Introduction to AI

SEARCH

Pedro Meseguer
Institut d'Investigació en Intel·ligència Artificial (IIIA)
Consejo Superior de Investigaciones Científicas (CSIC)

IIIA-CSIC INSTITUT D'INVESTIGACIÓ EN INTEL·LIGÈNCIA ARTIFICIAL

1

Contents

2

1.Introduction

Example: 8-puzzle

1	5	2
4		3
6	7	8

3

1. 8 puzzle

- It is a 3x3 board with 8 numbered tiles and a hole (= blank space).
- A tile adjacent to the blank space can slide into the blank space, which now appears where the tile was previously.
- The objective is to reach the goal state.

1	2	3
8		6
7	5	4

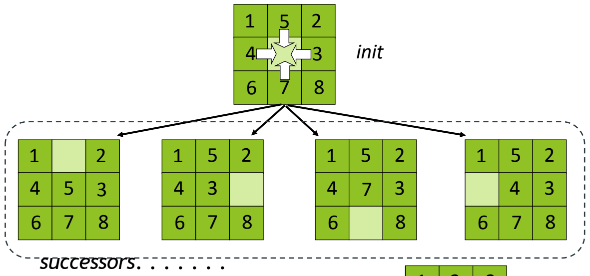
initial state

1	2	3
8		4
7	6	5

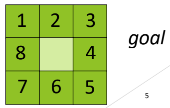
goal state

4

1.Example: Path finding in the 8 puzzle



Sequence of movements
(more than one)
from init to goal



5

1.Off-line search

Two separate phases:

1. Solution computation
2. Solution execution

Search for a complete solution	Execution

AI [search] is only concerned with the first phase

- Solution execution does not affect that solution: in general, this does not happen in the real world
- For problems in very controlled environments

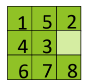
6

2.Concepts

- State: a possible problem configuration
- State space: all possible configurations (directed graph)
- Operators:
  - legal actions
  - they generate *successors* of a state
- States: initial and goal
  explicit ***may be implicit***
- Solution: path in the state space
  - sequence operators initial to goal
    (*sometimes*) goal state
- Problem instance: state-space plus initial and goal states

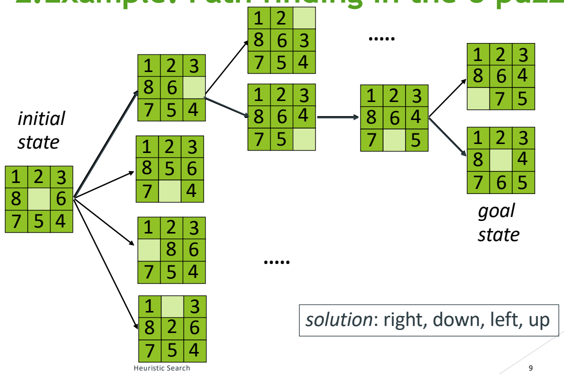
7

2.Example

- State: 
- State space: directed graph; 9! nodes; arcs are actions
- Operators:
  - legal actions are move the blank up, down, right, left
  - they generate *successors* of a state
- States: initial and goal
  explicit ***may be implicit*** *(test)*
- Solution:
  - sequence of operators from initial to goal
- Problem instance: well defined

8

2.Example: Path finding in the 8 puzzle



9

2.Off-line systematic search: tree search

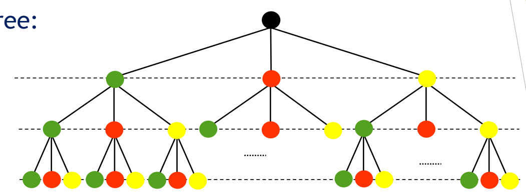
Systematic traversal of the state-space:
- It guarantees to find a solution, if one exists.
- Rubik's cube / Sliding puzzles: 8-puzzle, 15-puzzle, 24-puzzle

Exploring the state space of a problema as a tree:
- Root: initial state
- 1st level: the sucessors of root
- 2nd level: the sucessors of sucessors of the root
- ...
- dth level: the sucessors of nodes at level d-1

10

2.Off-line systematic search: blind search

Search tree:



Depth-first search (DFS): explore by **branches**

Breath-first search (BFS): explore by **levels**

11

2.Off-line systematic search: A* algorithm

A* algorithm:
- important in systematic search: best-first heuristic search
- known from long time (since 1968)

A* expand nodes (=generate successors) with mínimum f
- node lists: open (to be expanded) and closed (already expanded)
- where f(x) = g(x) + h(x)
  - g(x): the cost already spent to reach x from init
  - h(x): the expected cost to reach a solution from x *(heuristic)*
- expand the node with mínimum f in open list : gen. its successors

12

2.Off-line systematic search: heuristics

Heuristic:

- Estimates the distance to the closest goal
- Admissibility: never overestimates the real distance
- A* with an admissible heuristic: an optimal path to the goal
- Cheap to compute (impact in practice)

Examples of heuristics:

- 8-puzzle:
  - #tiles out of place
  - Sum Manhattan distance for each tile out of place

13

2.Off-line systematic search: combinatorial explosion

Size of state space grows quickly as problem size increases

- Sliding puzles:
  - 8-puzzle 9!
  - 15-puzzle 16!
  - 24-puzzle 25!
- Main reason of the failure of early AI

The state space of these problems is divided in two, unnconnected parts of the same size

Search (weak methods)

- are overhelmed by huge state spaces
- only work for *toy problems (memo causing AI winter 1973)*

14