Introduction to AI

Assignment I

Report

Tymur Lysenko

1 Algorithms description

Three algorithms were implemented for the exercise. The following subsections describe each.

1.1 Backtracking

At each point a player holding a ball has at most 4 directions to walk in and 8 directions to throw the ball in ($4 + 8 = 12$ actions in total). The algorithm considers all the "sensible" actions that can be performed from the current cell for each cell, until a human holding a ball either reaches the touchdown point or a point that is either left, right, above or below the touchdown point (since a player can see only in these directions). The main difference from brute-force is that not actually all actions are considered for each cell, only the "sensible". Since one is interested in the best solution (the one with the smallest amount of actions) it does not make sense to walk on the cells that have already been visited during the game, because eventually the algorithm will backtrack at the visited cell to consider other possible actions from this cell and it will have less actions performed than as if a player walks on the visited cell again. For the same argument it does not make sense to consider the solutions, where a passed ball flies over a visited cell.

1.2 Greedy backtracking

The algorithm is the same as the backtracking, but it is aware of the best solutions it was able to find so far: there is an initial approximation to the best solution and as it finds the better (smaller number of steps) solution it sets it as "the best so far". As the algorithm executes it checks if the number of steps of the current solution is less than or equal to the best known so far, if it is not, the current solution is discarded, because there is already one which is known to be better. The greedy approach works for this game, because of its rules.

1.3 Random search

The algorithm is the same as the backtracking, with the difference that it chooses the current actions randomly is run only 100 times.

2 Algorithms comparison & analysis

Test name	Backtracking		Greedy backtracking		Random search	
	Steps	Time (msec)	Steps	Time (msec)	Steps	Time (msec)
1-from-task	3	8	3	6	5	0
3-surrounded-by-orcs-far	-	<1000	-	<1000	-	<1000
6-10x10	-	>60000	15	375	-	<1000
9-human-chain	-	<5000	1	47	2	6
10-pass-to-win	-	~180000	3	719	17	23
13-diagonal-corridor-wide	3	32	3	13	-	0
14-maze	21	4	21	3	-	0
15-sparse	-	~1000	13	77776	25	50

Table 1: Comparison of the algorithms

In the table above "-" in the "Steps" column means that the algorithm was not able to solve the map.

The description of each map and maps themselves can be found in the corresponding tests.

As it becomes clear from the table the best algorithm in terms of time and accuracy is the greedy backtracking.

The usual backtracking succeeds in small maps or maps, where there are not much moves are sensible to make because of the orcs.

The random search works very fast due to the small number of attempts, but as the map becomes bigger it becomes hard to solve it with the algorithm.

2.1 Increasing the vision size to 2 cells

Increasing the vision size to 2 cells does not enable to solve maps that were unsolvable before (for the arguments see 2.2), it can only reduce the running time of an algorithm and increase the accuracy of the random search algorithm.

2.2 Impossible to solve maps

The necessary and sufficient condition for a map to be unsolvable is the conjunction of the following:

1. orcs surround the point (0, 0) from all sides
2. all humans to who the ball can be passed not able to walk towards the touchdown

2.3 Hard maps

As it can be seen from the description of the algorithms and from the data in the algorithms comparison table the bigger a map is and the less orcs it has the harder it is to solve.