Combinatoric Search Practice Exercises (Chpt. 3)

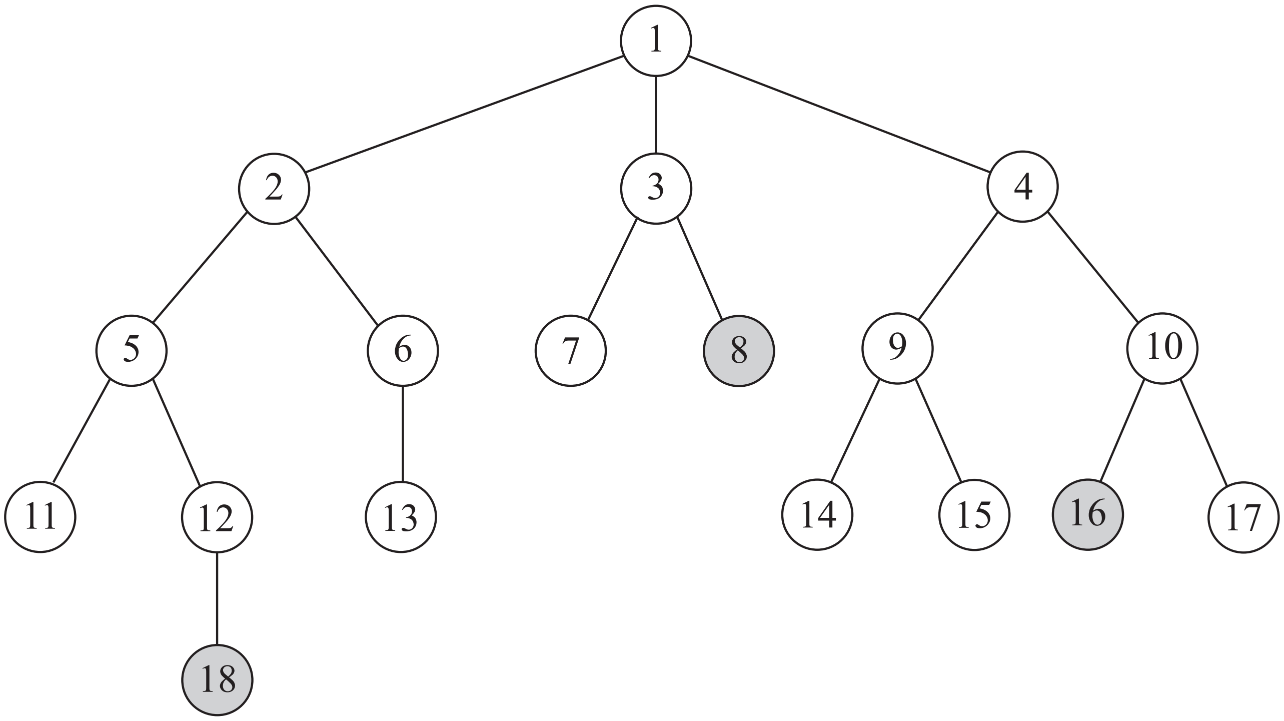
Prof. Jim Rehg
CS 3600 Introduction to Artificial Intelligence
School of Interactive Computing
Georgia Institute of Technology

February 26, 2018

These practice exercises are for your benefit in preparing for the exams. They will not be collected or graded. Solutions will be provided. Note that not all questions are representative of questions you will find on the exam, but the material covered by these questions will also be covered by the exams.

Question 1. Consider the following search tree, where the shaded nodes correspond to goal nodes. List the order in which nodes will be visited for the following four search strategies.[1] You should also show the evolving frontier/queue in order to get partial credit.

1. Breadth-First Search
2. Depth-First Search
3. Iterative Deepening Depth-First Search
4. Select a search algorithm with the property that node 16 will be selected as the solution (you may make any appropriate modification of the search tree as needed)



[1]The question is asking for the explored set, sometimes called closed list, of visited nodes at the time the search terminates.

SOLUTION

1. Breadth-First Search

Explored: 1 2 3 8

Frontier:

1
2 3 4
3 4 5 6

Note that the search terminates at 3 because BFS performs goal check on each generated successor. 8 will pass goal-test and terminate the search. There is a question of whether the solution node should be included on the explored list. Strictly speaking it shouldn't be there, as you would return goal state without updating the explored list data structure. But conceptually it is explored and so we include it, as it helps to visually explain why search terminated.

2. Depth-First Search

Explored: 1 2 5 11 12 18

Frontier:

1
2 3 4
5 6 3 4
11 12 6 3 4
12 6 3 4
18 6 3 4

Depth-First Search Alternative

Obtained by generating and pushing successor nodes left to right instead of right to left.

Explored: 1 4 10 17 16

Frontier:

1
4 3 2
10 9 3 2
17 16 9 3 2
16 9 3 2

3. Iterative Deepening DFS

Iteration 1
Explored: 1

Frontier:

1

Iteration 2
Explored: 1 2 3 4

Frontier:

1
2 3 4
3 4
4

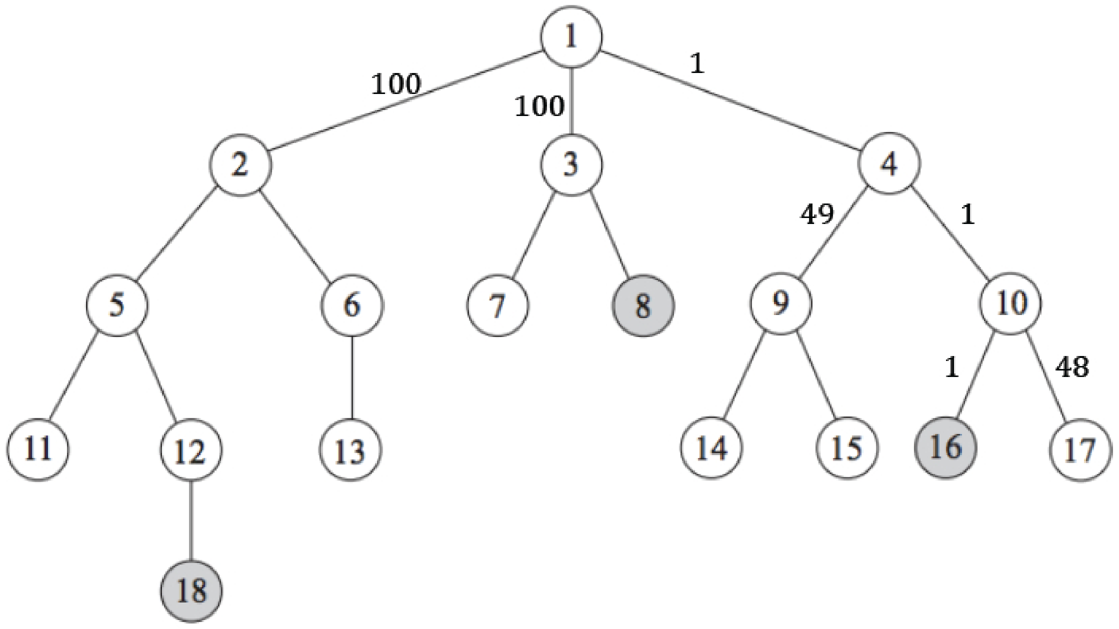
Iteration 3
Explored: 1 2 5 6 3 7 8

Frontier:

1
2 3 4
5 6 3 4
6 3 4
3 4
7 8 4
8 4

4. Selected Search Algorithm

There are many ways to achieve the stated goal. One way is to use *Uniform Cost Search* with appropriately assigned costs, one example is below.



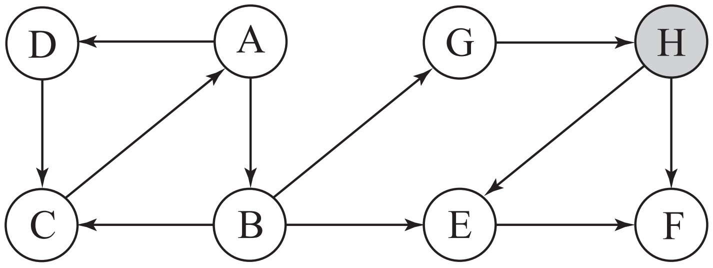
Explored: 1 4 10 16

<u>Frontier:</u>

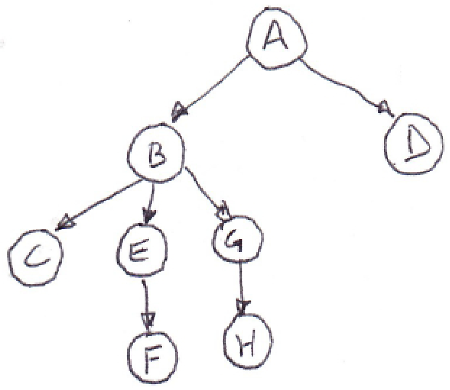
1(0)
4(1) 2(100) 3(100)
10(2) 9(50) 2(100) 3(100)
16(3) 17(50) 9(50) 2(100) 3(100)

Question 2. Consider the following graph. Assume we start the search at state "A" with goal state "H." At any point during search, if ties need to be broken, the node which is closest to the start of the alphabet (closest to A) will be selected. List the order in which nodes will be visited for the following two search methods:

1. Breadth-First Search
2. Depth-First Search



SOLUTION



1. Breadth-First Search

Explored: A B D C E G H

Frontier:

A
B D
D C E G
C E G
E G
G F

2. Depth-First Search

Explored: A B C E F G H

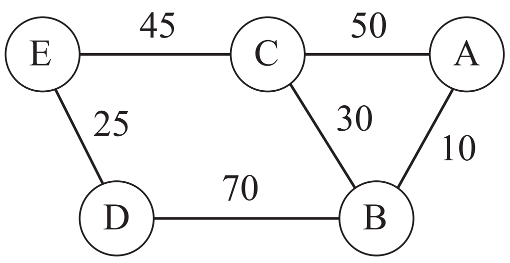
Frontier:

A
B D
C E G D
E G D
F G D
G D
H D

Question 3. Use **A* Search** to solve the map problem below, where the numbers give the distances between states.

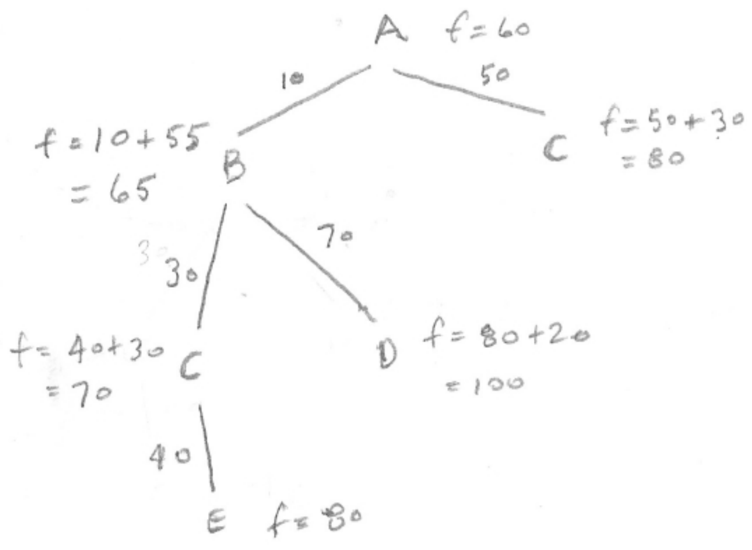
- **Initial State** = A
- **Goal State** = E
- The value of the heuristic function for each node is given in the table on the left.

n	$h(n)$
A	60
B	55
C	30
D	20
E	0



(a) Draw the search tree for A*, showing the evolving frontier.

SOLUTION



Frontier:

A(60)
B(65) C(80)
C(70) D(100)
E(80) D(100)

(b) Indicate below the order in which nodes will be expanded (i.e. put in the explored set)
Explored: A B C E

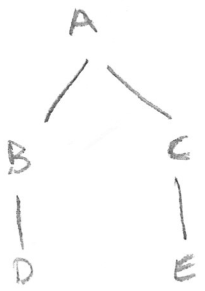
(c) For each of the possible node expansion orders shown below, indicate which *uninformed search algorithms* could have produced the given node order when applied to the graph above. Note: Be explicit about how you are breaking ties for nodes at the same depth.

$^{1}\boxed{A}\ ^{2}\boxed{C}\ ^{3}\boxed{E}$	BFS A+ DFS A+
$^{1}\boxed{A}\ ^{2}\boxed{B}\ ^{3}\boxed{C}\ ^{4}\boxed{D}\ ^{5}\boxed{E}$	Uniform Cost Search
$^{1}\boxed{A}\ ^{2}\boxed{B}\ ^{3}\boxed{C}\ ^{4}\boxed{E}$	BFS A-
$^{1}\boxed{A}\ ^{2}\boxed{C}\ ^{3}\boxed{B}\ ^{4}\boxed{E}$	Iterative Deep DFS A+

For the solution above, note that "A+" means break ties with the highest letter first, while "A-" means break ties with lowest letter first. Also remember that BFS checks for the goal as it generates each successor and before inserting into frontier, while DFS, Iterative Deepening DFS, and UCS all check the goal node upon popping from the frontier.
See the next page for worked examples.

BFS A↓

VISITED: A B C E

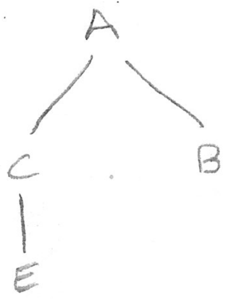


FRONTIER

~~A~~
~~B~~ C
~~C~~ D

BFS A↑

VISITED: A C E

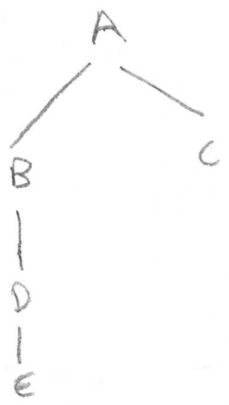


FRONTIER

~~A~~
~~C~~ B

DFS A↓

VISITED: A B D E

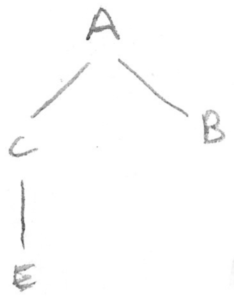


FRONTIER

~~A~~
~~B~~ C
~~D~~ C
~~E~~ C

DFS A↑

VISITED: A C E



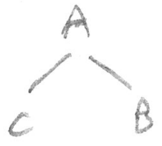
FRONTIER

~~A~~
~~C~~ B
~~E~~ B

ITER-DEEP DFS A↑

V: A C B E

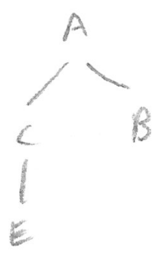
ITER #1:



FRONTIER

~~A~~
~~C~~ B
~~B~~

ITER #2:

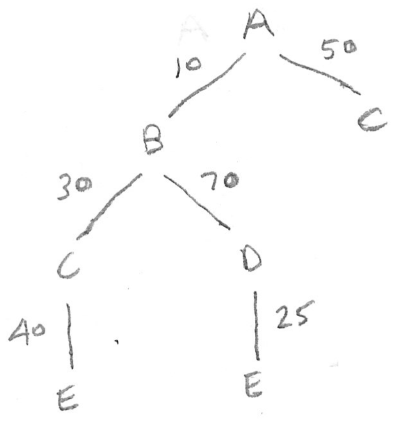


FRONTIER

~~A~~
~~C~~ B
~~E~~ B

UNIFORM COST SEARCH

V: A B C D E



FRONTIER

A(~~0~~)
B(~~10~~) C(50)
C(~~40~~) D(80)
D(~~80~~) E(80)
E(~~80~~) E(105)

FINAL PATH:
A B C E

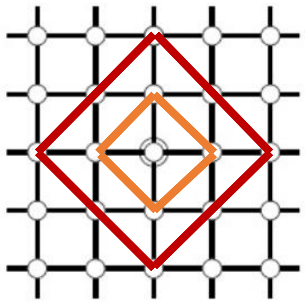
(d) Show that the heuristic function for this problem satisfies the *consistency* criteria.

Consistency requires that $h(n) \leq c(n, a, n') + h(n'), \forall n, n'$. Show this using a table:

n'	n	$c + h(n')$	$h(n)$
E	C	40	30
E	D	25	20
D	B	90	55
C	A	80	60
C	B	60	55
B	A	65	60

We can verify by inspection that the inequality holds for each row. Therefore the heuristic function is consistent.

4. (3.26 in textbook) Consider the unbounded version of the regular 2D graph shown in figure 3.9. The start state is at the origin, (0, 0), and the goal state is (x, y).



a. *What is the branching factor b in this state space?*
   - Branching factor: maximum number of successors of any node
   - The branching factor for this graph is 4

b. *How many distinct states are there at depth k (for k > 0)?*
   - This question is asking, "how many distinct states exist at depth k, meaning k steps away from the origin in some combination of x and y directions?"
   - See the figure above right, the orange lines denote the "frontier" of states reachable in 1 step, and the red lines donate the frontier reachable in 2 steps. You can see that all states on the frontier or inside the frontier will be reachable, since you are allowed to loop back to interior states.
   - The number of reachable states is $1 + 2k(k + 1)$

   You can derive this answer as follows: First, you can see that the "length" of the square defined by the frontier is $2k$. This is the number of reachable states which are the farthest from the origin in terms of steps. Second, since all of the frontiers at sizes below k are also reachable, the total reachable states are given by:

$$1 + \sum_{j=1}^{k} 4j = 1 + 4\sum_{j=1}^{k} j = 1 + \frac{4k(k + 1)}{2} = 1 + 2k(k + 1),$$

   where the 2nd equality follows from a standard formula for the sums of the positive integers. Note that the added 1 comes from including the state at the origin.
   Also note that your textbook says that there are approximately $2d^2$states within a distance d of a point on an infinite grid. Here we have derived the exact answer.

c. What is the maximum number of nodes expanded by breadth-first tree search?
   - $O(b^d)$, or the branching factor raised to the depth
   - This equals $O(4^{x+y})$ in this example

d. What is the maximum number of nodes expanded by breadth-first graph search?
   - To solve this, we use the same formula from part (b), but for a point (x, y), instead of a depth k. Since all the points on a diagonal are at the same depth, this depth equals x + y.
   - Substituting (x + y) for k, we get $1 + 2(x + y)(x + y + 1)$.

e. Is $h = |u - x| + |v - y|$ an admissible heuristic for a state at (u, v)? Explain.
   - Yes. This heuristic is the length of the path from the state to the goal.

f. How many nodes are expanded by A* graph search using h?

  - x + y, since the heuristic in (e) is perfect.

g. Does h remain admissible if some links are removed?
  - Yes. Removing links makes the path longer, so h becomes an underestimate.

h. Does h remain admissible if some links are added between adjacent states?
  - No. Adding some links could decrease the optimal path length by allowing shortcuts. In this case our heuristic is pessimistic.

Question 5. The graph separation property of Graph-Search (see Fig. 3.9 in your text) states that the frontier divides the explored and unexplored regions of state space.

(a) Prove the graph separation property by induction.

Let U, F, E be the *unexplored*, *frontier*, and *explored* sets of nodes, respectively. Separation is often written as $U \operatorname{sep} E|F$, which means that sets U and E are separated by F. This is true if and only if, for every $x \in U$ and $y \in E$ connected by a path P, there exists at least one node $n \in P$ such that $n \in F$. In words, every path between nodes in U and E passes through F.

Note: To prove by induction, show that the statement holds (trivially) at the start node. Assume that it is true after t steps of expanding a node, and then prove that it must also hold after $t + 1$ steps.

(b) Suppose your friend invents a more efficient variant of Uniform Cost Search in an undirected graph, where instead of generating all of the successor states for a given node n, only a subset of the possible successors, randomly selected, are generated. Is the separation property preserved under this modification? (why or why not) How is the optimality of Uniform Cost search affected?

(c) When performing graph search in a directed graph, expanding a parent node generates all of the child successors, where edges are directed from parent to child (e.g. in Problem 2 above, node A has two children, B and D, and is itself a child of C). Does graph separation hold in this case? (why or why not) Show that Uniform Cost Search is optimal in a directed graph.