Problem formulation exercises

(Source: https://aimacode.github.io/aima-exercises/, accessed in Oct 2022)

Exercise 01

Give a complete **problem formulation** for each of the following problems. It should be precise enough to be implemented.

1. There are six glass boxes in a row, each with a lock. Each of the first five boxes holds a key unlocking the next box in line; the last box holds a banana. You have the key to the first box, and you want the banana.
2. You start with the sequence ABABAECCEC, or in general any sequence made from A, B, C, and E. You can transform this sequence using the following equalities: AC = E, AB = BC, BB = E, and Ex = x for any x. For example, ABBC can be transformed into AEC, and then AC, and then E. Your goal is to produce the sequence E.
3. There is an $n \times n$ grid of squares, each square initially being either unpainted floor or a bottomless pit. You start standing on an unpainted floor square and can either paint the square under you or move onto an adjacent unpainted floor square. You want the whole floor painted.
4. A container ship is in port, loaded high with containers. There 13 rows of containers, each 13 containers wide and 5 containers tall. You control a crane that can move to any location above the ship, pick up the container under it, and move it onto the dock. You want the ship unloaded.

Exercise 02

Your goal is to navigate a robot out of a maze. The robot starts in the center of the maze facing north. You can turn the robot to face north, east, south, or west. You can direct the robot to move forward a certain distance, although it will stop before hitting a wall.

1. Formulate this problem. How large is the state space?
2. In navigating a maze, the only place we need to turn is at the intersection of two or more corridors. Reformulate this problem using this observation. How large is the state space now?
3. From each point in the maze, we can move in any of the four directions until we reach a turning point, and this is the only action we need to do. Reformulate the problem using these actions. Do we need to keep track of the robot's orientation now?

4. In our initial description of the problem we already abstracted from the real world, restricting actions and removing details. List three such simplifications we made.

Exercise 03

You have a ***9×9*** grid of squares, each of which can be colored red or blue. The grid is initially colored all blue, but you can change the color of any square any number of times. Imagining the grid divided into nine ***3×3*** sub-squares, you want each sub-square to be all one color but neighboring sub-squares to be different colors.

1. Formulate this problem in the straightforward way. Compute the size of the state space.
2. You need color a square only once. Reformulate, and compute the size of the state space. Would breadth-first graph search perform faster on this problem than on the one in (a)? How about iterative deepening tree search?
3. Given the goal, we need consider only colorings where each sub-square is uniformly colored. Reformulate the problem and compute the size of the state space.
4. How many solutions does this problem have?
5. Parts (b) and (c) successively abstracted the original problem (a). Can you give a translation from solutions in problem (c) into solutions in problem (b), and from solutions in problem (b) into solutions for problem (a)?

Exercise 04

Suppose two friends live in different cities and want to meet each other. On every turn, we can simultaneously move each friend to a neighboring city. The amount of time needed to move from city i to neighbor j is equal to the road distance d(i,j) between the cities. On each turn the friend that arrives first must wait until the other one arrives (and calls the first on his/her cell phone) before the next turn can begin. We want the two friends to meet as quickly as possible.

1. Write a detailed formulation for this search problem (hint: define some formal notation).
2. Let D(i,j) be the straight-line distance between cities i and j. Which of the following heuristic functions are admissible? (i) D(i,j); (ii) 2·D(i,j); (iii) D(i,j)/2.
3. Are there completely connected maps for which no solution exists?
4. Are there maps in which all solutions require one friend to visit the same city twice?