K(\bar{x}_A, \bar{x}_B) = 0$	$K(\bar{x}_A, \bar{x}_C) = 0$
$K(\bar{x}_B, \bar{x}_A) = 0$	$K(\bar{x}_B, \bar{x}_B) = 2$	$K(\bar{x}_B, \bar{x}_C) = 2$
$K(\bar{x}_C, \bar{x}_A) = 0$	$K(\bar{x}_C, \bar{x}_B) = 2$	$K(\bar{x}_C, \bar{x}_C) = 4$

Part B2 (5 points)

Assume that A, B, and C are all support vectors with non-zero weight obeying the following 2 constraints:

Constraint 1: $\sum \alpha_i y_i = 0$
Constraint 2:

- $h(\bar{x}) = +1$ for positive support vectors
- $h(\bar{x}) = -1$ for negative support vectors

Write down the coefficients in the following equations such that the equations, when solved, yield values for $\alpha_A, \alpha_B, \alpha_C$, and b, given the above constraints.

Eq. 1: $\underline{-1}\,\alpha_A + \underline{-1}\,\alpha_B + \underline{1}\,\alpha_C + \underline{0}\,b = \underline{0}$
Eq. 2: $\underline{0}\,\alpha_A + \underline{0}\,\alpha_B + \underline{0}\,\alpha_C + \underline{1}\,b = \underline{-1}$
Eq. 3: $\underline{0}\,\alpha_A + \underline{-2}\,\alpha_B + \underline{2}\,\alpha_C + \underline{1}\,b = \underline{-1}$
Eq. 4: $\underline{0}\,\alpha_A + \underline{-2}\,\alpha_B + \underline{4}\,\alpha_C + \underline{1}\,b = \underline{1}$

Part B3 (5 points)

Solve for $\alpha_A, \alpha_B, \alpha_C$ and b, and enter your values below:

$\alpha_A = 0 \quad \alpha_B = 1 \quad \alpha_C = 1 \quad b = -1$

Part B4 (5 points)

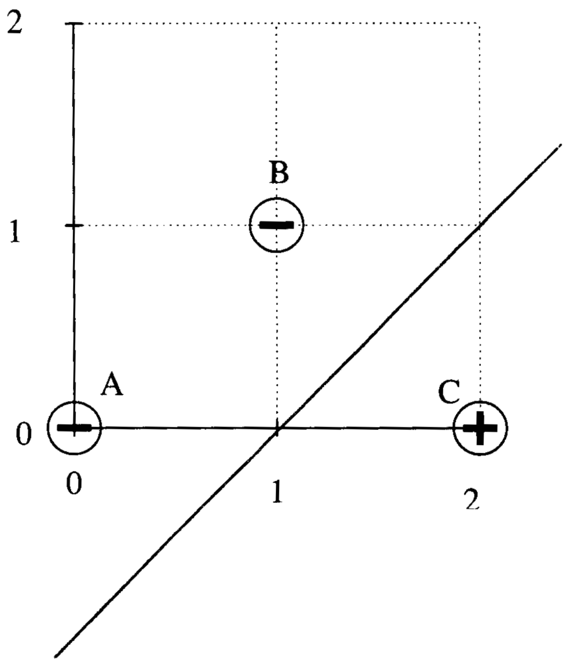
Write an expression for the classifier $h(x)$ by completing the if expression:

$$h(x) = \begin{cases} +1 & \text{if } \boxed{X_1 - X_2 > 1} \\ -1 & \text{otherwise} \end{cases}$$

Show your work here:

Part B5 (4 points)

Draw the *decision boundary* for the SVM classifier for the dataset:



Section III, Problem 3, Self Organizing Maps (20 points)

You are making a self-organizing map with nine nodes, which you initialize to:

[✓ 4]	9	42
77	2	12
8	5	17

You decide to randomly choose training points from the set of integers between 0 and 20.

The distance function you intend to use is as follows, where x is a **point from the map**, and y is a **training point**:

$$\text{distance}(x, y) = |x^2 - y|$$

That is not a typo: x is squared, y is not. The function you intend to use to modify a point is:

$$\text{modify}(x, y, \alpha) = |y - x| \times \alpha + x$$

For this problem, assume that α will always be 1/2.

Part A (5 points)

The first training sample is 18. Which point in the map do you choose? Circle the best answer on the map above.

Part B (5 points)

What is the new value of this point? 11

Part C (10 points)

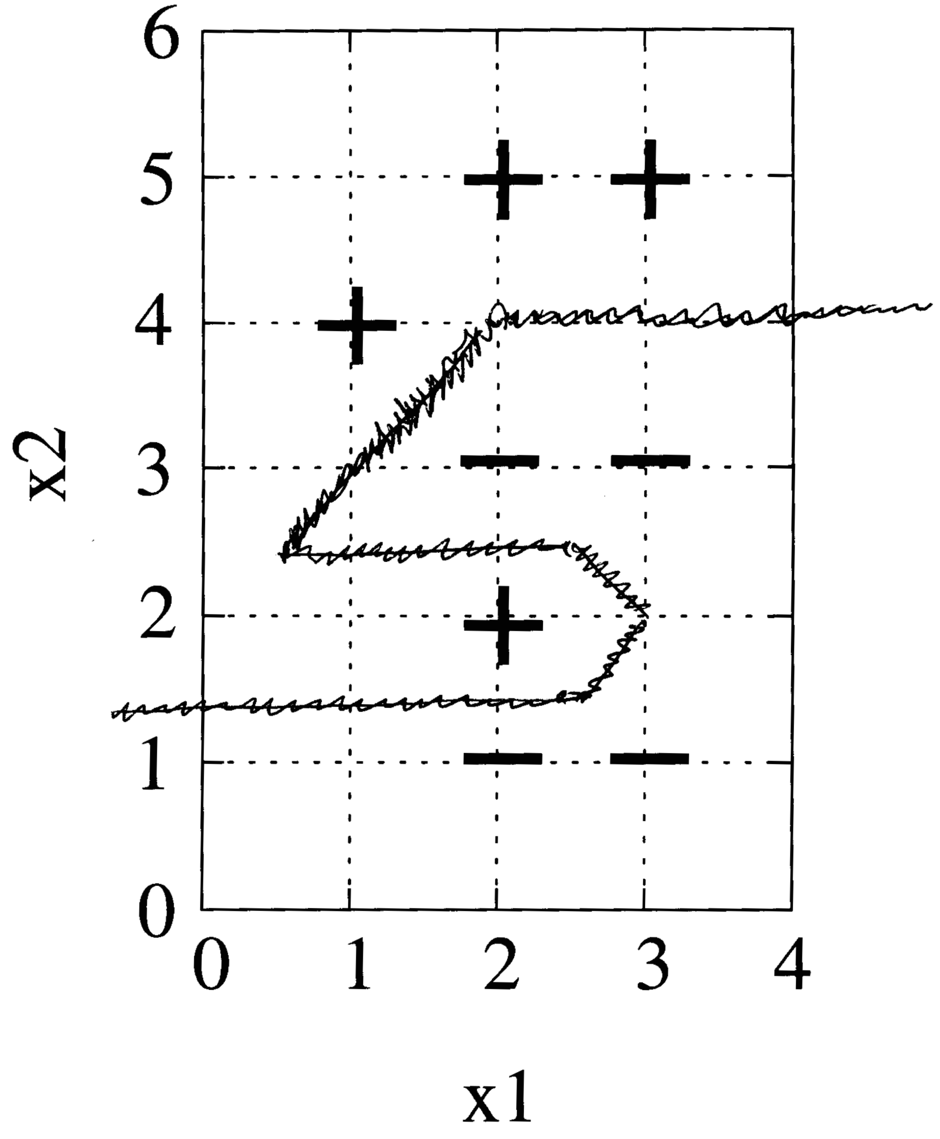
Suppose continue to train the map using random values of y in the range from 0 to 20. Further suppose no modification is made to neighbors of the winning point in the map. What will happen? Circle the best answer.

A. The map will converge toward predictable map values.
[✓ B.] All the map values will continue to get bigger
C. The map values will exhibit logarithmic relationship.
D. The map values will exhibit linear relationship.

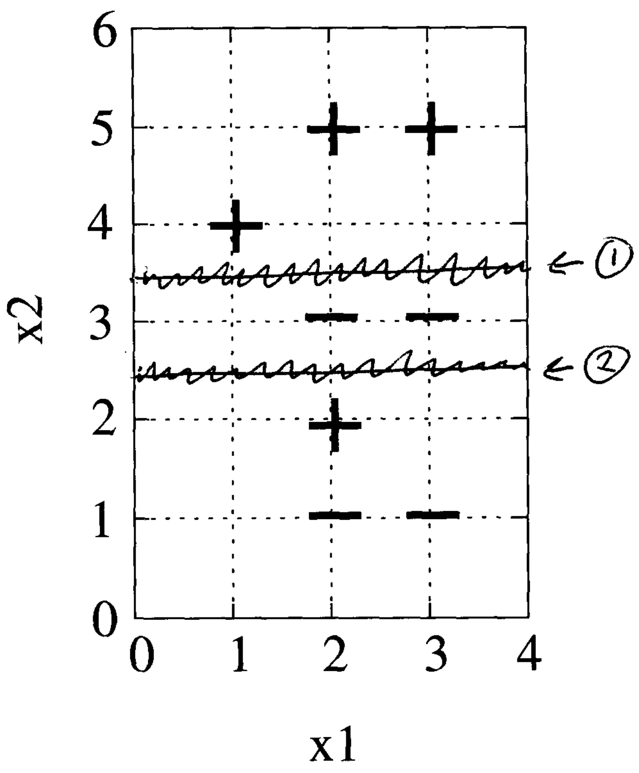
Section III, Problem 4, Nearest Neighbors, ID Trees and Boosting (36 points)

Part A, Nearest Neighbors (3 points)

Draw the decision boundary for a 1-nearest neighbor classifier in the following data set:



Part B, ID Trees (3 points)



In the plot above, draw the decision boundary for an identification tree classifier built using the average-disorder criterion and only using the following decision thresholds.

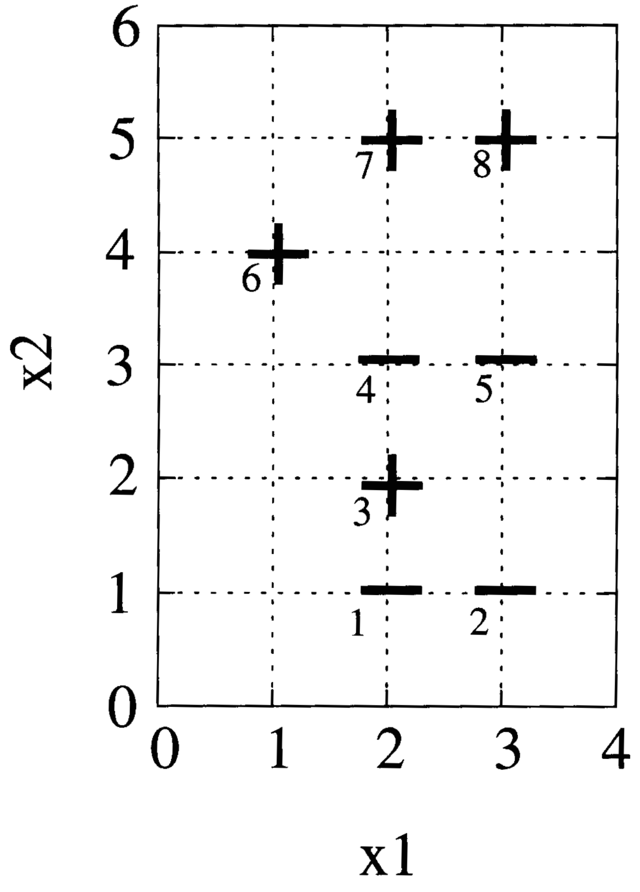
1. $h_1 : x1 > 1.5$
2. $h_2 : x1 > 2.5$
3. $h_3 : x2 > 1.5$
4. $h_4 : x2 > 2.5$
5. $h_5 : x2 > 3.5$
6. $h_6 : x2 > 4.5$

Only build the tree up to depth 2 even if the data is not perfectly classified! (That is, the resulting tree should only have at most 2 decision points along any path from root to leaf in the identification tree.) Also, **break ties** by using as preference order that in which the thresholds are given above: h_1 **is the most preferred while** h_6 **is the least preferred.**

Part C, Boosting (30 points)

Boosting notes have been reproduced in the supplement.

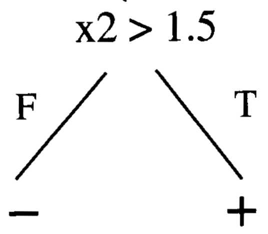
Consider the following training set. The number of each data point is drawn near that data point in the plot. (**Reproduced in supplement**)



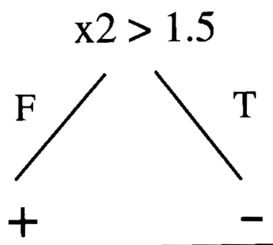
In this problem, the weak learner used by AdaBoost only considers each of the decision stump classifiers g_i, given in the next page, and outputs the one with the smallest (weighted) error. In case of ties, it outputs the classifier with the smallest index: g_7 is the **most preferred** while g_{12} is the **least preferred** (g_1 to g_6 vanished in the final minutes of exam writing). Both a formal mathematical definition and a drawing of the decision stump of each classifier are given in the next page.

(Reproduced in supplement)

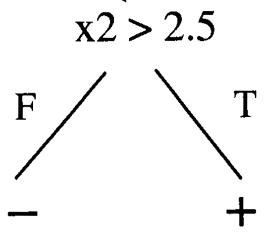
$$g_7(x) = \begin{cases} 1 & \text{if } x2 > 1.5, \\ -1 & \text{otherwise.} \end{cases}$$



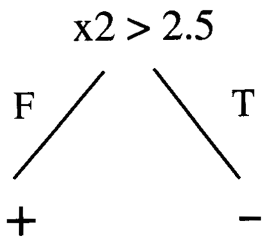
$$g_8(x) = -g_7(x)$$



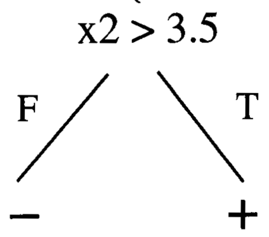
$$g_9(x) = \begin{cases} 1 & \text{if } x2 > 2.5, \\ -1 & \text{otherwise.} \end{cases}$$



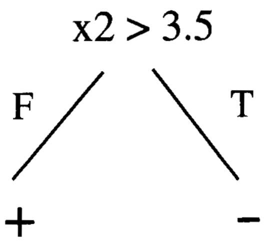
$$g_{10}(x) = -g_9(x)$$



$$g_{11}(x) = \begin{cases} 1 & \text{if } x2 > 3.5, \\ -1 & \text{otherwise.} \end{cases}$$



$$g_{12}(x) = -g_{11}(x)$$



Part C1 (14 points)

In this part, you are asked to complete a table (see below) corresponding to the weight, weak classifier, error, and classifier weight values generated by running AdaBoost for $T = 3$ iterations on the training data plotted above. The table has been partially filled out for you.

	Iteration t		
	1	2	3
$D_t(1)$	1/8	1/14	1/24
$D_t(2)$	1/8	1/14	1/24
$D_t(3)$	1/8	1/2	7/24
$D_t(4)$	1/8	1/14	6/24
$D_t(5)$	1/8	1/14	6/24
$D_t(6)$	1/8	1/14	1/24
$D_t(7)$	1/8	1/14	1/24
$D_t(8)$	1/8	1/14	1/24
h_t	g_{11}	g_7	g_{10}
ϵ_t	1/8	2/14	5/24
α_t	$\frac{1}{2} \ln 7$	$\frac{1}{2} \ln 6$	$\frac{1}{2} \ln 19/5$

The auxiliary tables that follow will not be graded but you can use them to help keep track of some of the calculations if you wish.

Data points that classifier mislabels			
g_7		g_8	
g_9		g_{10}	
g_{11}		g_{12}	

Classifier error at iteration $t = 2$			
g_7		g_8	
g_9		g_{10}	
g_{11}		g_{12}	

You may find the following formulas helpful.

$$\sqrt{x} = \frac{x}{\sqrt{x}}$$

$$e^{\ln a} = a$$

$$e^{\frac{1}{2}x} = \sqrt{e^x}$$

$$e^{-x} = \frac{1}{e^x}$$

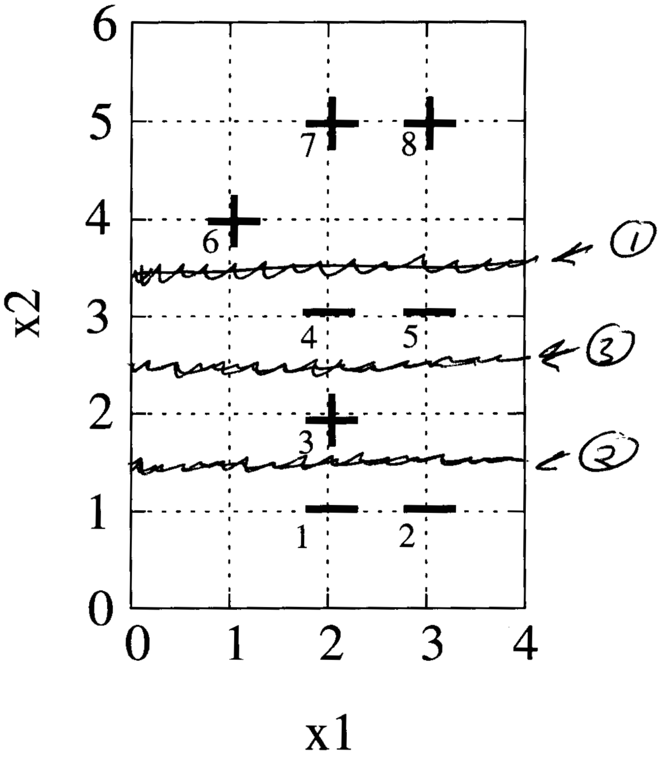
Part C2 (3 points)

Write down the final classifier $H(x)$. Please express only in terms of natural numbers and the natural logarithm function (ln). For example, if decision stumps g_7, g_8 and g_{10} are the weak classifiers selected during AdaBoost, then your answer should look like $H(x) = \text{sign}(\alpha_1 g_7(x) + \alpha_2 g_8(x) + \alpha_3 g_{10}(x))$, where α_1, α_2 and α_3 are appropriate weak-classifier weights found at each iteration, and expressed in terms of natural numbers and the natural-logarithm function.

$$H(x) = \text{sign}\left\{(\ln 7)\, g_{11}(x) + (\ln 6)\, g_7(x) + \left(\ln \tfrac{19}{5}\right) g_{10}(x)\right\}$$

Part C3 (4 points)

Draw the decision boundary for the final classifier $H(x)$ on the plot given below.



Part C4 (3 points)

Circle the numbers corresponding to the data points that the final classifier $H(x)$ classifies incorrectly, or circle "none" if all data points are correctly classified by $H(x)$.

[✓ none] 1 2 3 4 5 6 7 8

Part C5 (3 points)

Circle the numbers corresponding to the data points whose weights will **increase** if we were to continue with a fourth iteration of AdaBoost, or circle "none" if none will increase.

none [✓ 1] [✓ 2] 3 4 5 [✓ 6] [✓ 7] [✓ 8]

Part C6 (3 points)

Circle the numbers corresponding to the data points whose weights will **not change** if we were to continue with a fourth iteration of AdaBoost, or circle "none" if all will change.

[✓ none] 1 2 3 4 5 6 7 8