Artificial Intelligence

C1 (Week 3)

Each task is worth 1 point. Tasks with an asterisk do not count toward the maximum.

1. We consider simplified logic pictures on an $n \times n$ square.
   (a) Show an example of a picture that has exactly 2 solutions.
   (b) Show an example of a picture that has more than n solutions.

   In both cases, justify your answer.

2. Present the method used by people to solve logic pictures. You can use sites that allow you to solve such tasks on your own (for example, https://www.puzzle-nonograms.com/), or online tutorials for such tasks.

3. (2p, *) Task two of the P1 (as seen in the example of Pan Tadeusz) quite often reconstructs the original sentences. Check whether it is a reasonable reconstruction algorithm by comparing its effectiveness for Pan Tadeusz (percentage of fully correctly reconstructed verses) with the effectiveness of the random algorithm [1]. When presenting this task at the blackboard, you should present the results of the experiment, answer how you implemented the random algorithm, and at the request of the instructor show the code. The text of Pan Tadeusz can be found on the Free Readings website, you may have to adapt it to the task (e.g. by removing punctuation, changing the size of characters).

4. The poker task from P1 was conceived as one in which we perform random games and thus estimate the probability. However, it is possible to calculate this probability accurately (assisted by a computer). Tell us how? Tip 1: How many different hands of the Blotter are there, how many different hands of the Figurant? Tip 2: The product of the numbers in Tip 1 is large, but is this a problem for us?

5. (2p, *) Implement the solution from the previous task, tell what result you get.

6. Present a graph for a search problem in which there is a path from the starting point to the destination, and the Uniform Cost Search algorithm is unable to find it. Note: the graph does not have to correspond to a real-world task.

7. Consider the movement of a player in a maze filled with enemies, each of which moves cyclically along a prescribed route. The movement is discrete, that is, in each unit of time both the player and the enemies move by 1 field in one of the four basic directions. The player's goal is to reach the indicated field (treasure), and upon contact with an enemy, the game ends (and the player loses). Your goal is to design the maze (walls and enemy routes) in such a way that:
   (a) the enemies were quite numerous and moved along diversified routes,
   (b) the size of the state-space made it possible to perform a cross-search on an ordinary computer.

   How to achieve this? What will the state space and movements in it look like?

8. Another relatively difficult puzzle (for computers) is Sokoban (illustrate its rules, if necessary by finding relevant material on the Internet). It can be modeled in such a way that the movement is 1 step of the warehouseman (pushing or not the crate). Describe precisely[2] , how to create another state space in which the movements in the puzzle are "higher level" movements, and thus the solution is achievable in fewer steps.

[1] That is, one that draws a word division, satisfying two conditions: 1) after removing the spaces, we get a original string 2) all words are from the dictionary

[2] Of course, you don't have to implement the solution. But you should present your ideas in enough detail that a typical programmer would be able to implement them.

9. In this task we assume that we have a graph describing the road connections (including cost) between cities. Describe in detail the state space and the model for the following situations in which we consider moving a car through this network:
   (a) The cost is the amount of fuel needed to move from place to place, the tank has a certain capacity, and gas stations sq only in some nodes. We always refuel to a full tank, sq polq costs by whole numbers,
   (b) courier picks up, we have K packages to distribute to different locations.
10. We have a connected directed graph (we interpret nodes as places, and edges as possibilities to move from one place to another in one step). The graph is traversed by K friends (move synchronously, jumping from one nodes to another at the same time polqed). There can be any number of people in one node. The success is organized waging the meeting, i.e., presenting such a sequence of movements that all participants are in one place at one time. We consider two options:
    (a) In each turn, each participant must make a transition in the graph,
    (b) The participant can "fold", that is, decide not to change the position.

    Of course, both variants of this task can be modeled as searching the state space (although the state space for a graph with n nodes will be large, say how exactly). For selected[3] variant, suggest a more effective way to find the sequence leading to the meeting (or conclude that the meeting is impossible).
11. Lecture 2 ended with the presentation of a table with information about the basic search algorithms (without knowledge of a problem). Review this table and answer the following questions:
    (a) What is C* (this designation is not described, but can be deduced from the table)
    (b) Why there is such and not a different equation for UCS time limitation?
    (c) Why is DLS not optimal, but *iterative deepening is?*
    (d) Is it possible to relax the occurrence of the completeness condition in the table for a two-way search? How or why not?
    (e) Can the optimality condition in the table for a two-round search be relaxed? How or why not?

[3] Note for Students with an Olympic-algorithmic streak: there is an answer in both variants, as pointed out by Mr. BK in one of the previous editions of the subject, apparently even an Official Olympic Task came out of it.