AIFA
Basic State Space Search

08/01/2024

Koustav Rudra

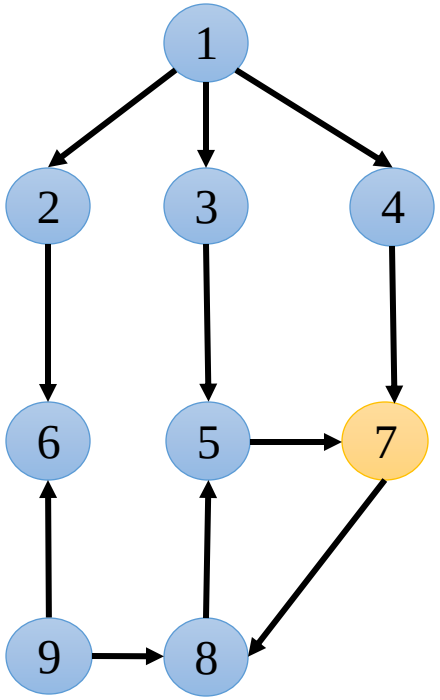
Basic Search Algorithm

- Initialize: Set OPEN={s}
- Fail:
  - If OPEN={}, Terminate with failure
- Select: Select a state, n, from OPEN
- Terminate:
  - If n∈G, terminate with success
- Expand:
  - Generate the successors of n using O and insert them in OPEN
- Loop:
  - Go to step 2

Which data structure should we use for OPEN?

Basic Search Algorithm

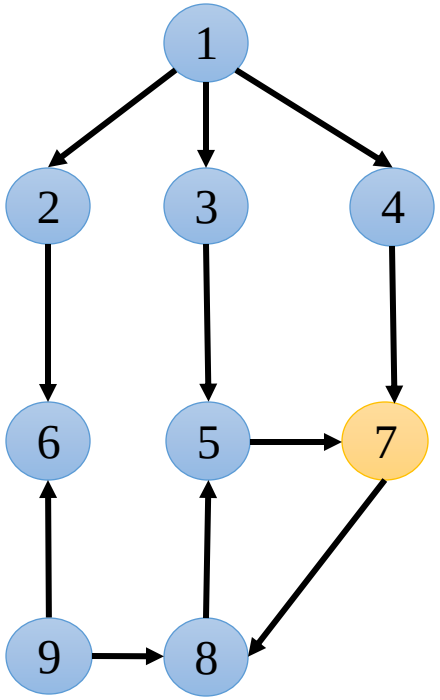
Open Set	Select State	Goal State	Terminate	Expanded Set
[1]	1	N	N	[4,3,2]
[4,3,2]	2	N	N	[4,3,6]
[4,3,6]	6	N	N	[4,3]
[4,3]	3	N	N	[4,5]
[4,5]	5	N	N	[4,7]
[4,7]	7	Y	Y	



Stack
Tie: Descending

Basic Search Algorithm

Open Set	Select State	Goal State	Terminate	Expanded Set
[1]	1	N	N	[4,3,2]
[4,3,2]	4	N	N	[3,2,7]
[3,2,7]	3	N	N	[2,7,5]
[2,7,5]	2	N	N	[7,5,6]
[7,5,6]	7	Y	Y	



Queue
Tie: Descending

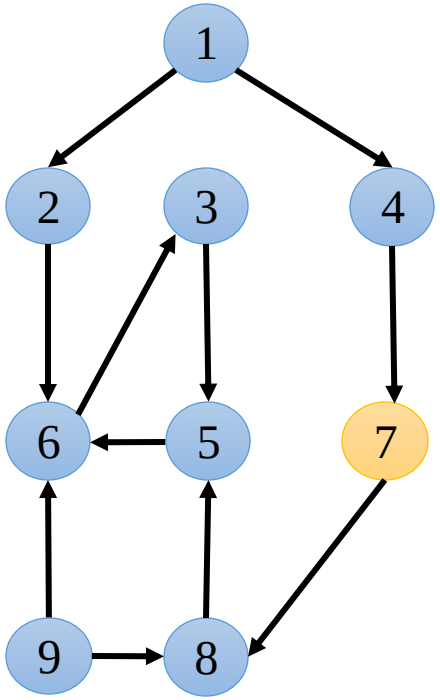
Basic Search Algorithm

- We don’t want to make whole state space explicit

- We only want to unfold that portion of the state space which is necessary to find out the goal

Basic Search Algorithm

Open Set	Select State	Goal State	Terminate	Expanded Set
[1]	1	N	N	[4,2]
[4,2]	2	N	N	[4,6]
[4,6]	6	N	N	[4,3]
[4,3]	3	N	N	[4,5]
[4,5]	5	N	N	[4,6]
[4,6]	6	N	N	[4,3]
[4,3]	3	N	N	[4,5]



Stack
Tie: Descending

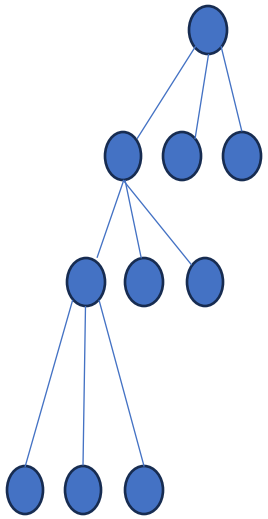
How to maintain part of the state space that are already visited?

Basics Search Algorithm

- OPEN is a queue (FIFO) vs a stack (LIFO)
- Is this algorithm guaranteed to terminate?
- Under what circumstances will it terminate?

Complexity

- b: branching factor d: depth of the goal



- Breadth first search:
  - Time: $O(b^d)$
  - Space: $O(b^d)$

- Depth first search:
  - Time: $O(b^m)$
    - m: depth of state space tree
  - Space: $O(bm)$
  - State space tree is the tree that is obtained by applying the state transition operators repeatedly on the set of states

Trade-off between Space and Time

- Iterative deepening
  - Perform DFS repeatedly using increasing depth bounds
  - Works in $O(b^d)$ time and O(bd) space

- Can we do something with time complexity?

Trade-off between Space and Time

- Iterative deepening
  - Perform DFS repeatedly using increasing depth bounds
  - Works in $O(b^d)$ time and O(bd) space

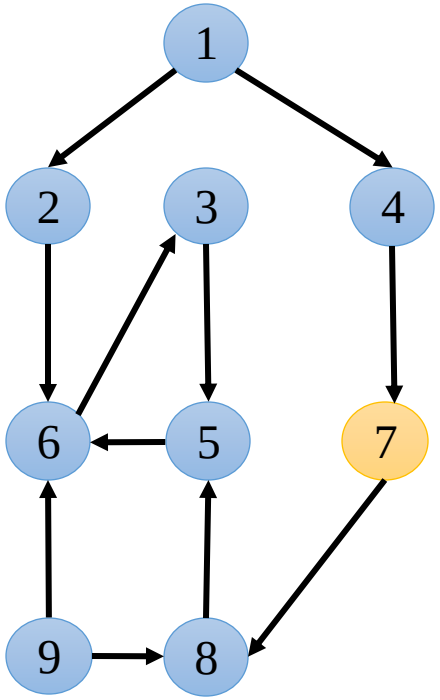
- Bi-directional search
  - Possible only if the operators are reversible
  - Works in $O(b^{\frac{d}{2}})$ time and $O(b^{\frac{d}{2}})$ space

Complexity Comparison

Criterion	BFS	DFS	Depth Limited
Time	b^d	b^m	b^l
Space	b^d	bm	bl
Optimal?	Yes	No	No
Complete?	Yes	No	Yes, if l>=d

Basic Search Algorithm

Open Set	Select State	Goal State	Terminate	Expanded Set
[1]	1	N	N	[4,2]
[4,2]	2	N	N	[4,6]
[4,6]	6	N	N	[4,3]
[4,3]	3	N	N	[4,5]
[4,5]	5	N	N	[4,6]
[4,6]	6	N	N	[4,3]
[4,3]	3	N	N	[4,5]



Stack
Tie: Descending

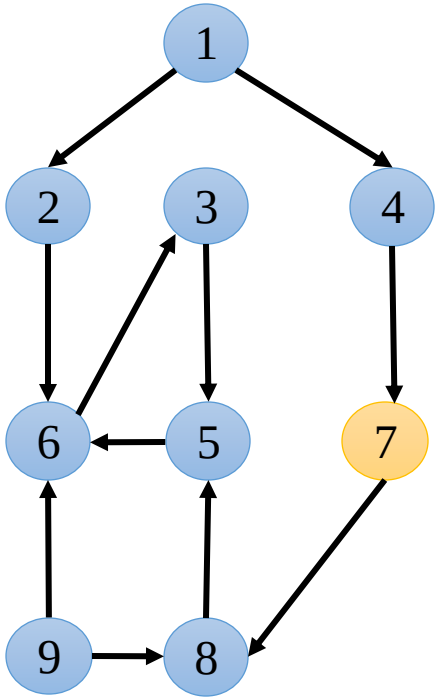
How to maintain part of the state space that are already visited?

Search Algorithm: Saving Explicit Space

- Initialize: Set OPEN={s}, CLOSED = {}
- Fail:
  - If OPEN={}, Terminate with failure
- Select: Select a state, n, from OPEN and
  - Save n is CLOSED
- Terminate:
  - If n∈G, terminate with success
- Expand:
  - Generate the successors of n using O
  - For each successor, m, insert m in OPEN,
    - Only if $m \notin [OPEN \cup CLOSED]$
- Loop:
  - Go to step 2

Basic Search Algorithm

Open Set	Select State	Goal State	Terminate	Expanded Set
[1]	1	N	N	[4,2]
[4,2]	2	N	N	[4,6]
[4,6]	6	N	N	[4,3]
[4,3]	3	N	N	[4,5]
[4,5]	5	N	N	[4,~~6~~]
[4]	4	N	N	[7]
[7]	7	Y	Y	



Stack
Tie: Descending

Search Algorithm: Saving Explicit Space

- Initialize: Set OPEN={s}, CLOSED = {}
- Fail:
  - If OPEN={}, Terminate with failure
- Select: Select a state, n, from OPEN and
  - Save n is CLOSED
- Terminate:
  - If n∈G, terminate with success
- Expand:
  - Generate the successors of n using O
  - For each successor, m, insert m in OPEN,
    - Only if $m \notin [OPEN \cup CLOSED]$
- Loop:
  - Go to step 2

How to maintain data structure to trace the path from start to goal?

Open Issues

- What will happen if state transition operators have associated cost?

- What will happen if the goal state have associated cost?

Thank You