Artificial Intelligence: a concise introduction

Vicenç Torra

School of Informatics
University of Skövde
SE-54128 Skövde, Sweden
e-mail: vtorra@his.se

Chapter 2. Search

A large number of problems can be formulated in a similar way so that they can be solved using the same type of algorithms. These algorithms are called *search algorithms*. In this chapter we show how to formulate different types of problems so that search algorithms can be applied. To do so, we first give an introduction to problem solving, that mainly consists in explaining how problems are expressed, and what means to find a solution to the problem. The concept of state space and a search within this state space is fundamental for this purpose. Then, we discuss how to build a solution and different ways to do so, that mainly correspond to different ways of searching in the state space.

1 Introduction to problem solving and search

1.1 State space and problem representation

1.1.1 Examples

We will use the following examples.

- Linear puzzle. This problem consists on a sequence of numbers built from the set $\{1, 2, 3, 4\}$ as e.g. $[1, 2, 3, 4]$. We can swap the first two numbers, the two numbers in the middle, or the last two numbers. Then, given a sequence we want to get another sequence. E.g., from the sequence $[4, 2, 3, 1]$ we want to reach $[1, 2, 3, 4]$.
- Shortest path. Given a map, this problem consists on finding the shortest path between two towns in the map. The map includes roads between towns and the distance between them.
- Symbolic integration of expressions. Given a numerical expression with an integral term, the goal is to obtain the equivalent expression without

an integral symbol. E.g., given

$$\int x^2 dx$$

we want to be able to find

$$\frac{1}{3}x^3.$$

Then, for example, which is the solution of

$$\int \left(x^2 e^x + x^3\right) dx.$$

To achieve this goal we have the integration rules.

- Theorem proving. Given an expression proof its validity by means of a set of axioms. For example, is it true

$$a \times (b + c) = a \times b + a \times c?$$

1.1.2 Modeling the environment of a system

- State. The environment is represented in terms of states. All possible environments need to be represented by a state.

In the example of the linear puzzle, an state is any of the possible sequences that can be built from the 4 numbers. There are $4! = 24$ possible sequences. Therefore, there are 24 possible states.

In the example of the shortest path, an state corresponds to a town. Then, we have as many states as towns in the map.

In the example of symbolic integration, an state is a mathematical expression with or without an integral symbol. To solve the problem in practice we may constraint which are the possible expressions to integrate. E.g., only polynomials? expressions including forms like e^{ax^b}? In general, we may have a large set of possible states, or even an infinite number of states.

In the example corresponding to theorem proving, an state is the list of logical formulas that we know that are true at a given moment.

1.1.3 Modeling the actions of a system

- Actions. They are modeled as transitions between states. So, formally, they are *A: State* $\rightarrow$ *State*.
- State space graph. the set of all states and the actions that act on these states define a graph.

Related concepts.

- Branching factor. The number of actions that can be applied to the states of a problem. In the linear puzzle the branching factor is three (for all states).

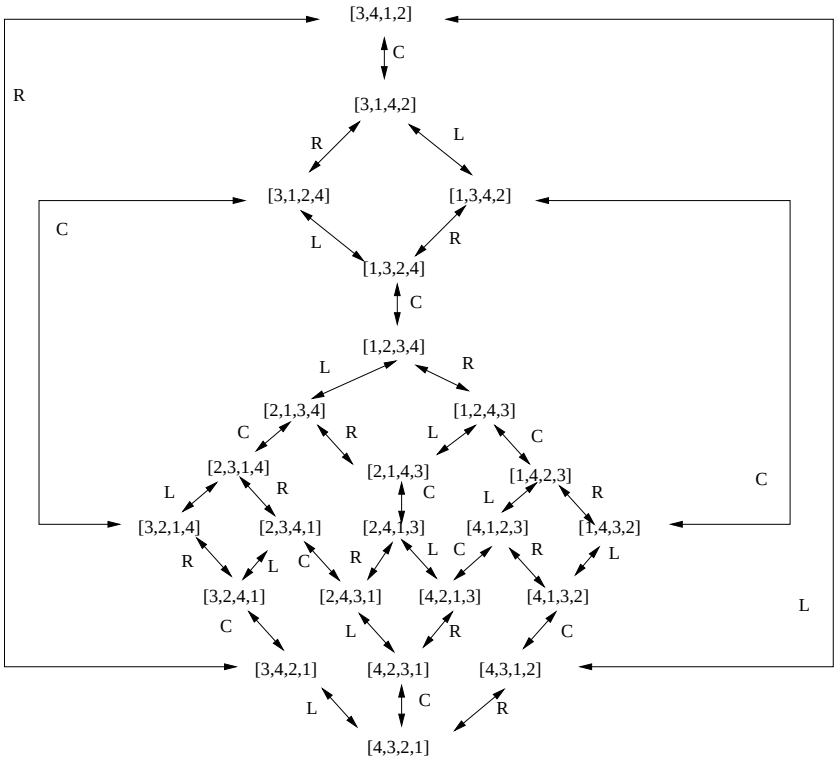

Figure 1: States space graph: linear puzzle.

1.1.4 Defining the problem

- Initial state. This corresponds to define how is the environment before applying any action.
- Objective function. We need to know which is the goal of the system. In some cases, there is a single state that is our goal. This is the case of the linear puzzle if we want to obtain the sequence $[1, 2, 3, 4]$, and the problem of the shortest path where the goal is to reach a town. In other cases, there are several states that may be suitable. The example of the theorem prover corresponds to this case. If we want to prove a formula, any state where the formula appears as true is a goal state. Or maybe we do not know which state is our goal but we may be able to recognize when a state satisfies our requirements. The example of the symbolic integration fits to this latter example. The goal is to find an expression that has no integral symbol, but we do not know which expression are we going to obtain (if we knew, the system is not needed!!). To model the goal we use a Boolean objective function which given a state returns true if the state satisfies our

requirements, and false otherwise.

1.2 Some classes of problems

- Constraint satisfaction problems. A set of variables and possible values for each value. The goal is to have a value assigned to each variable. There are some constraints on the assignments. An state is to have a set of variables already assigned and others not.
- Planning. The goal is to find a sequence of actions (a plan) to achieve a goal.

2 Building a solution

Implementation. We use a tree structure, to avoid repetitions and sharing paths.

- Root. Initial state
- Nodes. States
- Edges. Actions
- Leaves. Terminal states from paths

Additional concepts.

- Fringe. The nodes in the tree not yet expanded. Also known as open nodes.
- Closed nodes. The nodes in the tree that we have already expanded.

Search algorithm.

function search (problem) **return** solution

1. fringe := insert (make-node (initial-state (problem)))
2. ∃solution := false
3. **while** not ∃solution and not(empty(fringe)) **loop**
   (a) node := select node and delete (fringe)
   (b) **if** solution (node) **then** ∃solution:= true
   (c) **else** newNodes := expand (node, problem)
   (d) fringe := insert (fringe, newnodes, strategy)
   (e) **end if**
4. **end loop**
5. **if** ∃solution **then** return(node)
6. **else** return (no-solution)
7. **end if**

end function