Julian Loscombe jl14910, Ben Milne bm14368

Scotland Yard AI

Introduction

We have set up a game tree to use parallelised Alpha-Beta pruning in order to increase the depth that we can search to in the allotted time. In addition, we used an iterative depth search, continually running the tree and updating the best move on each iteration. When the time limit is near, we simply grab the last fully computed result.

Scoring

Our intuition told us that Dijkstra's algorithm would be a powerful tool as it tells us the separation between players, something that is important in Scotland Yard. Because of this we used it as the basis of our scoring function weighting it logarithmically by the ratio Mr X and the Detective's respective PageRanks. We also took into account the value of player's tickets to more closely match how players would play (e.g. to avoid using underground tickets when unnecessary). One special case we considered was to increase the score if the ticket used to get to the game state was a secret ticket and the detectives were close to Mr X.

Parallelism

After implementing the score function, we found that our game tree could only search to about depth 3 or 4 in the 15 second time limit. To improve the performance we decided to run the score functions in parallel, taking advantage of multiple cores of the processor. This improved the time taken to evaluate all of a node's children by an average of 1.6 times on a dual core CPU; we can now search to depth 4 or 5.

We also decided to let our game tree continue to run in the background whilst the detectives were making their moves. This meant we needed to remove unreachable nodes from the tree each time a detective made a move. To do this, we paused the game tree, pruned it and then restarted it at the same depth. The results of our pruning extends our search depth roughly 2 levels before reaching an equilibrium between the moves being pruned and the expanding search tree.

Results

When building our AI, we focused on making the game tree efficient over employing game tactics such as considering at what point Mr X becomes visible. It became apparent in the competition that because of this our AI made sensible moves and didn't do anything stupid, but it also refused to take advantage of certain situations.

Improvements

From the results of the competition, we feel that our AI could be significantly improved if we employed more game tactics. A particularly useful one would be to play a double move where the first move makes Mr X visible and the second move uses a secret ticket. This would increase the number of possible locations for Mr X and hence make the detective's job much harder. It became apparent when watching other teams and our AI, that a lack of forward thinking is a common issue with the game tree approach. This is something that human players are better at, although they would commonly not consider all possible moves. It would be interesting to see whether randomly pruning the tree to some degree would improve performance, looking further ahead rather than considering every possible move.