Department of Mathematics and Computer Science
University of Southern Denmark, Odense

December 27, 2009
Marco Chiarandini

DM533 - Introduction to Artificial Intelligence

Exercise Collection, Fall 2009

Exercise: Naive Bayesian Networks

1. Suppose that we collected the data in Table 1 concerning customers that buy or not a computer. The data samples are described by attributes *age*, *income*, *student*, and *credit*. The class label attribute, *buy*, that tells whether the person buys a computer, has two distinct values, *yes* and *no*. Considering these data as a training set for a naive Baysian classifier, classify the sample

$$\mathbf{X} = (age = youth, income = medium, student = yes, credit = fair)$$

RID	age	income	student	credit	C_i: buy
1	youth	high	no	fair	C_2: no
2	youth	high	no	excellent	C_2: no
3	middle-aged	high	no	fair	C_1: yes
4	senior	medium	no	fair	C_1: yes
5	senior	low	yes	fair	C_1: yes
6	senior	low	yes	excellent	C_2: no
7	middle-aged	low	yes	excellent	C_1: yes
8	youth	medium	no	fair	C_2: no
9	youth	low	yes	fair	C_1: yes
10	senior	medium	yes	fair	C_1: yes
11	youth	medium	yes	excellent	C_1: yes
12	middle-aged	medium	no	excellent	C_1: yes
13	middle-aged	high	yes	fair	C_1: yes
14	senior	medium	no	excellent	C_2: no

Solution

We need to maximize $\Pr(\mathbf{X}|C_i)\Pr(C_i)$, for $i = 1, 2$. $\Pr(C_i)$, the a priori probability of each class, can be estimated based on the training samples:

$$\Pr(buy = yes) = 9/14$$
$$\Pr(buy = no) = 5/14$$

To compute $\Pr(\mathbf{X}|C_i)$, for $i = 1, 2$, we recall that

$$\Pr(\mathbf{X}|C_i) \approx \prod_{k=1}^{n} \Pr(x_k|C_i)$$

we compute the following conditional probabilities:

$$\Pr(age = youth|buy = yes) = 2/9$$
$$\Pr(age = youth|buy = no) = 3/5$$
$$\Pr(income = medium|buy = yes) = 4/9$$

$$\Pr(income = medium|buy = no) = 2/5$$
$$\Pr(student = yes|buy = yes) = 6/9$$
$$\Pr(student = yes|buy = no) = 1/5$$
$$\Pr(credit = fair|buy = yes) = 6/9$$
$$\Pr(credit = fair|buy = no) = 2/5$$

Using the above probabilities, we obtain

$$\begin{aligned}\Pr(X|buy = yes) \quad = \quad & \Pr(age = youth|buy = yes) \\ & \Pr(income = medium|buy = yes) \\ & \Pr(student = yes|buy = yes) \\ & \Pr(credit = fair|buy = yes) \\ & = \tfrac{2}{9}\tfrac{4}{9}\tfrac{6}{9}\tfrac{6}{9} = 0.044.\end{aligned}$$

Similarly,

$$\Pr(X|buy = no) = \frac{3}{5}\frac{2}{5}\frac{1}{5}\frac{2}{5} = 0.019$$

To find the class that maximizes

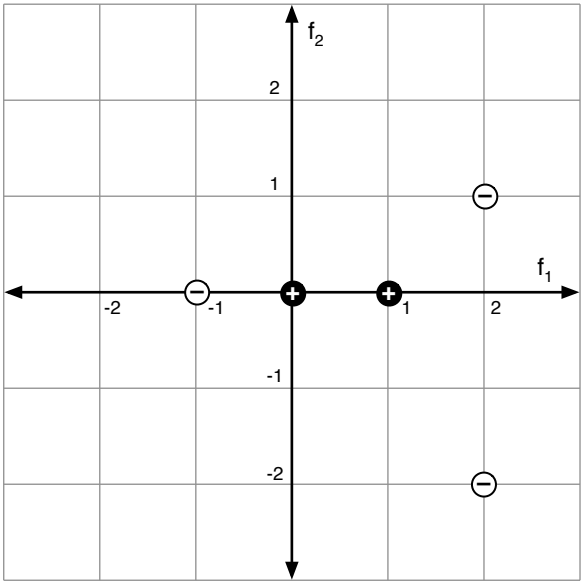
$$\Pr(X|C_i)\Pr(C_i),$$

we compute

$$\Pr(X|buy = yes)\Pr(buy = yes) = 0.028$$
$$\Pr(X|buy = no)\Pr(buy = no) = 0.007$$

Thus the naive Bayesian classifier predicts $buy = yes$ for sample $\mathbf{X}$.

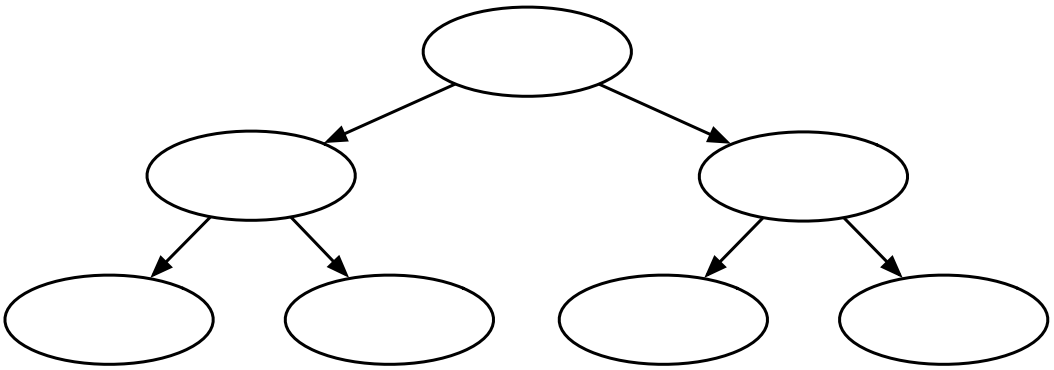
1 Decision Trees



Data points are: Negative: (-1, 0) (2, 1) (2, -2) Positive: (0, 0) (1, 0)

Construct a decision tree using the algorithm described in the notes for the data above.

1. Show the tree you constructed in the diagram below. The diagram is more than big enough, leave any parts that you don't need blank.



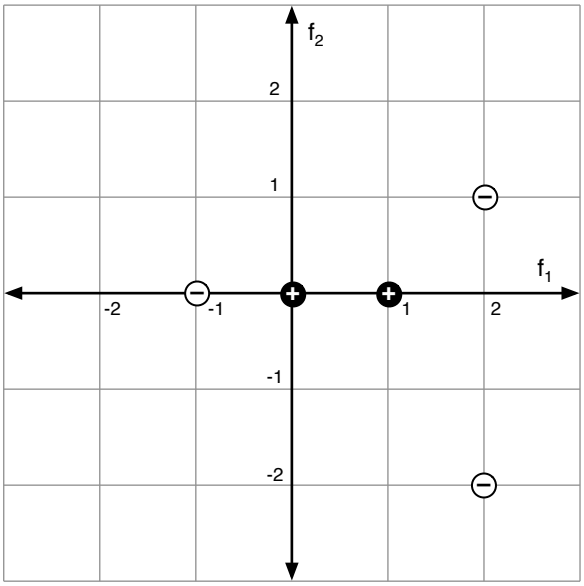
2. Draw the decision boundaries on the graph at the top of the page.

3. Explain how you chose the top-level test in the tree. The following table may be useful.

x	y	-(x/y)*lg(x/y)		x	y	-(x/y)*lg(x/y)
1	2	0.50		1	5	0.46
1	3	0.53		2	5	0.53
2	3	0.39		3	5	0.44
1	4	0.50		4	5	0.26
3	4	0.31				

4. What class does the decision tree predict for the new point: (1, -1.01)

2 Nearest Neighbors



Data points are: Negative: (-1, 0) (2, 1) (2, -2) Positive: (0, 0) (1, 0)

1. Draw the decision boundaries for 1-Nearest Neighbors on the graph above. Your drawing should be accurate enough so that we can tell whether the integer-valued coordinate points in the diagram are on the boundary or, if not, which region they are in.
2. What class does 1-NN predict for the new point: (1, -1.01) Explain why.
3. What class does 3-NN predict for the new point: (1, -1.01) Explain why.

6 Learning algorithms

For each of the learning situations below, say what learning algorithm would be best to use, and why.

1. You have about 1 million training examples in a 6-dimensional feature space. You only expect to be asked to classify 100 test examples.

2. You are going to develop a classifier to recommend which children should be assigned to special education classes in kindergarten. The classifier has to be justified to the board of education before it is implemented.

3. You are working for Am*z*n as it tries to take over the retailing world. You are trying to predict whether customer X will like a particular book, as a function of the input which is a vector of 1 million bits specifying whether each of Am*z*n's other customers liked the book. You will train a classifier on a very large data set of books, where the inputs are everyone else's preferences for that book, and the output is customer X's preference for that book. The classifier will have to be updated frequently and efficiently as new data comes in.

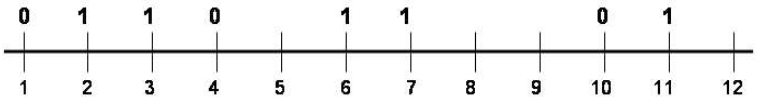
4. You are trying to predict the average rainfall in California as a function of the measured currents and tides in the Pacific ocean in the previous six months.

4 Machine Learning — Continuous Features (20 points)

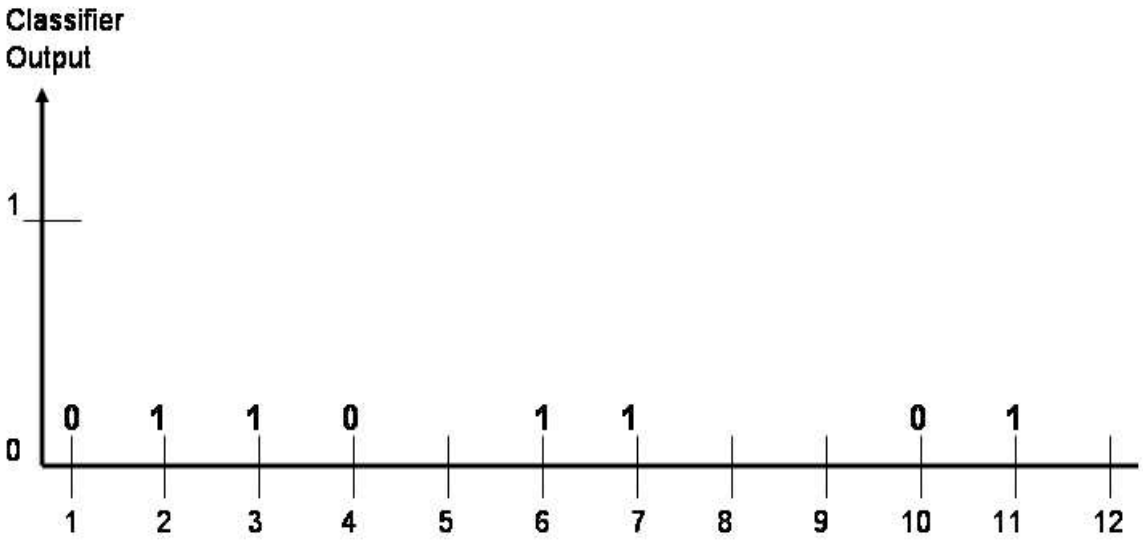
In all the parts of this problem we will be dealing with one-dimensional data, that is, a set of points (x^i) with only one feature (called simply x). The points are in two classes given by the value of y^i. We will show you the points on the x axis, labeled by their class values; we also give you a table of values.

4.1 Nearest Neighbors

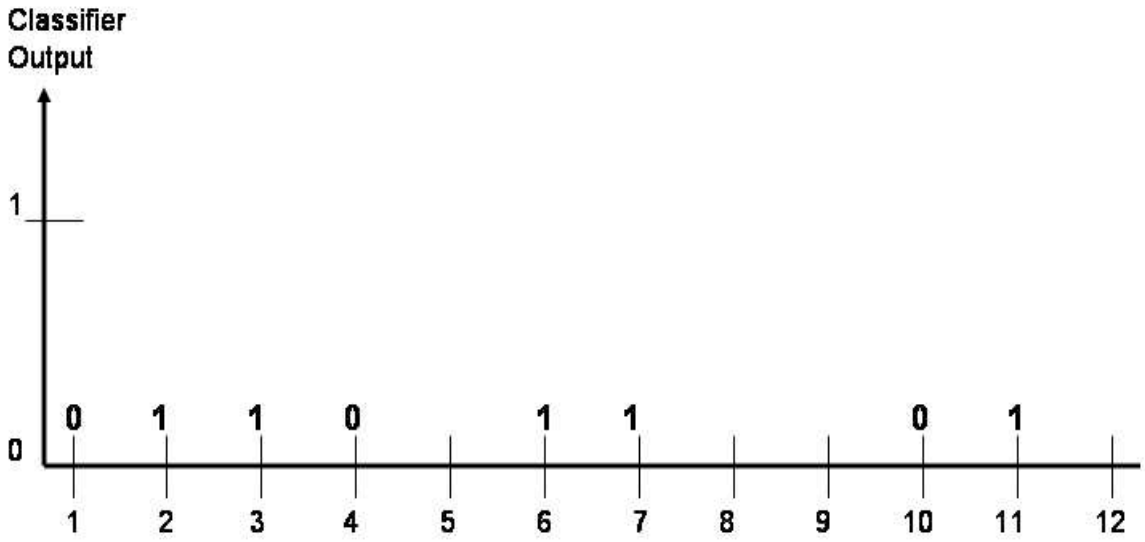
i	x^i	y^i
1	1	0
2	2	1
3	3	1
4	4	0
5	6	1
6	7	1
7	10	0
8	11	1



1. In the figure below, draw the output of a 1-Nearest-Neighbor classifier over the range indicated in the figure.

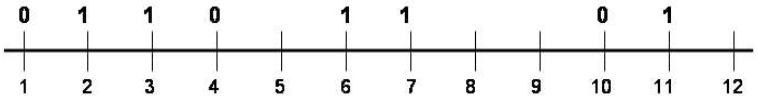


2. In the figure below, draw the output of a 5-Nearest-Neighbor classifier over the range indicated in the figure.



4.2 Decision Trees

Answer this problem using the same data as in the Nearest Neighbor problem above.



Which of the following three tests would be chosen as the top node in a decision tree?

$$x \leq 1.5 \qquad x \leq 5 \qquad x \leq 10.5$$

Justify your answer.

You may find this table useful.

x	y	-(x/y)*lg(x/y)		x	y	-(x/y)*lg(x/y)
1	2	0.50		1	8	0.38
1	3	0.53		3	8	0.53
2	3	0.39		5	8	0.42
1	4	0.50		7	8	0.17
3	4	0.31		1	9	0.35
1	5	0.46		2	9	0.48
2	5	0.53		4	9	0.52
3	5	0.44		5	9	0.47
4	5	0.26		7	9	0.28
1	6	0.43		8	9	0.15
2	6	0.53		1	10	0.33
5	6	0.22		3	10	0.52
1	7	0.40		7	10	0.36
2	7	0.52		9	10	0.14
3	7	0.52				
4	7	0.46				
5	7	0.35				
6	7	0.19				

6 Pruning Trees (20 points)

Following are some different strategies for pruning decision trees. We assume that we grow the decision tree until there is one or a small number of elements in each leaf. Then, we prune by deleting individual leaves of the tree until the score of the tree starts to get worse. The question is how to score each possible pruning of the tree.

For each possible definition of the score below, explain whether or not it would be a good idea and give a reason why or why not.

1. The score is the percentage correct of the tree on the training set.

2. The score is the percentage correct of the tree on a separate validation set.

3. The score is the percentage correct of the tree, computed using cross validation.

4. The score is the percentage correct of the tree, computed on the training set, minus a constant C times the number of nodes in the tree.

   C is chosen in advance by running this algorithm (grow a large tree then prune in order to maximize percent correct minus C times number of nodes) for many different values of C, and choosing the value of C that minimizes training-set error.

5. The score is the percentage correct of the tree, computed on the training set, minus a constant C times the number of nodes in the tree.

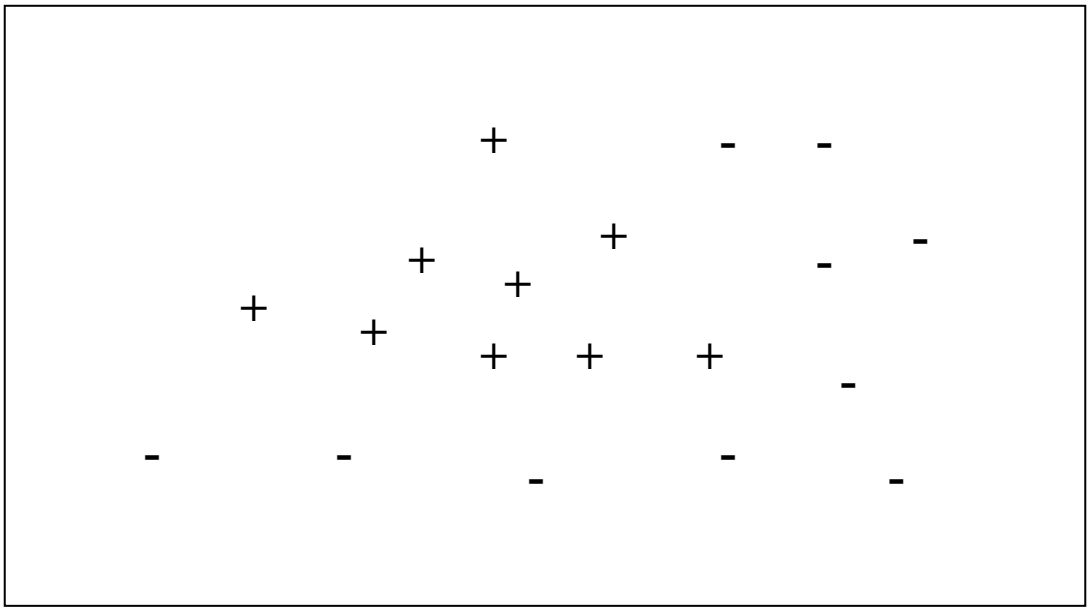
   C is chosen in advance by running cross-validation trials of this algorithm (grow a large tree then prune in order to maximize percent correct minus C times number of nodes) for many different values of C, and choosing the value of C that minimizes cross-validation error.

Problem 4: Learning (25 points)

Part A: (5 Points)

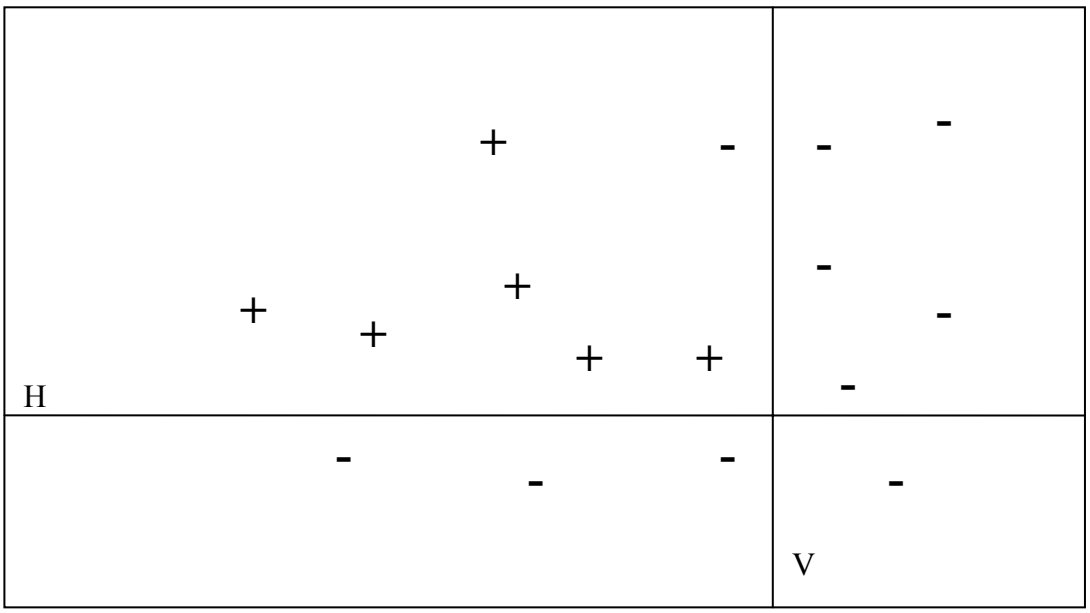
Since the cost of using a nearest neighbor classifier grows with the size of the training set, sometimes one tries to eliminate redundant points from the training set. These are points whose removal does not affect the behavior of the classifier for any possible new point.

1. In the figure below, sketch the decision boundary for a 1-nearest-neighbor rule and circle the redundant points.



2. What is the general condition(s) required for a point to be declared redundant for a 1-nearest-neighor rule? Assume we have only two classes (+, -). Restating the definition of redundant ("removing it does not change anything") is not an acceptable answer. Hint – think about the neighborhood of redundant points.

Part B: (5 Points)



Which of H or V would be preferred as an initial split for a decision (identification) tree? Justify your answer numerically.

x	y	-(x/y)*lg(x/y)
1	2	0.50
1	3	0.53
2	3	0.39
1	4	0.50
3	4	0.31
1	5	0.46
2	5	0.53
3	5	0.44
4	5	0.26
1	6	0.43
2	6	0.53
5	6	0.22
1	7	0.40
2	7	0.52
3	7	0.52
4	7	0.46
5	7	0.35
6	7	0.19

x	y	-(x/y)*lg(x/y)
1	8	0.38
3	8	0.53
5	8	0.42
7	8	0.17
1	9	0.35
2	9	0.48
4	9	0.52
5	9	0.47
7	9	0.28
8	9	0.15
1	10	0.33
3	10	0.52
7	10	0.36
9	10	0.14

Problem 2: Overfitting (20 points)

For each of the supervised learning methods that we have studied, indicate how the method could overfit the training data (consider both your design choices as well as the training) and what you can do to minimize this possibility. There may be more than one mechanism for overfitting, make sure that you identify them all.

Part A: Nearest Neighbors (5 Points)

1. How does it overfit?

2. How can you reduce overfitting?

Part B: Decision Trees (5 Points)

1. How does it overfit?

2. How can you reduce overfitting?

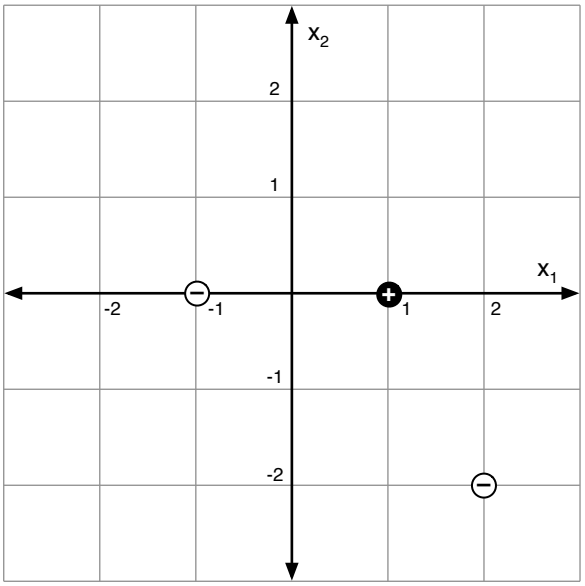
Problem 3: Spaminator (10 points)

Suppose that you want to build a program that detects whether an incoming e-mail message is spam or not. You decide to attack this using machine learning. So, you collect a large number of training messages and label them as spam or not-spam. You further decide that you will use the presence of individual words in the body of the message as features. That is, you collect every word found in the training set and assign to each one an index, from 1 to N. Then, given a message, you construct a feature vector with N entries and write in each entry a number that indicates how many times the word appears in that message.

Part A: (6 Points)

If you had to choose between a Nearest Neighbor implementation or an Decision Tree implementation, which would you choose? Justify your answer briefly both in terms of expected accuracy and efficiency of operation. Indicate the strength and weaknesses of each approach.

3 Perceptron (7 pts)



Data points are: Negative: (-1, 0) (2, -2) Positive: (1, 0). Assume that the points are examined in the order given here.

Recall that the perceptron algorithm uses the extended form of the data points in which a 1 is added as the 0th component.

1. The linear separator obtained by the standard perceptron algorithm (using a step size of 1.0 and a zero initial weight vector) is (0 1 2). Explain how this result was obtained.

2. What class does this linear classifier predict for the new point: (2.0, -1.01)

3. Imagine we apply the perceptron learning algorithm to the 5 point data set we used on Problem 1: Negative: (-1, 0) (2, 1) (2, -2), Positive: (0, 0) (1, 0). Describe qualitatively what the result would be.

6 Perceptron (8 points)

The following table shows a data set and the number of times each point is misclassified during a run of the perceptron algorithm, starting with zero weights. What is the equation of the separating line found by the algorithm, as a function of x_1, x_2, and x_3? Assume that the learning rate is 1 and the initial weights are all zero.

x_1	x_2	x_3	y	times misclassified
2	3	1	+1	12
2	4	0	+1	0
3	1	1	-1	3
1	1	0	-1	6
1	2	1	-1	11

Problem 2: Overfitting (20 points)

For each of the supervised learning methods that we have studied, indicate how the method could overfit the training data (consider both your design choices as well as the training) and what you can do to minimize this possibility. There may be more than one mechanism for overfitting, make sure that you identify them all.

Part A: Nearest Neighbors (5 Points)

1. How does it overfit?

2. How can you reduce overfitting?

Part B: Decision Trees (5 Points)

1. How does it overfit?

2. How can you reduce overfitting?

Part B: (4 Points)

Assume that you wanted to reduce the size of the feature vectors (during training and classification), for each of the approaches below indicate why it might be a good or bad idea.

1. Use only the words that appear in spam messages.

2. Eliminate words that are very common in the whole data set.

4 Search Problem formulation (23 points)

Consider a Mars rover that has to drive around the surface, collect rock samples, and return to the lander. We want to construct a plan for its exploration.

- It has batteries. The batteries can be charged by stopping and unfurling the solar collectors (pretend it's always daylight). One hour of solar collection results in one unit of battery charge. The batteries can hold a total of 10 units of charge.
- It can drive. It has a map at 10-meter resolution indicating how many units of battery charge and how much time (in hours) will be required to reach a suitable rock in each square.
- It can pick up a rock. This requires one unit of battery charge. The robot has a map at 10-meter resolution that indicates the type of rock expected in that location and the expected weight of rocks in that location. Assume only one type of rock and one size can be found in each square.

The objective for the rover is to get one of each of 10 types of rocks, within three days, while minimizing a combination of their total weight and the distance traveled. You are given a tradeoff parameter α that converts units of weight to units of distance. The rover starts at the lander with a full battery and must return to the lander.

Here is a list of variables that might be used to describe the rover's world:

- types of rocks already collected
- current rover location (square on map)
- current lander location (square on map)
- weight of rocks at current location (square on map)
- cost to traverse the current location (square on map)
- time since last charged
- time since departure from lander
- current day
- current battery charge level
- total battery capacity
- distance to lander
- total weight of currently collected rocks

1. Use a set of the variables above to describe the rover's state. Do not include extraneous information.

2. Specify the goal test.

3. Specify the actions. Indicate how they modify the state and any preconditions for being used.

4. Specify a function that determines the cost of each action.

5. This can be treated as a path search problem. We would like to find a heuristic. Say whether each of these possible heuristics would be useful in finding the optimal path or, if not, what's wrong with them. Let l be the number of rocks already collected.

   H1: The sum of the distances (in the map) from the rover to the $10 - l$ closest locations for the missing types of rocks.

   H2: The length of the shortest tour through the $10 - l$ closest locations for the missing types of rocks.

   H3: The distance back to the lander.

5 Search traces (21 points)

Consider the graph shown in the figure below. We can search it with a variety of different algorithms, resulting in different search trees. Each of the trees (labeled G1 though G7) was generated by searching this graph, but with a different algorithm. Assume that children of a node are visited in alphabetical order. Each tree shows all the nodes that have been visited. Numbers next to nodes indicate the relevant "score" used by the algorithm for those nodes.

For each tree, indicate whether it was generated with

1. Depth first search
2. Breadth first search
3. Uniform cost search
4. A* search
5. Best-first (greedy) search

In all cases a strict expanded list was used. Furthermore, if you choose an algorithm that uses a heuristic function, say whether we used

H1: heuristic 1 = $\{h(A) = 3, h(B) = 6, h(C) = 4, h(D) = 3\}$

H2: heuristic 2 = $\{h(A) = 3, h(B) = 3, h(C) = 0, h(D) = 2\}$

Also, for all algorithms, say whether the result was an optimal path (measured by sum of link costs), and if not, why not. Be specific.

Write your answers in the space provided below (not on the figure).

G1: 1. Algorithm:
2. Heuristic (if any):
3. Did it find least-cost path? If not, why?

G2: 1. Algorithm:
2. Heuristic (if any):
3. Did it find least-cost path? If not, why?

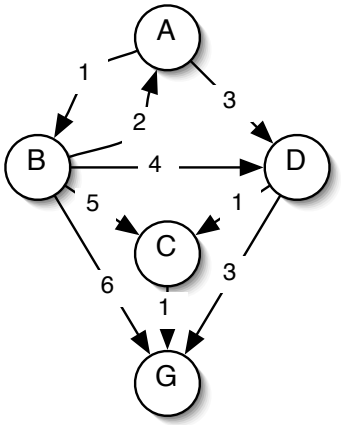
G3: 1. Algorithm:
2. Heuristic (if any):
3. Did it find least-cost path? If not, why?

G4: 1. Algorithm:
2. Heuristic (if any):
3. Did it find least-cost path? If not, why?

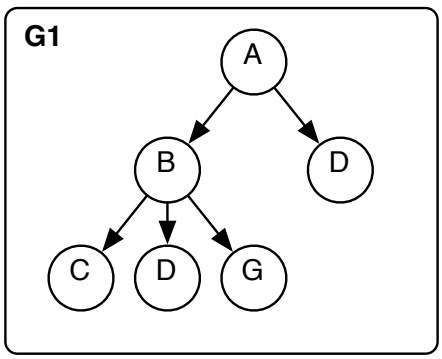
G5: 1. Algorithm:
2. Heuristic (if any):
3. Did it find least-cost path? If not, why?

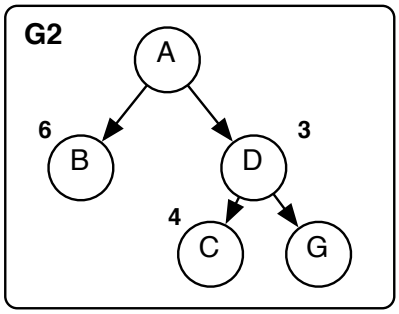
G6: 1. Algorithm:
2. Heuristic (if any):
3. Did it find least-cost path? If not, why?

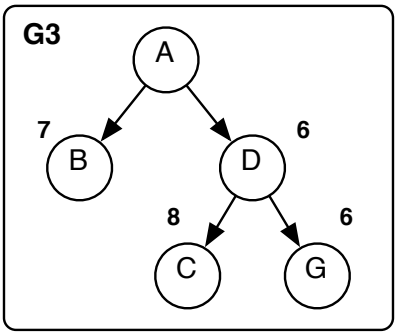
G7: 1. Algorithm:
2. Heuristic (if any):
3. Did it find least-cost path? If not, why?

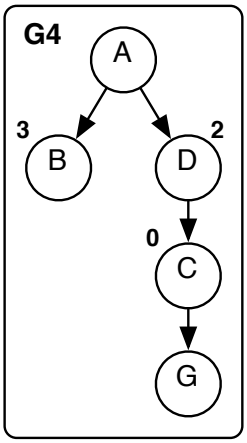


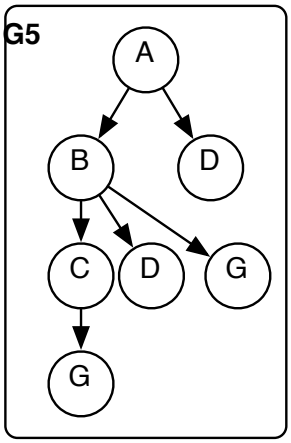
	H1	H2
A	3	3
B	6	3
C	4	0
D	3	2

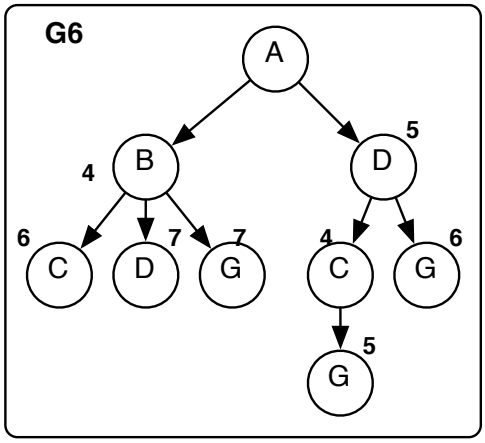


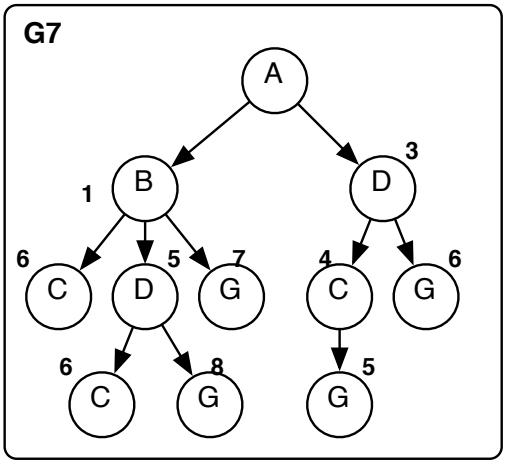












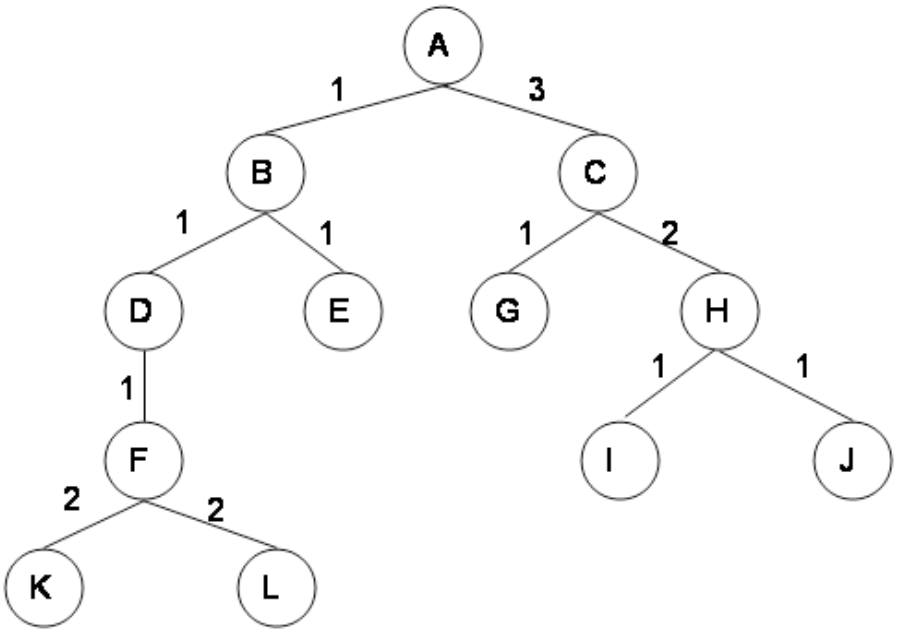
1.2 Algorithms

1. You are faced with a path search problem with a very large branching factor, but where the answers always involve a relative short sequence of actions (whose exact length is unknown). All the actions have the same cost. What search algorithm would you use to find the optimal answer? Indicate under what conditions, if any, a visited or expanded list would be a good idea.

2. You are faced with a path search problem with a very large branching factor, but where the answers always involve a relative short sequence of actions (whose exact length is unknown). These actions, however, have widely varying costs. What search algorithm would you use to find the optimal answer? Indicate under what conditions, if any, a visited or expanded list would be a good idea.

1 Tree Search (12 points)

Consider the tree shown below. The numbers on the arcs are the arc lengths.

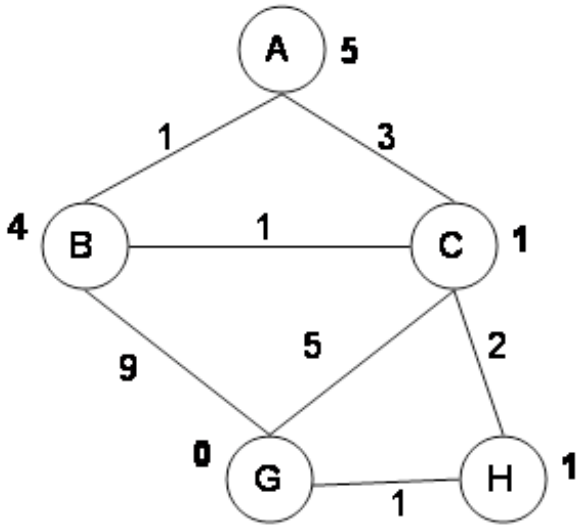


Assume that the nodes are expanded in alphabetical order when no other order is specified by the search, and that the goal is state G. No visited or expanded lists are used. What order would the states be expanded by each type of search? Stop when you expand G. Write only the sequence of states expanded by each search.

Search Type	List of states
Breadth First	
Depth First	
Progressive Deepening Search	
Uniform Cost Search	

2 Graph Search (10 points)

Consider the graph shown below where the numbers on the links are link costs and the numbers next to the states are heuristic estimates. Note that the arcs are undirected. Let A be the start state and G be the goal state.



Simulate A* search with a strict expanded list on this graph. At each step, show the path to the state of the node that's being expanded, the length of that path, the total estimated cost of the path (actual + heuristic), and the current value of the expanded list (as a list of states). You are welcome to use scratch paper or the back of the exam pages to simulate the search. However, please transcribe (only) the information requested into the table given below.

Path to State Expanded	Length of Path	Total Estimated Cost	Expanded List
A	0	5	(A)

3 Heuristics and A* (8 points)

1. Is the heuristic given in Problem 2 admissible? Explain.

2. Is the heuristic given in Problem 2 consistent? Explain.

3. Did the A* algorithm with strict expanded list find the optimal path in the previous example? If it did find the optimal path, explain why you would expect that. If it didn't find the optimal path, explain why you would expect that and give a simple (specific) change of state values of the heuristic that would be sufficient to get the correct behavior.

4 Search problem formulation (10 points)

A Mars rover has to leave the lander, collect rock samples from three places (in any order) and return to the lander.

Assume that it has a navigation module that can take it directly from any place of interest to any other place of interest. So it has primitive actions *go-to-lander*, *go-to-rock-1*, *go-to-rock-2*, and *go-to-rock-3*.

We know the time it takes to traverse between each pair of special locations. Our goal is to find a sequence of actions that will perform this task in the shortest amount of time.

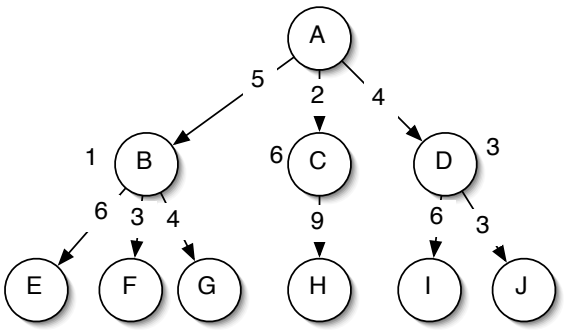
1. Formulate this problem as a search problem by specifying the state space, initial state, path-cost function, and goal test. Try to be sure that the state space is detailed enough to support solving the problem, but not redundant.

2. Say what search technique would be most appropriate, and why.

3. One possible heuristic evaluation function for a state would be the distance back to the lander from the location of the state; this is clearly admissible. What would be a more powerful, but still admissible, heuristic for this problem? (Don't worry about whether it's consistent or not.)

1 Tree Search (10 points)

Consider the tree shown below. The numbers on the arcs are the arc lengths; the numbers near states B, C, and D are the heuristic estimates; all other states have a heuristic estimate of 0.

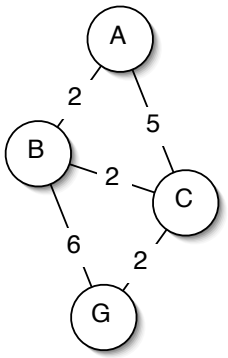


Assume that the children of a node are expanded in alphabetical order when no other order is specified by the search, and that the goal is state J. No visited or expanded lists are used. What order would the states be expanded by each type of search. Write only the sequence of states expanded by each search.

Search Type	List of states
Breadth First	
Depth First	
Progressive Deepening Search	
Best-First Search	
A* Search	

2 Graph Search (8 points)

Consider the graph shown below. Note that the arcs are undirected. Let A be the start state and G be the goal state.



Simulate uniform cost search with a strict expanded list on this graph. At each step, show the state of the node that's being expanded, the length of that path, and the current value of the expanded list (as a list of states).

State Expanded	Length of Path	Expanded List
A	0	(A)

3 $A*$ Algorithm (12 points)

1. Let's consider three elements in the design of the $A*$ algorithm:

   - The heuristic, where the choices are:
     - **arbitrary** heuristic
     - **admissible** heuristic
     - **consistent** heuristic
   - History:
     - **none**
     - **visited** list
     - **strict** expanded list
     - **non-strict** expanded list
   - Pathmax
     - **Use** pathmax
     - **Don't use** pathmax

   In the table below, indicate all the combinations that *guarantee* that $A*$ will find an optimal path. Not all rows have to be filled. If multiple values works for any of Heuristic, History and Pathmax, independent of the other choices, you can write the multiple values in one row. So

Heuristic	History	Pathmax
A,B	C	D,E

   can be used to represent all of: A,C,D; A,C,E; B,C,D; and B,C,E.

Heuristic	History	Pathmax

2. In the network of problem 2, assume you are given the following heuristic values:

$$A = 5; B = 4; C = 0; G = 0$$

Is this heuristic:

- Admissible? Yes No
- Consistent? Yes No

Justify your answer very briefly.

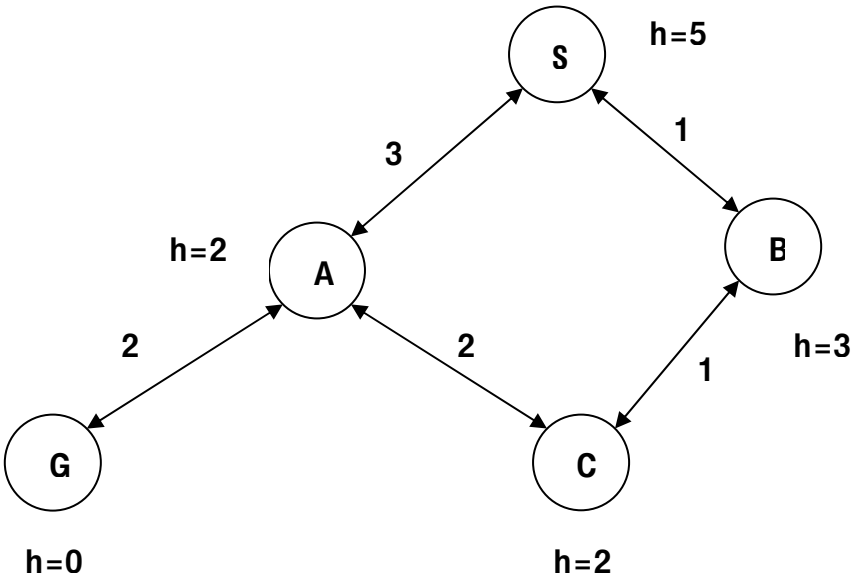
3. With the heuristic above will A* using a strict expanded list find the optimal path?

Yes No

Justify your answer very briefly.

Problem 1 – Search

Below is a graph to be searched (starting at S and ending at G). Link/edge costs are shown as well as heuristic estimates at the states. You may not need all the information for every search.



Draw the complete search tree for this graph. Label each node in the tree with the cost of the path to that node and the heuristic cost at that node. When you need to refer to a node, use the name of the corresponding state and the length of the path to that node.

For each of the searches below, just give a list of node names (state name, length of path) drawn from the tree above. Break ties using alphabetical order.

1. **Perform a depth-first search using a visited list. Assume children of a state are ordered in alphabetical order. Show the sequence of nodes that are expanded by the search.**

2. **Perform a best-first (greedy search) without a visited or expanded list. Show the sequence of nodes that are expanded by the search.**

3. **Perform a Uniform Cost Search without a visited or expanded list. Show the sequence of nodes that are expanded by the search.**

4. **Perform an A* search (no pathmax) without an expanded list. Show the sequence of nodes that are expanded by the search.**

Is the heuristic in this example

1. **admissible?**

2. **consistent?**

Justify your answer, briefly.

For each of the following situations, pick the search that is most appropriate (be specific about visited and expanded list). Give a one sentence reason why you picked it. If you write a paragraph, we will not read it.

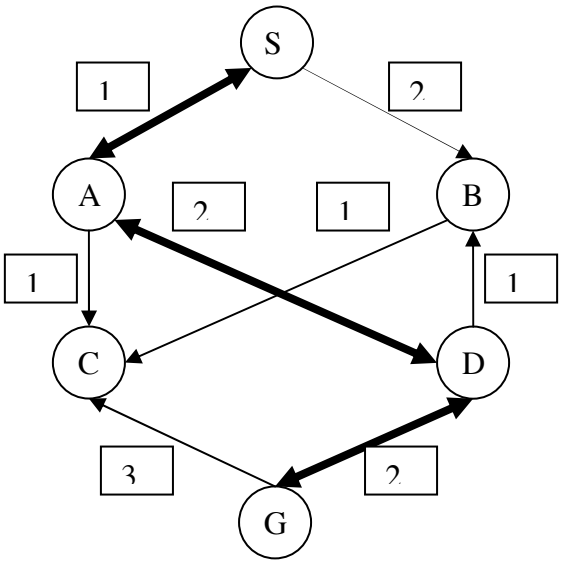
1. **We have a very large search space with a large branching factor and with possibly infinite paths. We have no heuristic. We want to find paths to the goal with minimum numbers of state.**

2. **We have a space with a manageable number of states but lots of cycles in the state graph. We have links of varying costs but no heuristic and we want to find shortest paths.**

3. **Our search space is a tree of fixed depth and all the goals are the leaves of the tree. We have a heuristic and we want to find any goal as quickly as possible.**

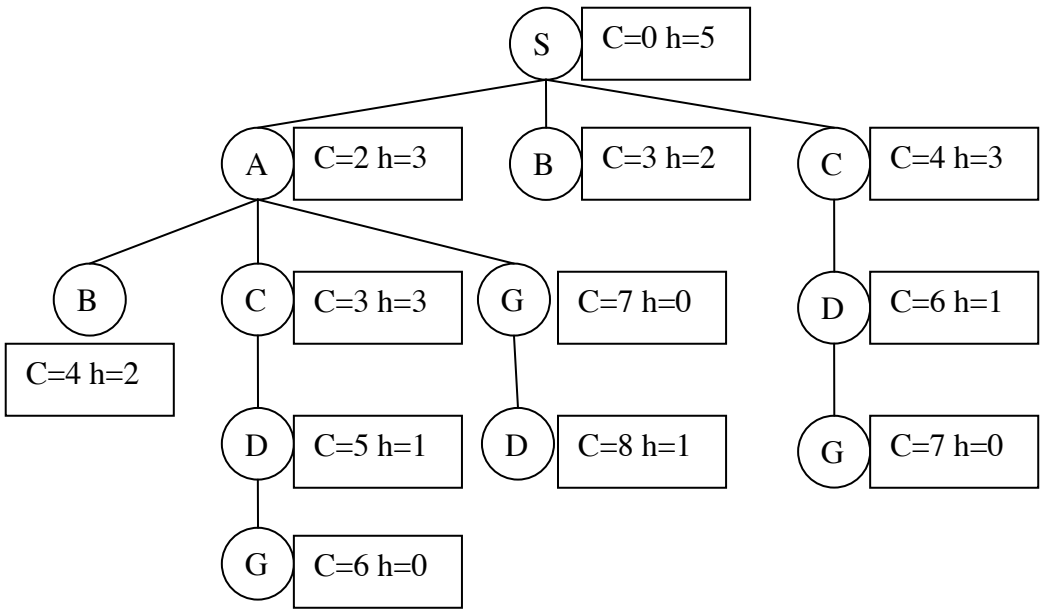
4. **We have a space with a manageable number of states but lots of cycles in the state graph. We have links of varying costs and an admissible heuristic and we want to find shortest paths.**

Problem 1: Search (25 points)



A. Construct the search tree for the graph above, indicate the path length to each node. The numbers shown above are link lengths. Pay careful attention to the arrows; some are bi-directional (shown thick) while some are uni-directional.

B. Using the following search tree (different from Part A), perform the searches indicated below (always from S to G). Each node shows both the total path cost to the node as well as the heuristic value for the corresponding state.



For each of the searches below, write the sequence of nodes **expanded** by the search. Specify a node by writing the name of the state and the length of the path (shown as C=x above), e.g. S0, B3, etc. Break ties using alphabetical order.

1. Depth First Search (no visited list)

2. Breadth First Search (with visited list)

3. Uniform Cost Search (with strict expanded list)

4. A* (without expanded list)

C. Choose the most efficient search method that meets the criteria indicated below. **Explain your choice.**

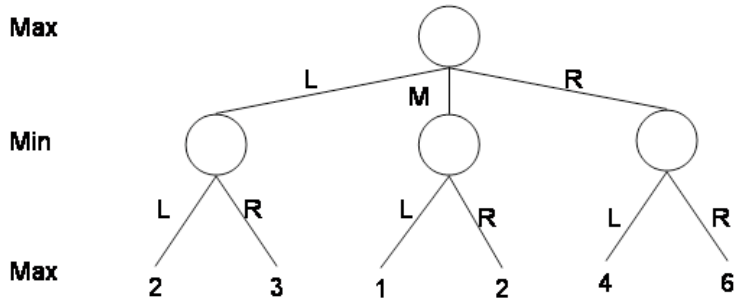
1. You are given a state graph with link costs. The running time of the algorithm should be a function of the number of states in the graph and the algorithm should guarantee that the path with shortest path cost is found.

2. You are given a state graph with link costs and consistent heuristic values on the states. The running time of the algorithm should be a function of the number of states in the graph and the algorithm should guarantee that the path with shortest path cost is found.

You are given a state graph with no link costs or heuristic values. The algorithm should find paths to a goal with the least number of states and the space requirements should depend on the depth of the first goal found and not be exponential in that depth.

6 Game Search (10 points)

Consider the game tree shown below. The top node is a max node. The labels on the arcs are the moves. The numbers in the bottom layer are the values of the different outcomes of the game to the max player.

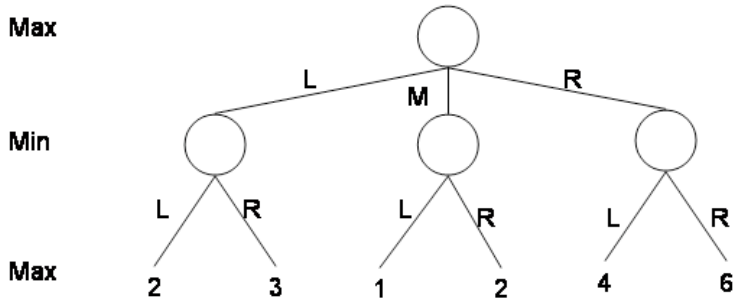


1. What is the value of the game to the max player?

2. What first move should the max player make?

3. Assuming the max player makes that move, what is the best next move for the min player, assuming that this is the entire game tree?

4. Using alpha-beta pruning, consider the nodes from **right to left**, which nodes are cut off? Circle the nodes that are not examined.



6.034 Quiz 1, Spring 2005

1 Search Algorithms (16 points)

1.1 Games

The standard alpha-beta algorithm performs a depth-first exploration (to a pre-specified depth) of the game tree.

1. Can alpha-beta be generalized to do a breadth-first exploration of the game tree and still get the optimal answer? Explain how or why not. If it can be generalized, indicate any advantages or disadvantages of using breadth-first search in this application.

2. Can alpha-beta be generalized to do a progressive-deepening exploration of the game tree and still get the optimal answer? Explain how or why not. If it can be generalized, indicate any advantages or disadvantages of using progressive-deepening search in this application.

1 Propositional Proof (20 pts)

Given the following statements:

1. $P \vee Q$
2. $P \rightarrow R$
3. $Q \rightarrow S$

Prove that $R \vee S$ is entailed, using resolution refutation.

Use the table below. Fill in the Formulas in the proof; in the Reason field, give the parent clause numbers. Start by including the given formulas in the form needed for resolution. You do not need to specify a Reason for the given information. We have given you more than enough space for your proof. You do not need to fill in every line.

Step	Reason	Formula
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		

2 English to FOL (15 points)

Write the following statements in First Order Logic:

1. "Every city has a postman that has been bitten by every dog in town."
   Use predicates:

   - $City(x)$ means x is a city
   - $Postman(x)$ means x is a postman
   - $Dog(x)$ means x is a dog
   - $Lives(x, y)$ means x lives in city y
   - $Bit(x, y)$ means x bit y

2. "All blocks supported by blocks that have been moved have also been moved."
   Use predicates:

   - $Block(x)$ means x is a block
   - $Supports(x, y)$ means x supports y
   - $Moved(x)$ means x has been moved

3 Logic semantics and interpretation (15 points)

Consider the following interpretation of a language with a unary predicates P, Q, and a binary predicate R.

- Universe $= \{1, 2, 3, 4\}$
- $P = \{\langle 1\rangle, \langle 3\rangle\}$
- $Q = \{\langle 2\rangle, \langle 4\rangle\}$
- $R = \{\langle 3, 2\rangle, \langle 4, 3\rangle, \langle 3, 1\rangle, \langle 4, 2\rangle, \langle 2, 1\rangle, \langle 4, 1\rangle\}$

Circle the sentences below that hold in that interpretation.

1. $\forall x.P(x)$
2. $\exists x.P(x)$
3. $\exists x.P(x) \wedge Q(x)$
4. $\exists x.P(x) \rightarrow Q(x)$
5. $\forall x.P(x) \rightarrow Q(x)$
6. $\forall x.P(x) \rightarrow \neg Q(x)$
7. $\forall x.Q(x) \rightarrow \neg P(x)$
8. $\forall x.\exists y.R(x, y)$
9. $\exists y.\forall x.R(x, y)$
10. $\forall x.P(x) \rightarrow \exists y.R(x, y)$
11. $\forall x.Q(x) \rightarrow \exists y.R(x, y)$

1 Clause form and resolution (10 points)

1. Circle the correct clause form for this formula from the choices below.

   $\forall x.\neg(\exists y.P(y) \rightarrow Q(y)) \rightarrow R(x)$

   (a) $P(y) \vee R(x)$
   $\neg Q(y) \vee R(x)$

   (b) $P(f(x)) \vee R(x)$
   $\neg Q(f(x)) \vee R(x)$

   (c) $P(A) \vee R(x)$
   $\neg Q(A) \vee R(x)$

   (d) $\neg P(y) \vee Q(y) \vee R(x)$

   (e) $\neg P(f(x)) \vee Q(f(x)) \vee R(x)$

   (f) $\neg P(A) \vee Q(A) \vee R(x)$

2. Perform resolution on the following clauses; show the unifier and the result.

   $P(f(A), A) \vee \neg Q(f(B), x)$
   $P(f(y), x) \vee Q(x, g(x))$

   (a) Unifier:

   (b) Result:

2 Proof (20 points)

This proof encodes the following argument:

- Every integer has at most one predecessor.
- Two is the predecessor of Three.
- The predecessor of Three is even.
- Therefore, Two is even.

To actually carry this out, we also need some axioms about equality, i.e. equality is transitive, equality is symmetric and equals can be substituted in predicates (such as Even).

Fill in any missing Clauses; in the Reason field, give the parent clause numbers, also fill in all the unifiers, written as a set of variable/value bindings.

Step	Reason	Clause	Unifier
1	Given	$\neg Equals(x, y) \vee \neg Equals(z, y) \vee Equals(z, x)$	None
2	Given	$\neg Equals(x, y) \vee Equals(y, x)$	None
3	Given	$\neg Equals(x, y) \vee \neg Even(x) \vee Even(y)$	None
4	Given	$\neg Pred(z, x) \vee Equals(z, Sk1(x))$	None
5	Given	$Pred(Sk0, Three)$	None
6	Given	$Even(Sk0)$	None
7	Given	$Pred(Two, Three)$	None
8	Given	$\neg Even(Two)$	None
9		$Equals(Two, Sk1(Three))$	
10		$Equals(Sk0, Sk1(Three))$	
11			
12		$Equals(Sk0, Two)$	
13			
14		$\neg Equals(Sk0, Two)$	
15		False	

3 FOL and Entailment (15 points)

Answer each of these questions with **no more than 4 sentences**.

1. Given two first-order logic sentences, A and B, how do you show that A entails B?

2. Given two first-order logic sentences, A and B, how do you show that A does not entail B?

3. What is it about a domain that would make you want to use first-order logic, rather than propositional logic?

2. (8 points) For each group of sentences below, give an interpretation that makes the first sentence(s) true and the last sentence false. Use $\{A, B, C\}$ as your universe.

   (a)

   $$\exists x.p(x) \wedge q(x) \wedge r(x, x)$$
   $$\forall x.p(x) \rightarrow \exists y.\neg r(x, y)$$
   $$\forall x.p(x) \rightarrow \exists y.\neg x = y \wedge r(x, y)$$

   $$\forall x.p(x) \vee \neg q(x)$$

   Solution:

   $$\begin{aligned} p &= \{< A >\} \\ q &= \{< A, C >\} \\ r &= \{< A, A >, < A, B >\} \end{aligned}$$

   (b)

   $$\forall x.p(x) \leftrightarrow \exists y.r(y, x)$$
   $$\forall x.\exists y.r(x, y)$$

   $$\forall x.\neg p(x)$$

   Solution:

   $$\begin{aligned} p &= \{< A >\} \\ r &= \{< A, A >, < B, A >, < C, A >\} \end{aligned}$$

   Alternately: (there are others, too)

   $$\begin{aligned} p &= \{< A, B, C >\} \\ r &= \{< A, A >, < B, B >, < C, C >\} \end{aligned}$$

6.825: Final Exam

There are 130 points total. Points for individual problems are indicated in bold.

1 Search

(**10**) You're a taxi driver. Your taxi can hold 4 passengers. Passengers pay a flat fee for a ride to the airport, so goal is to pick up 4 passengers and take them to the airport in the smallest number of miles. Your world can be modeled as a graph of locations with distances between them. Some, but not all, of the locations have passengers that you can pick up.

a. Describe the state space of this search problem.

b. What would be a good cost function for this search problem?

c. Now, consider a case where passengers have to pay according to how far away they are from the airport when they're picked up (note: they don't pay according to how long a ride they take in your taxi, but according to the length of the shortest path from their pickup-point to the airport).

   Describe the state space of this search problem.

d. What would be a good cost function for this version of the problem? You still have a desire to save gas.

e. Is uniform cost search guaranteed to find the optimal solution in either or both versions of the problem? Why or why not?

2 FOL Semantics

(**6**) Consider a world with objects **A**, **B**, and **C**. We'll look at a logical languge with constant symbols X, Y, and Z, function symbols f and g, and predicate symbols p, q, and r. Consider the following interpretation:

- $I(X) = \mathbf{A}$, $I(Y) = \mathbf{A}$, $I(Z) = \mathbf{B}$
- $I(f) = \{\langle \mathbf{A}, \mathbf{B}\rangle, \langle \mathbf{B}, \mathbf{C}\rangle, \langle \mathbf{C}, \mathbf{C}\rangle\}$
- $I(p) = \{\mathbf{A}, \mathbf{B}\}$

- $I(q) = \{\mathbf{C}\}$
- $I(r) = \{\langle \mathbf{B}, \mathbf{A}\rangle, \langle \mathbf{C}, \mathbf{B}\rangle, \langle \mathbf{C}, \mathbf{C}\rangle\}$

For each of the following sentences, say whether it is true or false in the given interpretation I:

a. $q(f(Z))$

b. $r(X, Y)$

c. $\exists w.f(w) = Y$

d. $\forall w.r(f(w), w)$

e. $\forall u, v.r(u, v) \rightarrow (\forall w.r(u, w) \rightarrow v = w)$

f. $\forall u, v.r(u, v) \rightarrow (\forall w.r(w, v) \rightarrow u = w)$

3 Interpretations

(**6**) Using the same set of symbols as in the previous problem, for each group of sentences below, provide an interpretation that makes the sentences true, or show that it's impossible.

a.
- $\exists w.p(w) \wedge \exists w.q(w)$
- $\neg\exists w.p(w) \wedge q(w)$
- $\forall u.p(u) \rightarrow \exists v.r(u, v)$

b.
- $\forall u.\exists v.r(u, v)$
- $\exists u, v.\neg r(u, v)$
- $\forall v.(\exists u.r(u, v)) \leftrightarrow p(v))$

c.
- $\forall u, v.(p(v) \rightarrow r(u, v))$
- $\exists u, v.\neg r(u, v)$
- $\exists v.p(v)$

4 Unification

(**6**) For each pair of literals below, specify a most general unifier, or indicate that they are not unifiable.

a. $r(f(x), y)$ and $r(z, g(w))$

b. $r(f(x), x)$ and $r(y, g(y))$

c. $r(a, C, a)$ and $r(f(x), x, y)$

1 First-Order Logic (24 points)

Here are some English sentences and their translation into clausal form.

1. Every car has a driver.
$$D(f(x_1), x_1)$$
2. The driver of a car is in the car.
$$\neg D(x_2, y_2) \vee In(x_2, y_2)$$
3. "In" is transitive.
$$\neg In(x_3, y_3) \vee \neg In(y_3, z_3) \vee In(x_3, z_3)$$
4. Drivers are people.
$$\neg D(x_4, y_4) \vee P(x_4)$$
5. Chitty (a car) is in the Stata garage.
$$In(C, SG)$$
6. Therefore, there is a person in the Stata garage. (This clause is the negation of the conclusion).
$$\neg P(x_6) \vee \neg In(x_6, SG)$$

We'd like to prove the conclusion using resolution refutation. This proof is kind of tricky, so we're going to tell you, in English, what the steps should be. For each step, say which of the previous clauses (P1 and P2 in the table) it can be derived from using resolution, what the resulting clause is and what the unifier is.

Step	P1	P2	Clause	Unifier
7	Every driver is in their car. (one term)			
8	If a car is in some location, then its driver is in that location. (two terms)			
9	The driver of a car is a person. (one term)			
10	The driver of Chitty is in the Stata garage. (one term)			
11	There is no car whose driver is in the Stata garage. (one term)			
12	False			

9 Resolution Proof (15 points)

Prove a contradiction from these clauses using resolution. For each new step, indicate which steps it was derived from (in columns labeled P1 and P2) and what the unifier was. Note that A is a constant and b, c, d, x, y, u, v, w are all variables.

Step	P1	P2	Clause	Unifier
1	XX	XX	$\neg P(x, f(x), y) \vee R(y, g(x))$	XXXXXXXXX
2	XX	XX	$\neg R(u, v) \vee \neg Q(v) \vee S(u, h(v))$	XXXXXXXXX
3	XX	XX	$Q(g(A))$	XXXXXXXXX
4	XX	XX	$\neg S(w, w)$	XXXXXXXXX
5	XX	XX	$P(b, c, h(d))$	XXXXXXXXX
6				
7				
8				
9				
10				

Given the following clauses, do a resolution refutation proof.

1. ¬P(x,f(x))∨¬R(f(x))∨¬Q(x,g(x))
2. ¬P(x2,y2)∨ Q(x2,y2)
3. ¬P(x3,y3)∨ R(y3)
4. P(A,x4) [Negated Goal]

Step	Parent	Parent	New Clause	MGU

C. Given the following clauses:

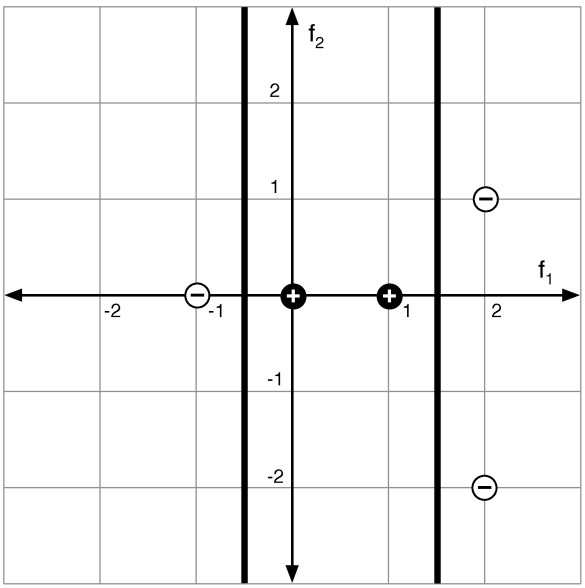
1. Hasjob(p, job(p))
2. ¬ Hasjob(p, k) ∨ Equal(job(p), k)
3. Hasjob(George, Fireman)
4. ¬ Equal(Fireman, Teacher)
5. ¬ Equal(x,y) ∨ ¬ Equal(y, z) ∨ Equal(x, z)
6. ¬ Equal(x,y) ∨ Equal(y,x)

Prove by resolution refutation that:
¬ Hasjob(George, Teacher)

Hint: think about the strategy for the proof before you start doing resolutions. How would you prove the result by hand?

Step	**Parent**	**Parent**		**Unifier**
7	Neg	Goal	Hasjob(George, Teacher)	
8				
9				
10				
11				
12				
13				
14				

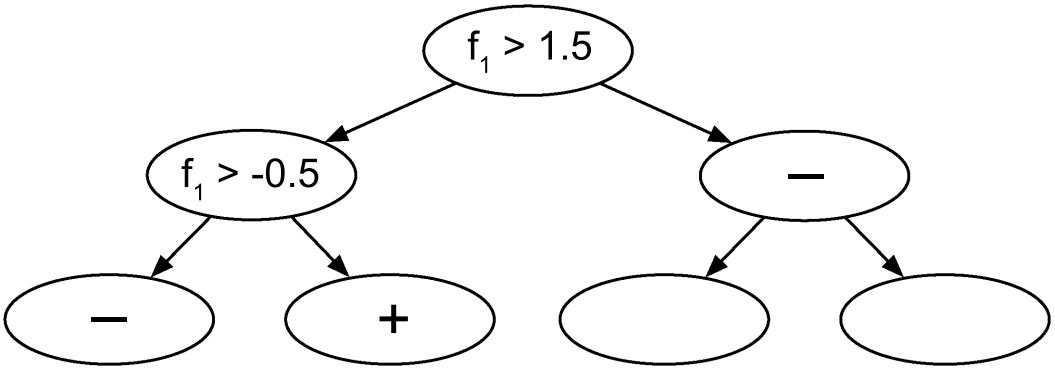
1 Decision Trees (13 pts)



Data points are: Negative: (-1, 0) (2, 1) (2, -2) Positive: (0, 0) (1, 0)

Construct a decision tree using the algorithm described in the notes for the data above.

1. Show the tree you constructed in the diagram below. The diagram is more than big enough, leave any parts that you don't need blank.



2. Draw the decision boundaries on the graph at the top of the page.

3. Explain how you chose the top-level test in the tree. The following table may be useful.

x	y	-(x/y)*lg(x/y)		x	y	-(x/y)*lg(x/y)
1	2	0.50		1	5	0.46
1	3	0.53		2	5	0.53
2	3	0.39		3	5	0.44
1	4	0.50		4	5	0.26
3	4	0.31				

Pick the decision boundary which falls halfway between each pair of adjacent points in each dimension, and which produces the minimum average entropy

$$\begin{aligned} AE &= q_< H(p_<) + (1 - q_<) H(p_>) \\ H(p) &= -p \lg(p) - (1 - p) \lg(1 - p) \end{aligned}$$

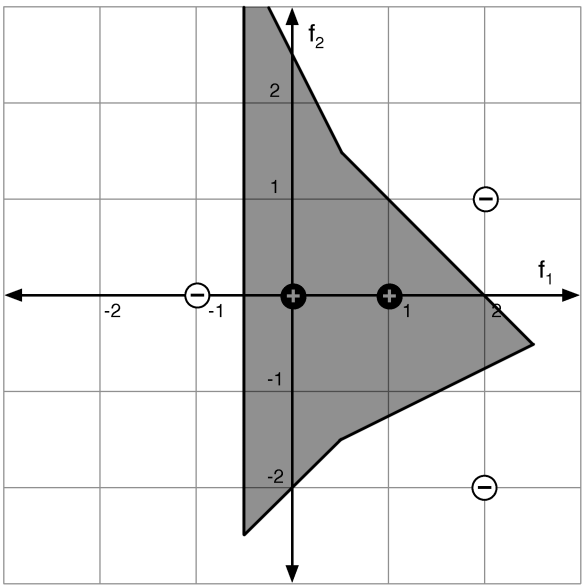
where $q_<$ is the fraction of points below the decision boundary, and $p_<$, $p_>$ are the fraction of positive (+) points below and above the decision boundary, respectively.

$$\begin{aligned} f_2 > \pm 0.5: \quad & AE = \frac{1}{5}(0) + \frac{4}{5}(1) = 0.8 \\ f_1 > 1.5: \quad & AE = \frac{3}{5} H\left(\frac{2}{3}\right) + \frac{2}{5}(0) = \frac{3}{5}(0.39 + 0.53) = 0.552 \\ f_1 > 0.5: \quad & AE = \frac{2}{5}(1) + \frac{3}{5} H\left(\frac{1}{3}\right) = \frac{2}{5} + \frac{3}{5}(0.39 + 0.53) = 0.952 \\ f_1 > -0.5: \quad & AE = \frac{1}{5}(0) + \frac{4}{5}(1) = 0.8 \end{aligned}$$

4. What class does the decision tree predict for the new point: (1, -1.01)

Positive (+)

2 Nearest Neighbors (8 pts)



Data points are: Negative: (-1, 0) (2, 1) (2, -2) Positive: (0, 0) (1, 0)

1. Draw the decision boundaries for 1-Nearest Neighbors on the graph above. Try to get the integer-valued coordinate points in the diagram on the correct side of the boundary lines.

2. What class does 1-NN predict for the new point: (1, -1.01) Explain why.

   Positive (+) since this is the class of the closest data point (1,0).

3. What class does 3-NN predict for the new point: (1, -1.01) Explain why.

   Positive (+) since it is the majority class of the three closest data points (0,0), (1,0) and (2,-2).

6 Learning algorithms (16 pts)

For each of the learning situations below, say what learning algorithm would be best to use, and why.

1. You have about 1 million training examples in a 6-dimensional feature space. You only expect to be asked to classify 100 test examples.

   Nearest Neighbors is a good choice. The dimensionality is low and so appropriate for KNN. For KNN, training is very fast and since there are few classifications, the fact that this will be slow does not matter. With 1 million training examples, neural net and SVM will be extremely expensive to train. Naive Bayes is plausible on computational grounds but likely to be less accurate than KNN.

2. You are going to develop a classifier to recommend which children should be assigned to special education classes in kindergarten. The classifier has to be justified to the board of education before it is implemented.

   A Decision Tree is a good choice since the resulting classifier will need to be understandable to humans.

3. You are working for Am*z*n as it tries to take over the retailing world. You are trying to predict whether customer X will like a particular book, as a function of the input which is a vector of 1 million bits specifying whether each of Am*z*n's other customers liked the book. You will train a classifier on a very large data set of books, where the inputs are everyone else's preferences for that book, and the output is customer X's preference for that book. The classifier will have to be updated frequently and efficiently as new data comes in.

   Naive Bayes is a good choice since it is fast to train and update. The dimensionality is high for Nearest Neighbors and Decision Trees. SVM's have to be re-trained from scratch if the data changes. Neural Nets could be trained incrementally but it will generally take a lot of iterations to change the current settings of the weights.

4. You are trying to predict the average rainfall in California as a function of the measured currents and tides in the Pacific ocean in the previous six months.

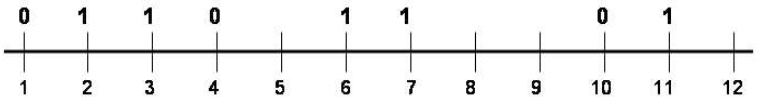
   This is a regression problem; neural nets with linear output functions, regression trees or locally weighted nearest neighbors are all appropriate choices.

4 Machine Learning — Continuous Features (20 points)

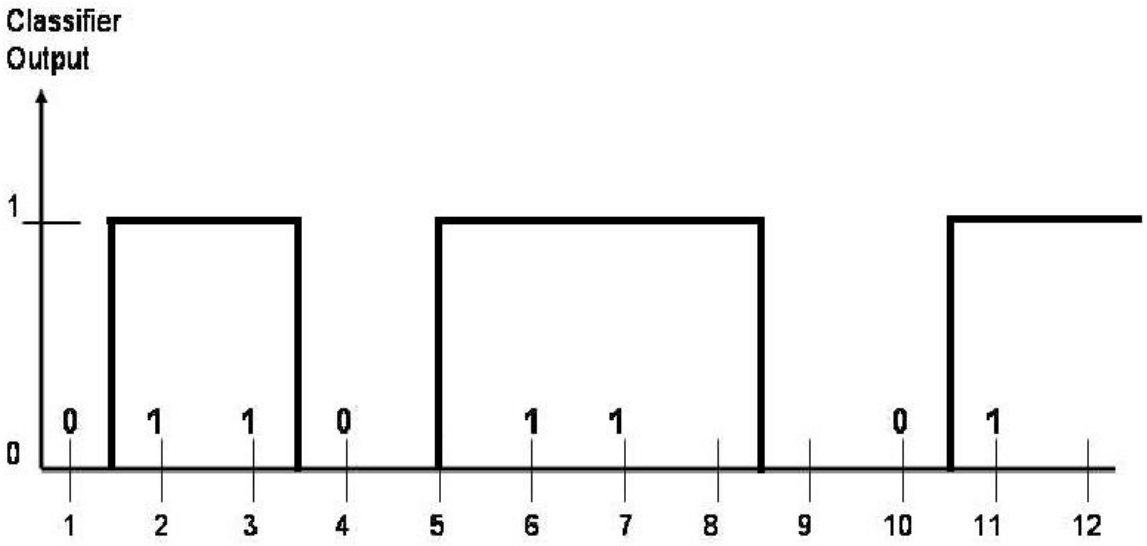
In all the parts of this problem we will be dealing with one-dimensional data, that is, a set of points (x^i) with only one feature (called simply x). The points are in two classes given by the value of y^i. We will show you the points on the x axis, labeled by their class values; we also give you a table of values.

4.1 Nearest Neighbors

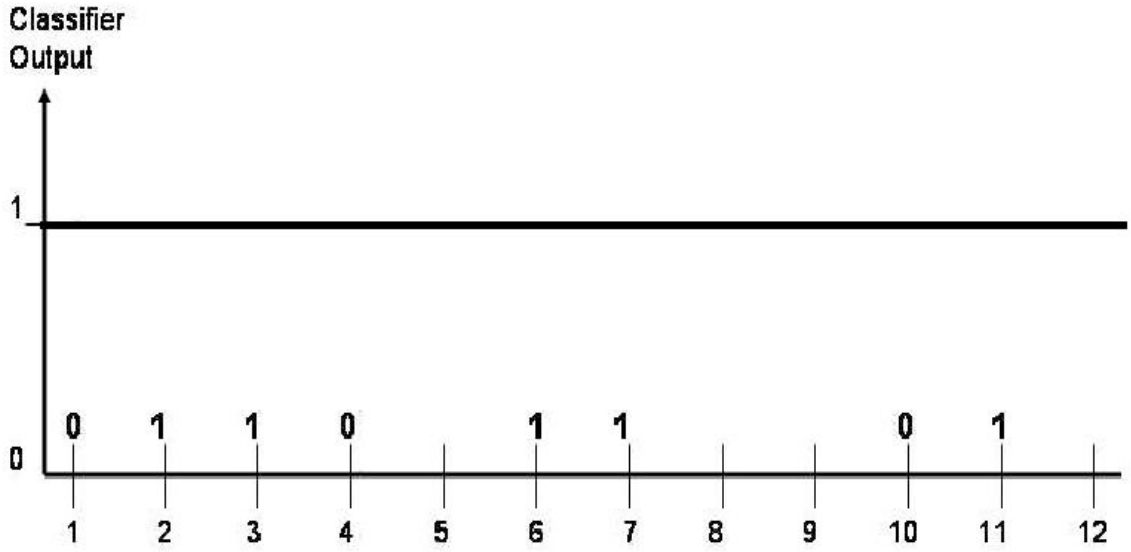
i	x^i	y^i
1	1	0
2	2	1
3	3	1
4	4	0
5	6	1
6	7	1
7	10	0
8	11	1



1. In the figure below, draw the output of a 1-Nearest-Neighbor classifier over the range indicated in the figure.

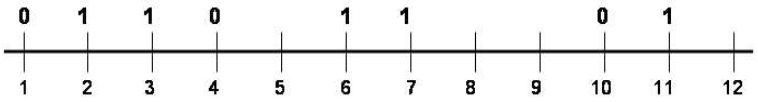


2. In the figure below, draw the output of a 5-Nearest-Neighbor classifier over the range indicated in the figure.



4.2 Decision Trees

Answer this problem using the same data as in the Nearest Neighbor problem above.



Which of the following three tests would be chosen as the top node in a decision tree?

$$x \leq 1.5 \qquad x \leq 5 \qquad x \leq 10.5$$

Justify your answer.

Recall that entropy for each side of a split is:

$$H = -p \log p - (1-p) \log(1-p)$$

So, for $x \leq 1.5$ *we have:*

$$\begin{aligned} H &= \frac{1(0) + 7\left(-\frac{5}{7}\log_2(\frac{5}{7}) - \frac{2}{7}\log_2(\frac{2}{7})\right)}{8} \\ H &= \frac{7(0.35 + 0.52)}{8} \\ H &= 0.761 \end{aligned}$$

while $x \leq 5$

$$\begin{aligned} H &= \frac{4\left(-\frac{2}{4}\log_2(\frac{2}{4}) - \frac{2}{4}\log_2(\frac{2}{4})\right) + 4\left(-\frac{3}{4}\log_2(\frac{3}{4}) - \frac{1}{4}\log_2(\frac{1}{4})\right)}{8} \\ H &= \frac{4(0.5 + 0.5) + 4(0.31 + 0.50)}{8} \\ H &= 0.905 \end{aligned}$$

and $x \leq 10.5$ *gives us:*

$$\begin{aligned} H &= \frac{1(0) + 7\left(-\frac{4}{7}\log_2(\frac{4}{7}) - \frac{3}{7}\log_2(\frac{3}{7})\right)}{8} \\ H &= \frac{7(0.46 + 0.52)}{8} \\ H &= 0.85 \end{aligned}$$

So, we choose the split with the least average entropy, which is $x \leq 1.5$.

6 Pruning Trees (20 points)

Following are some different strategies for pruning decision trees. We assume that we grow the decision tree until there is one or a small number of elements in each leaf. Then, we prune by deleting individual leaves of the tree until the score of the tree starts to get worse. The question is how to score each possible pruning of the tree.

For each possible definition of the score below, explain whether or not it would be a good idea and give a reason why or why not.

1. The score is the percentage correct of the tree on the training set.

   Not a good idea. The original tree was constructed to maximize performance on the training set. Pruning any part of the tree will reduce performance on the training set.

2. The score is the percentage correct of the tree on a separate validation set.

   A good idea. The validation set will be an independent check on whether pruning a node is likely to increase or decrease performance on unseen data.

3. The score is the percentage correct of the tree, computed using cross validation.

   Not a good idea. Cross-validation allows you to evaluate algorithms, not individual hypotheses. Cross-validation will construct many new hypotheses and average their performance, this will not tell you whether pruning a node in a particular hypothesis is worthwhile or not.

4. The score is the percentage correct of the tree, computed on the training set, minus a constant C times the number of nodes in the tree.

   C is chosen in advance by running this algorithm (grow a large tree then prune in order to maximize percent correct minus C times number of nodes) for many different values of C, and choosing the value of C that minimizes training-set error.

   Not a good idea. Running trials to maximize performance on the training set will not give us an indication of whether this algorithm will produce answers that generalize to other data sets.

5. The score is the percentage correct of the tree, computed on the training set, minus a constant C times the number of nodes in the tree.

   C is chosen in advance by running cross-validation trials of this algorithm (grow a large tree then prune in order to maximize percent correct minus C times number of nodes) for many different values of C, and choosing the value of C that minimizes cross-validation error.

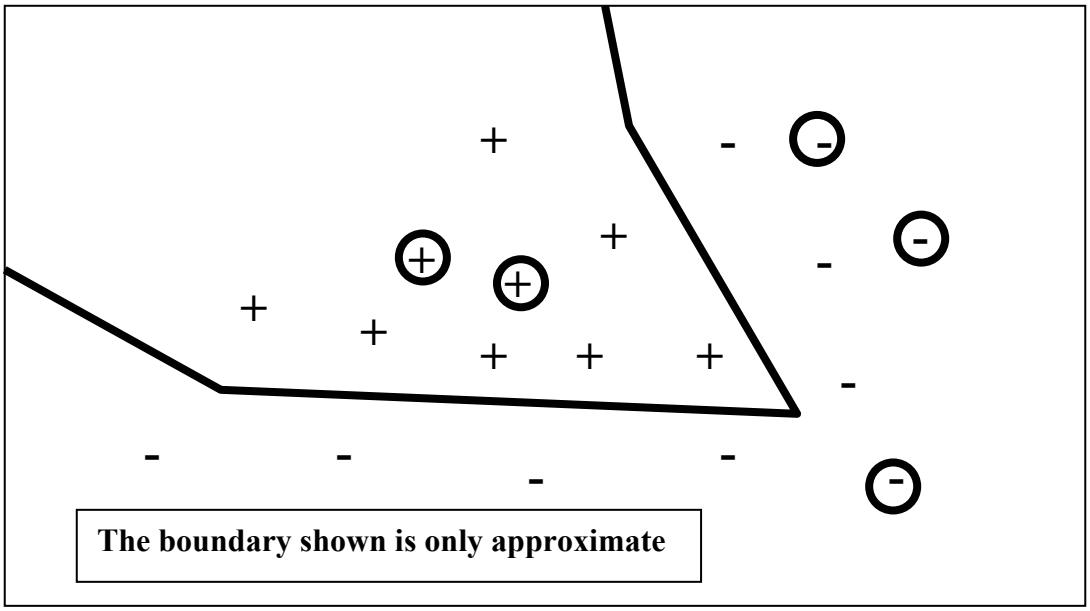
   A good idea when we don't have enough data to hold out a validation set. Choosing C by cross-validation will hopefully give us an effective general way of penalizing for complexity of the tree (for this type of data).

Problem 4: Learning (25 points)

Part A: (5 Points)

Since the cost of using a nearest neighbor classifier grows with the size of the training set, sometimes one tries to eliminate redundant points from the training set. These are points whose removal does not affect the behavior of the classifier for any possible new point.

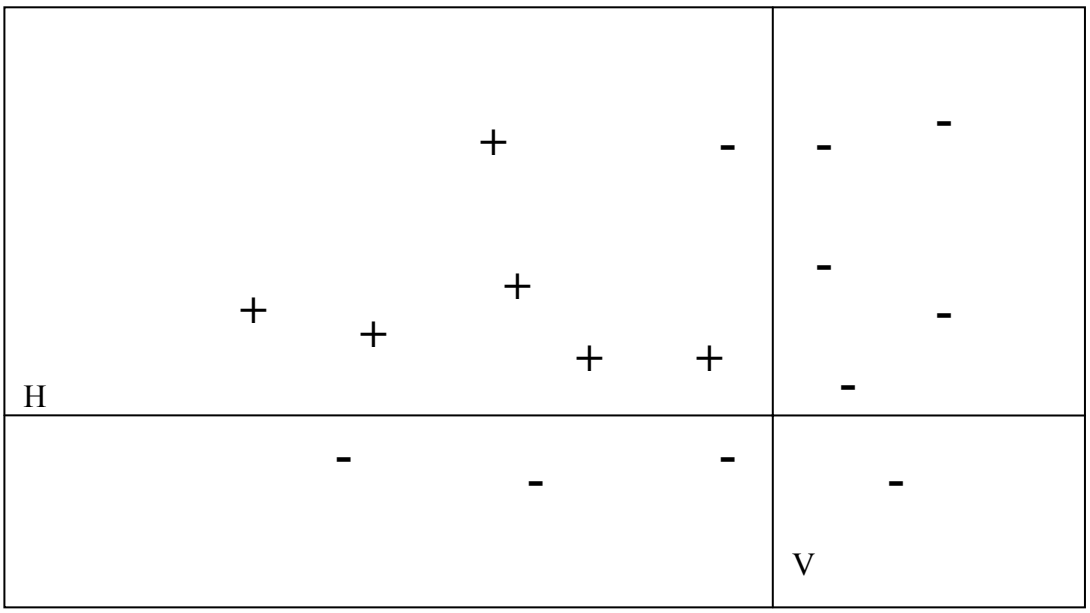
1. In the figure below, sketch the decision boundary for a 1-nearest-neighbor rule and circle the redundant points.



2. What is the general condition(s) required for a point to be declared redundant for a 1-nearest-neighor rule? Assume we have only two classes (+, -). Restating the definition of redundant ("removing it does not change anything") is not an acceptable answer. Hint – think about the neighborhood of redundant points.

Let the Voronoi cell for a training point be the set of points that are closest to that point (as opposed to some other training point). The Voronoi cell of a redundant point touches only on other Voronoi cells of points of the same class.

Part B: (5 Points)



Which of H or V would be preferred as an initial split for a decision (identification) tree? Justify your answer numerically.

V = 0.60625
H = 0.75

So, V is chosen

x	y	-(x/y)*lg(x/y)
1	2	0.50
1	3	0.53
2	3	0.39
1	4	0.50
3	4	0.31
1	5	0.46
2	5	0.53
3	5	0.44
4	5	0.26
1	6	0.43
2	6	0.53
5	6	0.22
1	7	0.40
2	7	0.52
3	7	0.52
4	7	0.46
5	7	0.35
6	7	0.19

x	y	-(x/y)*lg(x/y)
1	8	0.38
3	8	0.53
5	8	0.42
7	8	0.17
1	9	0.35
2	9	0.48
4	9	0.52
5	9	0.47
7	9	0.28
8	9	0.15
1	10	0.33
3	10	0.52
7	10	0.36
9	10	0.14

Problem 2: Overfitting (20 points)

For each of the supervised learning methods that we have studied, indicate how the method could overfit the training data (consider both your design choices as well as the training) and what you can do to minimize this possibility. There may be more than one mechanism for overfitting, make sure that you identify them all.

Part A: Nearest Neighbors (5 Points)

1. How does it overfit?
   Every point in dataset (including noise) defines its own decision boundary.
   The distance function can be chosen to do well on training set but less well on new data.

2. How can you reduce overfitting?
   Use k-NN for larger k
   Use cross-validation to choose k and the distance function

Part B: Decision Trees (5 Points)

1. How does it overfit?
   By adding new tests to the tree to correctly classify every data point in the training set.

2. How can you reduce overfitting?
   By pruning the resulting tree based on performance on a validation set.

Problem 3: Spaminator (10 points)

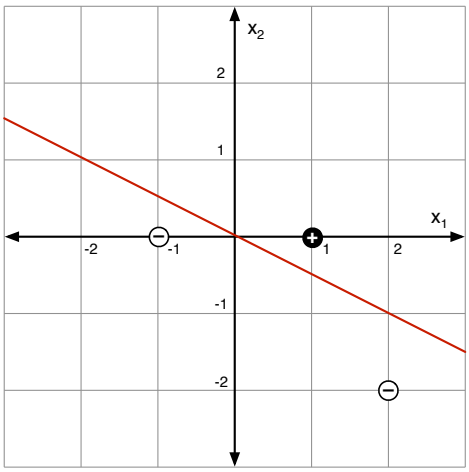
Suppose that you want to build a program that detects whether an incoming e-mail message is spam or not. You decide to attack this using machine learning. So, you collect a large number of training messages and label them as spam or not-spam. You further decide that you will use the presence of individual words in the body of the message as features. That is, you collect every word found in the training set and assign to each one an index, from 1 to N. Then, given a message, you construct a feature vector with N entries and write in each entry a number that indicates how many times the word appears in that message.

Part A: (6 Points)

If you had to choose between a Nearest Neighbor implementation or an Decision Tree implementation, which would you choose? Justify your answer briefly both in terms of expected accuracy and efficiency of operation. Indicate the strength and weaknesses of each approach.

Nearest Neighbors does not work well in high dimensions.
The biggest problem in using Nearest Neighbor would be choosing which of the features are relevant (which is related to choosing the distance metric). This is particularly severe in this application because of the huge numbers of probably irrelevant features (words).
The ID tree approach would spend initial effort in choosing which words were relevant to making the decision.
Nearest neighbor is also very expensive during classification for high dimensional feature vectors. ID trees would be much more efficient.
So, ID trees would be clearly better choice on all criteria.
However, Naïve Bayes would probably be better than either of them.

3 Perceptron (7 pts)



Data points are: Negative: (-1, 0) (2, -2) Positive: (1, 0). Assume that the points are examined in the order given here. Recall that the perceptron algorithm uses the extended form of the data points in which a 1 is added as the 0th component.

1. The linear separator obtained by the standard perceptron algorithm (using a step size of 1.0 and a zero initial weight vector) is (0 1 2). Explain how this result was obtained.

   The perceptron algorithm cycles through the augmented points, updating weights according to the update rule $w_{\text{new}} = w + y \cdot x$ after misclassifying points. The intermediate weights are given in the table below.

Test point	misclassified?	Updated weights
Initial weights		**0 0 0**
-: (1 -1 0)	**yes**	**-1 1 0**
-: (1 2 -2)	**yes**	**-2 -1 2**
+: (1 1 0)	**yes**	**-1 0 2**
-: (1 -1 0)	**no**	
-: (1 2 -2)	**no**	
+: (1 1 0)	**yes**	**0 1 2**
-: (1 -1 0)	**no**	
-: (1 2 -2)	**no**	
+: (1 1 0)	**no**	

2. What class does this linear classifier predict for the new point: (2.0, -1.01)

   The margin of the point is -0.01, so it would be classified as negative.

3. Imagine we apply the perceptron learning algorithm to the 5 point data set we used on Problem 1: Negative: (-1, 0) (2, 1) (2, -2), Positive: (0, 0) (1, 0). Describe qualitatively what the result would be.

   The perceptron algorithm would not converge since the 5 point data set is not linearly separable.

6 Perceptron (8 points)

The following table shows a data set and the number of times each point is misclassified during a run of the perceptron algorithm, starting with zero weights. What is the equation of the separating line found by the algorithm, as a function of x_1, x_2, and x_3? Assume that the learning rate is 1 and the initial weights are all zero.

x_1	x_2	x_3	y	times misclassified
2	3	1	+1	12
2	4	0	+1	0
3	1	1	-1	3
1	1	0	-1	6
1	2	1	-1	11

$$\begin{aligned}\bar{w} &= \eta \sum_{i=1}^{m} \alpha_i y^i \bar{x}^i \\ &= (12)(1)(1,2,3,1) + (3)(-1)(1,3,1,1) + (6)(-1)(1,1,1,0) + (11)(-1)(1,1,2,1) \\ &= (-8,-2,5,-2)\end{aligned}$$

So the equation of the separating line is

$$-2x_1 + 5x_2 - 2x_3 - 8 = 0$$

Problem 2: Overfitting (20 points)

For each of the supervised learning methods that we have studied, indicate how the method could overfit the training data (consider both your design choices as well as the training) and what you can do to minimize this possibility. There may be more than one mechanism for overfitting, make sure that you identify them all.

Part A: Nearest Neighbors (5 Points)

1. How does it overfit?
   Every point in dataset (including noise) defines its own decision boundary.
   The distance function can be chosen to do well on training set but less well on new data.

2. How can you reduce overfitting?
   Use k-NN for larger k
   Use cross-validation to choose k and the distance function

Part B: Decision Trees (5 Points)

1. How does it overfit?
   By adding new tests to the tree to correctly classify every data point in the training set.

2. How can you reduce overfitting?
   By pruning the resulting tree based on performance on a validation set.

4 Search Problem formulation (23 points)

Consider a Mars rover that has to drive around the surface, collect rock samples, and return to the lander. We want to construct a plan for its exploration.

- It has batteries. The batteries can be charged by stopping and unfurling the solar collectors (pretend it's always daylight). One hour of solar collection results in one unit of battery charge. The batteries can hold a total of 10 units of charge.
- It can drive. It has a map at 10-meter resolution indicating how many units of battery charge and how much time (in hours) will be required to reach a suitable rock in each square.
- It can pick up a rock. This requires one unit of battery charge. The robot has a map at 10-meter resolution that indicates the type of rock expected in that location and the expected weight of rocks in that location. Assume only one type of rock and one size can be found in each square.

The objective for the rover is to get one of each of 10 types of rocks, within three days, while minimizing a combination of their total weight and the distance traveled. You are given a tradeoff parameter α that converts units of weight to units of distance. The rover starts at the lander with a full battery and must return to the lander.

Here is a list of variables that might be used to describe the rover's world:

- types of rocks already collected
- current rover location (square on map)
- current lander location (square on map)
- weight of rocks at current location (square on map)
- cost to traverse the current location (square on map)
- time since last charged
- time since departure from lander
- current day
- current battery charge level
- total battery capacity
- distance to lander
- total weight of currently collected rocks

1. Use a set of the variables above to describe the rover's state. Do not include extraneous information.

   - **types of rocks already collected**
   - **current rover location (square on map)**
   - **time since departure from lander**
   - **current battery charge level**
   - **total weight of currently collected rocks (optional, depending on your choice of cost function)**

2. Specify the goal test.

   - **All types of rocks have been collected**
   - **rover at lander location**
   - **time since departure less than 3 days**

3. Specify the actions. Indicate how they modify the state and any preconditions for being used.

   charge **: precondition: none; effects: increases battery voltage by 1 unit, increases time-since-departure by 1 hour**

   move **: precondition: enough battery voltage to cross square; effects: decreases battery voltage by amount specified in map; increases time by amount specified in map; changes rover location**

   pick-up-rock **: precondition: enough battery voltage; effects: decreases battery voltage by 1 unit; changes types of rocks already collected**

4. Specify a function that determines the cost of each action.

   charge **: 0**

   move **: 10 meters**

   pick-up-rock **:** α *** weight-of-rocks-at-current-location**

5. This can be treated as a path search problem. We would like to find a heuristic. Say whether each of these possible heuristics would be useful in finding the optimal path or, if not, what's wrong with them. Let l be the number of rocks already collected.

   H1: The sum of the distances (in the map) from the rover to the $10 - l$ closest locations for the missing types of rocks.

   This heuristic is inadmissible.

   H2: The length of the shortest tour through the $10 - l$ closest locations for the missing types of rocks.

   This heuristic would take an impractical amount of time to compute; and while more reasonable than H1 is also inadmissible.

   H3: The distance back to the lander.

   This heuristic is admissible, but very weak.

5 Search traces (21 points)

Consider the graph shown in the figure below. We can search it with a variety of different algorithms, resulting in different search trees. Each of the trees (labeled G1 though G7) was generated by searching this graph, but with a different algorithm. Assume that children of a node are visited in alphabetical order. Each tree shows all the nodes that have been visited. Numbers next to nodes indicate the relevant "score" used by the algorithm for those nodes.

For each tree, indicate whether it was generated with

1. Depth first search
2. Breadth first search
3. Uniform cost search
4. A* search
5. Best-first (greedy) search

In all cases a strict expanded list was used. Furthermore, if you choose an algorithm that uses a heuristic function, say whether we used

H1: heuristic 1 = $\{h(A) = 3, h(B) = 6, h(C) = 4, h(D) = 3\}$

H2: heuristic 2 = $\{h(A) = 3, h(B) = 3, h(C) = 0, h(D) = 2\}$

Also, for all algorithms, say whether the result was an optimal path (measured by sum of link costs), and if not, why not. Be specific.

Write your answers in the space provided below (not on the figure).

G1: 1. Algorithm: **Breadth First Search**
2. Heuristic (if any): **None**
3. Did it find least-cost path? If not, why? **No. Breadth first search is only guaranteed to find a path with the shortest number of links; it does not consider link cost at all.**

G2: 1. Algorithm: **Best First Search**
2. Heuristic (if any): **H1**
3. Did it find least-cost path? If not, why?
No. Best first search is not guaranteed to find an optimal path. It takes the first path to goal it finds.

G3: 1. Algorithm: **A***
2. Heuristic (if any): **H1**
3. Did it find least-cost path? If not, why? **No. A* is only guaranteed to find an optimal path when the heuristic is admissible (or consistent with a strict expanded list). H1 is neither: the heuristic value for C is not an underestimate of the optimal cost to goal.**

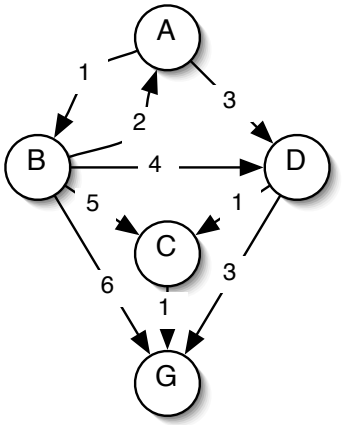
G4: 1. Algorithm: **Best First Search**
2. Heuristic (if any): **H2**
3. Did it find least-cost path? If not, why? **Yes. Though best first search is not guaranteed to find an optimal path, in this case it did.**

G5: 1. Algorithm: **Depth First Search**
2. Heuristic (if any): **None**
3. Did it find least-cost path? If not, why? **No. Depth first search is an any-path search; it does not consider link cost at all.**

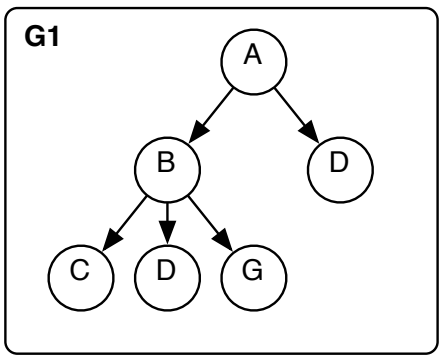
G6: 1. Algorithm: **A***
2. Heuristic (if any): **H2**

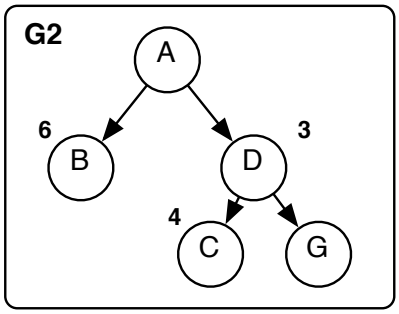
3. Did it find least-cost path? If not, why? **Yes. A* is guaranteed to find an optimal path when the heuristic is admissible (or consistent with a strict expanded list). H2 is admissible but not consistent, since the link from D to C decreases the heuristic cost by 2, which is greater than the link cost of 1. Still, the optimal path was found.**

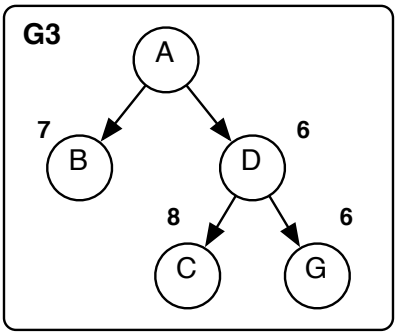
G7: 1. Algorithm: **Uniform Cost Search**
2. Heuristic (if any): **None**
3. Did it find least-cost path? If not, why? **Yes. Uniform Cost is guaranteed to find a shortest path.**

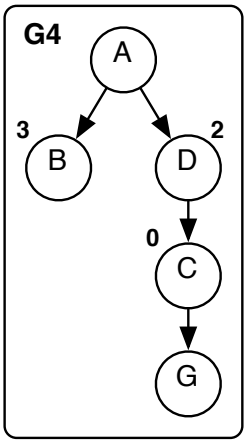


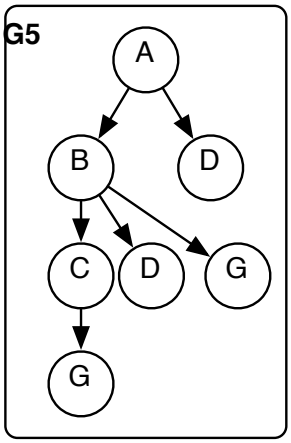
	H1	H2
A	3	3
B	6	3
C	4	0
D	3	2

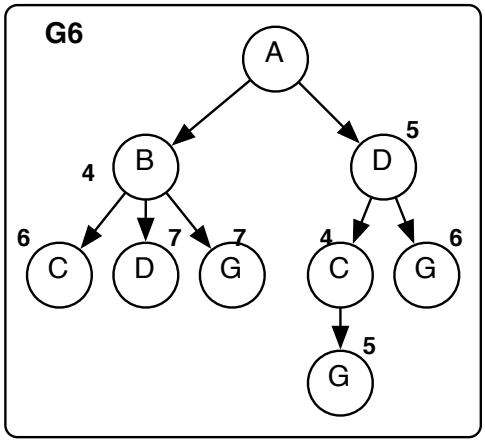


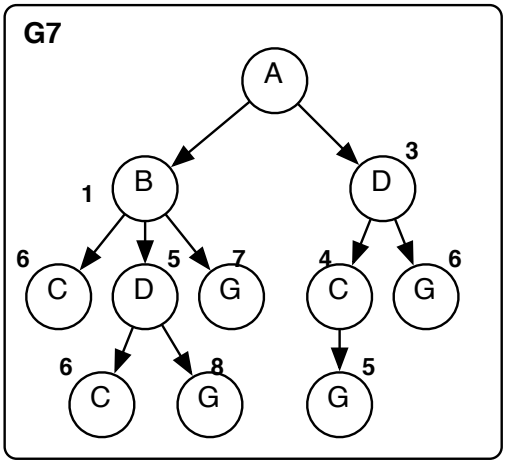












1.2 Algorithms

1. You are faced with a path search problem with a very large branching factor, but where the answers always involve a relative short sequence of actions (whose exact length is unknown). All the actions have the same cost. What search algorithm would you use to find the optimal answer? Indicate under what conditions, if any, a visited or expanded list would be a good idea.

   Progressive deepening (PD), with no visited or expanded list would probably be the best choice. All the costs are the same, so breadth-first (BF) and PD both guarantee finding the shortest path in that situation, without the overhead of uniform-cost search. Since the branching factor is high, space will be an issue, which is why we prefer PD over BF. If we were to use a visited list with PD, the space cost would be the same as BF and it would not make sense to pay the additional run-time cost of PD (repeated exploration of parts of the tree) if we give up the space advantage.

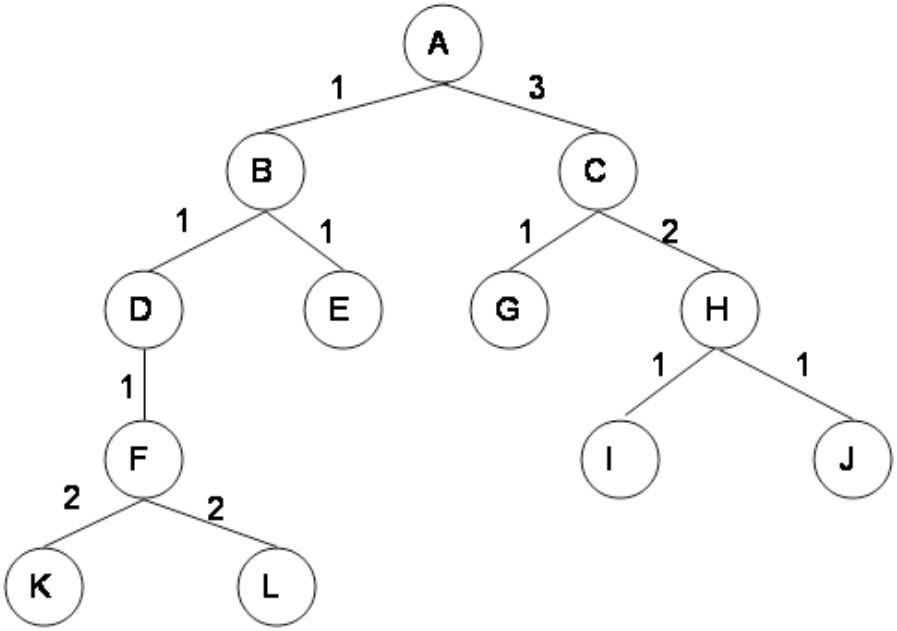
2. You are faced with a path search problem with a very large branching factor, but where the answers always involve a relative short sequence of actions (whose exact length is unknown). These actions, however, have widely varying costs. What search algorithm would you use to find the optimal answer? Indicate under what conditions, if any, a visited or expanded list would be a good idea.

   Since we have variable link costs, we should use Uniform Cost search to guarantee the optimal answer. The fact that the costs are highly variable is good, since we expect that we might be able to avoid exploring sub-trees with high cost. Note that we don't necessarily have a useful heuristic and so A* may not be applicable. Using an expanded list would make sense if the search space involves lots of loops, which would lead us to re-visit the same state many times. However, since we know that there's a relatively short path to the goal, it might not be worth the extra space.

6.034 Quiz 1, Spring 2004 — Solutions
Open Book, Open Notes

1 Tree Search (12 points)

Consider the tree shown below. The numbers on the arcs are the arc lengths.

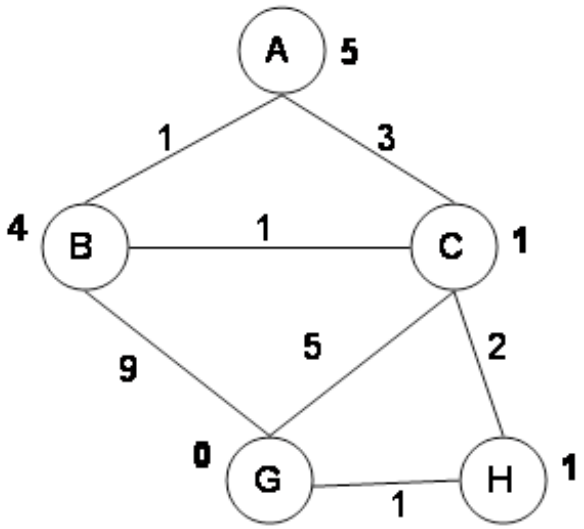


Assume that the nodes are expanded in alphabetical order when no other order is specified by the search, and that the goal is state G. No visited or expanded lists are used. What order would the states be expanded by each type of search? Stop when you expand G. Write only the sequence of states expanded by each search.

Search Type	List of states
Breadth First	A B C D E G
Depth First	A B D F K L E C G
Progressive Deepening Search	A A B C A B D E C G
Uniform Cost Search	A B D E C F G

2 Graph Search (10 points)

Consider the graph shown below where the numbers on the links are link costs and the numbers next to the states are heuristic estimates. Note that the arcs are undirected. Let A be the start state and G be the goal state.



Simulate A* search with a strict expanded list on this graph. At each step, show the path to the state of the node that's being expanded, the length of that path, the total estimated cost of the path (actual + heuristic), and the current value of the expanded list (as a list of states). You are welcome to use scratch paper or the back of the exam pages to simulate the search. However, please transcribe (only) the information requested into the table given below.

Path to State Expanded	Length of Path	Total Estimated Cost	Expanded List
A	0	5	(A)
C-A	3	4	(C A)
B-A	1	5	(B C A)
H-C-A	5	6	(H B C A)
G-H-C-A	6	6	(G H B C A)

3 Heuristics and A* (8 points)

1. Is the heuristic given in Problem 2 admissible? Explain.

   Yes. The heuristic is admissible because it is less than or equal to the actual shortest distance to the goal.

2. Is the heuristic given in Problem 2 consistent? Explain.

   No, the heurstic is not consistent. There are two places in the graph where consistency fails. One is between A and C where the drop in heuristic is 4, but the path length is only 3. The other is between B and C where the drop in heuristic is 3 but the path length is only 1.

3. Did the A* algorithm with strict expanded list find the optimal path in the previous example? If it did find the optimal path, explain why you would expect that. If it didn't find the optimal path, explain why you would expect that and give a simple (specific) change of state values of the heuristic that would be sufficient to get the correct behavior.

   A with a strict expanded list will not find the shortest path (which is ABCHG with cost 5). This is because the heuristic is not consistent. We can make the heuristic consistent by changing its value at C to be 3. There are other valid ways to make the graph consistent (change h(B) to 2 and h(A) to 3, for example) and those were right as well.*

4 Search problem formulation (10 points)

A Mars rover has to leave the lander, collect rock samples from three places (in any order) and return to the lander.

Assume that it has a navigation module that can take it directly from any place of interest to any other place of interest. So it has primitive actions *go-to-lander*, *go-to-rock-1*, *go-to-rock-2*, and *go-to-rock-3*.

We know the time it takes to traverse between each pair of special locations. Our goal is to find a sequence of actions that will perform this task in the shortest amount of time.

1. Formulate this problem as a search problem by specifying the state space, initial state, path-cost function, and goal test. Try to be sure that the state space is detailed enough to support solving the problem, but not redundant.

   - States: ⟨ *current-location, have-rock1?, have-rock2?, have-rock3?* ⟩
     These are state *variables*. The variable *current-location* ranges over the set {*lander, rock1, rock2, rock3* }. The other variables are binary.
   - Initial state: ⟨ *lander, no, no, no* ⟩
   - Path cost: sum of arc costs; arc cost = distance between locations
   - Goal test: ⟨ *lander, yes, yes, yes* ⟩

2. Say what search technique would be most appropriate, and why.

   We want a shortest path, so we need UCS or A. We might as well use A*, since it will probably be faster and there's a reasonable heuristic available.*

3. One possible heuristic evaluation function for a state would be the distance back to the lander from the location of the state; this is clearly admissible. What would be a more powerful, but still admissible, heuristic for this problem? (Don't worry about whether it's consistent or not.)

   This should have read "One possible heuristic evaluation function for a state would be the **amount of time** *required for the robot to go back to the lander from the location of the state..."*

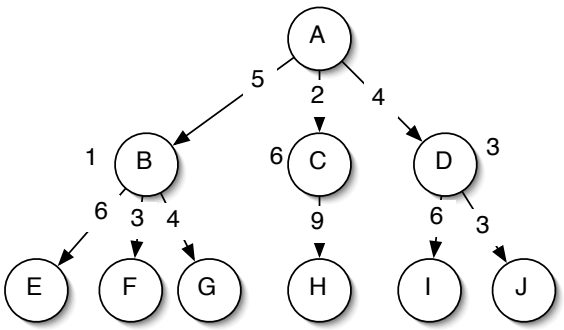
   So, because of the typo, we gave everyone a free two points on this problem.

   The answer we had in mind was the maximum, over uncollected rocks r, of the time to get from the current location to r, and the time to get from r to the lander.

6.034 Quiz 1, Spring 2003: Solutions v. 1.1
Open Book, Open Notes

1 Tree Search (10 points)

Consider the tree shown below. The numbers on the arcs are the arc lengths; the numbers near states B, C, and D are the heuristic estimates; all other states have a heuristic estimate of 0.

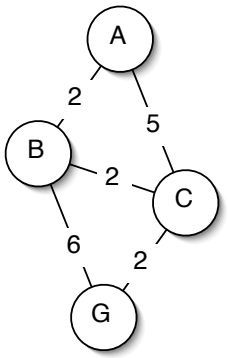


Assume that the children of a node are expanded in alphabetical order when no other order is specified by the search, and that the goal is state J. No visited or expanded lists are used. What order would the states be expanded by each type of search. Write only the sequence of states expanded by each search.

Search Type	List of states
Breadth First	A B C D E F G H I J
Depth First	A B E F G C H D I J
Progressive Deepening Search	A A B C D A B E F G C H D I J
Best-First Search	A B E F G D I J
A* Search	A B D J

2 Graph Search (8 points)

Consider the graph shown below. Note that the arcs are undirected. Let A be the start state and G be the goal state.



Simulate uniform cost search with a strict expanded list on this graph. At each step, show the state of the node that's being expanded, the length of that path, and the current value of the expanded list (as a list of states).

State Expanded	Length of Path	Expanded List
A	0	(A)
B	2	(B A)
C	4	(C B A)
G	6	(G C B A)

3 $A*$ Algorithm (12 points)

1. Let's consider three elements in the design of the $A*$ algorithm:

   - The heuristic, where the choices are:
     - **arbitrary** heuristic
     - **admissible** heuristic
     - **consistent** heuristic
   - History:
     - **none**
     - **visited** list
     - **strict** expanded list
     - **non-strict** expanded list
   - Pathmax
     - **Use** pathmax
     - **Don't use** pathmax

   In the table below, indicate all the combinations that *guarantee* that $A*$ will find an optimal path. Not all rows have to be filled. If multiple values works for any of Heuristic, History and Pathmax, independent of the other choices, you can write the multiple values in one row. So

Heuristic	History	Pathmax
A,B	C	D,E

   can be used to represent all of: A,C,D; A,C,E; B,C,D; and B,C,E.

Heuristic	History	Pathmax
Admissible	None, Non-Strict	Use, Don't Use
Consistent	None, Non-Strict, Strict	Use, Don't Use

2. In the network of problem 2, assume you are given the following heuristic values:

$$A = 5; B = 4; C = 0; G = 0$$

Is this heuristic:

- Admissible? **Yes** No
- Consistent? Yes **No**

Justify your answer very briefly.

It is admissible because it is always less than the length of the shortest path. It is not consistent because the difference between the heuristic values at B and C is 4, which is greater than the arc-length of 2.

3. With the heuristic above will A* using a strict expanded list find the optimal path?

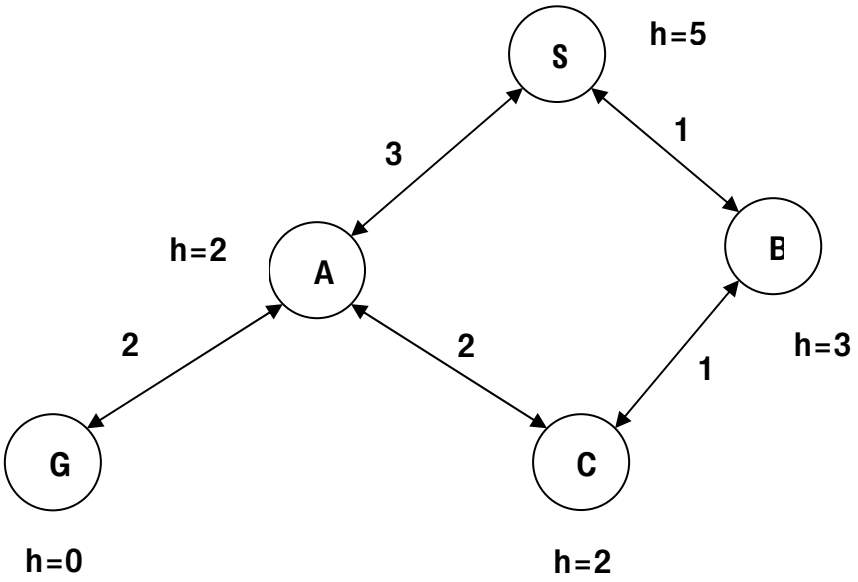
Yes **No**

Justify your answer very briefly.

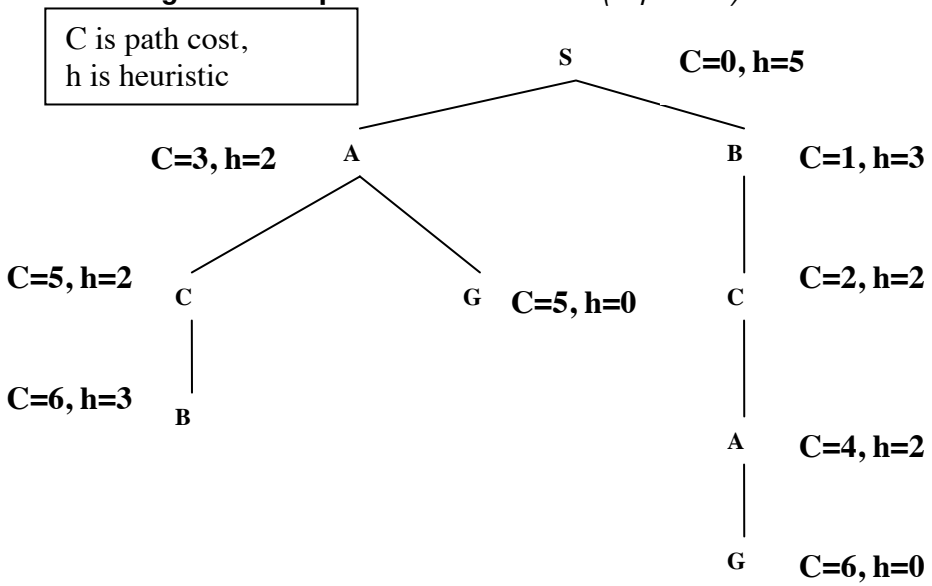
We will visit C first from A with estimated cost of 5, and because it's on the expanded list, even when we later find a path to C with estimated cost of 4, we won't expand it again.

Problem 1 – Search *(30 points)*

Below is a graph to be searched (starting at S and ending at G). Link/edge costs are shown as well as heuristic estimates at the states. You may not need all the information for every search.



Draw the complete search tree for this graph. Label each node in the tree with the cost of the path to that node and the heuristic cost at that node. When you need to refer to a node, use the name of the corresponding state and the length of the path to that node. *(5 points)*



s

For each of the searches below, just give a list of node names (state name, length of path) drawn from the tree above. Break ties using alphabetical order. *(2 points each)*

1. **Perform a depth-first search using a visited list. Assume children of a state are ordered in alphabetical order. Show the sequence of nodes that are expanded by the search.**

   S0, A3, C5, G5 note that B6 is not expanded because B is on visited list (placed there when S0 was expanded).

2. **Perform a best-first (greedy search) without a visited or expanded list. Show the sequence of nodes that are expanded by the search.**

   S0 (h=5), A3(h=2), G5(h=0)

3. **Perform a Uniform Cost Search without a visited or expanded list. Show the sequence of nodes that are expanded by the search.**

   S0, B1, C2, A3, A4, C5, G5 note that nodes are ordered first by cost then alphabetically when tied for cost.

4. **Perform an A* search (no pathmax) without an expanded list. Show the sequence of nodes that are expanded by the search.**

   S0(0+5), B1(1+3), C2(2+2), A3(3+2), G5(5+0)

Is the heuristic in this example

1. **admissible?** *Yes*

2. **consistent?** *No*

Justify your answer, briefly. *(3 points)*

All the h values are less than or equal to actual path cost to the goal and so the heuristic is admissible.
The heuristic drops from 5 at S to 3 at B while the path cost between S and B is only 1, and so the heuristic is not consistent.

For each of the following situations, pick the search that is most appropriate (be specific about visited and expanded list). Give a one sentence reason why you picked it. If you write a paragraph, we will not read it.

1. **We have a very large search space with a large branching factor and with possibly infinite paths. We have no heuristic. We want to find paths to the goal with minimum numbers of state.**

   Iterative deepening is the best choice, it uses little memory (like DFS) but guarantees finding the path with minimum number of states (like BFS).

2. **We have a space with a manageable number of states but lots of cycles in the state graph. We have links of varying costs but no heuristic and we want to find shortest paths.**

   Uniform Cost Search with a strict expanded list is the best choice, it guarantees finding shortest paths and the expanded list limits the cost to a function of the number of states, which is reasonable in this case. Recall that a visited list will interfere with the correct operation of UCS.

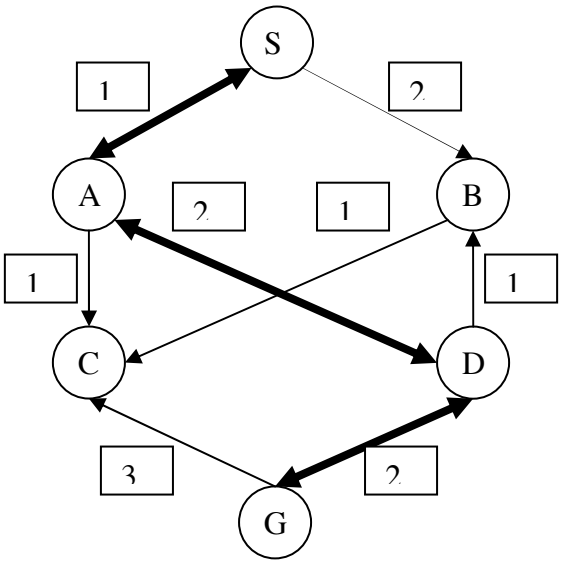
3. **Our search space is a tree of fixed depth and all the goals are the leaves of the tree. We have a heuristic and we want to find any goal as quickly as possible.**

   This has a typo which makes it ambiguous. If you read it as "all the leaves are goals", then depth-first search is the best choice (gets to the leaves fastest). If you read it as "all the goals are at the leaves", then the best choice is a greedy search (best first), which uses the heuristic to guide you to the part of the tree with the goals. In neither case is a visited or expanded list advisable since we are searching a tree (no loops).

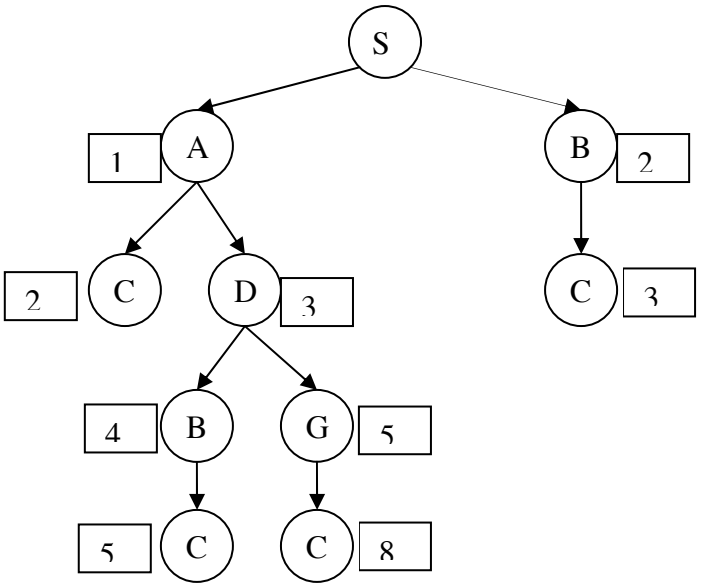
4. **We have a space with a manageable number of states but lots of cycles in the state graph. We have links of varying costs and an admissible heuristic and we want to find shortest paths.**

   This calls for A and a non-strict expanded list and, since we don't know that the heuristic is consistent, using pathmax. This allows us to use all the information we have and to avoid the extra cost due to cycles.*

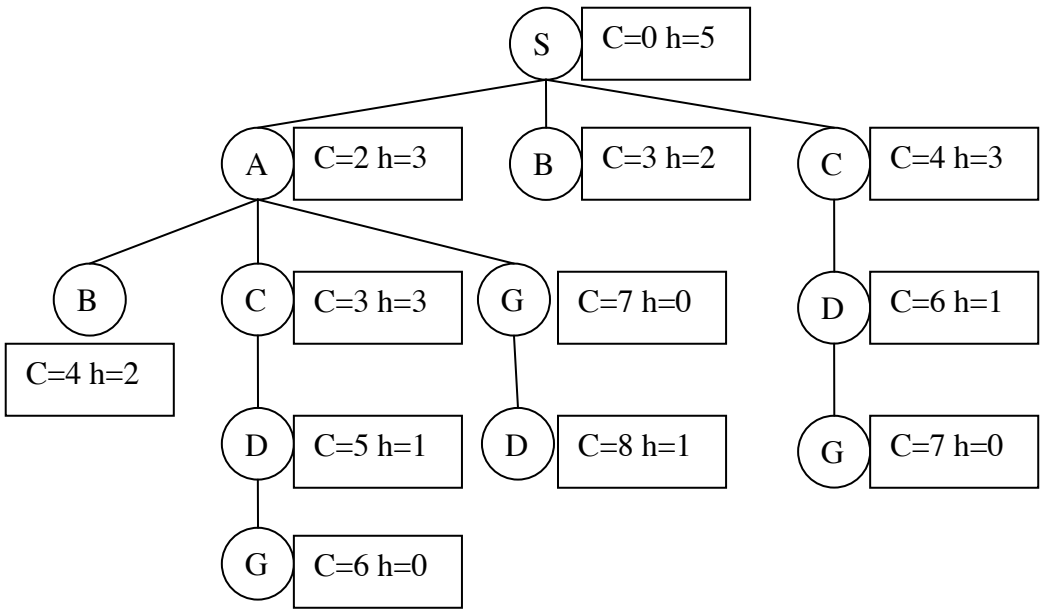
Problem 1: Search (25 points)



A. Construct the search tree for the graph above, indicate the path length to each node. The numbers shown above are link lengths. Pay careful attention to the arrows; some are bi-directional (shown thick) while some are uni-directional.



B. Using the following search tree, perform the searches indicated below (always from S to G). Each node shows both the total path cost to the node as well as the heuristic value for the corresponding state.



For each of the searches below, write the sequence of nodes **expanded** by the search. Specify a node by writing the name of the state and the length of the path (C above), e.g. S0, B3, etc. Break ties using alphabetical order.

1. Depth First Search (no visited list)

   S0, A2, B4, C3, D5, G6

2. Breadth First Search (with visited list)

   S0, A2, B3, C4, G7

3. Uniform Cost Search (with strict expanded list)

   S0, A2, B3, C3, D5, G6

4. A* (without expanded list)

   S0(+5), A2(+3), B3(+2), B4(+2), C3(+3), D5(+1), G6(+0)

C. Choose the most efficient search method that meets the criteria indicated below. **Explain your choice.**

1. You are given a state graph with link costs. The running time of the algorithm should be a function of the number of states in the graph and the algorithm should guarantee that the path with shortest path cost is found.

   UCS + expanded list
   UCS guarantees shortest paths, expanded list makes sure that the running time depends only on the number of states not the number of paths.

2. You are given a state graph with link costs and consistent heuristic values on the states. The running time of the algorithm should be a function of the number of states in the graph and the algorithm should guarantee that the path with shortest path cost is found.

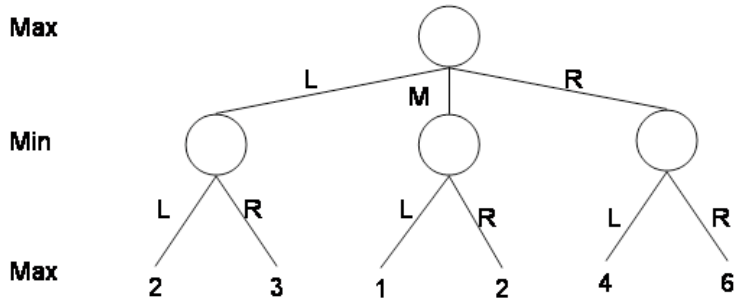
   A* + expanded list
   A* with consistent heuristic guarantees shortest paths, expanded list keeps the running time a function of number of states.

3. You are given a state graph with no link costs or heuristic values. The algorithm should find paths to a goal with the least number of states and the space requirements should depend on the depth of the first goal found.

   Iterative deepening
Guarantees minimum number of states on path to goal and the memory requirements are determined by the last depth-first search (at the level of the first goal found).

6 Game Search (10 points)

Consider the game tree shown below. The top node is a max node. The labels on the arcs are the moves. The numbers in the bottom layer are the values of the different outcomes of the game to the max player.



1. What is the value of the game to the max player?

   4

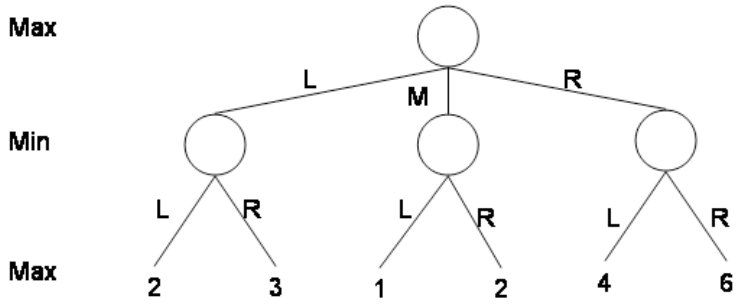
2. What first move should the max player make?

   R

3. Assuming the max player makes that move, what is the best next move for the min player, assuming that this is the entire game tree?

   L

4. Using alpha-beta pruning, consider the nodes from **right to left**, which nodes are cut off? Circle the nodes that are not examined.



   The nodes that are not examined are the left-most node labeled "2" and the node labeled "1."

6.034 Quiz 1, Spring 2005

1 Search Algorithms (16 points)

1.1 Games

The standard alpha-beta algorithm performs a depth-first exploration (to a pre-specified depth) of the game tree.

1. Can alpha-beta be generalized to do a breadth-first exploration of the game tree and still get the optimal answer? Explain how or why not. If it can be generalized, indicate any advantages or disadvantages of using breadth-first search in this application.

   No. The alpha-beta algorithm is an optimization on min-max. Min-max inherently needs to look at the game-tree nodes below the current node (down to some pre-determined depth) in order to assign a value to that node. A breadth-first version of min-max does not make much sense. Thinking about alpha-beta instead of min-max only makes it worse, since the whole point of alpha-beta is to use min-max values from one of the earlier (left-most) sub-trees to decide that we do not need to explore some later (right-most) subtrees.

   Some answers suggested that min-max inherently needs to go all the way down to the leaves of the game tree, where the outcome of the game is known. This is not true. Typically one picks some depth of look-ahead depth and searches to that depth, using the static evaluator to compute a score for the board position at that depth.

2. Can alpha-beta be generalized to do a progressive-deepening exploration of the game tree and still get the optimal answer? Explain how or why not. If it can be generalized, indicate any advantages or disadvantages of using progressive-deepening search in this application.

   Yes. Progressive-deepening involves repeated depth-first searches to increasing depths. This can be done trivially with min-max and alpha-beta as well, which also involve picking a maximum depth of lookahead in the tree. PD does waste some work, but as we saw in the notes, the extra work is a small fraction of the work that you would do anyways, especially when the branching factor is high, as it is in game trees. The advantage is that in timed situations you guarantee that you always have a reasonable move available.

6.034 Quiz 2, Spring 2006
Open Book, Open Notes

1 Propositional Proof (20 pts)

Given the following statements:

1. $P \vee Q$
2. $P \rightarrow R$
3. $Q \rightarrow S$

Prove that $R \vee S$ is entailed, using resolution refutation.

Use the table below. Fill in the Formulas in the proof; in the Reason field, give the parent clause numbers. Start by including the given formulas in the form needed for resolution. You do not need to specify a Reason for the given information. We have given you more than enough space for your proof. You do not need to fill in every line.

Step	Reason	Formula
1	Given	$P \vee Q$
2	Given	$\neg P \vee R$
3	Given	$\neg Q \vee S$
4	Negated Conclusion	$\neg R$
5	Negated Conclusion	$\neg S$
6	2 + 4	$\neg P$
7	6 + 1	Q
8	7 + 3	S
9	8 + 5	False

2 English to FOL (15 points)

Write the following statements in First Order Logic:

1. "Every city has a postman that has been bitten by every dog in the city."
   Use predicates:

   - $City(x)$ means x is a city
   - $Postman(x)$ means x is a postman
   - $Dog(x)$ means x is a dog
   - $Lives(x, y)$ means x lives in city y
   - $Bit(x, y)$ means x bit y

   $$\forall c.City(c) \rightarrow (\exists p.Postman(p) \land Lives(p, c) \land (\forall d.Dog(d) \land Lives(d, c) \rightarrow Bit(d, p)))$$

2. "All blocks supported by blocks that have been moved have also been moved."
   Use predicates:

   - $Block(x)$ means x is a block
   - $Supports(x, y)$ means x supports y
   - $Moved(x)$ means x has been moved

   $$\forall x.\forall y.Block(x) \land Block(y) \land Supports(x, y) \land Moved(x) \rightarrow Moved(y)$$

3 Logic semantics and interpretation (15 points)

Consider the following interpretation of a language with a unary predicates P, Q, and a binary predicate R.

- Universe $= \{1, 2, 3, 4\}$
- $P = \{\langle 1\rangle, \langle 3\rangle\}$
- $Q = \{\langle 2\rangle, \langle 4\rangle\}$
- $R = \{\langle 3, 2\rangle, \langle 4, 3\rangle, \langle 3, 1\rangle, \langle 4, 2\rangle, \langle 2, 1\rangle, \langle 4, 1\rangle\}$

Circle the sentences below that hold in that interpretation.

1. $\forall x.P(x)$
2. $\exists x.P(x)$ – Holds
3. $\exists x.P(x) \wedge Q(x)$
4. $\exists x.P(x) \rightarrow Q(x)$ – Holds
5. $\forall x.P(x) \rightarrow Q(x)$
6. $\forall x.P(x) \rightarrow \neg Q(x)$ – Holds
7. $\forall x.Q(x) \rightarrow \neg P(x)$ – Holds
8. $\forall x.\exists y.R(x, y)$
9. $\exists y.\forall x.R(x, y)$
10. $\forall x.P(x) \rightarrow \exists y.R(x, y)$
11. $\forall x.Q(x) \rightarrow \exists y.R(x, y)$ – Holds

6.034 Quiz 3 Solutions, Spring 2006
Open Book, Open Notes

1 Clause form and resolution (10 points)

1. Circle the correct clause form for this formula from the choices below.

   $\forall x.\neg(\exists y.P(y) \rightarrow Q(y)) \rightarrow R(x)$

   (a) $P(y) \vee R(x)$
   $\neg Q(y) \vee R(x)$

   (b) $P(f(x)) \vee R(x)$
   $\neg Q(f(x)) \vee R(x)$

   (c) $P(A) \vee R(x)$
   $\neg Q(A) \vee R(x)$

   (d) $\neg P(y) \vee Q(y) \vee R(x)$

   (e) $\neg P(f(x)) \vee Q(f(x)) \vee R(x)$
   - $\forall x.(\exists y.P(y) \rightarrow Q(y)) \vee R(x)$
   - $\forall x.(\exists y.\neg P(y) \vee Q(y)) \vee R(x)$
   - $\forall x.(\neg P(f(x)) \vee Q(f(x))) \vee R(x)$
   - $\neg P(f(x)) \vee Q(f(x)) \vee R(x)$

   (f) $\neg P(A) \vee Q(A) \vee R(x)$

2. Perform resolution on the following clauses; show the unifier and the result.

   $P(f(A), A) \vee \neg Q(f(B), x)$
   $P(f(y), x) \vee Q(x, g(x))$

   (a) Unifier:
   rename variables, so x in first clause is x_1 and x in the second clause is x_2, then $x_1/g(x_2), x_2/f(B)$ or, equivalently, $x_1/g(f(B)), x_2/f(B)$.

   (b) Result: $P(f(A), A) \vee P(f(y), f(B))$

2 Proof (20 points)

This proof encodes the following argument:

- Every integer has at most one predecessor.
- Two is the predecessor of Three.
- The predecessor of Three is even.
- Therefore, Two is even.

To actually carry this out, we also need some axioms about equality, i.e. equality is transitive, equality is symmetric and equals can be substituted in predicates (such as Even).

Fill in any missing Clauses; in the Reason field, give the parent clause numbers, also fill in all the unifiers, written as a set of variable/value bindings.

Step	Reason	Clause	Unifier
1	Given	$\neg Equals(x, y) \vee \neg Equals(z, y) \vee Equals(z, x)$	None
2	Given	$\neg Equals(x, y) \vee Equals(y, x)$	None
3	Given	$\neg Equals(x, y) \vee \neg Even(x) \vee Even(y)$	None
4	Given	$\neg Pred(z, x) \vee Equals(z, Sk1(x))$	None
5	Given	$Pred(Sk0, Three)$	None
6	Given	$Even(Sk0)$	None
7	Given	$Pred(Two, Three)$	None
8	Given	$\neg Even(Two)$	None
9	4,7	$Equals(Two, Sk1(Three))$	$z/Two, x/Three$
10	4,5	$Equals(Sk0, Sk1(Three))$	$z/Sk0, x/Three$
11	1,9	$\neg Equals(z, Sk1(Three)) \vee Equals(z, Two)$	$x/Two, y/Sk1(Three)$
12	10,11	$Equals(Sk0, Two)$	$z/Sk0$
13	3,8	$Equals(x, Two) \vee \neg Even(x)$	y/Two
14	6,13	$\neg Equals(Sk0, Two)$	$x/Sk0$
15	12,14	False	

3 FOL and Entailment (15 points)

Answer each of these questions with **no more than 4 sentences**.

1. Given two first-order logic sentences, A and B, how do you show that A entails B?

   To prove A entails B, we can use FOL Resolution by using A as the KB, introducing $\neg B$ and trying to reach a contradiction. While proof by FOL resolution is sound, it is only semi-decidable; if A does indeed entail B, this will eventually find a contradiction, but if it A does not entail B, it may loop forever.

2. Given two first-order logic sentences, A and B, how do you show that A does not entail B?

   To prove that A does not entail B, we need to find an interpretation for which A is true and B is not. Explicitly searching for such an interpretation is in general intractable (because the number of potential interpretations is infinite and even the size of the interpretation may be infinite).

3. What is it about a domain that would make you want to use first-order logic, rather than propositional logic?

   One situation where FOL is useful is for infinite (or very large) domains; there is no way to express infinite domains in propositional logic. FOL lets you express general statements about relationships among types of objects in the world (or objects based on their properties), rather than merely statements about individual objects themselves.

2. (8 points) For each group of sentences below, give an interpretation that makes the first sentence(s) true and the last sentence false. Use $\{A, B, C\}$ as your universe.

   (a)

   $$\exists x.p(x) \wedge q(x) \wedge r(x, x)$$
   $$\forall x.p(x) \rightarrow \exists y.\neg r(x, y)$$
   $$\forall x.p(x) \rightarrow \exists y.\neg x = y \wedge r(x, y)$$

   $$\forall x.p(x) \vee \neg q(x)$$

   Solution:

   $$\begin{aligned} p &= \{< A >\} \\ q &= \{< A, C >\} \\ r &= \{< A, A >, < A, B >\} \end{aligned}$$

   (b)

   $$\forall x.p(x) \leftrightarrow \exists y.r(y, x)$$
   $$\forall x.\exists y.r(x, y)$$

   $$\forall x.\neg p(x)$$

   Solution:

   $$\begin{aligned} p &= \{< A >\} \\ r &= \{< A, A >, < B, A >, < C, A >\} \end{aligned}$$

   Alternately: (there are others, too)

   $$\begin{aligned} p &= \{< A, B, C >\} \\ r &= \{< A, A >, < B, B >, < C, C >\} \end{aligned}$$

6.825: Final Exam

There are 130 points total. Points for individual problems are indicated in bold.

1 Search

(**10**) You're a taxi driver. Your taxi can hold 4 passengers. Passengers pay a flat fee for a ride to the airport, so goal is to pick up 4 passengers and take them to the airport in the smallest number of miles. Your world can be modeled as a graph of locations with distances between them. Some, but not all, of the locations have passengers that you can pick up.

a. Describe the state space of this search problem.

b. What would be a good cost function for this search problem?

c. Now, consider a case where passengers have to pay according to how far away they are from the airport when they're picked up (note: they don't pay according to how long a ride they take in your taxi, but according to the length of the shortest path from their pickup-point to the airport).

   Describe the state space of this search problem.

d. What would be a good cost function for this version of the problem? You still have a desire to save gas.

e. Is uniform cost search guaranteed to find the optimal solution in either or both versions of the problem? Why or why not?

2 FOL Semantics

(**6**) Consider a world with objects **A**, **B**, and **C**. We'll look at a logical languge with constant symbols X, Y, and Z, function symbols f and g, and predicate symbols p, q, and r. Consider the following interpretation:

- $I(X) = \mathbf{A}$, $I(Y) = \mathbf{A}$, $I(Z) = \mathbf{B}$
- $I(f) = \{\langle \mathbf{A}, \mathbf{B}\rangle, \langle \mathbf{B}, \mathbf{C}\rangle, \langle \mathbf{C}, \mathbf{C}\rangle\}$
- $I(p) = \{\mathbf{A}, \mathbf{B}\}$

- $I(q) = \{\mathbf{C}\}$
- $I(r) = \{\langle \mathbf{B}, \mathbf{A}\rangle, \langle \mathbf{C}, \mathbf{B}\rangle, \langle \mathbf{C}, \mathbf{C}\rangle\}$

For each of the following sentences, say whether it is true or false in the given interpretation I:

a. $q(f(Z))$

b. $r(X, Y)$

c. $\exists w.f(w) = Y$

d. $\forall w.r(f(w), w)$

e. $\forall u, v.r(u, v) \rightarrow (\forall w.r(u, w) \rightarrow v = w)$

f. $\forall u, v.r(u, v) \rightarrow (\forall w.r(w, v) \rightarrow u = w)$

3 Interpretations

(**6**) Using the same set of symbols as in the previous problem, for each group of sentences below, provide an interpretation that makes the sentences true, or show that it's impossible.

a.
- $\exists w.p(w) \wedge \exists w.q(w)$
- $\neg\exists w.p(w) \wedge q(w)$
- $\forall u.p(u) \rightarrow \exists v.r(u, v)$

b.
- $\forall u.\exists v.r(u, v)$
- $\exists u, v.\neg r(u, v)$
- $\forall v.(\exists u.r(u, v)) \leftrightarrow p(v))$

c.
- $\forall u, v.(p(v) \rightarrow r(u, v))$
- $\exists u, v.\neg r(u, v)$
- $\exists v.p(v)$

4 Unification

(**6**) For each pair of literals below, specify a most general unifier, or indicate that they are not unifiable.

a. $r(f(x), y)$ and $r(z, g(w))$

b. $r(f(x), x)$ and $r(y, g(y))$

c. $r(a, C, a)$ and $r(f(x), x, y)$

2 FOL Semantics

a. T

b. F

c. F

d. T

e. F

f. T

3 Interpretations

a.
- $I(p) = \{\mathbf{A}\}$
- $I(q) = \{\mathbf{C}\}$
- $I(r) = \{\langle \mathbf{A}, \mathbf{B} \rangle\}$

b.
- $I(p) = \{\mathbf{B}, \mathbf{C}\}$
- $I(r) = \{\langle \mathbf{A}, \mathbf{B} \rangle, \langle \mathbf{B}, \mathbf{B} \rangle, \langle \mathbf{C}, \mathbf{C} \rangle\}$

c.
- $I(p) = \{\mathbf{A}\}$
- $I(r) = \{\langle \mathbf{A}, \mathbf{A} \rangle, \langle \mathbf{B}, \mathbf{A} \rangle, \langle \mathbf{C}, \mathbf{A} \rangle\}$

4 Unification

a. $\{z/f(x), y/g(w)\}$

b. not unifiable

c. $\{a/f(x), x/C, y/f(x)\}$

5 Clausal Form

a. $r(f(y), y) \vee s(f(y), y)$

b. $\neg r(x, y) \vee p(y)$

c. $\neg r(f(y), y) \vee p(f(y))$

6 Operator Descriptions

a. $\forall s.on(s) \rightarrow \mathit{off}(\mathit{result}(\mathit{push}(s)))$ and $\forall s.\mathit{off}(s) \rightarrow on(\mathit{result}(\mathit{push}(s)))$

b. (Pre: on, Eff: off, ¬ on) and (Pre: off, Eff: on, ¬ off)

6.034 Quiz 2 Answers, Spring 2004

1 First-Order Logic (24 points)

Here are some English sentences and their translation into clausal form.

1. Every car has a driver.
$$D(f(x_1), x_1)$$
2. The driver of a car is in the car.
$$\neg D(x_2, y_2) \vee In(x_2, y_2)$$
3. "In" is transitive.
$$\neg In(x_3, y_3) \vee \neg In(y_3, z_3) \vee In(x_3, z_3)$$
4. Drivers are people.
$$\neg D(x_4, y_4) \vee P(x_4)$$
5. Chitty (a car) is in the Stata garage.
$$In(C, SG)$$
6. Therefore, there is a person in the Stata garage. (This clause is the negation of the conclusion).
$$\neg P(x_6) \vee \neg In(x_6, SG)$$

We'd like to prove the conclusion using resolution refutation. This proof is kind of tricky, so we're going to tell you, in English, what the steps should be. For each step, say which of the previous clauses (P1 and P2 in the table) it can be derived from using resolution, what the resulting clause is and what the unifier is.

Step	P1	P2	Clause	Unifier
7	Every driver is in their car. (one term)			
	1	2	$In(f(x_1), x_1)$	$\{x_2/f(x_1), y_2/x_1\}$
8	If a car is in some location, then its driver is in that location. (two terms)			
	3	7	$\neg In(x_1, z_3) \vee In(f(x_1), z_3)$	$\{x_3/f(x_1), y_3/x_1\}$
9	The driver of a car is a person. (one term)			
	1	4	$P(f(x_1))$	$\{x_4/f(x_1), y_4/x_1\}$
10	The driver of Chitty is in the Stata garage. (one term)			
	5	8	$In(f(C), SG)$	$\{x_1/C, z_3/SG\}$
11	There is no car whose driver is in the Stata garage. (one term)			
	6	9	$\neg In(f(x_1), SG)$	$\{x_6/f(x_1)\}$
12	False			
	10	11	Nil	$\{x_1/C\}$

9 Resolution Proof (15 points)

Prove a contradiction from these clauses using resolution. For each new step, indicate which steps it was derived from (in columns labeled P1 and P2) and what the unifier was. Note that A is a constant and b, c, d, x, y, u, v, w are all variables.

This is just an example answer; there were lots of orders in which this could be done.

Step	P1	P2	Clause	Unifier
1	XX	XX	$\neg P(x, f(x), y) \vee R(y, g(x))$	XXXXXXXXX
2	XX	XX	$\neg R(u, v) \vee \neg Q(v) \vee S(u, h(v))$	XXXXXXXXX
3	XX	XX	$Q(g(A))$	XXXXXXXXX
4	XX	XX	$\neg S(w, w)$	XXXXXXXXX
5	XX	XX	$P(b, c, h(d))$	XXXXXXXXX
6	1	5	$R(h(d), g(b))$	$\{x/b, c/f(x), y/h(d)\}$
7	2	6	$\neg Q(g(b)) \vee S(h(d), h(g(b)))$	$\{u/h(d), v/g(b)\}$
8	4	7	$\neg Q(g(b))$	$\{w/h(d), d/g(b)\}$
9	3	8	*False*	$\{b/A\}$
10				

Given the following clauses, do a resolution refutation proof. *(10 points)*

1. ¬P(x,f(x))∨¬R(f(x))∨¬Q(x,g(x))
2. ¬P(x2,y2)∨ Q(x2,y2)
3. ¬P(x3,y3)∨ R(y3)
4. P(A,x4) [Negated Goal]

Step	Parent	Parent	New Clause	MGU
5	*3*	*4*	*R(y3)*	*x3=A, y3=x4*
6	*2*	*4*	*Q(A, y2)*	*x2=A, y2=x4*
7	*1*	*5*	*¬P(x,f(x))∨¬Q(x,g(x))*	*y3=f(x)*
8	*6*	*7*	*¬P(A,f(A))*	*x=A, y2=g(x)*
9	*4*	*8*	*( )*	*x4=f(A)*

or

Step	Parent	Parent	New Clause	MGU
5	*1*	*4*	*¬R(f(A)) ∨¬Q(A,g(A))*	*x=A, x4=f(x)*
6	*3*	*5*	*¬P(x3,f(A)) ∨¬Q(A,g(A))*	*y3=f(A)*
7	*2*	*6*	*¬P(x3,f(A)) ∨¬P(A,g(A))*	*x2=A, y2=g(A)*
8	*4*	*7*	*¬P(A,g(A))*	*x3=A, x4=f(A)*
9	*4*	*8*	*( )*	*x4=g(A)*

There are other possibilities as well. Note that (whenever possible) you want to use the shortest clauses in the resolution steps.

C. Given the following clauses:

1. Hasjob(p, job(p))
2. ¬ Hasjob(p, k) ∨ Equal(job(p), k)
3. Hasjob(George, Fireman)
4. ¬ Equal(Fireman, Teacher)
5. ¬ Equal(x,y) ∨ ¬ Equal(y, z) ∨ Equal(x, z)
6. ¬ Equal(x,y) ∨ Equal(y,x)

Prove by resolution refutation that:
¬ Hasjob(George, Teacher)

Hint: think about the strategy for the proof before you start doing resolutions. How would you prove the result by hand?

Step	**Parent**	**Parent**		**Unifier**
7	Neg	Goal	Hasjob(George, Teacher)	----------
8	2	7	Equal(job(George), Teacher)	p=George k=Teacher
9	2	3	Equal(job(George), Fireman)	p=George k=Fireman
10	9	6	Equal(Fireman, job(George))	x=job(George) y=Fireman
11	5	10	¬ Equal(job(George),z) ∨ Equal(Fireman, z)	x=Fireman y=job(George)
12	8	11	Equal(Fireman,Teacher)	z=Teacher
13	4	12	Contradiction	
14				