6.034 Final Examination
December 15, 2004

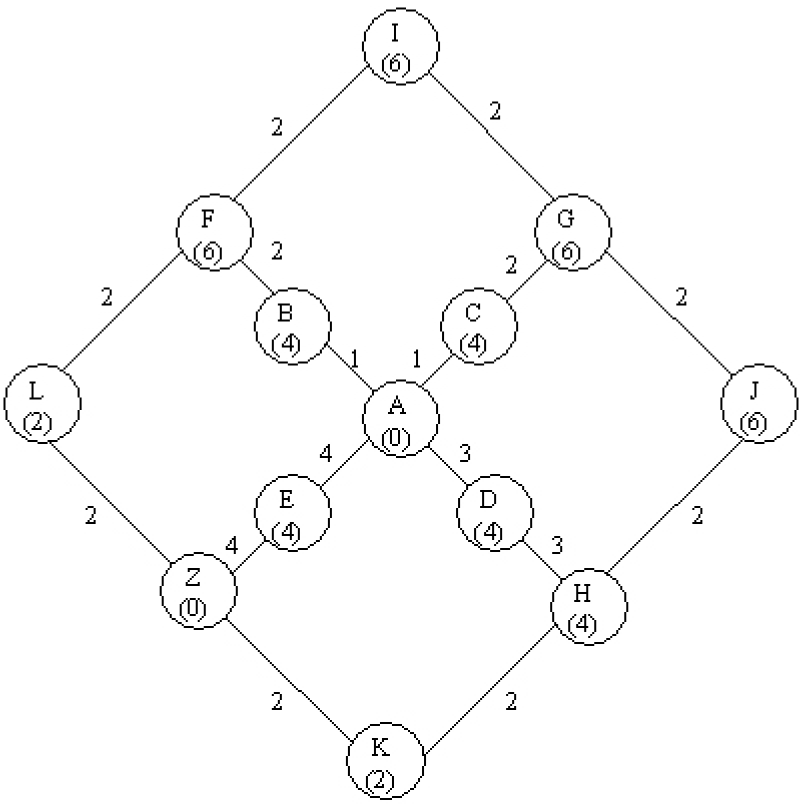
Name	
EMail	

There are many pages in this examination because we have reproduced many diagrams for your problem-solving convenience. We recommend you work the problems in order to maximize points per unit time.

Problem number	Maximum	Score	Grader
1	17		phw rcb lo sn vm llo lpb js
2	15		phw rcb lo sn vm llo lpb js
3	22		phw rcb lo sn vm llo lpb js
4	17		phw rcb lo sn vm llo lpb js
5	14		phw rcb lo sn vm llo lpb js
6	15		phw rcb lo sn vm llo lpb js
Total	100		

Problem 1: Search (17 Points)

Consider the following graph. Note that each link is bidirectional and labeled with its cost, and each node contains a number in parenthesis that corresponds to a heuristic estimate of the cost remaining to reach the goal node. For the following questions you are to search from node **A** to node **Z** (start at **node A** and find a path to **node Z**), using the prescribed search algorithm. Break any ties that arise by alphabetical order (extend the node that comes first alphabetically first). For this problem, the words extended and expanded are interchangeable. Make sure to search **from A to Z.**



Part A (1/2 point)

What node should you start your search from? Yes, it's that easy.

Part B (1/2 point)

What node should you search to? Yes, it's that easy.

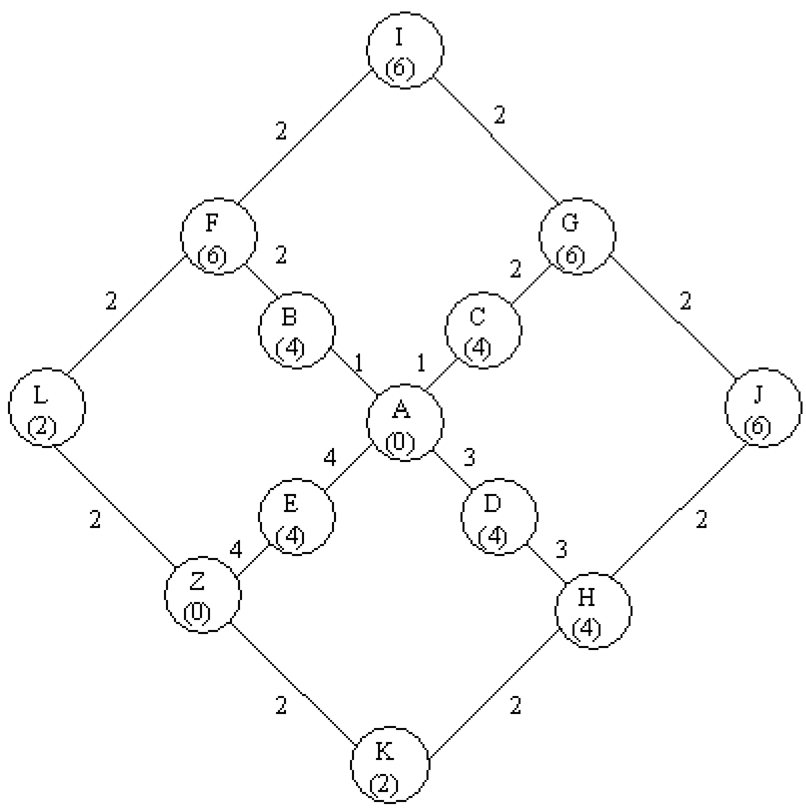
Part C (3 points)

You are to perform breadth-first-search. Do not use an enqueued list or an extended list. Do not extend any partial path to a node that it already contains. Use the convention that a path to the goal is only returned when the goal node is extended, not as soon as it appears in the queue.

Write down all the nodes extended, in the order in which they are extended.

Give the full path that is returned.

For your convenience, the graph is redrawn:



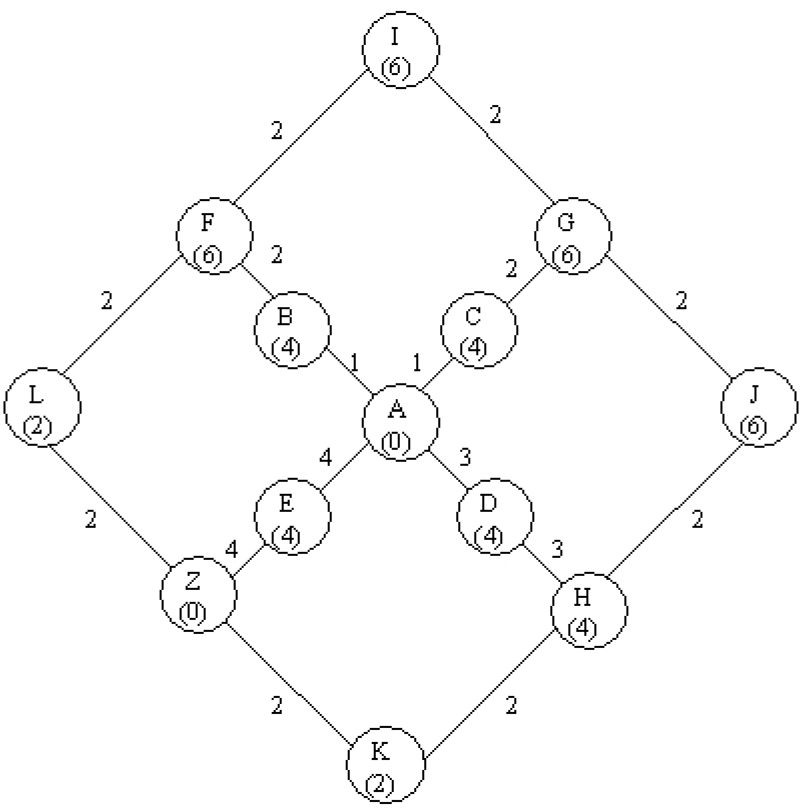
Part D (3 points)

Perform depth-first-search (with backup). Do not use an enqueued list or an extended list. Do not extend any partial path to a node that it already contains. Use the convention that a path to the goal is only returned when the goal node is extended, not as soon as it appears in the queue.

Write down all the nodes extended, in the order in which they are extended.

Give the full path that is returned.

For your convenience, the graph is redrawn:



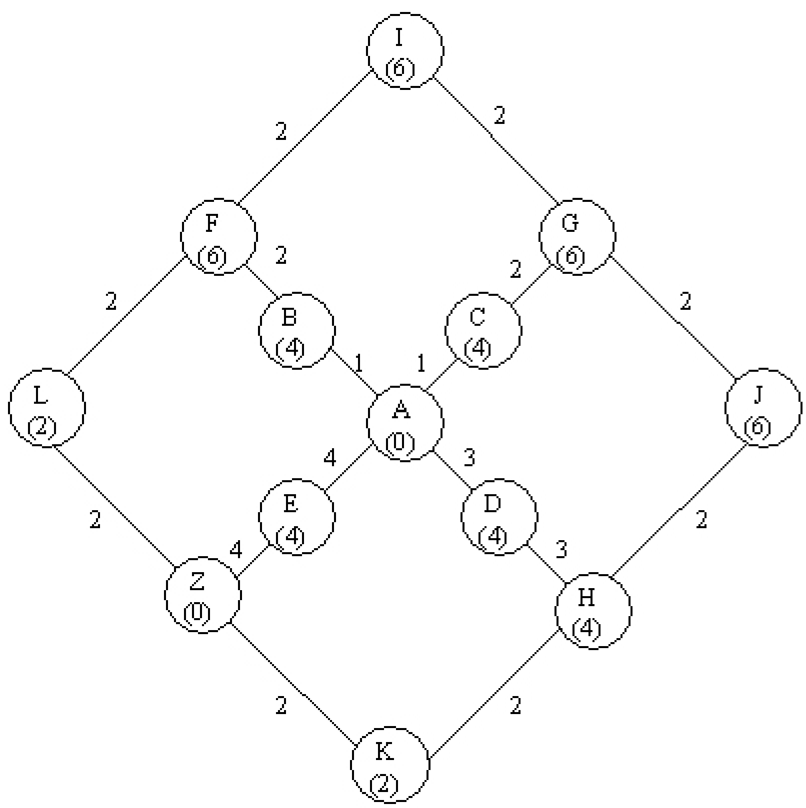
Part E (3 points)

Perform A* search using the estimates shown. As always with A*, you are to use an extended list. Do not extend any partial path to a node that it already contains.

Write down all the nodes extended, in the order in which they are extended.

Give the full path that is returned.

For your convenience, the graph is redrawn:



Part F (2 points)

Suppose you implemented depth-first-search, with backup, without an enqueued list or extended list but **forget to disallow the extension of partial paths to nodes which they already contain**. Would your depth-first-search implementation find a path from A to Z?

Yes No

Is it true that in general the specified search will always find a path from one node to another (in any graph), if such a path exists?

Yes No

Part G (2 points)

Suppose you implemented breadth-first-search, but **forget to disallow the expansion of partial paths to nodes which they already contain.** Would your breadth-first-search implementation find a path from A to Z?

Yes No

Is it true that in general the specified search will always find a path from one node to another (in any graph), if such a path exists?

Yes No

Part H (3 points)

Did your A* search produce the optimal path?

Yes No

Explain why or why not. Hint: check your answer against the diagram.

Problem 2: Constraint Satisfaction (15 points)

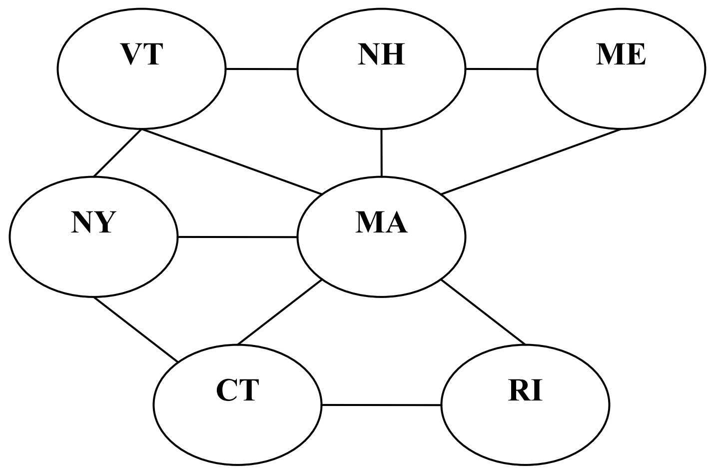
Tired of being overshadowed by New York City, upstate New York has claimed independence from the city of New York, and changed its name to New Yorkshire. As part of this campaign of reaffirmation of identity, New Yorkshire has asked to become part of New England. The governors of New England have reacted by forming a council to determine the admission of New Yorkshire to their group. Consensus has been reached in the council on the basis of map coloring:

(1) If maps of the Extended New England can still be colored Red Green and Blue, New YorkShire will be admitted.

(2) In addition to that, Massachusetts legislature (seeking an opportunity to forever define the political color of its state) has required that any new coloring should have Massachusetts shown in **Blue**

(3) Not willing to concede on its identity Vermont has further added the restriction that should New England be extended, any future map have Vermont shown in **Green** to honor their motto: “The Green Mountain State”.

Your task, as consultant to New England’s council in AI matters, is to determine if New England can be extended to include the brand new state of New Yorkshire.
In this problem you will be asked to color the Extended New England. The graph of extended new England is displayed in the figure to the right.
The edges in the graph are to be seen as **not-same** constraints.



1 Domains (1 point)

Write Down the initial domain of all the variables according to the specifications of the council:

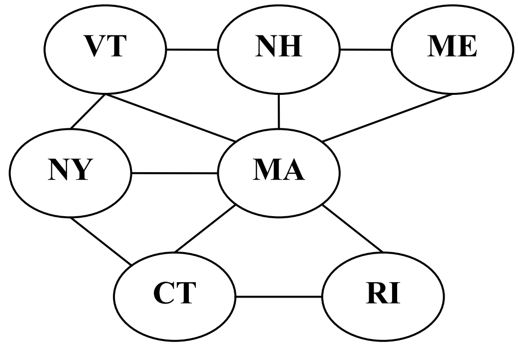
CT	MA	ME	NH	RI	VT	NY
{R, G, B}	{ }	{ }	{ }	{ }	{ }	{ }

In the following three questions you will be asked to solve the constraint satisfaction problem using:

1) Backtracking with Forward Checking + propagate through domains of one
2) Backtracking with Forward Checking
3) Backtracking

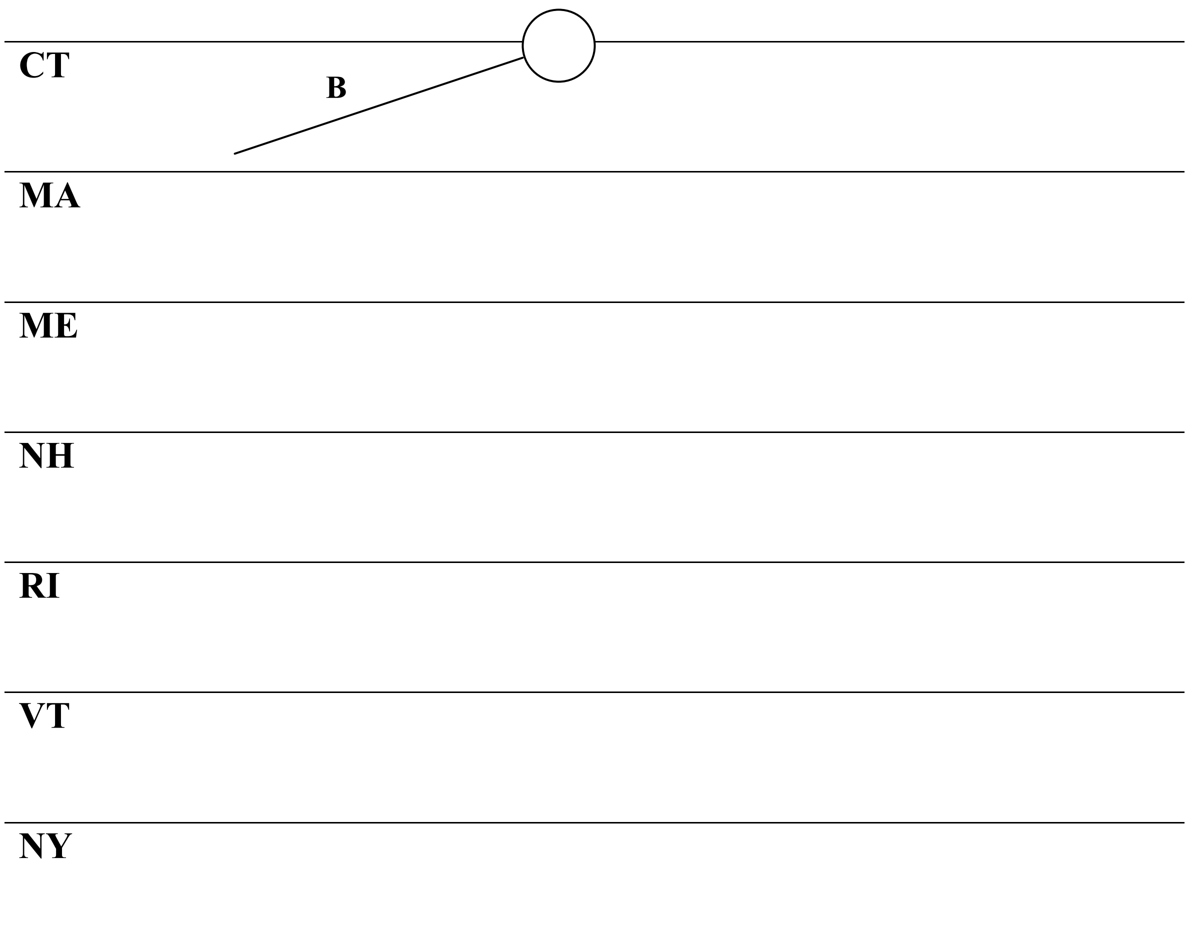
Note that we have ordered the searches we ask you to do so that the one that we think generates the largest tree is last. For your convenience the graph is presented in every page. .

2. **Backtracking + forward checking + propagate through domains of one (5 points)**

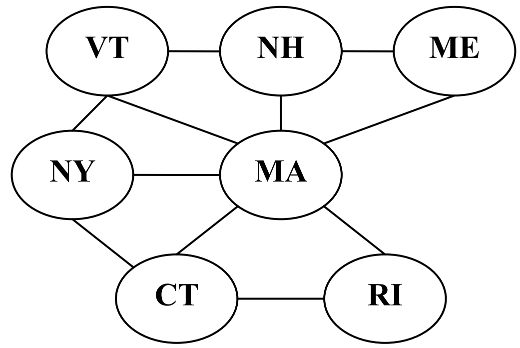


In this problem you must examine the values for each domain in the following order: **B →R →G.**

Remember the order is **B → R →G.**



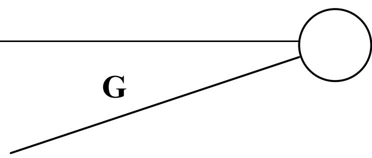
3. **Backtracking + Forward Checking (4 points)**



In this problem you must examine the values for each domain in the following order: **G → B →R** As usual, you are to check for consistency after each variable assignment.

Remember the order is now **G → B →R.**

CT



MA

ME

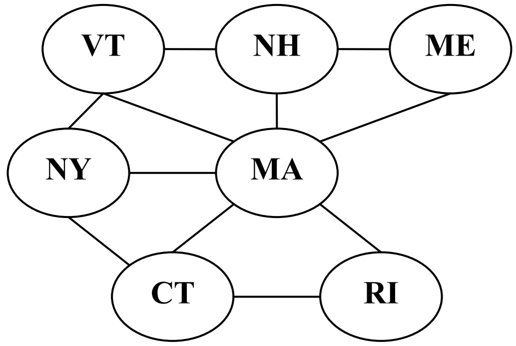
NH

RI

VT

NY

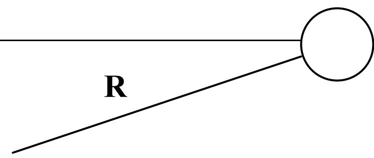
4. **Backtracking (4 points)**



In this problem you must examine the values for each domain in the following order: **R → G → B.** As usual, you are to check for consistency after each variable assignment.

Remember the order is now **R → G → B.**

CT



MA

ME

NH

RI

VT

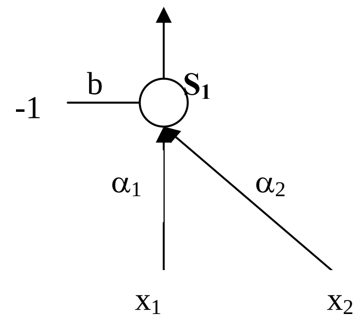
NY

Problem 3: Neural Networks (22 points)

In this problem you are asked to build a neural network step by step, and then reason about the properties of this network. The problem has many questions but the amount of work required is much less than it may seem at first glance. We first guide you in constructing a particular type of neural network. Then, we ask you to reason about the network you just built.

Building one Neural Network Unit (5 parts, 2 points)

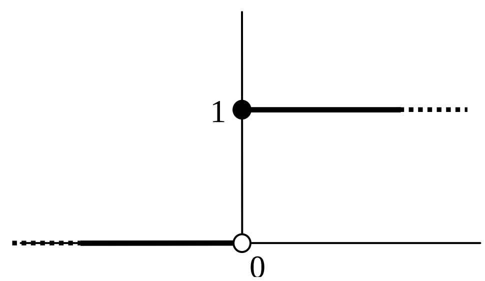
Consider the network unit shown on the right



At first, we are interested in building a network using a perceptron type step function as the threshold unit rather than a sigmoid functions. Formally, the step function threshold unit is defined as follows:

$$S(input) = \begin{cases} 0 & if\ input < 0 \\ 1 & if\ input \geq 0 \end{cases}$$

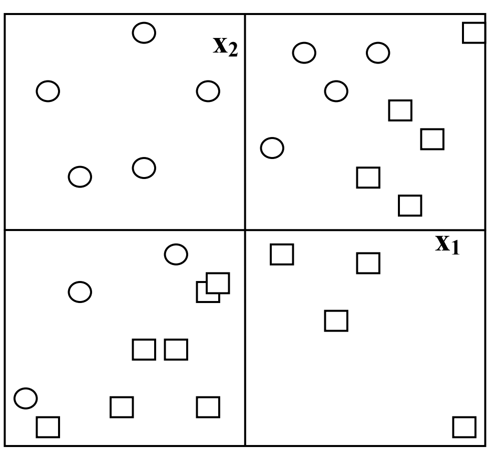
The step function is shown in the figure to the right. **Note the convention adopted for the output when input=0, which will matter later on.** With this definition, the output of the unit in the figure can be written as



$$\mathbf{y_1 = S_1(\alpha_1\, x_1 + \alpha_2\, x_2 - b).}$$

Where S_1 is a step threshold function.

Consider the following data, in which **squares** stand for class y=0 (**□, y=0)**, and **circles** stand for class y=1**, (○,y=1))**:



1. Draw the decision boundary

2. Write the equation for the decision boundary

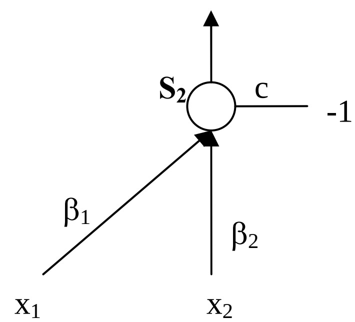
Implementing the decision boundary for the unit above given this sample data implies finding a relationship between the alphas α.

3. What is the relationship between the alphas α?

4. What should the offset (b) be.

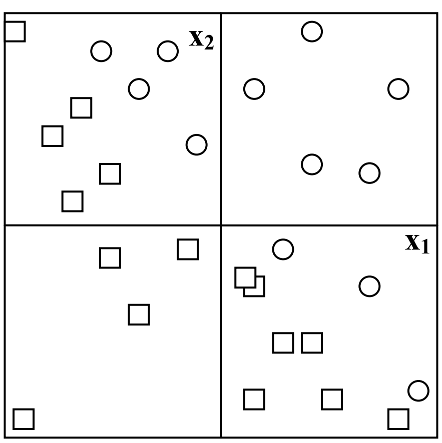
5. Is there any additional constraint on α_1 needed to ensure that the network produces the right answer on each side of the boundary? (Yes/No, if yes, write it down)

Building another Unit (5 parts, 1 point)



Consider now a second Neural Network unit, like the one on the right. You should use the same step threshold function as in part 1.

Consider the following data:



6. Draw the decision boundary

7. Write the equation for the decision boundary

8. What is the relationship between the betas β.?

9. What should the offset (c) be?

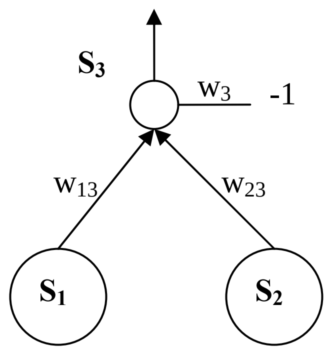
10. Is there any additional constraint on β_1. (Yes/No, if yes, write it down)

Building the final unit (1 part, 1 point)

We want to construct a third unit that takes as input the outputs of S1 and S2.:

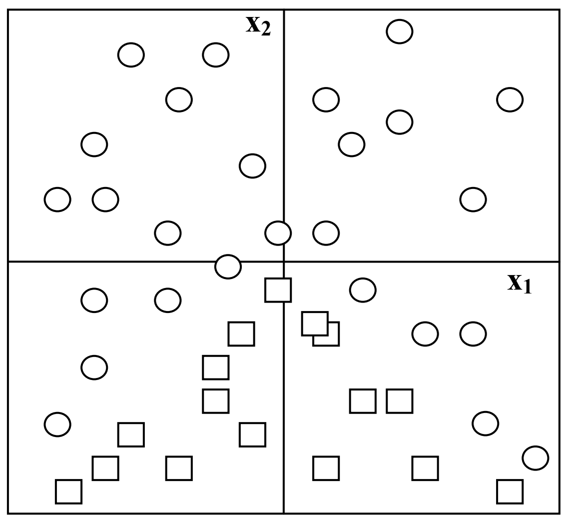
11. Complete the expression for output of the third threshold unit in terms of:

x_1 and x_2,
the functions S_1 and S_2,
and your knowledge about the αs and the βs.



S_3(**)**

For the rest of the problem we will consider the following sample Data.



This data may look familiar, but as a wise Spanish proverb says: "At night, all cats look brown," which implies that you should wait until light shines on the problem before jumping to conclusions.

So are the weights constrained? (4 parts, 8 points)

Consider the three data points

X_1	x_2
0	-2
3	0
-2	1

12. Looking at the formula you obtained in part 11, use the table above to determine three constraints on the weights for the final unit (w_{13}, w_{23}, w_3)

c1

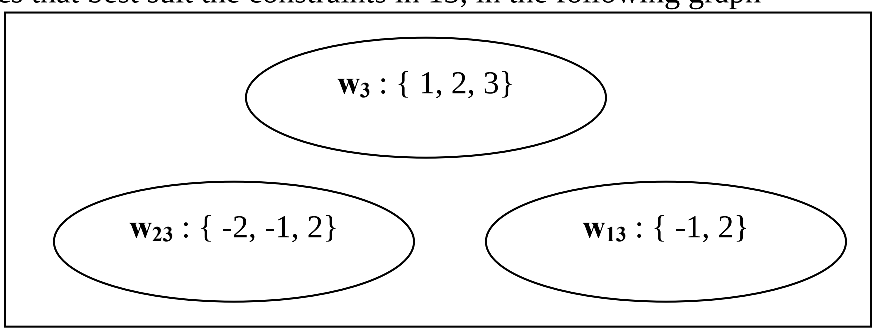
c2

c3

Consider now the following variables and domains:

Variable	Domain
$\mathbf{w_{13}}$	{-1,2}
$\mathbf{w_{23}}$	{-2,-1,2}
$\mathbf{w_3}$	{1,2,3}

13. Draw the arcs that best suit the constraints in 13, in the following graph



14. Run ***constraint propagation*** (a.k.a. ***arc consistency***) on the above graph, and fill the table below as you proceed (if some arc does not exist in your graph, just **cross** it out) :

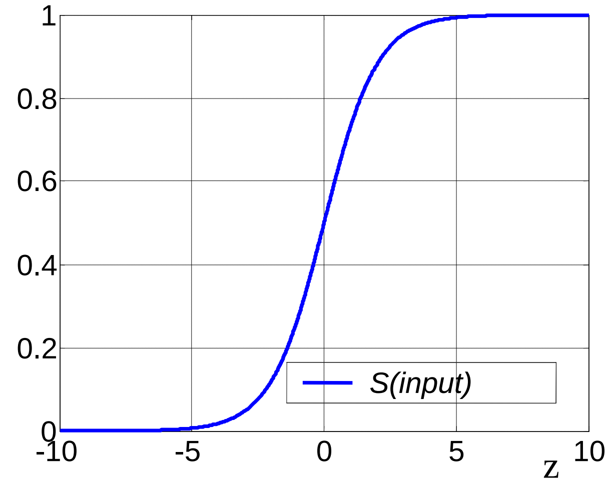
Arc	Resulting domain
$\mathbf{w_3 \rightarrow w_{23}}$	$\mathbf{w_3} \in \{ \quad \}$
$\mathbf{w_{13} \rightarrow w_3}$	$\mathbf{w_{13}} \in \{ \quad \}$
$\mathbf{w_{23} \rightarrow w_{13}}$	$\mathbf{w_{23}} \in \{ \quad \}$
$\mathbf{w_{13} \rightarrow w_{23}}$	$\mathbf{w_{13}} \in \{ \quad \}$
$\mathbf{w_3 \rightarrow w_{13}}$	$\mathbf{w_3} \in \{ \quad \}$
Cassini → Titan	**Huygens** ∈ {crashed, landed}
$\mathbf{w_{23} \rightarrow w_3}$	$\mathbf{w_{23}} \in \{ \quad \}$

15. What are the possible solution(s) you get?

A sigmoid neural network (5 parts, 10 points)

Now asume that we want to use sigmoid units instead of step units. Recall that a sigmoid unit would look like:

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



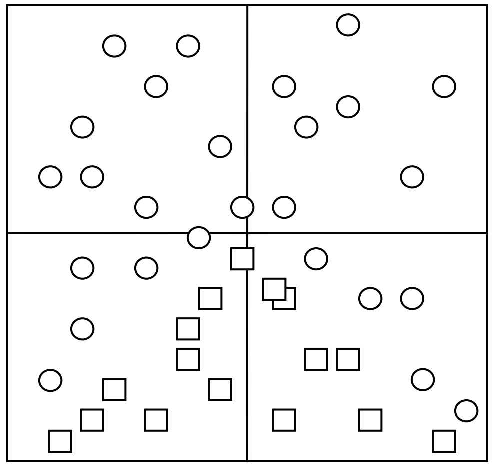
16. Show the new output of $\mathbf{s_3}$ in terms of $\mathbf{x_1}$ and $\mathbf{x_2}$. **Note that s1 and s2 must not appear in your solution.** Look back at question 11 and replace the step functions that appeared there by the sigmoid expression. Set the αs and the βs to +1 or -1 in accordance with your answers to part 5 and part 10. Simplify the result as much as you can.

s_3( )

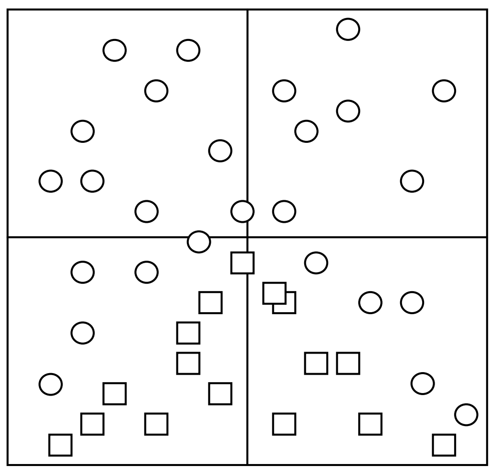
17. Recall that the output of a sigmoid function is continuous. However, for classification, we use the following relation:

$$class = \begin{cases} 0 & if\ s(input) < 1/2 \\ 1 & if\ s(input) \geq 1/2 \end{cases}$$

Draw *approximately* the decision boundary of the **sigmoid network** if the weights in the final unit were: $\mathbf{w_{13}}$=2, $\mathbf{w_{23}}$=2, $\mathbf{w_{3}}$=2. and the αs and βs are +1 or -1, as determined in part 16.



18. Do the same assuming the weights were $\mathbf{w_{13}}$=2, $\mathbf{w_{23}}$=2, $\mathbf{w_{3}}$=1 using the same αs and βs as before.

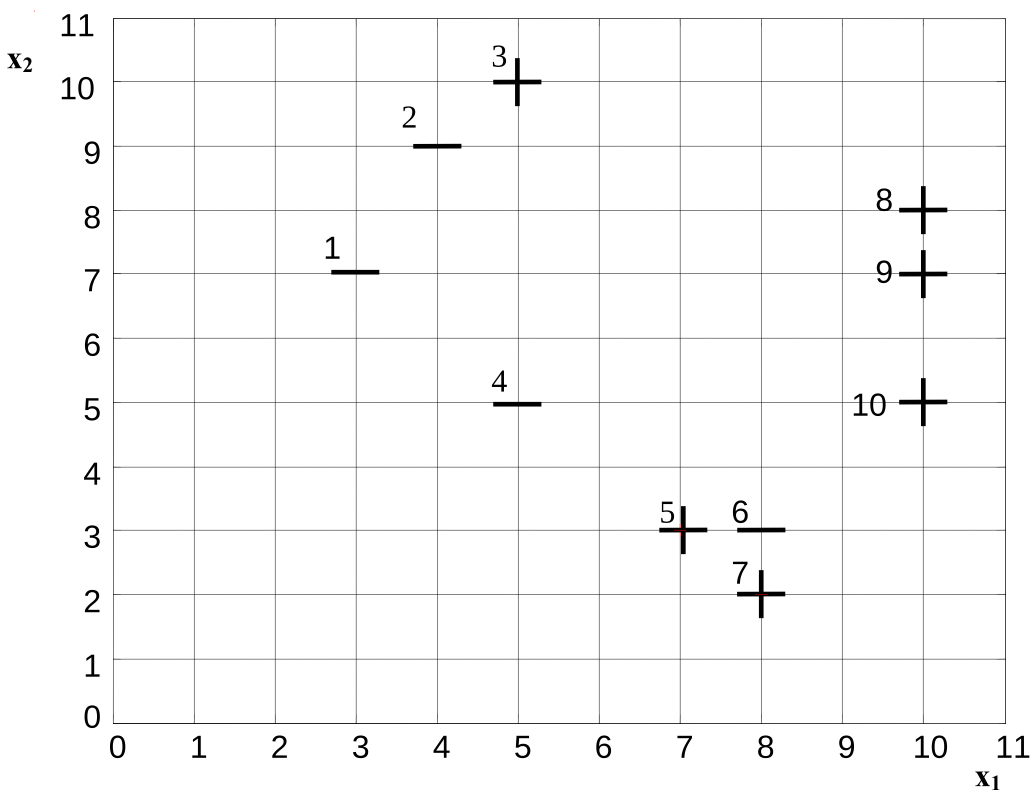


19. Comment on the most salient difference, if any, between the decision boundaries produced by a sigmoid unit and a step threshold unit.

20. What would you change, if anything, in the sigmoid function to arbitrarily approximate the decision boundary when using the step function?

Problem 4: Boosting (17 points)

Consider the following dataset, in which points are numbered for easier identification later on and where plus and minuses show the class information. For your convenience, **the diagram is reproduced on a tear-off sheet at the end of this examination.**



For this problem, we will consider the following set of decision stumps to be the hypothesis space of the weak learner. (A decision stump is a one-test decision tree.)

Stump	Variable	Threshold
h1	x_1	6
h2	x_1	7.5
h3	x_2	4

Remember that the signs at each side of these thresholds are to be determined based on the training data. For example, looking at the x2=4, with initial uniform weights, both sides of the boundary are going to be positive, and thus there will be 4 mistakes (the 4 minuses).

__**Tie break-up:**__

a) If when determining the class at each side of the threshold you get a tie, you shall ignore the weights and decide based on regular count of each class.

b) If when choosing between stumps, you get a tie, you shall choose by the number associated with each stump; **lower number takes precedence**.

Part A: 2D data, … crunch the booster(2 parts, 13 points)

1. (10 points) Boosting is all about iterating. Fill in the two tables below with the result of carrying on three iterations of the boosting algorithm. **At each iteration, choose the classifier with the lowest *ε*.** Remember that

- **D(i)** stands for the weight of sample point ***i***.
- ε is the error of the classifier chosen at that iteration
- α is the weight of the classifier in the final majority vote

Note, filling in the first two iterations is worth 80% of the points of this question, filling the third column is worth the remaining 20%.

Stump error	***t=1***	***t=2***	***t=3***
h1			
h2			
h3			

Note, even if it is possible to start boosting with an arbitrary set of weights on the data, for this problem you are expected to **start with uniform weighting** of the data.

We have filled in one of the answers for you so as to demonstrate the form we require for your answers.

Iterations	***t=1***	***t=2***	***t=3***
D(1)			
D(2)			
D(3)			
D(4)			
D(5)			
D(6)			
D(7)			
D(8)			
D(9)			
D(10)			
Stump(h1,..h3)			**h1, + if $x_1 \geq 6$** **- if $x_1 < 6$**
ε			
α			

2. (3 points) Classify the training dataset according to the final classifier depending on the iteration you stopped at, or feel more confident with.

I stopped at t = [ ]

Sample Point	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Class (+/-)										

Part B: 3D data, … fooling the booster (2 parts, 4 points)

We have just received word that the data above was missing one dimension. Here are the extra feature values for each data point:

Sample Point	x_3
1	.9
2	1
3	2
4	.5
5	1.5
6	0.8
7	2.5
8	1.75
9	2.1
10	1.9

Assume the stumps for this dimension have threshold values at the midpoints between adjacent sample values. Assume we restart boosting with uniform weights on the samples

3. (2 points) What can you reuse from part 1?

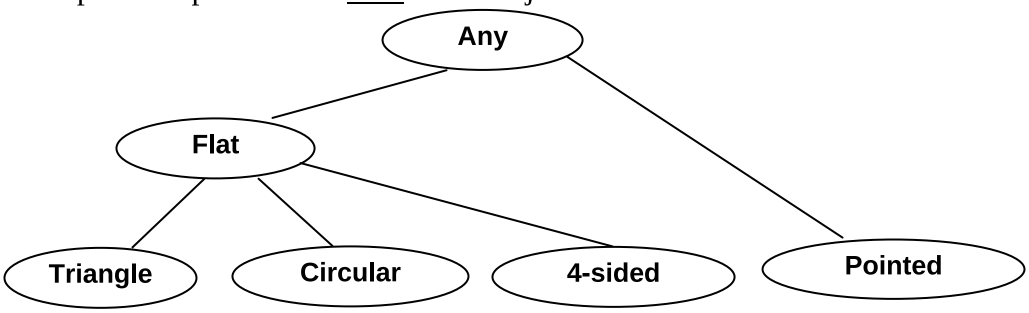
a. The errors of the stumps (h1-h3), and the initial weights
b. The α values of the winning stumps.
c. (a) and the weights for each iteration.
d. Everything.
e. All of the above
f. None of the above

4. (2 points) Boosting actually does not help when given this problem. Explain why.

Problem 5: Concept Learning (14 points)

You wish to teach Marvin the Robot the concept **CUP** using images A through G below. Marvin, as it happens, uses Winston's concept learning system, along with a vision system that delivers for each image:

1. a shape description of the base of the object:



2. whether the object has a handle or not (only A, F, and G have handles)
3. whether the object is closed or open (only C and F are closed)
4. the ratio of width to height – a positive real number

Image	***A***	***B***	***C***	***D***	***E***	***F***	***G***
Ratio	0.5	0.5	0.5	2	0.4	0.3	0.35

Here are the 6 images you present to Marvin: (Hits are "positive" examples, while misses are "negative" examples). Images A, B, C, D, F are circular, E is pointed, G is 4-sided.

A - HIT

B - HIT

C - MISS

D - HIT

E - HIT

F – MISS

G - MISS

PART A: (6 points)

For each of the images presented to Marvin, write an English description of the model Marvin builds, and determine which heuristics, if any, Marvin uses to update his model:

Image	***Heuristics Used***	***Marvin's Model of* CUP**
A	None	has handle, is open, circular bottom, ratio=.5
B		
C		
D		
E		
F		
G		

Here are the list of heuristics Marvin could use: (see pg. 356 in Chapter 16, reproduced on tear-of sheet at the end of the examination)

require-link, forbid-link, climb-tree, enlarge-set, drop-link, close-interval

PART B: (4 points)

Which images cause Marvin the Robot to **<u>generalize</u>** his model? __________________

Which images cause Marvin the Robot to **<u>specialize</u>** his model? __________________

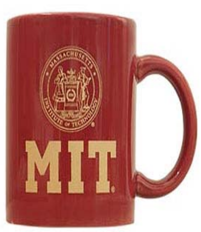
PART C (4 points)

Suppose you trained Marvin the Robot on the concept **POT**, and Marvin arrived at the following model:

Marvin's Model:

> **POT**: must be open, must have handle, and $1.4 < \text{width/height} < 1.6$

You have a choice to present Marvin with one of the 5 images:

A	***B***	***D***	***X***	***Y***
				
has handle, open, flat bottom, width/height=.5	no handle, open, flat bottom, width/height=.4	no handle, open flat bottom, width/height=2.0	has handle, open, flat bottom, width/height=1.5	has handle, open, flat bottom, width/height=3.0
MISS	MISS	MISS	HIT	HIT

Which image should you present to Marvin next?

Give a short explanation of why you chose the image below:

Problem 6: Powerful Ideas (15 points)

Explain in four terse sentences or fewer how you would use self organizing maps to identify unusual events in a corpus of children's books. You are welcome to include diagrams in your explanation, but you can get full credit without any.

Diagrams:

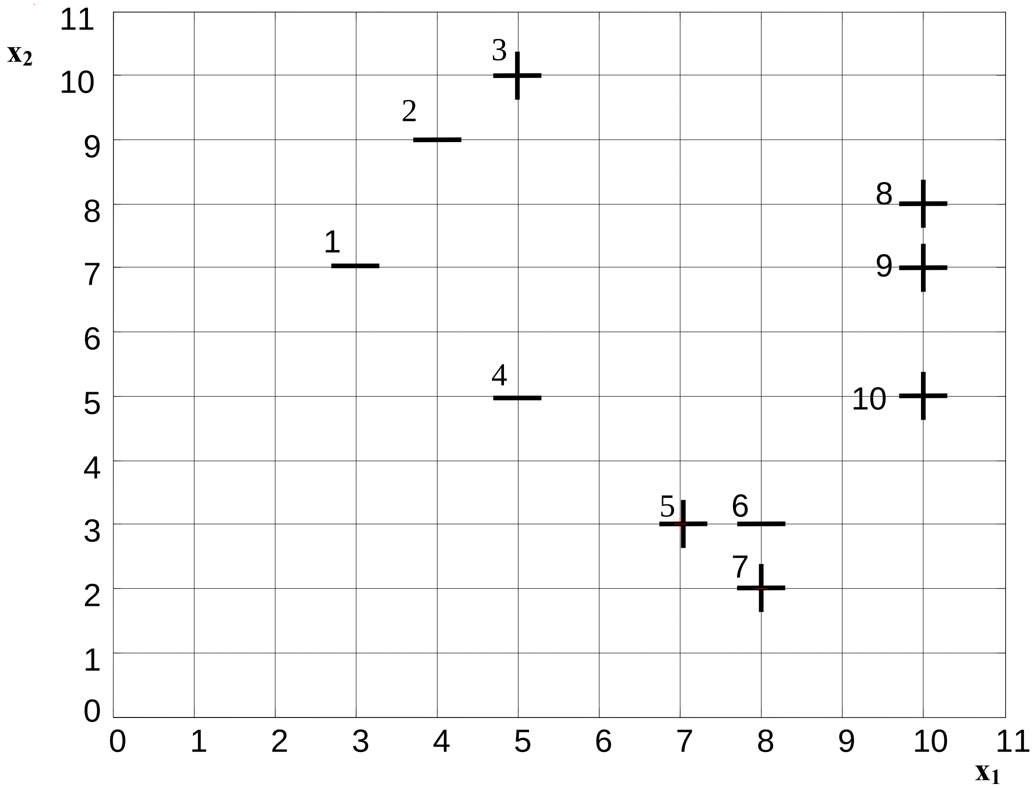
Explain in four terse sentences or fewer how Yip and Sussman synthesized data in their phonological-rule learning program and how the synthesized examples are related to examples used by Winston's arch-learning program. You are welcome to include diagrams in your explanation, but you can get full credit without any.

Dagrams:

Explain in four terse sentences or fewer how you would be guided by Sinha's experiments if you were to try to write a face recognition program. You are welcome to include diagrams in your explanation, but you can get full credit without any.

Dagrams:

Tear-off sheet---you need not hand this in. Do not write any answers on this page.



Tear-off sheet---you need not hand this in. Do not write any answers on this page.

- The **require-link** heuristic is used when an evolving model has a link in a place where a near miss does not. The model link is converted to a Must form.
- The **forbid-link** heuristic is used when a near miss has a link in a place where an evolving model does not. A Must-not form is installed in the evolving model.
- The **climb-tree** heuristic is used when an object in an evolving model corresponds to a different object in an example. Must-be-a links are routed to the most specific common class in the classification tree above the model object and the example object.
- The **enlarge-set** heuristic is used when an object in an evolving model corresponds to a different object in an example and the two objects are not related to each other through a classification tree. Must-be-a links are routed to a new class composed of the union of the objects' classes.
- The **drop-link** heuristic is used when the objects that are different in an evolving model and in an example form an exhaustive set. The drop-link heuristic is also used when an evolving model has a link that is not in the example. The link is dropped from the model.
- The **close-interval** heuristic is used when a number or interval in an evolving model corresponds to a number in an example. If the model uses a number, the number is replaced by an interval spanning the model's number and the example's number. If the model uses an interval, the interval is enlarged to reach the example's number.