**University of Kent**

**May/June 2020**

# **Examination Answer Document**

|  |  |
| --- | --- |
| Exam Paper Release Date and Time | * 12.05.2020 * 14.00 British Summer Time (BST) |
| Exam Paper Submission Date and Time | * 13.05.2020 * 17.00 British Summer Time (BST) |

Before starting this examination, you **MUST enter your Examination Number** above **(double click)**; failure to provide this means your mark cannot be recorded and **you may be awarded a 0 (zero)** for this examination.

|  |  |
| --- | --- |
| Question Number: | 1 |

1A)

N = 5  
Pop size = 3

Initial pop:

{1, 2, 3, 4, 5}

{1, 3, 2, 5, 4}

{1, 5, 4, 3, 2}

1B)

The fitness function: The lowest total number of miles travelled after visiting all addresses and returning to the HQ.

1C)

Crossover would not be appropriate as there is no valid way to crossover the addresses without creating invalid sequences, e.g. duplicated addresses, missed addresses, etc. there is also no possible crossover point as the address is a fixed location that cannot be changed.

1D)

Best fitness: 32.5

Worst fitness: 32.5

1E)

Ranked:

4 (90%~)

2 (10%~)

1, 3, 5 (all been visited previously - 0%)

Site 4 has the higher pheromone count (15) than site 2 (9) and it is also closer to site 5 (2.2) than 2 is (7.1). This means that site 4 has a much higher probability of being visited next than site 2. Site 2 would then be visited after site 4 as it is the final remaining site. I have estimated the probability of each site being visited. Site 4 is much more likely than site 2 so I estimated a 90% and 10% split respectfully. The ant would then return to the HQ (1) without visiting 3 or 5, as they’ve already been visited.

|  |  |
| --- | --- |
| Question Number: | 2 |

2A)

|  |  |
| --- | --- |
| **Best-first search** | **Uniform-cost search** |
| Informed search – has information about the distance from current state to the goal. | Uninformed search – has no additional information about the distances from nodes to the goal. |
| Sub-optimal – doesn’t always find the most optimal solution | Optimal – Always finds the most optimal solution |
| Saves space | Uses a lot of space |
| Efficient | inefficient |
| Relatively low cost and low space usage | More computationally expensive than best-first search and can use lots of physical space |
| Termination is guaranteed | Termination is guaranteed |

2B)

The problem can be modelled as a search problem by having all possible states listed and having the states that these states could travel to listed. Then the search algorithm would iterate through each state, trying out each valid state that this state could travel to until it reaches a solution. This could be modelled in a ‘nextconfig’ method, using iterative deepening to find the most optimal solution. The states could be represented in binary with a 0 or 1 signifying whether the purchases (goose, fox, beans and the boat) are present. This is shown below in the start and end configurations.

Start configuration:

{1, 1, 1, 1}

(Goose, Beans, Fox, Boat)

This show that all the purchases are present on the initial side, as is the boat.

End configuration:

{0, 0, 0, 0}

The initial side contains no goose, beans, fox or boat. They have all been moved across the river.

Below are the possible states with the quantity listed (goose, fox, beans, boat):

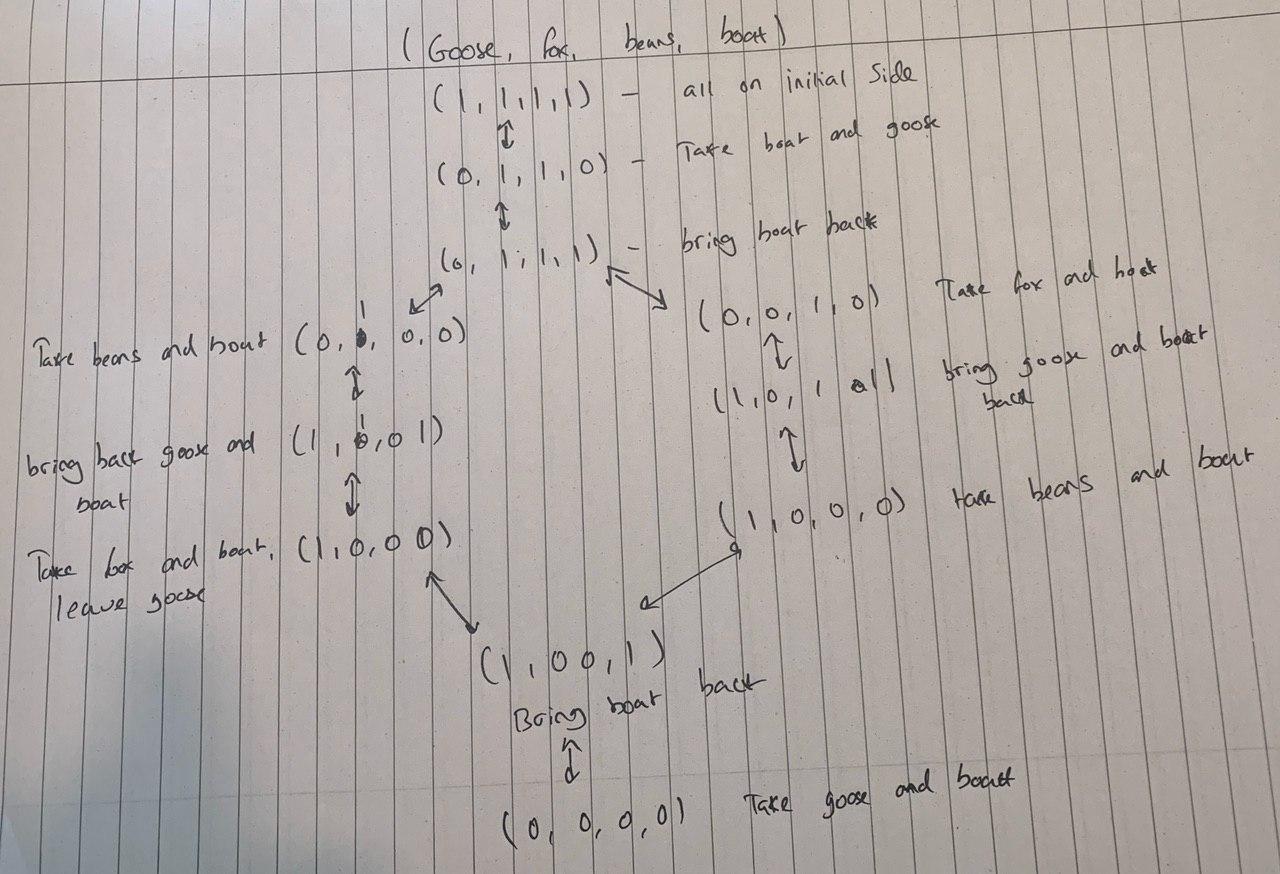


Figure 1: The states of the solution for 2B

2C)

An algorithm that would be appropriate to use for this problem would be iterative deepening. I would choose this algorithm because states would need to be revisited to find the correct solution which wouldn’t be possible with some other search algorithms, e.g. best-first search due to deja-vu checks. States would need to be revisited to ensure that all possible solutions are considered. Iterative deepening would also allow for checks to ensure that each state being visited is valid. It would also be a very efficient algorithm for a problem as small as this one with only a few variables. Iterative deepening will also find the optimal solution whereas best-first search would not.

2D)

Alpha-beta pruning is an algorithm/technique used to increase the efficiency of the minimax algorithm. It can be applied at any depth of a tree. Alpha-beta pruning reduces the number of moves the minimax algorithm evaluates by comparing whether each value given by the minimax algorithm is greater than the tree root. The algorithm is then used to prune nodes in the tree that do not affect the final decision. Removing these nodes that do not need to be searched because a better move exists improves the speed and efficiency of the algorithm as there is less of the tree to iterate over.

|  |  |
| --- | --- |
| Question Number: | 3 |

3A)

A drawing of a perceptron with the figures labelled:

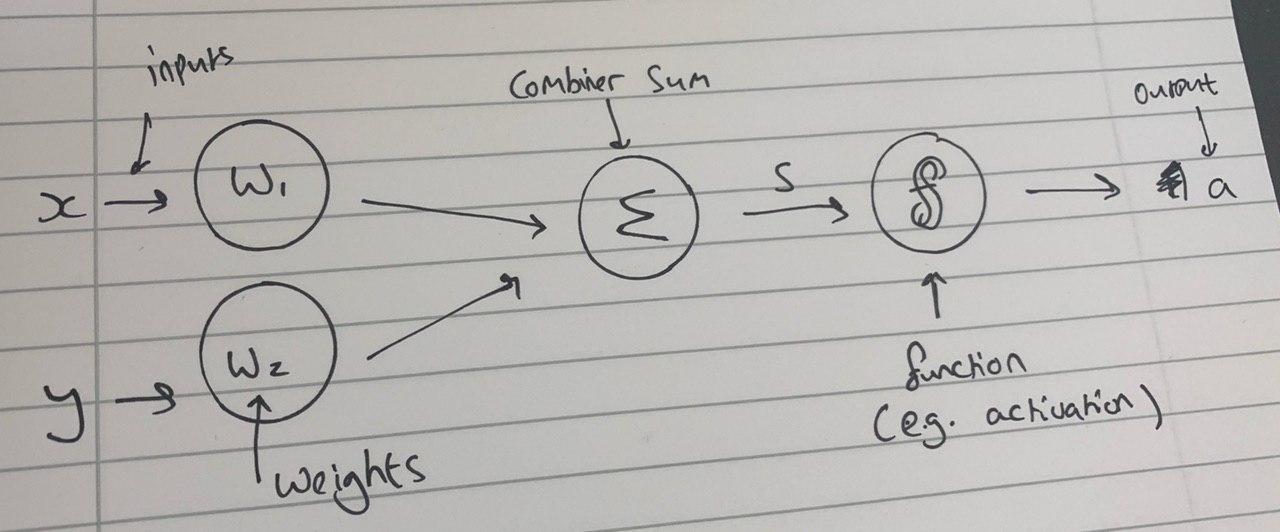


Figure 2: A perceptron with figures labelled

3D)

The perceptron could never successfully differentiate between the e-bikes and the pushbikes in this scenario for a number of reasons. Firstly, the bikes all look too similar, there is very little distinction between the e-bikes and the pushbikes with all the bikes sharing aesthetic features with no distinguishing features to show whether it is an e-bike or a push-bike. This would make it incredibly difficult for the perceptron to be able to distinguish between the two based on appearance. Secondly, the size of the frame has very little correlation to whether the bike is a pushbike or an e-bike, the red e-bike with a frame size between 48cm and 50cm is the smallest bike on the graph, with another e-bike being quite large and then the final one falling directly in the middle. This broad range of sizes means that the perceptron would be unable to distinguish the bikes based on frame size. Finally, the same applies to speed. Two of the e-bikes are the fastest bikes in the perceptron but the red e-bike is the slowest, which would make distinguishing difficult again as the range of data is very broad. Even when all of these characteristics are put together, the range of frame size, bike speeds and the appearance of the bikes makes it too difficult to determine which category of bike it belongs to. The e-bikes are also mixed in-together with the push-bikes in the graph. This means the perceptron would not be able to distinguish between them as they’re too deeply mixed.