Artificial Intelligence Assignment 3 Readme Document

Author:
Joshua Wang
UNI: jw3123

1. The programming language in the project is C++
2. The version of programming language is C++ 1998 standard
3. Development environment is:
   Application: Sublime text2
   Operating System: Windows 8
   Processor: Intel i7-4702MQ
   Memory: 8GB DDR3
4. How run and compile on CLIC:
   (1) compile the file by typing:
   g++ player-jw3123.cpp (<bin_filename>)
   (2)execute the binary file:
   ./ (<bin_filename>) n m s
   (where n is the game board dimension, m is winning chain length, and s is move selection time limit)

A command line, indicating how to choose modes, will prompt. In mode 1 and mode 2, it will ask to enter an odd number for AI to move first; or an even number for stdin or random placing first.

5. The design of the evaluation function:

(1) The evaluation function records the number of different lengths chains (both the agent and the rival) and only the chains of one open end or two open ends will be recorded. (In other words, the two-sided blocked chains will not be recorded). This information is the crux of the result of the game-when one party gets one m-1 two open ends chain (m is winning chain length), or two m-2 two open ends chains, or one m-1 one open end chain with one m-2 two open ends chains, it will win the game. With the information recorded, we can reduce the depth of tree. When none of the above victory condition is met, use a linear function to map the number of these chains to a score, which is based on the intuition that the more these desirable chains there are, more likely it will go to the victory state.

(2) However, we discover that as the agent completes a m-1 chain, it might place stones at other places for pointless defending rather than achieving victory. This is because these place all have the same score, i.e., infinity. A modification is made by

including the depth of the node of the state to the evaluation function so that with the same original value, the shallower nodes will have a higher score. This scheme solved the problem, and we can see in mode two that sometimes the agent will form a chain “determinedly”.

(3) The information of the evaluation function is retrieved by parsing the board in all 4 directions, thus the complexity is *$O(n^2)$*. An way to handle the information by parsing only 4 lines, which will be *$O(n)$*, across the newly laid stone, is tried. The difficulty of this implementation is that the information for evaluation function (number of desirable chains over the whole board) is global, which is hard to implemented by looking only four lines.

6. Outcome of the program:

If not particularly mentioned , we used n=10, m=5, s=10.

Mode1: the agent is offensive, no matter it is the first or second player, and always wants to make a chain of 4 and 3. If the agent play as x (first), I would have to take defense all the time. However, sometimes it overlook holes that can be filled to be 5, and that is my opportunity to win.

Mode2:

As mentioned in 5(2), in mode 2 , the agent “determinedly” form a chain regardless of the random placing. The following is an example of an outcome.

```
. . . . . . . . . x
. . . . . . . . . .
. . . . x . . . . .
. x . . . . . . . .
. . . . . . . . . .
. . . . . . . . . .
. . . . . . . . . .
o o o o o . . . . .
. . . x . . . . . .
x . . . . . . . . .
```

Mode3:

An interesting phenomenon about mode 3 is the evaluation function and m drastically affect the result. The current version of evaluation function will result like a real game when m=6. However, if we add the information about the chains with length 1, two agents will just enlarge chains, and makes x the winner. Also, the same situation will appear when m=5 with the current evaluation function.