# Project 2: Adversarial Search

Due: 23:59, March 23, 2019

## 1 DotA

**1.1 Background** DotA 2 is a multi-player online battle arena (MOBA) video game in which two teams (the Radiant and the Dire), each with **5 players**, compete to collectively destroy a large structure defended by the opposing team known as the "Ancient", thus the name Defense of the Ancient, or DotA.

The AI research community has grown unprecedented interests in real-time strategy (RTS) and MOBA games after winning against the best human Go players in 2017. In 2018, AI bots made a breakthrough: OpenAI's best performing AI bot defeated an amateur team of 5 human players in relaxed game rules.[1] In 2019 (during this semester), we have also witnessed DeepMind's AlphaStar defeating strong human players in Starcraft II.

**1.2 Project Goal: Given the current rosters and pool, choose the next hero that eventually maximizes your advantage against your opponent.** You are the captain of the Radiant in a DotA 2 tournament. You and your opponent must choose from the same pool of $N$ heroes, and each hero can only be picked once. You will take turns choosing (Radiant $\rightarrow$ Dire $\rightarrow$ Radiant $\rightarrow$ ... $\rightarrow$ Dire),[2] and you each must choose on your turn (i.e., no skipping your turn). You will choose first. The process ends after your team and your opponent team both have exactly 5 heroes (i.e., 10 distinct heroes are picked in total).

### 1.3 Hero Attributes

#### 1.3.1 A Hero's Power:

Not all heroes are created equally. They not only look different, but also have different skills, strengths, etc. In this project, all the characteristics of a hero are quantified using a single number called **power**, $p$. It is possible that some heroes are over-powered, i.e., they have higher power than others.

#### 1.3.2 Team's Mastery over a Hero:

Another factor that affects your advantage is how well your team can play a hero. This attribute is referred to as **mastery**, $m$. For example, your team is good at playing Meepo and has 1.0 mastery; your opponents,

[1] However, the AI bots were later defeated by world champion teams: `https://blog.openai.com/the-international-2018-results`

[2] This is a simplified version, not the picking order adopted in a real game. Interested readers can also watch this video for how hero selection affects the win rate. Maneuvering aspects are omitted in this project, but interested reader may read this interactive description to get an idea how a game AI works.

on the other hand, are very bad at playing Meepo, and have 0.3 mastery. In this case, Meepo, whose power is 67.0, contributes 67.0 * 1.0 = 67.0 to your advantage if you choose him; if you did not choose Meepo but your opponents did, their gain in advantage is 67.0 * 0.3 = 20.1.

### 1.3.3 Synergy:

The third and last factor that affects your advantage is hero **synergy**, $s$. Some combination of heroes are especially powerful in DotA. This is because some heroes are good at disabling the opponents while having little damage outputs; others can carry the whole team by efficiently obliterating an opponent, but they are also very fragile. In this project, the type of a hero is indicated by the last digit of hero IDs: when there are 5 heroes that all have different last digits in your roster, you gain a bonus of the 120 points of advantage.

For example, you picked Hero 10001, 10002, 10003, 10004, 10005. Suppose each hero individually contributes 100 points of advantage to your team, you will then have a total of $5 \times 100 + 120 = 620$ advantage points. In contrast, if you choose Hero 20001 instead of Hero 10005, your overall advantage will be just 500.

### 1.3.4 Advantage:

Your advantage, $A$, is calculated using the following equation:

$$A = \left(S_{radiant} + \sum_{i=1}^{N_{radiant}} m_i \times p_i\right) - \left(S_{dire} + \sum_{j=1}^{N_{dire}} m_j \times p_j\right), \tag{1}$$

where

- $p_i$ is the power of the $i$-th hero you choose,
- $m_i$ is your team's mastery over it,
- $p_j$ is the power of the $j$-th hero your opponent chooses,
- $m_j$ is the mastery of the opponent team over it,
- $S_{radiant}$ is the synergy bonus of the Radiant (your team), and
- $S_{dire}$ is synergy bonus of the Dire (the opponent team).
- $N_{radiant}$ is number of heroes that you have chosen so far ($N_{radiant} \leq 5$).
- $N_{dire}$ is number of heroes that your opponent have chosen ($N_{dire} \leq 5$).

$S_{radiant}$ and $S_{dire}$ are both constant values (120) if a team has 5 heroes whose last digit of hero IDs are all different. Your opponent team's advantage against you is $-A$. We refer to A as **advantage** and the product $m_i \times p_i$ as **gain** in advantage.

Given a hero pool and both teams' mastery data, you want to maximize your advantage $A$ while your opponent wants to minimize $A$. This is henceforth an adversarial search problem.

## 2 Data

### 2.1 Hero Attributes

- Hero ID: a unique 5-digit number (32-bit integer with range [10001, 99999]). No two heroes have the same ID in a test case.
- Hero's power $p$: a 64-bit double in the range $[0.0, 200.0]$.

- Your team's mastery $m_i$ over this hero: a 64-bit double in the range $[0.0, 1.0]$.
- Your opponent team's mastery $m_j$ over this hero: a 64-bit double in the range $[0.0, 1.0]$.
- Membership indicator: (integer) whether this hero
  - has already been picked by you (1),
  - picked by the opponent (2), or
  - is still available in the pool (0).

Note that for simplicity, the total number of chosen heroes (marked 1 and 2) is always even, so you are always the one who is going to choose a hero.

These attributes will appear in the input file in the above order separated by commas. Note that the end of a line is always a line feed (LF) character (ASCII `\x0a`, a.k.a. `\n` in many programming languages). An example can be found in the following line:
`50601,102.300000,0.9487,0.8917,0`

**2.2 Input Format** The input file (named `input.txt`) has the following form:
First line: the number of heroes in the pool; this number is always great then or equal to 10).
Second line: algorithm to use,

- either **minimax** for *minimax* algorithm,
- or **ab** for *alpha-beta pruning*.

Each of the rest of the lines is a hero's attribute. An example is as follows:

```
15
ab
75201,96.192554,0.134675,0.822285,0
64504,97.537866,0.239586,0.572906,0
83601,177.631835,0.547191,0.251238,1
87705,131.105311,0.931969,0.653667,1
25202,154.053795,0.075295,0.993499,0
39202,70.595928,0.208003,0.255219,0
42904,6.518783,0.426926,0.432817,0
12703,75.011463,0.456201,0.037517,2
55502,108.823333,0.127049,0.159396,0
82301,112.524738,0.755311,0.406141,0
46902,12.316246,0.511181,0.217463,1
98403,131.132871,0.236225,0.612434,2
79802,158.762381,0.880846,0.579115,2
88903,187.685198,0.277158,0.679969,0
79604,67.906728,0.279886,0.628635,0
```

In this example, the first available hero (Hero 75201) has power 96.192554. Your team is really bad at using this hero (with mastery 0.134675) whereas your opponent team is good at it (with mastery 0.822285). Your team has already taken Hero 83601, Hero 87705, and Hero 46902 whereas the opponent team has Hero 12703, Hero 98403, and Hero 79802.

Lastly, we will not provide an invalid input file to you.

**2.3 Output Format** The output file (named `output.txt`) should have just one line which is the ID of the hero which your team is going to choose next. The end of line is again an LF (ASCII `\x0a`).
An example is as follows:
`79605`

**2.4 Tie Breaking** If there is a tie (multiple heroes potentially contribute exactly the same amount of advantage and are thus equally good to choose), please always choose the one with the smallest ID.

**2.5 Branching** Please iterate through nodes in ascending order of Hero IDs. For example, if the pool has Hero 10001 and Hero 50002, branch out to Hero 10001 first.

## 3 Project Rules

1. The main file of your solution should have a base name `project2cs360s2019`.
2. Your program must finish each test case within 3 minutes on Vocareum. Although upper bounds have been provided, test cases are created with this time limit in mind, and you do not need to expect the maximum size of all parameters in one test case. **However, you should expect cases that reach at least one upper bound.**
3. You may use **ONLY C++, Java, Python 2.7**, and **any libraries that are included in Vocareum**.
4. C++ and Java files will be compiled using the following commands (you may confirm this will work by submitting your code for testing on the small submission test cases):
   (a) C++
      - `g++ -o project2cs360s2019 project2cs360s2019.cpp`
      - `timeout 180s ./project2cs360s2019`

   (b) Java
      - `javac project2cs360s2019.java`
      - `timeout 180s java project2cs360s2019`

   (c) Python
      - `python project2cs360s2019.py`
5. Projects must be submitted through Vocareum. Please only upload your program code. Don't create any subfolders or upload any other files. Please refer to this to get started.
6. You are strongly encouraged to submit your program 24 hours ahead of the deadline to avoid any last-minute submission issues on Vocareum.
7. There is a grace period of 24 hours during which time your submission will be accepted, but with a 20% penalty. After 24 hours, your submission will not be accepted. Requests for grading a previous version/submission will not be considered, so please make sure your last submission is correct.

## 4 Academic Honesty and Integrity

All project material is checked vigorously for dishonesty. Violations of academic honesty are forwarded to the Office of Student Judicial Affairs. To be safe, you are urged to err on the side of caution. Do not copy work from another source. Sanctions for dishonesty will be reflected in your permanent record and will negatively impact your future success. As a general guide:

- Do not copy code or written material from another student.
- Do not collaborate on this project. The project is to be solved individually.
- Do not copy code off the web. This is easier to detect than you may think.

- Do not share any test cases that you may create to check your programs behavior in more complex scenarios with other students.
- Do not copy code from past students.
- Do ask the CPs/Graders/TAs/professors if you are unsure about whether certain actions constitute dishonesty. It is better to be safe than sorry.