State-Space Search: Introduction

CSE 415: Introduction to Artificial Intelligence
University of Washington
Spring, 2017

Outline

- Motivation: Problem solving
- Example 1: The Missionaries and Cannibals puzzle
- Example 2: Towers of Hanoi
- State, operator, state space
- Operators, preconditions, moves
- State space as a tree
- Example 3: The Travelling Salesman Problem
- State space as a graph
- Blind search methods: DFS, BFS.
- Example 4: Farmer, Fox, Chicken and Grain puzzle

Motivation

A useful form of intelligence is the ability to solve problems.
The standard AI approach to solving a problem is to formulate it as a search through a space of possible solutions or a space of partial solutions and then systematically search the space.

This was the idea behind the "General Problem Solver" program created by A. Newell, H. Simon in 1961.

The concepts underlying state-space search form a foundation on which many techniques and advanced methods are built.

"Looking Ahead"

A key idea in search is "looking ahead."

An agent should answer the question:

What will be the consequences of different sequences of actions or possible decisions?

Problem solving via search is sometimes called "planning."

Missionaries and Cannibals Puzzle

http://www.learn4good.com/games/puzzle/boat.htm

Missionaries and Cannibals Puzzle

MMM
CCC | R |
B

____
____

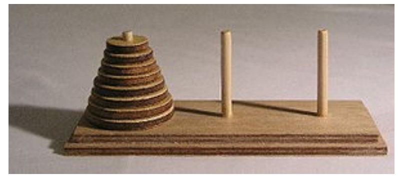
| R | MMM
CCC
B

Missionaries and Cannibals Puzzle

What sequence of legal moves takes us from the initial state to the goal state?

This sequence is a *solution*.

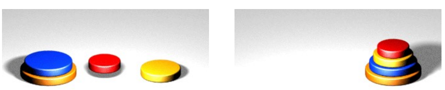
Towers of Hanoi



What sequence of legal moves takes us from the initial state to the goal state?

http://en.wikipedia.org/wiki/Tower_of_Hanoi

Towers of Hanoi



(a) An intermediate state, (b) goal state.

http://en.wikipedia.org/wiki/Tower_of_Hanoi

States

A *state* consists of a complete description or snapshot of a situation that can be arrived at during the solution of a problem.

Initial state: the starting position or arrangement, prior to any problem-solving actions being taken.

Goal state: the final arrangement of elements or pieces that satisfies the requirements for a solution to the problem.

State Space

The set of all possible states – the arrangements of the elements or components to be used in solving a problem – forms the space of states for the problem. This is known as the *state space*.

Moves

A *move* is a transition from one state to another.

An *operator* is a partial function (from states to states) that can (sometimes) be applied to a state to produce a new state, and also, implicitly, a move.

A sequence of moves that leads from the initial state to a goal state constitutes a *solution.*

Operator Preconditions

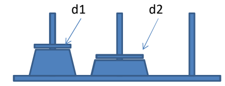
Precondition: A necessary property of a state in which a particular operator can be applied.

Example: In checkers, a piece may only move into a square that is vacant. Thus, Vacant(place) is a precondition on moving a piece into place.

Example: In Chess, a precondition for moving a rook from square A to square B is that all squares between A and B be vacant. (A and B must also be either in the same row or the same column.)

A conjunction of such preconditions that is sufficient to make the application of an operator legal can serve as "the" precondition for the operator.

Operator Representation



Example, in Towers of Hanoi puzzle:

Operator:
- Name: Move-1-2
- Purpose: Move a disk from Peg 1 to Peg 2
- Precondition (predicate):
  - There is a disk d1 on Peg 1, and
  - d1 is the topmost disk on Peg 1,
  - and either there is no disk on Peg 2 or
    - there is a disk d2 on Peg such that
    - d2 is the topmost disk on Peg 2,
    - and diameter of d2 is greater than diameter of d1.
- State transformation (function):
  - Remove disk d1 from Peg 1 and
  - put it on top of disk d2 on Peg 2.

Search Trees

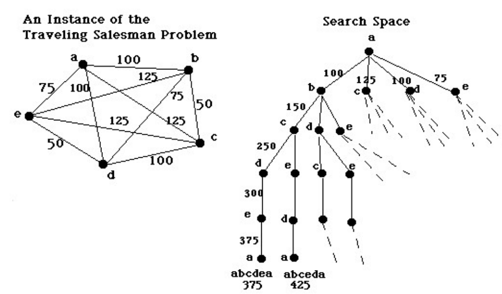
By applying operators from a given state we generate its children or *successors*.

Successors are *descendants* as are successors of descendants.

If we ignore possible equivalent states among descendants, we get a tree structure.

Depth-First Search: Examine the nodes of the tree by fully exploring the descendants of a node before trying any siblings of a node.

Example: Travelling Salesman Problem



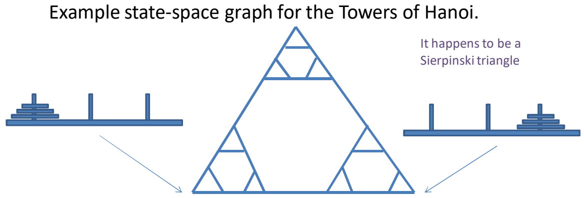
http://www.cs.trincoll.edu/~ram/cpsc352/notes/search.html

Search Graphs

There may be multiple paths (move sequences) from the initial state to any given state.

There may be cycles.

Example state-space graph for the Towers of Hanoi.



Depth-First Search: Iterative Formulation

1. Put the start state on a list OPEN
2. If OPEN is empty, output "DONE" and stop.
3. Select the first state on OPEN and call it S.
   Delete S from OPEN.
   Put S on CLOSED.
   If S is a goal state, output its description
4. Generate the list L of successors of S and delete from L those states already appearing on CLOSED.
5. Delete any members of OPEN that occur on L.
   Insert all members of L at the *front* of OPEN.
6. Go to Step 2.

Ss State-space Search

Breadth-First Search: Iterative Formulation

1. Put the start state on a list OPEN
2. If OPEN is empty, output "DONE" and stop.
3. Select the first state on OPEN and call it S.
   Delete S from OPEN.
   Put S on CLOSED.
   If S is a goal state, output its description
4. Generate the list L of successors of S and delete
   from L those states already appearing on CLOSED.
5. Delete any members of OPEN that occur on L.
   Insert all members of L at the *end* of OPEN.
6. Go to Step 2.

Ss State-space Search

Farmer, Fox, Chicken & Grain

A farmer has to get his fox, chicken, and grain across the river.
The boat can hold only the farmer and one item.
The fox cannot be left alone with the chicken.
The chicken cannot be left alone with the grain.
How does the farmer get everything across?

States?
Operators?
Preconditions ?