Problem Set 1 (20% of your total score)

Different Searches, Heuristic, Local Search

Deadline: July 5th, 11:55pm, 2020.
Perfect score: 100.

Assignment Instructions:

Team Work: The problem set can be complete by a group of maximum two people. There is no solo bonus.

Submission Rules: Submit your reports electronically as a PDF document through Sakai (`sakai.rutgers.edu`). Do not submit Word documents, raw text, or hardcopies etc. Make sure to generate and submit a PDF instead. Each team of students should submit only a single copy of your solutions and indicate all team members on their submission. Failure to follow these rules may result in lower grade for the problem set.

Late Submissions: You are supposed to finish your work before the deadline specified at the top. Submissions after the deadline are not acceptable.

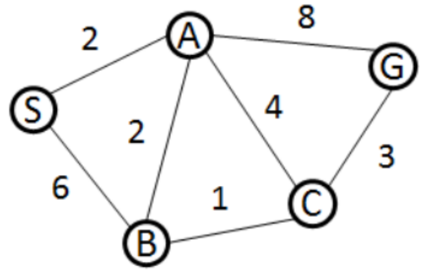
Score Calculation: The total score in the problem set may not sum up to 100. You total score will scale up to 100 and then be multiplied by the weight of the problem set (20 % for this one).

Extra Credit for LaTeX: You will receive 3 extra credit points on your original score (before scale up and multiplication) if you submit your answers as a typeset PDF (using LaTeX, in which case you should also submit electronically your latex source code). Unlike projects, in the problem sets, you are allowed to scan your handwritten work. If you choose to, make sure your scanned work is legible for TAs to grade, otherwise you are responsible for the points lost due to illegibility of your work.

Collusion, Plagiarism, etc.: Each team must prepare its solutions independently from others, i.e., without using common notes, code or worksheets with other students or trying to solve problems in collaboration with other students. You must indicate any external sources you have used in the preparation of your solution. Do not plagiarize online sources and in general make sure you do not violate any of the academic standards of the department or the university. Failure to follow these rules may result in failure in this problem set.

Other rules: For other grading rules, please refer to the homepage of our course website on sakai. Thanks.

Problem 1 [70 points, each search 7 points]: Fig. 1 provides a graph with associated edge costs and a table with values of the heuristic function. The adjacency list for a node is ordered alphabetically, i.e., node A has the adjacency list <B,C,G,S>. Using the tree-search and graph-search algorithms from our class (also provided below), manually execute BFS, DFS, uniform-cost, greedy best-first, and A* from the starting node S to the goal node G. For each of the 10 combinations, provide as your answer a search tree, the nodes on the queue (frontier), and the cost of the associated solution. For tree-search, if an infinite loop is encountered, you only need to provide a search tree with enough depth to demonstrate the infinite loop. You may draw your search tree manually and scan them for submission.



State	$h(x)$
S	7
A	6
B	2
C	1
G	0

Figure 1: A graph with associated edge costs and heuristics

```
Tree-Search(G, x_I)
AddToQueue(x_I, Queue);
while(!IsEmpty(Queue))
  x ← Front(Queue);
  if(x is goal) return solution;
  for each successor n_i of x
    AddToQueue(n_i, Queue);
return failure;
```

Figure 2: Pseudocode for tree-search

```
Graph-Search(G, x_I)
AddToQueue(x_I, Queue);
while(!IsEmpty(Queue))
  x ← Front(Queue);
  if(x.expanded) continue;
  x.expanded ← true;
  if(x is goal) return solution;
  for each successor n_i of x
    if(!n_i.expanded) AddToQueue(n_i, Queue);
return failure;
```

Figure 3: Pseudocode for graph-search

Problem 2 [18 points, each subproblem 6 points]: Which of the following are admissible, given admissible heuristics h_1, h_2? Which of the following are consistent, given consistent heuristics h_1, h_2?

- $h(n) = min\{h_1(n), h_2(n)\}$
- $h(n) = wh_1(n) + (1 - w)h_2(n),$ where $0 \leq w \leq 1$
- $h(n) = max\{h_1(n), h_2(n)\}$

Problem 3 [12 points]: Four-Queens problem. Consider the four-queens setup given in Fig. 4. Apply hill-climbing(in this case, hill-descending) to the problem until no more moves are possible. The objective function value of a given setup is equal to the number of pairs of attacking queens without considering blocking. For example, the top-left queen in Fig. 4 is attacking all three other queens. Each queen is only allowed to move in the column she belongs. For breaking ties, if there are multiple choices for moving the queens, first use the left most column with the best move and then favor the least number of blocks moved. You need to show **at each iteration** the objective function value **for all 16 positions** and the move that you will make.

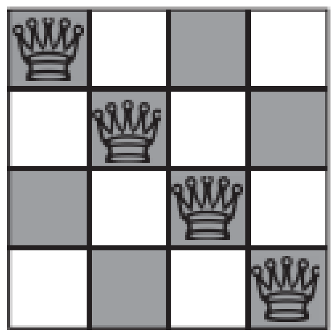


Figure 4: A 4-queens problem instance