Algorithms Final report

Andrew Ojemann

**Abstract**

This semester I decided to work on a Hero selection algorithm for once of the most popular game genres in the world, MOBAs. MOBAs are player vs. player games where 5 players team up to take on an opposing team of the same size. What I did was create a basic selection algorithm based on player experience, success rate, as well as global success rates with specific heroes. I also looked into calculating in a counter picking element into the draft phase. Using this algorithm I created a prototype of an application that can recommend picks once player data is added.

**Introduction**

One of the most popular genres of games in the world at the moment are the group of games know as MOBAs. Before I get into the specifics of my project I'll give a little context as to the way a MOBA works so that my project will be in context.

A MOBA is a competitive game played in 5v5 teams. Players take turns selecting characters from a shared roster and then fight each other in a Player vs. Player environment. Games in this genre include such titles as League of Legends, Dota2, Smite, and Heroes of the Storm. The important part of the game I will be focused on is the pre-game pick/ban phase where players select what characters they are going to play with. All players must pick from the same pool of characters and no two players may choose the same character. Teams for the sake of clarity we be called the Red and Blue team. Teams pick in alternating order so that the first player on Blue teams picks then the first Two players on Red then the next two on Blue and so on till all players have picked. This ends up looking like this 1-2-2-2-2-1 (colors indicating team). This is a very important part of the game that takes place even before the players set foot on the battleground.

For my application I chose to use Heroes of the Storm, a new MOBA in development by Blizzard, for my test case. It is a game which I have personal interest in and data is available for.

My application calculates what it thinks the best hero picks for players on a team are, based on their win rates with each hero, the number of games played on each hero, the global win rate of each hero, and the roles currently filled on the team. Players start up the app and indicate which side their team is picking on, red or blue. Then they take turns drafting against the machine. When it's the players teams turn to pick the app calculates what it believes to be the best pick for the player currently picking, when it's the opposing teams turn to pick the player enters whatever the enemy picks into the application.

**Methods**

Instructions for running: When running the game first pick red or blue form the top, this will let the computer know which team is your team. It will follow the pick order shown above, picking for your team when appropriate. Your teams pick will be on the right under the hero list, your opponents on the left. When it is not your teams turn to pick, the app will wait for you to enter what your opponent picked, in a real situation you would see what they picked in game and enter it, but for this test you can pick any random characters you want. To pick a character imply click on the portrait in the hero selection list in the middle of the screen.

My algorithm uses a weighted win rate formula: (Wins + constant \* win rate for all players) /(wins + losses + constant). It also makes sure to never pick duplicates of a hero that has already been selected, and makes sure to always fill out the roles needed by a team. It also adds an additional 10% win rate penalty for heroes a player has played less than 10 games on. Testing has shown that players still learning a hero will significantly underperform, and as such should be discouraged from trying new heroes in draft till they have more practice in a less competitive game mode.

Attached is the application, there is a PC build as well as the unity project I created it with. The math is done in the weighted win rate script, while all iteration and storage is handled by the master control script.

Data was gathered from a third party site: <https://www.hotslogs.com/Default>.

**Results**

I was able to create a working application that implemented the algorithm in a easy to use interface. For sample players is this prototype I used my own data for player 1: <https://www.hotslogs.com/Player/Profile?PlayerID=980105>. For testing purposes, player 2, 3, and 4, are made up players that have 50% win rates on all heroes, with 2, 20, and 100 games played on each respectively. For the final player, I created a made up player with 50 games of every hero, with random win rates.

I was originally planning on calculating in champion counter pick win rates as well, however Heroes of the Storm has no official API yet and I don't know much about scraping data off web pages, so i was entering the data manually. There are 35 heroes currently, which means 1190 win rates to enter manually. I unfortunately ran out of time to do this, however, the code for calculating win rates with the counter picking component is implemented in the weighted win rate script, it's just not hooked up, because it has no purpose without data.

**Discussion**

In MOBAs players have a limited amount of time to consider picks, for Heroes of the Storm its 30 seconds. What this application dose is do a ton of math very quickly to give player a suggested starting point of what to pick. Currently the app only displays what it thinks the best choice is, because I felt this would keep the app simple and easy to understand for first time users, but it would be trivial to have it display the best 3 or 5, to give players a wider range of options.

As expected upon testing the app without counter picking gives fairly consistent results, returning picks that are both strong for the player as well as good overall. Players 2, 3, and 4 follow the global percent win rate chart as close as possible. The difference in their number of games played only effects their distribution away from the center, 50%, with player 4 at 100 games being the most tightly grouped. This algorithm is most useful and interesting on datasets of real players who's varying win rates and experience make this kind of pattern impossible to see quickly when under time pressure.

I think that with access to a real-time API, which will likely be released eventually, as many other MOBAs already have them, this app could provide players with valuable information to help improve their draft phase, the part of the game which is currently hardest to practice for and improve at. With the growth of esports, where professional players are paid to play these game, and coaches are hired to train them, this kind of tool could have a real market in the near future. It also has application for balancing the game as well. Developers can use it to simulate the effects changes will have on pick rates in order to get a more complete picture before making changes.

**Sources**

Background Research:

1. http://en.wikipedia.org/wiki/Greedy\_algorithm (element of need to pick)
2. http://en.wikipedia.org/wiki/Stable\_marriage\_problem (possible applications in pocket pick, with heavy modification)
3. http://rsta.royalsocietypublishing.org/content/367/1906/4339
4. <http://en.wikipedia.org/wiki/Rating_Percentage_Index>
5. <http://stackoverflow.com/questions/7509219/weighted-win-percentage-by-number-of-games-played>

Materials not relevant but contain academic descriptions of MOBAs:

1. http://research.ijcaonline.org/icrtitcs2012/number10/icrtitcs1465.pdf
2. http://www.fdg2014.org/papers/fdg2014\_paper\_36.pdf