Avik Bag
Prof. Balduccini
CS 380
Assignment 2

Written Assignment

Problem 1

The best way I would say is by going up from Eloise. This is because a parent will have one mother, which is a definitely way of tracing. Also, more likely than not, the mother will be associated to one husband. But if we were to trace from ancestor to descendant, there may be several descendants making the tree larger and longer to search for.

Problem 2

- This can definitely slow down the algorithm. If the condition is set such that the lower the action cost value, the more optimized the result is, it may cause the search algorithm to search the entire state space. It may end up being counter productive, making the algorithm slower.

- Yes, having a limit as to how low the step constant will help in the case of a graph. It will limit not go any lower than what it needs to. Also, I feel that increasingly negative values could be caused by loops seeing that certain moves will lower it's cost value without any end. In the case of a tree, I don't think this might come in to handy, as there are no cycles and propagation would move from one unique node to another unique node.

- This shows that the cost function is not effective in determining the optimal path to the solution in a tree. The cost function should be able to identify a certain parameter that is unique in getting to the solution. There may be several paths to the solution and it should be able to pick out the path with the lowest cost, and not stop searching when a certain cost value is achieved. The optimal behavior for this agent is ineffective. It can find any path that is small, even though it might not be able to reach the solution. Also, any visited state should be recorded and checked against so that no loops are present in the graph search algorithm.

- The reason why normal humans do not indefinitely roam about scenic routes is because humans do not have unlimited time and unlimited fuel. These are the limiting factors for defining the state space and actions for artificial agents to stop looping. This would be considered the negative constant, making sure the AI agent does not go into an endless loop.

- Floor cleaning AI(Roomba) determining the time and moves for which to clean the house depending on the state. The limiting factor should be power consumption.

Problem 3

```
backTrack(stateList , depthBound) {

    state = first element of stateList
    if state is a member of the rest of stateList, return FAIL
    if goal(state), return PASS
    if length(stateList) > depthBound, return FAIL
    moves = getPossibleMoves(state)

    for each move m in ruleSet {
            newState = applyMoveCloning(state,m)
            newStateList = addToFront(newState,stateList)
            path = backTrack(newStateList, depthBound)
            if path != FAILED return append(path,m)
            if path != PASS return PRINT(path)
    }
}
```

So the main issue here was the return NULL when the goal state was reached. Instead, if the goal state was reached, a true flag should be passed which would either return the path or print it. This way, the loop keeps on going and all the paths are visited.

Problem 4

For this part of the question, firstly we check whether the sum of all the marbles in each basket is divisible by 3. If we end up with a whole number, then it passes, else not possible.

For determining each move, it is assumed that the order of moves will be as follows.

(1 -> 2)
(1 -> 3)
(2 -> 1)
(2 -> 3)
(3 -> 1)
(3 -> 2)

and there are going to be rules that govern the validity of the moves. Firstly, it will check whether the number of marbles moves is at most double the number of marbles present in the receiving basket. Next, it will check the number present. If I the value of the basket on the receiving or giving end is the average value, then no move will be executed.

For the first part, the backtrack will not work as there are no solutions within depth 3.

For the second part, there are several solutions within depth 4 itself. The first solution is the one that shows up, and this is as follows

(6,5,1)
(4,7,1)
(4,6,2)
(4,4,4)

for the third part, there are a total of 4 possible paths. As shown in the table below

(6,5,1)	(6,5,1)	(6,5,1)	(6,5,1)
(4,7,1)	(5,5,2)	(4,7,1)	(7,4,1)
(4,6,2)	(4,5,3)	(5,4,3)	(6,4,2)
(4,4,4)	(4,4,4)	(4,4,4)	(4,4,4)

Depth first search will yield the same result as the second part, which is

(6,5,1)
(4,7,1)
(4,6,2)
(4,4,4)

Breadth first search will yield the same result as the second part, which is

(6,5,1)
(4,7,1)
(4,6,2)
(4,4,4)