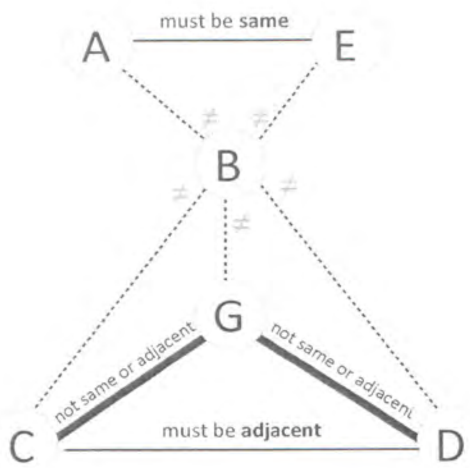
Constraint graph for this problem




Domains for this problem

*(Do not reduce these domains before starting search.)*

| | |
|---|---|
| A | 1 2 3 4 5 |
| B | 1 2 3 4 |
| C | 1 2 3 4 5 |
| D | 1 4 |
| E | 3 |
| G | 1 2 3 4 5 |

Fill out this worksheet as you draw your search tree. There may be more rows than you need.

1. Every time you **assign a variable** or **remove a variable from the propagation queue**, fill out a new row in the table. (The same variable might appear in more than one row, especially if you have to backtrack.)
2. In that row, indicate **which variable you assigned or de-queued**; write its **assigned** value if it has one (e.g. X=x), otherwise just write its **name** (X). In the second column, list the **values that were just eliminated from neighboring variables** as a result. If no values were just eliminated, write **NONE** instead.
3. If your search has to backtrack after assigning or de-queuing a variable: first, **finish listing** all values eliminated from neighboring variables in the current row. Next, check the "backtrack" box in that row. Then, continue with the next assignment in the following row as usual.
4. If you add several variables to your propagation queue at once, break ties by adding variables to your propagation queue **in alphabetical order**.

E has no more options so we back track

...es ( singleton domain

| | Var assigned or de-queued | List all values just eliminated from neighboring variables | Back track |
|---|---|---|---|
| 1 | A=1 | E≠3, B≠1 | ☒ |
| 2 | A=2 | E≠3, B≠2 | ☒ |
| 3 | A=3 | B≠3 | ☐ |
| 4 | E=3 | NONE | ☐ |
| 5 | D=1 | B≠1 C≠3,4,5,1 G≠1,2 | ☐ |
| 6 | C | B≠2 G≠ ~~1~~,3 | ☐ |
| 7 | B | G≠4 | ☐ |
| 8 | G | NONE | ☐ |
| 9 | B=4 | NONE | ☐ |
| 10 | G=5 | NONE | ☐ |
| 11 | C=2 | NONE | ☐ |
| 12 | | | ☐ |
| 13 | | | ☐ |
| 14 | | | ☐ |

don't eliminate values that have already been eliminated

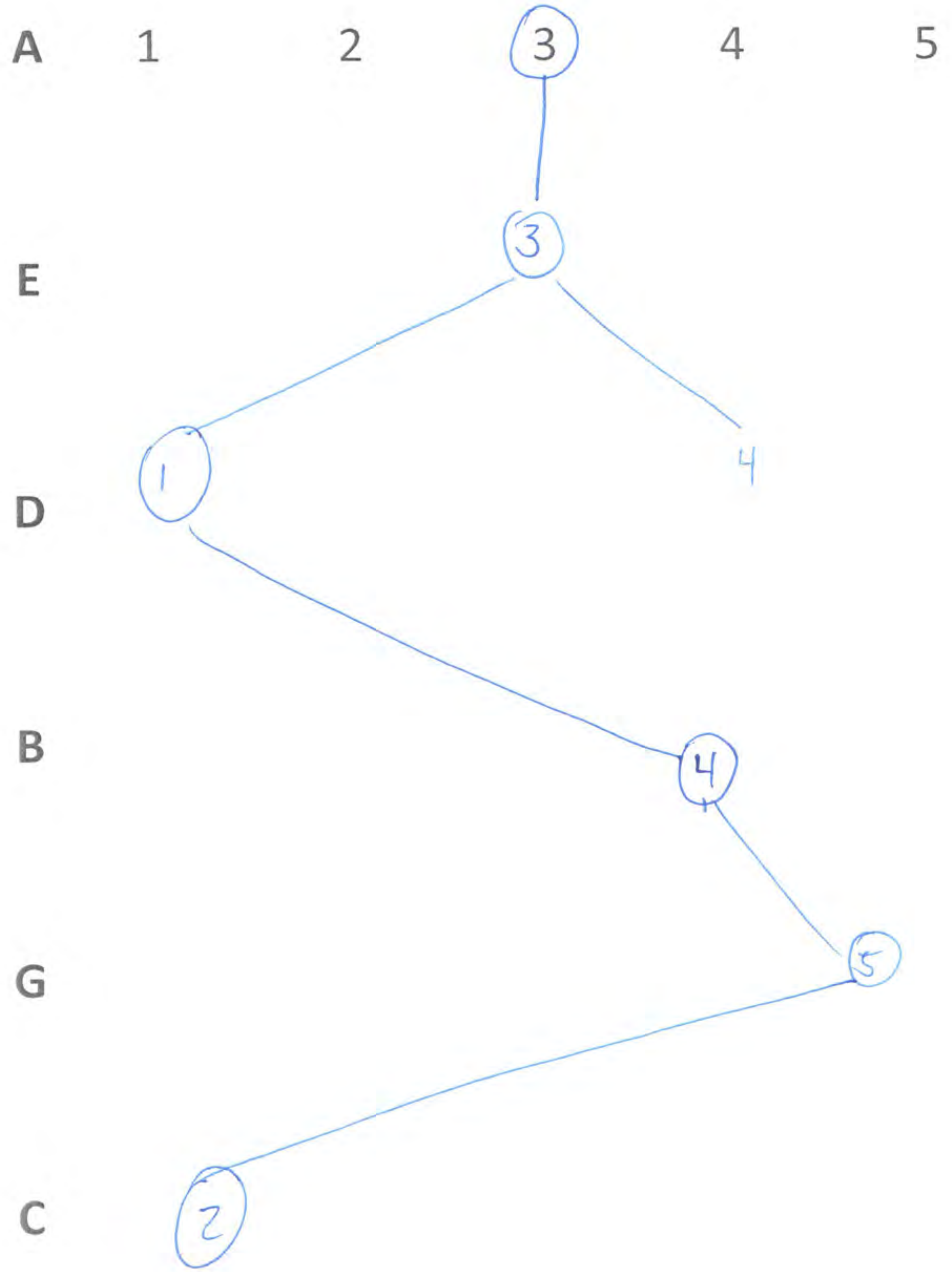
Example row showing an assigned variable

| | | | |
|---|---|---|---|
| ex | X = 3 | Y ≠ 3, 4 Z ≠ 3 (example) | ☑ |

Example row showing a de-queued (propagated) variable

| | | | |
|---|---|---|---|
| ex | X | W ≠ 1, 4 (example) | ☐ |

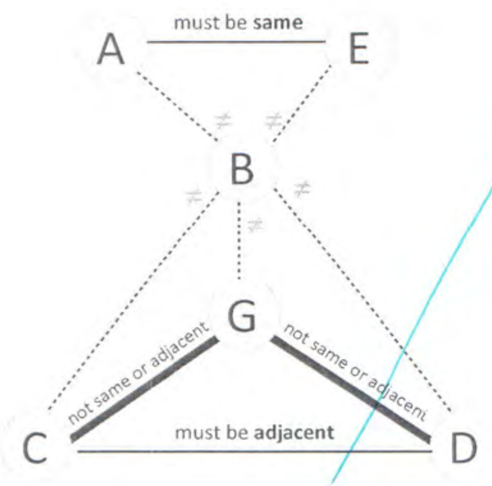
Draw your search tree on this page.



**Note: This is a duplicate page provided for your convenience. Use this copy only if you want to start over. There are more questions on pages 14-15.**

☐ **I want to start over; grade these pages.**

Constraint graph for this problem




Domains for this problem

*(Do <u>not</u> reduce these domains before starting search.)*

| | |
|---|---|
| A | 1 2 3 4 5 |
| B | 1 2 3 4 |
| C | 1 2 3 4 5 |
| D | 1 4 |
| E | 3 |
| G | 1 2 3 4 5 |

| | Var assigned or de-queued | List all values just eliminated from neighboring variables | Back track | | Var assigned or de-queued | List all values just eliminated from neighboring variables | Back track |
|---|---|---|---|---|---|---|---|
| 1 | | | ☐ | 8 | | | ☐ |
| 2 | | | ☐ | 9 | | | ☐ |
| 3 | | | ☐ | 10 | | | ☐ |
| 4 | | | ☐ | 11 | | | ☐ |
| 5 | | | ☐ | 12 | | | ☐ |
| 6 | | | ☐ | 13 | | | ☐ |
| 7 | | | ☐ | 14 | | | ☐ |

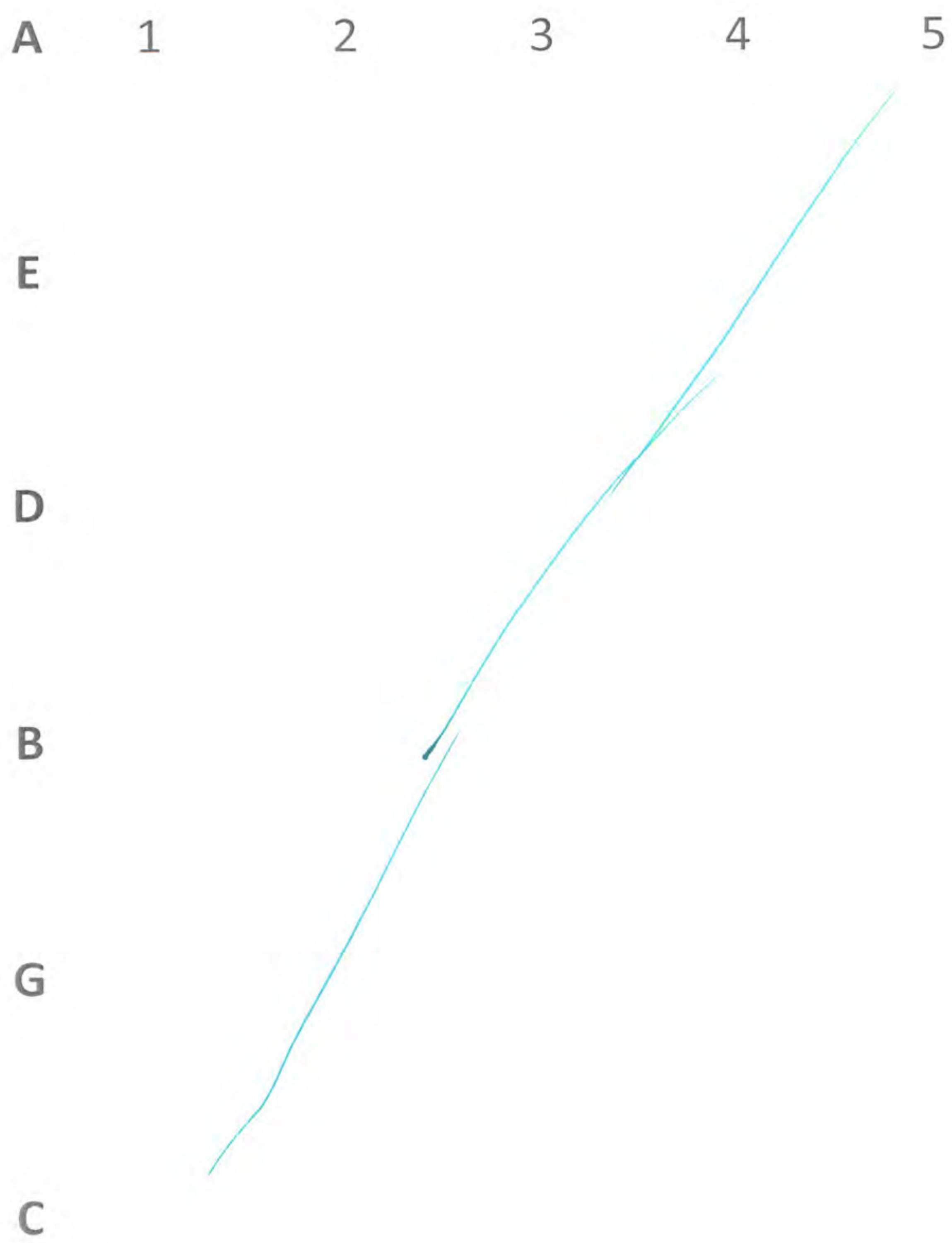
**Example row showing an assigned variable**

| | | | |
|---|---|---|---|
| ex | X = 3 | Y ≠ 3, 4 Z ≠ 3 (example) | ☑ |

**Example row showing a de-queued (propagated) variable**

| | | | |
|---|---|---|---|
| ex | X | W ≠ 1, 4 (example) | ☐ |

[This is a duplicate copy provided for your convenience.]
Draw your search tree on this page.



## Part C (6 points)

**C1 (3 points)** Based on the results of your search in Part B, write the initials of the mages (**A, B, C, D, E, G**) in their assigned homes among the five worlds.

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| D | C | AE | B | G |

(If no solution exists, circle NO SOLUTION instead.)

NO SOLUTION

**C2 (3 points)** Based <u>only on the tree</u> you drew in Part B, what can you conclude about the number of solutions to this problem (i.e. consistent ways of assigning mages to worlds)?

(A) There are **no solutions** to this problem—it's over-constrained. → would have to back track till no solutions left

(B) There is **exactly one** solution to this problem. → would have to draw the whole tree

(C) There is **at least one** solution to this problem, maybe more.

(D) There is **definitely more than one** solution to this problem. → would need to keep searching for another solution

## Problem 3: Spiritual and Right-Now

Circle the **one best** answer for each of the following question. There is **no penalty for wrong answers**, so it pays to guess in the absence of knowledge.

1. Sussman is motivated by a desire to understand:

   1. How English emerged from Germanic and French roots.
   2. How to write more secure software.
   3. How to write provably correct software.
   4. How thinking works.
   5. How to treat autism.

2. Sussman's explanation of propagators featured an example involving:

   1. How stories change in time as they are retold.
   2. Fast implementation of rule chaining.
   3. Calculating the distance to a galaxy.
   4. Proving a program to be correct.
   5. Tweets.

3. In Sussman's propagator system, cells:

   1. Pass on information distorted by a random number generator.
   2. Perform complex logical operations using table lookup.
   3. Achieve great speed by using multiple cores.
   4. Hold information about values, rather than values.
   5. Have values that reflect cultural beliefs.

4. Boyden describe how rats can be:

   1. Trained to stand on two legs.
   2. Respond to light without normal retinal photoreceptors.
   3. Rendered vicious by cauterization of areas in the forebrain.
   4. Shown to dream about movement through a familiar maze.
   5. Made obese by electrical stimulation of brain areas associated with hunger.

5. Boyden described a procedure involving direct electrical stimulation of brain tissue so as to:

   1. Cure chronic indigestion.
   2. Suppress disturbing memories.
   3. Recall childhood experiences.
   4. Stop hand tremors.
   5. Reduce symptoms associated with Tourette's syndrome.

6. Optogenetics has to do with:

   1. Evolution of the mammalian visual system.
   2. Optimization of brain circuitry by survival of the fittest.
   3. Optical stimulation of genetically altered neurons.
   4. Manipulation of the genetic code using lasers.
   5. Poor visual acuity associated with old age.

# 6.034 Quiz 3
# 15 November 2013

| Name | PRACTICE #2 |
| --- | --- |
| email | |

Circle your TA **(for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

**Michael Fleder** **Giuliano Giacaglia** **Dylan Holmes**

**Casey McNamara** **Robert McIntyre** **Duncan Townsend**

**Mark Seifter** **Sam Sinai** **Prashan Wanigasekara**

| Problem number | Maximum | Score | Grader |
| --- | --- | --- | --- |
| 1 | 50 | | |
| 2 | 50 | | |
| Total | 100 | | |

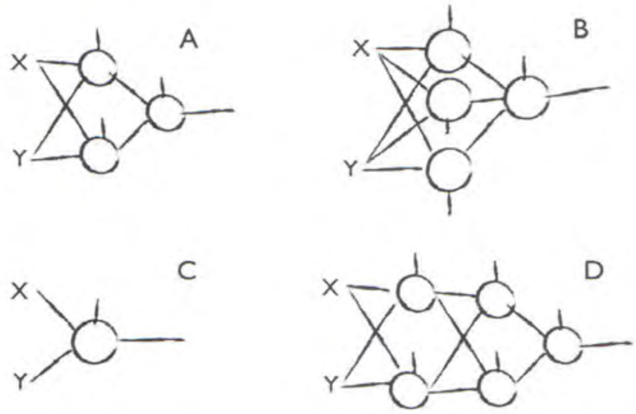
There are 8 pages in this quiz, including this one, but not including blank pages and tear-off sheets. Tear-off sheets with duplicate drawings and data are located after the final page of the quiz. As always, open book, open notes, open just about everything, including a calculator, but no computers.

# Problem 1: Neural Networks (50 points)

## Part A: The Paleplane puzzle (24 points)

After a strange incident involving a malfunctioning quiz problem, 6.034 TA Dylan Holmes has vanished from within Prof. Winston's locked office! Detective (and distant relative) Sherlock Holmes has just broken in to look for clues.

He finds four suspicious neural nets drawn on a whiteboard:




Nearby, he also finds ten ominous figures which may represent the output of some of the neural nets:

| A B D |
|---|

(A and B)

| NONE |
|---|

Not possible w/lines

| NONE |
|---|

A and (B XOR C)
=> 3 lines and 2 levels after

| D |
|---|

A XOR B

◦ Each neuron in 1st level represents line
-XOR - Needs 2 levels
AND, OR - Only need 1 level

| A B C D |
|---|

1 line

| B |
|---|

3 lines
AND

| B |
|---|

3 lines
OR, AND

| A B D |
|---|

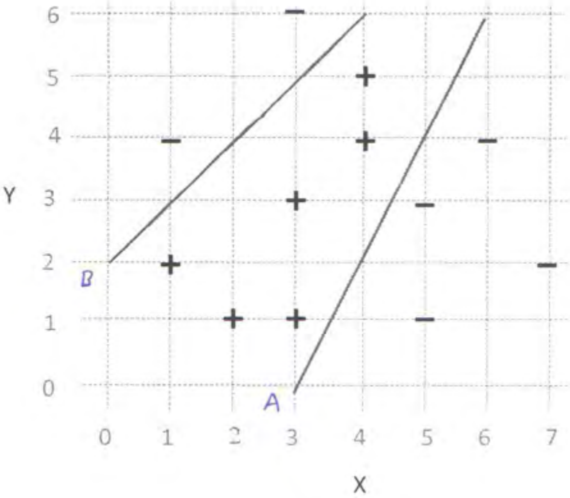
2 ANDS
(A AND B)

In the box below each figure, **list all** the neural nets (A, B, C, D) that could have produced that figure as output. **If none of the neural nets could produce the figure, write NONE instead.** Assume all the neural nets use the unit step threshold function, that is:

$$u(x)=\begin{cases}0 & x<T\\ 1 & x\geq T\end{cases}$$

**Part B: Heads and Tails (26 points)**
Suddenly, Sherlock's friend John runs into the office waving a map. It turns out that a fearsome
mathematician has been using a neural net to predict Dylan's location!

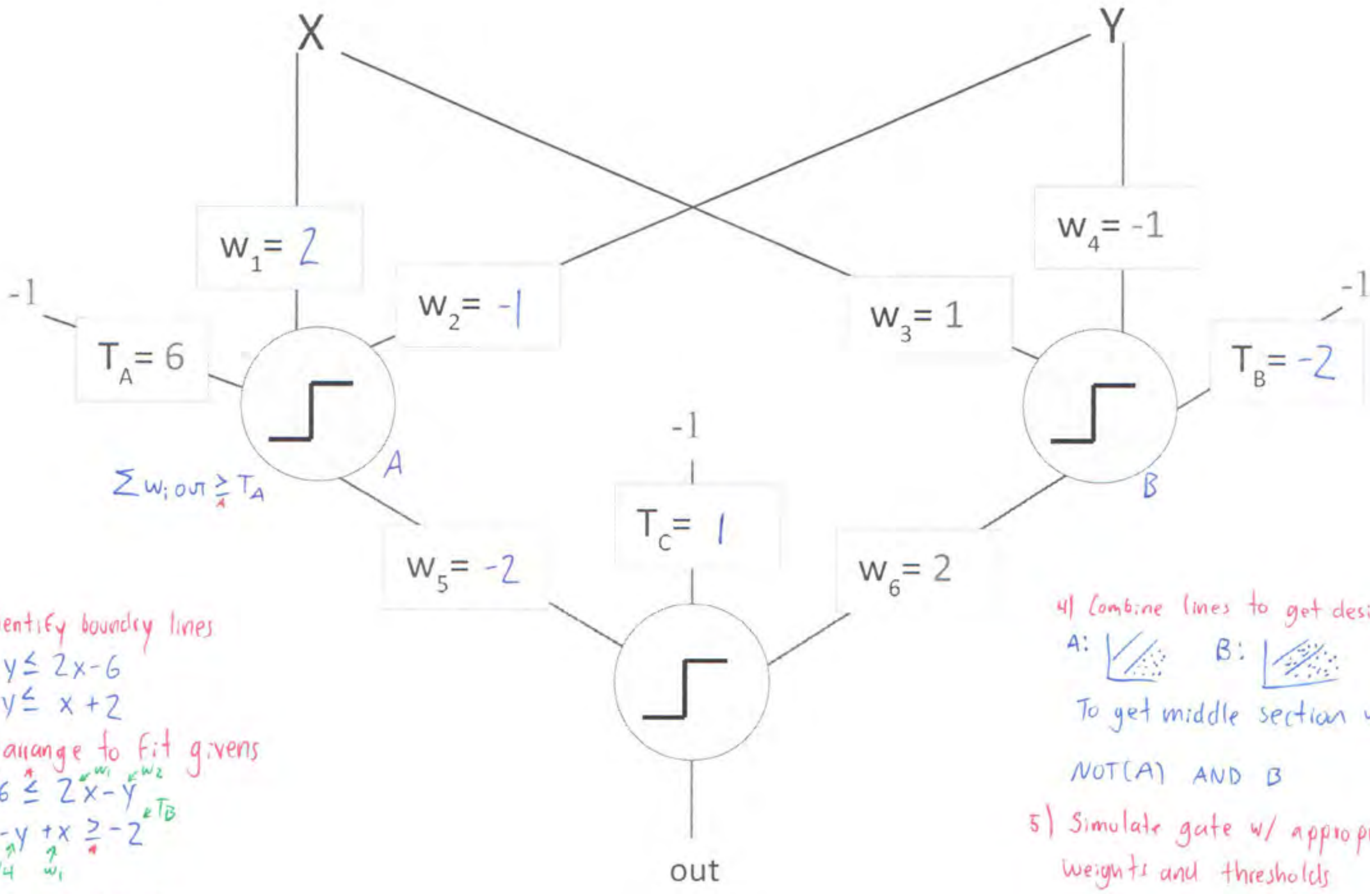
On the right is a campus map of MIT; points marked +
are places that Dylan has visited, and points marked – are places that Dylan has avoided.




Below is the neural net that was trained to predict Dylan's location—it produces an
output of 1 when its inputs lie between the two lines drawn on the map, and an output of 0 otherwise.

**B1. (20 points)** Find weights and thresholds for this neural net which would enable it to produce the described behavior. **Each value should be an integer**, and some of the values have been filled in for you.

**Note:** A copy of these diagrams, along with scratch space, is provided on a tear-off sheet after the last page of the exam.

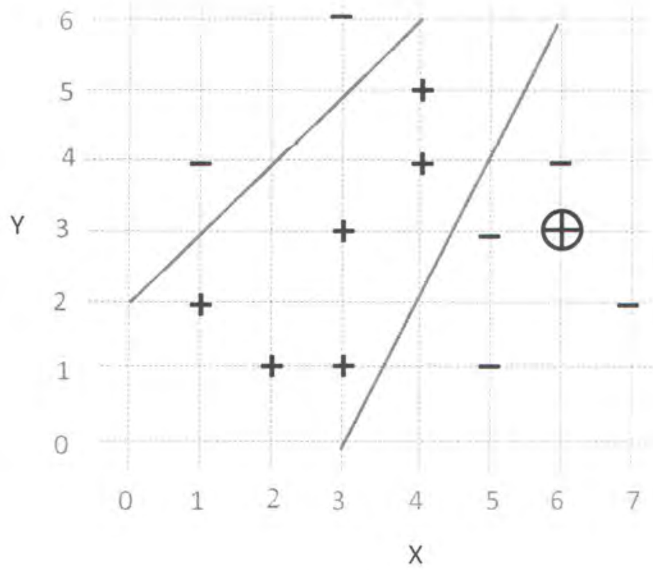



$\sum w_i \, out \geq T_A$

Identify boundry lines
A: $y \leq 2x - 6$
B: $y \leq x + 2$
Rearrange to fit givens
A: $6 \leq 2x - y$
B) $-y + x \geq -2$
Fill in first level

4) Combine lines to get desired output
A: B:
To get middle section we want
NOT(A) AND B
5) Simulate gate w/ appropriate weights and thresholds

**B2. (6 points).** Prof. Winston arrives, appropriately surprised to see two fictional strangers reading a map in his locked office. John explains that he just saw Dylan at the point ⟨6,3⟩ . He wants to adjust the neural net from part B1 to account for this new positive training point – but he knows that neural nets are fragile, so he doesn't want to tamper with it very much. Intrigued, Prof. Winston asks him how he would do it.



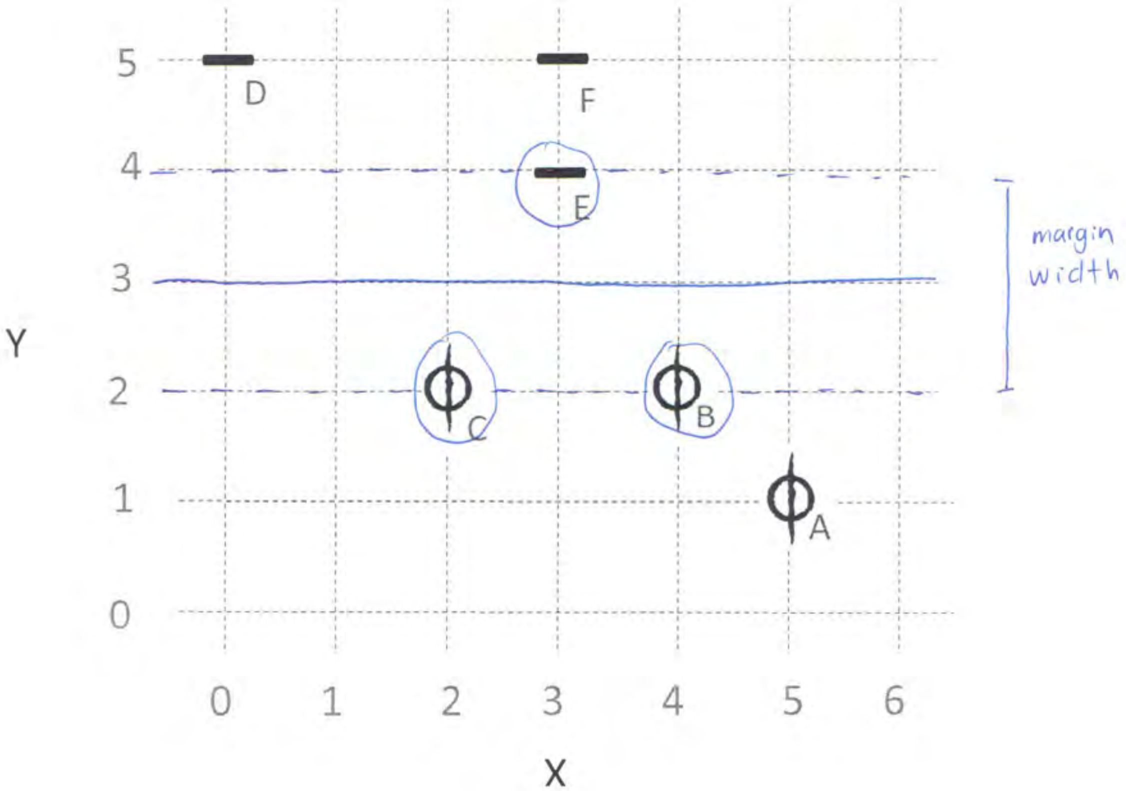

Your task is to change as few of the weights/thresholds as possible so that the neural net from part B1 correctly classifies **all** of the training points—including the new positive point at ⟨6,3⟩ . (You do **not** need to use backward propagation; you can solve by inspection.)

**List** a minimal set of weights and thresholds ($w_1$, $w_2$, $w_3$, $w_4$, $w_5$, $w_6$, $T_A$, $T_B$, $T_C$) you would need to change. If none of the weights/thresholds need to change, write "NO CHANGE". If the task is impossible, write "IMPOSSIBLE" instead.

IMPOSSIBLE. We would need more than 2 lines to classify the new data so simply changing w's and T's won't work.

# Problem 2: Oil vectors support machines (50 points)

The great wizard Urza has just learned that his home is being contaminated by the spread of glistening oil. He has collected several soil samples from the vicinity and marked them as either contaminated (Φ) or normal (–).




## Part A (30 points):

Drawing upon his mastery of support vector machines, Urza wants to construct a linear boundary that separates the contaminated samples from the normal samples by the widest possible margin.

1) Convex Hull method (wrap rubber band)

3 cases


### A1 (10 points).

On the diagram above,

1. Draw the SVM boundary as a <u>solid line.</u>
2. Draw the gutters as <u>dashed lines</u>.
3. **<u>Circle</u>** all of the support vectors.

**A2 (20 pts).** Based on the boundary you drew:

(a) Determine the width of the margin. → distance b/w gutters

2

(b) Solve for the vector $\overline{w}$ and offset $b$. (Recall the positive side of the boundary is defined by $\vec{w} \cdot \vec{x} + b \geq 0$. Contaminated samples ($\Phi$) should be classified as positive; normal samples (–) should be classified as negative).

$\overline{\mathbf{w}} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$ $\quad$ b = 3

(c) Solve for the supportiveness values $\alpha_i$.

$\alpha_A = 0$ $\quad$ $\alpha_D = 0$

$\alpha_B = 1/4$ $\quad$ $\alpha_E = 1/2$

$\alpha_C = 1/4$ $\quad$ $\alpha_F = 0$

Show your work for partial credit:

1) Draw decision boundries

2) Find equation of boundry line

$y = 3$

3) Rewrite equation in form:

$\vec{w} \vec{x} + b = 0$

$\text{class}(\vec{x}) = \begin{cases} + & \text{if } \vec{w}\vec{x} + b \geq 0 \\ - & \text{else} \end{cases}$

↑ pick training point and check sign

$[0, -1] \begin{bmatrix} x \\ y \end{bmatrix} + 3 = 0 \quad \Rightarrow -y + 3 = 0 \quad \Rightarrow y = 3$ ✓

Check: Point 0,0 is supposed to be + ✓

4) Scale equation such that:

$\vec{w} x_i + b = y_i$

$c\left([0, -1]\begin{bmatrix} 2 \\ 2 \end{bmatrix} + 3\right) = 1$

$\Rightarrow c = 1$

Scale $w$ and $b$ appropriatly

$w = [0, -1]$

$b = 3$

5) Find $\alpha$'s

Non support $\alpha$'s $= 0$

support $\alpha$'s $> 0$

$\alpha_A = \alpha_D = \alpha_F = 0$

$\sum_i y_i \alpha_i = 0$

$\vec{w} = \sum y_i \alpha_i x_i$

(1) $\alpha_C + \alpha_B - \alpha_E = 0$

$\begin{bmatrix} 0 \\ -1 \end{bmatrix} = \alpha_C \begin{bmatrix} 2 \\ 2 \end{bmatrix} + \alpha_B \begin{bmatrix} 4 \\ 2 \end{bmatrix} - \alpha_E \begin{bmatrix} 3 \\ 4 \end{bmatrix}$

(2) $0 = 2\alpha_C + 4\alpha_B - 3\alpha_E$

$-1 = 2\alpha_C + 2\alpha_B - 4\alpha_E$

⋮ Math

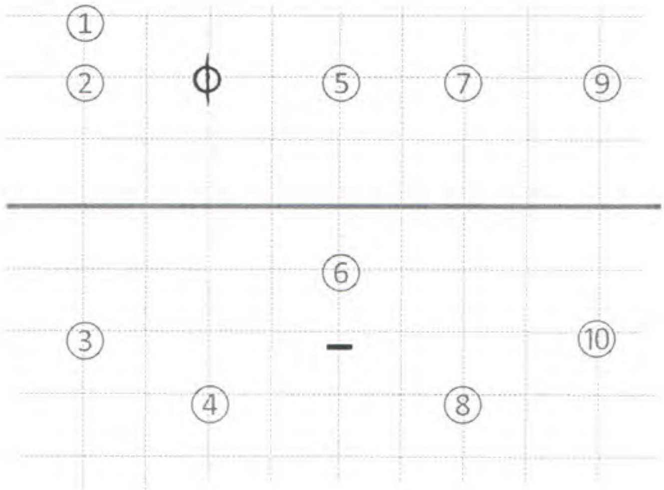
$\alpha_E = 1/2$

$\alpha_B = 1/4$

$\alpha_C = 1/4$

## Part B (20 points):

### B1 (10 points)




The SVM boundary in this diagram was produced by a training set consisting of just three training points. (Some of them, but not necessarily all of them, are support vectors.) Although you know the location of the two training points shown, you've forgotten the location of the third training point —and whether it was contaminated or normal!

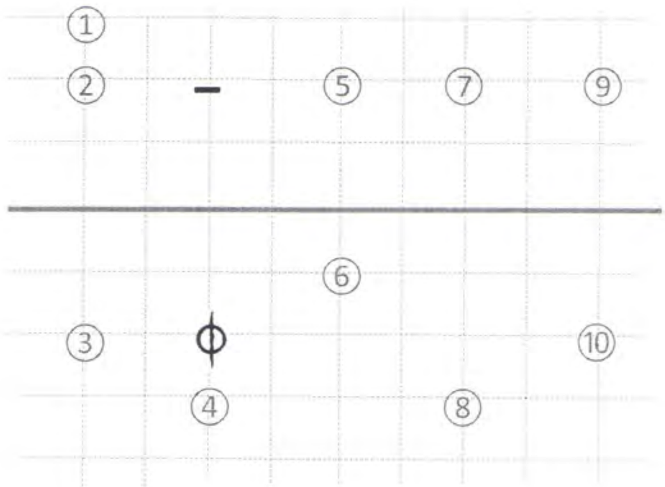
Out of the ten locations marked ①-⑩, where could the third training point have been?

**List** all the locations ①-⑩ where the third training point could have been. If it could not have been in any of those locations, write NONE instead.

3, 5, 7, 9

Think about 3 cases in page 5 and what would happen if you added a point.

**B2 (10 points).** Suppose the first two training points were arranged like this instead: Out of the ten locations marked ①-⑩, where could the third training point have been?




**List** all the locations ①-⑩ where the third training point could have been. If it could not have been in any of those locations, write NONE instead.

1, 2, 3, 4, 5, ,7,8,9,10

# 6.034 Quiz 3
# 14 November 2012

| | |
|---|---|
| Name | WREKA AND I |
| email | |

Circle your TA **(for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

**Dylan Holmes** **Sarah Lehmann** **William Adams**

**Igor Malioutov** **Robert McIntyre** **Ami Patel**

**Sila Sayan** **Mark Seifter** **Stephen Serene**

| Problem number | Maximum | Score | Grader |
|---|---|---|---|
| 1 | 50 | | |
| 2 | 50 | | |
| Total | 100 | | |

| Problem number | Maximum | Score | Grader |
|---|---|---|---|
| 3 | 6 | | |

**We recommend you reserve a few minutes for problem three.**

There are 8 pages in this quiz, including this one, but not including blank pages and tear-off sheets. Tear-off sheets with duplicate drawings and data are located after the final page of the quiz. As always, open book, open notes, open just about everything, including a calculator, but no computers.

# Problem 1: Neural Nets (50 points)

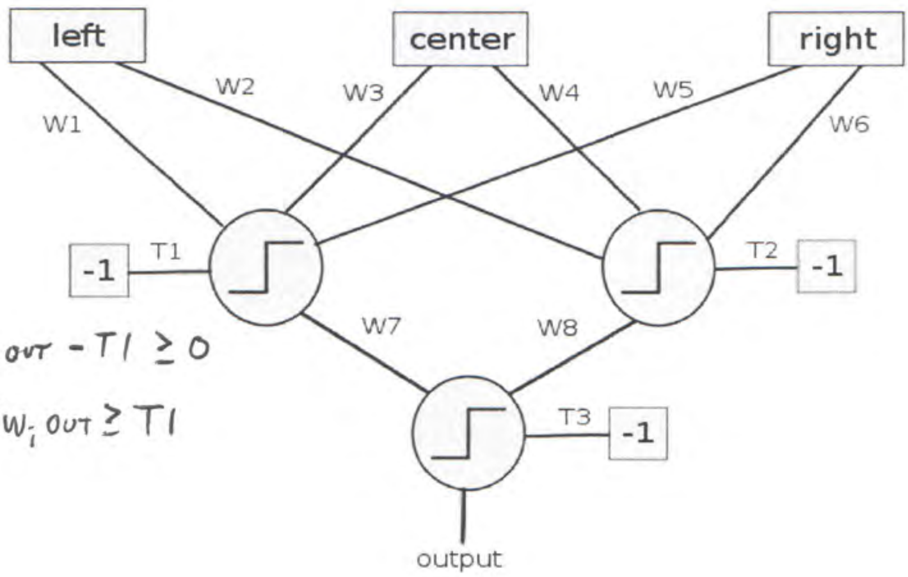
## Part A (10 points)

In order to build a Terminator robot bodyguard, John Conner is designing a neural network that can recognize vertical edges. The inputs to the network consist of three pixels—a left pixel, a center pixel, and a right pixel—each of which can have a value between 0 (black) and 255 (white).

John decides that the **if the values of the left and right inputs differ by more than 100, then the network detects an edge, so the output of the network should be +1. Otherwise, the output of the network should be 0.**

**Assign weights and thresholds** to the network below to implement this behavior.

In the diagram below, the **boxes denote inputs**; all the **threshold inputs have a fixed value of -1**; and all neurons use the **unit step function** $step(x)=\begin{cases}0, & x<0\\ 1, & x\geq 0\end{cases}$, where $x$ is the weighted sum of the neuron's inputs, and *step(x)* is the neuron's output.




$\sum w_i out - T1 \geq 0$

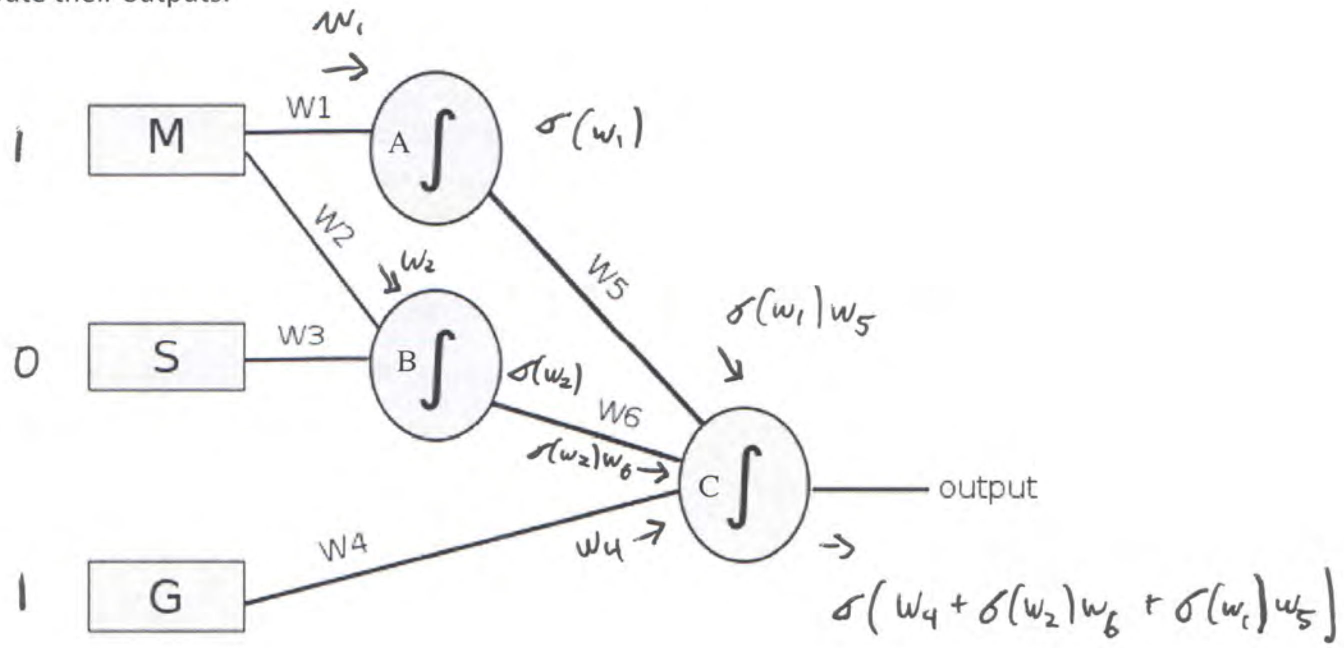
$\sum w_i out \geq T1$

| | | |
|---|---|---|
| T1: 100 | W1: -1 | W5: 1 |
| T2: 100 | W2: 1 | W6: -1 |
| T3: 1 | W3: 0 | W7: 1 |
| | W4: 0 | W8: 1 |

## Part B (40 points)

Now, John Conner has built a neural net which distinguishes human-shaped Terminator robots from real humans based on three input features: whether they **M**isuse colloquial expressions, whether they have **S**uper strength, and whether they are **G**ood with guns. An output value of **< ½ means that the network classifies the input as a human;** an output value of **> ½ means that the network classifies the input as a robot**.

Note: All neurons in this network use the **sigmoid function σ(x) with its threshold at zero** to compute their outputs.




**B1 (15 points)**: John decides to test this network on the character River from the show *Firefly*. River is a **human** (desired output = 0) with the following features:

| Input Feature | Value |
|---|---|
| Misuses colloquial expressions (M) | 1 |
| Super strength (S) | 0 |
| Good with guns (G) | 1 |

Write an expression for the network's output Z for River in terms of the input features, the unknown weights (W1, W2, etc.), and the sigmoid function σ(x).

$$Z = \sigma\left[ w_5 \sigma(w_1) + w_6 \sigma(w_2) + w_4 \right]$$

**B2 (15 points):** Next, John Conner uses back propagation to train the neural net, using River as an example. **Write an expression for W1'**, the updated value of W1, in terms of the input features, the initial weights (W1, W2, etc.), the network output Z, and the training rate R. (Note: **Your expression might not include all of these variables.**)

$W_1$ is involved between $M \to A$ so we want $\Delta W_{M\to A}$

$\Delta W_{M\to A} = r \cdot \overset{1}{OUT_M} \cdot \underbrace{\delta_A}$

$\delta_A = \overset{\sigma(w_1)}{OUT_A}(1-OUT_A)\sum(\overset{w_5}{W_{A\to C}}\,\delta_C)$

$= \sigma(w_1)(1-\sigma(w_1))\, w_5\, \delta_C$

$\delta_C = OUT_C(1-OUT_C)(\overset{0}{OUT^*} - \underset{z}{OUT})$

$= z(1-z)(-z)$

$\Delta W_{M\to A} = W_1' - W_1 = r\,\delta_A \Rightarrow \boxed{W_1' = r\,\delta_A + W_1}$

$\Delta W_{A\to B} = r \cdot OUT_A \cdot \delta_B$

$\delta_B = OUT_B(1-OUT_B)(\overset{\text{desired}}{OUT^*} - \overset{\text{actual}}{OUT})$

if B in final layer

else:

$OUT_B(1-OUT_B)\sum_{\substack{\text{outgoing}\\ C_i}} W_{B\to C_i}\,\delta C_i$

**B3 (5 points):** How would the updated value for W1 compare to the value above if you started with a different initial value for W3 but kept all the other initial weights the same? **(Circle the best answer.)**

SAME DIFFERENT DEPENDS

Explain.

The input S is 0 so $W_3$ does not matter.

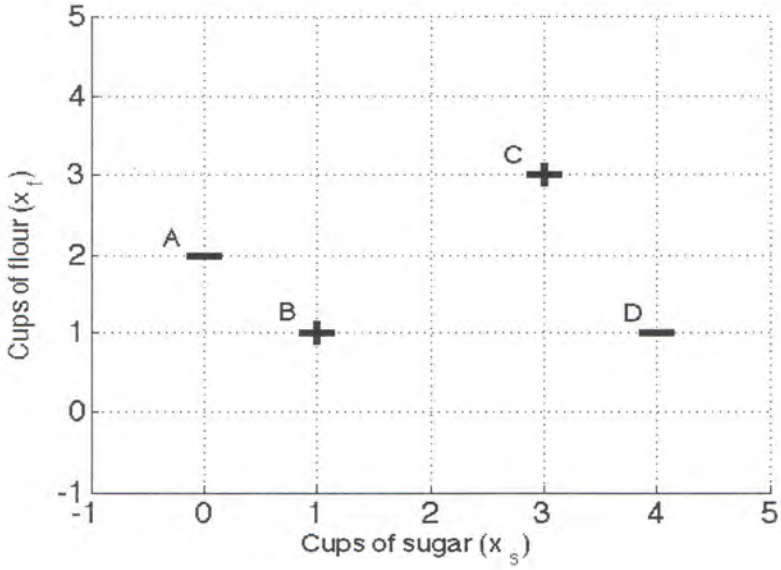
**B4 (5 points):** Now, John Connor performs a series of experiments using all sorts of networks, rates, and initial weights. In all his experiments, he trains his neural net with the same set of Terminator robots and humans (not including River). In some of the experiments, all the training examples are recognized correctly after training, but in those experiments, River is never classified correctly after training. What does this suggest about River?

All networks group River with the terminators so River must have exactly the same features as one of the robots used as a training example.

# Problem 2: Support Vector Machines (50 points)

Recently, you've been promised a lot of cake; unfortunately, many of those promises turn out to be false. Sometimes, you can get your hands on the supposed cakes' recipe beforehand. You decide to use this information to help **distinguish the cakes that are true (+) from the cakes that are lies (-)**.

You draw on prior experience to create a small training dataset for yourself, based on the cakes' sugar ($x_s$) and flour ($x_f$) content (shown below). Then, you apply SVMs to this problem.




**A (10 points)**

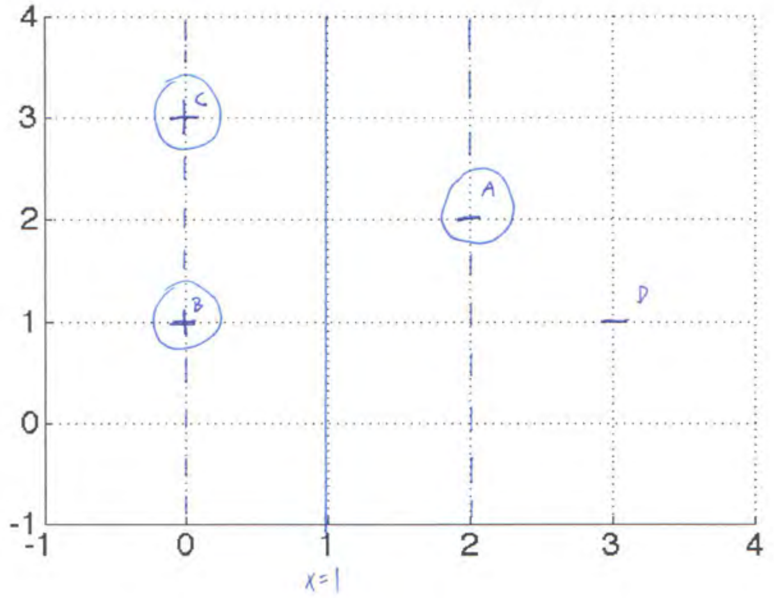
Unfortunately your data is not linearly separable in terms of the features $x_s$ and $x_f$. Atlas suggests using the following transformation on your feature vectors:

$$\varphi(x_s, x_f) = (|x_s - x_f|, x_f)$$

What is the corresponding kernel function K(**u**,**v**) in terms of $u_s$, $u_f$, $v_s$, and $v_f$ where **u** = ($u_s$, $u_f$) and **v** = ($v_s$, $v_f$)?

**B (15 points)**

**Draw** the transformed training points on the following graph. **Label** the transformed training points with the same classification labels (+, -) and letters (A,B,C,D) as the original training points. **Draw** the SVM decision boundary with a heavy solid line and **draw** the gutters with heavy dashed lines. **Circle** the support vectors.




**C (15 points)**

Solve for the values of **w**, b, and the alphas for the training set.

**w** = [-1, 0] $\alpha_A$ = 1/2 $\alpha_C$ = 1/4

b = 1 $\alpha_B$ = 1/4 $\alpha_D$ = 0

↑ b/c not a support vector

Show work for partial credit:

1) Draw decision boundries
2) Find equation of boundry line
$x=1$
3) Rewrite equation in the form:

$\vec{w}\cdot\vec{x}+b=0$

$\text{class}(\vec{x}) = \begin{cases} + & \text{if } \vec{w}\cdot\vec{x}+b \geq 0 \\ - & \text{else} \end{cases}$

$[-1,0]\begin{bmatrix}x\\y\end{bmatrix} + \overset{\text{Find}}{1} = 0$

4) Scale eqn ↑ such that:

$\vec{w}\cdot\vec{x}_i + b = y_i$ ← +1 if ⊕, -1 if ⊖

← support vector

$c\left([-1,0]\begin{bmatrix}0\\1\end{bmatrix}+1\right)=1$

$c=1$

To

5) Find $\alpha$'s:

Non support $\alpha$'s = 0
support $\alpha$'s > 0

$\sum_i y_i \alpha_i = 0$

$\vec{w} = \sum y_i \alpha_i x_i$

$\alpha_C + \alpha_B - \alpha_A = 0$

← point of C

$\begin{bmatrix}-1\\0\end{bmatrix} = \alpha_C\begin{bmatrix}0\\3\end{bmatrix} + \alpha_B\begin{bmatrix}0\\1\end{bmatrix} - \alpha_A\begin{bmatrix}2\\2\end{bmatrix}$

↑ can be split into two equations

$-1 = \alpha_C 0 + \alpha_B 0 - \alpha_A 2$

$0 = \alpha_C 3 + \alpha_B - \alpha_A 2$

Now solve system:

$\alpha_A = 1/2$

$0 = \alpha_C 3 + \alpha_B - 1$

$\alpha_B = -\alpha_C 3 + 1$

$\alpha_C + -\alpha_C 3 + 1 - 1/2 = 0$

$-2\alpha_C + 1/2 = 0$

$\alpha_C = 1/4$

$1/4 + \alpha_B - 1/2 = 0$

$\Rightarrow \alpha_B = 1/4$

**D (5 points)**

How would a recipe with 1⅔ cup granulated sugar and 2 cups all-purpose flour ($x_s = 1.\overline{6}$, $x_f = 2$) be classified by this SVM? **(Circle the best answer)**:

**True (+)** **Lie (-)** **Can't be determined**

Show work for partial credit:

1) Transform points to new coordinates

$(1.\bar{6}, 2) \rightarrow (1/3, 2)$

2) Check where point lies

(+) => True

**E (5 points)**

Someone promises you a cake with no flour ($x_f = 0$). For what values of sugar ($x_s$), will you classify such a cake as true (+)? (Include negative values of $x_s$, if any).

Answer: $-1 < x_s < 1$

Show work for partial credit:

$(x_s, 0) \rightarrow (|x_s|, 0)$

$|x_s| < 1$

$\Rightarrow -1 < x_s < 1$

Fernando Trujano

## Quiz 4, Problem 1 : Adaboost (50 points)

**Instructions:** The following boosting questions are all independent of each other. (They might have different numbers of training points, different classifiers, etc.)

★There are 5 problems. Each one is worth 10pts.★

❶ **(10 pts)** I have fifteen training points (A, B, C, ..., M, N, O) and six classifiers ($h_1$, $h_2$, $h_3$, $h_4$, $h_5$, $h_6$) . The following table shows which points each classifier misclassifies: there's an *X* in a cell if the classifier misclassifies the training point, otherwise the cell is blank. (For example, $h_1$ misclassifies points A, B, C, D, E.) Finally, notice that all the classifiers make overlapping errors.

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $h_1$ | X | X | X | X | X | | | | | | | | | | |
| $h_2$ | X | | | | | X | X | X | X | | | | | | |
| $h_3$ | | X | | | | X | | | | X | X | X | | | |
| $h_4$ | | | X | | | | X | | | X | | | X | X | |
| $h_5$ | | | | X | | | | X | | | X | | X | | X |
| $h_6$ | | | | | X | | | | X | | | X | | X | X |

A) Is it possible to construct a perfect ensemble classifier out of these classifiers ($h_1$, $h_2$, $h_3$, $h_4$, $h_5$, $h_6$) , either by using Adaboost or by picking values of α by hand? (Circle one)

$\boxed{\text{YES}}$ NO

Explain.

Just assign $\alpha = 1$ to each h.

This will work because every point is correctly classified by $4/6$ classifiers.

B) Regardless of your above answer, if you performed five rounds of Adaboost, which weak classifiers would Adaboost choose? (Break ties by picking the weak classifier that comes first in this list: $\mathbf{h_1}$, $\mathbf{h_2}$, $\mathbf{h_3}$, $\mathbf{h_4}$, $\mathbf{h_5}$, $\mathbf{h_6}$).

**Note:** You may solve this problem by inspection.

| | Round 1 | Round 2 | Round 3 | Round 4 | Round 5 |
|---|---|---|---|---|---|
| **Weak Classifier** | $h_1$ | $h_2$ | $h_3$ | $h_4$ | $h_5$ |

❷ **(10 pts)** Lily says "I always write my training weights so that they all have the same denominator (and integer numerators). Imagine my surprise when I started working on my latest pset and found that all of my training weights had the same *odd* denominator!" Megan replies "Surely that's impossible because a subset of the weights must always sum to ½."

It isn't impossible in one particular round; what were the weights of Lily's points?

| $\text{weight}_A$ | $\text{weight}_B$ | $\text{weight}_C$ | $\text{weight}_D$ | $\text{weight}_E$ |
|---|---|---|---|---|
| $1/5$ | $1/5$ | $1/5$ | $1/5$ | $1/5$ |

Next, if just points A and B were misclassified that round, what were the updated weights in the following round?

| $\text{weight}_A$ | $\text{weight}_B$ | $\text{weight}_C$ | $\text{weight}_D$ | $\text{weight}_E$ |
|---|---|---|---|---|
| $1/4$ | $1/4$ | $1/6$ | $1/6$ | $1/6$ |

$w_{new} = \begin{cases} \frac{1}{2} \frac{1}{1-\epsilon} w_{old} & ✓ \\ \frac{1}{2} \frac{1}{\epsilon} w_{old} & ✗ \end{cases}$ $\epsilon = \sum_{wrong} w_i = 2/5$ $w_{new_{A,B}} = \frac{1}{2}\left(\frac{5}{2}\right)\left(\frac{1}{5}\right) = 1/4$ $w_{new_C} = \left(\frac{1}{2}\right)\left(\frac{5}{3}\right)\left(\frac{1}{5}\right) = 1/6$

❸ **(10 pts)** The following weights cannot possibly be right. Why not?

| $\text{weight}_A$ | $\text{weight}_B$ | $\text{weight}_C$ | $\text{weight}_D$ | $\text{weight}_E$ |
|---|---|---|---|---|
| 1/14 | 1/14 | ? | 3/14 | 8/14 |

$\text{weight}_E > 1/2$ which can't happen since:

$$\sum_{correct} w_i = 1/2 = \sum_{wrong} w_i$$

❹ **(10 pts)**. I have an unspecified number of training points. In the first round of boosting, I chose a weak classifier with $0 < \varepsilon < ½$. After updating the weights for the second round, all of my training points had weights (strictly) less than ½.

A) What is the smallest number of training points I could have had? $\boxed{5}$

B) Suppose I had *exactly* that many training points—what voting power (α) was assigned to the classifier I picked in the first round? $\boxed{\frac{1}{2}\ln(3/2)}$

Show your work for partial credit. In particular, be sure to justify your answer to (A) by explaining why your answer works, and why no smaller answer will work

1 and 2 won't yield an $\epsilon < 1/2$

3 - misclassifying 2 on first round will yield $w_i = 1/4 \quad 1/4 \quad 1/2$

4 " " $\epsilon = 1/2$

5 " " $\epsilon = 2/5$ and $w_i = 1/4 \quad 1/4 \quad 1/6 \quad 1/6 \quad 1/6$

$\epsilon = \sum_{wrong} w_i$

$\alpha = \frac{1}{2}\ln\left(\frac{1-\epsilon}{\epsilon}\right)$

Since $\epsilon = 1/2$

$\alpha = \frac{1}{2}\ln(3/2)$

$w_{new} = \begin{cases} \frac{1}{2}\frac{1}{1-\epsilon} w_{old} & ✓ \\ \frac{1}{2}\frac{1}{\epsilon} w_{old} & ✗ \end{cases}$

❺ **(10 pts)** Find the values of the missing weights below, **assuming weight$_A$ is less than or equal to weight$_E$ in each round.**

| Round # | weight$_A$ | weight$_B$ | weight$_C$ | weight$_D$ | weight$_E$ |
|---|---|---|---|---|---|
| 6033 | 1/18 | 2/9 = 4/18 | 3/9 = 6/18 | 5/18 | 2/18 |
| 6034 | 1/12 | 2/12 | 3/12 | 5/12 | 1/12 |

**Hint #1:** Solve for the weights in round 6034 first.

**Hint #2:** Remember the formula for updating the weights. $w_{new} = \begin{cases} \frac{1}{2}\frac{1}{\epsilon} w_{old}, & \textit{if misclassified} \\ \frac{1}{2}\frac{1}{1-\epsilon} w_{old} & \textit{otherwise} \end{cases}$

Show your work for partial credit:

Round 6.034

$w_a = 1/12$ since $w_D + 1/12 = 1/2$ and a subset must $= 1/2$ and $w_D + w_B$ or $w_D + w_C$ would go over

$w_A = 1 - w_B - w_C - w_D - w_E = 1/12$

Round 6033

Find $\epsilon$ that round using 6034. $w_{new} = \frac{1}{2}\frac{1}{\epsilon} w_{old}$ (assume D was ✗) $\Rightarrow \frac{5}{12} = \frac{1}{2}\frac{1}{\epsilon}\frac{5}{18} \Rightarrow \epsilon = 1/3$

Use $\epsilon$ to find $w_{A_{new}} = 1/12 = \left(\frac{1}{2}\right)\left(\frac{1}{\epsilon}\right)(w_{old}) \Rightarrow w_{old_A} = 1/18$

Find $w_D$

## Quiz 4, Problem 2 : Probabilistic Inference (50 points)

### Part A: BAYES nets (20 points)

In the figure below, there are three Bayes nets and some independence statements. For each of the statements below and each Bayes net, circle True if the statement is true for the net, or False if the statement is false for the net.

**Note:** Assume that all entries in all conditional probability tables are distinct; this means that the only independence statements that are true are the ones enforced by the shape of the network.

d-separation

parents

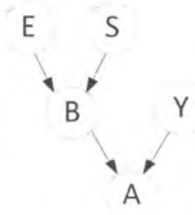
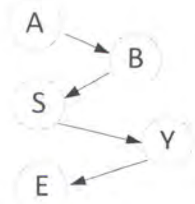
1) Draw ancestral Graphs
vars in equation + parents

2) Moralize graph
connect parents

3) Disorient graph
remove edge arrows

4) Delete givens and edges

PATH => Guaranteed independent
~PATH => Not ⫫

| | Net 1 | Net 2 | Net 3 |
|---|---|---|---|
| |  |  |  |
| E is independent of S | True $\boxed{\text{False}}$ | $\boxed{\text{True}}$ False | True $\boxed{\text{False}}$ |
| $B \perp\!\!\!\perp A \mid Y$ P(B\|Y,A) ≠ P(B\|Y) | $\boxed{\text{True}}$ False | $\boxed{\text{True}}$ False | $\boxed{\text{True}}$ False |
| $A \perp\!\!\!\perp E \mid BY$ Given B and Y, A is independent of E | $\boxed{\text{True}}$ False | $\boxed{\text{True}}$ False | $\boxed{\text{True}}$ False |
| $S \perp\!\!\!\perp B \mid Y$ S is independent of B given Y | $\boxed{\text{True}}$ False | True $\boxed{\text{False}}$ | True $\boxed{\text{False}}$ |
| $B \perp\!\!\!\perp E$ P(B\|E) = P(B) | True $\boxed{\text{False}}$ | True $\boxed{\text{False}}$ | True $\boxed{\text{False}}$ |

$E \perp\!\!\!\perp S$ Y E—S E S S→Y→E

$B \perp\!\!\!\perp A \mid Y$ B—A B—A A—B

$A \perp\!\!\!\perp E \mid BY$ No path A⟷E if remove B Y

$S \perp\!\!\!\perp B \mid Y$ B S S—B B—S

$B \perp\!\!\!\perp E$ Path B⟷E

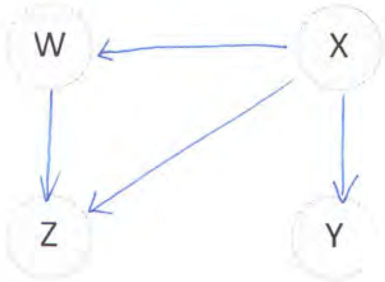
## Part B (5 points)

Your friend tells you that she can compute every joint probability between four variables (W, X, Y, Z) as the following product of conditional probabilities:

$X \to W \quad X \to Y \quad W,X \to Z$

$$P(W, X, Y, Z) = P(X)P(W \mid X)P(Y \mid X)\, P(Z \mid W,X)$$

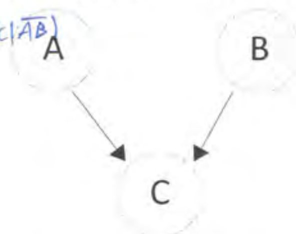
Draw arrows in the Bayes net below to reflect the conditional independence properties described by this equation. (You don't need to draw the conditional probability tables; just the arrows are enough.)




## Part C (25 points)

Compute the following probabilites for the Bayes net shown on the right. Express your answers in terms of fractions, integers, and *x*.

P(A=True) = 1/3 P(B=True) = 1/2




| A | B | P(C=True\|A,B) |
|---|---|---|
| True | True | 1-*x* |
| True | False | *x* |
| False | True | *x* |
| False | False | *x*/2 |

**C1 (5 pts).** The probability that C is true.

$P(C) = P(A)P(B)P(C \mid AB) + P(A)P(\bar{B})P(C \mid A\bar{B}) + P(\bar{A})P(B)P(C \mid \bar{A}B) + P(\bar{A})P(\bar{B})P(C \mid \bar{A}\bar{B})$

$(\tfrac{1}{3})(\tfrac{1}{2})(1-x) + (\tfrac{1}{3})(\tfrac{1}{2})(x) + (\tfrac{2}{3})(\tfrac{1}{2})(x) + (\tfrac{2}{3})(\tfrac{1}{2})(\tfrac{x}{2})$

$\tfrac{1}{6} - \cancel{\tfrac{x}{6}} + \cancel{\tfrac{x}{6}} + \tfrac{2x}{6} + \tfrac{x}{6} = \tfrac{3x}{6} + \tfrac{1}{6} = \tfrac{3x+1}{6}$

**C2 (5 pts).** The probability that B is true, given that C is true.

$P(B \mid C) = \dfrac{P(BC)}{P(C)}$

$\tfrac{3x+1}{6}$ from above

$P(BC) = P(ABC) + P(\bar{A}BC)$
$= P(A)P(B)P(C \mid AB) + P(\bar{A})P(B)P(C \mid \bar{A}B)$
$(\tfrac{1}{3})(\tfrac{1}{2})(1-x) + (\tfrac{2}{3})(\tfrac{1}{2})(x)$
$= \tfrac{1}{6} + \tfrac{x}{6}$

$\Rightarrow P(B \mid C) = \dfrac{x+1}{3x+1}$

**C3 (10 pts).** The probability that B is true, given that C is true and A is false.

$(\tfrac{2}{3})(\tfrac{1}{3})(x)$

$P(B \mid C\bar{A}) = \dfrac{P(C\bar{A}B)}{P(\bar{A}C)} = \dfrac{P(\bar{A}BC)}{P(\bar{A}BC) + P(\bar{A}\bar{B}C)} = \tfrac{2}{3}$

$(\tfrac{2}{3})(\tfrac{1}{2})(\tfrac{x}{2})$

**C4 (5 pts).** Consider the inequality $P(B|C) < P(B|C\overline{A})$. Find a value of *x* which makes this inequality true and which satisfies the requirements of the Bayes net on the previous page. If it's impossible, write IMPOSSIBLE instead.

$x = 1/2$

Show your work.

From last part

$$\frac{x+1}{3x+1} < \frac{2}{3}$$

$$3x + 3 < 6x + 2$$

$$1 < 3x$$

$$x > 1/3$$

# 6.034 Quiz 4
# 6 December 2013

| | |
|---|---|
| Name | Fernando Trujano |
| email | trujano@mit.edu |

Circle your TA **(for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

| | | |
|---|---|---|
| **Michael Fleder** | **Giuliano Giacaglia** | **Dylan Holmes** |
| **Casey McNamara** | **Robert McIntyre** | **Duncan Townsend** |
| **Mark Seifter** | **Sam Sinai** | **Prashan Wanigasekara** |

| Problem number | Maximum | Score | Grader |
|---|---|---|---|
| 1 | 50 | | |
| 2 | 50 | | |
| Total | 100 | | |

| | | | |
|---|---|---|---|
| SRN | 6 | | |

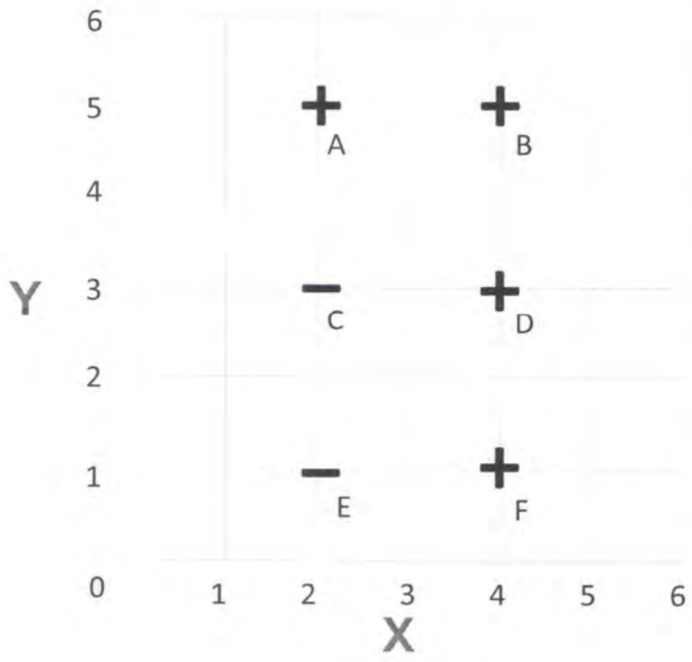
There are 10 pages in this quiz, including this one, but not including blank pages and tear-off sheets. As always, open book, open notes, open just about everything, including a calculator, but no computers.

# Problem 1: Adaboost (50 points)

After attending a spectacular time traveler's convention, fearsome mathematician Ada Lovelace has hijacked a time machine, determined to investigate important developments in artificial intelligence.

## Part A (50 points)

After attending an Adaboost lecture in 1995, Lovelace has rushed home to the 19th century to solve the following homework problem:




| Classifier | Misclassified training points |
|---|---|
| $X \geq 3$ | A |
| $X \geq 5$ | A B D F |
| $Y \geq 4$ | D F |
| $Y \geq 6$ | A B D F |

Reminder: The classifier $\mathbf{X \geq T}$ classifies a point as positive if it's to the *right* of a certain vertical line; otherwise, it classifies the point as negative.

Similarly, the classifier $\mathbf{Y \geq T}$ classifies a point as positive if it's *above* a certain horizontal line; otherwise, it classifies the point as negative.

Feel free to draw these classifiers on the diagram.

**A1 (4 points).** Fill in the table of "Misclassified training points" above. For each weak classifier, list **all** the training points (A,B,C,D,E,F) which it misclassifies. If the classifier doesn't misclassify any points, write NONE instead.

**A2 (24 points).** Perform three rounds of boosting using the above training data and weak classifiers. In each round, choose the weak classifier with the lowest error rate. In case of a tie, choose the weak classifier that comes first in this list: **X≥3, X≥5, Y≥4, Y≥6**. You must use only these classifiers.

In any round, if boosting would terminate instead of choosing a classifier, write "NONE" for the weak classifier (h) and for the voting power (α). Then, leave all remaining spaces blank.

| | Round #1 | Round #2 | Round #3 |
|---|---|---|---|
| **weight$_A$** | 1/6 Initialize to 1/N | $\left(\frac{1}{2}\right)(6)\left(\frac{1}{6}\right) = 5/10$ | 5/16 |
| **weight$_B$** | 1/6 | $\left(\frac{1}{2}\right)\left(\frac{6}{5}\right)\left(\frac{1}{6}\right) = 1/10$ | 1/16 |
| **weight$_C$** | 1/6 | 1/10 | 1/16 |
| **weight$_D$** | 1/6 | 1/10 | 4/16 |
| **weight$_E$** | 1/6 | 1/10 | 1/16 |
| **weight$_F$** | 1/6 | 1/10 | 4/16 |
| **Error rate (ε) of X ≥ 3** | 1/6 | 5/10 | 5/16 |
| **Error rate (ε) of X ≥ 5** | 4/6 | 8/10 | 14/16 |
| **Error rate(ε) of Y ≥ 4** | 2/6 | 2/10 | 8/16 |
| **Error rate (ε) of Y ≥ 6** | 4/6 | 8/10 | 14/16 |
| **Weak classifier chosen this round (h)** | $X \geq 3$ ← smallest $\epsilon$ | $Y \geq 4$ | $X \geq 3$ |
| **Voting power (α)** | $\frac{1}{2}\ln\left(\frac{5/6}{1/6}\right)$ $\frac{1}{2}\ln(5)$ | $\frac{1}{2}\ln(4)$ | $\frac{1}{2}\ln\left(\frac{11/16}{5/16}\right) = \frac{1}{2}\ln(11/5)$ |

You may show your work below for partial credit:

$$\epsilon = \sum_{\text{wrong } i} w_i$$

$$\alpha = \frac{1}{2}\ln\left(\frac{1-\epsilon}{\epsilon}\right)$$

$$w_{new} = \begin{cases} \frac{1}{2}\cdot\frac{1}{1-\epsilon} w_{old} & \text{if } ✓ \\ \frac{1}{2}\cdot\frac{1}{\epsilon} w_{old} & \text{if } ✗ \end{cases}$$

Adaboost:

1) Initialize $w_i = 1/N$
2) Calculate $\epsilon$'s
3) Pick $h_i$ with smallest $\epsilon$
4) Calculate $\alpha_i$
5) Add $\alpha_i h_i(x)$ to $H$
6) Adjust $w_i$ for next round
7) done?

Done when: (either)

1) H is good enough - (classifies enough training pts correctly)
2) max # rounds completed
3) best classifier $\epsilon = 1/2$

$H = h(x \geq 3)\frac{1}{2}\ln(5) + h(y \geq 4)\frac{1}{2}\ln(4) + h(x \geq 3)\frac{1}{2}\ln(11/5)$

↑either +1 or −1

**A3 (6 points).** How would the ensemble classifier H produced after three rounds of boosting classify the following points? (Circle the best answer in each case.)

| | | | |
|---|---|---|---|
| (X=0, Y=0) | POSITIVE | NEGATIVE | CAN'T BE DETERMINED |
| (X=0, Y=7) | POSITIVE | NEGATIVE | CAN'T BE DETERMINED |
| (X=7, Y=7) | POSITIVE | NEGATIVE | CAN'T BE DETERMINED |

**A4 (6 points).** Suppose Lovelace decides to continue boosting until she completes a total of 2013 rounds. What classifier will she pick in the 2013th round?

(**Check the box for the correct answer, then fill in its blank.**)

☐ **Outcome A**: Adaboost must terminate at or before the 2013th round. In fact, it will terminate instead of picking a classifier in Round ______.

☒ **Outcome B:** Adaboost will reach the 2013th round without terminating. In the 2013th round, it will pick the weak classifier $X \geq 3$.

odd round: $x \geq 3$
even round: $y \geq 4$

**A5 (4 points).** While keeping in touch with a brilliant yet harried cryptographer in the 20th century, Lovelace learns that there was a weak classifier missing from the original set. Apparently, the missing classifier misclassifies exactly three of the training points, and if you add it to the original list of classifiers, you can use boosting to classify the training data perfectly. (Note: The missing classifier could be any kind of classifier, not just a line classifier like **X ≥ T** or **Y ≥ T**)

Immediately, she deduces which three points it must have misclassified. Circle the **three** training points it must have misclassified.

**A** **B** **C** **D** **E** **F**

Since they have the lowest weights in round 3 => lowest $\epsilon$

**A6 (6 points).** The cryptographer is quite surprised by these results. He remarks that since the missing classifier misclassifies three out of the six training points, it has an error rate of 50% and hence should be no better than a coin toss—it's useless as a weak classifier.

Explain what he means. (for example, briefly summarize what happens to the weights and to the voting power α if you pick a classifier with error rate of ½ during a round of boosting):

If $\epsilon = 1/2$ the weights will not change after updating and $\alpha = \ln(1) = 0$ so the classifier would have no effect
↳ that's why we terminate if the best classifier $\epsilon = 1/2$

Based on the argument you just made, the cryptographer believes that Adaboost would never pick the missing classifier. Concisely explain why the Adaboost algorithm would, in fact, pick it.

Weights change every round => $\epsilon$'s change too.

## Part B (bonus)

**This is a bonus problem. Your score will be unaffected if you skip it or answer incorrectly. You can get +2 extra credit points if you answer correctly.**

Having successfully solved a variety of boosting problems, Lovelace decides to supercharge her knowledge by kidnapping and interrogating a 21st century teaching assistant to check her work. (Needless to say, locked office doors are not a problem for time machines.)

Immediately, the TA points out that the following weights cannot possibly be correct, no matter what problem they're from. Why not?

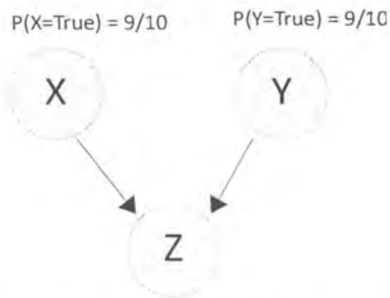
| | **$\text{weight}_P$** | **$\text{weight}_Q$** | **$\text{weight}_R$** | **$\text{weight}_S$** | **$\text{weight}_T$** | **$\text{weight}_U$** |
|---|---|---|---|---|---|---|
| **Round ###** | 1/21 | 2/21 | 3/21 | 4/21 | 5/21 | 6/21 |

No subset of the weights add up to ½ and

$\sum_{\text{misclassified}} w_i = \frac{1}{2} = \sum_{\text{correct}} w_i$

# Problem 2: Bayesian Inference

## Part A (5 points)




| X | Y | P(Z=True\|X,Y) |
|---|---|---|
| True | True | 8/10 |
| True | False | 2/10 |
| False | True | 2/10 |
| False | False | 1/10 |

This Bayes Net depicts independence relations among three boolean variables X, Y, and Z. Consider the following probabilities:

- **(A)** P(X = True | Y = True, Z = True)
- **(B)** P(X = True | Y = False, Z = True)
- **(C)** P(X = True | Z = True)

Your task is to arrange these probabilities from smallest to largest (<u>Hint</u>: this problem does not require significant calculation.) **<u>Circle the best answer.</u>**

| | | |
|---|---|---|
| A < B < C | B < A < C | C < A < B |
| A < C < B | $\boxed{B < C < A}$ | C < B < A |

## Part B (45 points)

The *New York Times* has just acquired a batch of leaked NSA documents, and they have enlisted you to determine the security classification of each document: Secret (**S**), Restricted (**R**), or Lunch receipt (**L**).

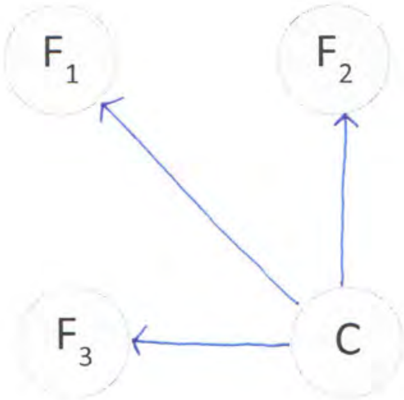
Based on your 6.034 expertise, you decide to build a Naïve Bayes classifier to do the job. First, you collect 600 previously leaked documents that are already marked as Secret (S), Restricted (R), or Lunch receipt (L). You notice several reoccurring keywords that could be relevant when determining classification:

| Keyword feature | # of **Secret** documents containing keyword | # of **Restricted** documents containing keyword | # of **Lunch receipt** documents containing keyword |
|---|---|---|---|
| $F_1$ "urgent" | 70 | 50 | 10 |
| $F_2$ "spaghetti" | 1 | 2 | 50 |
| $F_3$ "surveillance" | 20 | 25 | 1 |

And the total number of documents with each security classification is:

| Total # of **Secret** documents | Total # of **Restricted** documents | Total # of **Lunch receipt** documents |
|---|---|---|
| 100 | 200 | 300 |

**B1** **(4 points):** The unfinished Bayes net below has a node for each keyword feature $F_1, F_2, F_3$ and a node for the security classification C in {S,R,L}. Draw arrows between the nodes such that the arrows express the independence relations assumed when using a Naive Bayes classifier.




Given classification, features conditionally independent of each other.

**B2** **(12 points):** Based on the data in the tables above, compute estimates for the following probabilities (You may leave your answers in terms of fractions):

given

| | | | | | |
|---|---|---|---|---|---|
| P(C = S) | $100/600$ | P(C = R) | $200/600$ | P(C = L) | $300/600$ |
| $P(F_1 \mid C = S)$ | $70/100$ | $P(F_1 \mid C = R)$ | $50/200$ | $P(F_1 \mid C = L)$ | $10/300$ |
| $P(F_2 \mid C = S)$ | $1/100$ | $P(F_2 \mid C = R)$ | $2/200$ | $P(F_2 \mid C = L)$ | $50/300$ |
| $P(F_3 \mid C = S)$ | $20/100$ | $P(F_3 \mid C = R)$ | $25/200$ | $P(F_3 \mid C = L)$ | $1/300$ |

Show your work for partial credit:

From chart

$$P(F_1 \mid C=S) = \frac{P(F_1, C=S)}{P(C=S)} = \frac{70/\cancel{600}}{100/\cancel{600}} = 70/100$$

⋮

**B3** **(24 points):** Now that you've collected your training data, you decide to look at the first unlabelled document from the NSA, which says:

> "**Urgent** notice: operation **spaghetti** is under **surveillance** by the 6.034 staff."
>
> (In other words, this document has $F_1$=true, $F_2$=true, $F_3$=true.)

In order to decide how to classify this document, let the variable $Q$ denote the prior probability that a document contains all three keywords:

$$Q = P(F_1\text{=true}, F_2\text{=true}, F_3\text{=true}).$$

Compute the following conditional probabilities. Show your work for partial credit, and express your answers solely in terms of numbers/ratios and the variable $Q$. (You don't have to compute a numerical value for $Q$).

ayes Law

1. $P(C = S \mid F_1 = \text{true}, F_2 = \text{true}, F_3 = \text{true}) =$ $\boxed{\left(\frac{70}{100}\right)\left(\frac{1}{100}\right)\left(\frac{20}{100}\right)\left(\frac{100}{600}\right)\left(\frac{1}{Q}\right)}$

$$= \frac{P(F_1F_2F_3 \mid C=S)\,P(C=S)}{P(F_1F_2F_3)} = \frac{\left(\frac{70}{100}\right)\left(\frac{1}{100}\right)\left(\frac{20}{100}\right)\left(\frac{100}{600}\right)}{Q}$$

$$P(F_1F_2F_3 \mid C=S) = P(F_1 \mid C=S)\,P(F_2 \mid C=S)\,P(F_3 \mid C=S) = \left(\frac{70}{100}\right)\left(\frac{1}{100}\right)\left(\frac{20}{100}\right)$$

$$P(F_1 \mid C=S) = \frac{P(F_1, C=S)}{P(C=S)}$$

2. $P(C = R \mid F_1 = \text{true}, F_2 = \text{true}, F_3 = \text{true}) =$ $\boxed{\left(\frac{50}{200}\right)\left(\frac{2}{200}\right)\left(\frac{25}{200}\right)\left(\frac{200}{600}\right)\left(\frac{1}{Q}\right)}$

3. $P(C = L \mid F_1 = \text{true}, F_2 = \text{true}, F_3 = \text{true}) =$ $\boxed{\left(\frac{10}{300}\right)\left(\frac{50}{300}\right)\left(\frac{1}{300}\right)\left(\frac{300}{600}\right)\left(\frac{1}{Q}\right)}$

**B4 (5 points).** Based on your answers above, what security classification does your Naïve Bayes classifier assign to your document "Urgent notice: operation spaghetti is under surveillance by the 6.034 staff."? (**Circle one**.)

| | | | |
|---|---|---|---|
| Secret | Restricted | Lunch receipt | The classification cannot be determined with the given data. |

Because $P(C=S \mid F_1 F_2 F_3)$ is the largest.

↩ Note: The back of this page contains Problem 3, Spiritual and Right Now

# 6.034 Quiz 4

## 3 December 2014

| | |
|---|---|
| Name | Fernando Trujano |
| Email | |

Circle your TA **(for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

| | | |
|---|---|---|
| **Josh Blum** | **Elisa Castañer** | **Pedro Cattori** |
| **Martin Gale** | **Malcom Gilbert** | **Dylan Holmes** |
| **Jessica Noss** | **Duncan Townsend** | **Siyao Xu** |

| Problem number | Maximum | Score | Grader |
|---|---|---|---|
| 1 | 50 | | |
| 2 | 50 | | |
| Total | 100 | | |

| | | | |
|---|---|---|---|
| SRN | 6 | | |

There are 9 pages in this quiz, including this one. As always, open book, open notes, open just about everything, including a calculator, but no computers.

# Problem 1: Adaboost (50 points)

## Part A: Two rounds of boosting (32 points)

### A1 (22 points)

You have six training points (A, B, C, D, E, F) and five classifiers ($h_1$, $h_2$, $h_3$, $h_4$, $h_5$) which make the following misclassifications:

| Classifier | Misclassified training points (A, B, C, D, E, F) | | | | | |
|---|---|---|---|---|---|---|
| $h_1$ | A | | | D | | F |
| $h_2$ | | | | D | | |
| $h_3$ | | B | C | | | |
| $h_4$ | A | B | | | | F |
| $h_5$ | | B | C | D | | |

Perform two rounds of boosting with these classifiers and training data. In each round, pick the classifier with the **lowest error rate**. Break ties by picking the classifier that comes first in this list: $h_1$, $h_2$, $h_3$, $h_4$, $h_5$. Space for scratch work is provided on the following page.

| | Round 1 | Round 2 |
|---|---|---|
| $weight_A$ | 1/6 | |
| $weight_B$ | | |
| $weight_C$ | | |
| $weight_D$ | | |
| $weight_E$ | | |
| $weight_F$ | | |
| Error rate of $h_1$ | | |
| Error rate of $h_2$ | | |
| Error rate of $h_3$ | | |
| Error rate of $h_4$ | | |
| Error rate of $h_5$ | | |
| weak classifier (h) | | |
| classifier error ($\varepsilon$) | | |
| voting power ($\alpha$) | | |

Space provided for scratch work

## A2 (6 points)

Three of the training points (B, D, F) have been selected below. For each one, decide whether the ensemble classifier $H(\vec{x})$ produced after two rounds of boosting misclassifies or correctly classifies that point. Circle the best answer in each case. If the answer can't be determined from the available information, circle "Can't tell" instead.

| Training point | Classification by ensemble classifier | | |
|---|---|---|---|
| B | Correctly classified | Misclassified | Can't tell |
| D | Correctly classified | Misclassified | Can't tell |
| F | Correctly classified | Misclassified | Can't tell |

## A3 (4 points)

Suppose you continue the Adaboost procedure from Part A for a **total** of 2014 rounds. (You may assume it doesn't terminate before then.) If you always pick the classifier with the **lowest** error rate, which training data point will have the **smallest** weight at the end of the 2014$^{th}$ round? (Circle one.)

**A** **B** **C** **D** **E** **F**

never misclassified

## Part B (18 points)

This section consists of questions about Adaboost *in general*—they do not rely on the preceding section. Decide whether each of the statements below is true or false. Circle the one best answer in each case.

True $\boxed{\text{False}}$ 1. Adaboost accounts for outliers by lowering the weights of training points that are repeatedly misclassified.

Increase weights of misclassified

True $\boxed{\text{False}}$ 2. When you update weights, the training point with the smallest weight in the previous round will always increase in weight.

Not if point w/ smallest weight is classified correctly

$\boxed{\text{True}}$ False 3. Four weak classifiers that make disjoint errors can always be used to construct a perfect classifier by picking $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = 1$.

True $\boxed{\text{False}}$ 4. In each round, any weak classifier with an error rate less than ½ must misclassify fewer than half of the training points.

Could misclassify lots of points w/ small weights

True $\boxed{\text{False}}$ 5. After two rounds of boosting, the ensemble classifier $H(\vec{x})$ always behaves like the conjunction (AND) of two weak classifiers.

It's a vote w/ weights/voting powers.

$\boxed{\text{True}}$ False 6. If you have 100 training points, the updated weight of a training point can never be greater than ¾.

Updated weight cannot be greater 1/2

True $\boxed{\text{False}}$ 7. A weak classifier will never have an error rate greater than ½.

$\epsilon = 1$ if it misclassifies everything

True $\boxed{\text{False}}$ 8. Once a weak classifier is picked in a particular round, it will never be chosen in any subsequent round.

$\boxed{\text{True}}$ False 9. Weak classifiers with error rates (ε) close to 100% are assigned negative voting powers (α).

$$\alpha = \frac{1}{2} \ln\left(\frac{1-\epsilon}{\epsilon}\right)$$

## Problem 2: Bayesian Inference (50 points)

★ Note: This problem consists of four independent subproblems. ★

### Part A: Bayes rule and model selection (10 points)

Suppose you have three coins in a bag: the first coin is fair, with HEADS on one side and TAILS on the other. The second has HEADS on both sides. The third has TAILS on both sides.

Your friend takes a coin out of the bag at random and flips it. When it lands, you observe that the side of the coin facing up is HEADS. What is the probability that this is the fair coin?

| $\frac{2}{3}$ | $\frac{1}{2}$ | $\boxed{\frac{1}{3}}$ | $\frac{1}{4}$ |
|---|---|---|---|
| (A) | (B) | (C) | (D) |

**Check your intuition:** You should treat each coin as a different model $M_1, M_2, M_3$ for explaining the evidence $E$ of having HEADS on one side. Thus, the probability that the coin is fair is $P(M_1|E)$, which you can compute using Bayes' Law.

*Space for scratch work.*

$$P(C_1 \mid H) = \frac{\overset{1/2}{P(H \mid C_1)}\ \overset{1/3}{P(C_1)}}{\underset{\text{3 Heads, 3 Tails in bag} \Rightarrow 1/2}{P(H)}} = 1/3$$

1/3 equal chance of picking coin

3 Heads
3 Tails in bag => 1/2

## Part B: Sensitivity and specificity (20 points)

Senioritis is a rare but treatable condition that occurs in about 20 out of every 10,000 individuals in the general population. A recently developed test for senioritis—the PUNT scan —is 99.5% sensitive and 80% specific. (This means that 995 out of 1000 people with senioritis correctly test positive, and 800 out of every 1000 people without senioritis correctly test negative. The PUNT scan always reports either "positive" or "negative".)

For notation, we can let $D$ be the variable "You have senioritis" and let $T$ be the variable "You test positive for senioritis". Then the information above is:

| | |
|---|---|
| $P(D)$ | 20 out of 10,000 |
| $P(T \mid D)$ | 995 out of 1,000 |
| $P(T \mid D)$ | 800 out of 1,000 |

**Part B1 (10 points)** What is the probability of obtaining a positive test result, regardless of whether you have senioritis?

The marginal probability of a positive test result is *approximately* (circle one):

0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%

*You must show your work to receive credit. Write down the equations you intend to solve, if any, and indicate what values you're plugging in. You probably won't need a calculator, since you only need an approximate final answer.*

$$P(T) = \underset{\frac{995}{1000}}{P(T \mid D)}\ \underset{\frac{20}{10000}}{P(D)} + \underset{\frac{200}{1000}}{P(T \mid \bar{D})}\ \underset{\frac{9080}{1000}}{P(\bar{D})}$$

$$\approx 20\%$$

**Part B2 (10 points)** Suppose your PUNT scan returns a positive result. In this case, the probability that you indeed have senioritis is *most nearly* (circle one):

0% 10% 20% 30% 40% 50% 60% 70% 80% 90% 100%

*<u>You must show your work to receive credit.</u> Write down the equations you intend to solve, if any, and indicate what values you're plugging in. You probably won't need a calculator, since you only need an approximate final answer.*

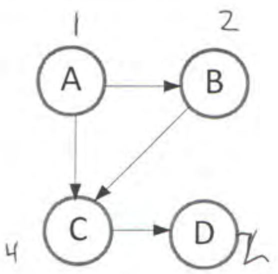
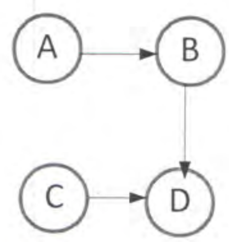
$$P(D \mid T) = \frac{\overbrace{P(T \mid D)}^{995/1000}\ \overbrace{P(D)}^{20/10000}}{\underbrace{P(T)}_{20\%}}$$

$\approx 0\%$

## Part C: Applying d-separation (16 points)

In the figure below, there are two Bayes nets and some independence statements. For each of the statements below and each Bayes net, circle TRUE if the statement is true for the net, and FALSE if the statement is false for the net.

**Note:** Assume that the only independence statements that are true are the ones enforced by the shape of the network.

| |  |  |
|---|---|---|
| A is independent of C. | TRUE FALSE | TRUE FALSE |
| Given C, A is independent of D. | TRUE FALSE | TRUE FALSE |
| P(B\|DAC) = P(B\|AC) | TRUE FALSE | TRUE FALSE |

$B \perp\!\!\!\perp D \mid\mid AC$

Assuming all of the variables are boolean, how many parameters does each Bayes net have? (The number of parameters is the total number of entries in all probability tables.)

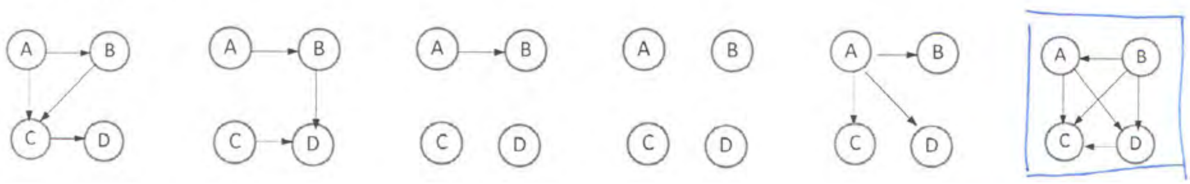
| | | |
|---|---|---|
| # of parameters | 9 | 8 |

# params for a node $= 2^{\# \text{parents}}$

## Part D: Make no assumptions (4 points)

In exactly **one** of the Bayes nets below, **none of the variables are independent**. Circle it.

**Hint:** A Bayes net that makes no independence assumptions is a model that is consistent with every joint probability table. You may find it useful to consider how many parameters it must have as a result.



## Quiz 1, Problem 1: Rule-based systems (50 points)

### Part A: General questions (19 points)

**A1 (15 points)** For each of the following statements, circle the **single best answer**.

1. **True** / $\boxed{\textbf{False}}$: During backward chaining, at most one rule can match the current hypothesis.
Many can match though at most 1 will fire

2. **True** / $\boxed{\textbf{False}}$: During forward chaining, at most one rule can match per round.

3. **True** / $\boxed{\textbf{False}}$: During forward chaining, a rule may match in multiple rounds, but each rule can fire *at most* once. Rule can fire in different rounds

4. **True** / $\boxed{\textbf{False}}$: In each round of forward chaining, the first rule that matches will fire.
Only fires if it would generate new assertion

5. **True** / $\boxed{\textbf{False}}$: In each round of forward chaining, at most one new assertion can be added to the list of assertions. Assertions. Then might have more than one

**A2 (4 points)** In general, which of the following conditions can cause backward chaining to short circuit? That is, which of these will cause the evaluation of subtrees to stop? (Circle **ALL** answers that apply)

A) A child of an AND subtree returns true.

B) A child of an AND subtree returns false.

C) A child of an OR subtree returns true.

D) A child of an OR subtree returns false

E) The backward chainer cannot find a match for the hypothesis in the list of assertions.

F) The backward chainer cannot find a match for the hypothesis in the antecedent of any rule.

## Part B: Backward chaining (8 points)

**Rules:**

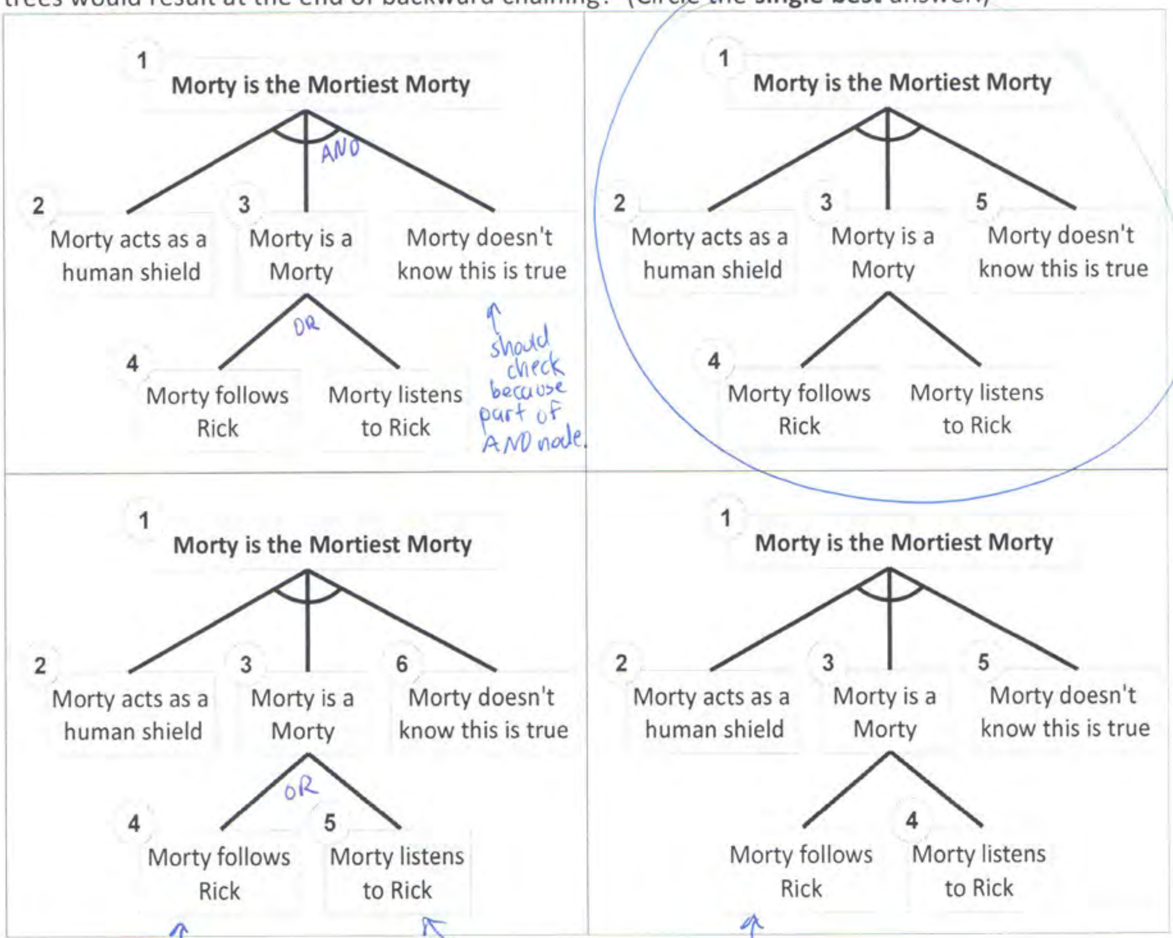
```
R0    IF   (OR '(?x) follows Rick',
               '(?x) listens to Rick'),
      THEN '(?x) is a Morty'

R1    IF   (AND '(?y) acts as a human shield',
                '(?y) is a Morty',
                '(?y) doesn't know this is true'),
      THEN '(?y) is the Mortiest Morty'
```

**Assertions:**

A0: Morty follows Rick
A1: Morty listens to Rick
A2: Morty acts as a human shield
A3: Morty doesn't know this is true

0) Check if Goal in assertion
1) See what rule can produce goal and add conditions to tree. Multiple matches => use OR Node.
2) Set new goal and recurse. Short circuit: AND subgoal is False

**B1 (7 points)** Suppose you are performing backward chaining with this set of rules and assertions, starting from the hypothesis "Morty is the Mortiest Morty". Numbers indicate the order in which hypotheses are checked. Unnumbered nodes have not been explored. Which of the following trees would result at the end of backward chaining? (Circle the **single best** answer.)




evaluates to true so no need to check

check rules in order

**B2 (1 point)** Based on the results of backward chaining, is the hypothesis "Morty is the Mortiest Morty" true? (Circle one)

YES          NO          CAN'T TELL

rules that could be true using what we know.

## Part C: Forward chaining (12 points)

1) Go through all rules and mark matches
2) Fire first matched rule that you can
3) Repeat

Then statement not in assertions

Consider the following rule-based system:

**Rules:**

```
R0     IF    (OR '(?x) follows Rick',
                 '(?x) listens to Rick'),
       THEN '(?x) is a Morty'

R1     IF    (AND '(?y) acts as a human shield',
                  '(?y) is a Morty',
                  '(?y) doesn't know this is true'),
       THEN '(?y) is the Mortiest Morty'
```

**Assertions:**

A0: Morty follows Rick
A1: Morty listens to Rick
A2: Morty acts as a human shield
A3: Morty doesn't know this is true

Suppose you are going to perform three rounds of forward chaining using these rules and assertions. (Hint: It may help to actually perform the forward chaining.)

**C1 (3 points)** In the first round, which rule(s) will match? (Circle one)

**Only R0**          **Only R1**          **Both R0 and R1**          **Neither R0 nor R1**

**C2 (9 points)** Which rule will fire in each round? Circle the **single best answer** in each case.

| | | | | |
|---|---|---|---|---|
| Round 1 | **Only R0** | **Only R1** | **Both R0 and R1** | **Neither R0 nor R1** |
| Round 2 | **Only R0** | **Only R1** | **Both R0 and R1** | **Neither R0 nor R1** |
| Round 3 | **Only R0** | **Only R1** | **Both R0 and R1** | **Neither R0 nor R1** |

only one rule can fire at a time

## Part D: More Forward Chaining (11 points)

These are independent questions about unrelated rule-based systems. Assume there are no DELETE statements in any system.

**D1 (3 points)** Suppose a rule matches in the first round of forward chaining and **does not contain any "NOT"s in its antecedent**. Is this rule guaranteed to match in every subsequent round? (Circle one)

YES NO

**D2 (3 points)** Suppose a rule matches in the first round of forward chaining and **contains a "NOT" in its antecedent**. Is this rule guaranteed to match in every subsequent round? (Circle one)

YES NO

New assertion could be added to make the NOT False

**D3 (5 points)** Consider the following rule and possible sets of assertions. If you were to perform forward chaining, which of the sets would cause the statement `'Rick says "This is Rickdiculous!"'` to be **added** to the list of assertions? Circle **all** that apply.

```
R0    IF    AND('(?x) is lost',
                NOT '(?x) is going to find Morty')
      THEN '(?x) says "This is Rickdiculous!"'
```

Set 1: A0: Rick is lost ← If assertion not there ⇒ equivalent to NOT beigng true

Set 2: A0: Rick is lost
A1: Rick is ***not*** going to find Morty

Set 3: A0: Rick is going to find Morty
A1: Rick is lost

Set 4: A0: Rick is lost
A1: Rick says "This is Rickdiculous!"

Set 5: A0: Rick says "This is Rickdiculous!"

## Quiz 1, Problem 2: Search and Games (50 points)

### Part A: Search Questions (27 points)

Here's a list of search algorithms and questions. Each question is worth **3 points**. For each question, list **ALL** of the search algorithms (A,B,C,D,E,F,G,H) that apply. If none of the search algorithms apply, write **NONE** instead. Some algorithms may be used more than once, and some may not be used at all.

| A | B | C |
|---|---|---|
| A* search | Branch and bound (no heuristic, no extended set) | Depth-first search (with backtracking) |
| **D** | **E** | **F** |
| Branch and bound **with a heuristic** (no extended set) | Branch and bound **with an extended set** (no heuristic) | *Breadth*-first search |
| **G** | **H** | |
| *Best*-first search | Hill-climbing (with backtracking) | |

① Which of these algorithms add children to the front of the agenda like a stack (without sorting the agenda)?

C, H

② Which of these algorithms add children to the back of the agenda like a queue (without sorting the agenda)?

F

③ Which of these algorithms will need to know the weights of the edges in the graph?

A, B, D, E

④ Which of these algorithms will need a heuristic estimate of remaining distance to the goal?

A, D, G, H

⑤ Which of these algorithms will always find the shortest path to the goal? (If an algorithm uses a heuristic, do not assume that the heuristic is admissible or consistent.)

B, E

⑥ Which of these algorithms can use a consistent heuristic to always find the shortest path to the goal (if one exists)? Think carefully.

AD consistent => Ad missable

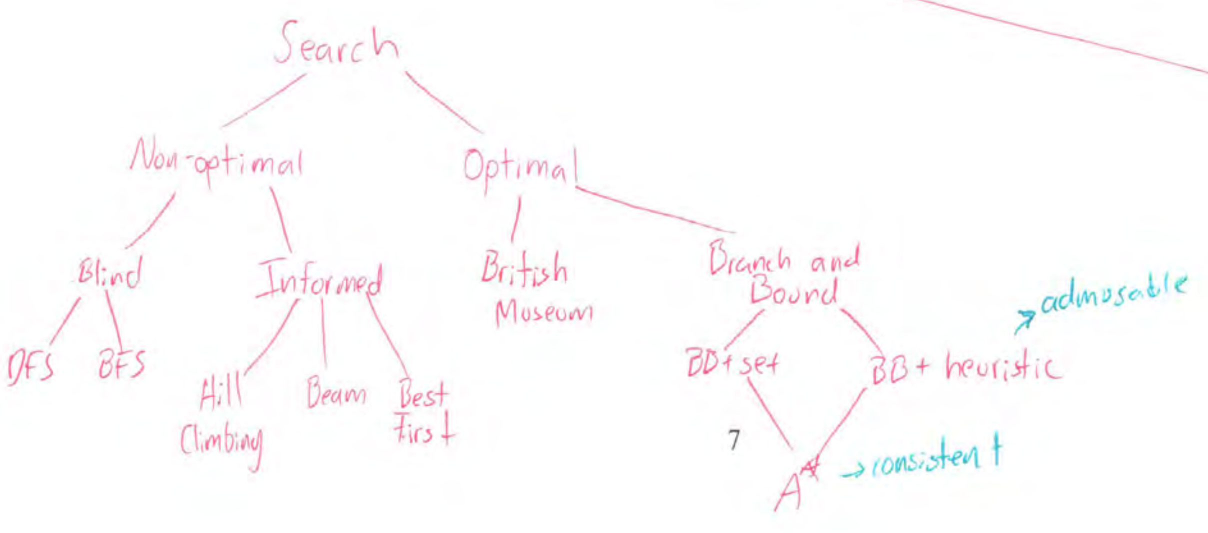
⑦ Which of these algorithms have a built-in limit on the size of the agenda?

Beam Search

⑧ Which of these algorithms use some kind of cost function to determine which path to extend next?

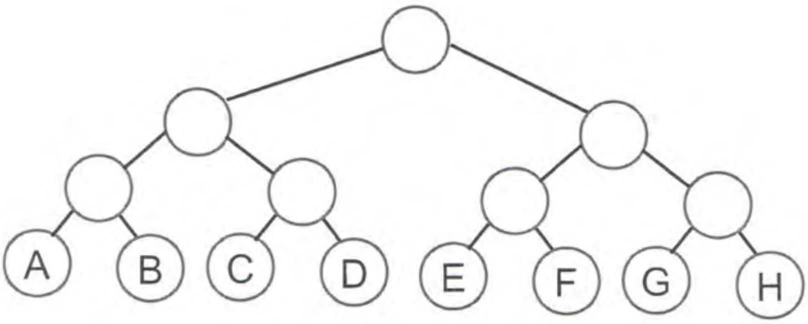
heuristic, path length, or heuristic + pl

~~⑨ Which of these algorithms may continue searching even after finding a path to the goal?~~



## Part B (14 points)

In this problem, you will evaluate how alpha-beta pruning performs in the best case and in the worst case.



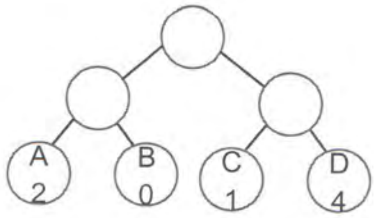

**B1 (7 points)** Suppose you perform alpha-beta pruning on the above game tree. List ***all*** of the leaf nodes you would have to **statically evaluate** in the **best** case, i.e. if the static values in the tree cause alpha-beta to prune the greatest possible number of nodes.

**B2 (7 points)** Suppose you perform alpha-beta pruning on the above game tree. List ***all*** of the leaf nodes you would have to **statically evaluate** in the **worst** case, i.e. if the static values in the tree cause alpha-beta to prune the least possible number of nodes.

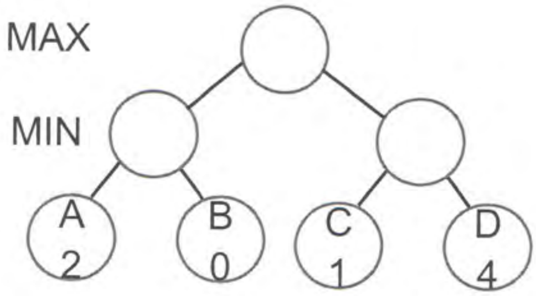
## Part C (9 points)

While performing alpha-beta pruning with progressive deepening, you generate this game tree that looks <u>two</u> moves ahead.

In the next round of progressive deepening, you will look <u>three</u> moves ahead.

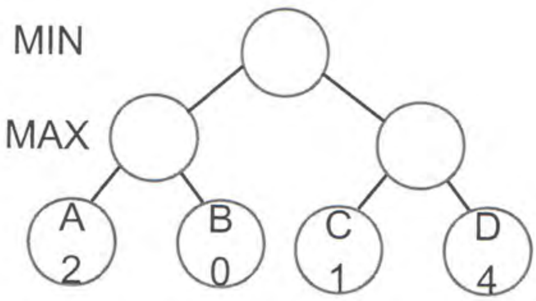



**C1 (5 points)** How should you reorder this tree to maximize the possibility of pruning in the next round **if MAX will make the first move**?




List the leaf nodes (A, B, C, D) in left-to-right order as they appear in the rearranged tree.

**C2 (4 points)** How should you reorder this tree to maximize the possibility of pruning in the next round **if MIN will make the first move**?




List the leaf nodes (A, B, C, D) in left-to-right order as they appear in the rearranged tree.

# 6.034 Quiz 1
# 26 September 2014

| Name | |
|---|---|
| Email | |

Circle your TA **(for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

| | | |
|---|---|---|
| **Josh Blum** | **Elisa Castañer** | **Pedro Cattori** |
| **Malcom Gilbert** | **Dylan Holmes** | **Jessica Noss** |
| **Theophilus Teeyay** | **Duncan Townsend** | **Siyao Xu** |

Robert Luo

| Problem number | Maximum | Score | Grader |
|---|---|---|---|
| 1 | 40 | | |
| 2 | 60 | | |
| Total | 100 | | |

There are 12 pages in this quiz, including this one, but not including tear-off sheets. Tear-off sheets with duplicate drawings and data are located after the final page of the quiz. As always, open book, open notes, open just about everything, including a calculator, but no computers.

This page contains no quiz material.

## Problem 1: Rule-based systems (40 points)

Because roommates are often unpredictable, Josh decides to use his knowledge of rule-based systems to help him predict their behavior. Based on his roommates' previous actions, he constructs a list of rules and a list of assertions about them, as follows:

**Rules:**

```
P0   IF    OR('(?A) plays basketball',
              '(?A) is lazy'),
     THEN  '(?A) hasn't eaten'

P1   IF    '(?A) likes to code',
     THEN  '(?A) is working'

P2   IF   AND( '(?A) isn't busy',
               '(?B) enjoys playing basketball'),
     THEN  '(?A) plays basketball'

P3   IF   AND( '(?A) isn't busy',
              NOT('(?A) is lazy')),
     THEN '(?A) will get his ID'

P4   IF   AND( '(?A) hasn't eaten',
               '(?A) isn't busy',
               '(?B) likes to cook'
                NOT('(?A) plays basketball')),
     THEN '(?A) has dinner with (?B)'
```

**Assertions for Part A** (backward chaining):

A0: Josh is lazy
A1: Mother Aryan enjoys playing basketball
A2: William likes to cook
A3: Louis likes to code

For your convenience, a copy of these rules and assertions is provided on a tear-off sheet after the last page of the quiz.

## Part A: Backward chaining (17 points)

Make the following assumptions about backward chaining:

- The backward chainer tries to find a matching assertion in the list of assertions. If no matching assertion is found, the backward chainer tries to find a rule with a matching consequent. In case none are found, then the backward chainer assumes the hypothesis is **false.**
- The backward chainer never alters the list of assertions, so it can derive the same result multiple times.
- Rules are tried in the order they appear.
- Antecedents are tried in the order they appear.
- Lazy evaluation/short circuiting is in effect.

### A1 (14 points)

Simulate backward chaining with the hypothesis:

Josh has dinner with William

Write all the hypotheses the backward chainer looks for in the database in the order that the hypotheses are looked for. The table may have more lines than you need. We recommend that you use the space provided on the next page to draw the goal tree that would be created by backward chaining from this hypothesis. The goal tree will help us to assign partial credit, in the event you have mistakes on the list.

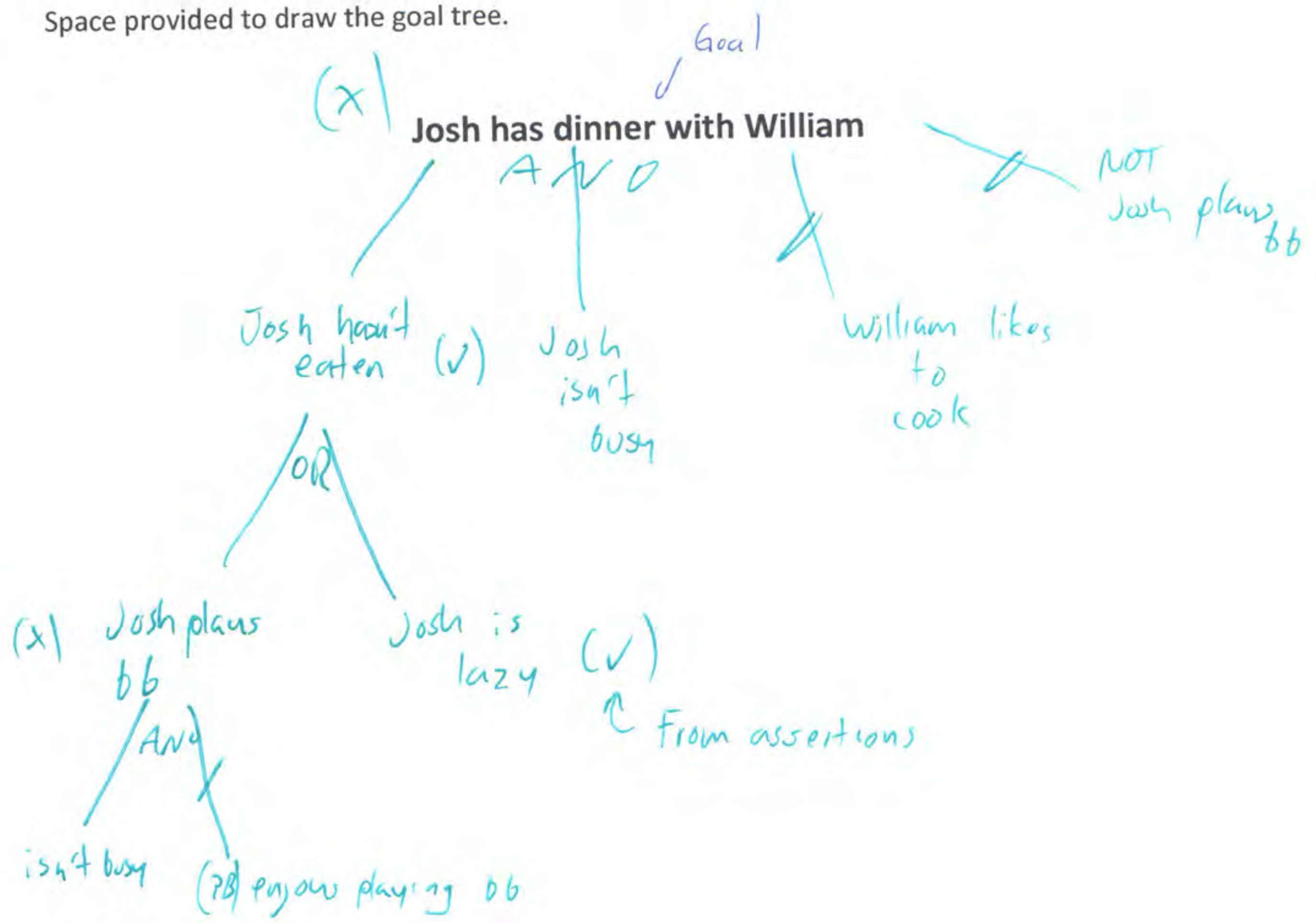
| | |
|---|---|
| 1 *Josh has dinner with William* | 9 |
| 2 Josh hasn't eaten | 10 |
| 3 Josh plays basketball | 11 |
| 4 Josh isn't busy | 12 |
| 5 Josh is lazy | 13 |
| 6 Josh isn't busy | 14 |
| 7 | 15 |
| 8 | 16 |

### A2 (3 points)

Does Josh have dinner with William? (Does backward chaining prove the hypothesis "Josh has dinner with William"?) Circle one:

YES NO

Space provided to draw the goal tree.




1) See what rules can produce Goal or check assertions and add conditions to tree. IF more than one rule matches add to tree with or node.

2) Set new goal and recurse.
   Short circuit: AND subgoal is False
   OR subgoal is true

## Part B: Forward chaining (23 points)

Make the following assumptions about forward chaining:

- Assume rule-ordering conflict resolution
- New assertions are added to the bottom of the list of assertions.
- If a particular rule matches assertions in the list of assertions in more than one way, the matches are considered in the order corresponding to the top-to-bottom order of the matched assertions. Thus, if a particular rule has an antecedent that matches both A1 and A2, the match with A1 is considered first.

### B1 (20 points)

Patrick, Josh's new roommate, has just moved into town. Josh quickly adds a new assertion to the list to account for his first impression of this unknown character.

rules that could be true using what we know

For each iteration

**Assertions for Part B** (forward chaining):

1) Go through all rules and mark matches
2) Fire first matched rule that you can
3) Repeat

Then statement not in assertion

A0: Josh is lazy
A1: Mother Aryan enjoys playing basketball
A2: William likes to cook
A3: Louis likes to code
**A4: Patrick isn't busy**

Perform forward chaining using the rules mentioned on page 3 and <u>this new list</u> of assertions. For the first two iterations, fill out the first two rows in the table below, noting the rules whose antecedents match the data, the rule that fires, and the new assertions that are added by the rule. For the remainder, supply only the fired rules and new assertions.

There may be more rows in the table than you'll need.

| | Matched | Fired | New Assertions Added to the List of Assertions |
|---|---|---|---|
| 1 | P0, P1, P2, P3 | P0 | Josh hasn't eaten |
| 2 | P0, P1, P2, P3 | P1 | Lois is working |
| 3 | | P2 | Patrick Plays basketball |
| 4 | | P0 | Patrick hasn't eaten |
| 5 | | P3 | Patrick will get his id |
| 6 | | | |
| 7 | | | |
| 8 | | | |

### B2 (3 points)

Given the assertions in part B1, which of the rules (P0, P1, P2, P3, P4) will never fire during forward chaining? (If all of the rules will fire, write NONE instead.)

P4

# Problem 2: Optimal Search (60 points)

## Part A: Search me (9 points)

**A1.** You have a summer internship with a travel agency. Your first task is to find an automobile route from Boston to San Francisco, but not just any route: you have to find the *most scenic* route.

- The road connecting any two cities has a scenic value, so the most scenic route is the one whose sum of city-to-city scenic values is **maximum**. NP hard
- There is no limit to how scenic a road connecting two cities can be.
- Your customers want to feel like they are making progress, so your route must include only connections traveled from east to west.
- You are to work with a map that includes only major cities and the roads connecting them, so you are not worried about any exponential blowup that could make certain search methods computationally infeasible, but of course you would like your search method to be as efficient as possible.

Of the following, your best choice is (circle the **single best** answer):

1 British Museum algorithm ← Need to look at all paths !

2 Hill climbing

3 Bi-directional breadth-first search.

4 Bi-directional depth-first search.

5 Branch and bound search

**Note:** a *bi-directional search* expands one path starting from S, then one from G, then one from S, and so on, until a terminal node on a path from S is the same as a terminal node on a path from G.

**A2.** Your second task is to find an automobile route from point S to point G. This time, you want to find the route that goes through the fewest cities. Of course, you would like to be as efficient as possible.

Of the following, your best choice is (circle the **single best** answer):

1 British Museum algorithm

2 Hill climbing

3 Bi-directional breadth-first search. Ignores values but minimizes path length

4 Bi-directional depth-first search.

5 Branch and bound search

**A3.** Completely exhausted after your first, hard day at the travel company, which is in Paris, you have to walk to your apartment because the taxis are on a one-day strike. You are not sure how to get there, and you have no map, but your apartment is near the Eiffel tower, visible everywhere in the city.

Of the following, your best choice is (circle the **single best** answer):

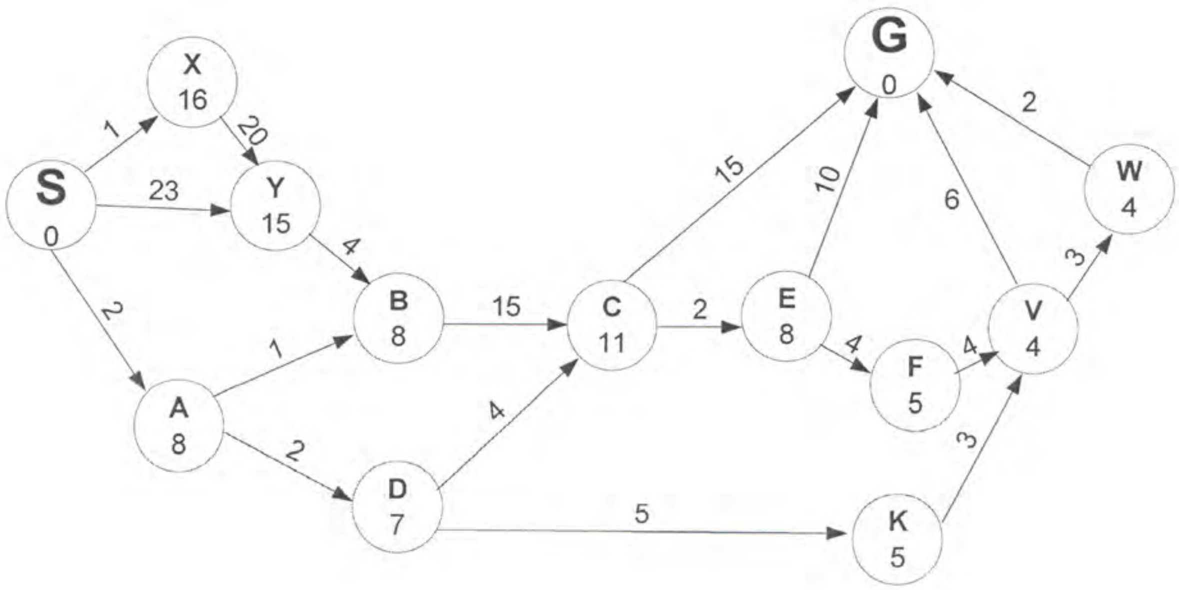
1 British Museum algorithm
2 Hill climbing
3 Bi-directional breadth-first search.
4 Bi-directional depth-first search.
5 Branch and bound search

## Part B: So busted! (17 points)

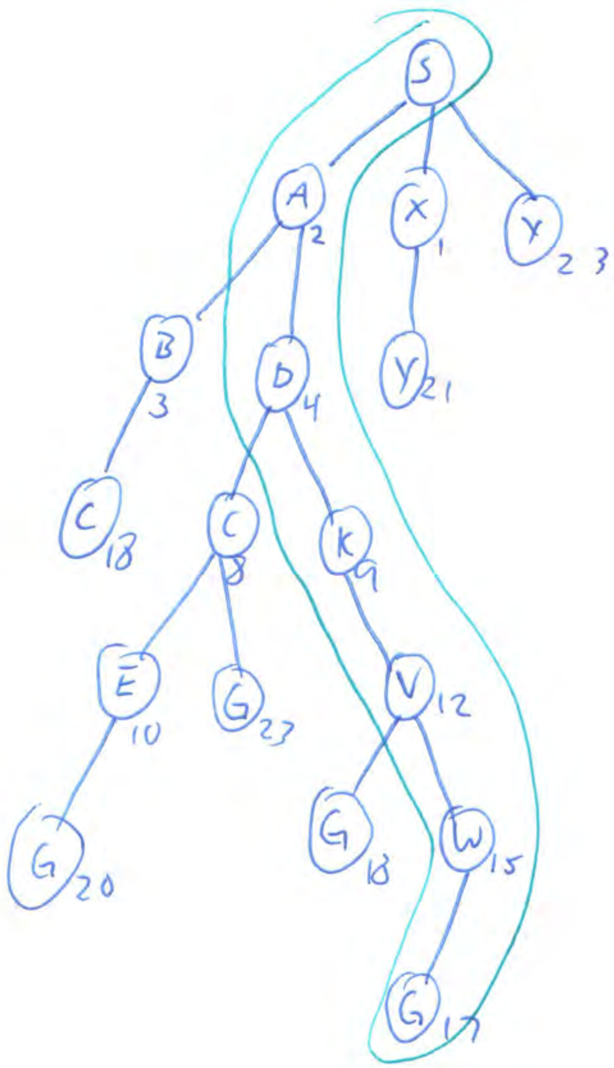
**B1 (14 points)**

Long-suffering teen Candace has just discovered her siblings goofing off and must race home to tell their mother. She decides to use **branch and bound, with no heuristic and no extended set**, to find the quickest route home starting from node **S** and ending at node **G**.

Referring to the graph below, use branch and bound (with no heuristic and no extended set) to find a route from node S to node G. The numbers inside each node indicate heuristic estimates of distance to the goal, which you will not need in this part. For credit, **draw your search tree** on the following page.



- **Draw the children of each node in alphabetical order**, and **break ties using lexicographic/dictionary order** (for example, if two paths S-P-U and S-Q-T are tied, you should extend path S-P-U before path S-Q-T).
- **Note #1:** For your convenience, a copy of the graph is provided on a tear-off sheet after the last page of the quiz.
- **Note #2:** Notice that the edges in this graph are **one-way**; paths must only go in the direction of the arrows.




- For path with shortest cost
- expand end node
  - calculate new path costs
- Repeat until G expanded

**B2 (3 points)**
What path does Candace find using branch and bound?

SADKVWG

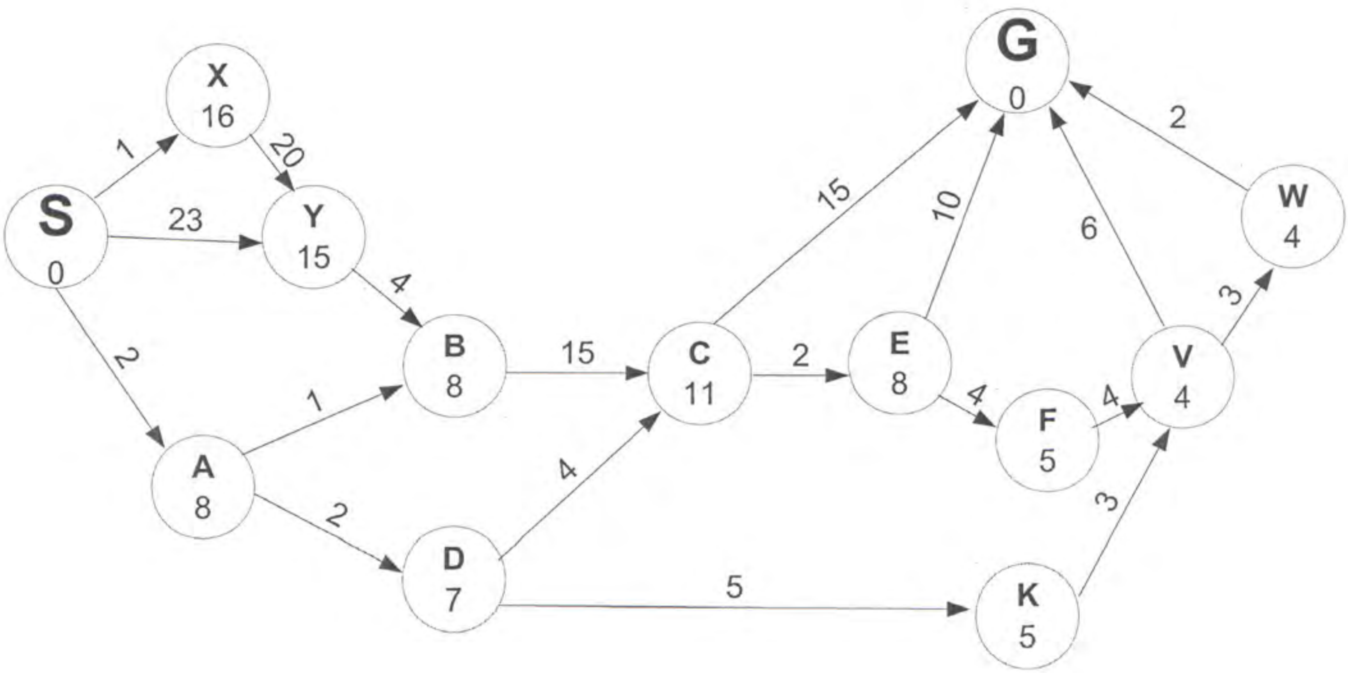
What is its total path length?

17

## Part C: Scram! (25 points)

Not eager to get scolded by their mother, Candace's siblings decide to look for a way home themselves—so, they ask an ostrich herder for heuristic estimates of the distance home from each node, then decide to use **A* search** (that is: branch and bound, with a heuristic and with an extended set) to look for a way home. The numbers inside each node indicate heuristic estimates of distance to the goal, which you will need in this part.

*(Here is a copy of the graph that was used in part B)*

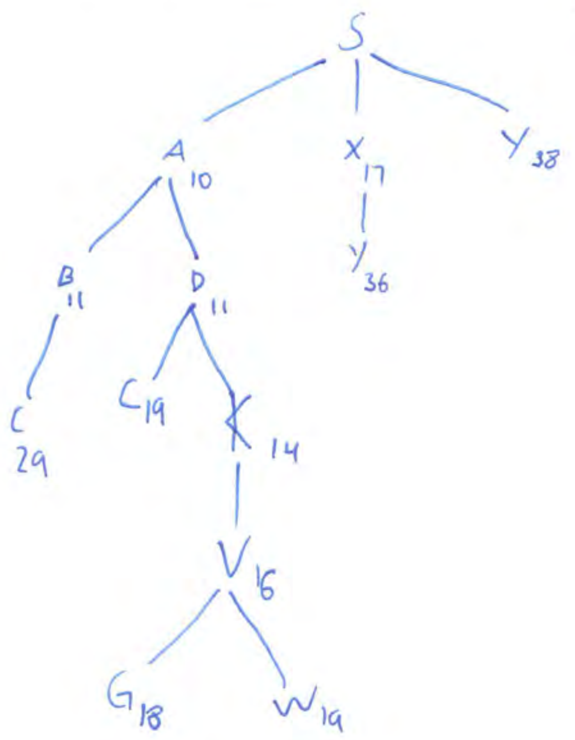


**C1 (14 points)**

Use A* search to find a route from node S to node G. For credit, **draw your search tree** below.

- **Draw the children of each node in alphabetical order**, and **break ties using lexicographic/dictionary order** (for example, if two paths S-P-U and S-Q-T are tied, you should extend path S-P-U before path S-Q-T).
- **Note #1:** For your convenience, a copy of the graph is provided on a tear-off sheet after the last page of the quiz.
- **Note #2:** Notice that the edges in this graph are **<u>one-way</u>**; paths must only go in the direction of the arrows.

· Path cost + heuristic of end node




**C2 (6 points)**

List the extended nodes in the order you extended them during A* search.

SABDKVXG

**Important!** The quiz continues on the next page. ↪

**C3 (3 points)**
What path do the siblings find using A* search?

SADKVG

What is its total path length?

18

**C4 (2 points)**
Regardless of your answers to parts C1 and C2, assume that branch and bound found a shorter path than A*. Very briefly, explain this result.

Inconsistent + inadmissive heuristic
(w is overestimate

# 6.034 Quiz 2

## 15 October 2014

| | |
|---|---|
| Name | Fernando Trujano PRACTICE |
| Email | |

Circle your TA **(for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

| | | |
|---|---|---|
| **Josh Blum** | **Elisa Castañer** | **Pedro Cattori** |
| **Jack Florey** | **Malcom Gilbert** | **Dylan Holmes** |
| **Jessica Noss** | **Duncan Townsend** | **Siyao Xu** |

| Problem number | Maximum | Score | Grader |
|---|---|---|---|
| 1 | 50 | | |
| 2 | 50 | | |
| Total | 100 | | |

| | | | |
|---|---|---|---|
| SRN | 6 | | |

There are 15 pages in this quiz, including this one. As always, open book, open notes, open just about everything, including a calculator, but no computers.

# Problem 1: Identification of Trees (50 points)

## Part A: Identification Trees (28 points)

While walking along the Charles River in the fall, your friend points out six trees and tells you whether they are Maple or Oak trees. Eager to identify other trees yourself, you observe what appear to be their key features. In particular, you record whether each tree **Has leaves (Yes or No)**, whether it has **Orange foliage (Yes or No)**, its **Leaf shape (Pointy or Rounded)**, and its **Bark texture (Glossy, Furrowed, or Smooth)**. Your resulting training data is summarized in the table below:

| Tree type | Has leaves* | Orange foliage | Leaf shape | Bark texture |
|---|---|---|---|---|
| Oak | Yes | Yes | Pointy | Glossy |
| Oak | Yes | No | Pointy | Furrowed |
| Oak | Yes | No | Rounded | Furrowed |
| Maple | Yes | Yes | Pointy | Furrowed |
| Maple | Yes | Yes | Pointy | Smooth |
| Maple | Yes | No | Pointy | Smooth |

*Although some trees lose their leaves in the fall, all six of these trees still have leaves.

**A1 (4 points)** Based on these six training points, order the four feature tests (**Has leaves**, **Orange foliage**, **Leaf shape**, and **Bark texture**) from least to greatest disorder. (You don't need to show your calculations; you may use your intuition. If you prefer not to use your intuition, see part A2 below.) Break any ties alphabetically.

| Bark Texture | Leaf shape | Orange Foliage | Has leaves |
|---|---|---|---|
| LEAST DISORDER | | | GREATEST DISORDER |

**A2 (8 points)** Next, compute the disorder of each feature test. Use the table of logarithms below to express your answer as sums and products of decimals and fractions only. Your final answer should have **no logarithms** in it. Space is provided on the next page to show your work.

| Test | Has leaves | Orange foliage | Leaf shape | Bark texture |
|---|---|---|---|---|
| Disorder | 1 | .9 | .8 | .45 |

$$-\left[\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right] = 1 \qquad -\left[\frac{2}{5}\log_2\frac{2}{5} + \frac{3}{5}\log_2\frac{3}{5}\right] \approx 0.97 \qquad -\left[\frac{1}{3}\log_2\frac{1}{3} + \frac{2}{3}\log_2\frac{2}{3}\right] \approx 0.9$$

$$-\left[\frac{1}{4}\log_2\frac{1}{4} + \frac{3}{4}\log_2\frac{3}{4}\right] \approx 0.8 \qquad -\left[\frac{1}{5}\log_2\frac{1}{5} + \frac{4}{5}\log_2\frac{4}{5}\right] \approx 0.72 \qquad -\left[\frac{1}{6}\log_2\frac{1}{6} + \frac{5}{6}\log_2\frac{5}{6}\right] \approx 0.65$$

Show your disorder calculations for partial credit. For your convenience, a copy of the training data is provided on the next page.

$$d = -1 \sum_c \frac{\#\,\text{pts } c}{\#\,\text{pts branch}} \log_2\left(\frac{\#\,\text{pts } c}{\#\,\text{pts } b}\right)$$

$$D = \sum_b \frac{\#\,\text{pts } b}{\#\,\text{total pts}} \cdot d(b)$$

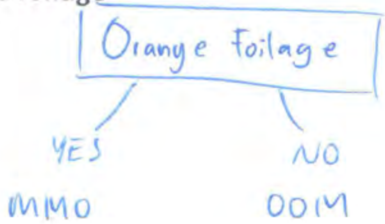
**Has leaves**


$$d_{YES} = -\frac{3}{6}\log_2\left(\frac{3}{6}\right) + \frac{3}{6}\log_2\left(\frac{3}{6}\right) = 1$$

$$d_{NO} = 0\log_2(0) + 0\log_2(0) = 0$$

$$D = \frac{6}{6} d_{YES} + \frac{0}{6} d_{NO} = 1 \qquad \boxed{D = 1}$$

**Orange foliage**



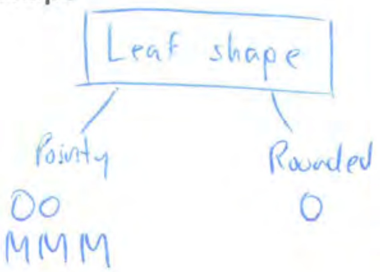

$$d_{YES} = -\frac{2}{3}\log_2\frac{2}{3} + \frac{1}{3}\log_2\frac{1}{3} \approx .9$$

$$d_{NO} = \text{"} \approx .9$$

$$D = \frac{3}{6}(.9) + \frac{3}{6}(.9) = .9$$

$$\boxed{D = .9}$$

**Leaf shape**

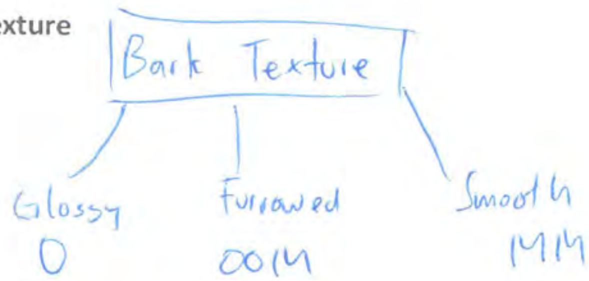



$$d_{Pointy} = -\frac{2}{5}\log_2\left(\frac{2}{5}\right) + \frac{3}{5}\log_2\left(\frac{3}{5}\right) \approx .97$$

$$d_{Roundy} = -\frac{1}{1}\log_2\left(\frac{1}{1}\right) + \frac{0}{1}\log_2\left(\frac{0}{1}\right) = 0$$

$$D = \left(\frac{5}{6}\right) d_{Pointy} + \left(\frac{1}{6}\right) d_{Round} = .8 \qquad \boxed{D = .8}$$

**Bark texture**




$$d_G = 0$$

$$d_F = .9$$

$$d_S = 0$$

$$D = \frac{1}{6} d_G + \frac{3}{6} d_F + \frac{2}{6} d_S = .45$$

$$\boxed{D = .45}$$

**A3 (14 points)**
Choosing from the four feature tests (**Has leaves**, **Orange foliage**, **Leaf shape**, and **Bark texture**), construct the complete disorder-minimizing identification tree for classifying the training data as Maple or Oak.

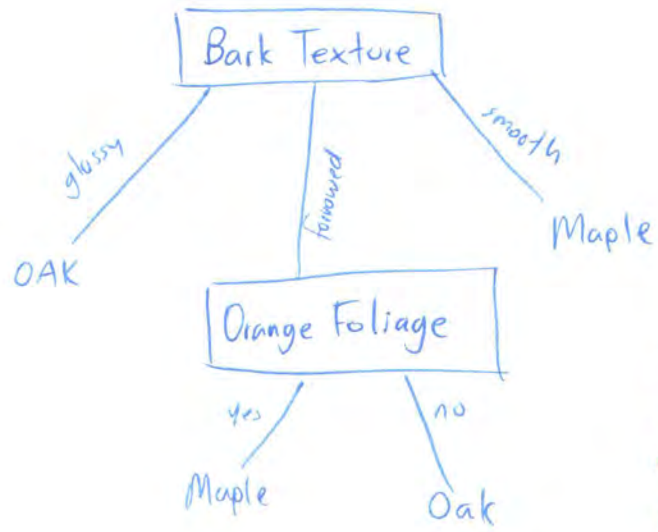
(For your convenience, a copy of the training data is included here.)

| Tree type | Has leaves* | Orange foliage | Leaf shape | Bark texture |
|---|---|---|---|---|
| Oak | Yes | Yes | Pointy | Glossy |
| Oak | Yes | No | Pointy | Furrowed |
| Oak | Yes | No | Rounded | Furrowed |
| Maple | Yes | Yes | Pointy | Furrowed |
| Maple | Yes | Yes | Pointy | Smooth |
| Maple | Yes | No | Pointy | Smooth |

*Although some trees lose their leaves in the fall, all six of these trees still have leaves.

**Draw your identification tree** in the space below:

1) Pick test w/ least disorder
2) Recalculate disorder for tests using remaining points




← Test with least disorder

Only consider Forrowed and recalculate

Has leaves: YES → OOM; NO
$d_{YES} = -\frac{2}{3}\log\frac{2}{3} + \frac{1}{3}\log\frac{1}{3} = .9$
$d_{NO} = 0$
$D = .9$

Orange Foliage: NO → OO; YES → M
$d_{NO} = 0$
$d_{YES} = 0$
$D = 0$

Leaf shape: Pointy → OM; Rounded → O
$d_P = -\frac{1}{2}\log\frac{1}{2} + \frac{1}{2}\log\frac{1}{2} = 1$
$d_R = 0$
$D = 2/3$

Bark Texture ← Already used
furrowed → OOM; glossy → X; smooth → X
$D = .9$

Additional space provided for disorder calculations, if you need it.

**A4 (2 points)**

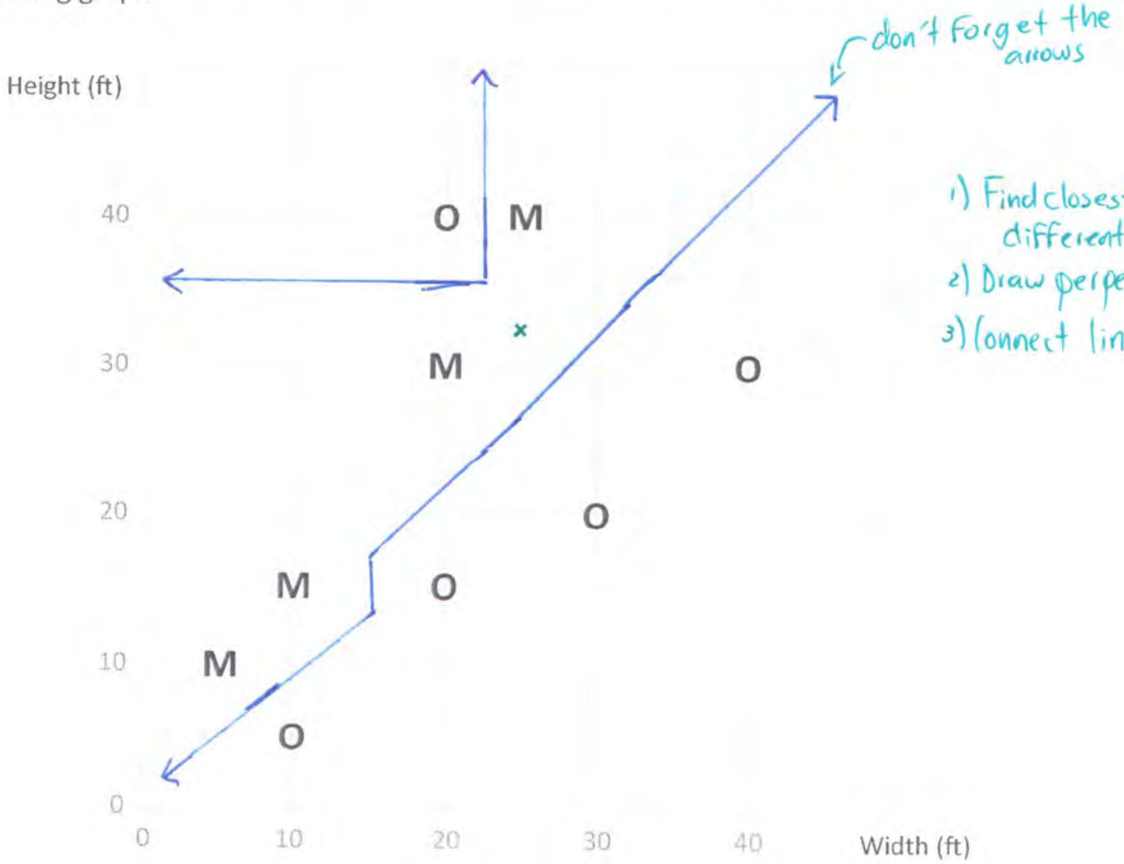
How would your identification tree classify a specimen with the following features? Just follow branches of id Tree

| Has leaves | Orange foliage | Leaf shape | Bark texture |
|---|---|---|---|
| Yes | Yes | Rounded | Furrowed |

Circle one: MAPLE OAK CAN'T TELL

## Part B: Nearest neighbors (22 points)

You plan to go out and identify more trees the next day, but that night there is an early snowstorm! To your dismay, you find that a blanket of snow has obscured the leaves and bark of all the trees, so you can no longer use those as distinguishing features. Undaunted, you decide to venture out into the fresh snow and use trigonometry to measure the width and height of trees that are known to be Maple (**M**) or Oak (**O**). Your measurements are plotted on the following graph:




1) Find closest pair of different types
2) Draw perpendicular bisecto
3) Connect lines

**B1 (12 points)** On the graph above, **draw the decision boundaries** produced by 1-Nearest Neighbors.

**B2 (8 points)** One of your classmates hears that you've been identifying trees. She wants to know what type of tree is knocking snow onto her second-story windowsill, so she calls you with the information that her tree is **25 ft wide** and **32 ft tall**. How would this tree be classified with each of the following classifiers? (Circle the best answer in each case.)

| | | | |
|---|---|---|---|
| 1-Nearest Neighbors? | MAPLE | OAK | CAN'T TELL |
| 3-Nearest Neighbors? | MAPLE | OAK | CAN'T TELL |
| 5-Nearest Neighbors? | MAPLE | OAK | CAN'T TELL |
| 9-Nearest Neighbors? | MAPLE | OAK | CAN'T TELL |

Consider # neighbors close to point x and classyfy as most repeated neighbor.

**B3 (2 points)** Finally, you want to use Nearest Neighbors to classify the pre-snowstorm data in Part A. You realize it's possible, but you must first: (Circle the **one** best answer.)

(A) Pick just two of the features to plot.

(B) Define a distance metric for non-numeric features.

(C) Collect twice as many training points.

(D) Use cross-validation to guarantee overfitting.

(E) Devise an admissible heuristic.

Need to decide what "near" is for non-numeric features in Part A for the six mages.

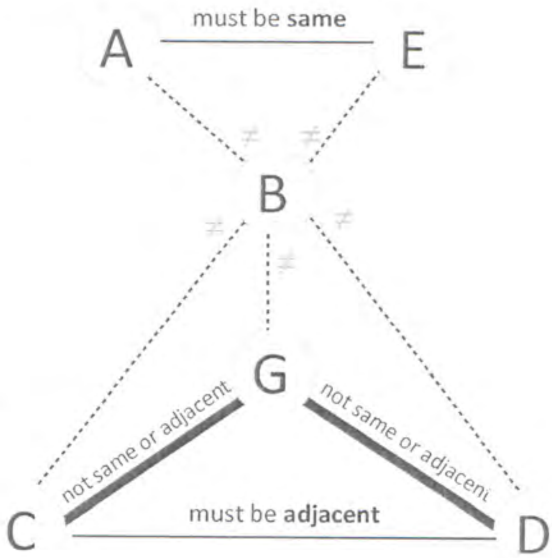
## Problem 2: Dominian Domiciles (50 points)

Six powerful mages (**A**jani, **B**olas, **C**handra, **D**ack, **E**lspeth, and **G**ideon) have enlisted you to help them settle down and find homes in the multiverse. They have five neighboring worlds to choose from, arranged in a row as shown. Some worlds could be empty, and some worlds could have more than one occupant.

| | | | | |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |

Naturally, mages are very particular about where they live, so your assignments must satisfy the following requirements:

1. **A**jani is **E**lspeth's mentor. They must live on the **same world**.
2. Everyone hates **B**olas. No one can live on the same world as him.
3. **C**handra wants to spy on **D**ack secretly—she must live on a world that's **adjacent to** **D**ack's world.
4. **G**ideon is a law mage. He despises the criminals **C**handra and **D**ack so much that **G**ideon **cannot live on the same world—or even on a world adjacent to** **C**handra's or **D**ack's.




(For your convenience, a graph of these constraints is shown here.)

Finally, the mages inform you that you can simplify the domains of these variables, as follows:

- **D**ack **must live on either World 1 or 4**, because he's a wanted criminal everywhere else.
- **E**lspeth **must live on World 3**, because she's stuck there and cannot leave.
- **B**olas **can't live on World 5**, because he fears its strange magic.

Thus, the initial domains of your variables are as follows:

| | |
|---|---|
| A | 1 2 3 4 5 |
| B | 1 2 3 4 |
| C | 1 2 3 4 5 |
| D | 1 4 |
| E | 3 |
| G | 1 2 3 4 5 |

## Quiz 1, Problem 1: Rule-based systems (50 points)

### Part A: General questions (19 points)

**A1 (15 points)** For each of the following statements, circle the **single best answer**.

1. **True** / $\boxed{\textbf{False}}$: During backward chaining, at most one rule can match the current hypothesis.
Many can match though at most 1 will fire

2. **True** / $\boxed{\textbf{False}}$: During forward chaining, at most one rule can match per round.

3. **True** / $\boxed{\textbf{False}}$: During forward chaining, a rule may match in multiple rounds, but each rule can fire *at most* once. Rule can fire in different rounds

4. **True** / $\boxed{\textbf{False}}$: In each round of forward chaining, the first rule that matches will fire.
Only fires if it would generate new assertion

5. **True** / $\boxed{\textbf{False}}$: In each round of forward chaining, at most one new assertion can be added to the list of assertions. Assertions. Then might have more than one

**A2 (4 points)** In general, which of the following conditions can cause backward chaining to short circuit? That is, which of these will cause the evaluation of subtrees to stop? (Circle **ALL** answers that apply)

A) A child of an AND subtree returns true.

B) A child of an AND subtree returns false.

C) A child of an OR subtree returns true.

D) A child of an OR subtree returns false

E) The backward chainer cannot find a match for the hypothesis in the list of assertions.

F) The backward chainer cannot find a match for the hypothesis in the antecedent of any rule.

## Part B: Backward chaining (8 points)

**Rules:**

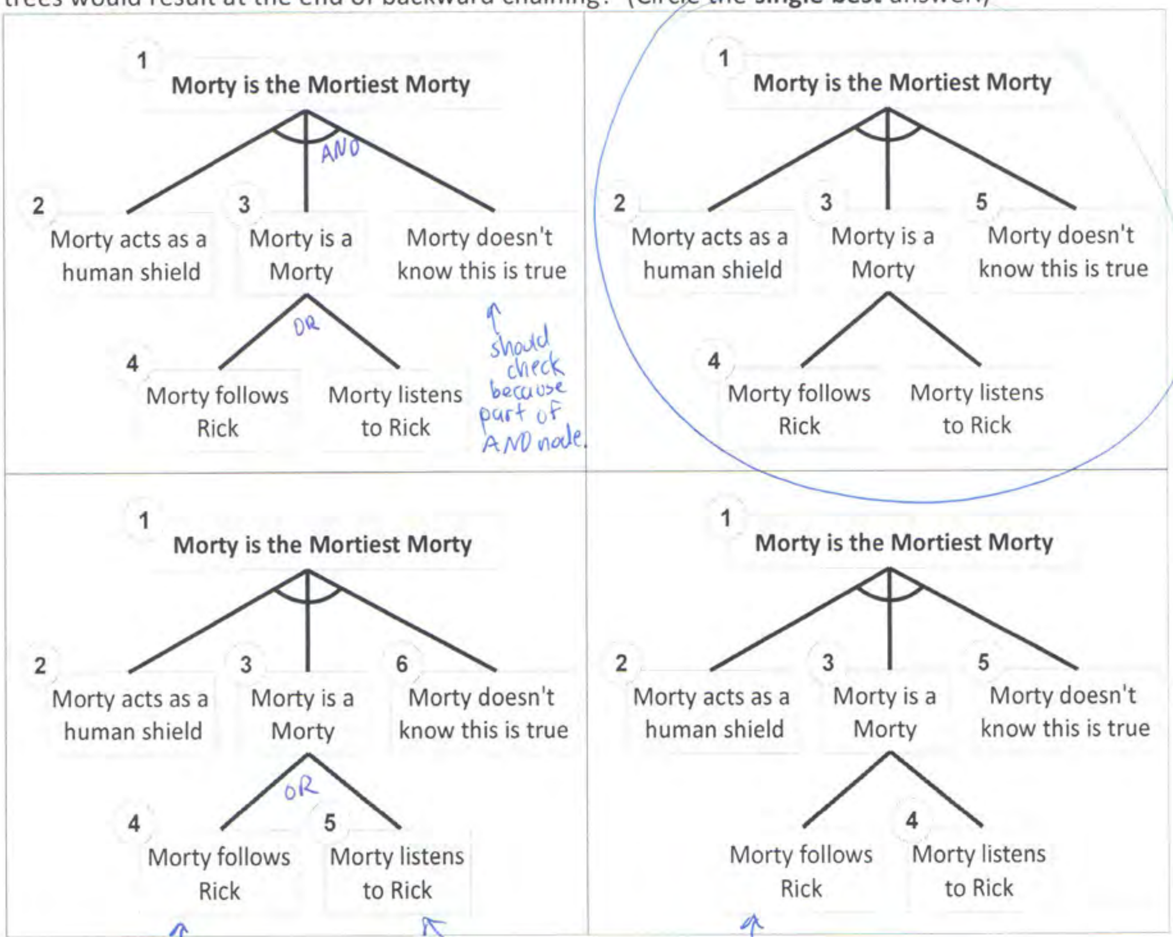
```
R0    IF   (OR '(?x) follows Rick',
               '(?x) listens to Rick'),
      THEN '(?x) is a Morty'

R1    IF   (AND '(?y) acts as a human shield',
                '(?y) is a Morty',
                '(?y) doesn't know this is true'),
      THEN '(?y) is the Mortiest Morty'
```

**Assertions:**

A0: Morty follows Rick
A1: Morty listens to Rick
A2: Morty acts as a human shield
A3: Morty doesn't know this is true

0) Check if Goal in assertion
1) See what rule can produce goal and add conditions to tree. Multiple matches => use OR Node.
2) Set new goal and recurse. Short circuit: AND subgoal is False

**B1 (7 points)** Suppose you are performing backward chaining with this set of rules and assertions, starting from the hypothesis "Morty is the Mortiest Morty". Numbers indicate the order in which hypotheses are checked. Unnumbered nodes have not been explored. Which of the following trees would result at the end of backward chaining? (Circle the **single best** answer.)




evaluates to true so no need to check

check rules in order

**B2 (1 point)** Based on the results of backward chaining, is the hypothesis "Morty is the Mortiest Morty" true? (Circle one)

YES NO CAN'T TELL

rules that could be true using what we know.

## Part C: Forward chaining (12 points)

1) Go through all rules and mark matches
2) Fire First matched rule that you can
3) Repeat

Then statement not in assertions

Consider the following rule-based system:

**Rules:**

```
R0    IF    (OR '(?x) follows Rick',
                '(?x) listens to Rick'),
      THEN '(?x) is a Morty'

R1    IF    (AND '(?y) acts as a human shield',
                 '(?y) is a Morty',
                 '(?y) doesn't know this is true'),
      THEN '(?y) is the Mortiest Morty'
```

**Assertions:**
A0: Morty follows Rick
A1: Morty listens to Rick
A2: Morty acts as a human shield
A3: Morty doesn't know this is true

Suppose you are going to perform three rounds of forward chaining using these rules and assertions. (Hint: It may help to actually perform the forward chaining.)

**C1 (3 points)** In the first round, which rule(s) will match? (Circle one)

**Only R0** **Only R1** **Both R0 and R1** **Neither R0 nor R1**

**C2 (9 points)** Which rule will fire in each round? Circle the **single best answer** in each case.

| | | | | |
|---|---|---|---|---|
| Round 1 | **Only R0** | **Only R1** | **Both R0 and R1** | **Neither R0 nor R1** |
| Round 2 | **Only R0** | **Only R1** | **Both R0 and R1** | **Neither R0 nor R1** |
| Round 3 | **Only R0** | **Only R1** | **Both R0 and R1** | **Neither R0 nor R1** |

only one rule can fire at a time

## Part D: More Forward Chaining (11 points)

These are independent questions about unrelated rule-based systems. Assume there are no DELETE statements in any system.

**D1 (3 points)** Suppose a rule matches in the first round of forward chaining and **does not contain any "NOT"s in its antecedent**. Is this rule guaranteed to match in every subsequent round? (Circle one)

**YES** **NO**

**D2 (3 points)** Suppose a rule matches in the first round of forward chaining and **contains a "NOT" in its antecedent**. Is this rule guaranteed to match in every subsequent round? (Circle one)

**YES** **NO**

New assertion could be added to make the NOT False

**D3 (5 points)** Consider the following rule and possible sets of assertions. If you were to perform forward chaining, which of the sets would cause the statement `'Rick says "This is Rickdiculous!"'` to be **added** to the list of assertions? Circle **all** that apply.

```
R0    IF   AND('(?x) is lost',
               NOT '(?x) is going to find Morty')
      THEN '(?x) says "This is Rickdiculous!"'
```

Set 1: A0: Rick is lost ← If assertion not there ⇒ equivalent to NOT beiyng true

Set 2: A0: Rick is lost
A1: Rick is ***not*** going to find Morty

Set 3: A0: Rick is going to find Morty
A1: Rick is lost

Set 4: A0: Rick is lost
A1: Rick says "This is Rickdiculous!"

Set 5: A0: Rick says "This is Rickdiculous!"

## Quiz 1, Problem 2: Search and Games (50 points)

### Part A: Search Questions (27 points)

Here's a list of search algorithms and questions. Each question is worth **3 points**. For each question, list **<u>ALL</u>** of the search algorithms (A,B,C,D,E,F,G,H) that apply. If none of the search algorithms apply, write **NONE** instead. Some algorithms may be used more than once, and some may not be used at all.

| **A** | **B** | **C** |
|---|---|---|
| A* search | Branch and bound (no heuristic, no extended set) | Depth-first search (with backtracking) |
| **D** | **E** | **F** |
| Branch and bound **with a heuristic** (no extended set) | Branch and bound **with an extended set** (no heuristic) | *Breadth*-first search |
| **G** | **H** | |
| *Best*-first search | Hill-climbing (with backtracking) | |

① Which of these algorithms add children to the front of the agenda like a stack (without sorting the agenda)?

C, H

② Which of these algorithms add children to the back of the agenda like a queue (without sorting the agenda)?

F

③ Which of these algorithms will need to know the weights of the edges in the graph?

A, B, D, E

④ Which of these algorithms will need a heuristic estimate of remaining distance to the goal?

A, D, G, H

⑤ Which of these algorithms will always find the shortest path to the goal? (If an algorithm uses a heuristic, do not assume that the heuristic is admissible or consistent.)

B, E

⑥ Which of these algorithms can use a consistent heuristic to always find the shortest path to the goal (if one exists)? Think carefully.

AD consistent => Ad missable

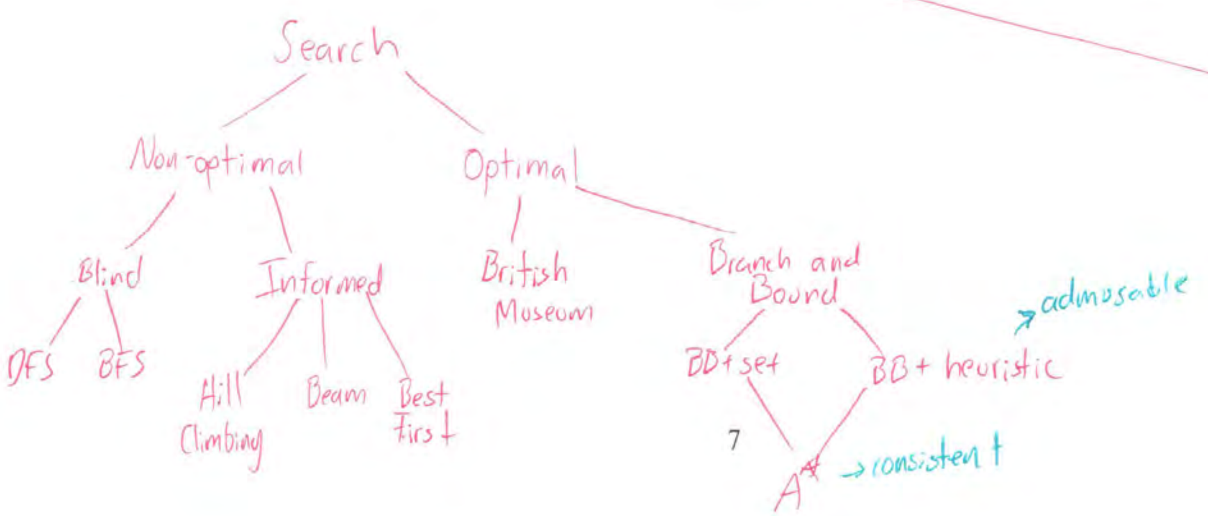
⑦ Which of these algorithms have a built-in limit on the size of the agenda?

Beam Search

⑧ Which of these algorithms use some kind of cost function to determine which path to extend next?

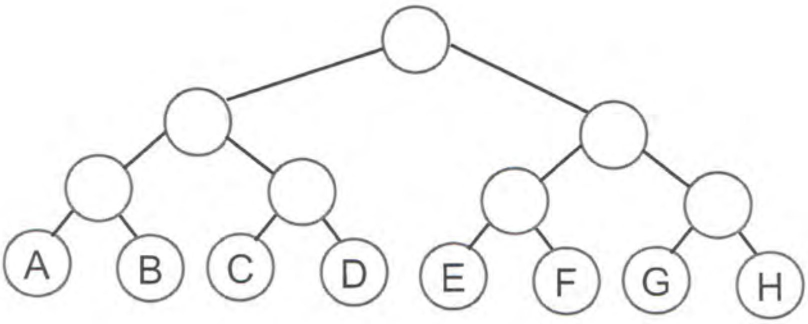
heuristic, path length, or heuristic + pl

⑨ Which of these algorithms may continue searching even after finding a path to the goal?



## Part B (14 points)

In this problem, you will evaluate how alpha-beta pruning performs in the best case and in the worst case.



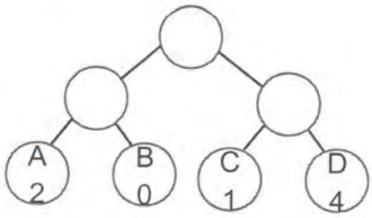

**B1 (7 points)** Suppose you perform alpha-beta pruning on the above game tree. List ***all*** of the leaf nodes you would have to **statically evaluate** in the **best** case, i.e. if the static values in the tree cause alpha-beta to prune the greatest possible number of nodes.

**B2 (7 points)** Suppose you perform alpha-beta pruning on the above game tree. List ***all*** of the leaf nodes you would have to **statically evaluate** in the **worst** case, i.e. if the static values in the tree cause alpha-beta to prune the least possible number of nodes.

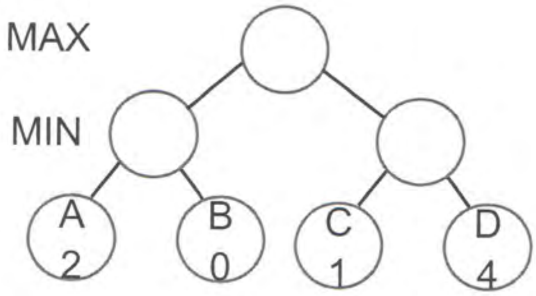
## Part C (9 points)

While performing alpha-beta pruning with progressive deepening, you generate this game tree that looks two moves ahead.

In the next round of progressive deepening, you will look three moves ahead.

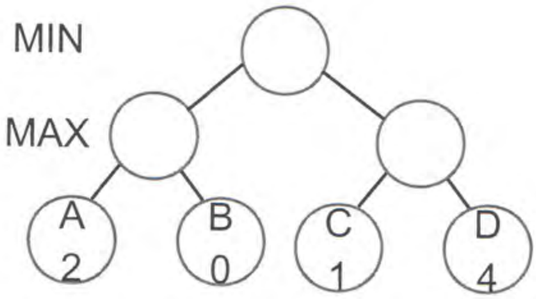



**C1 (5 points)** How should you reorder this tree to maximize the possibility of pruning in the next round **if MAX will make the first move**?




List the leaf nodes (A, B, C, D) in left-to-right order as they appear in the rearranged tree.

**C2 (4 points)** How should you reorder this tree to maximize the possibility of pruning in the next round **if MIN will make the first move**?




List the leaf nodes (A, B, C, D) in left-to-right order as they appear in the rearranged tree.

# 6.034 Quiz 1
# 26 September 2014

| | |
|---|---|
| Name | |
| Email | |

Circle your TA **(for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

**Josh Blum** **Elisa Castañer** **Pedro Cattori**

**Malcom Gilbert** **Dylan Holmes** **Jessica Noss**

**Theophilus Teeyay** **Duncan Townsend** **Siyao Xu**

Robert Luo

| Problem number | Maximum | Score | Grader |
|---|---|---|---|
| 1 | 40 | | |
| 2 | 60 | | |
| Total | 100 | | |

There are 12 pages in this quiz, including this one, but not including tear-off sheets. Tear-off sheets with duplicate drawings and data are located after the final page of the quiz. As always, open book, open notes, open just about everything, including a calculator, but no computers.

**This page contains no quiz material.**

## Problem 1: Rule-based systems (40 points)

Because roommates are often unpredictable, Josh decides to use his knowledge of rule-based systems to help him predict their behavior. Based on his roommates' previous actions, he constructs a list of rules and a list of assertions about them, as follows:

**Rules:**

```
P0   IF     OR('(?A) plays basketball',
               '(?A) is lazy'),
     THEN   '(?A) hasn't eaten'

P1   IF     '(?A) likes to code',
     THEN   '(?A) is working'

P2   IF    AND( '(?A) isn't busy',
                '(?B) enjoys playing basketball'),
     THEN   '(?A) plays basketball'

P3   IF    AND( '(?A) isn't busy',
               NOT('(?A) is lazy')),
     THEN '(?A) will get his ID'

P4   IF    AND( '(?A) hasn't eaten',
                '(?A) isn't busy',
                '(?B) likes to cook'
                 NOT('(?A) plays basketball')),
     THEN '(?A) has dinner with (?B)'
```

**Assertions for Part A** (backward chaining):

A0: Josh is lazy
A1: Mother Aryan enjoys playing basketball
A2: William likes to cook
A3: Louis likes to code

For your convenience, a copy of these rules and assertions is provided on a tear-off sheet after the last page of the quiz.

## Part A: Backward chaining (17 points)

Make the following assumptions about backward chaining:

- The backward chainer tries to find a matching assertion in the list of assertions. If no matching assertion is found, the backward chainer tries to find a rule with a matching consequent. In case none are found, then the backward chainer assumes the hypothesis is **false.**
- The backward chainer never alters the list of assertions, so it can derive the same result multiple times.
- Rules are tried in the order they appear.
- Antecedents are tried in the order they appear.
- Lazy evaluation/short circuiting is in effect.

### A1 (14 points)

Simulate backward chaining with the hypothesis:

Josh has dinner with William

Write all the hypotheses the backward chainer looks for in the database in the order that the hypotheses are looked for. The table may have more lines than you need. We recommend that you use the space provided on the next page to draw the goal tree that would be created by backward chaining from this hypothesis. The goal tree will help us to assign partial credit, in the event you have mistakes on the list.

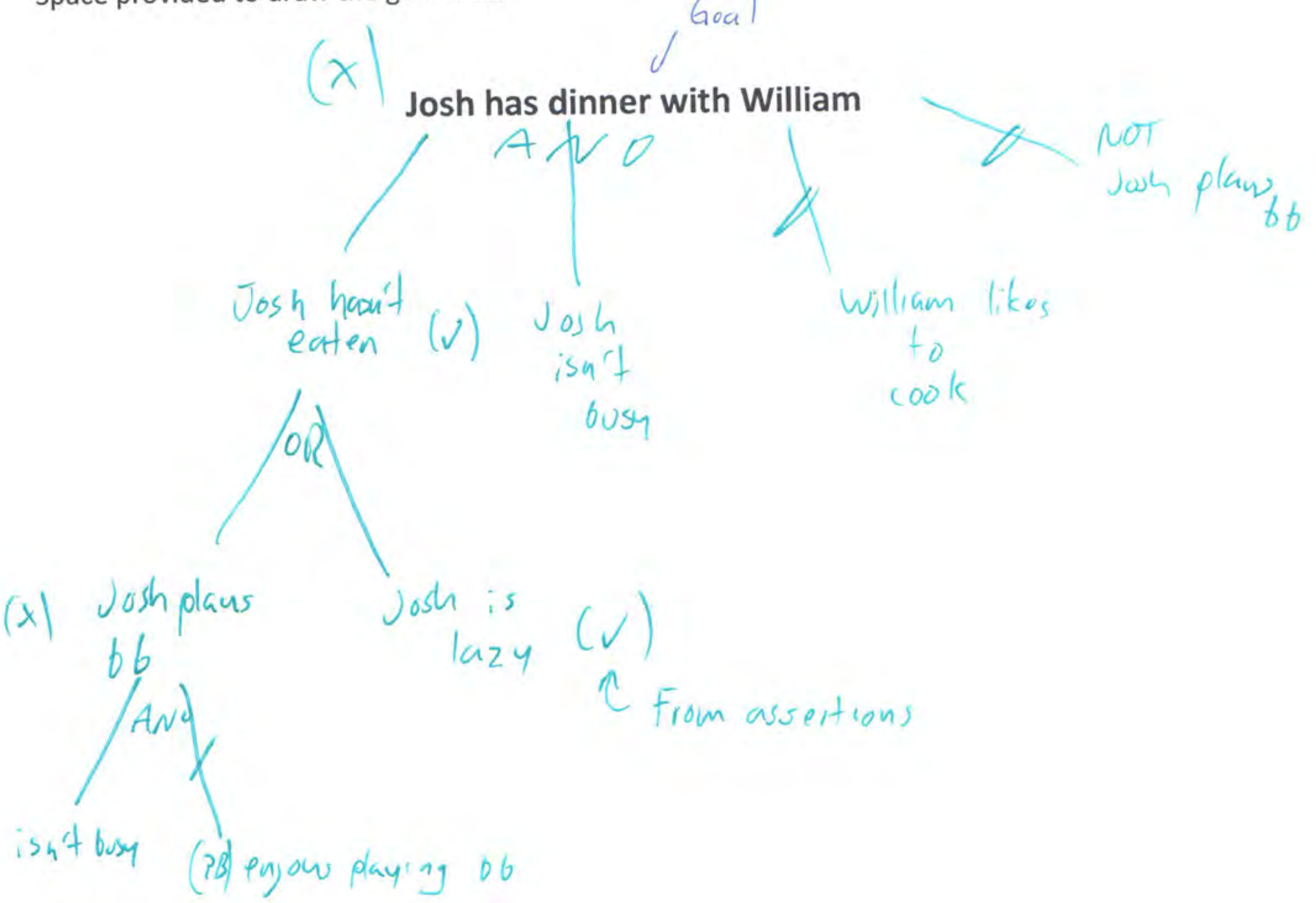
| | |
|---|---|
| 1 *Josh has dinner with William* | 9 |
| 2 Josh hasn't eaten | 10 |
| 3 Josh plays basketball | 11 |
| 4 Josh isnt busy | 12 |
| 5 Josh is lazy | 13 |
| 6 Josh isn't busy | 14 |
| 7 | 15 |
| 8 | 16 |

### A2 (3 points)

Does Josh have dinner with William? (Does backward chaining prove the hypothesis "Josh has dinner with William"?) Circle one:

YES NO

Space provided to draw the goal tree.




1) See what rules can produce Goal or check assertions and add conditions to tree. IF more than one rule matches add to tree with or node.

2) Set new goal and recurse.
   Short circuit: AND subgoal is False
   OR subgoal is true

## Part B: Forward chaining (23 points)

Make the following assumptions about forward chaining:

- Assume rule-ordering conflict resolution
- New assertions are added to the bottom of the list of assertions.
- If a particular rule matches assertions in the list of assertions in more than one way, the matches are considered in the order corresponding to the top-to-bottom order of the matched assertions. Thus, if a particular rule has an antecedent that matches both A1 and A2, the match with A1 is considered first.

### B1 (20 points)

Patrick, Josh's new roommate, has just moved into town. Josh quickly adds a new assertion to the list to account for his first impression of this unknown character.

rules that could be true using what we know

For each iteration

**Assertions for Part B** (forward chaining):

1) Go through all rules and mark matches
2) Fire first matched rule that you can
3) Repeat

Then statement not in assertion

A0: Josh is lazy
A1: Mother Aryan enjoys playing basketball
A2: William likes to cook
A3: Louis likes to code
**A4: Patrick isn't busy**

Perform forward chaining using the rules mentioned on page 3 and <u>this new list</u> of assertions. For the first two iterations, fill out the first two rows in the table below, noting the rules whose antecedents match the data, the rule that fires, and the new assertions that are added by the rule. For the remainder, supply only the fired rules and new assertions.

There may be more rows in the table than you'll need.

| | Matched | Fired | New Assertions Added to the List of Assertions |
|---|---|---|---|
| 1 | P0, P1, P2, P3 | P0 | Josh hasn't eaten |
| 2 | P0, P1, P2, P3 | P1 | Lois is working |
| 3 | | P2 | Patrick Plays basketball |
| 4 | | P0 | Patrick hasn't eaten |
| 5 | | P3 | Patrick will get his id |
| 6 | | | |
| 7 | | | |
| 8 | | | |

### B2 (3 points)

Given the assertions in part B1, which of the rules (P0, P1, P2, P3, P4) will never fire during forward chaining? (If all of the rules will fire, write NONE instead.)

P4

# Problem 2: Optimal Search (60 points)

## Part A: Search me (9 points)

**A1.** You have a summer internship with a travel agency. Your first task is to find an automobile route from Boston to San Francisco, but not just any route: you have to find the *most scenic* route.

- The road connecting any two cities has a scenic value, so the most scenic route is the one whose sum of city-to-city scenic values is **maximum**. NP hard
- There is no limit to how scenic a road connecting two cities can be.
- Your customers want to feel like they are making progress, so your route must include only connections traveled from east to west.
- You are to work with a map that includes only major cities and the roads connecting them, so you are not worried about any exponential blowup that could make certain search methods computationally infeasible, but of course you would like your search method to be as efficient as possible.

Of the following, your best choice is (circle the <u>**single best**</u> answer):

1 British Museum algorithm ← Need to look at all paths
2 Hill climbing
3 Bi-directional breadth-first search.
4 Bi-directional depth-first search.
5 Branch and bound search

**Note:** a *bi-directional search* expands one path starting from S, then one from G, then one from S, and so on, until a terminal node on a path from S is the same as a terminal node on a path from G.

**A2.** Your second task is to find an automobile route from point S to point G. This time, you want to find the route that goes through the fewest cities. Of course, you would like to be as efficient as possible.

Of the following, your best choice is (circle the <u>**single best**</u> answer):

1 British Museum algorithm
2 Hill climbing
3 Bi-directional breadth-first search. Ignores values but minimizes path length
4 Bi-directional depth-first search.
5 Branch and bound search

**A3.** Completely exhausted after your first, hard day at the travel company, which is in Paris, you have to walk to your apartment because the taxis are on a one-day strike. You are not sure how to get there, and you have no map, but your apartment is near the Eiffel tower, visible everywhere in the city.

Of the following, your best choice is (circle the **single best** answer):

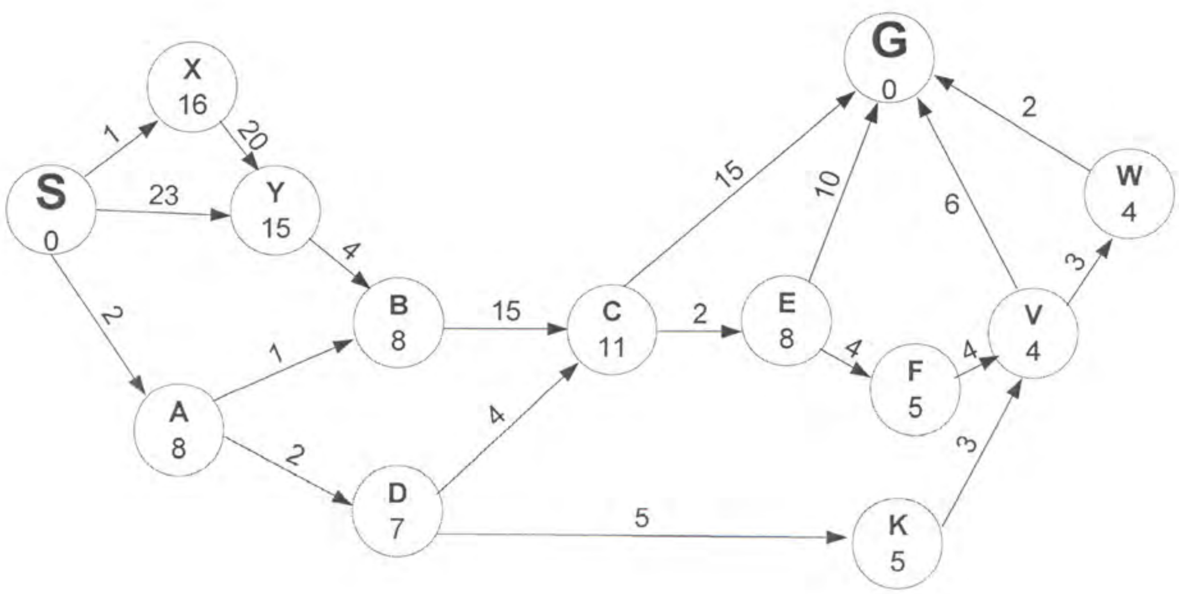
1 British Museum algorithm
2 Hill climbing
3 Bi-directional breadth-first search.
4 Bi-directional depth-first search.
5 Branch and bound search

## Part B: So busted! (17 points)

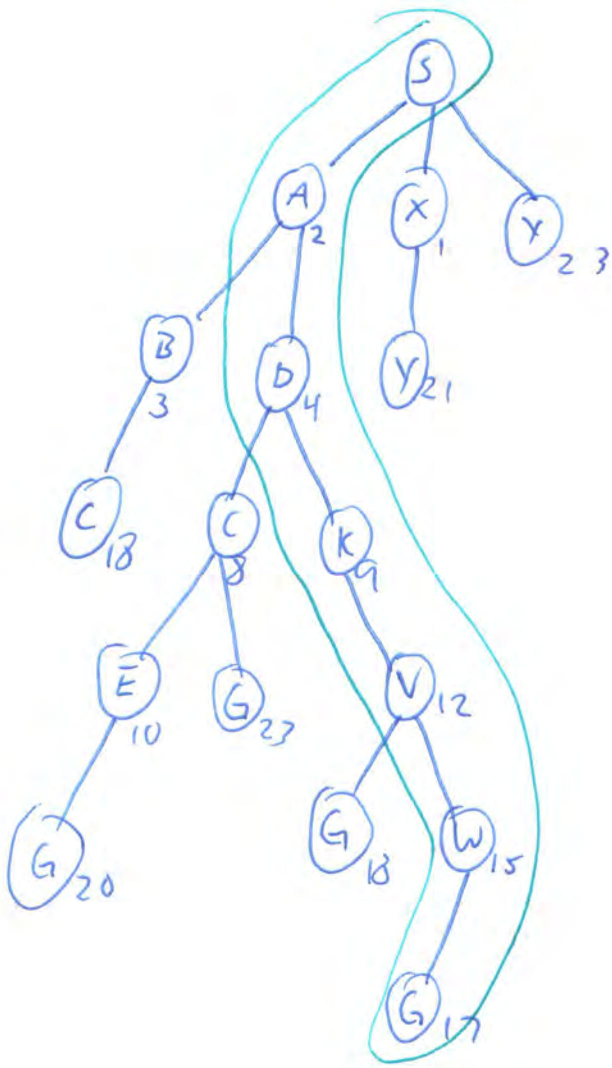
**B1 (14 points)**

Long-suffering teen Candace has just discovered her siblings goofing off and must race home to tell their mother. She decides to use **branch and bound, with no heuristic and no extended set**, to find the quickest route home starting from node **S** and ending at node **G**.

Referring to the graph below, use branch and bound (with no heuristic and no extended set) to find a route from node S to node G. The numbers inside each node indicate heuristic estimates of distance to the goal, which you will not need in this part. For credit, **draw your search tree** on the following page.



- **Draw the children of each node in alphabetical order**, and **break ties using lexicographic/dictionary order** (for example, if two paths S-P-U and S-Q-T are tied, you should extend path S-P-U before path S-Q-T).
- **Note #1:** For your convenience, a copy of the graph is provided on a tear-off sheet after the last page of the quiz.
- **Note #2:** Notice that the edges in this graph are **one-way**; paths must only go in the direction of the arrows.




- For path with shortest cost
- expand end node
  - calculate new path costs
- Repeat until G expanded

**B2 (3 points)**
What path does Candace find using branch and bound?

SADKVWG

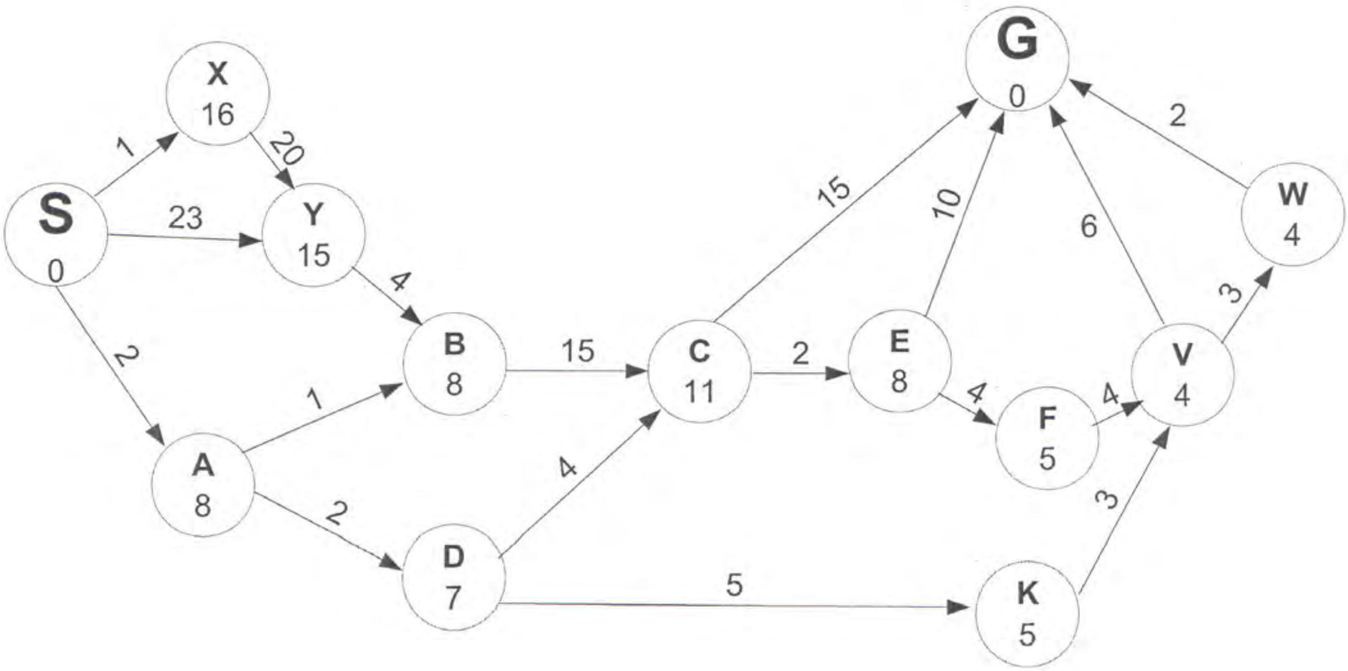
What is its total path length?

17

## Part C: Scram! (25 points)

Not eager to get scolded by their mother, Candace's siblings decide to look for a way home themselves—so, they ask an ostrich herder for heuristic estimates of the distance home from each node, then decide to use **A* search** (that is: branch and bound, with a heuristic and with an extended set) to look for a way home. The numbers inside each node indicate heuristic estimates of distance to the goal, which you will need in this part.

*(Here is a copy of the graph that was used in part B)*

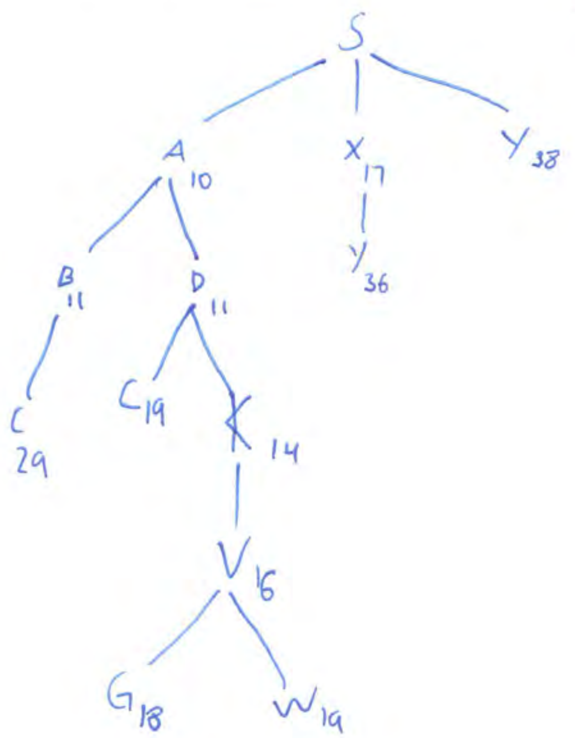


**C1 (14 points)**

Use A* search to find a route from node S to node G. For credit, **draw your search tree** below.

- **Draw the children of each node in alphabetical order**, and **break ties using lexicographic/dictionary order** (for example, if two paths S-P-U and S-Q-T are tied, you should extend path S-P-U before path S-Q-T).
- **Note #1:** For your convenience, a copy of the graph is provided on a tear-off sheet after the last page of the quiz.
- **Note #2:** Notice that the edges in this graph are **one-way**; paths must only go in the direction of the arrows.

· Path cost + heuristic of end node




**C2 (6 points)**

List the extended nodes in the order you extended them during A* search.

SABDKVXG

**Important!** The quiz continues on the next page. ↪

**C3 (3 points)**
What path do the siblings find using A* search?

SADKVG

What is its total path length?

18

**C4 (2 points)**
Regardless of your answers to parts C1 and C2, assume that branch and bound found a shorter path than A*. Very briefly, explain this result.

Inconsistent + inadmissive heuristic
(w is overestimate

# 6.034 Quiz 2

## 15 October 2014

| Name | Fernando Trujano PRACTICE |
|---|---|
| Email | |

Circle your TA **(for 1 extra credit point)**, so that we can more easily enter your score in our records and return your quiz to you promptly.

| | | |
|---|---|---|
| **Josh Blum** | **Elisa Castañer** | **Pedro Cattori** |
| **Jack Florey** | **Malcom Gilbert** | **Dylan Holmes** |
| **Jessica Noss** | **Duncan Townsend** | **Siyao Xu** |

| Problem number | Maximum | Score | Grader |
|---|---|---|---|
| 1 | 50 | | |
| 2 | 50 | | |
| Total | 100 | | |

| | | | |
|---|---|---|---|
| SRN | 6 | | |

There are 15 pages in this quiz, including this one. As always, open book, open notes, open just about everything, including a calculator, but no computers.

# Problem 1: Identification of Trees (50 points)

## Part A: Identification Trees (28 points)

While walking along the Charles River in the fall, your friend points out six trees and tells you whether they are Maple or Oak trees. Eager to identify other trees yourself, you observe what appear to be their key features. In particular, you record whether each tree **Has leaves (Yes or No)**, whether it has **Orange foliage (Yes or No)**, its **Leaf shape (Pointy or Rounded)**, and its **Bark texture (Glossy, Furrowed, or Smooth)**. Your resulting training data is summarized in the table below:

| Tree type | Has leaves* | Orange foliage | Leaf shape | Bark texture |
|---|---|---|---|---|
| Oak | Yes | Yes | Pointy | Glossy |
| Oak | Yes | No | Pointy | Furrowed |
| Oak | Yes | No | Rounded | Furrowed |
| Maple | Yes | Yes | Pointy | Furrowed |
| Maple | Yes | Yes | Pointy | Smooth |
| Maple | Yes | No | Pointy | Smooth |

*Although some trees lose their leaves in the fall, all six of these trees still have leaves.

**A1 (4 points)** Based on these six training points, order the four feature tests (**Has leaves**, **Orange foliage**, **Leaf shape**, and **Bark texture**) from least to greatest disorder. (You don't need to show your calculations; you may use your intuition. If you prefer not to use your intuition, see part A2 below.) Break any ties alphabetically.

| Bark Texture | Leaf shape | Orange Foilage | Has leaves |
|---|---|---|---|
| LEAST DISORDER | | | GREATEST DISORDER |

**A2 (8 points)** Next, compute the disorder of each feature test. Use the table of logarithms below to express your answer as sums and products of decimals and fractions only. Your final answer should have <u>**no logarithms**</u> in it. Space is provided on the next page to show your work.

| Test | Has leaves | Orange foliage | Leaf shape | Bark texture |
|---|---|---|---|---|
| **Disorder** | 1 | .9 | .8 | .45 |

$$-\left[\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right] = 1 \qquad -\left[\frac{2}{5}\log_2\frac{2}{5} + \frac{3}{5}\log_2\frac{3}{5}\right] \approx 0.97 \qquad -\left[\frac{1}{3}\log_2\frac{1}{3} + \frac{2}{3}\log_2\frac{2}{3}\right] \approx 0.9$$

$$-\left[\frac{1}{4}\log_2\frac{1}{4} + \frac{3}{4}\log_2\frac{3}{4}\right] \approx 0.8 \qquad -\left[\frac{1}{5}\log_2\frac{1}{5} + \frac{4}{5}\log_2\frac{4}{5}\right] \approx 0.72 \qquad -\left[\frac{1}{6}\log_2\frac{1}{6} + \frac{5}{6}\log_2\frac{5}{6}\right] \approx 0.65$$

Show your disorder calculations for partial credit. For your convenience, a copy of the training data is provided on the next page.

$$d = -1 \sum_c \frac{\#\text{pts } c}{\#\text{pts branch}} \log_2\left(\frac{\#\text{pts } c}{\#\text{pts } b}\right)$$

$$D = \sum_b \frac{\#\text{pts } b}{\#\text{totalpts}} \cdot d(b)$$

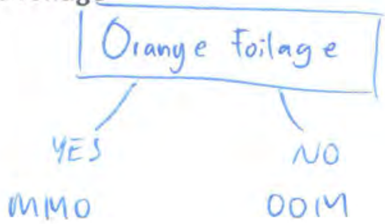
**Has leaves**


$$d_{YES} = -\frac{3}{6}\log_2\left(\frac{3}{6}\right) + \frac{3}{6}\log_2\left(\frac{3}{6}\right) = 1$$

$$d_{NO} = 0\log_2(0) + 0\log_2(0) = 0$$

$$D = \frac{6}{6} d_{YES} + \frac{0}{6} d_{NO} = 1 \qquad \boxed{D = 1}$$

**Orange foliage**



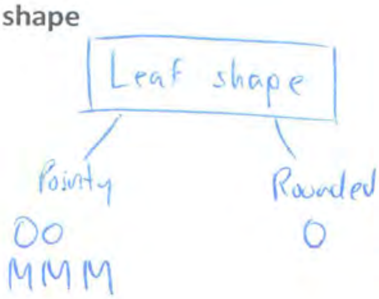

$$d_{YES} = -\frac{2}{3}\log_2\frac{2}{3} + \frac{1}{3}\log_2\frac{1}{3} \approx .9$$

$$d_{NO} = \quad '' \qquad \approx .9$$

$$D = \frac{3}{6}(.9) + \frac{3}{6}(.9) = .9$$

$$\boxed{D = .9}$$

**Leaf shape**

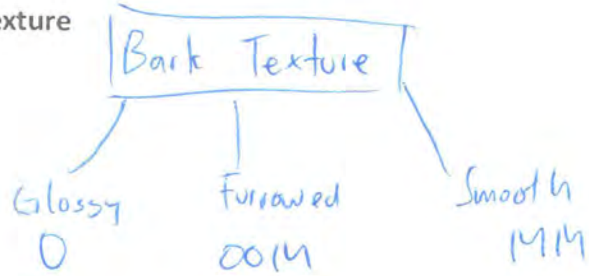



$$d_{Pointy} = -\frac{2}{5}\log_2\left(\frac{2}{5}\right) + \frac{3}{5}\log_2\left(\frac{3}{5}\right) \approx .97$$

$$d_{Roundy} = -\frac{1}{1}\log_2\left(\frac{1}{1}\right) + \frac{0}{1}\log_2\left(\frac{0}{1}\right) = 0$$

$$D = \left(\frac{5}{6}\right) d_{Pointy} + \left(\frac{1}{6}\right) d_{Round} = .8 \qquad \boxed{D = .8}$$

**Bark texture**




$$d_G = 0$$

$$d_F = .9$$

$$d_S = 0$$

$$D = \frac{1}{6} d_G + \frac{3}{6} d_F + \frac{2}{6} d_S = .45$$

$$\boxed{D = .45}$$

**A3 (14 points)**

Choosing from the four feature tests (**Has leaves**, **Orange foliage**, **Leaf shape**, and **Bark texture**), construct the complete disorder-minimizing identification tree for classifying the training data as Maple or Oak.

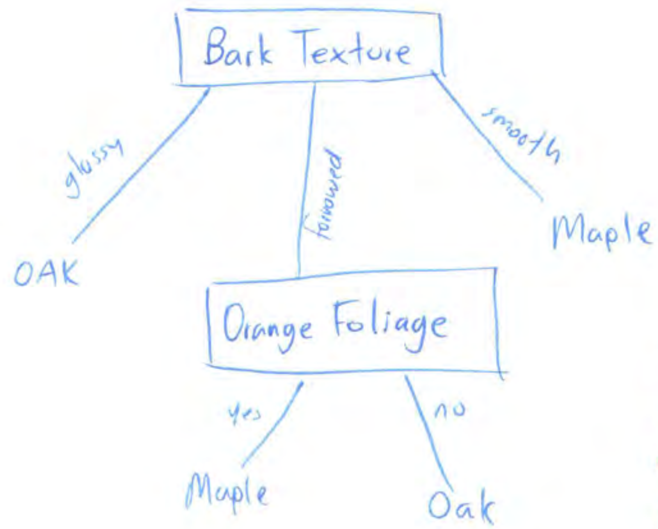
(For your convenience, a copy of the training data is included here.)

| Tree type | Has leaves* | Orange foliage | Leaf shape | Bark texture |
|---|---|---|---|---|
| Oak | Yes | Yes | Pointy | Glossy |
| Oak | Yes | No | Pointy | Furrowed |
| Oak | Yes | No | Rounded | Furrowed |
| Maple | Yes | Yes | Pointy | Furrowed |
| Maple | Yes | Yes | Pointy | Smooth |
| Maple | Yes | No | Pointy | Smooth |

*Although some trees lose their leaves in the fall, all six of these trees still have leaves.

**Draw your identification tree** in the space below:

1) Pick test w/ least disorder
2) Recalculate disorder for tests using remaining points




← Test with least disorder

Only consider Furrowed and recalculate

Has leaves: YES: OOM; NO

$d_{YES} = -\frac{2}{3}\log\frac{2}{3} + \frac{1}{3}\log\frac{1}{3} = .9$

$d_{NO} = 0$

$D = .9$

Orange Foliage: NO: OO; YES: M

$d_{NO} = 0$

$d_{YES} = 0$

$D = 0$

Leaf shape: Pointy: OM; Rounded: O

$d_P = -\frac{1}{2}\log\frac{1}{2} + \frac{1}{2}\log\frac{1}{2} = 1$

$d_P = 0$

$D = 2/3$

Bark Texture ← Already used

furrowed: OOM; glossy: X; smooth: X

$D = .9$

Additional space provided for disorder calculations, if you need it.

**A4 (2 points)**

How would your identification tree classify a specimen with the following features? Just follow branches of id Tree

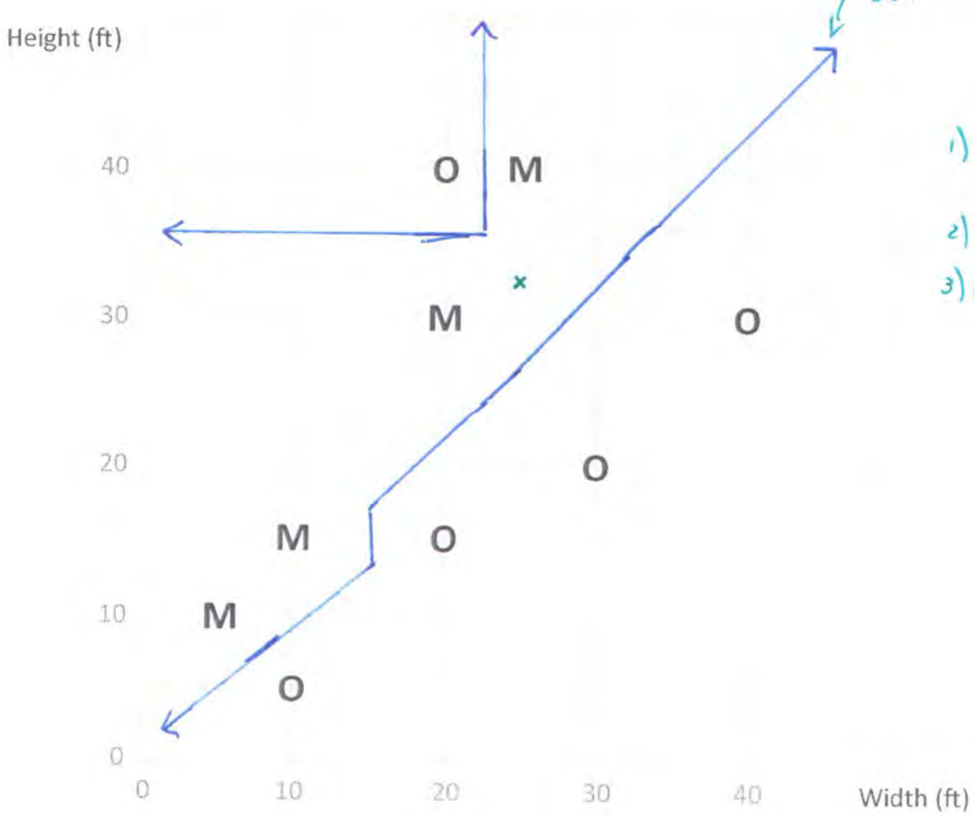
| Has leaves | Orange foliage | Leaf shape | Bark texture |
|---|---|---|---|
| Yes | Yes | Rounded | Furrowed |

Circle one: MAPLE OAK CAN'T TELL

## Part B: Nearest neighbors (22 points)

You plan to go out and identify more trees the next day, but that night there is an early snowstorm! To your dismay, you find that a blanket of snow has obscured the leaves and bark of all the trees, so you can no longer use those as distinguishing features. Undaunted, you decide to venture out into the fresh snow and use trigonometry to measure the width and height of trees that are known to be Maple (**M**) or Oak (**O**). Your measurements are plotted on the following graph:

don't forget the arrows




1) Find closest pair of different types
2) Draw perpendicular bisecto[r]
3) Connect lines

**B1 (12 points)** On the graph above, **<u>draw the decision boundaries</u>** produced by 1-Nearest Neighbors.

**B2 (8 points)** One of your classmates hears that you've been identifying trees. She wants to know what type of tree is knocking snow onto her second-story windowsill, so she calls you with the information that her tree is **25 ft wide** and **32 ft tall**. How would this tree be classified with each of the following classifiers? (Circle the best answer in each case.)

| | | | |
|---|---|---|---|
| 1-Nearest Neighbors? | **MAPLE** | **OAK** | **CAN'T TELL** |
| 3-Nearest Neighbors? | **MAPLE** | **OAK** | **CAN'T TELL** |
| 5-Nearest Neighbors? | **MAPLE** | **OAK** | **CAN'T TELL** |
| 9-Nearest Neighbors? | **MAPLE** | **OAK** | **CAN'T TELL** |

Consider # neighbors close to point × and classyfy as most repeated neighbor.

**B3 (2 points)** Finally, you want to use Nearest Neighbors to classify the pre-snowstorm data in Part A. You realize it's possible, but you must first: (Circle the **one** best answer.)

(A) Pick just two of the features to plot.

(B) Define a distance metric for non-numeric features.

Need to decide what "near" is for non-numeric features in Part A

(C) Collect twice as many training points.

(D) Use cross-validation to guarantee overfitting.

(E) Devise an admissible heuristic.

for the six mages.

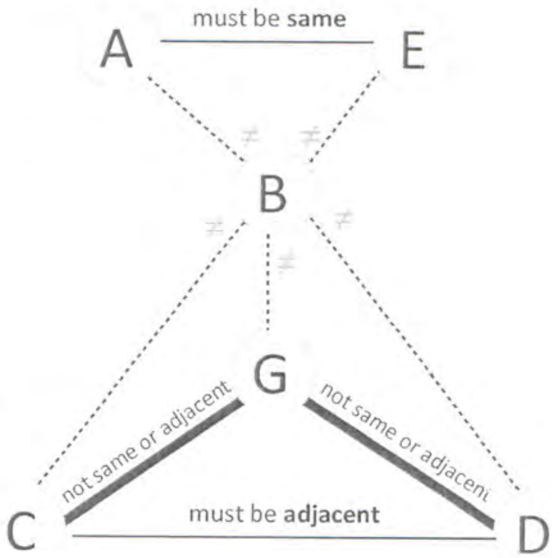
## Problem 2: Dominian Domiciles (50 points)

Six powerful mages (**A**jani, **B**olas, **C**handra, **D**ack, **E**lspeth, and **G**ideon) have enlisted you to help them settle down and find homes in the multiverse. They have five neighboring worlds to choose from, arranged in a row as shown. Some worlds could be empty, and some worlds could have more than one occupant.

| | | | | |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |

Naturally, mages are very particular about where they live, so your assignments must satisfy the following requirements:

1. **A**jani is **E**lspeth's mentor. They must live on the <u>**same world**</u>.
2. Everyone hates **B**olas. No one can live on the same world as him.
3. **C**handra wants to spy on **D**ack secretly—she must live on a world that's **adjacent to** **D**ack's world.
4. **G**ideon is a law mage. He despises the criminals **C**handra and **D**ack so much that **G**ideon **cannot live on the same world—or even on a world adjacent to** **C**handra's or **D**ack's.




(For your convenience, a graph of these constraints is shown here.)

Finally, the mages inform you that you can simplify the domains of these variables, as follows:

- **D**ack **must live on either World 1 or 4**, because he's a wanted criminal everywhere else.
- **E**lspeth **must live on World 3**, because she's stuck there and cannot leave.
- **B**olas **can't live on World 5**, because he fears its strange magic.

Thus, the initial domains of your variables are as follows:

| | | | | | |
|---|---|---|---|---|---|
| A | 1 | 2 | 3 | 4 | 5 |
| B | 1 | 2 | 3 | 4 | |
| C | 1 | 2 | 3 | 4 | 5 |
| D | 1 | | | 4 | |
| E | | | 3 | | |
| G | 1 | 2 | 3 | 4 | 5 |