CS5804 Homework 2

Homework must be submitted electronically following the instructions on the course homepage. Make sure to explain you reasoning or show your derivations. You will lose points for unjustified answers, even if they are correct.

Written Problems

1. (Based on 5.10 in R+N) Consider the family of generalized tic-tac-toe games, defined as follows. Each particular game is specified by a set $\mathcal{S}$ of squares and a collection $\mathcal{W}$ of winning positions. Each winning position is a subset of $\mathcal{S}$. For example, in standard tic-tac-toe, $\mathcal{S}$ is a set of 9 squares and $\mathcal{W}$ is a collection of 8 subsets of $\mathcal{S}$: the three rows, the three columns, and the two diagonals. In other respects, the game is identical to standard tic-tac-toe. Starting from an empty board, players alternate placing their marks on an empty square. A player who marks every square in a winning position wins the game. It is a tie if all squares are marked and neither player has won.

   (a) (3 points) Let $N = |\mathcal{S}|$, the number of squares. Give a nontrivial[1] upper bound on the number of nodes in the complete game tree for generalized tic-tac-toe as a function of N. Explain why it is an upper bound.

   (b) (3 points) Give a nontrivial lower bound on the size of the game tree for the worst case, where $\mathcal{W} = \{\}$.

   (c) (3 points) Propose a plausible evaluation function that can be used for any instance of generalized tic-tac-toe. The function may depend on $\mathcal{S}$ and $\mathcal{W}$.

   (d) (3 points) Assume that it is possible to generate a new board and check whether it is a winning position in $100N$ machine instructions and assume a 2 gigahertz processor. Ignore memory limitations. Using your estimate in (a), roughly how large a game tree can be completely solved by alpha-beta in a second of CPU time? a minute? an hour?

2. (3 points; based on 5.19 in R+N) Consider the following procedure for choosing moves in games with chance nodes:

   - Generate some dice-roll sequences (say, 50) down to a suitable depth (say, 8).
   - With known dice rolls, the game tree becomes deterministic. For each dice-roll sequence, solve the resulting deterministic game tree using alpha-beta.
   - Use the results to estimate the value of each move and to choose the best.

   Will this procedure work well? Why or why not?

[1] You must provide justification why your bound isn't overly loose. For example, you would not get credit for reporting a bound of ∞ or (one bazillion)$^{N^{N^N}}$.