Assignment1

Name:Kaichen Zhang ID:40000160

Part I: Prolog

Prolog solutions are contained in the zip file.

Part II: Knowledge representation

Q-2.A

Answer:

A: $\forall X$ (isProfessor(X) → isSmart(X))

B: $\forall X$ (isProfessor(X) ∧ depCS(X) → isCrazy(X))

C: isStudent(X) ∧ hasGoodGPA(X) → onDeanList(X)

D: $\exists X \forall Y \forall Z$ (isStudent(X) ∧ likesCourse(Y) → ¬haveAssign(Z))

E: $\exists X \forall Y$ (¬(isProfessor(X) ∧ isSmart(X)) →¬likes(Y,X))

Q-2.B

Answer:

A.

1. X {A→X}
2. Y {X∧B∧E→Y}
3. Z {Y∧D→Z}

B.

Goal to prove: Z

L∧F→Z

    C→L

    C

Y∧D→Z

    X∧B∧E→Y

    A→X

    A

    B

    E

Sub-goal: Φ

C.

Forward chaining method is more efficient in this case.

Q-3.B

Answer:

A.

1. member(Tony)
2. member(Simon)
3. member(Ellen)
4. ∀X member(X) → biker(X)∨skier(X)∨(biker(X)∧skier(X))
5. ∀X biker(X) →¬likes(X,rain)
6. ∀X skier(X) →likes(X,snow)
7. ∀X likes(Tony,X) →¬likes(Ellen,X)
8. ∀X ¬likes(Tony,X) → likes(Ellen,X)
9. likes(Tony,rain)∧likes(Tony,snow)

B.

1. member(Tony)
2. member(Simon)
3. member(Ellen)

4a. ¬member(X) ∨biker(X)∨skier(X)∨biker(X)
4b. ¬member(X) ∨biker(X)∨skier(X)∨skier(X)
4. ¬member(X) ∨biker(X)∨skier(X)
5. ¬biker(X) ∨¬likes(X,rain)
6. ¬skier(X) ∨likes(X,snow)
7. ¬likes(Tony,X) ∨¬likes(Ellen,X)
8. likes(Tony,X) ∨ likes(Ellen,X)

9a. likes(Tony,rain)
9b. likes(Tony,snow)

C.

Goal: biker(Ellen) ∧¬skier(Ellen)

Add

10. ¬biker(Ellen) ∨ skier(Ellen) to the knowledge base.

Unit Resolution:

* member(Ellen) {Ellen/X} ¬member(X)∨biker(X)∨skier(X)

Resolution:

* skier(Ellen)∨biker(Ellen) { } ¬biker(Ellen) ∨ skier(Ellen)
* skier(Ellen)

Unit Resolution:

* likes(Tony,snow) {Tony/X} ¬likes(Tony,X) ∨¬likes(Ellen,X)

Unit Resolution:

* ¬likes(Ellen, snow) {Ellen/X} ¬skier(X) ∨likes(X,snow)

* ¬skier(Ellen) {} skier(Ellen)

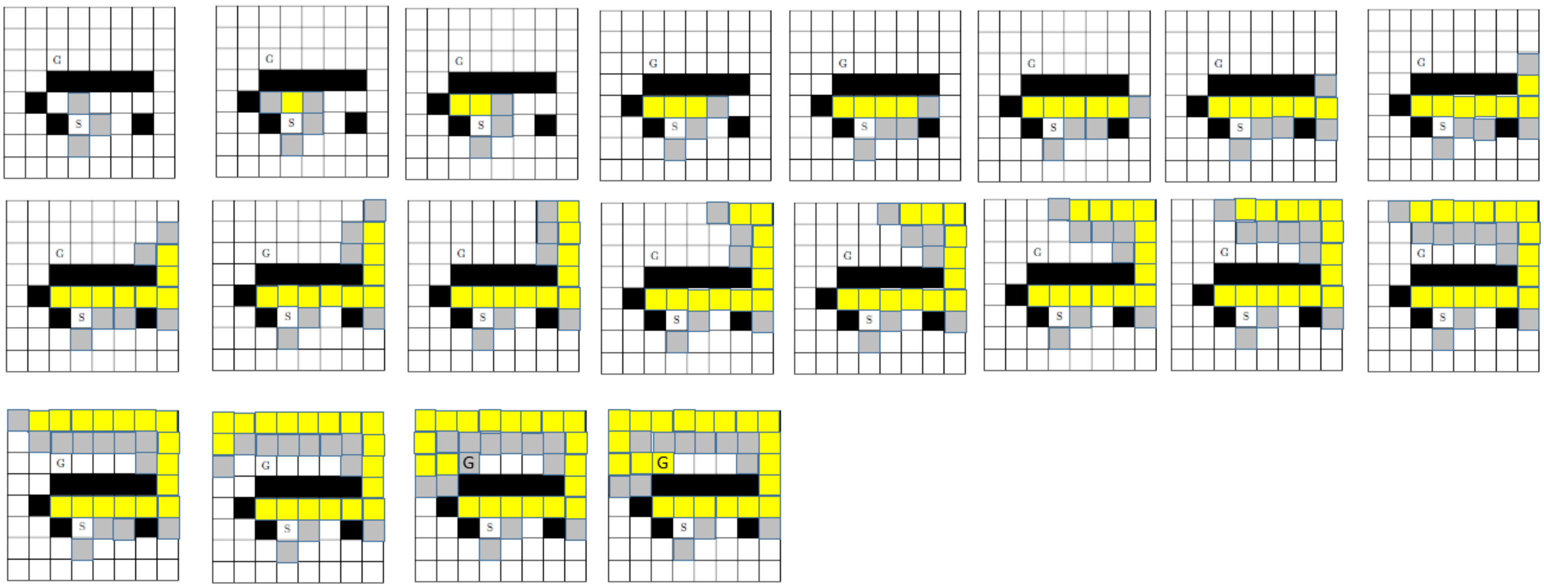
Φ

Part III: State Space Search

Q-4.A

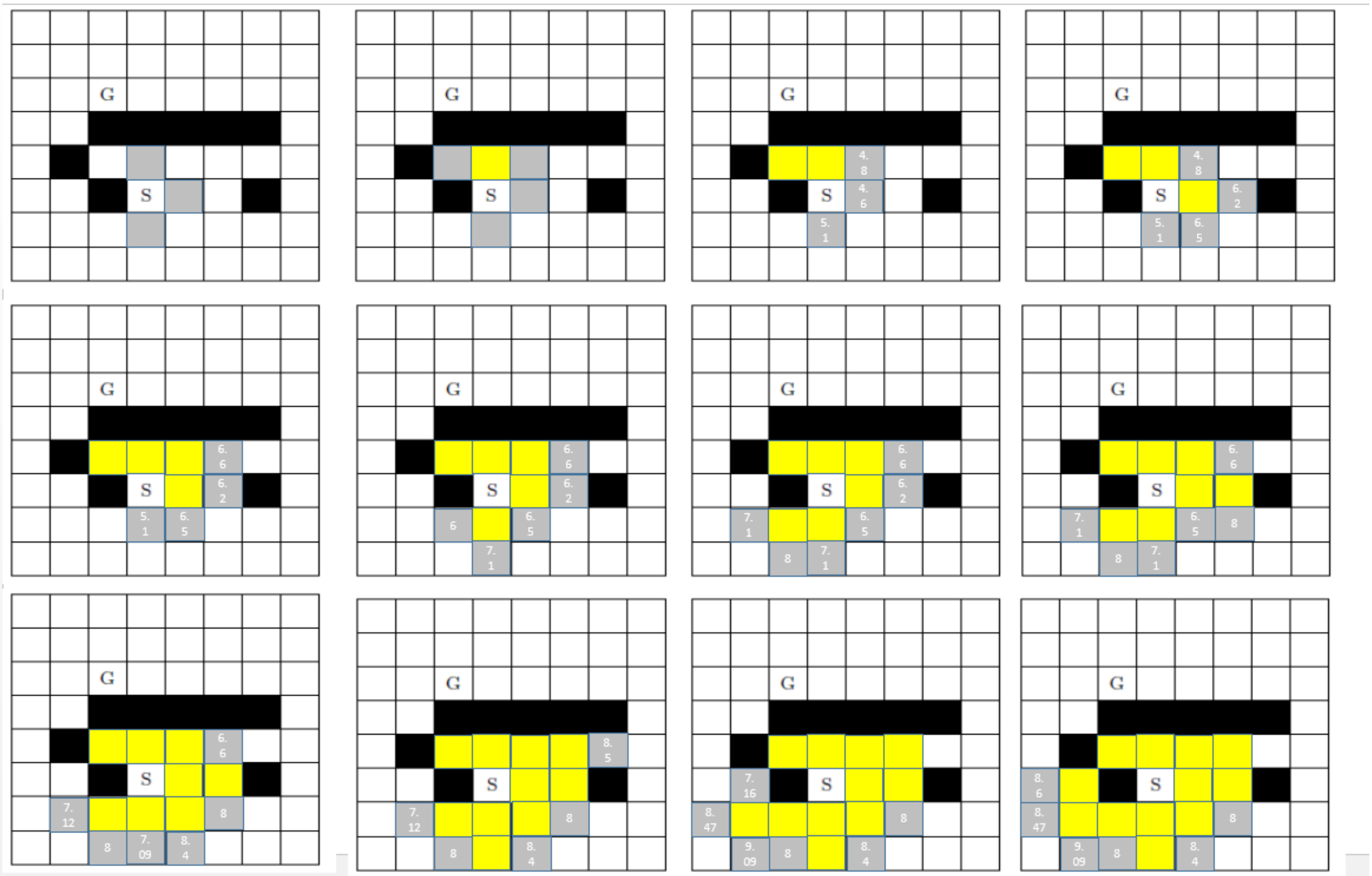
Answer:

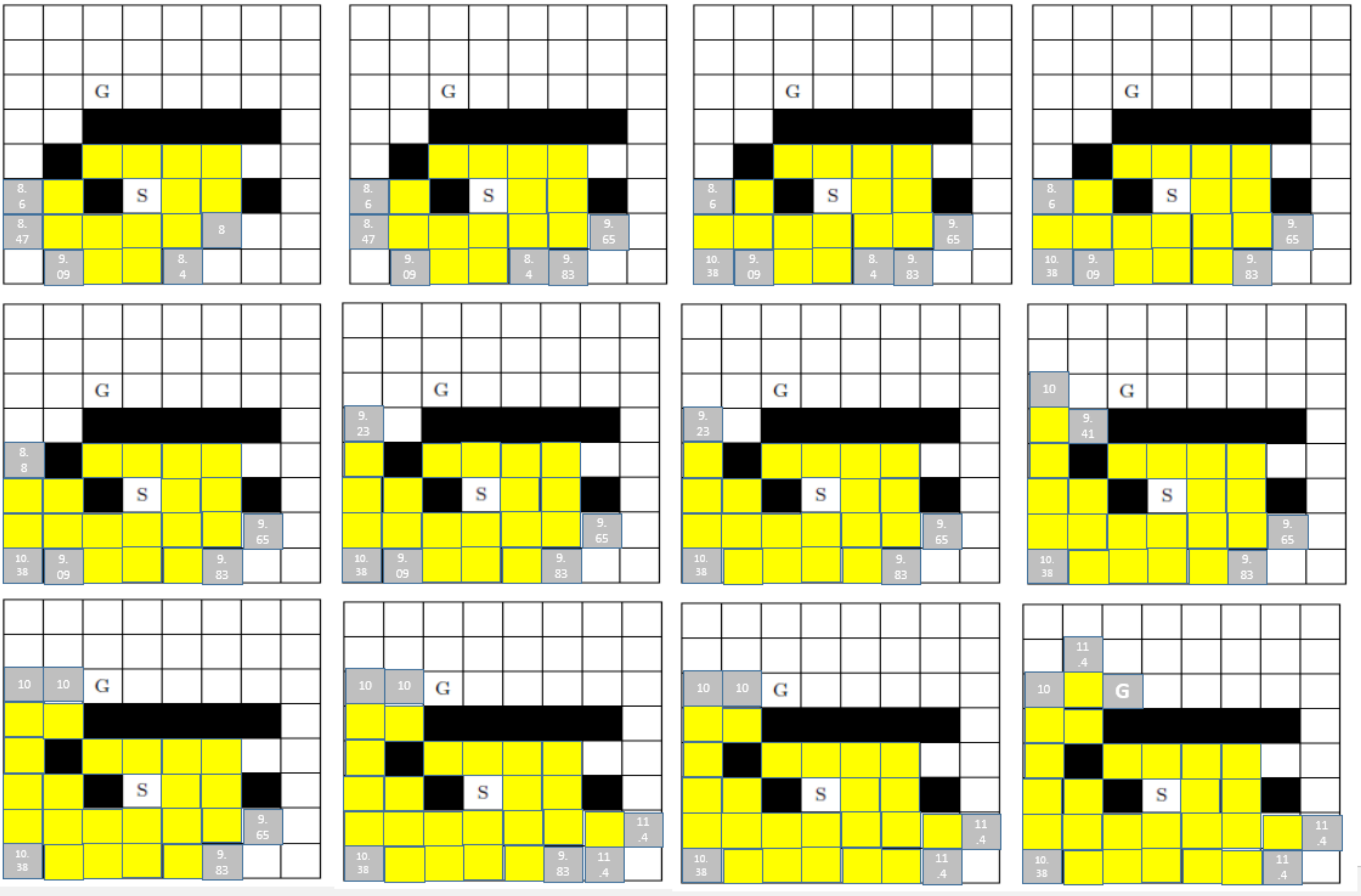
a. The grey blocks are in the open list, the yellow blocks are visited.

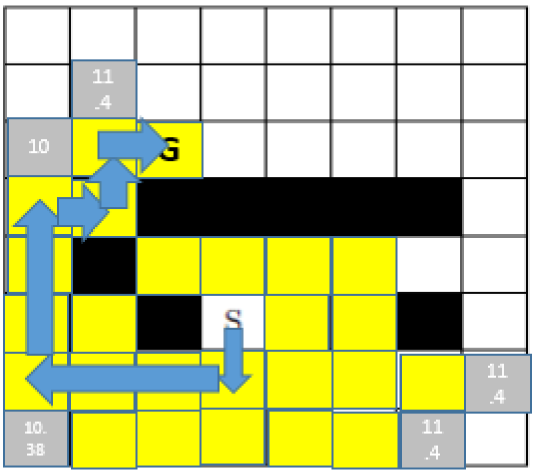


b. A* search. Assume the heuristic is the value of direct line distance from current node to the goal.

f(x) = cost(x) +h(x) ,each step cost 1. The value of f is calculated in the block.

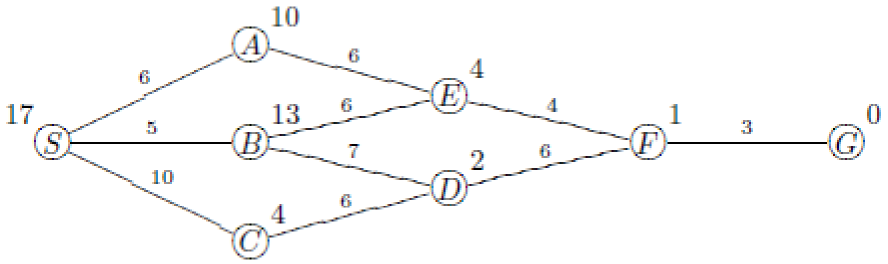






11 .4
10
G
S
11 .4
10. 38
11 .4

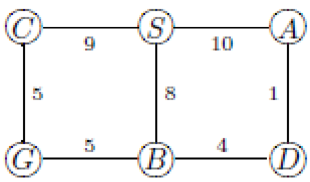
Q-4.B
Answer:



current	Open list	Closed list
S	B22,A23 ,C27	
B	A23 ,C27,E41,D42	S
A	C27,E39,D42	S,B
C	D37,E39	S,B,A
D	E39,F45	S,B,A,C
E	F45	S,B,A,C,D
F	G49	S,B,A,C,D,E
G		S,B,A,C,D,E,F
		S,B,A,C,D,E,F,G

Solution path is S→C→D→F→G, cost is 49.

Q-4.C
Answer:



	S	A	B	C	D	G
heuristic	0	0	4	3	0	0

Threshold=8

Open list	Closed list
S0	
C9,B8,A10	S
G17, C9,A10	S,B

Threshold=9

Open list	Closed list
S0	
C9,B8,A10	S
G17,B8,A10	S,C
D16,G17, A10	S,C,B

Threshold=10

Open list	Closed list
S0	
C9,B8,A10	S
G17,B8,A10	S,C
D16,G17, A10	S,C,B
D16,G17	S,C,B,A

Threshold=16

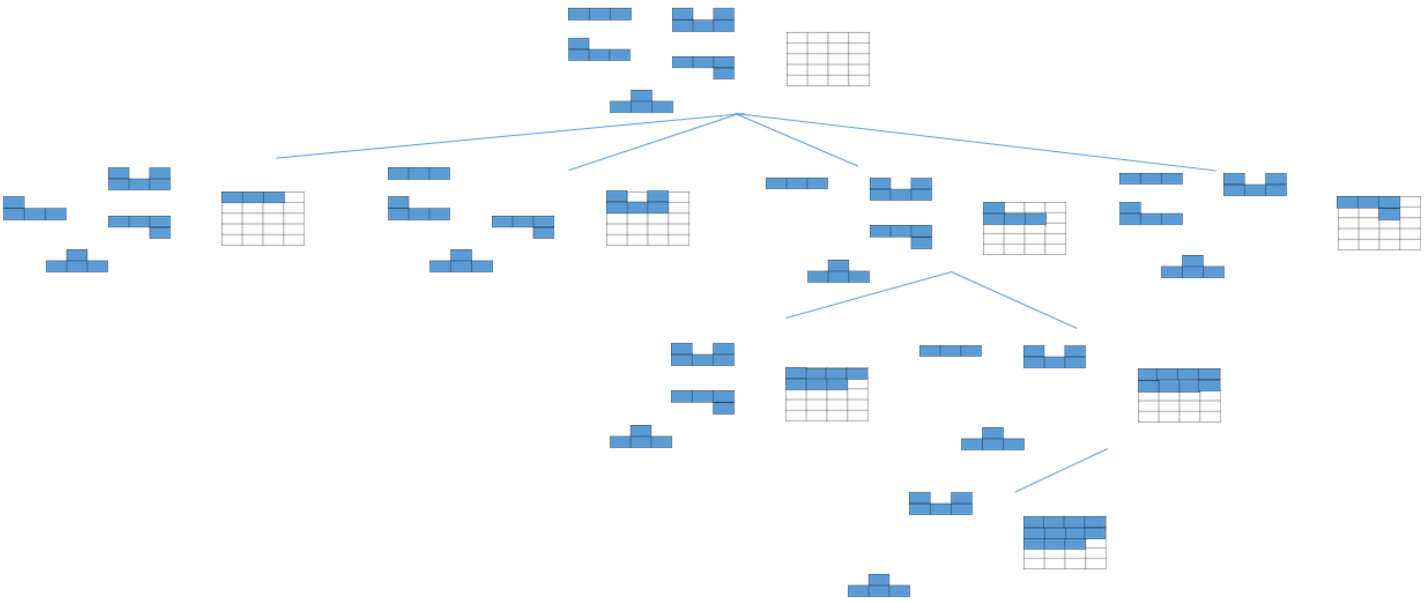
Open list	Closed list
S0	
C9,B8,A10	S
G17,B8,A10	S,C
D16,G17, A10	S,C,B
D16,G17	S,C,B,A
G17	S,C,B,A,D

Threshold=17

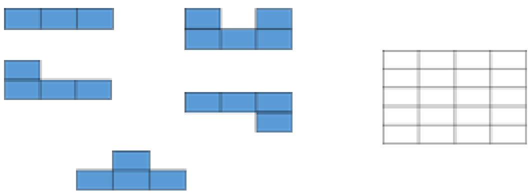
Open list	Closed list
S0	
C9,B8,A10	S
G17,B8,A10	S,C
D16,G17, A10	S,C,B
D16,G17	S,C,B,A
G17	S,C,B,A,D
	S,C,B,A,D,G

Find the goal.

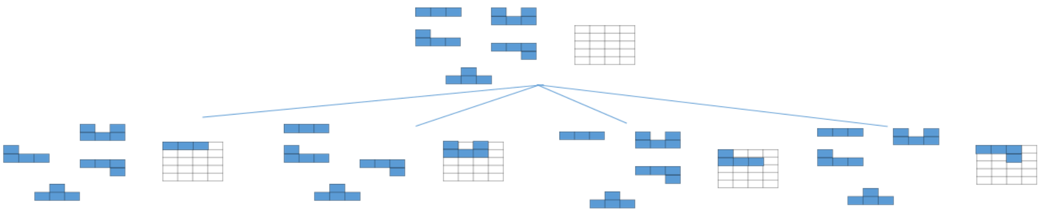
Q-4.D
Answer:
a) We consider the child of current node is the right side adjacent node, the child of the last cell is the first cell in the row below. And traverse the path cell by cell, row by row. Start from the left top corner.



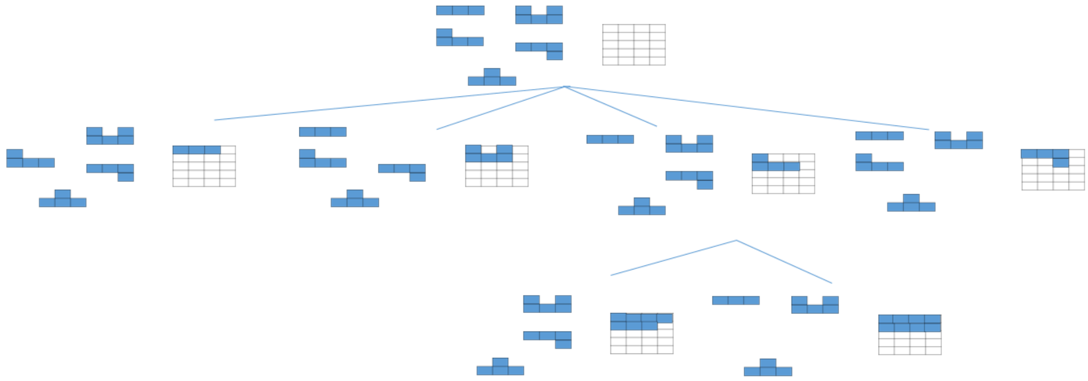
b)
Limit = 0



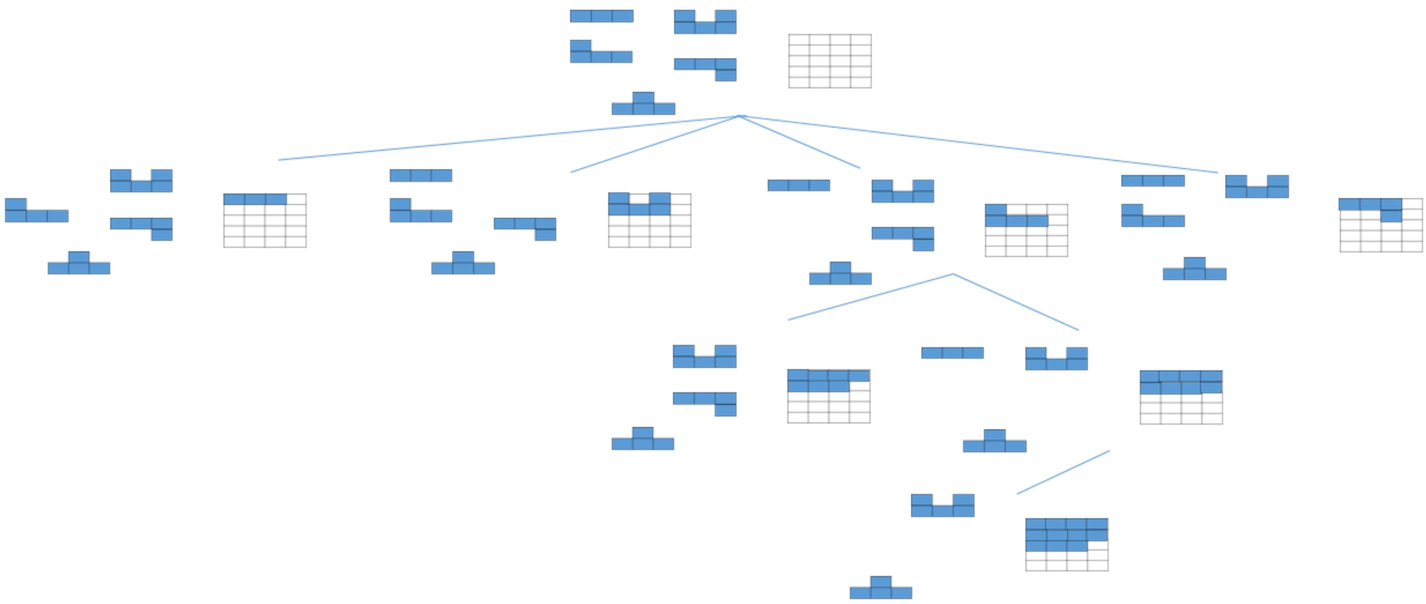
Limit =1



Limit = 2



Limit =3



	Completeness	Optimality	Time	Space complexity
BFS	Yes	Yes	$O(b^{d+1})$	$O(b^{d+1})$
IDS	Yes	Yes	$O(b^{d})$	O(bd)
backtracking	Yes	No	O(n!)	O(n!)

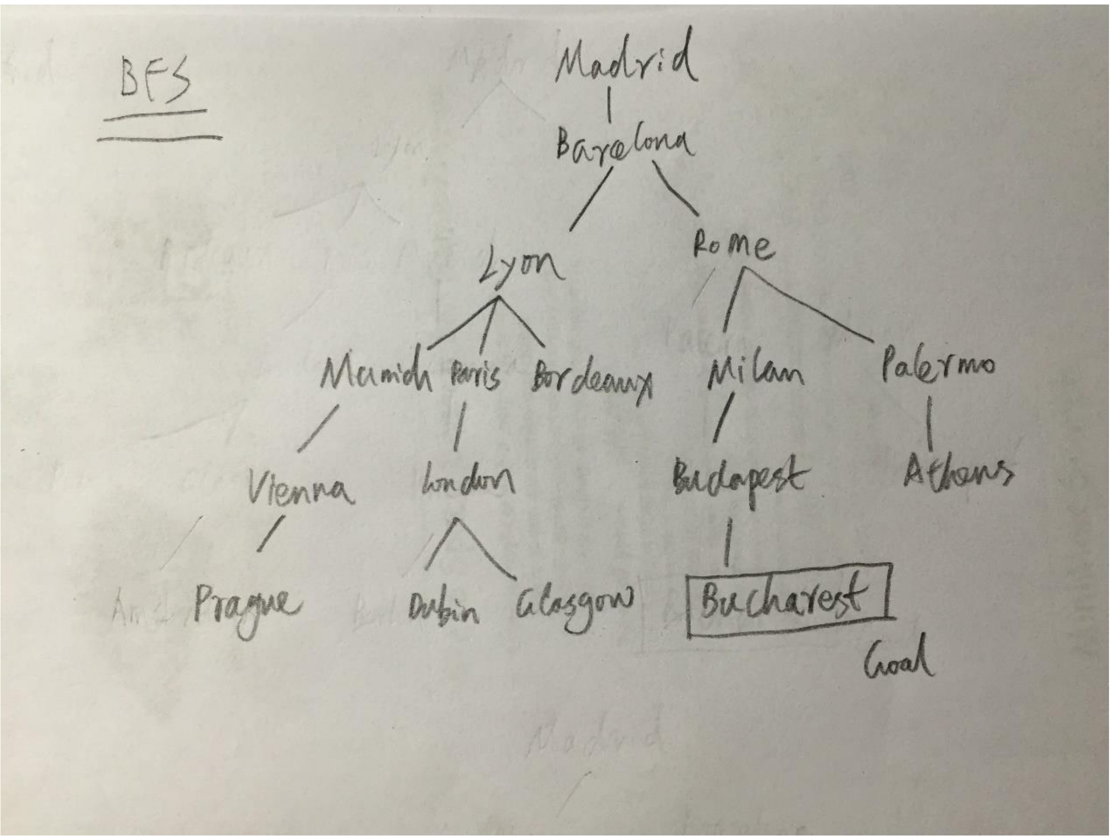
Q-4.E

Answer:

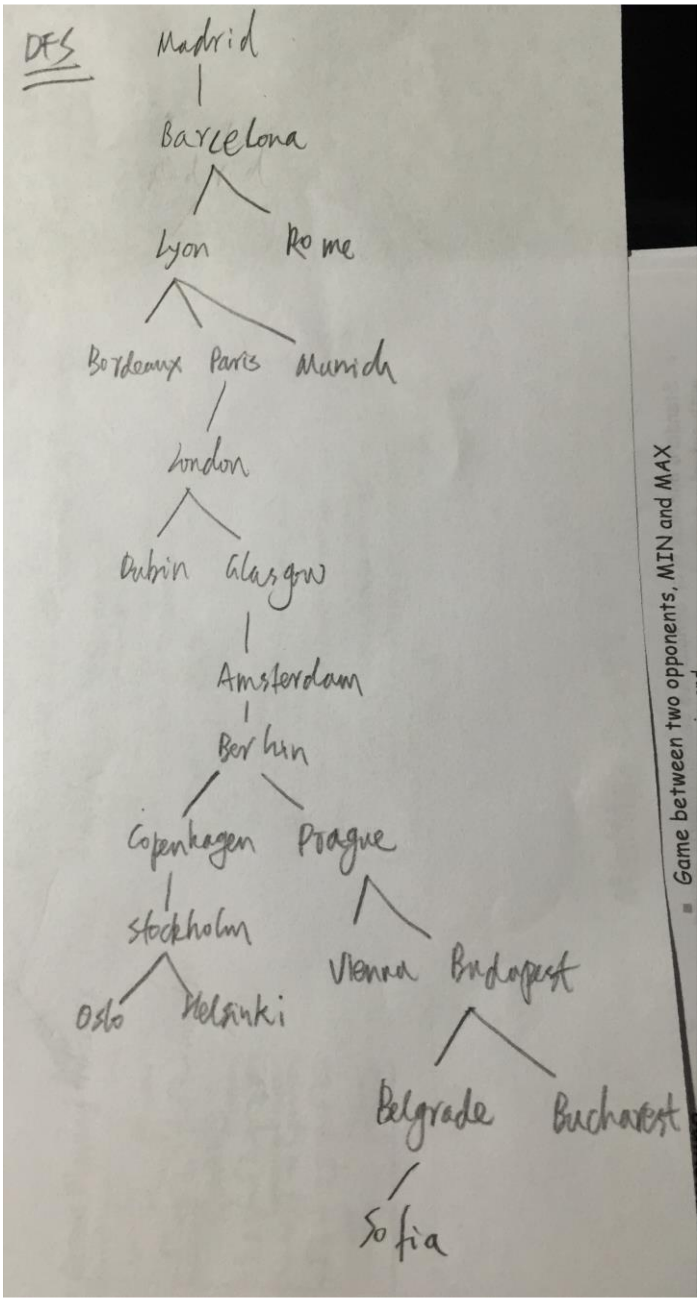
a) The initial state is starting from Madrid, the goal state is arriving Bucharest. Action is move from a city to its successors. The state space is the set of all states travelling from Madrid to Bucharest.

Its size is (27-1)!.

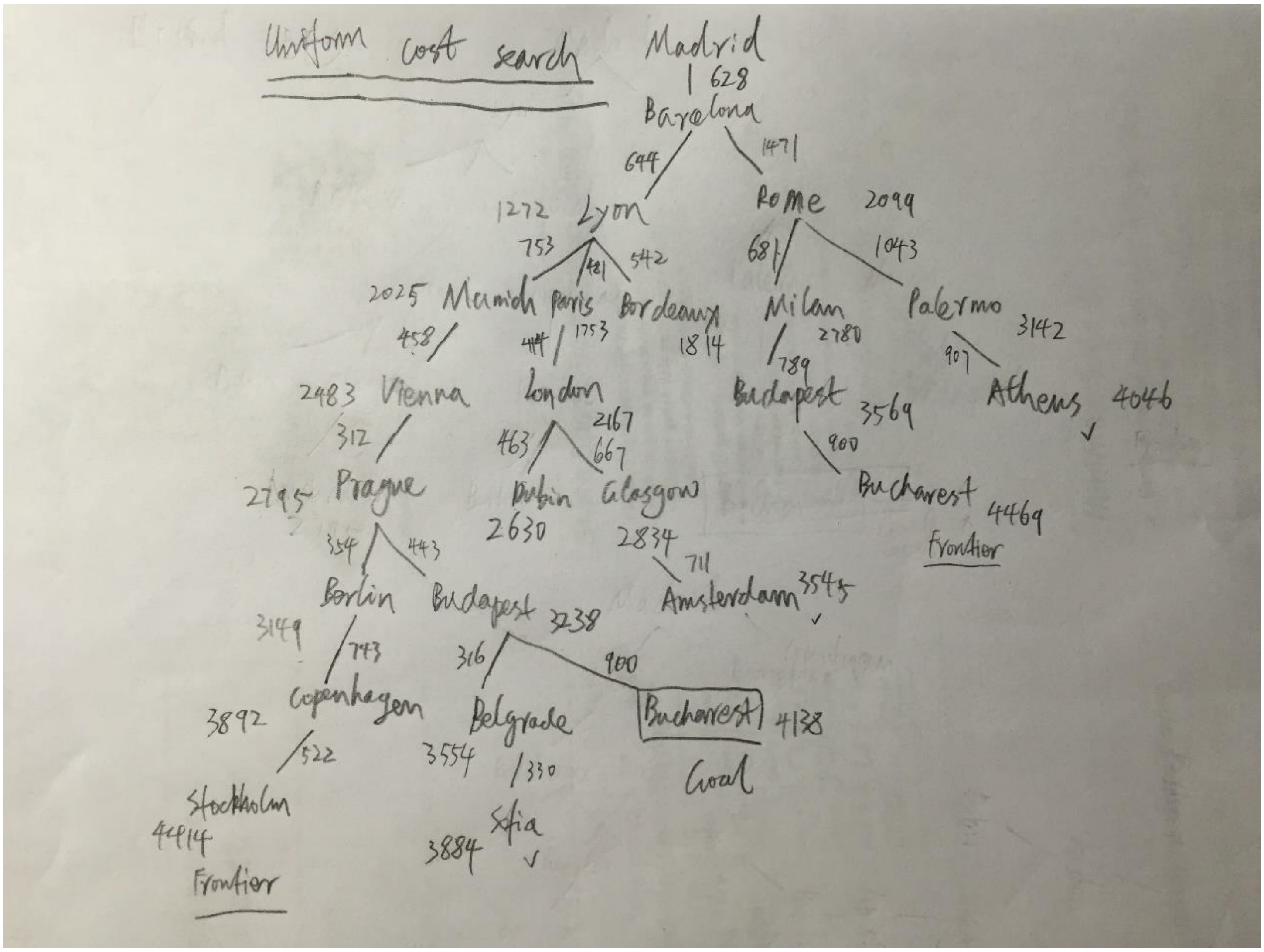
b) Search tree for BFS expands 17 nodes.



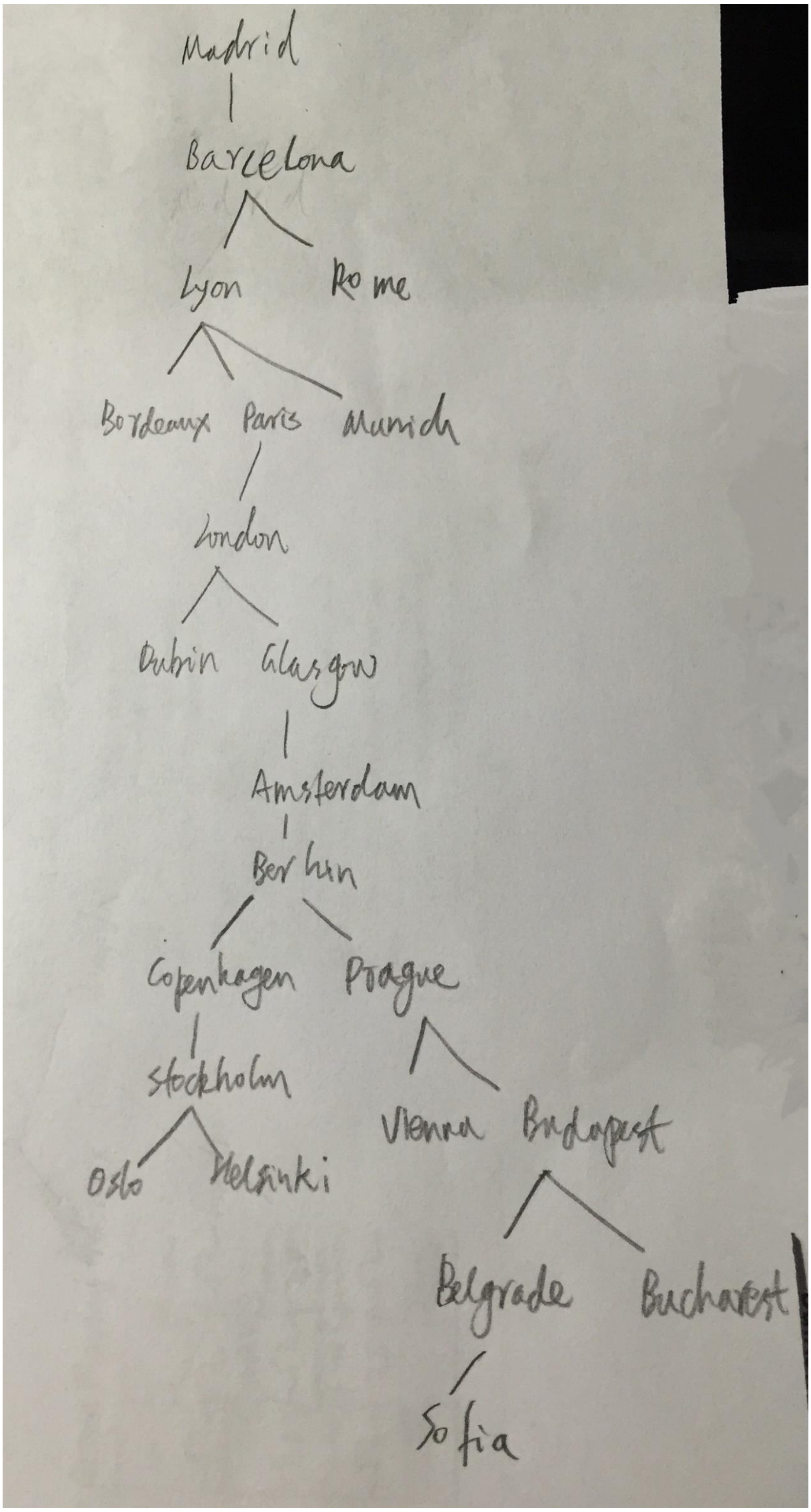
c) Search tree for DFS expands 22 nodes



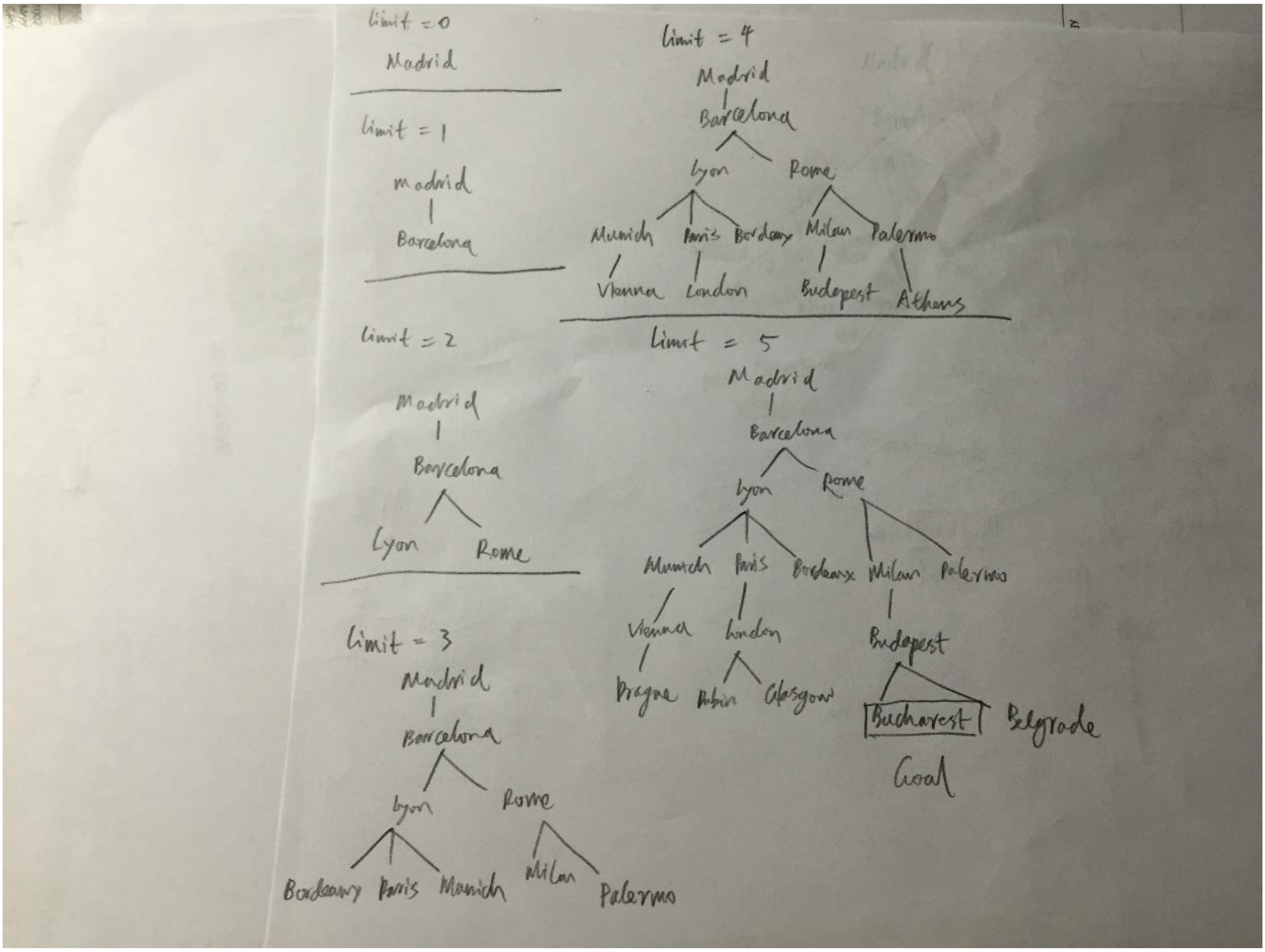
d) Search tree for uniform cost search expands 23 nodes.



e) Backtracking



f) Iterative deepening search.



g) BFS can always find the solution but time and space cost is high. DFS needs little space and can get the solution fast if the branching factor is acceptable. Uniform cost search is kind of the best first search, using the path cost as evaluation function, it's an exhaustive search, and the first solution maybe not the best.

For this problem, iterative deepening is the best way to solve it. Less space and efficient.

If we don't define the loop constraint for the search algorithm, the DFS will get stuck into loops, and may not get solution in reasonable time.