AI homework 1 Theory 1003391

Q1)

Initial State:
side1 = {M,M,M,C,C,C}
side2 = {}

Goal-State:
side1 = {}
side2 = {M,M,M,C,C,C}

State-space:

1. Side1{M, M, M, C, C, C}, Side2{} (START STATE)
2. Side1{M, C}, Side2{M, M, C, C}
3. Side1{C, C}, Side2{M, M, M, C}
4. Side1{M, M, M, C}, Side2{C, C}
5. Side1{M, M, C, C}, Side2{M, C}
6. Side1{}, Side2{M, M, M, C, C, C} (GOAL STATE)

Actions: (from respective State)

1. Send (C, C) => goes to state 4, Send (M, C) -> goes to state 5
2. Send (M, C) => goes to state 6
3. Send (C, C) => goes to state 6
4. Send (M, M) => goes to state 2
5. Send (M, C) => goes to state 2, Send (M, M) => goes to state 3

Path-cost: Each action/move will have 1 unit cost

Q2

a)
In a graph search, we keep track of the nodes visited in an explored set, and do not visit or expand them again

In a tree search, we do not keep track of the nodes visited, and there is no explored set, hence the same node can be visited multiple times

b)
A state is a representation of a configuration, a node is a data structure that constitutes part of a search tree, and a node include state, parent, node, action, path cost, depth

c)
The explored set keeps track of the same nodes, the reason being there can be more than one path to the same node

Q3a)

BFS as a graph search

1. frontier = [a] explored_set = [] solution = []
2. frontier = [ab, ac] explored_set = [a] solution = []
3. frontier = [ac, abd] explored_set = [b, a] solution = []
4. frontier = [abd] explored_set = [c, b, a] solution = []
5. frontier = [abdx] explored_set = [d, c, b, a] solution = []
6. frontier = [] explored_set = [x, d, c, b, a] solution = [abdx]

Q3b)

DFS as a graph search

1.frontier = [a] explored_set = [] solution = []
2.frontier = [ac, ab] explored_set = [a] solution = []
3.frontier = [acd, ab] explored_set = [c, a] solution = []
4.frontier = [acdx, ab] explored_set = [d, c, a] solution = []
5.frontier = [ab] explored_set = [x, d, c, a] solution = [acdx]

Q3c) a, b and d

Q3d) a, c, and d

q4a)

1.frontier = [a] explored_set = [] solution = []

2.frontier = [ab, ac] explored_set = [a] solution = []

4.frontier = [ac, abd] explored_set = [b, a] solution = []

5.frontier = [abd, ace, acf] explored_set = [c, b, a] solution = []

6.frontier = [ace, acf, abdh, abdx] explored_set = [d, c, b, a] solution = []

7.frontier = [acf, abdh, abdx, acei] explored_set = [e, d, c, b, a] solution = []

8.frontier = [abdh, abdx, acei] explored_set = [f, e, d, c, b, a] solution = []

9.frontier = [abdx, acei] explored_set = [h, f, e, d, c, b, a] solution = []

10.frontier = [acei] explored_set = [x, h, f, e, d, c, b, a] solution = [abdx]

q4b)

1.frontier = [a] explored_set = []solution = []

2.frontier = [ac, ab] explored_set = [a] solution = []

3.frontier = [acf, ace, ab] explored_set = [c, a] solution = []

4.frontier = [acfi, ace, ab] explored_set = [f, c, a] solution = []

5.frontier = [acfih, ace, ab] explored_set = [i, f, c, a] solution = []

6.frontier = [acfihx, ace, ab] explored_set = [h, i, f, c, a] solution = []

7.frontier = [ace, ab] explored_set = [x, h, i, f, c, a] solution = [acfihx]

q5a) Path cost in brackets

1. frontier = [a(0) ] explored_set = [] solution = []

2. frontier = [ac(3) , ab(5)] explored_set = [a] solution = []

3. frontier = [ab(5), acf(8), ace(153)] explored_set = [c, a] solution = []

4. frontier = [acf(8), abd(55), ace(153)] explored_set = [b, c, a] solution = []

5. frontier = [acfi(11), acfe(13), abd(55), ace(153)] explored_set = [f, b, c, a] solution = []

6. frontier = [acfe(13), acfih(14), abd(55), ace(153)] explored_set = [i, f, b, c, a] solution = []

7. frontier = [acfih(14), acfed(15), abd(55), ace(153)] explored_set = [e, i, f, b, c, a] solution = []

8. frontier = [acfed(15), acfihx(24), abd(55), ace(153)] explored_set = [h, e, i, f, b, c, a] solution = []

9. frontier = [acfedx(16), acfihx(24), abd(55), ace(153)] explored_set = [d, h, e, i, f, b, c, a] solution = []

10. 9. frontier = [acfihx(24), abd(55), ace(153)] explored_set = [x, d, h, e, i, f, b, c, a] solution = [acfedx(16)]