Phần 1: Search

[10] **Evaluation Functions for Heuristic Search**

Say we define the evaluation function for use in a heuristic search problem as

$f(n) = (1 - w)g(n) + wh(n)$

where $g(n)$ is the cost of the best path found from the start state to state n, $h(n)$ is an admissible

heuristic function that estimates the cost of a path from n to a goal state, and $0.0 \le w \le 1.0$

(a) [9] What search algorithm do you get when

(i) $w = 0.0$ Uniform-Cost search

(ii) $w = 0.5$

A* search

(iii) $w = 1.0$

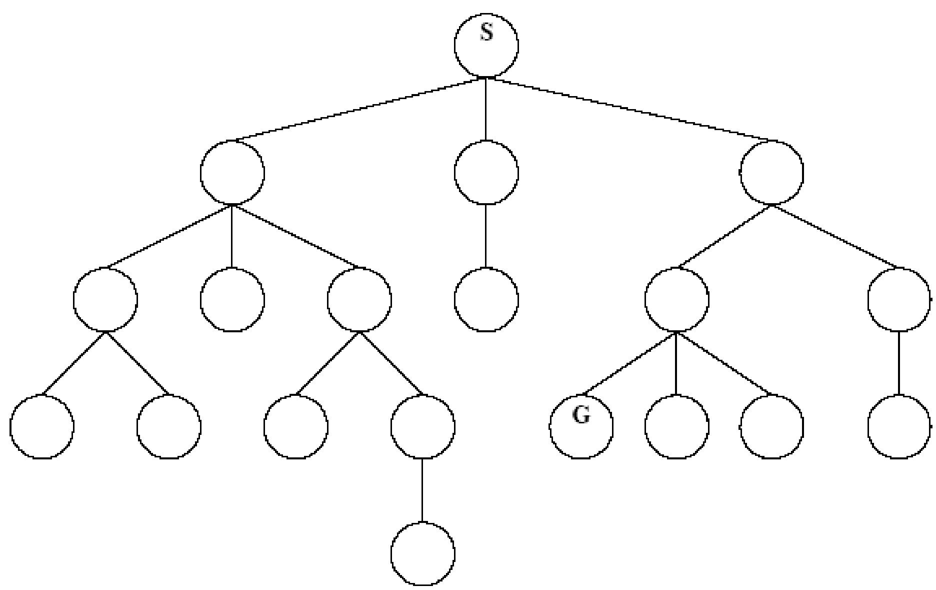
Greedy Best-First search

(b) [1] Based on your answer to (a), for what range of values of w would you expect your algorithm is admissible?

$0 \le w \le 0.5$

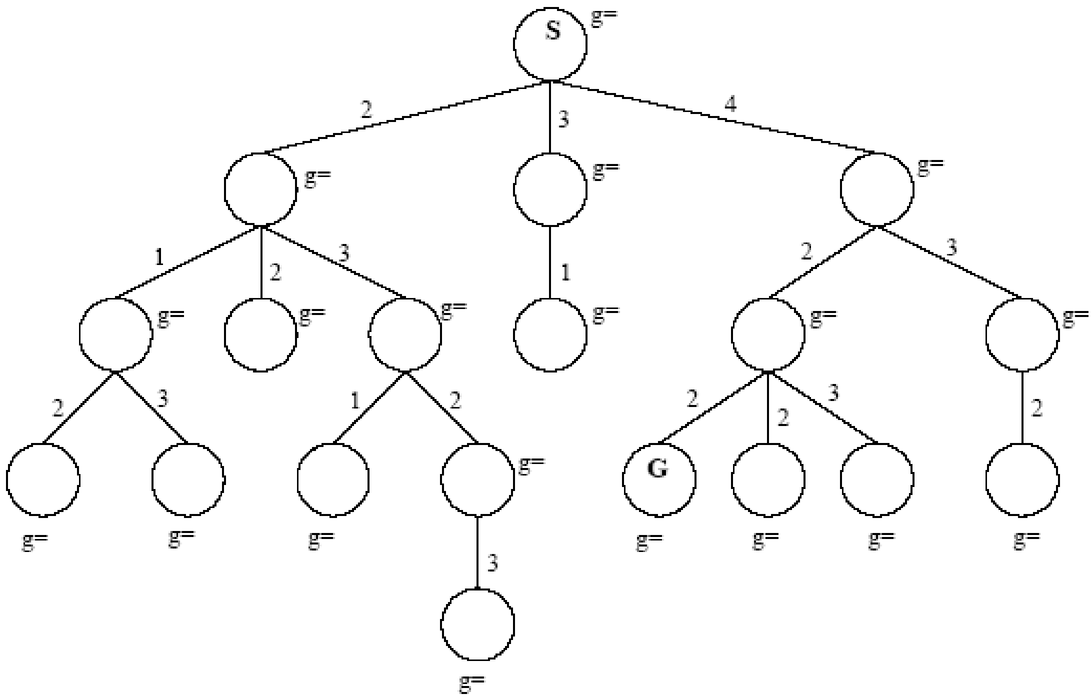
1. This problem considers three search methods applied to a state space that is a tree, where the problem is to find a path from start state S to goal state G. All arcs are directed, from the higher node to the lower node in the figures below.

(a) Depth-First Iterative Deepening. Expand children left to right. Write beside a node the number or numbers indicating when that node is expanded or the goal test performed, starting from 1.

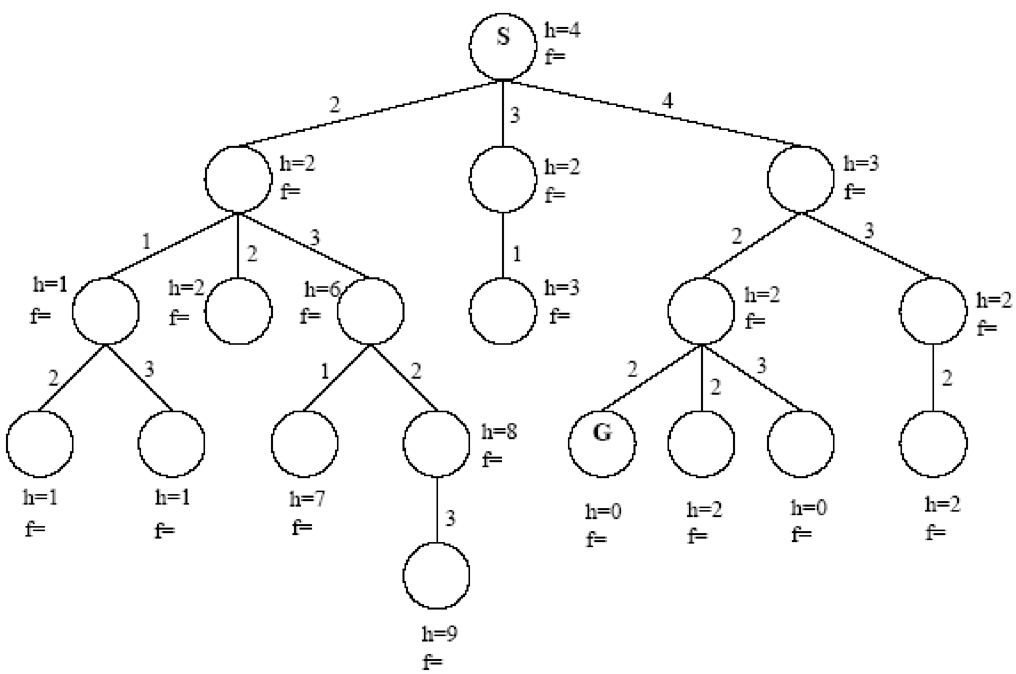


(b) Uniform-Cost Search. Each arc is labeled with the associated operator cost. For each node expanded or the goal test performed, write down the number indicating when it was expanded, starting from 1. Also, for each node inserted in the

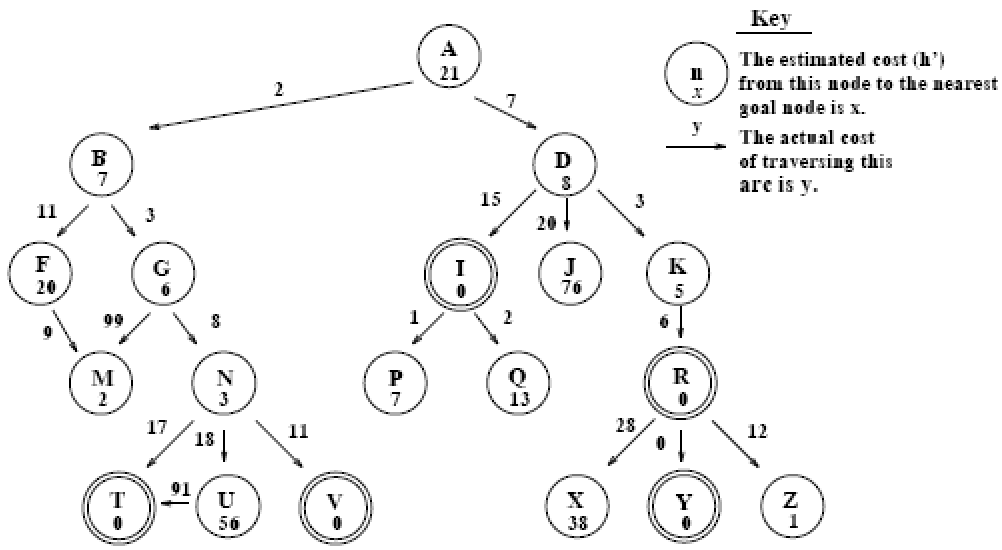
NODES list, write its g value next to the node. Do not write the g value for nodes that are never inserted into the NODES list.



(c) A* Search. Each arc indicates the operator cost, and each node's heuristic function value is indicated beside it. For each node expanded or the goal test performed, write down the number indicating when it was expanded. Also, for each node inserted in the NODES list, write its f value next to the node. Do not write the f value for nodes that are never inserted into the NODES list.



2. Consider the search graph drawn below. The initial state is at the top, and goal states are represented by double circles. *Note that arcs are directed*. Which goal state is reached will depend on the search strategy applied. For each of the search strategies listed below, indicate which goal state is reached (if any) and list, in order, the states *explored*. (A state is explored when the item containing it is *removed* from the OPEN list.) Assume that the NEXT-STATES function returns a state's successors in the same left-to-right order as in the search graph.



Depth-First Search
Goal state reached: _____ States explored: ____________________________________

Breadth-First Search
Goal state reached: _____ States explored: ____________________________________

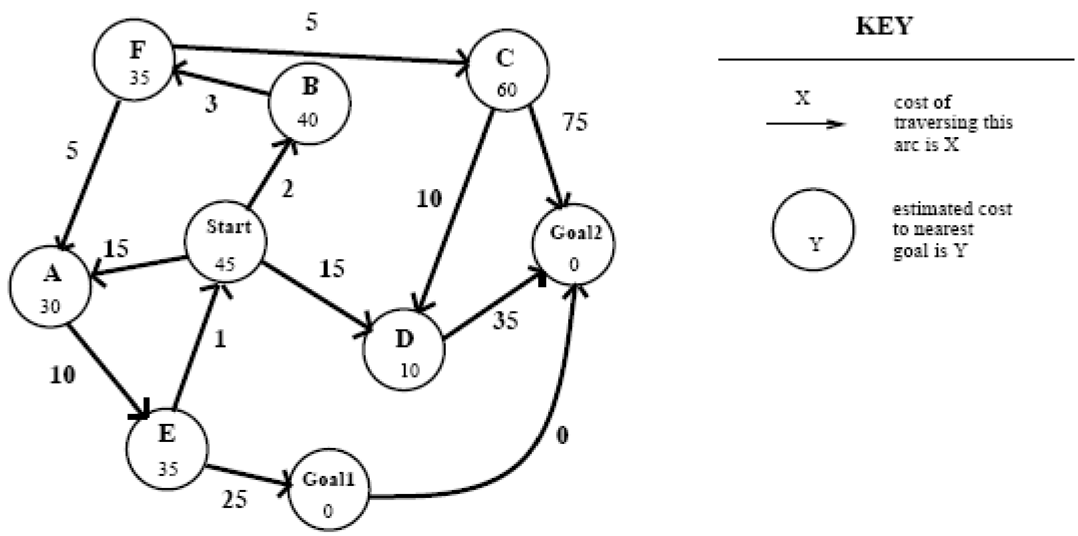
Best-First Search
Goal state reached: _____ States explored: ____________________________________

Beam Search (with a beam width of 2)
Goal state reached: _____ States explored: ____________________________________

Hill Climbing
Goal state reached: _____ States explored: ____________________________________

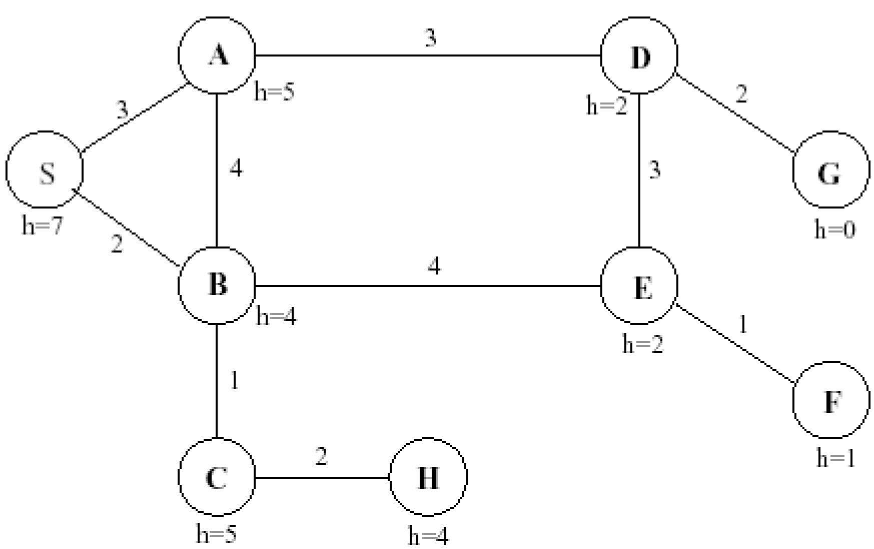
A* Search
Goal state reached: _____ States explored: ____________________________________

3. Consider the state-space graph drawn below; the start and goal states are labeled. Note that arcs are directed. For each of the search strategies listed below, indicate which goal state is reached (if any) and list, in order, the states expanded. (A state is expanded when it is removed from the OPEN list.) Assume that all search strategies use an OPEN and a CLOSED list, and that best first search checks for shorter paths to nodes. When all else is equal, nodes should be expanded in alphabetical order.



[16] **Search Methods**

Consider the following *state space* where the arcs represent all of the legal successors of a node, and all arcs are bi-directional, meaning the successor function can be applied from either node. The cost of each successor function is given by the number on the arc. The value of a heuristic evaluation function, *h*, if computed at a state, is shown along side each node. The start state is S and the goal is G.



When a node is expanded, assume its children are put in the NODES list in alphabetical order so that the child closest to the front of the alphabet is removed before its other siblings (for all uninformed searches and for ties between siblings in informed searches). Do *not* generate a child node if that same node is an *ancestor* of the current node *in the*

search tree. When selecting a node from NODES, in case of ties between non-siblings, use FIFO order to select the node that has been in NODES longest.
Give the sequence of nodes as they are **removed** from the NODES list (for expansion or before halting at the goal) for each of the following search methods.
(a) [4] Depth-First search (expand A before B)
(b) [4] A* heuristic search
(c) [4] Greedy Best-First search
(d) [4] Hill-Climbing

PROBLEM 2 - Heuristic Functions (10 points)
Imagine you are trying to solve the following puzzle using the *A** search algorithm. The puzzle involves two-digit numbers from 10 to 99. There are two legal actions: you can add 1 to any digit except 9 and you can subtract 1 from any digit except 0. (E.g., 10 can be changed, in one step, to 20 and 11, while 99 can be changed to 89 and 98). There is one restriction: you can never create the number 55. All actions have cost 100. The task is to find the shortest solution path from a given *start* number to a given *goal* number. Design and justify an *admissible* heuristic function for this puzzle.

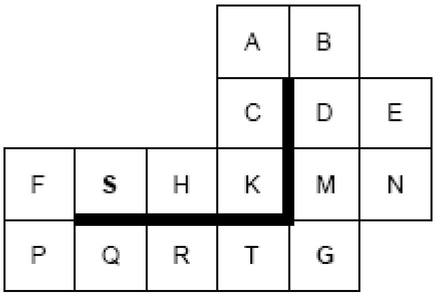
[14] Consider the following grid-structured search space shown in Fig. 1 in which the goal is marked G, the starting node is marked S, and obstacles are marked X. From each node, the possible successors are the four nodes corresponding to the north, south, east and west neighbors of the current node, marked M in Fig. 2.

```
    5         G           5                     5
    4       X             4     M               4     1
  y 3         X         y 3   M N M           y 3   4 N 2
    2   S                 2     M               2     3
    1                     1                     1
      1 2 3 4 5             1 2 3 4 5             1 2 3 4 5
          x                     x                     x
 Fig. 1 Example grid   Fig. 2 Successor nodes  Fig. 3 Ranking successors
```

Determine the sequence of locations (given as (x_coord, y_coord) values) which are expanded when using **Best-First search** and the heuristic function used is *h'* (*N*) = | *x_coord* (*N*) - *x_coord* (*G*) | . For example, in Fig. 2, the x-coordinate of the node marked N is 3, so *h'(N)* = | 3 - 5 | = 2. In sorting nodes on the OPEN list that have the same heuristic value, assume that ties are broken on the basis of when nodes are added to the list of nodes to explore next (earliest first) and among the successor nodes of a given node ties are broken on the basis of the ordering shown in Fig. 3 (smallest ranking first). Also assume that previously visited nodes are never again added to the OPEN list.

[30] **Search Methods**

Consider the following maze in which the successors of a cell include any adjacent cell in the
directions North, South, East, and West of the current cell, except at the boundary of the maze or when a barrier (thick line) exists. For example, successors(m) = {d,n,g}. Assume each type of move (N,S,E,W) has cost 1.



The problem is to find a path from cell **S** to cell **G**. Assuming children are always expanded in the
order East, South, West, North, then Breadth-First Search (without duplicates) would visit cells in the order: SHFKPCQARBTDG. If a search method needs to *break ties*, expand first the node
corresponding to the letter closest to the front of the alphabet.
For parts (a-d) below, what is the order of nodes expanded (plus the goal node if it is found) by each of the following search methods?
(a) [5] Depth-First Search. Assume cycles are detected and eliminated by never expanding a node
containing a state that is repeated on the path back to the root.
SHKCABDENMG
(b) [7] Greedy Search. Use as the heuristic function *h(state)* = Manhattan distance (aka Step distance, SPD, in HW2) from *state* to **G** assuming there were *no* barriers. For example, *h(K)*=2 and *h(S)*=4. Remove redundant states.
SHKCABDMG
(c) [5] Hill-Climbing Search. Use the same heuristic function as in (b).
SHK
(d) [7] Algorithm A or A* Search. Use the same heuristic function as in (b). Remove redundant states.
SHKCFPQRTG
(e) [2] Is *h* an admissible heuristic? (Yes/No)
Yes
(f) [2] Is *h* 2(*n*) = *min* (2, *h* (*n*)) an admissible heuristic? (Yes/No)
Yes
(g) [2] Is *h* 3(*n*) = *max* (2, *h* (*n*)) an admissible heuristic? (Yes/No)
No