TA: Jade Cheng
ICS 311
Quiz Solution #7
April 29, 2009

Quiz #7:

Show the time (algorithm) and space (data structures) complexities of the following searches. Briefly justify your answers. Assume an average branching factor of *b*. [4 pts]

a. Depth-first search

Answer:

Time: $\Theta(b^d)$ where d equals to the max depth explored.

Space: $\Theta(b \times d)$ expands one child at each level with b nodes, leaving the remaining children on the stack.

b. Breadth-first search

Answer:

Time: $\Theta(b^d)$ where d equals depth of shortest path to a solution.

Space: $\Theta(b^d)$ BFS expands all children at each level, thus there are b^d nodes on the queue at level d.

c. Best-first (heuristic) search

Answer:

Time: $\Theta(b^d)$ and $\Omega(d)$ where d equals depth of shortest path to a solution.

Space: $\Theta(b^d)$ Best-First Search uses a priority queue, Ideally, one or a small number of “best” nodes are expanded at each level, using space similar to Depth-First Search. The better the heuristic, the fewer “side” nodes are expanded. If all heuristic values return 1, Breadth-First Search is performed.

Are the following searches guaranteed to find a shortest path to a solution? State the conditions, if any, for your answers. [3 pts]

a. Depth-first search

Answer:

No – It's like following the left all in a maze. Only with iterative deepening (a few levels at a time) or a very restricted search space (low branching factor and/or multiple goals) will DFS find close to a shortest path.

b. Breadth-first search

Answer:

Yes – All nodes in one level are expanded before any in the next one are expanded. Thus, any goal found takes the minimum number of moves.

c. Best-first search

Answer:

Yes – If the heuristic never over-estimates the actual path to the goal from the current node.

Compare time complexities. [3 pts]

Algorithm A is described by: $T(n) = n^2 + 2n + 8$ **.**

Algorithm B is described by: $T(n) = 4n1gn + 41gn + 16$ **.**

a-1. Which algorithm would be faster when $n=2$.

Answer:

$A(2)=4+4+8=16$, $B(2)=8+4+16=28$, therefor A is faster.

a-2. Which algorithm would be faster when $n=8$.

Answer:

$A(8)=64+16+8=88$, $B(8)=96+12+16=124$, therefor A is faster.

b. Calculate the value of *n* where they both take about the same amount of time. (hint: it's larger than 8).

Answer:

For $n=16$, they are about equal. $A(16)=256+32+8=296$, while $B(16)=256+16+16=288$ (B is a little faster).

c. Which algorithm is more efficient. (Extra credit)

Answer:

B is $\Theta(n1gn)$ algorithm, a lower time complexity than A, which is $\Theta(n^2)$. When *n* gets large, B will be faster than algorithm A. In this case, when $n \geq 16$, B is faster.