6.824 Artificial Intelligence, Fall 1994
Problem Set 7 Solutions

Problem 1. There are two interpretations of statement "nodes reacheable by path of length n":

1. nodes that are not more than n operations away from the root of the search tree
2. nodes that are exactly n operations away from the root of the tree

Case 1. The running time of BFS can be estimated by the number of nodes it visits. As the number of nodes reacheable by paths of length at most n is $\leq cn$ then the running time is $O(n)$.

The running time of ID depth first search can be estimated as

$$\sum_{i=1}^{n} ci = c\sum_{i=1}^{n} i = c\frac{n(n-1)}{2} = O(n^2)$$

.

Case 2. The number of nodes the BFS algorithm visits is:

$$\sum_{i=1}^{n} ci = c\sum_{i=1}^{n} i = c\frac{n(n-1)}{2} = O(n^2)$$

In the IDA depth first search the number of nodes on level k of the search tree is $\leq ck$ and every node in this level is visited $n-k+1$ times, where n is the length of the solution path. Therefore running time can be estimated as:

$$\sum_{i=1}^{n}\sum_{k=1}^{i} ck = c\sum_{i=1}^{n}\sum_{k=1}^{i} i = c\sum_{i=1}^{n}\frac{i(i-1)}{2} = O(n^3)$$

Problem 2. Consider a graph search problem with 2 operators each with cost 2 (the search has branching factor 2). There are 2^d different nodes that can be reached by paths of length d (assuming that any node can be reached by exactly one path.) We allow for more than one goal node and for any node n we define $h^*(n)$ to be the cost of the shortest path from n to a goal node.

At each node n one of the operators must move one step closer to the goal, i.e., there is an operator o such that $h^*(o(n))$ is $h^*(n) - 2$. Such an operator will be called a *good move* from the node n. In this problem we assume that there is only one good move from n. The operator other than the good move will be called the bad move. We assume that the bad move does not change the distance to the goal, i.e., if o' is a bad move from n then $h^*(o'(n))$ equals $h^*(n)$.

We assume a heuristic function h such that for any node n we have that $h(n) = \frac{1}{2}h^*(n)$.

We define the *added cost* of a path $m \leadsto n$ to be the projected cost of the path $m \leadsto n$ minus the projected cost of the empty path $m \leadsto m$.

Part a. Let G be the number of good moves in a path and let B be the number of bad moves in a path.

We can compute added cost in the following way:

$$\begin{aligned}
ProjectedCost(m \leadsto n) &= 2(B+G) + h(n) && (0.1)\\
ProjectedCost(m \leadsto m) &= h(m) && (0.2)\\
&= \frac{1}{2}h^*(m) && (0.3)\\
&= G + \frac{1}{2}h^*(n) && (0.4)\\
&= G + h(n) && (0.5)\\
& && (0.6)
\end{aligned}$$

so

$$\begin{aligned}
AddedCost(m \leadsto n) &= ProjectedCost(m \leadsto n) - ProjectedCost(m \leadsto m) && (0.7)\\
&= 2(B+G) + h(n) - (G + h(n)) && (0.8)\\
&= 2B + G && (0.9)\\
& && (0.10)
\end{aligned}$$

Part b. Let $M(k)$ be the number of nodes that can be reached by paths with *added cost* of *exactly* k. As any node can be reached by exactly one path, then the number of nodes that can be reached by paths with added cost k correspond to the number of such paths. Now path of added cost k can be:

1. path with good move followed by a path with added cost $k-1$; or
2. path with bad move followed by the path with added cost $k-2$.

Then the recurrence relation for $M(k)$ is:

$$M(k+2) = M(k+1) + M(k)$$

which is the well known Fibonacci sequence.

Part c. Order of growth of $M(k)$ can be computed from the solution of the Fibbonaci reccurence:

$$M(k) = \frac{\Phi^k - \hat{\Phi}^k}{\sqrt{5}}$$

where $\Phi = \frac{1+\sqrt{5}}{2}, \hat{\Phi} = \frac{1-\sqrt{5}}{2}$

So $M(k) = O(\Phi^k)$

Part d. Assume that the length of the solution path is d. Then the running time complexity for breath first search is $O(2^d)$. For A^* the running time complexity is given as $O(N^*(c) \log N^*(c))$ where $N^*(c)$ is the number of nodes reacheable a path with projected cost of at most c. As solution has length d, its cost is $2d$ and projected cost for every path is its $addedcost + d$. But then $N^*(2d)$ can be computed as:

$N^*(2d) = \sum_{i=0}^{d} M(i) = O(\Phi^d \log \Phi^d)$

To compare the two methods we can set their running times to be equal.

$$c_1 2^{d_1} \approx c_2 \Phi^{d_2} log \Phi^{d_2}$$

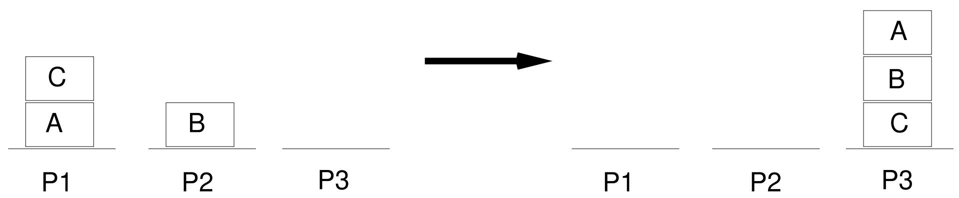
$$log c_1 + d_1 \log 2 \approx \log c_2 + d_2 \log \Phi + \log d_2 \log \Phi$$

and by ignoring lower order terms we get:

$$\frac{d_2}{d_1} \approx \frac{1}{\Phi} \approx 1.44$$

So A^* can find paths about 44% longer than breadth first search.

Problem 3. Three step plan that solves the Sussman anomaly is:



1. $move(C, A, P3)$
2. $move(B, P2, C)$
3. $move(A, P1, B)$

GPS procedure as descibed in the problem statement cannot solve the problem in 3 steps. The intuitive reason for this is that GPS works on one subgoal at a time, and starts to work on others only after the first is completely solved.

As solution presented above is the only three step solution, then all and only operators from it, must be used by GPS. Therefore whenever the solution plan is produced by GPS contains different operator, the plan must be suboptimal and more than 3 steps long. We will show that GPS cannot find the 3 step solution by showing that it will never reproduce optimal sequence of steps.

In step 2 the GPS procedure selects an operator that has on its add list a proposition from Ω and not from Σ. The only operators from the optimal plan that satisfy this are (1) $move(A, P1, B)$ and (2) $move(B, P2, C)$.

(1) If the move $move(A, P1, B)$ is chosen by GPS, then proposition $clear(A)$ must be satisfied. GPS can satisfy $clear(A)$ by using operator of type $move(*, A, *)$ which clears A. As only $move(C, A, P3)$ is in the optimal plan then GPS can produce only the sequence $move(C, A, P3)$ followed by $move(A, P1, B)$, which is suboptimal and thus cannot lead to 3 steps plan.

(2) If $move(B, P2, C)$ is chosen, its prerequistes are immediately satisfied (prerequisites are in Σ). Therefore the plan will start with step $move(B, P2, C)$, which again leads to suboptimal planning sequence.

From (1) and (2) it follows that plan of length 3 cannot be generated by GPS.

A plan that can be found by GPS:

1. $move(C, A, P3)$ - subgoal: $clear(A)$
2. $move(A, P1, B)$ - goal: $on(A, B)$
3. $move(A, B, P1)$ - subgoal: $clear(B)$
4. $move(B, P2, C)$ - goal: $on(B, C)$
5. $move(A, P1, B)$ - goal: $on(A, B)$