CS573 — Artificial Intelligence I
Fall 2001
Assignment #1
Due Friday, Oct. 12th in class

This homework may be too hard or too easy. Don't panic! I will adjust future assignments based on your reactions to this one.

1. **Abstraction.**

   (a) R&N suggest a compact representation of the Missionaries and Cannibals problem (pg. 67). How many states are their in this representation?

   (b) Suppose each person is explicitly represented by name in the state description. Now how many states are there?

   (c) *You should think about the following question, and be prepared to discuss it in class; you do not need to hand in your answer.* How might the problem of task of creating an abstract representation of a problem be cast as a state-space search problem?

2. **Search space formulation.** Consider the problem of a student planning an entire course of studies at UW in advance. The desired outcome is an assignment of classes to quarters (1st quarter, 2nd quarter, *etc.*) which satisfies the requirements of the major. To simplify the problem, assume the following:

   - There is some set of classes C such that the student must take each of the classes in C exactly once. (*I.e.*, there are no alternatives of the form: you must take either CSE473 or CSE432.)
   - For each class c_i there is a (possibly empty) set $P_i \subset C$ of prerequisite classes that the student must take strickly before she takes c_i.
   - The student must take between one and four classes each quarter.

   (a) Provide a detailed representation of the search space. Your description should provide an exact specification of the components of the state description (state space, initial state, operators, goal test), of the constraints under which each operator can be applied, and the effects of each operator. You may use either English or pseudocode. Make sure that your formulation of the search space has no problem with repeated states, *i.e.*, that no node can be reached from the initial state by two different paths.

   (b) Somewhat shortsightedly, the student's only aim is to complete the degree in the shortest possible time, thereby minimizing the amount of money spent on tuition. Tuuition is charged per quarter, not per unit, and all quarters cost the same amount. Specify a cost function for the operators in your space so that the cost of a solution will correspond to the student's preferences.

(c) Construct a reasonable and admissible heuristic $h()$ for this search space, given your cost function. (For example, a zero heuristic function is admissible but not reasonable; for different reasons, a heuristic function that computes the optimal schedule is not reasonable either.) Hint: as discussed in the book, try and relax some of the constraints on your operators.

3. **A* and search.** Read the section in R&N on **bidirectional search** (pg. 80). The idea of bidirectional search is to search both forward from the initial state A and backward from the goal B, and stop when the two searches intersect: *i.e.*, when a node expanded from one direction and a node expanded from the other both correspond to the same state s, thus generating a complete path from the inital state to the goal by merging those two paths. Typically, bidirection search is breadth first, which is guaranteed to be complete and optimal, but often quite inefficient.

   We assume that the can compute both the predecessors and successors of a node, and that all edges have cost 1.

   Suppose we try to use bidirectional search with A*. In other words, assume we have an admissible heuristic h_B which estimates the distance from any state to the goal B, and another heuristic h_A which estimates the distance from any state to the initial state A.

   The forward A* search uses $f_A(n) = g_A(n) + h_B(n)$, where $g_A(n)$ is the actual length of the shortest path currently found from A to n.

   The backward A* search uses $f_B(n) = g_B(n) + h_A(n)$, where $g_B(n)$ is the actual length of the shortest path currently found from B to n.

   Assume that both f_A and f_B are monotonic (see pg. 97).

   - Is bidirectional A* complete? Why or why not?
   - Show that bidirectional A* is not optimal, by giving an example where s is the state where the two searches meet, but the path made by conjoining the path from A to s and s to B is not a shortest path from A to B.
   - Now, we are going to let the algorithms pool some information, in order to let them verify whether a generated path from A to s to B is optimal. State a simple sufficient condition for the path from A to s to B to be optimal. This condition should be something that the two algorithms can verify given the information they have at the time s is found. There is no need to prove your condition; state it precisely. Hint: consider what is going on in $Heap_A$, the priority queue used by the forward search, and $Heap_B$, the priority queue used by the backwards search.