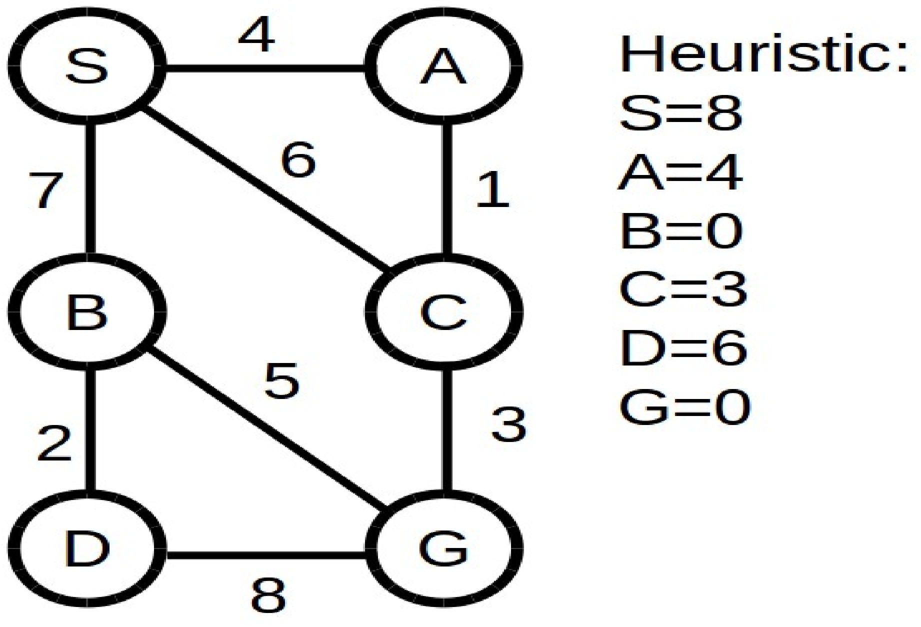
4511W, Fall-2017
ASSIGNMENT 2:
Assigned: 10/1/17 Due: Sunday 10/8/17 at 11:55 PM Submit on moodle (in a zip if you have multiple files)

Written/drawn:

Problem 1. (15 points)
Run A* on the following graph with the shown heuristics. The initial state is "S" and the goal state is "G". Path costs are shown on the edges. You should show at every step (1) the fringe nodes and their associated f-cost and (2) the nodes you have fully explored.

After running the A* search, prove or disprove whether the heuristic given is (3) admissible and/or (4) consistent.



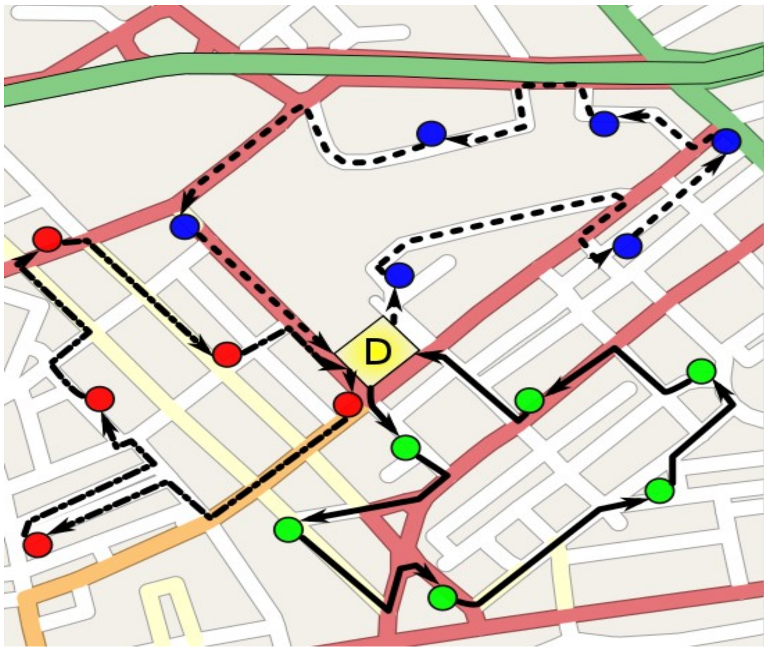
Problem 2. (20 points)
Prove or disprove the following:

(1) If h_1 and h_2 are admissible heuristics, then $h_3 = (h_1+h_2)/2$ is also admissible.

(2) If the optimal cost from the initial state to a goal is C in some problem, then A* will never expand any node N with f(N) > C if the heuristic is admissible.

Problem 3. (15 points)
Find an admissible non-trivial heuristic (i.e. not h(n) = 0, or something similarly trivial) for the problems below. For all of the problems, assume you are using an incremental problem description.

(1) UPS needs to send packages from a depot to houses using a fixed number of trucks. The picture below is where the depot ("D") needs to send 17 packages (each destination is a circle) with 3 trucks (red, blue and green colorings). The trucks need to choose which houses and in which order they are going to visit. After delivering to all the houses, the trucks must return to the depot. The goal is to minimize the distance traveled by all the trucks. You do not need to worry exact driving instructions, you can assume you know the distance between all locations. The main focus here is the ordering each truck visits the houses.



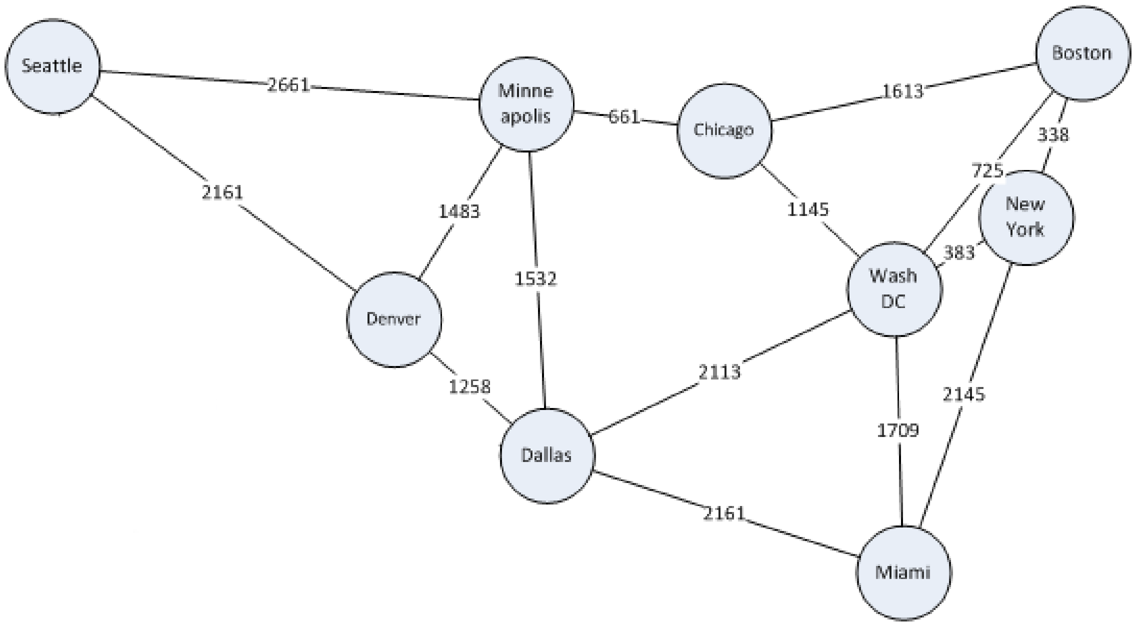
(2) You need to take 40 more credits to finish your major (each class is 4 credits). You cannot take two classes that have lectures at the same time. Some some classes are not offered every semester and some classes have prerequisite classes that you must take first. Describe what heuristic you can use to find a schedule that lets you graduate as fast as possible (fewest number of semesters) without taking more than 16 credits per semester.

(3) Finding a solution to Sokoban (see link below). The goal is to move crates onto designated squares where you can only move crates when next to them and you can only push them forwards (i.e. you and the crate must move in the same direction and you will end up where the crate was before moving). The goal is to solve the puzzle in as few moves as possible.
https://en.wikipedia.org/wiki/Sokoban

Problem 4. (20 points)
Show a step-by-step process of finding the goal (11 New York) from the initial state (1 Seattle) using iterative-deepening depth first search. At each step you must provide: (1) current state of explored nodes, (2) current state of fringe/frontier nodes (3) maximum depth of the search. You must also be consistent in the order you add nodes to your fringe list (clockwise or counter clockwise).

Note: Treat this graph as a tree (i.e. search "Minneapolis" via both "Seattle -> Minneapolis" and "Seattle -> Denver -> Minneapolis" if applicable).

Note2: You do not need to search cycles (and can exclude them to save time). So "Seattle -> Denver -> Minneapolis -> Seattle" does not need to be searched. Nor does "Seattle -> Minneapolis -> Seattle".



(More on next page)

Programming (python):

The book provides code for the algorithms presented. For this class, we will use the python version of the code. Download the python code here:
https://github.com/aimacode/aima-python

The code requires python3 to run (but some of the tests are written for python 2). For this assignment, you will need to know how to use the implemented genetic algorithm. Of note are:
/root/search.py
/root/tests/test_search.py

I was having trouble running test_search.py directly, but it will be useful to reference.

Problem 5. (30 points)
Compare solutions to the n-queens problem between an incremental depth-first-search and a complete-state genetic algorithm. For this we will assume that "n" is both the board size and the number of queens placed. The incremental depth-first-search is the n-queens problem provided here (also next to this pdf on the webpage):
http://www-users.cselabs.umn.edu/classes/Fall-2017/csci4511/assignments/nqueens.cpp

To compile the C++ file on a cselabs machine simply type:
```
g++ nqeens.cpp
```

To get the run-time of code on a cse-labs machine put "time" before the program, so for example:
```
C++: time ./a.out
python: time python3 myFile.py
```

... then look for this in the output:

```
8.736u 0.000s 0:11.75 91.3% 0+0k 0+0io 0pf+0w
```

This corresponds to an 8.736 second run-time(of computation).

Part 1. Run nqueens.cpp one time each for n = 8, 10, 11, 12 and 13. Record the run times.

Part 2. Run the genetic algorithm (read instruction for AIMA code above) with a population of 100 (100 different "things" reproducing each generation) and a mutation rate of 10%. Stop after making 1000 generations and show (1) the fitness and (2) the run-time.

Show these two pieces of information for **five trials** when n = 8, 10 and 11, 12, 13, 15, 20. Also, for each "n", provide the fitness value of an actual solution.

Part 3. Write a short paragraph analyzing the strengths and weaknesses of both depth first search and genetic algorithms. Use your results from Part 1 and 2 to support your arguments. You can assume that nqueens.cpp takes hours to find n=14.