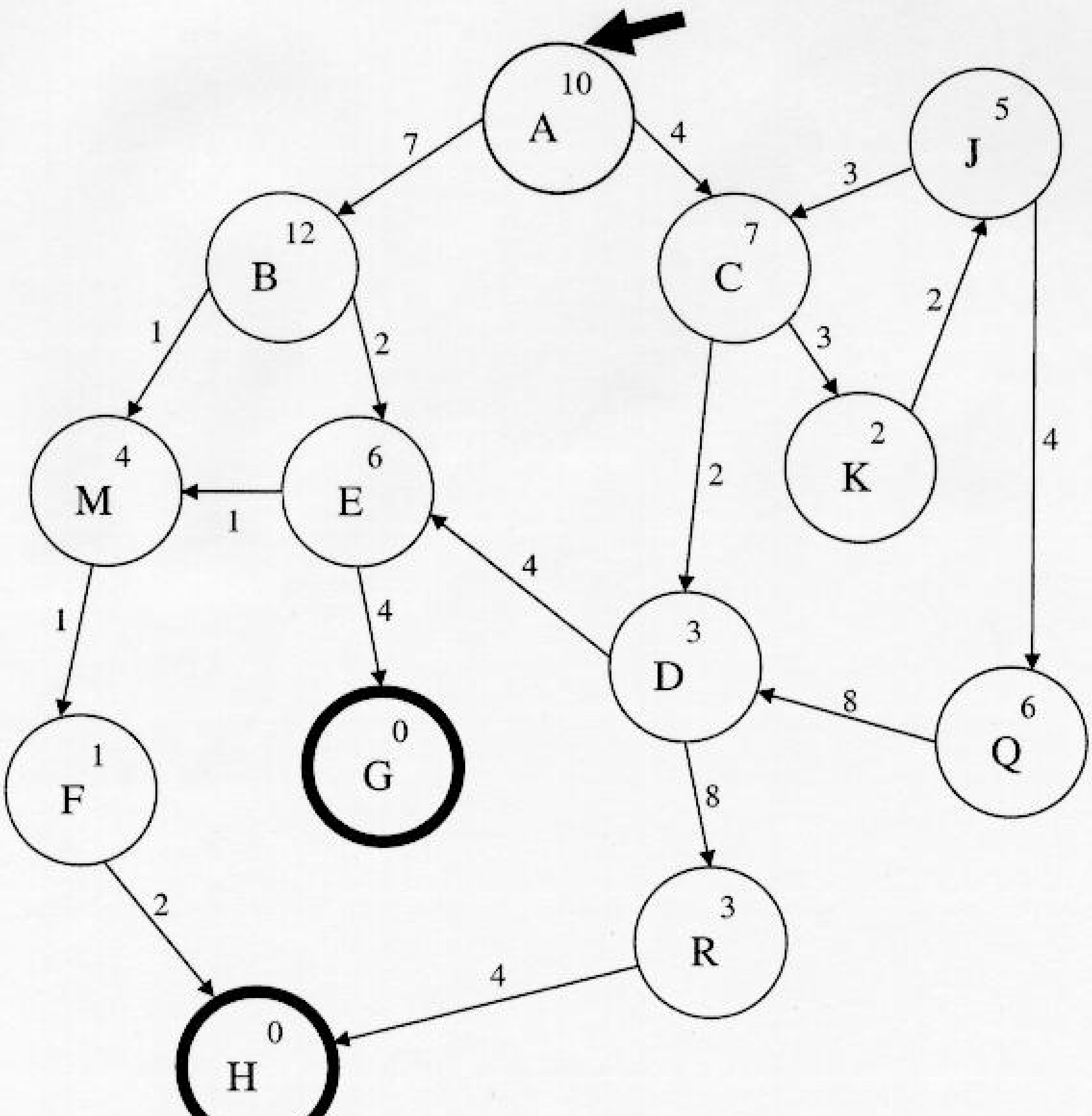
C1 Artificial Intelligence (25 Points)
Search

a) (6 Points) Suppose that h1 and h2 are both admissible heuristics and that h1 dominates h2 (i.e., for every node n, $h1(n) \geq h2(n)$). Prove that Algorithm A^* with heuristic h1 expands only a subset of the nodes expanded by Algorithm A^* with heuristic h2.

b) (12 Points) Consider the following graph. The number inside each node is h – the estimate of the cost of a least cost path from the node to a goal node. The actual cost of the operator that generates a successor of a node is shown on the arrow from the parent to the successor. For example, 10 is the estimate of the cost of the minimal cost path from A to a goal node; the cost of generating B from A is 7; the cost of generating C from A is 4. Notice that A is the start node, and G and H are goal nodes.



C1 Artificial Intelligence (Continued)

i. Is h an admissible heuristic? Explain why or why not.

ii. Using Iterative Deepening A* (IDA*) Search, list the order in which the nodes would be explored in the graph on each iteration of the search. The search should use the usual $f(n) = g(n) + h(n)$ where $g(n)$ is the actual cost of the path from the start to the node n, and $h(n)$ is the estimate of the cost of the minimal cost path from n to a goal node.

c) (7 Points) Consider Simplified Memory-Bounded A* Algorithm (SMA*).

i. What advantage does SMA* have over IDA*?

ii. Describe the Operation of SMA*.

C2 Artificial Intelligence (25 points)
Search

To interpret simple line drawings of trihedral solids—objects for which exactly three planar surfaces meet at each vertex—requires the correct classification of each visible vertex and edge segment as one of several possible types reflecting a physically plausible arrangement.

Shown below in Figure 1a are the four types of vertex for trihedral objects, which are distinguished by how many octants surrounding them are *filled* by solid object as opposed to empty space (object sides have been shaded here for clarity, but shading is omitted in general). Each number in the figure indicates the number of filled octants surrounding that vertex. There are also four interpretations of each line in such drawings. *Folds* are where two visible surfaces meet, either at a convexity or a concavity. Convex folds are labeled with a "+" and concave ones with a "-" (an example object is labeled in Figure 1b below). *Blades* are where two surfaces meet but only one is visible; on the other side of the line is background. Which side is which is denoted with an arrow ">", where the background is on the left as one moves in the direction of the arrow along the line.

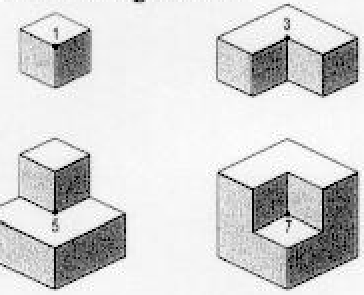


Figure 1a

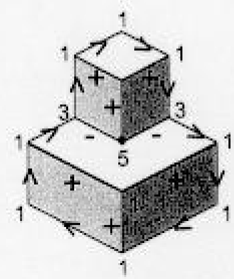


Figure 1b

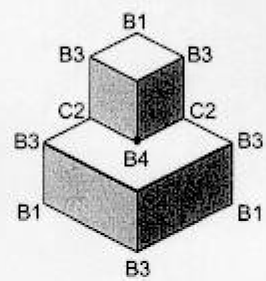


Figure 1c

Let a *junction* be a subset of the lines in a figure which intersect one another. (We will restrict our attention to trihedral objects for which the point of intersection of a junction is a vertex, though this is not true in general). Only a small subset of the possible labelings of edges which meet at a given junction make physical sense. The 12 physically plausible junctions are shown below in Figure 2. These can be broken into three types: "L" junctions, which consist of two intersecting lines (A1-A4, B1-B2 in the figure); "W" junctions, which consist of three intersecting lines with one angle > 180 degrees (B3-B4, C1), and "Y" junctions, which consist of three intersecting lines with all angles < 180 degrees (C2-C4).

A given junction's type can be determined by simple inspection. If it is an "L," it can have 6 possible labels; if a "W," it can have 3; and if a "Y," it can also have 3. A *labeling* of a given figure thus results from assigning a label of the correct type to each junction on the object. Since adjacent junctions share an edge, they may disagree on the label of the shared edge and thus be inconsistent. For example, neighboring junctions labeled A2 and C4 are inconsistent no matter which of their edges is shared because the former labels every edge with ">" and the latter with "+", which are not the same. A *consistent* labeling of an object is one in which all junction labels agree with their neighbors. A consistent labeling of the example object using the junction notation is shown in Figure 1c.

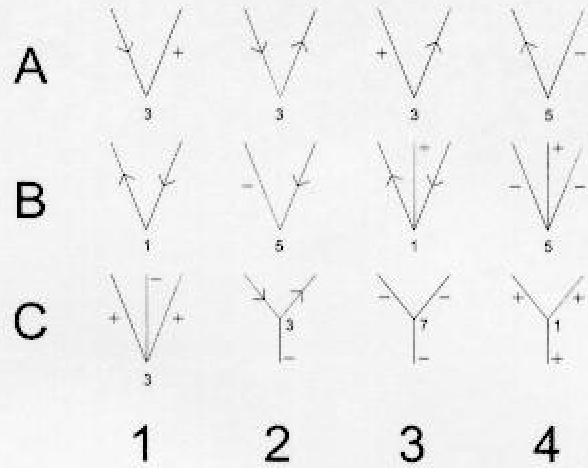


Figure 2

(a) (5 points) If you wanted to look at all possible labelings of the object in Figure 1b in order to count the number of consistent ones, how many different labelings would you have to evaluate? What is the general time complexity of such a "generate and test" algorithm for finding all consistent labelings?

(b) (5 points) Suppose you are searching the space of labelings to find one that is consistent using an iterative improvement method like gradient descent or simulated annealing. Briefly explain the *min-conflicts* heuristic and how it might be applied to the problem of consistent labeling.

(c) (15 points) Consider the cube object in the upper-left hand corner of Figure 1a. Two random junction labelings are shown below:

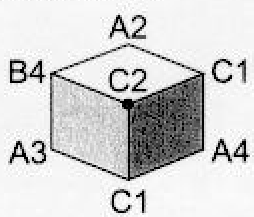


Figure 3a

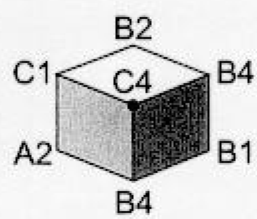


Figure 3b

(i) If you are using simulated annealing with the min-conflicts formula you gave in **(b)** as the energy E to be *minimized*, a transition probability of $e^{-\Delta E/T}$, and an annealing schedule of $T = 0.9t$ (where t is time and T is temperature), what would the probability be at $t = 5$ of accepting a move from the labeling in Figure 3a to that in Figure 3b? From Figure 3b to 3a?

(ii) If you are using gradient descent with the min-conflicts heuristic to change exactly one junction label per step, show the next labeling that would be visited starting from Figure 3b. Clearly indicate which junction label changes and how many conflicts are removed. Could there be any problems with local minima using this approach? How might you address them if there are?

C3 Artificial Intelligence (25 Points)
Learning

You are trying to use machine learning to build a decision tree for deciding whether a course is good or bad. You are given a set of courses from the past with an evaluation of Good or Bad for each, along with the values of five attributes: LEVEL, TIME, PROFESSOR, TOPIC, and SIZE.

a) (13 points) Suppose that in constructing your decision tree, 24 of the training examples reach the node N and that you have a choice of using attribute TIME or attribute TOPIC to split the node. Suppose further that the TIME attribute distributes the 24 examples as follows:

EARLY	MID-DAY	LATE
Good – 3	Good – 2	Good – 4
Bad – 5	Bad – 6	Bad – 4

Further suppose that the attribute SIZE distributes the 24 examples as follows:

< 10 Students	10-30 Students	31-50 Students	> 50 Students
Good – 1	Good – 6	Good – 1	Good – 1
Bad – 7	Bad – 2	Bad – 3	Bad – 3

Using information theory, determine whether you should use TIME or SIZE as the attribute for decision-making at node N. Give any formulae that you are using and show the details of your work. (You must give the formulae and insert the correct values into them, but you need not carry out all of the computations – however, be sure to state how you would make the final decision about which attribute to use.)

b) (6 points) What is decision tree pruning and what problem does it attempt to overcome? How is information theory used in decision tree pruning (be specific).

c) (6 points) Consider a decision tree constructed by our decision tree learning algorithm on a training set. We can prune the tree by taking an almost-terminal node in the tree (one that is right above a leaf) and eliminating it, collapsing its children into a single leaf (with the class at the new leaf being the most frequent class in the training instances corresponding to that leaf). Consider the following options:

I. This operation can (strictly) decrease the accuracy.
II. This operation must (strictly) decrease the accuracy.
III. This operation can (strictly) increase the accuracy.
IV. This operation must (strictly) increase the accuracy.

- Interpreting "accuracy" as accuracy on the original training set, which of the above statements are true? State all that apply and justify your response.
- Interpreting "accuracy" as accuracy on an independent test set, which of the above statements are true? State all that apply and justify your response.

C4 Artificial Intelligence (25 Points)
Planning

a) (8 Points) Plan Operator Representation

Consider the action of toggling a light stitch whose effect is to turn the light on if it is currently off, and to turn the light on if it is currently off. Assuming that the predicates in our language are *On(x)* and *Off(x)*, is it possible to provide a pure STRIPS description of the toggle operation as a single operator? If so, write that operator. If it is not, explain the difficulty and show how the behavior could be captured using a set of STRIPS operators.

b) (17 Points) POPs Planning

In Stanford's amazing new VRISC machine, there are only three registers, called A, B, and C. Initially they contain 1, 2, and 3 respectively. There is only one instruction, called *Assign(x, y)*, which copies the value contained in register *y* into register *x*. The only predicate we need to describe the situation is *Value(x, v)*, which says that register *x* contains value *v*. We would like to use a partial-order planning algorithm to construct a plan to switch the contents of A to B.

i. Show the STRIPS representation of the Assign operator with its precondition(s) and effect(s).

ii. Leaving about 3 inches of empty space in the middle, draw the initial plan for this problem, using the standard pictorial notation.

iii. Now the planner decides to achieve the goal condition that B should contain a 1. Add an *Assign* step to your diagram (copying it over from the previous step) so that this condition is achieved. *Make sure to bind variables as appropriate.*

iv. Now suppose that the planner decides to achieve the precondition of this *Assign* step by connecting it to the *Start* step. Update your diagram (by copying it over from the previous step) to reflect this, marking any changes in bindings.

v. Now suppose the planner decides to achieve the other goal condition, namely that A should contain a 2. Explain in words (with changes to your diagram if that makes it easier to explain) the sequence of events that occur when it adds a new *Assign* step to achieve this, up to the point where the addition of the step with all of its ramifications is complete.