HW#3 Solutions-572

3.9 The **missionaries and cannibals** problem is usually stated as follows. Three missionaries and three cannibals are on one side of a river, along with a boat that can hold one or two people. Find a way to get everyone to the other side without ever leaving a group of missionaries in one place outnumbered by the cannibals in that place. This problem is famous in AI because it was the subject of the first paper that approached problem formulation from an analytical viewpoint (Amarel, 1968).

a. Formulate the problem precisely, making only those distinctions necessary to ensure a valid solution. Draw a diagram of the complete state space.

b. Implement and solve the problem optimally using an appropriate search algorithm. Is it a good idea to check for repeated states?

c. Why do you think people have a hard time solving this puzzle, given that the state space is so simple?

a :

Let m_R denotes the number of missionaries at the right side of the river, m_L the number of missionaries at the left side, c_R the number of cannibals at the right side and c_L the number of cannibals at the left side, $B = \{L, R\}$ denoting the boat is on the left or the right side, then we can denote any state of the world as $(m_L - c_L | m_R - c_R) : B$, where $0 \leq m_L, m_R, c_L, c_R \leq 3$. In addition, we have the following constraints: $m_R + m_L = 3$; $c_R + c_L = 3$; $(m_L \geq 1) \Rightarrow (m_L \geq c_L)$; $(m_R \geq 1) \Rightarrow (m_R \geq c_R)$. Assume that we start at the left side of the river, and that at least one person has to use the boat for moving from a side to the other.

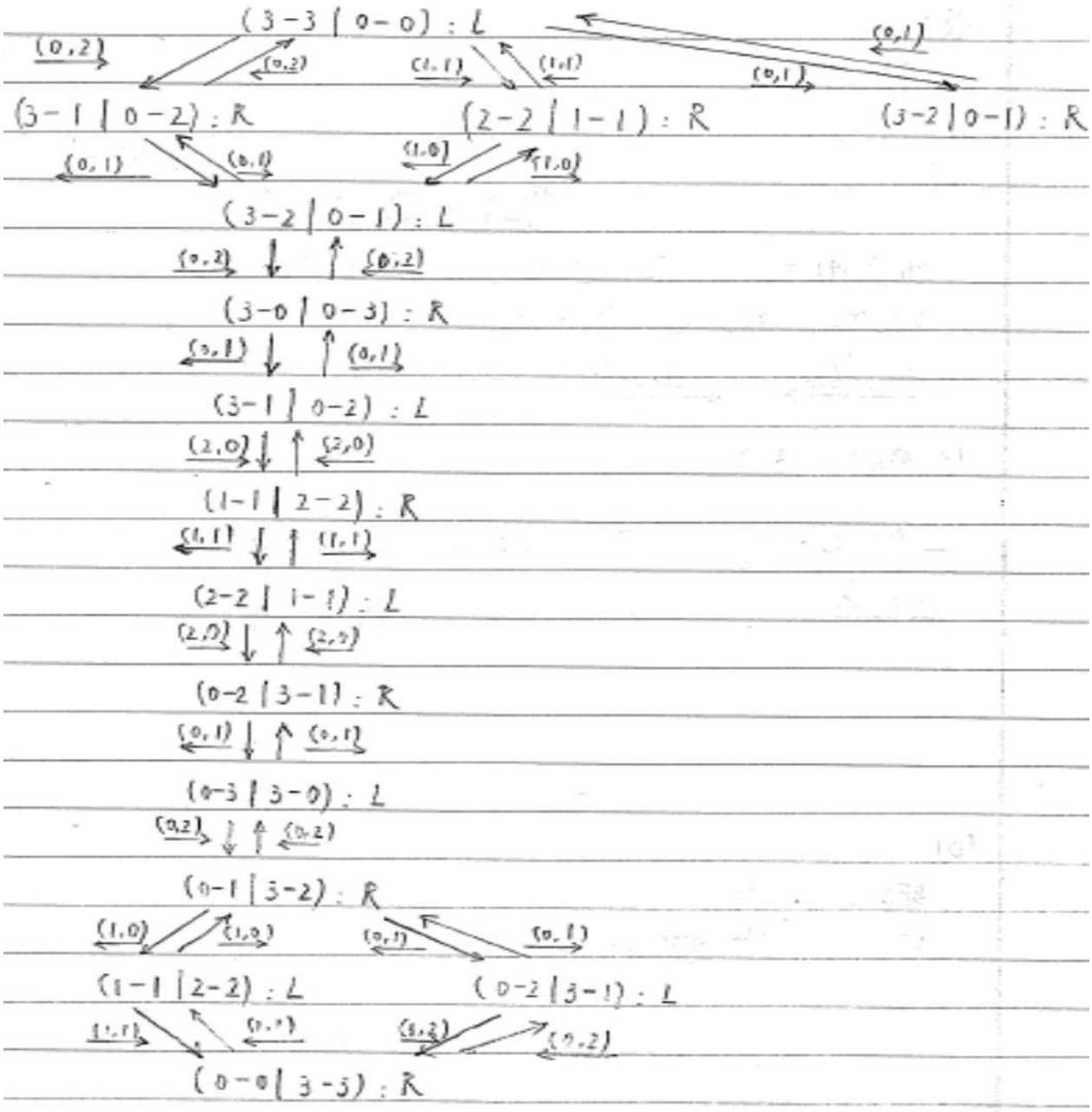


Figure 2: Diagram of the complete state space

b :

The search space is small, so any optimal algorithm works.

Using DFS, the solution (according to Figure 2) is as follows: $(3-3|0-0) : L \rightarrow (3-1|0-2) : R \rightarrow (3-2|0-1) : L \rightarrow (3-0|0-3) : R \rightarrow (3-1|0-2) : L \rightarrow (1-1|2-2) : R \rightarrow (2-2|1-1) : L \rightarrow (0-2|3-1) : R \rightarrow (0-3|3-0) : L \rightarrow (0-1|3-2) : R \rightarrow (1-1|2-2) : L \rightarrow (0-0|3-3) : R$. Yes, it is a good idea to check for the repeated states because we can avoid infinite loops and make the search space small.

C :

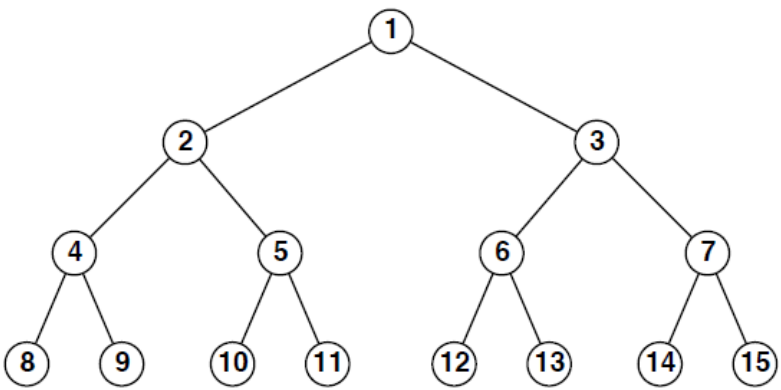
It is not obvious that almost all moves are either illegal or revert to the previous state. There is a feeling of a large branching factor, and no clear way to proceed.

3.15 Consider a state space where the start state is number 1 and each state k has two successors: numbers $2k$ and $2k + 1$.

a. Draw the portion of the state space for states 1 to 15.

b. Suppose the goal state is 11. List the order in which nodes will be visited for breadthfirst search, depth-limited search with limit 3, and iterative deepening search.

Solution: **a**



b. Breadth-first: 1 2 3 4 5 6 7 8 9 10 11
Depth-limited: 1 2 4 8 9 5 10 11
Iterative deepening: 1; 1 2 3; 1 2 4 5 3 6 7; 1 2 4 8 9 5 10 11