

Theory of Computer Games 2018 - Project 6

In the series of projects, you are required to develop AI programs that play [Threes!](#), the origin of other 2048-like games.

Overview: **Join the Threes! tournament.**

1. Fine-tune the player and the antagonistic environment.
2. Make sure that the program can establish a stable internet connection with the arena.
3. Compete in the tournament.

Specification:

1. The rules are the same as the rules for Project 5:
2. Train the agent by temporal difference learning and any other possible improvements.
3. Statistic is required, and the requirement is the same as Project 2's requirement.
 - a. Average score.
 - b. Maximum score.
 - c. Speed (action per second).
 - d. Win rate of each tiles.
4. The arena protocol is required.
 - a. Input arena commands (variables are marked in bold):
 - i. **#ID open AGENT : AGENT**
You will receive this message when a new game is pending, where **#ID** is the game ID, and **AGENT : AGENT** are the names of player and environment.
 - ii. **#ID ?**
You will receive this message when your agent needs to take an action.
 - iii. **#ID ACTION**
You will receive this message when your opponent took an action, where **ACTION** is the action code.
Action codes for player: **#X**, where **X** is either U, D, L, or R. The regular expression is `# [UDLR]`.
Action codes for environment: **PT+H**, where **P** is the position of new tile; **T** is the index value of new tile; **H** is the hint of next tile. The regular expression is `[0-9A-F] [0-9AB] \+[1234]`, e.g., F3+2, 51+1, 45+3, A2+4.
 - iv. **#ID close FLAG**
You will receive this message when a game just finished, where **FLAG** is the closing flag (the winner, or the reason of closing the game, etc.).
 - v. **@ login**
You will receive this message once you establish a connection to the server, and you should reply your account and agents.
 - b. Output arena commands:

- i. **#ID ACTION**
This message should be sent when you need to perform an action. The action code definition is the same as above.
 - ii. **#ID open OPEN_REPLY**
This message should be sent when you decide to join a game, where **OPEN_REPLY** should be either `accept` or `reject`.
 - iii. **@ login ACCOUNT AGENT (ROLE) . . .**
This message should be sent when you receive the login message. If you have multiple agents, separate them with a space.
5. Internet connection is required.
 - a. Send and receive arena commands over SSL connections.
 6. Speed requirement: At least **10 actions** per second (over the internet).
 7. Implementation details:
 - a. Your program should be able to compile and run under the workstation of NCTU CS.
 - i. Write a makefile (or CMake) for the project.
 - i. C++ is highly recommended for TCG.
You may choose other programming language to implement your project.
 - b. Your implementation needs to follow the statistic output format.
(see Project 2's spec for details)
 - c. Please refer to the sample code for the details of arena protocol.
 8. Your program should be able to serialize and deserialize the weight tables of N-tuple network.

Methodology:

1. There is **no minimum # of bonus tiles requirement** in Project 6.
 - a. Generating all the tiles from the bag is now acceptable.
 - b. **At any timestamp of an episode**, $0 \leq \frac{\text{\# of bonus tiles}}{\text{\# of total generated tiles}} \leq \frac{1}{21}$.
2. The arena protocol has already been implemented in the sample code.
 - a. While you need to modify some lines. (e.g. add hint tile processing code)
 - b. Also, you don't need to handle the internet connection by yourself, just keep your program using standard input and output. We will provide you with a tool to link your program with internet.

Submission:

1. Your solution **should be archived in zip/rar/7z file**, and **named as XXXXXXXX . zip**, where XXXXXXXX is the student ID (e.g. 0356168 . zip).
 - a. Pack your **source files, makefiles**, and other relative files in the archive.
 - b. Do **NOT** upload the statistic output or the network weights.
 - c. Provide the version control repository of your project (URL), while do **NOT** upload the hidden folder (e.g. **.git** folder).
2. Your project should be able to run under the workstations of NCTU CS.

- a. **Test your project on workstations.**
- b. Only run your project on workstations reserved for TCG. Do not occupied the normal workstations, otherwise you will get banned.
- c. Respect the rights of others. Do not occupied the resources of workstation.

Scoring Criteria:

1. Demo: **TBD.**
2. Framework: Connect to the arena server.
 - a. **Your score is not counted if the program cannot connect to the server.**
3. Ranking (100 points):
 - a. The score is based on the ranking, between 70 and 100.
 - b. Please refer to e3 discussion forum for the rules of the tournaments. (TBD)
4. Maximum and minimum score (bonus):
 - a. The player/environment who achieve the max/min score in the tournament will get an extra 1 point as the bonus.
5. Penalty: **TBD.**
 - a. Please refer to e3 discussion forum for any penalties in the tournaments.

Hints:

Having some problems? Feel free to ask on the Discussion of e3 platform.

You may use [Github Student Developer Pack](#) or [Bitbucket](#) for the version control.

Remember to share the sources on sharing platform, for example, [GitHub Gist](#).