Blocks world problem

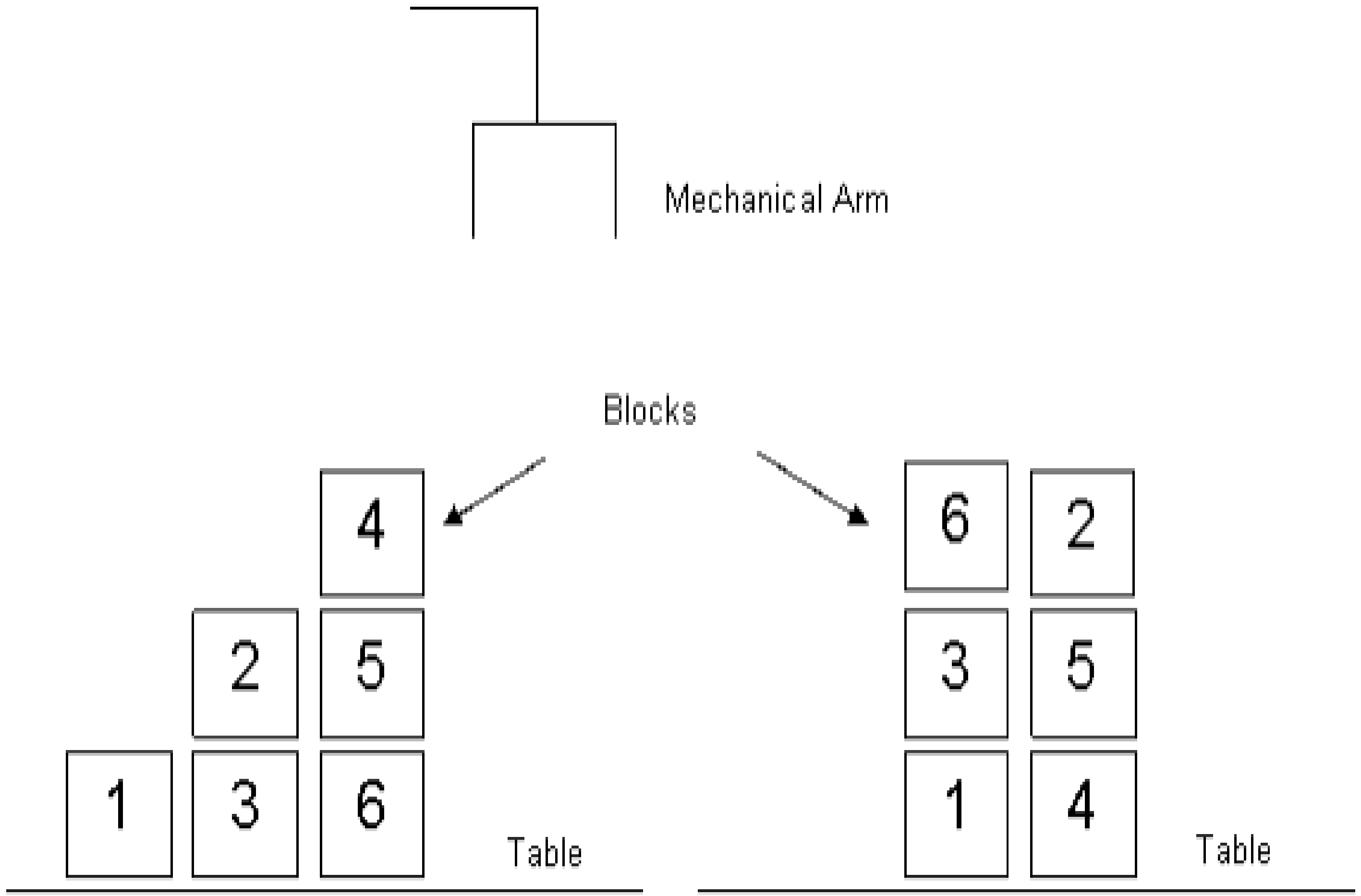
Blocks world problem

Blocks world (or, the world of blocks) is a model domain used in artificial intelligence to explore different approaches to automated reasoning. This model is used to illustrate that a given algorithm can perform planning, or that it is efficient in terms of the number of calculations required to find a solution or in terms of the length of that solution.

Description

There is a table on which some uniform blocks (cubes) are placed. Some blocks may or may not be stacked on other blocks. We have a robot arm to pick up or put down the blocks.

The robot arm can move only one block at a time, and no other block should be stacked on top of the block which is to be moved by the robot arm. The problem is to change the configuration of the blocks from a given initial state to a given goal state



Start state Goal state

States:

Configurations of the blocks satisfying the conditions. The configurations can be specified using the following predicates where A and B denote arbitrary blocks:

> ON(A,B) : block A is on block B.
> ONTABLE(A) : block A is on the table.
> CLEAR(A) : block A has nothing on it.
> HOLDING(A) : the arm holds block A.
> ARMEMPTY : the arm holds nothing.

Initial state: The initial configuration of blocks.

Actions:

UNSTACK(A,B) : pick up clear block A from block B and hold it in the arm

STACK(A,B) : place block A held in the arm onto clear block B

PICKUP(A) : lift clear block A with the empty arm

PUTDOWN(A) : place block A held in the arm onto a free space on the table.

Goal state: A certain configuration of the blocks.

Path cost: Number of actions taken to reach the goal state.

The blocks world problem has always a trivial solution: All blocks not already correctlypositioned for the goal state be set off onto the table (one at a time with the mechanical arm),and then reassembled in the proper order on top of any blocks already correctly positioned.

A solution consists of the following 12 moves in that order:

1. UNSTACK(2, 3)
2. PUTDOWN(2)
3. UNSTACK(4, 5)
4. PUTDOWN(4)
5. UNSTACK(5, 6)
6. STACK(5, 4)
7. PICKUP(2)
8. STACK(2, 5)
9. PICKUP(3)
10. STACK(3, 1)
11. PICKUP(6)
12. STACK(6, 3)

Cryptarithmetic puzzle

This is an example for a special type of problem known as "constraint satisfaction problem". Constraint satisfaction problems are extensively studied and are the subject of intense research in artificial intelligence.

Description

- A cryptarithmetic puzzle is a mathematical exercise where the digits of some numbers are represented by letters.
- Each letter represents a unique digit.
- It will be assumed that the leading digits of numbers are not zero.
- The problem is to find the digits such that a given mathematical equation is satisfied.

Illustrative example

- Find the digits to be assigned to the letters such that the following equation is true (the digits assigned to T and F should not be 0):

```
  T W O +
  T W O
-------
F O U R
-------
```

Solution

With some trial and error we can get the following solution:

F = 1, O = 4, R = 8, T = 7, U = 6, W = 3.

That this is indeed a solution can be verified easily as follows

```
  7 3 4 +
  7 3 4
-------
1 4 6 8
-------
```

Here is another solution.

```
    8 2 6 +
    8 2 6
-------
1 6 5 2
-------
```

There may still be other solutions.

Problem formulation

Using the idea of states the problem can be formulated as follows:

- States: Let there be n different letters where n ≤ 10. A problem state is an ordered n-tuple of digits (d1, d2, . . . , dn) representing the digits to be assigned to the integers.
- Initial state: The initial state can be considered as the ordered n-tuple all of whose elements are 0's.

• Actions: Increase the value assigned to a letter by 1. • Transition model: Given a state and action, this returns the resulting state.

• Goal test: We have reached state (d1, d2, . . . , dn) which satisfies the constraints of

the problem.

• Path cost: Number of actions.

Solution

- There are algorithms for solving the cryptarithmetic puzzle. Implementations of these algorithms by hand computations is difficult.
- The method of trial and error keeping in mind the constraints to be satisfied may be applied to obtain solutions.