Predicting Win/Loss Records using Starcraft 2 Replay Data

Final Project, Team 31

Evan Cox
Stanford University
evancox@stanford.edu

Snir Kodesh
Stanford University
snirk@stanford.edu

Dan Preston
Stanford University
dpreston@stanford.edu

1. INTRODUCTION

Our project involves exploring the features of the Starcraft 2 match graph. Starcraft 2 is a competitive popular "Real Time Strategy" (RTS) game. These types of games do not incur turns and are done "live" such that each player is essentially managing his/her team in real-time with others. Players start with an initial collection of economic units and "town center," and over the course of the game manage their economy, army units, and available technology to best their opponent by destroying all of the opponents buildings before their own are destroyed. After games are played, a replay is saved detailing pertinent information about the game, such as who participated, who won, who had a better economy score, technology score, army score, etc.

The multiplayer aspect of Starcraft 2 is an extremely popular feature of the game. Players can either choose the players they play against in custom games, or can try to improve their ranking by playing ladder games. Players have rankings which either decrease or increase with losses or wins respectively in ladder games. The goal of our project is to predict wins/losses between any two given players that participate in Starcraft 2. To accomplish this goal we can formulate the win/loss prediction problem as a variant of the canonical link prediction problem in social networks.

As a result of our analysis, we developed an intuition for the nuances and difficulties of the project. It is a three-part story: First, global features provide context surrounding the overall ability of a player and provides essentially a prior for his/her ability. Second, local features provide importance when examining a hypothetical game, most importantly when two players have similar global "rankings". Finally, although the network is sparse, we cannot limit our analysis to a particular level of minimum embeddedness, as the context provided by the "outlier" edges (i.e., those with no shared nodes) proved to be critical in analyzing the network.

This work consists of four primary parts. First, we outline our methodology for gathering and synthesizing the data. This includes gathering game-specific data from multiple sites, and tying this together with general data about players collected from a "master" site. Second, we perform some top-level statistical analysis, analyze structural properties involving the network, and present some of the challenges that appeared during analysis. Third, we examine several different methods using global and local features for win prediction and their resulting performance. Lastly, we explore possible directions for future research.

2. DATA COLLECTION

The synthesis of our data required more overhead than querying a database or parsing a single flat file. Our initial plan was to contact Blizzard in the hopes that they would provide us with some pre-packaged data. That turned out to be fruitless, and we turned to building our system to harvest SC2 data. We identified several repositories for user-uploaded content, in the form of *.sc2replay* files. These files store proprietary Starcraft 2 "replay" data, and includes many of the properties and facts about the game. We began by first building a custom scraper for each site, in addition to one for the main StarCraft2 UID resource (which indexes all UID's and relevant statistics for that player, without the granularity of a per-game basis). We then examine the DOM and determined what pathway to traverse in order to arrive at a downloadable file, and thus we were able to construct a large list of direct pointers to replay files. Subsequently, we used PHP SC2Replay1.30 (*http://code.google.com/p/phpsc2replay/*) to parse out relevant data from each individual file .

Our data was originally parsed from Starcraft 2 replay sites *gamereplays.org/starcraft2* and *sc2.replayers.com*. Unfortunately some of the *.sc2replay* format is unknown and changes as the game is updated with subsequent patches. As a result we could only use a subset of the replays on each site for our data, as a result of the problems and deficiencies of the PHP SC2replay 1.30 parser. Additionally since we were measuring player to player (1v1) relationships, we excluded replays that were of games where more than 2 people were participating. Of the 3,500 replays on *sc2.replayers.com* we managed to successfully parse and extract about 2,500 1v1 replays. Of the 24,000 on *gamereplays.org/starcraft2* we managed to successfully parse and extract about 12,000 1v1 replays.

The replays contain rich data, that we were able to extract by parsing the files. This included the game duration in seconds, the server region the game was played on (match servers are region locked), average actions per minutes (keys pressed / button clicks), the winner, each players chosen race (there are 3 possible races in Starcraft), each player's unique identifier (UID), and the map the game was played on. Using the UID we were able to extract further information by crawling a site, *sc2ranks.com*, about individual players. *sc2ranks.com* contains rankings of the 1.5 million Starcraft 2 players that have played a game online. The site contains rich information about players such as their name, the league they play in (Bronze, Silver, Gold, Platinum, Diamond), their total number of points, total wins, total losses and win percentage. Leagues denote overall skill level whereas points within a league give a more granular account of a player's relative ranking. Players swap between leagues by ascending to the top of their league and remaining there for an unspecified amount of time. As a result we were able to extract replays which contained latent information regarding the in game performance of players, as well as other information about the game, as well as extract metadata concerning the players, to form a more complete picture of the relationship between a player-to-player matchup.

3. STRUCTURAL ANALYSIS

In this section, we explore some of the interesting properties of the Starcraft 2 network, including the degree distribution, win percentages and points associated with players. Additionally, we are interested in the breakdown of different leagues and races, relative to their win percentages and player performance. Finally, based on these plots, we hope to gain an understanding of some of the basic dynamics of the network: what biases may exist in the data, and what we can infer from the network in general.

The following analysis assumes we create a directed graph, where a node represents a player and a directed edge $A \rightarrow B$ means that player B beat player A. If there are multiple games played between two players, the player who has won more against his/her opponent achieves an incoming edge. Thus, an incoming edge indicates that the player has an overall winning record against the opponent.

To begin, we examine the log-log degree distribution of the network in Figure 1. Note that degree is defined as the total number of incoming and outgoing edges from a node (i.e., how many games a player has played). Closely tied to the degree of a node, we will explore what it means to be a good player versus a popular player later in this work. It is clear the distribution follows a power law, noted by the negative linear slope in the log-log plot. This has two possible interpretations: 1) Great players stay great players, and thus win the majority of the games (a "rich get richer" dynamic), 2) certain players are more popular and thus get their replays submitted to the website more often than others. Also, it interesting to note that using the estimation in Clauset et al., we obtain $\alpha = 2.16$ for our power law distribution.

Additionally, it is interesting to examine other properties as they relate the the degree of nodes. For example, Figure 2 shows win percentage as a function of the node degree. This shows that, in general, the average number of wins for every-

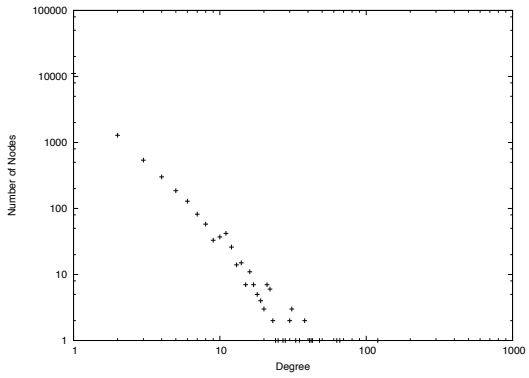

Figure 1: Log-log plot of the degree distribution

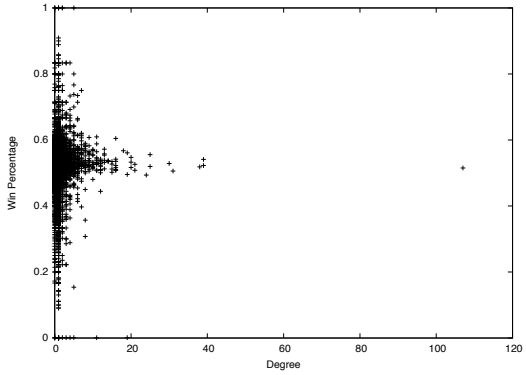

Figure 2: Degree vs Win Percentage

one is around 50%. There is variance around smaller number of degrees, but as the number of degrees grows larger, the win percentage actually starts to converge toward 50%. This is quite interesting in the context of our problem, such that the best players may not have the best win percentage, but should be examined using other factors (such as comparing them against other good players). Finally, it should be noted that there is a heavier distribution of players with sub-50% records with very low degrees, suggesting that very young players likely lose more often.

We examined another aspect of win percentage: What if we run PageRank [2] on the graph, and compare the PageRank to the win percentage? In this way, we may be able to compare what the network structure indicates (via PageRank) versus aggregate data about players (win percentage). Thus, in Figure 3, we see this comparison. On the X-axis we show the PageRank (lower is better), and on the Y-axis we show the win percentage. This, again, shows that win percentage is likely a poor indicator of success. On the other hand, one can notice a slight increase in win percentage for the very highest ranked players in the network. Thus, there may be some correlation between their "influence" in the network, and their win percentage.

Aside from degree distributions, there are other interesting properties associated with this data. For example, Table 1 shows the average PageRank for each of the leagues (where

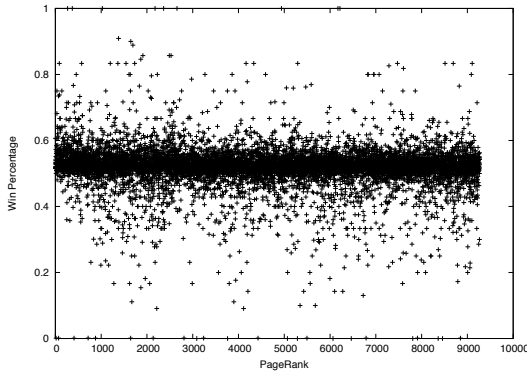

Figure 3: PageRank vs Win Percentage

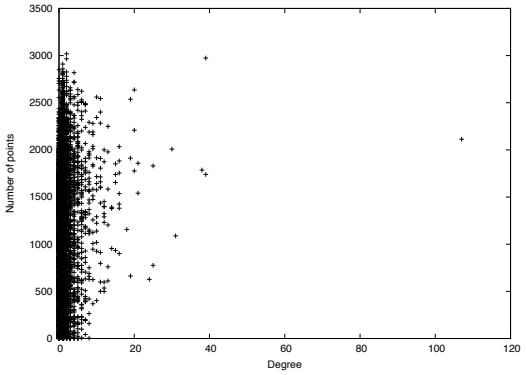

Figure 4: Degree vs Player Points

Table 1: Average PageRank for each league

League	Avg PageRank
Bronze	4788.42
Silver	4844.13
Gold	4558.52
Platinum	4607.49
Diamond	4575.04

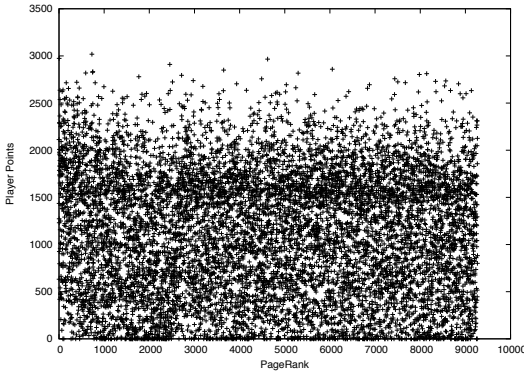

Figure 5: PageRank vs Player Points

lower PageRank is better). Note the leagues are in ascending order of height; in other words Bronze is worst and Diamond is best. It is interesting to note that while generally the better leagues have better ranks, Gold outperforms Diamond and Bronze outperforms Silver. Ideally, this would not occur, but it is not clear where the deficiency lies: Is the Starcraft 2 league system flawed? Perhaps different leagues are harder and there is not enough activity between different leagues? These questions are an important next step in the analysis, and will be included in the final report.

We plot the number of player points as a function of degree in Figure 4. This graph shows that the number of points associated with a player does not necessarily correlate with the number of games he/she played. Finally, we examine Figure 5, which plots the number of player points as a function of the PageRank for the player. This indicates that points correlate very little with the PageRank. If, in fact, either of these prove to be good indicators of the player's ability to win a game, this will then suggest the other metric is of little use to the analyst. In the following section, we will show that PageRank ends up proving to be a successful indicator of success, and furthermore enhances our hypothesis that the number of player points is not a good predictor for success in a game.

4. WIN PREDICTION

In the previous section, we attempted to find structure regarding degree, win percentage, PageRank, and other characteristics. Unfortunately, there is no clear feature that provides separation of nodes (players). In this section, we first explore heuristic features to attempt win prediction, followed by an exploration of global features (e.g., PageRank) to predict wins. Then, we explore methods for examining the local relationships between two players in order to predict a winner. Using these two results, we present our discovery: a clear winner using a hybrid model. Finally, we present our results using a common method when dealing with sparse networks, a particular difficulty in our network.

4.1 Heuristics and Global Network Features

Below, we describe several methods that compare either a single feature about the two players, the game features, or global characteristics. Table 2 summarizes the results of the experiments.

Random Used for a baseline, the *random* algorithm randomly selects a player to win the game. Of course, this is expected to be roughly 50%.

LeagueRank When Player A faces Player B, we choose the winner based on the highest league ranking (according to Bronze, Silver, ..., Diamond). If both players are from the same league, then the number of points are chosen to differentiate the players.

WinPctRank Players A and B are compared as a function of the overall win performance, discounting all other factors. If they have the same win percentage, the algorithm defaults to the number of points.

PointsRank Players A and B are compared as a function of the number of points the player has received on Battle.Net. The calculation of this value is proprietary, but it is used to

evaluate players within the system to determine ranking.

DegreeRank Players A and B are compared as a function only of the node's degree. If the degree is equal, the algorithm reverts to win percentage, then to the number of points.

PageRank [2] Using the structure defined in Section 3, we gather the PageRank values for the network. From these values, we then select the player with the highest PageRank value as the winner. Note that in this methodology, we tested over each instances, and ensured that we removed the current game's edge between the two nodes before running PageRank. In this way, the network contains no context about the game for which we are predicting.

SVM+Game Features In this method, we learn an SVM model using global features associated with the game (version, map, duration, location), player details (league, number of points, win-percentage), and game-specific player details (actions-per-minute, race). From here, we then train and test using 5-fold cross validation to obtain the accuracy of our model. Finally, we attempt 3 different SVM kernels for completeness, in order to set an appropriate baseline comparison.

Table 2 summarizes the results of these algorithms. Most of the associated algorithms win out over the *Random* baseline comparison. In particular, the ranking algorithms perform the best. Specifically, when we choose the highest *PageRank* player, we obtain the highest percentage accuracy. This indicates that the network structure of this data is crucial to understanding who the best players are in the network. In some respects, this is to be expected, as the number of wins is not necessarily the best indicator of future performance. On the other hand, if one has beat many other players with many wins, then they are likely to be a good player (which is precisely the dynamic for which PageRank is attempting to capture).

There are a few key messages to take away from these analyses. First, league rank and win percentage are poor indicators of future performance, which is likely due to the way in which players climb ladders and play more challenging opponents as they play more often. In addition, points have very little relevance to a player's abilities, as shown by the lowest heuristic score in the table. Finally, and most interestingly, the degree of the node is not a good indicator of a player's performance. This suggests two major things: 1) players are not self-selecting, in that the players who appear most are not necessarily the best and most likely to be winners, and 2) there are many nodes that have low degree. The former point is crucial, as we can make stronger assumptions about the network, such that the importance of "fringe" players is not to be discounted and will aid in a global feature classifier. Specifically, this indicates that a popular player *is not* necessarily a great player.

4.2 Local Features

While PageRank is a clever utilization of the global network structure, we were interested in analyzing how the local network structure surrounding a given pair of nodes can be utilized to improve prediction. PageRank lends itself well to

Table 2: Comparison of heuristic and global feature win-prediction algorithms

Algorithm	Accuracy
Random	0.500
LeagueRank	0.556
WinPctRank	0.561
PointsRank	0.549
DegreeRank	0.592
PageRank	**0.705**
SVM (Linear Kernel)	0.497
SVM (Polynomial Kernel)	0.533
SVM (RBF Kernel)	0.512

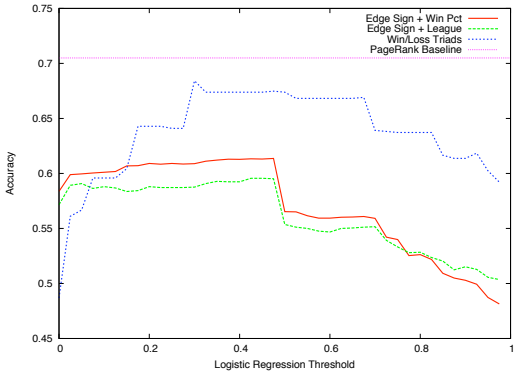

Figure 6: Local feature methods, compared with best heuristic method (PageRank)

the win/loss semantics of giving losers giving weight to the winners, but local network features allows for a more nuanced analysis of by looking exclusively at the games played by the given players, while ignoring large, potentially irrelevant portions of the network.

The local features we used were adapted from the "Balance" theoretic features shown in Leskovec et. al [1]. A balance feature's basic unit is the triad. If w is a common neighbor of u and v, the u, v, and w form a triad. Using triads, we can characterize the relationships between two nodes by their relationships that they have with their common neighbors.

Balance features posit statements such as, "The enemy of my friend is my enemy." Here a positive edge represents friendship, and a negative edges represent antagonistic relationships. Letting a positive relationships value = 1, and a negative relationships value = -1, with nodes u, v, and common neighbor w, that $sign(u, v) = sign(u, w) * sign(v, w)$.

In our work we incorporate several different semantics of sign and direction. Additionally, we include the different triad types as feature inputs to our classifier, in addition to balance.

Below, the methods are described. In each case, we use a basic logistic regression model with the features described. Figure 6 compares each method, where the X-axis is the threshold used to separate positive and negative edges and the Y-axis is the accuracy given that threshold.

Edge Sign + Win Percentage In this method, we use an undirected graph. Each edge represents a game, and each edge has a sign: If the player that *won* has a higher winning percentage than the player that *lost*, we give the edge a positive (+) sign. Otherwise, if the player that lost has a higher winning percentage, we give the edge a negative (−) sign. Finally, for each edge, we create 9 total features:

- Total incoming negative edges for each node (2 features)
- Total incoming positive edges for each node (2 features)
- Total number of common neighbors (1 feature)
- Counts for each triad type: For each common neighbor w of nodes u and v, count the number of times we see +/+, +/-, -/+, -/-. (4 features)

Edge Sign + League Exactly the same formulation as *Edge Sign + Win Percentage*, but instead of using Win Percentage as the criteria for choosing sign, we use the player with the higher League ranking. Note, if the player is in the same league, we revert to win percentage.

Win/Loss Triads Instead of using edge sign, we consider the original directed graph we used when performing heuristic tests. Thus, it will be a slightly similar formulation as the edge sign features, but we will consider win/loss comparisons instead. Thus, we have 9 features:

- Total losses for each node (2 features)
- Total wins for each node (2 features)
- Total number of common neighbors (1 feature)
- Counts for each triad type: For each common neighbor w of nodes u and v, count the number of times we see W/W, W/L, L/W, L/L. (4 features)

When examining the comparison of these methods, we notice that the winner is the final method, *WL_Triads* (excluding the PageRank baseline). It has a peak accuracy of 0.687, and outperforms the other edge sign techniques. The reason is likely intuitive to a practitioner: as we saw from the structural analysis of the network, win percentage and league are poor indicators of talent and thus because we chose to use them as our criteria for choosing the sign, we were unlikely to get a strong classification.

These methods are purely local methods, that consider only a two-step approach from each pair of nodes for each edge. Unfortunately, none of these methods were able to beat the PageRank global feature heuristic tested above. On the other hand, it did perform significantly better than random (0.687 vs 0.50), and thus we should not discount the importance of local features in this network.

One idea for improvement would be to combine the two methods, and predict wins using both the local and global features. In this way, we may be able to improve the overall accuracy. In fact, this is exactly what occurred. Figure 7 compares each of the local feature-based algorithms, but includes a 10th feature: its PageRank value. As can be seen, the *WL_Triads + PageRank* algorithm was able to outperform all of the local algorithms and also the global PageRank baseline, with a peak accuracy of 0.781. In the end, this algorithm ended up being the clear winner among all algorithms attempted in these experiments.

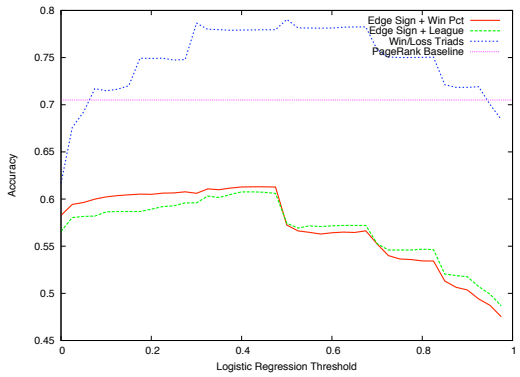

Figure 7: Hybrid feature methods, compared with best heuristic method (PageRank)

Why do the features that count the combinations of W/W, W/L, L/W, L/L provide insight? First, we can analyze this quantitatively. In the winning model, *WL_Triads + PageRank*, each of the triad features had the coefficients that appear in Table 3 (averaged over all tests).

Table 3: Average coefficients for triads in *WL_Triads + PageRank* model (logistic regression model)

Feature	Coefficient
W/W	0.0053
W/L	0.1921
L/W	0.1401
L/L	1.532×10^{-6}

where the coefficients are gathered from a generic logistic regression model and normalized to 1. It is striking that the two features W/L and L/W are weighted highest. This is likely to lead from the intuition that when Player A has lost to Player C in the past, and Player B has won against Player C in the past, then Player B is likely to win against Player A. The coefficient data seems to reflect this, and in fact when two players have similar records against the same player, it suggests further that it provides little information to the analyst.

4.3 Sparse Network Challenges

One of the challenges in this network is that it is quite sparse. For example, there are only 784 edges with nodes that share other nodes. Furthermore, the graph density is 6.714×10^{-6} using the formula:

$$density = \frac{m}{n(n-1)} \tag{1}$$

where m is the number of edges and n is the number of nodes. To provide some intuition on the values, 0.0 would mean no edges, and 1.0 would be a complete graph. Thus, to alleviate this problem, we examined the network where we only included edges with a *minimum embeddedness* of 1. In other words, all edges in which there were no other shared nodes (i.e., no triads) were removed. In this case, we had 784 edges, and a graph density of 0.00706.

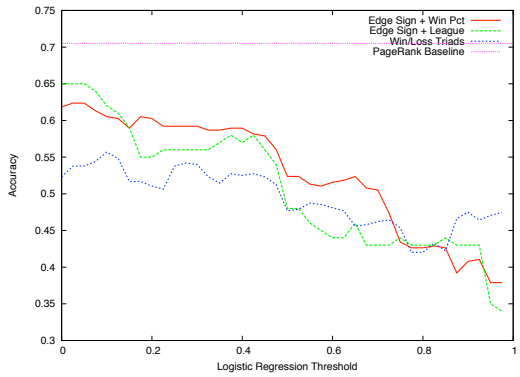

Figure 8: Local feature methods, compared with best heuristic method (PageRank). Network is pruned to include only edges with minimum embeddedness of 1.

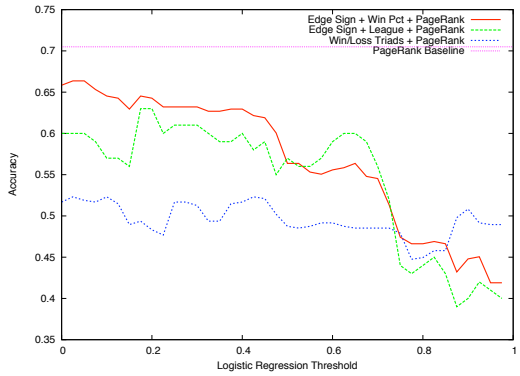

Figure 9: Hybrid feature methods, compared with best heuristic method (PageRank). Network is pruned to include only edges with minimum embeddedness of 1.

We replicated the experiments from the previous sections with our new network, as shown in Figure 8 (local features only) and Figure 9 (hybrid features). The purple line is provided as a baseline, which represents the heuristic PageRank value, built using the entire network, but testing only the edges in the new pruned graph. This allows us to see the impact of losing the context of all of the sparse edges in the network.

Because we have removed all edges that have no local context, it could be intuitive to expect that accuracies would increase if we only examined the subset of nodes with a minimum embeddedness. In other words, because we are largely guessing at random for those nodes without any local context (i.e., zero embeddedness), we are likely to improve when only considering high-signal edges. Unfortunately, this was not the case in our experiments, as we see the accuracy for the same method drops significantly. We suspect that, in general, this is caused by a lack of overall global information that was originally provided by the PageRank feature. In other words, because we have large amounts of knowledge about winning against other players, even if the number of edges are not dense, the mass gathered through a PageRank iteration is significant when comparing two instances.

5. FUTURE WORK

One of the shortcomings of our project was that the amount of data we collected with respect to the true size of the network is miniscule. With 40,000 replays, this represents an extremely small fraction of the network that could be explored. A natural step to remedy this problem would be to partner with Blizzard Entertainment, the makers of Starcraft 2 obtain access to the entire database, as they were unresponsive to our initial requests.

Additionally, our parser provided more information that we were currently using. For example, the parser can extract the units/buildings produced by given players, and the order in which they were produced. This "Build Order" gives insight into a given player's style, and also is an implicit time series. Using given time series clustering methods or similarity metrics we could cluster players according to their play style, obtaining an analysis of what play styles tend to work well against other play styles.

Moreover, the parser gives us access to what region the match was played on. Matches in Starcraft 2 are region locked, a player on the North American server cannot play a game against a player on the South Korean server. It would be interesting to analyze what strategies are popular on different networks, and how strategies propagate throughout time. This would be doable, given that our parser can extract time stamps from the game, but would again, hinge on access to a more complete database of games for a true picture of strategy propagation.

Another item of interest that we can measure in our network is the strength of particular racial match-ups, and map match-ups. We can ask questions such as "Do lower ranked players of race X consistently beat higher ranked players of race Y?", and "Does race X win at a higher rate on map A against race Y?" Additionally the parser can extract richer data than we have presented here. For example, the parser can extract specific actions by players. For example, the parser can determine a "build order" that denotes what a player built an when. An example of this would be

"10 seconds, built a worker"

"24 seconds, built production facility X"

"36 seconds, built warrior type Y"

Using these build orders we can determine common strategies that players use in the network, and perhaps similarly, conventions used to counter particular strategies (for example, sacrificing your long-term economy for a burst military force with the hope that you can "rush" your opponent before they can establish their style of play). With these additional dimensions, we hope to incorporate the matter of in-game race (Terran, Protoss, and Zerg) to try and make accurate predictions with respect to certain in-game stages (economic score, natural expansions, etc), in addition to considering multiple players across multiple races (i.e. does Player X playing race X retain his advantage when playing Player

Y's race Y). For example, given player X with a certain economic establishment on race Y, what is the likelihood of victory against his opponent (real or imaginary) with an established economy of Z. Our hypothesis is that due to the high in-game balance (of races), along with specialization which players embark when choosing to play specific races, we will present interesting findings when introducing these additional dimensions.

Countering strategies may also arise, and a player's mastery of a particular countering strategy (bucketed by certain traits, such as military:economic spending), may introduce additional inputs to our prediction algorithm. We also expect to encounter similar findings when using several similar variables, such as map advantages. Here, we expect that top-level players will have a more well-rounded profile, performing fairly equivalently across map types (or alternatively, maybe just those maps that are used in sponsored tournaments). With that said, it would be interesting to see whether we can identify any behavioral patterns across certain maps–perhaps players favor a certain race or strategy depending on what opportunities the map presents (economic favoritism, militaristic advantages, etc), and subsequently, what handicap that enables in a 1v1 matchup. Using the parser identified above, we've discovered we can extract extremely granular data, which we intend to utilize more going forward in the next two weeks to extract some of these more subtle–but intriguing–characteristics of gameplay.

6. REFERENCES

[1] J. Leskovec, D. Huttenlocher, and J. Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, pages 641–650. ACM, 2010.

[2] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.