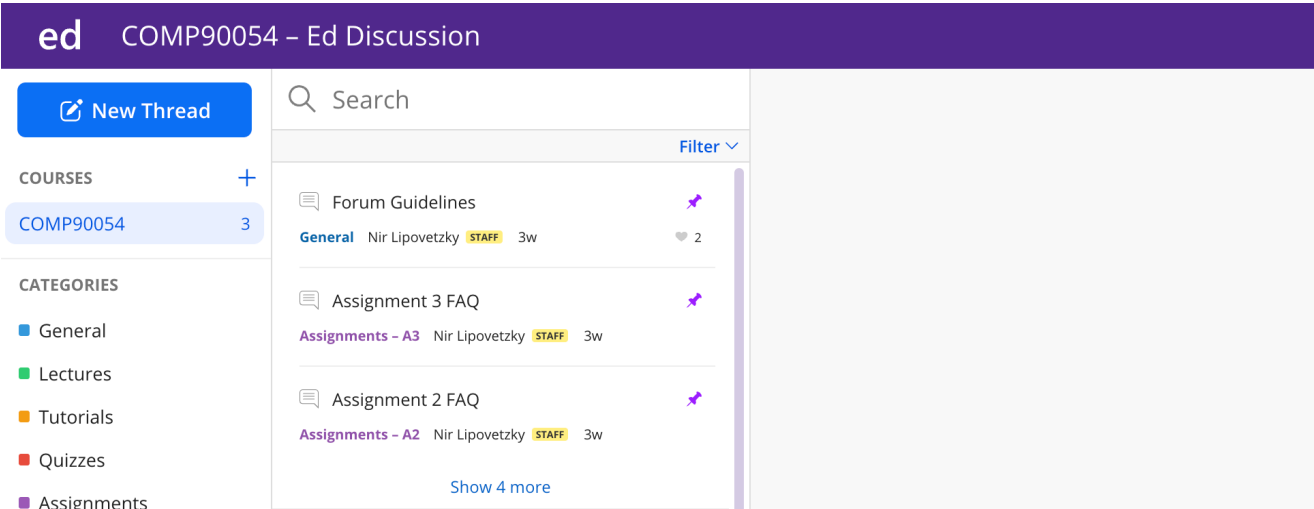


Week 2: Blind Search

COMP90054 – AI Planning for Autonomy

- Tutor Name: Thao Le
- Questions? Ed Discussion



The screenshot shows the 'ed COMP90054 - Ed Discussion' forum interface. On the left, there is a sidebar with a 'New Thread' button and a list of 'COURSES' and 'CATEGORIES'. The 'COURSES' list includes 'COMP90054' with a count of 3. The 'CATEGORIES' list includes 'General', 'Lectures', 'Tutorials', 'Quizzes', and 'Assignments'. The main content area features a search bar, a 'Filter' dropdown, and a list of forum threads. The threads listed are 'Forum Guidelines', 'Assignment 3 FAQ', and 'Assignment 2 FAQ'. Each thread shows its category, author (Nir Lipovetzky), role (STAFF), and time (3w). The 'Forum Guidelines' thread has 2 likes. A 'Show 4 more' link is visible at the bottom of the thread list.

ed COMP90054 - Ed Discussion

[New Thread](#)

COURSES +

COMP90054 3

CATEGORIES

- General
- Lectures
- Tutorials
- Quizzes
- Assignments

Search

Filter ▾

- Forum Guidelines
General Nir Lipovetzky STAFF 3w 2
- Assignment 3 FAQ
Assignments - A3 Nir Lipovetzky STAFF 3w
- Assignment 2 FAQ
Assignments - A2 Nir Lipovetzky STAFF 3w

[Show 4 more](#)

Key concepts

- State-space model
- Blind search algorithms:
 - Breadth First Search (BFS)
 - Depth First Search (DFS)
 - Iterative Deepening (ID)

Problem 1

State space model:

$$P = \langle s_0, S, S_G, A, f, c \rangle$$

- s_0 is an initial state

s_1

- S is a set that includes all states in the state space

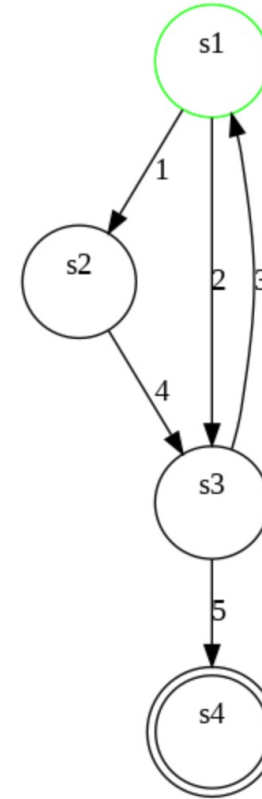
$$S = \{s_1, s_2, s_3, s_4\}$$

- S_G is a goal set with all goal states

$$S_G = \{s_4\}$$

▼ Problem 1:

Following the above example, define the state-space model of the graph:



Problem 1

State space model:

$$P = \langle s_0, S, S_G, A, f, c \rangle$$

- A is an action set that includes all possible actions you can take from a state

$$A(s_1) = \{(s_1, s_2), (s_1, s_3)\}$$

$$A(s_4) = ?$$

$$A(s_4) = \{ \}$$

- f is a transition function: $f(s, a) = s'$

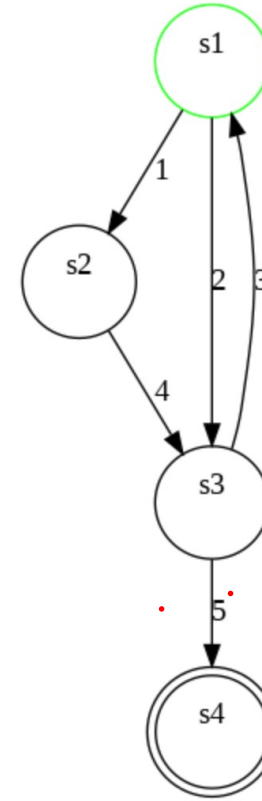
$$f(s_1, (s_1, s_2)) = s_2$$

- c is a cost function between two states

$$c(s_1, s_2) = 1$$

Problem 1:

Following the above example, define the state-space model of the graph:

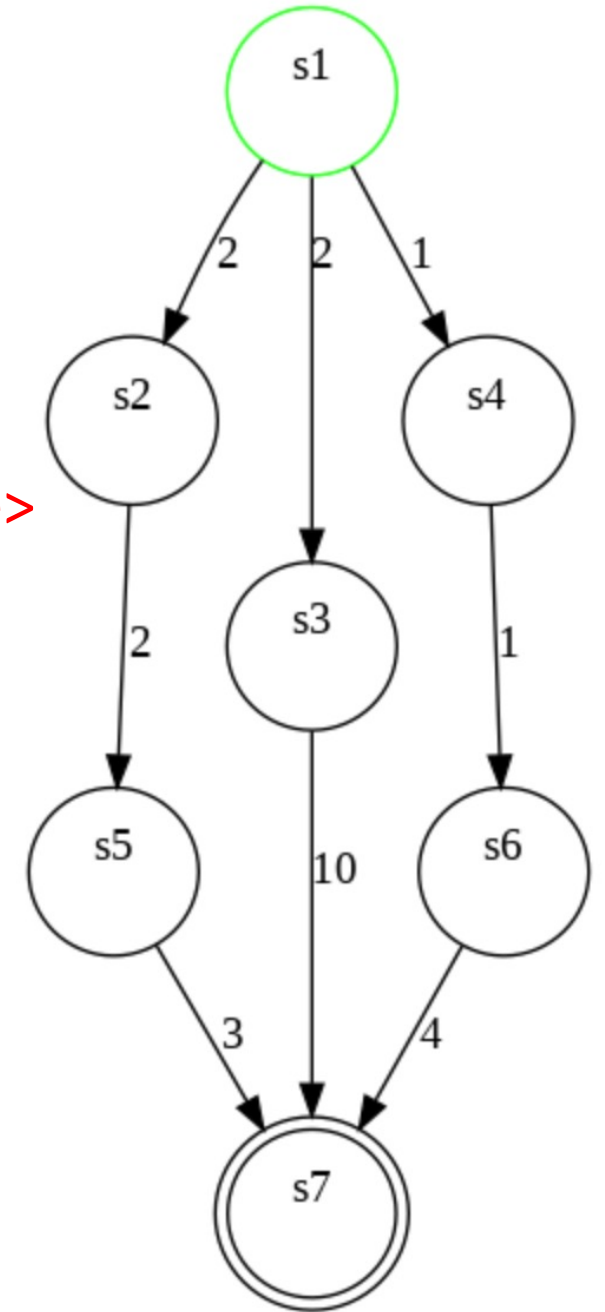


Problem 2

State vs Search node?

Search node $n = \langle \text{state}, \text{accumulated cost } g(n), \text{id of parent node} \rangle$

$n_0 = \langle s_1, 0, \text{None} \rangle$



Problem 2

Breadth First Search (BFS)

Queue: FIFO

Search node $n = (\text{state}, \text{accumulated cost } g(n), \text{id of parent node})$

$n_0 = (s_1, 0, \text{None})$

$n_1 = (s_2, 2, 0)$

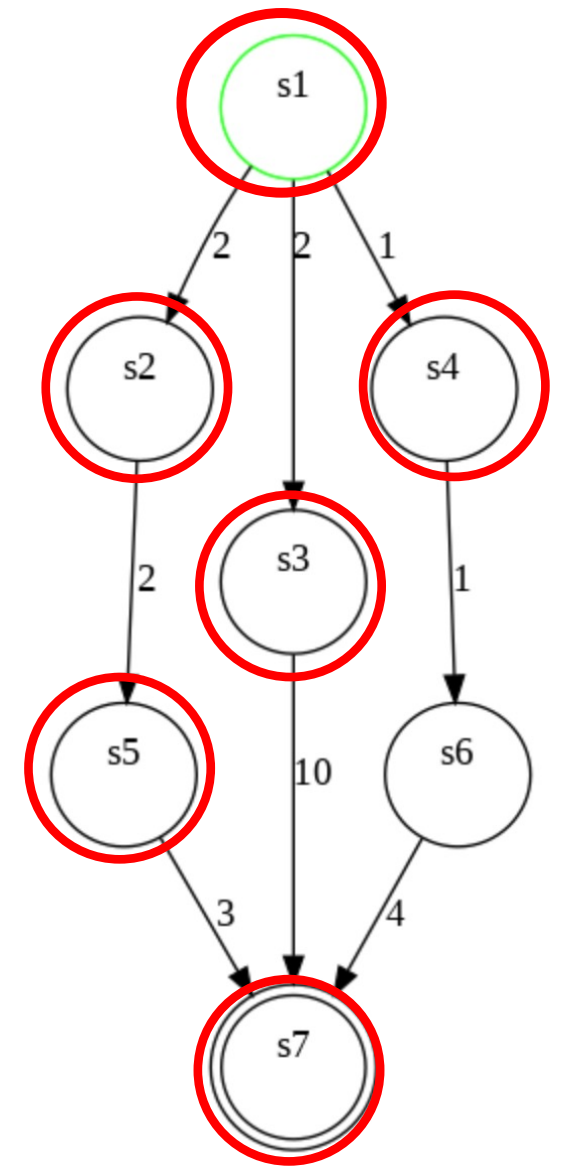
$n_2 = (s_3, 2, 0)$

$n_3 = (s_4, 1, 0)$

$n_4 = (s_5, 4, 1)$

$n_5 = (s_7, 12, 2)$

Solution: $s_1 \rightarrow s_3 \rightarrow s_7$



Problem 2

Depth First Search (DFS)

Stack: LIFO

Search node $n = (\text{state}, \text{accumulated cost } g(n), \text{id of parent node})$

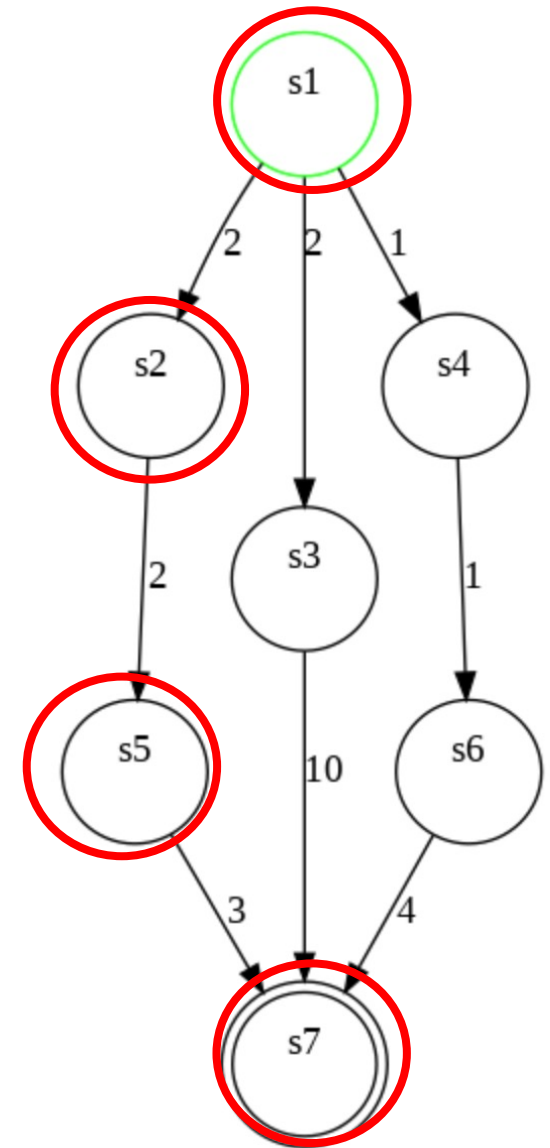
$n_0 = (s_1, 0, \text{None})$

$n_1 = (s_2, 2, 0)$

$n_2 = (s_5, 4, 1)$

$n_3 = (s_7, 7, 2)$

Solution: $s_1 \rightarrow s_2 \rightarrow s_5 \rightarrow s_7$

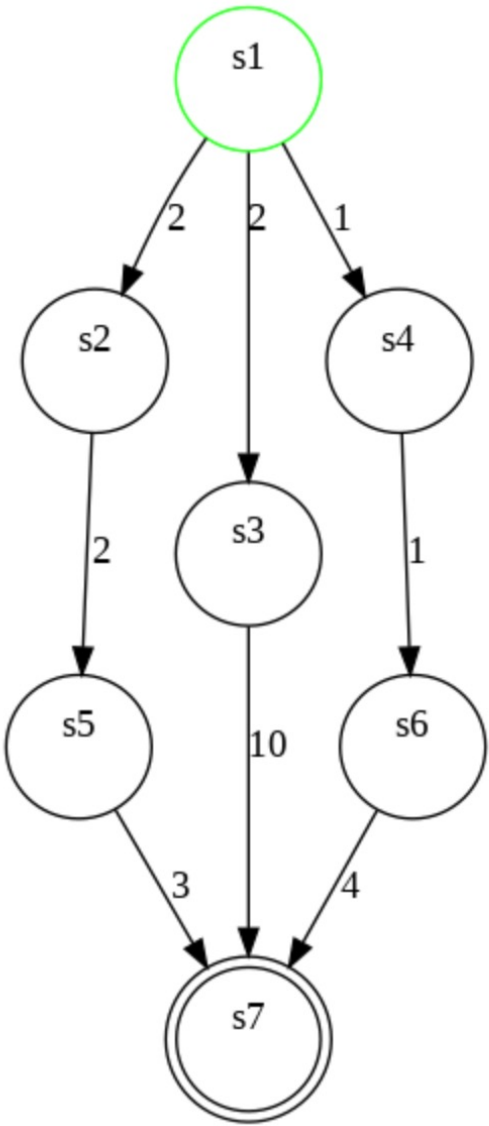


Question 2

Depth First Search (DFS)

	Open (Stack)	Close (Visited)
Iteration 1	n0 = <s1, 0, None>	
Iteration 2	n1 = <s2, 2, 0> n2 = <s3, 2, 0> n3 = <s4, 1, 0>	n0
Iteration 3	n4 = <s5, 4, 1> n2 = <s3, 2, 0> n3 = <s4, 1, 0>	n0, n1
Iteration 4	n5 = <s7, 7, 4> n2 = <s3, 2, 0> n3 = <s4, 1, 0>	n0, n1, n4
Iteration 5	n2 = <s3, 2, 0> n3 = <s4, 1, 0>	n0, n1, n4, n5

(s1, 0 ,None),
(s2, 2 ,0),
(s5, 4 ,1),
(s7, 7 ,2)



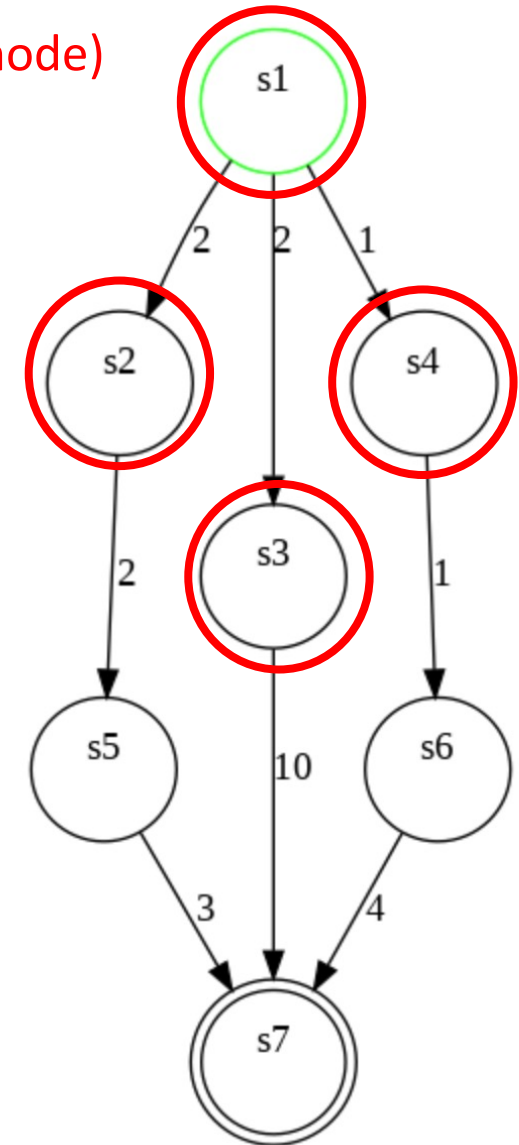
Problem 2

Search node $n = (\text{state}, \text{accumulated cost } g(n), \text{id of parent node})$

Iterative Deepening (ID)

DFS with a depth limit

Limit	Step	Open (Stack)	Close
0	1	$n_0 = (s_1, 0, \text{None})$	
	2		n_0
1	3	$n_1 = (s_1, 0, \text{None})$	
	4	$n_2 = (s_2, 2, 1)$ $n_3 = (s_3, 2, 1)$ $n_4 = (s_4, 1, 1)$	n_1
	5	$n_3 = (s_3, 2, 1)$ $n_4 = (s_4, 1, 1)$	n_1, n_2
	6	$n_4 = (s_4, 1, 1)$	n_1, n_2, n_3
	7		n_1, n_2, n_3, n_4



Problem 2

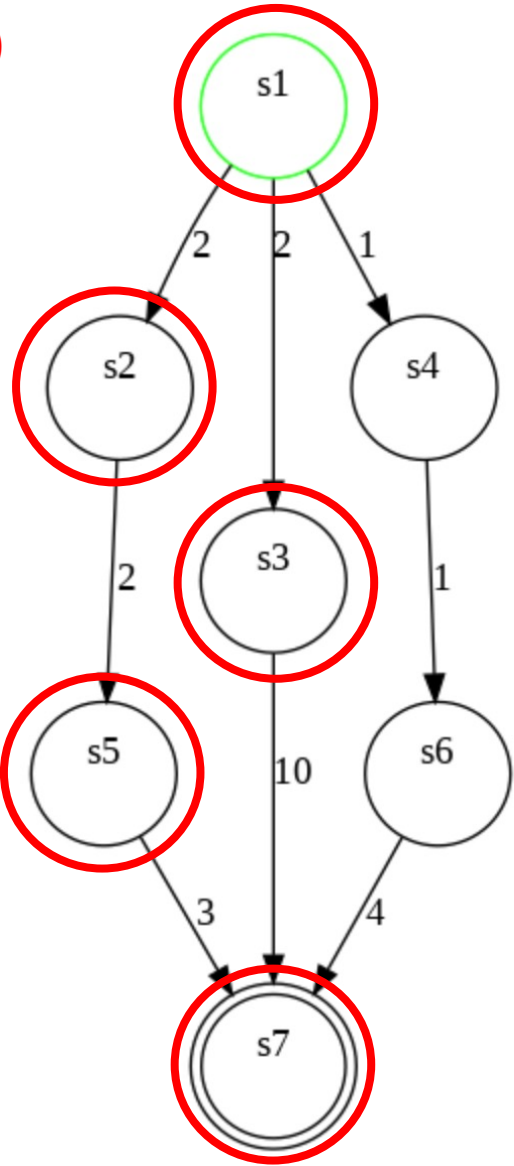
Search node n = (state, accumulated cost g(n), id of parent node)

Iterative Deepening (ID)

DFS with depth limit

Limit	Step	Open (Stack)	Close
2	8	n5 = (s1, 0, None)	
	9	n6 = (s2, 2, 5) n7 = (s3, 2, 5) n8 = (s4, 1, 5)	n5
	10	n9 = (s5, 4, 6) n7 = (s3, 2, 5) n8 = (s4, 1, 5)	n5, n6
	11	n7 = (s3, 2, 5) n8 = (s4, 1, 5)	n5, n6, n9
	12	n10 = (s7, 12, 7) n8 = (s4, 1, 5)	n5, n6, n9, n7
	13	n8 = (s4, 1, 5)	n5, n6, n9, n7, n10

n0,
n1, n2, n3, n4,
n5, n6, n9, n7, n10



Problem 2

Search node $n = (\text{state}, \text{accumulated cost } g(n), \text{id of parent node})$

Iterative Deepening (ID)

DFS with depth limit

Expansion node order

$n_0 = (s_1, 0, \text{None}),$

Expansion node order

$n_0,$
 $n_1, n_2, n_3, n_4,$
 $n_5, n_6, n_9, n_7, n_{10}$

$n_1 = (s_1, 0, \text{None}),$

$n_2 = (s_2, 2, 1),$

$n_3 = (s_3, 2, 1),$

$n_4 = (s_4, 1, 1),$

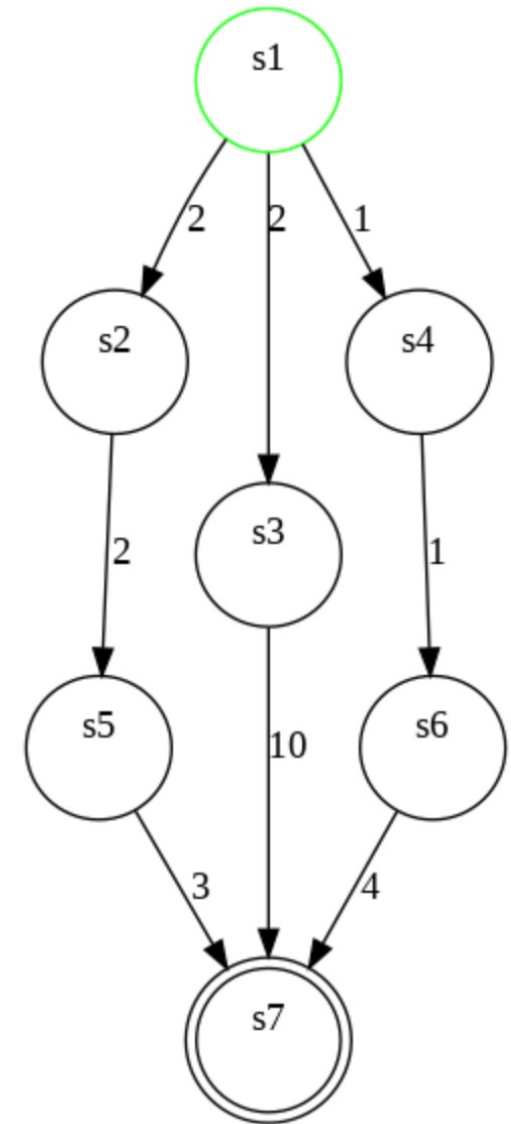
$n_5 = (s_1, 0, \text{None}),$

$n_6 = (s_2, 2, 5),$

$n_9 = (s_5, 4, 6),$

$n_7 = (s_3, 2, 5),$

$n_{10} = (s_7, 12, 7)$



Solution: $s_1 \rightarrow s_3 \rightarrow s_7$

Problem 2

Q2: What is the actual optimal solution?

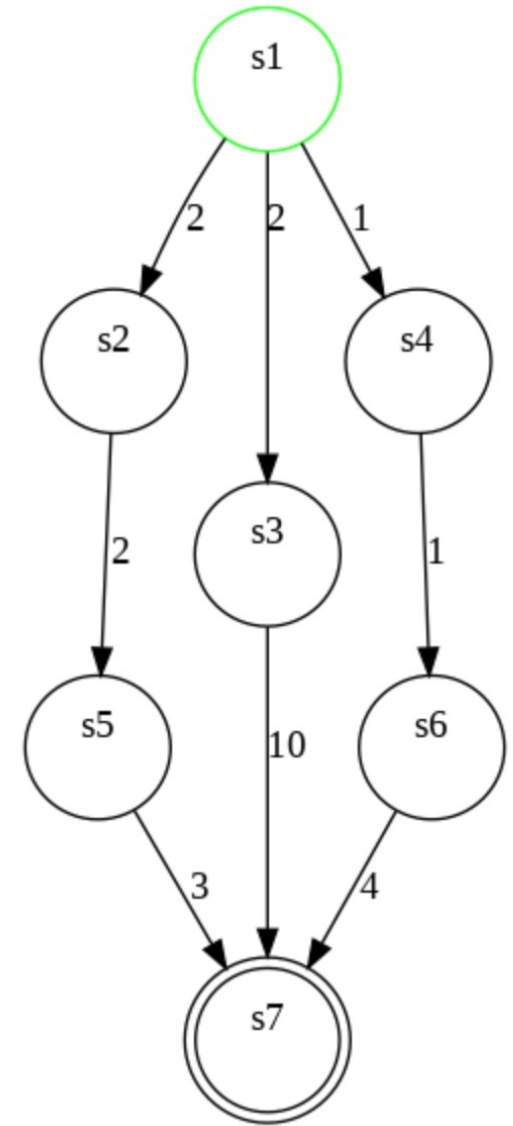
s1 -> s4 -> s6 -> s7

Q3: Explain under which conditions the algorithms guarantee optimality?

BrFS and ID will be optimal if the costs are equal, such as, all cost are 1

Q4: Can any of the previous algorithms be adapted to account for $g(n)$ in order to make it optimal?

Dijkstra. Expanding the node with lowest accumulated cost, instead of the lowest depth.



Problem 3

Describe a simple example of *Travelling Salesman Problem* along with its corresponding **State Space Model**.

Definition should be brief, clear, and *compact* (*compact* means using mathematical notation to define sets, i.e. $S = \{x|x \in V\}$ to define that there are as many states as elements in the set V , and pseudo-code, i.e. to define the transition function.)

1. State space S
2. Initial state $s_0 \in S$
3. Set of goal states $S_G \subseteq S$
4. Applicable actions function $A(s)$ for each state $s \in S$
5. Transition function $f(s, a)$ for $s \in S$ and $a \in A(s)$
6. Cost of each action $c(a)$ for $a \in A(s)$

Hint: Given

- V = a set of cities
- v_{start} = a starting city location
- E = a set of edges specifying if there is an edge between two cities $\langle v1, v2 \rangle$
- V' = a set of cities that have been visited

Note for the TSP :

- A city can be visited more than once
- You can start at any city
- The goal state is to visit all cities and no need to go back to the starting city

Problem 3

Hint: Given

- V = a set of cities
- v_{start} = a starting city location
- E = a set of edges specifying if there is an edge between two cities $\langle v_1, v_2 \rangle$
- V' = a set of cities that have been visited

a state = \langle current city, a set of visited cities \rangle

an edge/action = \langle current city, next city \rangle

Initial state $s_0 = \langle v_{start}, \{v_{start}\} \rangle$

Goal state $S_G = \{ \langle v_{current}, V \rangle \mid v_{current} \in V \}$

State $S = \{ \langle v_{current}, V' \rangle \mid v_{current} \in V \wedge V' \subseteq V \}$

Action $A(\langle v_{current}, V' \rangle) = \{ \langle v_{current}, v_{next} \rangle \mid \langle v_{current}, v_{next} \rangle \in E \}$

Transition $f(\langle v_{current}, V' \rangle, \langle v_{current}, v_{next} \rangle) = \langle v_{next}, V' \cup \{v_{next}\} \rangle$

c($\langle v_{current}, v_{next} \rangle$) = **cost**($\langle v_{current}, v_{next} \rangle$)

Criteria for Evaluating Search Strategies

Completeness: Is the strategy guaranteed to find a solution when there is one?

Optimality: Are the returned solutions guaranteed to be optimal?

Criteria for Evaluating Search Strategies

Consider when the graph is finite and cyclic

	BFS	DFS	ID
Complete	Yes	No	Yes
Optimal	Yes when costs are all equal	No	Yes when costs are all equal
Time complexity	$O(b^d)$	$O(b^m)$	$O(b^d)$
Space complexity	$O(b^d)$	$O(b.m)$	$O(b.d)$

b is the maximal branching factor,
d is the depth of the shallowest goal state
m is maximal depth