Searching With Costs

13/01/2025

Koustav Rudra

Search and Optimization

- Given: [S, s, O, G]

- To find:
  - A minimum cost sequence of transitions to a goal state
  - A sequence of transitions to the minimum cost goal
  - A minimum cost sequence of transitions to a minimum cost goal

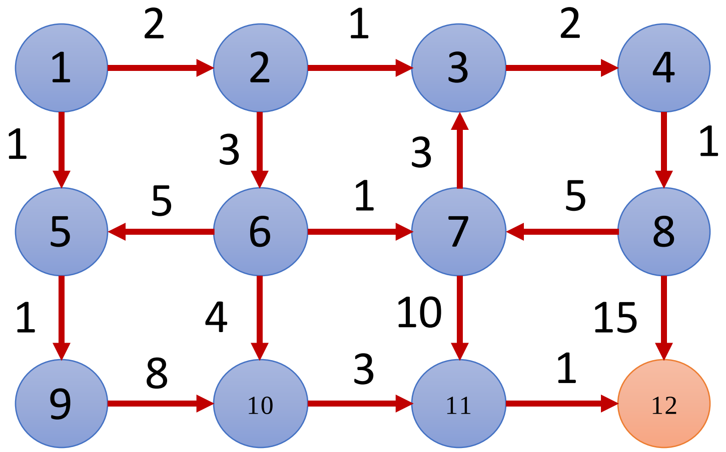
Search with Cost

- Initialize: Set OPEN={s}, CLOSED = {}, Set C(s)=0
- Fail:
  - If OPEN={}, Terminate with failure
- Select: Select the minimum cost state, n, from OPEN and
  - Save n is CLOSED
- Terminate:
  - If n∈G, terminate with success

Search with Cost

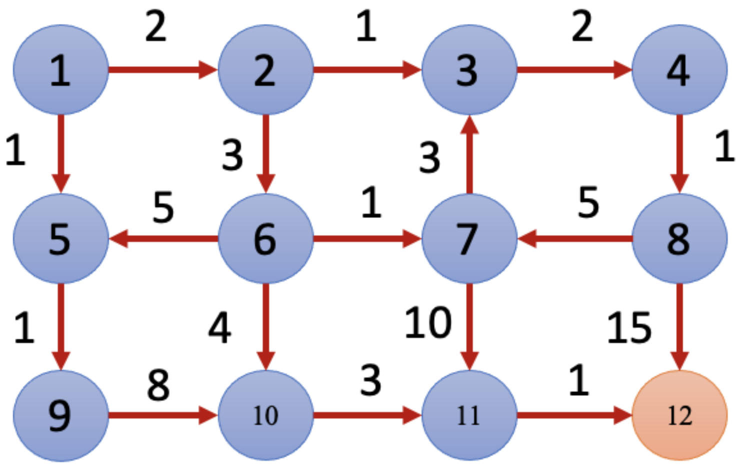
- Expand:
  - Generate successors of n using O
  - For each successor, m:
    - If m∉[OPEN∪CLOSED]
      - Set $C(m) = C(n) + C(n,m)$
      - Insert m in OPEN
    - If m∈[OPEN∪CLOSED]
      - Set C(m) = $min\begin{cases} C(m) \\ C(n) + C(n,m) \end{cases}$
      - If C(m) has decreased and $m \in CLOSED$
        - Move m to OPEN
- Loop
  - Go to step 2

Search with Cost



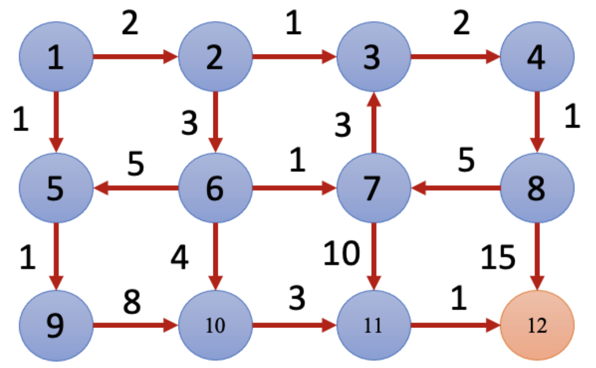
OPEN SET	SELECT	GOAL	EXPANDED	CLOSED
[1(0)]	1(0)	N	[2(2),5(1)]	[1(0)]
[2(2),5(1)]	5(1)	N	[2(2),9(2)]	[1(0),5(1)]
[2(2),9(2)]	2(2)	N	[9(2),3(3),6(5)]	[1(0),5(1),2(2)]
[9(2),3(3),6(5)]	9(2)	N	[3(3),6(5),10(10)]	[1(0),5(1),2(2),9(2)]
[3(3),6(5),10(10)]	3(3)	N	[6(5),10(10),4(5)]	[1(0),5(1),2(2),9(2),3(3)]

Uniform Cost Search



OPEN SET	SELECT	GOAL	EXPANDED	CLOSED
[3(3),6(5),10(10)]	3(3)	N	[6(5),10(10),4(5)]	[1(0),5(1),2(2),9(2),3(3)]
[6(5),10(10),4(5)]	6(5)	N	[10(9),4(5),7(6)]	[1(0),5(1),2(2),9(2),3(3),6(5)]
[10(9),4(5),7(6)]	4(5)	N	[10(9),7(6),8(6)]	[1(0),5(1),2(2),9(2),3(3),6(5),4(5)]
[10(9),7(6),8(6)]	7(6)	N	[10(9),8(6),11(16)]	[1(0),5(1),2(2),9(2),3(3),6(5),4(5),7(6)]
[10(9),8(6),11(16)]	8(6)	N	[10(9),11(16),12(21)]	[1(0),5(1),2(2),9(2),3(3),6(5),4(5),7(6),8(6)]
[10(9),11(16),12(21)]	10(9)	N	[11(12),12(21)]	[1(0),5(1),2(2),9(2),3(3),6(5),4(5),7(6),8(6),10(9)]

Uniform Cost Search



OPEN SET	SELECT	GOAL	EXPANDED	CLOSED
[10(9),7(6),8(6)]	7(6)	N	[10(9),8(6),11(16)]	[1(0),5(1),2(2),9(2),3(3),6(5),4(5),7(6)]
[10(9),8(6),11(16)]	8(6)	N	[10(9),11(16),12(21)]	[1(0),5(1),2(2),9(2),3(3),6(5),4(5),7(6),8(6)]
[10(9),11(16),12(21)]	10(9)	N	[11(12),12(21)]	[1(0),5(1),2(2),9(2),3(3),6(5),4(5),7(6),8(6),10(9)]
[11(12),12(21)]	11(12)	N	[12(13)]	[1(0),5(1),2(2),9(2),3(3),6(5),4(5),7(6),8(6),10(9),11(12)]
[12(13)]	12(13)	Y		

Searching with Cost

- If all operator costs are positive, then the
  - Algorithm finds the minimum cost sequence of transitions to a goal

  - No state comes back to OPEN from CLOSED

  - When expanding a node, there is no other node which has lesser cost
    - Because we have explored the frontier up to that cost
    - Any node beyond that frontier is going to add more positive cost to it

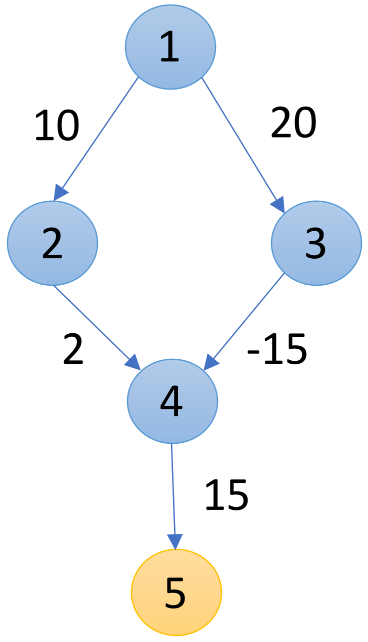
- What will happen if operators have unit cost?
  - BFS

Searching with Cost

- What will happen if we have negative edge cost?
  - Might be possible that successor of a node have negative cost
  - May have to bring a node from CLOSED to OPEN

- What will happen in case of negative cycle?

Searching with Cost



OPEN SET	SELECT	GOAL	EXPANDED	CLOSED
[1(0)]	1(0)	N	[2(10),3(20)]	[1(0)]
[2(10),3(20)]	2(10)	N	[3(20),4(12)]	[1(0),2(10)]
[3(20),4(12)]	4(12)	N	[3(20),5(27)]	[1(0),2(10),4(12)]
[3(20),5(27)]	3(20)	N	[4(5),5(27)]	[1(0),2(10),~~4(12)~~,3(20)]
[4(5),5(27)]	4(5)	N	[5(20)]	[1(0),2(10),~~4(12)~~,3(20),4(5)]
[5(20)]	5(20)	Y		

Branch and Bound

13/01/2025

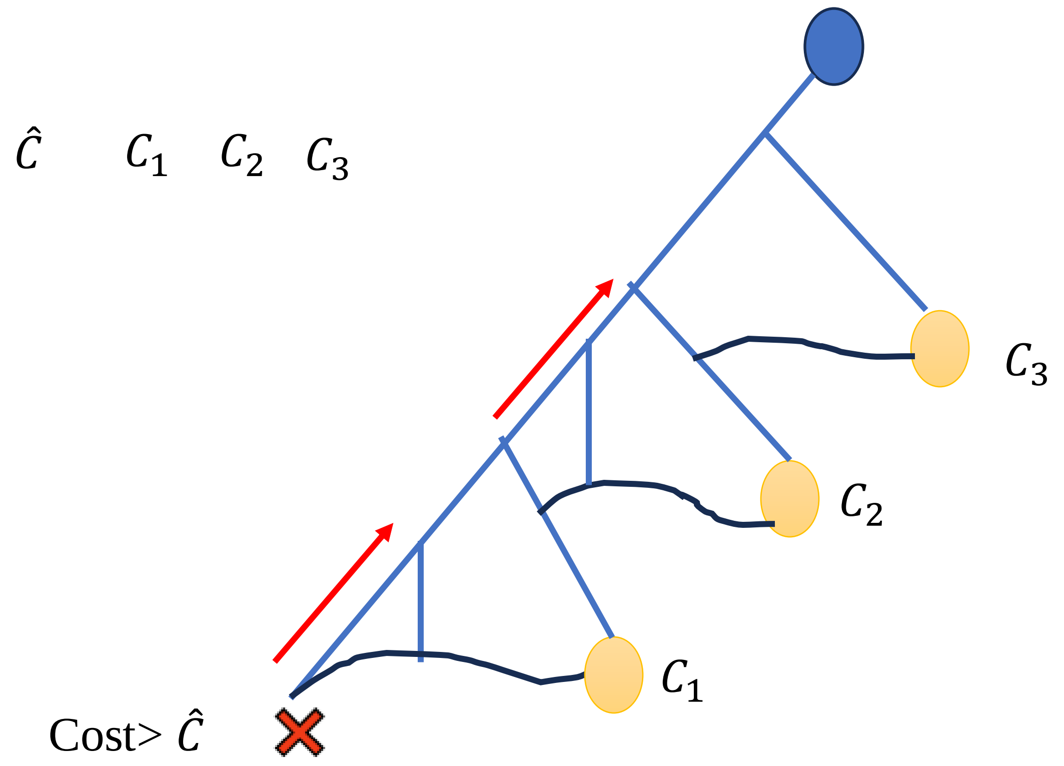
Koustav Rudra

Branch and Bound

- We know an upper bound on solution cost

- We can do BFS/DFS to find out a goal and its associated cost
  - But it is not guaranteed that it is least cost goal
  - However, it works as an upper bound

Branch and Bound



- How long should we do this?
  - Until we search the entire search space
- Apply a greedy approximation
- Get a relatively tight upper bound

Branch and Bound

- Initialize: Set OPEN={s}, CLOSED = {}, Set C(s)=0, $C^* = \infty$
- Fail:
  - If OPEN={}, then return C^*
- Select: Select a state, n, from OPEN and save in CLOSED
- Terminate:
  - If n∈G, and C(n) < C^*, then
    - C^*=C(n) and Go To Step 2

Branch and Bound

- Expand:
  - If C(n) < C^* , Generate successors of n using O
  - For each successor, m:
    - If m∉[OPEN∪CLOSED]
      - Set $C(m) = C(n) + C(n, m)$
      - Insert m in OPEN
    - If m∈[OPEN∪CLOSED]
      - Set C(m) = $min \begin{cases} C(m) \\ C(n) + C(n, m) \end{cases}$
      - If C(m) has decreased and $m \in CLOSED$
        - Move m to OPEN

- Loop:
  - Go to step 2

Search Factors

- Branching factor

- Some state spaces which are shallow, but which have a lot of breadth
  - Travelling Salesman Problem
  - BSBB / Iterative Deepening

- Some state spaces depth are very large

Thank You