Theory of Computer Games 2022

Report of Project #1

Student name: 劉子齊 Jonathan
Student ID: 311605004

In project 1, our goal is to implement a simple heuristics-based player. For the framework of the game, and the environment, I followed the given sample.

The original given method was to play randomly, but care nothing about the reward. In the following method which I implemented, I applied the backward method and an 8 x 4 tuple network. In my implementation, I calculate the corresponding reward of each operations, which are “slide up”, “slide down”, “slide left”, “slide right”. By calculating reward of each potential operations, I will choose the operation with the highest reward and act to it. The following figure is the implementation of my greedy sliding method.

```
179  class greedy_slider : public random_agent {
180  public:
181      greedy_slider(const std::string& args = "") : random_agent("name=greedy role=slider " + args),
182          opcode({ 0, 1, 2, 3 }) {}
183
184      virtual action take_action(const board& before) {
185          //std::shuffle(opcode.begin(), opcode.end(), engine);
186          board::reward max_reward = -1;
187          int best_op;
188          for (int op : opcode) {
189              board::reward reward = board(before).slide(op);
190              if(op == 0) {
191                  reward = 8 * reward + 1;
192              }
193              if(op == 1) {
194                  reward = 8 * reward + 1;
195              }
196              if(op == 2) {
197                  if(before[3][3] != 0)
198                  reward = 2 * reward + 1;
199              }
200              if(op == 3) {
201                  if(before[0][0] != 0)
202                  reward = 2 * reward + 1;
203              }
204
205              if(reward > max_reward) {
206                  max_reward = reward;
207                  best_op = op;
208              }
209
210          }
211          if(max_reward > -1) return action::slide(best_op);
212          return action();
213      }
214
215  private:
216      std::array<int, 4> opcode;
217  };
```

With my greedy sliding method introduced above, I obtained a way better result than the original strategy with the score of “92.3” in the provided Linux workstation, which is shown in the following figure.