Department of Mathematics and Computer Science
University of Southern Denmark, Odense

November 17, 2009
Marco Chiarandini

DM533 - Introduction to Artificial Intelligence

Assignment 1, Fall 2009

The submission deadline for this assignment is **17.00 of Friday, November 27**.

For submission instructions, please refer to Assignment 0. Failure to submit properly could result in a non-passed assignment. It is not required in order to pass the assignment that all exercises are carried out correctly but do an honest effort at most of the questions, possibly all.

The first exercise is an example of possible assignment that could also appear in the written exam to test base knowledge of the course content. The other exercises in the assignments are meant at testing the capacity to apply the techniques learned. Where they do not involve programming, they could also appear at the exam. This assignment assumes knowledge from chapter 3, 4, 6, 7, 8 of the course book.

Exercises

1. Give the size, in terms of the branching factor b and the depth of search l for the open list (fringe) in each of the searches:

   a) depth-first

   b) breadth-first

   c) best-first

   d) what is the size of the closed list in each of these situations?

   Finally, define admissibility for an heuristic in A*.

   [Answers in `doc/ex1.txt`]

2. Consider the following 2-player game. Cookies are laid out on a rectangular grid $n \times m$. The cookie in the top left position is poisoned, as shown in Figure 1. The two players take turns making moves; at each move, a player is required to eat a remaining cookie, together with all cookies to the right and/or below it (see Figure 1, for example). The loser is the player who has no choice but to eat the poisoned cookie.

   Draw the search tree for this game in a 3×3 grid expanded by the left-to-right alpha-beta pruning algorithm. (Do not draw unexpanded subtrees.)

   Is this game a fair or an impartial game? That is, can one of the players always make moves that are guaranteed to lead to a win? Does the conclusion changes if $n \neq m$?

   [Answers to this question can be hand-written and then scanned in a file to put in `doc`. Name of the file `cookies.pdf` and/or `cookies.txt`.]

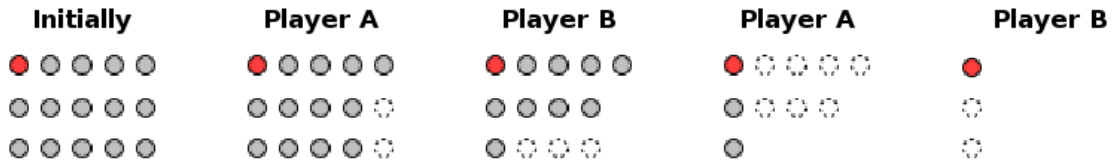

Figure 1: A sequence of moves in the game of Exercise 1 starting with a 3×5 grid. Player A must eat the last block and so loses.

3. The following code from the python aima repository implements the MIN-MAX algorithm for two-players games.

```
def minimax_decision(state, game):
    player = game.to_move(state)

    def max_value(state):
        if game.terminal_test(state):
            return game.utility(state, player)
        v = -infinity
        for (a, s) in game.successors(state):
            v = max(v, min_value(s))
        return v

    def min_value(state):
        if game.terminal_test(state):
            return game.utility(state, player)
        v = infinity
        for (a, s) in game.successors(state):
            v = min(v, max_value(s))
        return v

    # Body of minimax_decision starts here:
    action, state = argmax(game.successors(state),
                           lambda ((a, s)): min_value(s))
    return action
```

Let's consider a three-player game (without alliances) and let's call 0, 1, 2 the three players. Each terminal state has now associated three values indicating the likelihood of winning of player 0, 1 and 2, respectively.

a) Does the code above implement a MIN-MAX algorithm also for the three-player game? If not what has to be changed?

b) Complete the game tree of Figure 2 by filling the backed-up value triples for all remaining nodes.

[Write your answer in `doc/multiplayer.txt`.]

4. The game of *Sim* is played by two players on a board consisting of six dots ('vertices'). Each dot is connected to every other dot by a line (see Figure 3).

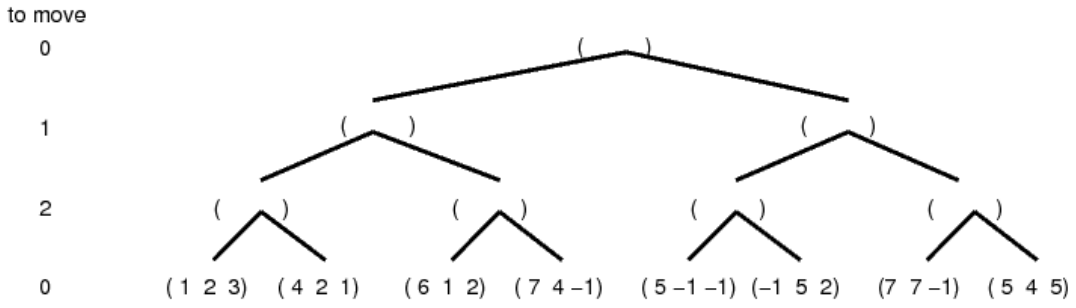

Figure 2: The game tree of Exercise 2.

Two players take turns coloring any uncolored lines. One player colors in one color, and the other colors in another color, with each player trying to avoid the creation of a triangle made solely of their color; the player who completes such a triangle loses immediately.

The game Sim is one example of a *Ramsey* game. Another Ramsey game uses pencils of three colors. The two players alternately color an edge of the graph, using any color they want to, until a player loses by completing a mono-chromatic triangle. [source: Wikipedia]

What you should do:

(a) Using the source code `games.py` provided from the course book web repository, implement the first Ramsey game, ie, the Sim game. It should be possible for a human player to play against the computer, and moreover the computer should make "reasonable" moves, i.e., it should be able to win sometimes. To asses this, compare the computer against the random move strategy. If your program does not beat the random move strategy, it does not count as a valid solution to this exercise. The computer response time should be "reasonable". On average, response time should not be longer than 5 seconds.[1] However you should also evaluate whether playing the perfect strategy would be feasible in terms of time (in other terms, whether you can answer this specific question before the hand in deadline expires).

(b) Determine whether we can play the game also in a more general setting of n vertices and, in affirmative case, which feature makes the computational hardness.

(c) Implement as for point 1 the second Ramsey game on a graph of size ≥ 6.

Hints on solving the exercise

If the state space is too large to just make an implementation of the minimax-algorithm or the alpha-beta-search, an evaluation function becomes necessary. That is, you need to decide the value of a given coloring heuristically. In this case, experiment with the cut-off-depth of the algorithm together with your evaluation function to examine the possible trade-off between longer calculation time and smarter moves.

Deliverables

[1] In Python, use the `time` module and the its `clock()` function to measure elapsed time.

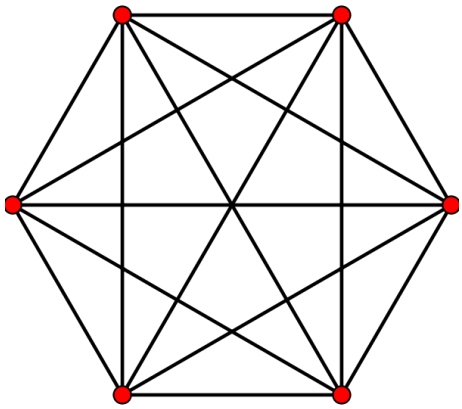

Figure 3: The K_6 graph for playing Sim.

You should hand in a short report in `doc/ramsey.txt` where you summarize your findings and two files `sim.py` and `ramsey.py` in `src`. Running your `sim.py` program, it should be possible for the instructor to play by stating the edge to color declaring the corresponding tuple `(1,3)`, that indicates the coloring of the edge $(1,3)$.

You are encouraged to constitute small groups and to let your programs compete against each other within your group. This should be straightforward if you use the framework of the code provided. If you do this, report the experience in `sim` and `ramsey.txt`. (Note: this encouragement is not about collaborating in the solution of the exercises in the assignment that you should instead carry out individually).

5. Using the python code `logic.py` or `gprolog` and given the following KB:

   > If the unicorn is mythical, then it is immortal, but if it is not mythical, then it is a mortal mammal. If the unicorn us either immortal or a mammal, then it is horned. The unicorn is magical if it is horned.

   Is the unicorn mythical? How about magical? Horned?

   [Answers in `doc/ex5.txt`]

6. Exercise 7.2 of the course book. You may use Prolog or the `logic.py` code from the Python repository.

   [Answers in `doc/ex6.txt`]

7. Propositional Logic. Assume we have the following propositions: *BatteryDead*, *RadioWorks*, *OutOfGas*, and *CarStarts*. (You may use the abbreviations B, R, O, C in your answer.)

   a) What is the total number of possible models?

   b) How many models are there in which the following sentence is false?
   $(RadioWorks \wedge CarStarts) \implies (\neg OutOfGas \wedge \neg BatteryDead)$

   c) Is the above sentence equivalent to a set of Horn clauses? Explain.

d) Prove that the above sentence is not entailed by the sentence $RadioWorks \implies \neg BatteryDead$.

[Answers in `doc/ex7.txt`]

8. First-order logic. Let $M(x)$ be true if x is a mail carrier; $B(x)$ be true if x lives in Odense; and $K(x, y)$ be true if x knows y. Translate the following sentences into first-order logic:

   a) There are at least two mail carriers who live in Odense.

   b) All the mail carriers who live in Odense know each other.

   [Answers in `doc/ex8.txt`]

9. Resolution.

   a) Explain how to prove that two sentences A and B are logically equivalent using resolution.

   b) Consider the two
   A: $\forall x[\exists y P(x, y)] \implies Q(x)$
   B: $\forall x, y\ P(x, y) \implies Q(x)$
   Prove that A entails B.

[Answers in `doc/ex9.txt`]