Solving Problems by Searching

2. Missionaries(nha tu Sy) and Cannibals(con Quy) Problem

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.

Imagine the scene in real life: three members of the Arawaskan tribe, Alice, Bob, and Charles, stand at the edge of the crocodile-infested Amazon river with their new-found friends, Xavier, Yolanda, and Zelda. All around them birds cry, a rain storm beats down, Tarzan yodels, and so on. The missionaries Xavier, Yolanda, and Zelda are a little worried about what might happen if one of them were caught alone with two or three of the others, and Alice, Bob, and Charles are concerned that they might be in for a long sermon that they might find equally unpleasant. Both parties are not quite sure if the small boat they find tied up by the side of the river is up to making the crossing with two aboard.

To formalize the problem, the first step is to forget about the rain, the crocodiles, and all the other details that have no bearing in the solution. The next step is to decide what the right operator set is. We know that the operators will involve taking one or two people across the river in the boat, but we have to decide if we need a state to represent the time when they are in the boat, or just when they get to the other side. Because the boat holds only two people, no "outnumbering" can occur in it; hence, only the endpoints of the crossing are important. Next, we need to abstract over the individuals. Surely, each of the six is a unique human being, but for the purposes of the solution, when it comes time for a cannibal to get into the boat, it does not matter if it is Alice, Bob, or Charles. Any permutation of the three missionaries or the three cannibals leads to the same outcome. These considerations lead to the following formal definition of the problem:

States: a state consists of an ordered sequence of three numbers representing the number of missionaries, cannibals, and boats on the bank of the river from which they started. Thus, the start state is (3,3,1).

Operators: from each state the possible operators are to take either one missionary, one cannibal, two missionaries, two cannibals, or one of each across in the boat. Thus, there ! are at most five operators, although most states have fewer because it is necessary to avoid j illegal states. Note that if we had chosen to distinguish between individual people then j there would be 27 operators instead of just 5.

Goal test: reached state (0,0,0).

Path cost: number of crossings.

This state space is small enough to make it a trivial problem for a computer to solve. People have a hard time, however, because some of the necessary moves appear retrograde. Presumably, humans use some notion of "progress" to guide their search. We will see how such notions are used in the next chapter.

Cannibals and Missionaries is one of the basic AI problems, wherein you have to get all the missionaries and cannibals on the opposite side of the river. There are two constraints, however:
1. A boat cannot move without anybody present on it, i.e. at-least one of the missionaries or the cannibal must be present on the boat to mo3ve. The boat can be occupied by a maximum of two person at a time.
2. If at any point of time, the number of cannibals exceed the number of missionaries on a side of the river, the cannibals will eat missionaries and the puzzle will not be solved.

The the aim of this post is to analyze this problem and then present a code which solves it.

To solve a problem in AI, we need to describe it with few parameters. For this problem, we need to define the following parameters

1. Initial State: <3 3> . The first parameter is the number of missionaries, the second parameter is the number of cannibal.

2. Final State: <0 0>.

3. Actions: This parameter defines, what all actions we can take during any state of the problem. There are 5 different actions that are possible here.

- <1 0> moving 1 missionary and 0 cannibal from one side to another
- <2 0> moving 2 missionary and 0 cannibal from one side to another
- <1 1> moving 1 missionary and 1 cannibal from one side to another
- <0 1> moving 0 missionary and 1 cannibal from one side to another
- <0 2> moving 0 missionary and 2 cannibal from one side to another

Thus, using these actions, we can reach to the goal. Another important point to mention here is that - alternatively we reduce the number and increase the number on the source side of the river. This is because, the boat cannot move without anyone occupying it. This also tells that the final position of the boat will be on the destination side of the river.

Now, since we have initial state <3 3> and we know all actions that we can perform, we can make a graph and traverse the graph. To create a graph, we have to apply all these five actions on the current state. For e.g. the initial state will lead to the following five new states on the source side <2 3>, <1 3>, <2 2>, <3, 2>, <3 1>. These are the states that are achieved by subtraction of the actions from the initial state. The next level of child will be found by addition of the action (because, the boat is on destination side, and is coming back to the source side, and since it cannot move empty, it will bring with itself some missionaries or cannibals, which needs to be added).

We have to keep a check on the invalid states, i.e. no side (source or destination) can have more than 3 missionaries or 3 cannibals. Similarly, they cannot have less than 0 missionaries or 0 cannibals. Also, at any point of time, the number of missionaries must be greater than or equal to the number of cannibals. This condition is *not* true when number of missionaries is 0 (because, cannibals cannot eat missionaries, when there are none).

The path is found by implementing breadth-first search. We explore each level completely before moving to the next level. If at any level, we encounter goal state, we say that we have found the solution.

The code below has two classes. The class Node is used to maintain the state and the other class MissCanibal implements the graph search algorithm.

Implement the missionaries and cannibals problem and use breadth-first search to find the shortest solution. Is it a good idea to check for repeated states? Draw a diagram of the complete state space to help you decide.

2. **Bài toán mã đi tuần: (đi theo hướng hình chữ L)**
Cho bàn cờ kích thước nxn. Con mã xuất phát từ một ô (x0, y0). Viết chương trình tìm đường đi của con mã qua tất cả các ô, mỗi ô đúng 1 lần rồi trở về ô xuất phát.

/*

a. Vua: được di chuyển theo đường chéo, hàng ngang hoặc cột dọc nhưng mỗi lượt đi chỉ đi được 1 ô.

b. Hậu: được di chuyển nhiều ô trên bàn cờ theo đường chéo hàng ngang hoặc cột dọc.

c. Xe: có thể di chuyển nhiều ô trên bàn cờ chỉ cần cùng cột dọc hoặc hàng ngang.

d. Tượng: có thể di chuyển nhiều ô trên cùng đường chéo mà nó đứng.

e. Mã: từ vị trí hiện tại, mã có thể đi đến 1 trong những ô gần nhất nhưng không nằm trên cùng hàng ngang cột dọc, hay đường chéo với ô nó đang đứng.

f. Tốt: mỗi bên có 8 quân tốt. Ở nước đi đầu tiên tốt có thể tiến 1 hoặc 2 nước trên cùng cột dọc. Từ lần di chuyển thứ 2, con tốt đó chỉ thể tiến 1 ô. Tốt ăn theo đường chéo trong phạm vi 1 ô hướng về phía trước. Sức mạnh của quân tốt được phát huy cao nhất khi tốt được phong cấp thành 1 trong các quân hậu, xe, tượng, mã. Trong cờ vua, trước khi phong cấp, tốt không thể lùi.

Lưu ý: các quân cờ phải được di chuyển theo quy luật trên và không được di chuyển bằng cách nhảy qua các quân cờ khác trừ quân mã.

*/

3. Cài đặt bài toán 8 hậu các dùng thuật toán leo đồi. ()

4. Viết chương trình giải bài toán tám số bằng thuật giải A*.

5. Viết chương trình giải bài toán tháp Hà Nội bằng thuật giải A*.

6. Cài đặt bài toán 8 hậu các dùng thuật toán gen.