UCLA Samueli School of Engineering

CS161 WEEK2 DISCUSSION 1C

Danfeng Guo

Contributed by Yewen W' and Shirley C's previous course materials

Agenda

SEARCH PROBLEM FORMULATION

DFS & BFS

Search Problem

5 KEY ELEMENTS

Initial state

State space

Actions

Successor function F(cur_state, action) = next_state

Goal test

Search Problem

EXAMPLE: EIGHT QUEENS

(0→8)

Initial State: Empty board

State Space: All arrangements of 8 queens

Actions: Put a queen on the board

Successor function: The arrangement after putting the new queen

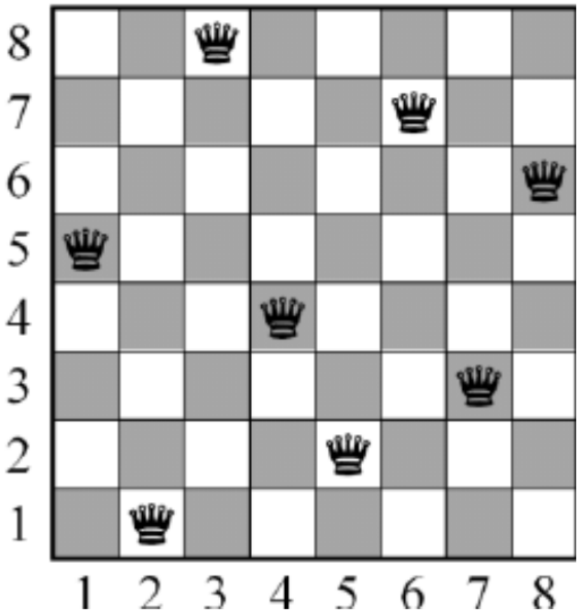
Goal test: whether current arrangement meets ~~the~~ conditions

Number of cases in total:

$64 \times 63 \cdots \times 57$

Action: Put a queen on a new row

$8 \times 7 \times 6 \times \cdots \quad 8!$

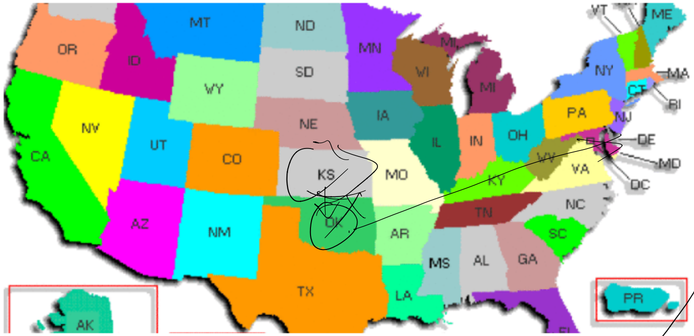


Search Problem

PRACTICE

Drive from CA to DC
Initial state: CA
Actions: Move to a neighboring state
Successor function: f(cur-state, drive) → new-state
Goal test: DC ?

visit = [ ]



cost / purpose = # state
= gasoline

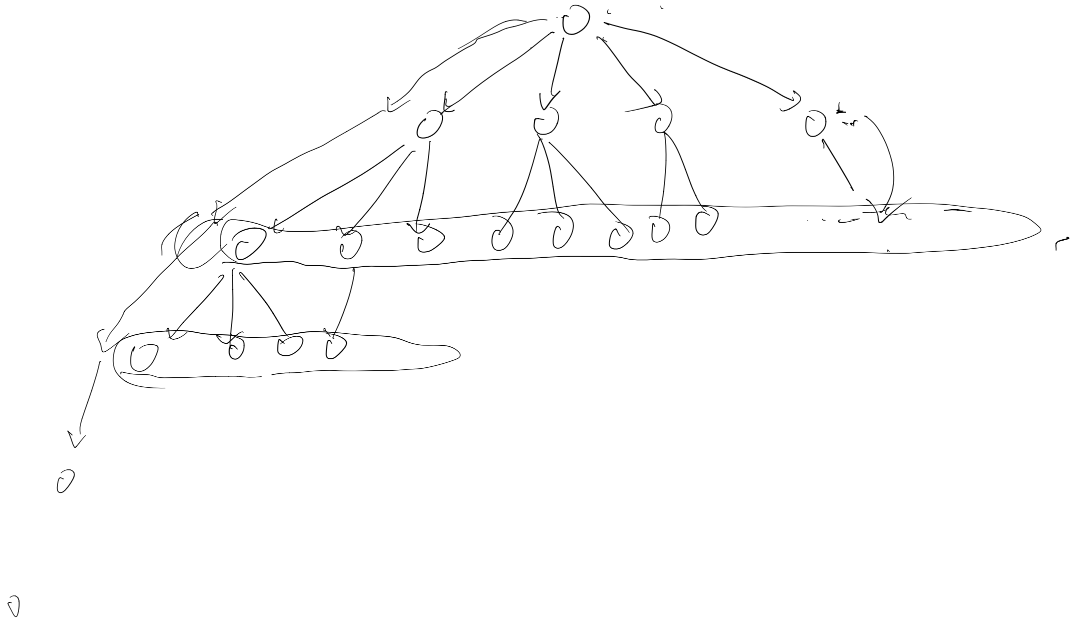
Search Problem

INFORMED VS UNINFORMED

Whether we have information on our goal state that helps our decision making

We know the dist from cur_state to DC.

DFS & BFS



DFS & BFS

4 EVALUATORS

REFRESH YOUR MEMORY

cost is uniform

Y if no infinite loops

	BFS	DFS	Limited DFS
Complete	Y	N	N
Optimal	Y	N	N
Time complexity	$O(b^d)$	$O(b^m)$	$O(b^l)$
Space complexity	$O(b^d)$	$O(bm)$	$O(bl)$

limited depth = l

branching factor : b

depth : d

max-depth : m

Practice

ROTTING ORANGES

You are given an m x n grid where each cell can have one of three values:

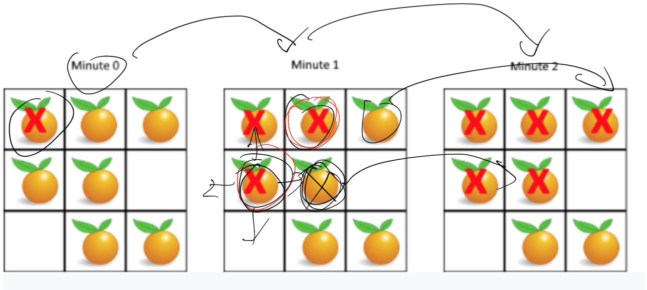
0 representing an empty cell,

1 representing a fresh orange, or

2 representing a rotten orange.

Every minute, any fresh orange that is **4-directionally adjacent** to a rotten orange becomes rotten.

Return *the minimum number of minutes that must elapse until no cell has a fresh orange*. If *this is impossible, return* -1.



Practice

- BFS or DFS?
- Divide the whole programs into parts

  What functions do we need?

① Get all rotten oranges (state)

② Know the neighbors, expand (action)

③ How many total oranges (test)

- Initial state
- Get the loc of rotten oranges
- Expand
- Check

Repeat until no further expansion

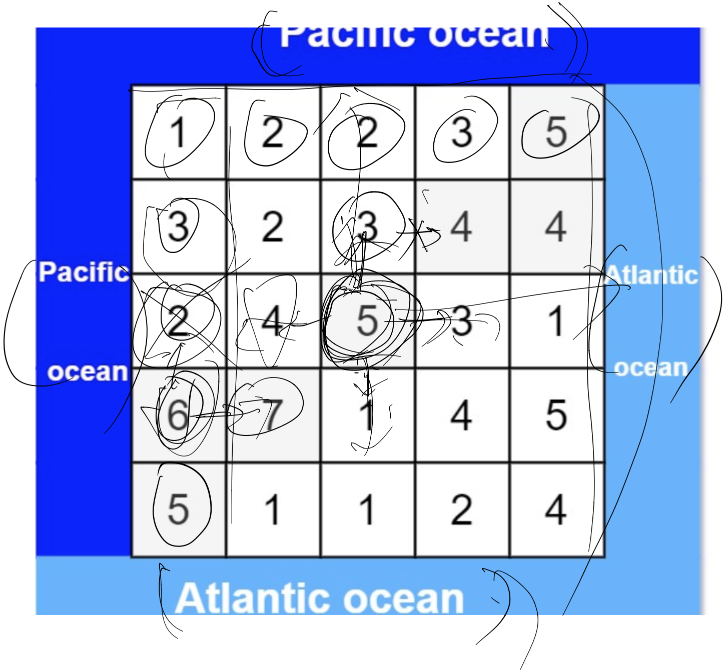
Practice

PACIFIC ATLANTIC WATER FLOW

You are given an m x n integer matrix heights representing the height of each unit cell in a continent. The **Pacific ocean** touches the continent's left and top edges, and the Atlantic ocean touches the continent's right and bottom edges.

Water can only flow in four directions: up, down, left, and right. Water flows from a cell to an adjacent one with an equal or lower height.

Return *a list of grid coordinates where water can flow to both the Pacific and Atlantic oceans*.



UCLA Samueli School of Engineering

Practice

PACIFIC ATLANTIC WATER FLOW

start at a loc x_1, y_1

explore its neighbors

if moveable, then (x_n, y_n)

~~test~~

① keep tracking of test

② keep tracking of visited pts

Start from a loc near Pacific

Search 4 directions

If it can be a source of water, update cur_loc

Do it again for Atlantic

Intersection

Homework1

- TREE-CONTAINS
  - Recursively check whether (car lst) = x
    - → t, return t
    - → nil, explore (cdr lst)
- TREE-MIN
  - Recursion again
  - If only has 1 element, return that element
  - Otherwise return the minimum of TREE-MIN((car lst)) and TREE-MIN((cdr lst))
  - But it is an ordered tree!
  - So we only need to look at the most left
  - (car (car (car ….))) until single atom

Homework1

- TREE-ORDER
  - Recursion.
  - If only 1 element, returns it
  - If 3 element, concatenate 3 parts.
  - [element2 (TREE-ORDER element1) (TREE-ORDER element3)]
  - Be clear of the argument format
- SUB-LIST
  - Recursion
  - if LEN=0: nil
  - if START=0, and LEN!=0: (cons (car lst) (SUB-LIST lst START (- LEN 1)))
  - Otherwise: (SUB-LIST (cdr lst) (- START 1) LEN)

Homework1

- SPLIT
  - Super easy if you know the length. use sub-list
- BTREE-HEIGHT
  - Recursion
  - If only one atom: return 0
  - Otherwise: return 1 + max(left height, right height)

Homework1

- LIST2BTREE
  - If only one element, return it
  - Otherwise, split it, and concatenate
    - [(LIST2BTREE left) (LIST2BTREE right)]

Q&A