Ultra Rapid Figures Process Book

Sean Bennett and Charles Zhang

Overview and Motivation: Provide an overview of the project goals and the motivation for it. Consider that this will be read by people who did not see your project proposal.

We're both really into playing League of Legends, and we're very excited about the new game mode, Ultra Rapid Fire, that was released on April Fools Day. This new game mode has brought about a lot of changes, and we would like to use visualizations to further explore those changes in a way that no website or service currently allows.

One of the most important things in a game like League of Legends is game balance. There are 124 champions to choose from, and the game developers, Riot Games, do their best to make sure no one champion is categorically stronger than any other. Normal game modes have been around for years, so Riot has had plenty of time to make sure that all champions and items are as balanced as possible in normal. However, since this game mode is so new, there are drastic balance issues that we are excited to learn more about.

Related Work: Anything that inspired you, such as a paper, a web site, visualizations we discussed in class, etc.

This project is something we were interested in doing since the very beginning of the class, so there is no one clear motivation for the project other than our interest in the game. However, there are some websites that currently assess the current balance/win rates of champions in the game, as well as popularity and win rates of item builds, although none use particularly compelling visualizations. Some of these websites/visualizations can be seen here:

- http://www.metalol.net/champions/
- http://www.op.gg/statistics/champion/

However, these websites all focus on statistics gathered from games played in normal game modes, not Ultra Rapid Fire mode, since Ultra Rapid Fire mode is such a brief phenomenon. This means that our project will answer questions that not many other websites/developers are trying to solve.

We were also impressed and partially inspired by last year's Data Driven Dota project. Although our project will be much different because it will assess fairness and balance on a general level instead of looking at statistics for specific users, some of the elements of their visualization would be very useful in our visualization.

Questions: What questions are you trying to answer? How did these questions evolve over the course of the project? What new questions did you consider in the course of your analysis?

We would like to explore exactly how the game of League of Legends has changed due to the introduction of the Ultra Rapid Fire Game mode. Riot Games

puts in a lot of effort making sure champions and items are balanced and fair in normal games, but URF disrupts that balance.

We want to be able to explore the popularity and effectiveness of certain champions, popularity and effectiveness of items, changes in game pace, etc. in this new game mode. The visualization we create will give users an idea of how to compose the strongest team with the best item builds, find sleeper picks, and avoid picking champions that don't live up to their hype.

Some of the questions that we hope will be answered by our visualization include the following:

- Which champions are the most popular in URF?
- Which champion should I pick to give my team the greatest chance to win an URF game?
- Given the champion I want to play, which items should I buy, and which should I stay away from in order to maximize my chances of winning the match?
- Which champion choices result in the shortest game durations? Is this because they are very good, or because they are very bad?
- How are metrics like kills, deaths, and assists affected by champion selection?

Data: Source, scraping method, cleanup, etc.

Riot Games (the developer of League of Legends) has an API to pull data for specific games. With the introduction of Ultra Rapid Fire mode, they also introduced a function that, given a specified time, would return a randomized set of game ids of games completed within five minutes before the specified time. This appears to be the only way to gather data for URF games, so we wrote a script that, for every five minute interval since the introduction of URF, called the function and saved the game ids that were returned. This resulted in a list of 35,000 games, which our script then exported as a csv file.

The Riot API also includes a function that, given a specific game id, returns data for that game. However, the data for each game is in a json of about 150 kB in size, so it would be impractical to save all that data and use it in our visualization. We wrote a script that, for each game id we had, used the Riot API function to pull the json containing all data for that game, then pulled specific fields to create a new json that could be used to construct the data for our visualization. We saved all of these shrunken-down jsons, each representing one game's worth of data.

Our final data processing script began with an empty json frame with space for 123 dicts, each of which represented all the metrics we were interested in for each champion. Then, for each game data json we had for all of our 35,000 games, we added data to the master json for each champion in that game to reflect each champion's performance in each game.

We noticed that 35,000 games worth of data resulted in a json that was quite large (87 MB), so we initially considered using a smaller subset of games (around 1,000 games), which yielded a master json file of only around 1.6 MB, which seemed much more manageable for our project. Back at that time, we were gathering quite a bit of data from each game that was played, to the point

that it would be impossible to neatly include all the data in our visualization. Instead of using a smaller dataset (1000 games instead of 35,000), we decided to gather less data from each games, but still use all 35,000 games worth of data. We chose this method because it gave use a large enough sample to feel like our findings accurately represented all Ultra Rapid Fire games played.

Exploratory Data Analysis: What visualizations did you use to initially look at your data? What insights did you gain? How did these insights inform your design?

We initially considered using a central visualization that took the form of a bubble chart, in which each champion was displayed as a bubble. The y-axis was average win-rate of the champion, the x-axis was the average game duration for the champion, and the size of the bubble corresponded to the popularity of the champion. While this gave us an interesting chart and answered some of our main questions, it didn't lend itself well to being an interactive visualization, and was mostly static. This showed us that we needed to come up with a design that would be very dynamic as we clicked on different champions and other various elements of our visualization and led us to the direction we are now planning to take.

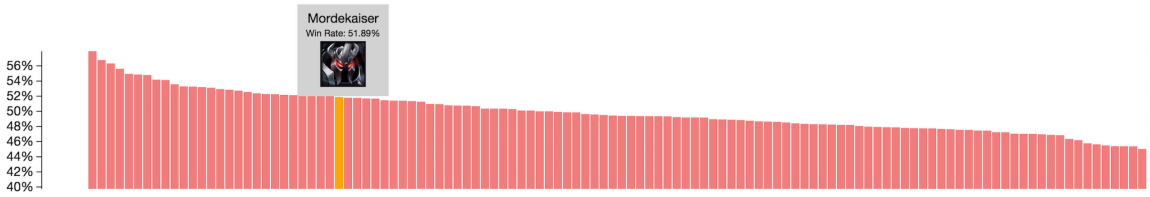
Design Evolution: What are the different visualizations you considered? Justify the design decisions you made using the perceptual and design principles you learned in the course.

We were originally going to implement a bubble chart that was static and represented each champions win/lose rate. However, we quickly realized that this might cause problems since it was possible that bubbles would overlap. To fix this problem, we decided instead to change the bubble chart to a force layout chart. This way, overlapping wouldn't be a problem, and we could change this chart to become a cool display for filtered champions at the same time. We were very pleased with how this force layout turned out, as we had not seen similar visual effects implemented in other projects.

However, we wanted to be able to filter all possible metrics in our visualization by game duration to figure out how champion and item balance shifted from shorter games to longer games. Since champion win rate was encoded into our bubble size and win rate per champion changes based on game duration, we would have needed to dynamically resize our bubbles and the images of the champions within them. Dynamically resizing 124 images for every tick of the game duration brush takes a lot of processing power and slowed our visualization down to levels we were uncomfortable with. For this reason, we decided to change our main visualization to encode champion popularity into the bubble size. This meant the bubbles and images would not require resizing because champion popularity is a static field that is not affected by game duration.

We still wanted a way to show win rates of all champions at the same time, and decided to use a bar chart. The bar chart allows the user to easily see

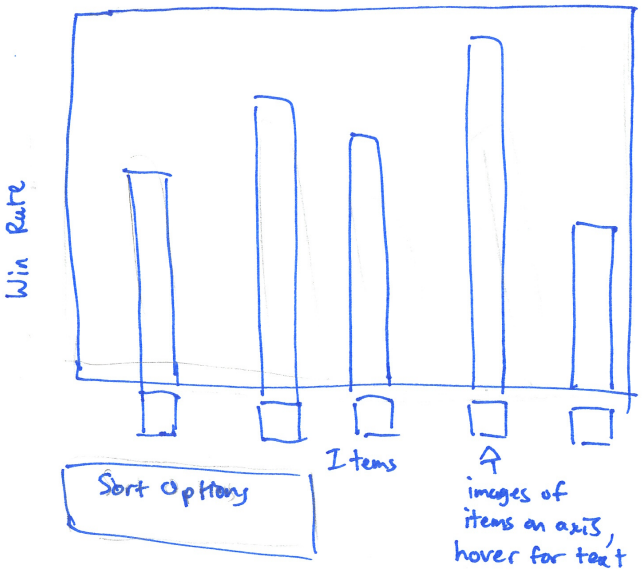
the win rate of each champion for the selected group of game durations. The only minor downside is that it is not immediately obvious which champion is which since putting 124 labels on a bar chart is impractical, but we implemented custom tooltips to show the champion's name, image, and win rate upon hovering over its bar. The following is a screenshot of using the win rate bar chart to determine win rate information about a specific champion:

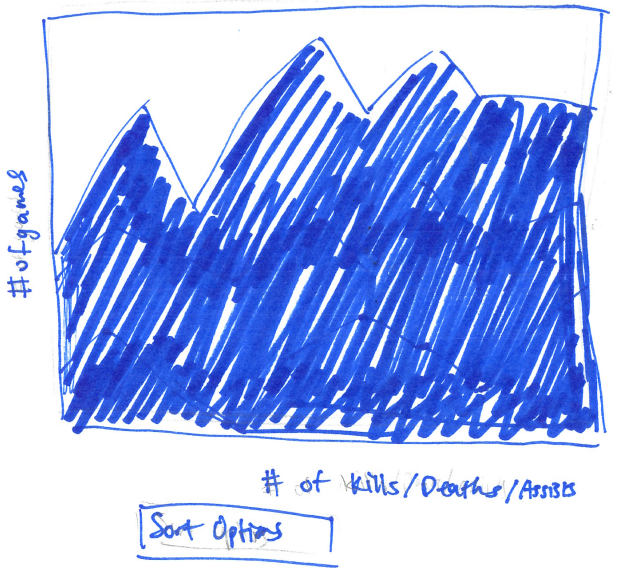


To preserve the option of selecting champions based on their win rate, we decided to allow users to click on champions in either the popularity-based force bubble layout or in the win-rate-based bar chart. Selecting a champion in either chart selects the champion in both charts, and brings their information up in a separate champion profile.

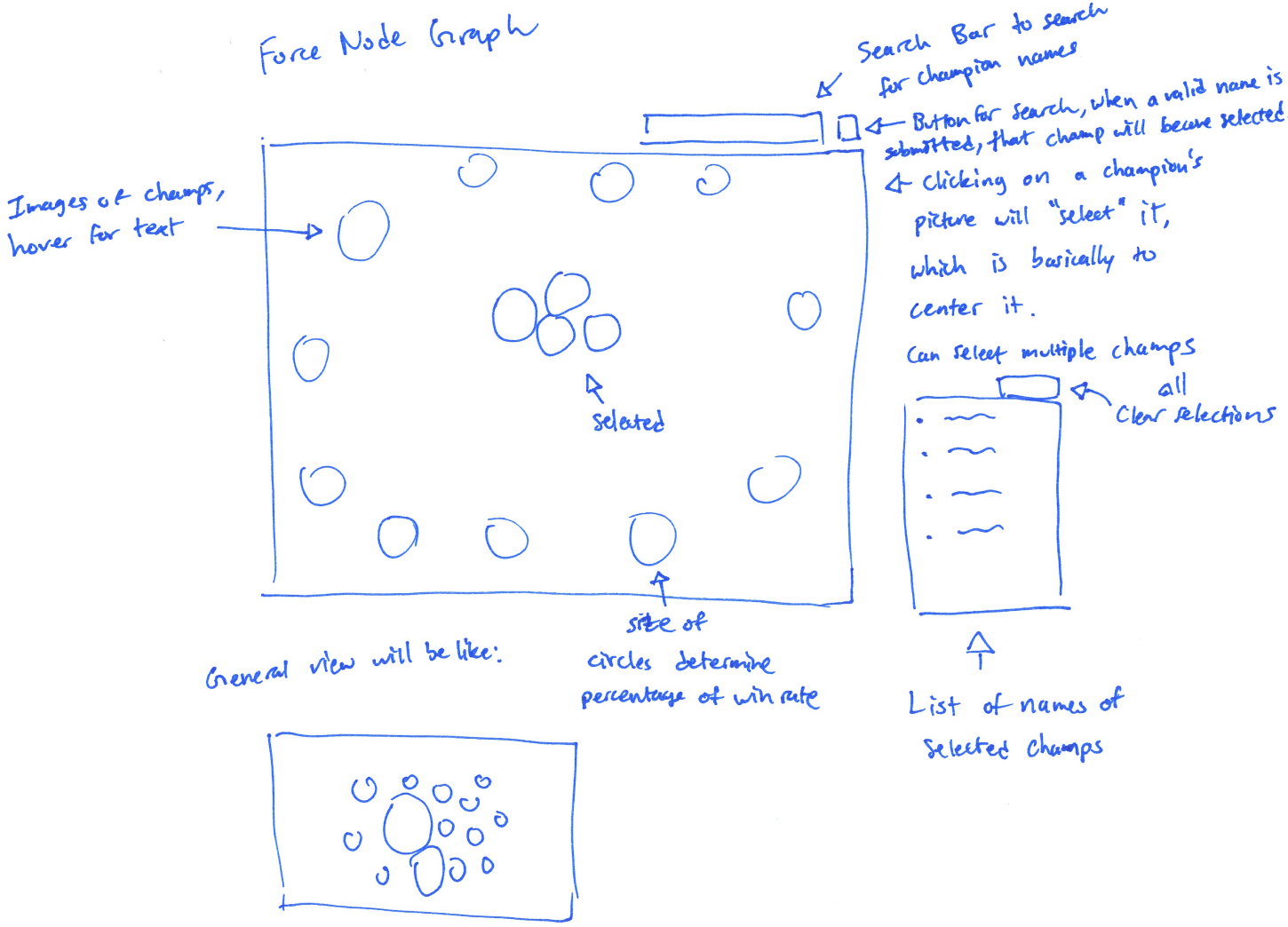
The idea of including small rectangular champion profiles arose from wanting to allow the user to see more information about each selected champion. In this class, we learned that the user's eye should be drawn to the most interesting parts of the visualization. Consequently, we decided to include the bright and flashy item/champion images in these champion profiles, since we felt they were important and deserved the user's attention.

Explanation of graphs here.





Force Node Graph

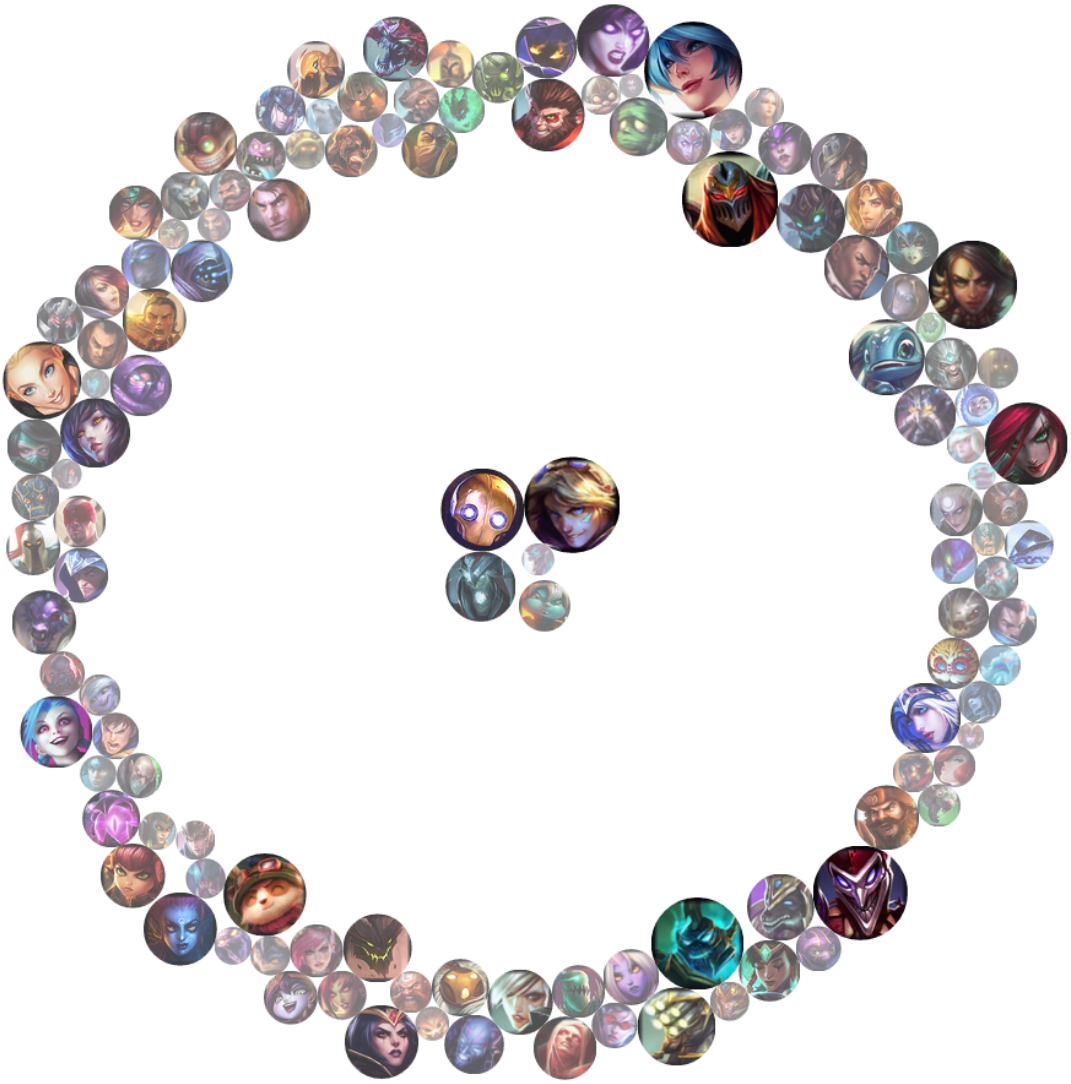


Implementation: Describe the intent and functionality of the interactive visualizations you implemented. Provide clear and well-referenced images showing the key design and interaction elements.

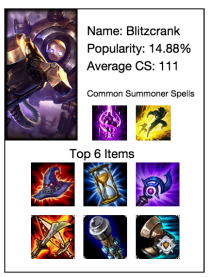
The main chart of our visualization is the force bubble layout in which champion popularity is encoded in bubble size and opacity. A screenshot of this layout can be seen below:

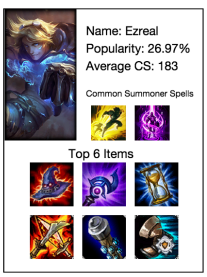
When any of the bubbles is selected, that bubble moves to the center of the layout and the other bubbles a repelled away from it like so:

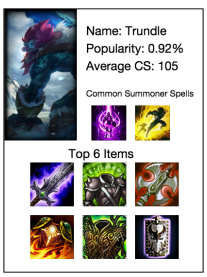
Up to five bubbles can be selected, after which any additional bubbles will replace bubbles that are already in the center, following a LIFO replacement order:

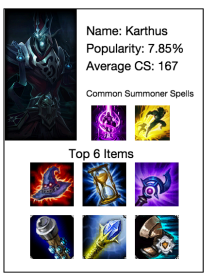


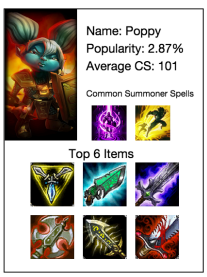
Selecting champions in this layout will bring up their champion profiles, as well as highlight their bars in the win-rate bar chart as seen below:



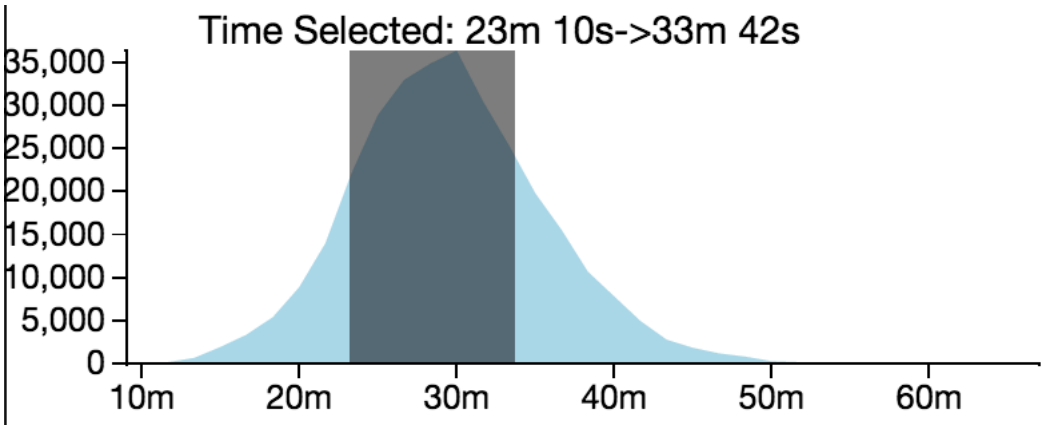






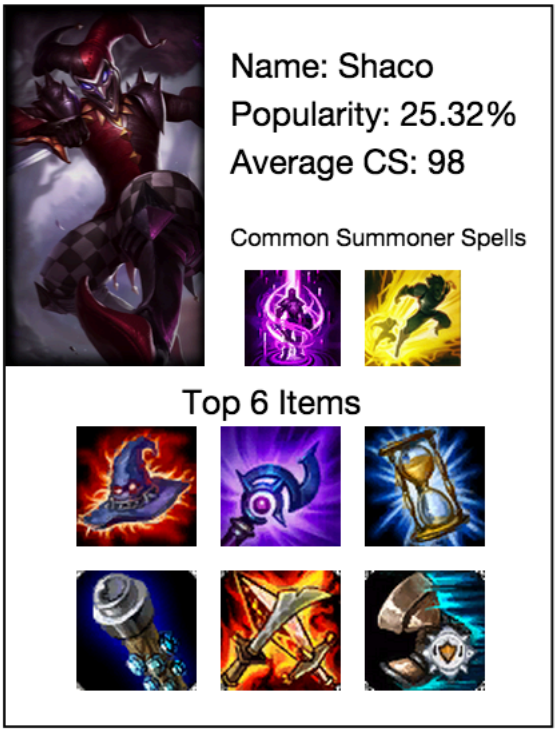


Alternatively, champions can be selected by clicking on their respective bars in the win-rate visualization. Our visualization also includes an area chart showing the distribution of game durations for all games played. This layout is interactive and can be brushed to filter the rest of the visualization by a specified group of game durations as seen below:



Filtering by a game duration interval changes the data shown in all other charts/graphs in our visualization except for the population-based bubble force layout, since it doesn't make sense to filter popularity based game duration.

The champion profile visualization holds up to five champion profiles that show statistics of champions selected in other areas of our visualization. Each profile has the following basic appearance:



Each champion profile contains fields for the champion's picture, name, popularity, average creep score, most common summoner spells, and 6 most

popular items. The average creep score, most common summoner spells, and top 6 items changed based on the user's interaction with the game duration brushing layout, and it can be interesting to see that items that are popular on a given champion in short games may not be nearly as popular on that same champion in long games. The champion profiles themselves are generated when that champion is selected either in the force bubble layout or the win rate bar chart layout.

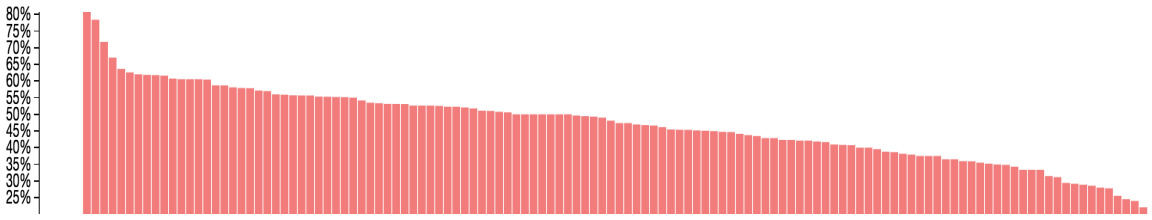
Altogether, these various elements form an engaging, interactive visualization from which a user can glean much valuable information about optimal strategies for Ultra Rapid Fire mode.

Evaluation: What did you learn about the data by using your visualizations? How did you answer your questions? How well does your visualization work, and how could you further improve it?

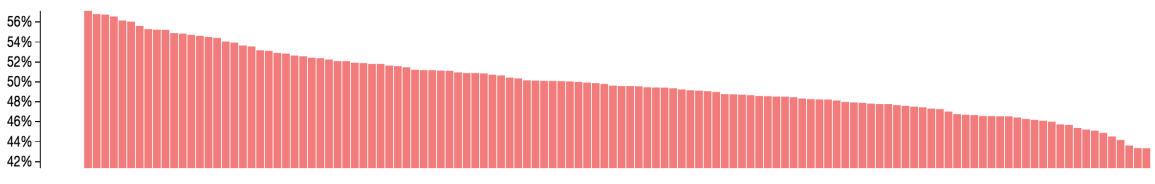
We came to some really interesting conclusions from using our visualization. One of our main goals coming into this was to figure out how well champion win rates lined up with how often champions were played. One would imagine that the most popular/frequently played champions would have the highest win rates and vice versa, thus explaining why people wanted to play or stay away from those champions.

We found that this was not the case, meaning that the majority of League of Legends players didn't do a good job of figuring out which champions were the strongest in this new game mode. For example, one of the most popular champions during URF was Nidalee, who was picked by 19.46% of teams that were able to pick her. A 20% pick rate is astoundingly high, but it turned out that her win rate was only 47.91% for the 35,000 games we looked at. On the other hand, one champion, Galio, had a very low pick rate of 2.32%, but an overall win rate of 56.75%. In games under 20 minutes, Galio's win rate shot up to a whopping 80.49%, which is completely unheard of. By highlighting Galio's early game strength in contrast with his low popularity, we accomplished our goal of identifying a strong sleeper pick to use in Ultra Rapid Fire mode.

Another interesting trend that we stumbled across while analyzing our visualization was that early-game champions were much more dominant than late-game champions. The following screenshot shows the win-rate distribution of champions in games that lasted less than 20 minutes:



And the following screenshot shows the win-rate distribution of champions in games that lasted more than 30 minutes:



We can easily see that there are a few champions that completely dominate short games, while there are no champions in longer games that have such a clear advantage. For example, in the sub-20-minute games, Galio, Sona, and Zyra all have win rates of above 70%, while in the over-30-minute games, the champion with the highest win-rate, Fiora, only has a win rate of 57.13%. These findings suggest the best way to ensure victory in an Ultra Rapid Fire games is to choose a team of 5 champions that excel in the early game, then try to end the game in 20 minutes or less.

Questions about optimal item builds for specific champions can be answered on a case-by-case basis, as the champion profiles show the most popular 6 items for each selected champion. This feature proved to be a useful guide for newer players who are overwhelmed and not sure what items work best in this new game mode.

As far as improvements, given more time, we might have tried to come up with some fancier transitions within our various charts/graphs, but we're very satisfied with how it turned out. We might also consider gathering data from normal games to compare to our Ultra Rapid Fire findings to further highlight the differences the two game modes.