CS472 Foundations of Artificial Intelligence
Fall 2000
Assignment 2

Due Wednesday, September 27 at the beginning of class.

Collaboration: You are allowed to work in groups of 2-4 students for this assignment. Collaboration between groups is not allowed. Working alone is fine. In all cases, you must write up the solutions yourself; no collaboration is allowed for the write-up.

1. **Heuristic Search (15 points)**

   (a) (5 points) We showed in lecture (see Heuristic Search, Slides 13–14) and in Chapter 4.1 of R&N that every admissible heuristic h, when used in the pathmax equation, will lead to monotonically nondecreasing f values along any path. That is, if n' is not visited yet and is a direct successor of n then $f(n') \geq f(n)$. Does the implication go the other way? That is, does monotonicity of f imply admissibility of the associated h? Explain.

   (b) (5 points) Invent a heuristic function for the 8-puzzle that sometimes overestimates. Be sure to explain the intuition behind the function you create.

   (c) (5 points) Using A^* search, show how the heuristic from part b can lead to a suboptimal solution on a particular problem instance of the 8-puzzle. If you cannot find such an instance, explain why.

2. **Static Evaluation Functions and Minimax (25 points)**

   This problem involves the game of **checkers**. If you are not familiar with the rules of the game, you can find descriptions at one of the following locations:

   - http://www.playsite.com/games/board/checkers/index.html
   - http://www.centralconnector.com/GAMES/checkers.html

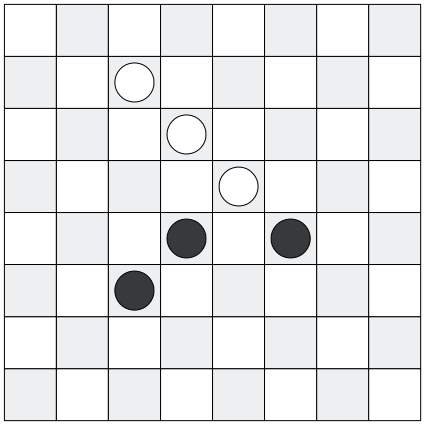

Figure 1: A board configuration from a typical game of checkers. Here none of the pieces are crowned (i.e. there are no kings), white started from the top, while black started from the bottom and it's black's turn now.

Note, that it is advisable to read the rules anyway. The course staff also thought they knew the rules, but it turned out that they were mistaken.

(a) (10 points) Approximately how many possible games of checkers are there? Be explicit about all of the assumptions you make to derive the approximation. One such possibly useful assumption is the following:

> It is a well known fact (actually a rule in the game) that if one of the players thinks that a specific configuration is a draw, but the other doesn't agree, the latter player should prove he/she can win the game in no more than 40 additional moves.

(b) (10 points) Specify a sensible static evaluation function for checkers. Briefly explain why (and possibly when) it is sensible.

(c) (5 points) Use your evaluation function from the previous problem to perform a 3-level (i.e. two moves lookahead) minmax search, starting from the configuration in Figure 1. Feel free to omit symmetric positions in order to reduce the branching factor of the tree.

3. $\alpha - \beta$ **Pruning (15 points)**

These questions refer to the game tree in Figure 1. For all questions, assume that the player making the move at the root is a "maximizing" player, i.e. he or she wants to get to a state with a high utility value.

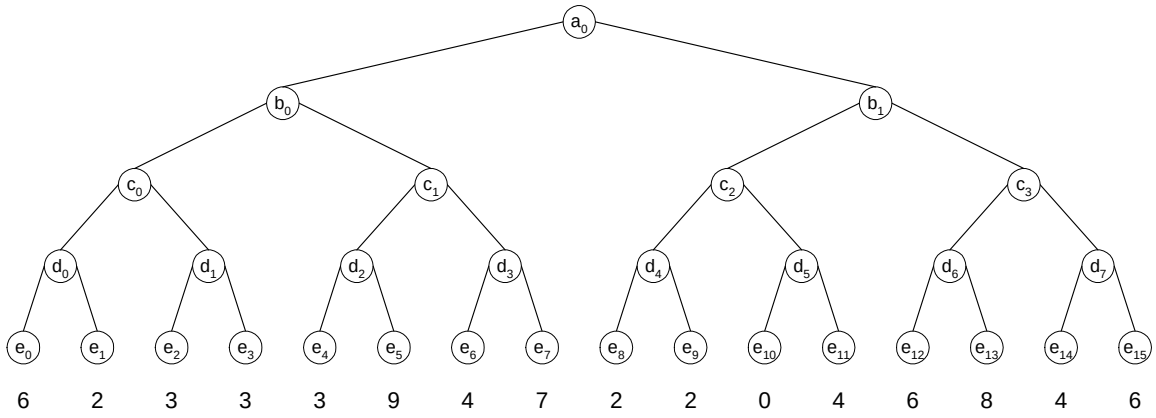

Figure 2: Game Tree for $\alpha - \beta$ Pruning.

(a) (5 points) What is the solution? That is, which move should be made next and what is the expected value of that move?

(b) (5 points) Using $\alpha - \beta$ pruning (and standard left-to-right evaluation of nodes), how many leaves get evaluated? Indicate all parts of the tree that are cut off. Indicate the winning path or paths. Strike out all static evaluation values that do not need to be computed.

(c) (5 points) How does the answer to the previous problem change if right-to-left evaluation of nodes is used?

4. **Three-Player Games (15 points)**

Let us consider the problem of search in a three-player game. (You can assume no alliances are allowed for now.) We will call the players 0, 1, and 2 for convenience. The first change is that the evaluation function will return a list of three values, indicating (say) the likelihood of winning for players 0, 1, and 2, respectively.

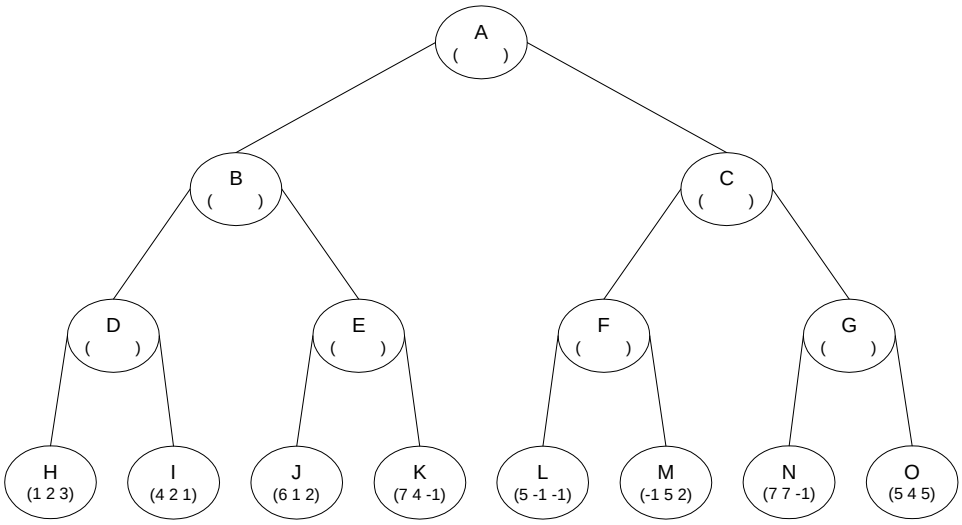

Figure 3: Three-Player Game Tree.

(a) (5 points) Complete the game tree in Figure 3 by filling in the backed-up value triples for all remaining nodes, including the root. Assume that player 0 moves first, then player 1, then player 2.

(b) (5 points) Write a new Minimax algorithm for the three-player game.

(c) (5 points) Discuss the problems that might arise if players could form and terminate alliances (in addition to making moves "on the board").

5. **Constraint Satisfaction Problems with Heuristics (15 points)**

Consider the crossword puzzle generation problem from Homework 1. (It is available at www.cs.cornell.edu/Courses/cs472/2000fa/Materials/hwk1.pdf. Now suppose we have an arbitrary crossword grid and a fairly big dictionary (e.g. with more than 50000 words). One of the ways to efficiently formulate this problem as a CSP is to make every word slot (string of consecutive horizontal or vertical squares in the grid) a variable whose domain is the set of all words from the dictionary of the appropriate length. Further, for each intersection in the grid we impose a binary constraint that the letter at which the two words intersect should match.

Specify how the **most-constrained-variable** and **least-constraining-value** heuristics might each be **inexpensively** applied. Be sure that your specification of these heuristic functions is clear and precise; provide pseudo-code if it is necessary to ensure clarity.

What to turn in: You'll have to turn in BOTH a paper and an electronic copy of the assignment.

- **On paper:** All assignments for this course must be typewritten. (Those portions of any assignment that are easier to draw by hand can be drawn in by hand.) For this assignment, turn in the answers to each question separately. (We'll have five boxes in which to deposit your assignment: one box for solutions to question 1, one box for solutions to question 2, etc.) At the top of the first page of the solution for each question, include (1) your name, (2) your e-mail address, and (3) an alphabetically ordered list of everyone in your homework group.
- **Electronic:** To protect you in the unlikely case that your homework is lost after you turn it in, we also require that you submit an electronic copy of your work (by the date and time shown above).

  1. Create a single ZIP file that contains all of the files connected with your homework solution. If you have only one file, please still create a ZIP file for it anyway (because the default transfer mode for the ftp server is binary). Unix users can use *tar* and *gzip*.
  2. Name your file with your e-mail address followed by an underscore and the assignment number, in this case "2". The final filename extension will be either ZIP or *gz* depending on what you did in step 1. So aaa999@cornell.edu_2.ZIP or aaa999@cornell.edu_2.tar.gz would be two possible filenames for this assignment.
  3. Next, FTP the file to "playpen.cs.cornell.edu" with username "cs472" and password "deepblue". Note that this directory provides write-only, no-overwrite access. This means that you get to upload exactly one version of your assignment.

  Please see the course web site for more details on electronic submission and for instructions for what to do if you need to submit a revised version of the assignment.