CSE422 Theory Assignment

Md. Shamiul Islam

ID: 17301108

Section: 08

Answer to the Question no: 1

ⓐ Initial state:

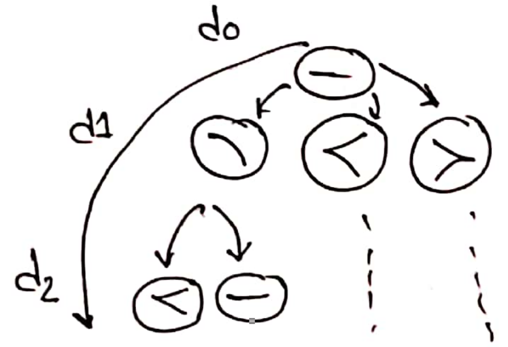
we can choose any one piece from the set. Lets begin with straight piece.

Goal state:

According to the given task, is to connect those piece into a railway that has no overlapping tracks. so all the pieces should be used in a single connected track.

step cost: we can take one cost per each piece. So it well be equal as the number of pieces.

ⓑ



According to my choice, it will be better option if we go with depth-wise, if we use depth-wise search algorithm or simply i can say depth-first search algorithm (DFS) the we will get a better solution. ~~[illegible]~~

There are many repeated states. so it might be good to use depth-first search.

ⓒ here, if we remove any one of the "fork" piece it makes the problem unsolvable. from the above given railway set, we can assume that there must be equal numbers of pegs and holes, a solution has no overlapping or loose attachment. Removing any of the fork violets this property. Every fork is creating two tracks and only a fork can rejoin those tracks into one, therefore if a fork is missing it won't work.

ⓓ here, in total piece $(16+12+2+2) = 32$ piece. So the depth will be 32 as well. If we predict each of the piece is unique, for each peg. as we are starting from straight line, now we can say that, $(12 + 2\times16 + 2\times2 + 2\times2\times2)$

$= 12+32+4+8$

$= 56$ predictions

also we can get from permutation of each pieces,

$$\frac{168^{32}}{12!\,16!\,2!\,2!} = 4\times10^{48}$$

As we move to our depth, it will be seen that that branching factor is getting narrow.

Answer to the Question no: 2

ⓐ we know that to form a closed loops through out the citis the traveling salesperson problem (TSP) is using to find the shortest path. Hence, MST is relaxed version of that because it gives shortest path is need to be a closed loop. if this is a generally a fully connected graph and it is always shorter than or equal to a closed loop. Also if we relaxed the constraints for 'TSP' such a way that each city can be visited more than one time or twice, then the repeated cost are not counted, this give us a solution for mst problem.

(b) MST dominates straight line distance.

we are literally saying this MST always gives a better and higher value. MST always includes current node initial node and goal node must either be straight line between them or sometimes it may seen some extra line to cover the path. so the statement is true in this manner.

(c) If the actions are just the name of the city to go to we can only go to a city we have not visited. unless we've visited them all, in this case we can only go back to start or initial phase.

(d) TSP is a touring problem in which each city must be visited exactly once. The aim is to find the shortest tour algorithm.

TSP Tour (G,C)

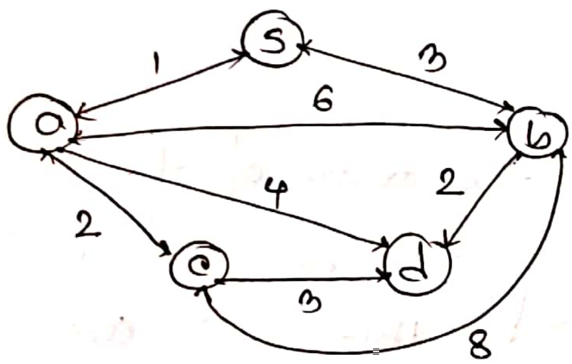
Select a vertex r, U[G] to be root vertex compute minimum spanning tree. T for G from root r!

using MST (G, c, r)

assume, L be the list of vertices visited in preorder tree of T.

return the result R that visits vertices in the order of L

Solve with A* search algorithm,



here, c is goal node

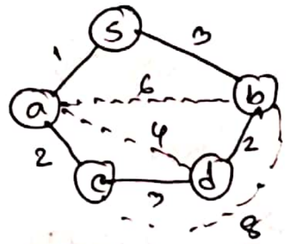
we know that,

total cost = past cost + heuristic value

$$f(n) = g(n) + h(n)$$

(i) $s \to a$, $f(a) = 1 + (3+2+3) = 9$

(ii) $s \to b$, $f(b) = 3 + (1+2+3) = 9$



then,

(iii) $s \to a \to c$, $f(c) = (1+2) + (3+2) = 8$

(iv) $s \to b$, $f(b) = 3 + (1+2+3) = 9$

∴ $s \to a \to b$ and $s \to a \to d$ can't construct connected tree.

now,

(v) $s \to a \to c \to d$, $f(d) = (1+2+3)+3 = 9$

another,

(vi) $s \to b \to d \to c$, $f(c) = (3+2+3) + (1+2) = 11$

$\therefore$ $s \to b \to a$ & $s \to b \to c$ can't construct tree connection.

Return to previous stab,

(vii) $s \to a \to \cancel{b} \to \cancel{d} \to b$, $f(b) = (1+2+3+2) + (0) = 8$

(viii) $s \to b \to d \to c$, $f(c) = (3+2+3) + (2+1) = 11$

~~pa~~ A* path $\to$ $s \to a \to c \to d \to b \to c$

$f(b) = (1+2+3+2+3) + 0$

$= 11$

path cost $= \cancel{9}$ 11

$\therefore$ path, $s \to a \to c \to d \to b < c$

cost, 11

Answer to the Question no: 3

(a) Local beam search with k=1,

It starts with k=1 is hill climbing search.

It scrutinizes all neighbouring states of these k- states then selects the k number of best neighboring states. This also follows the greedy local search. In this search is started in one random start state and looks at all the neighboring states and continues ~~in~~ the search in the direction of best neighbor, this search is also called Hill- climbing search.

(b) Local beam search with one initial state and no limit on the number of states retained, this statement follows Breadth first search (BFS).

(c) Simulated annealing with T=0 at all times, this type search is similar to Hill- climbing search.

Sun Mon Tue Wed Thu Fri Sat

Date:

(d) Simulated annealing with $T\alpha$ at all times,
this similar to random walk search, cause every successor would be accepted with probability 1.
In this case this also can be DFS.

(e) Genetic algorithm with population size N = 1..
this given statement is identical to random walk search. i.e., if here 2 selected parents will be ~~sa~~ same then crossover yield have an exact copy of the individual., ~~simi~~ small chance of mutation.

Answer to the question no: 4

There are some discrete assembly space determined by the joint angles at every place where two pieces are connected. So, now we can define a state as a set of oriented, connected pieces and associated angles is the range of [-10, 10] in addition set of non-connected pieces too. We can allow for layouts of tracks in which they lie on top of one another or we can discard it.

The important portion is the set of allowed moves, we can unlink any piece or link an unlink piece to an open peg at any allowed angle.

In general, there will be ~~an~~ no unique shortest solution for given angle change in term of consequent changes to other angles. and some changes may be discarded.

Answer to the Question no: 5

Sun Mon Tue Wed Thu Fri Sat
Date :

ⓐ From question 2,
we know that, stated problem can be solved through hill climb search algorithm, but here we are considering the optional value. So here if we use hill climb searching ~~but~~ it will give us local maxima, input and some variables carried as neighbor nodes.
In this case we will consider optional value ~~search~~.

problem to generate the minimum cost.
local variable: current, node
neighbor, node

now,
loop,
neighbor ← [highest valued successor of current neighbor]
if (value [neighbor] ≤ value (current)) {
return state [current]
}
current ← neighbor.

So, if we compare our obtained result with A* search algorithm then we get,

A* search always provide cheapest solution,

$\therefore$ A* search is $\rightarrow$ $f(n) = g(n) + h(n)$

total cost = path cost + heuristic value.

From this algorithm, it returns cheapest cost ~~with~~. ~~state~~ It traverse next step by calculating the heuristic cost value. hill climb,

On the other hand, this return the state when it achieved a local maxima point. and the next state is mostly the node with highest value successor.

A* vs Hill climb:

A* gives cheapest solution but hill climb moves in the direction of increasing cost. A* return a state with less cost, but hill climb reaches a locally maximize state.

A* vs Genetic:

→ least cost, but genetic generated single state by combining two parent state.

→ cheapest solution, genetic repeat individual fits enough or enough time elapsed.

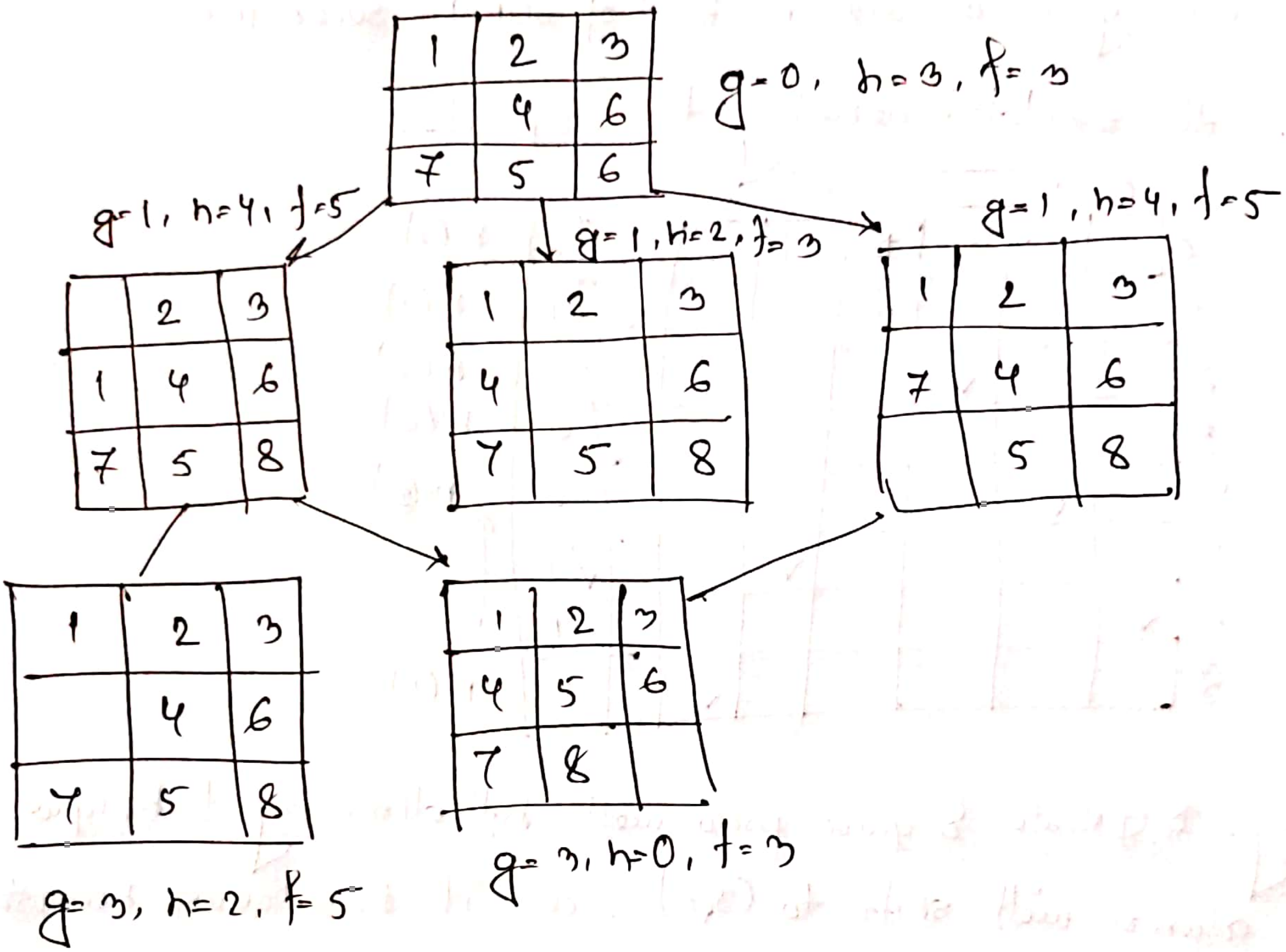
(b) connecting all cities into an undirected path, pick two points along the path at random. Split the path at those points. ~~process~~ create three pieces. Trying all six possible way to connect the three pieces.

Keeping the best value, we can disconnect the other path.

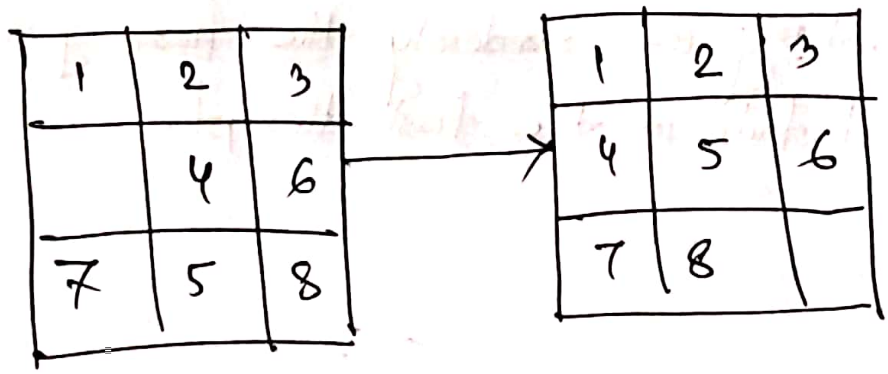
Iterate it for better improvement.

Answer to the Question no: 6

here minimizing problem



now, 8 puzzle Random restart hill climbing



If we look at previous 8 chart,
we are able to solve this given problem in only 3 steps.
So, I think ~~there is~~, restart randomly is not needed. This will also give us similar kind of ~~situation~~ scenario.

now, with simulated annealing for 8 puzzle,

	0	1	2	3	4	5	6	7	
0				~~✓~~				2	h(2)
1							✓	1	h(1)
2			✓					2	h(2)
3								0	h(0)
4		✓						2	h(2)
5					✓			✓ 2	
6	✓							1	
7						✓		1	h(1)

using, ~~8 puzzle~~ 8 queen prob with hill climbing technique 8th column will shift to (3,7), as it has lowest heuristic value & it is a minimizing problem.

Random restart hill climbing use randomly the ~~first~~ generated initial position when it fails in the first attempt.