Starcraft Map Imbalance Prediction Based on Chosen Build Order

Andrew Spann, Eleanor Lin, and Hyunseung Kang

December 11, 2009

Abstract

We utilize the distribution of early building decisions on Starcraft maps to predict the winning percentages in each racial matchup. Tournament replays from 5661 games on 7 maps were parsed. The win rate on a given map was predicted from a regression on the remaining maps' build order distributions and logistic-space win rates. The method is only highly effective at predicting the outcomes for Zerg vs Protoss games. Our data set was found to be less biased than anticipated, showing that Starcraft is a balanced game despite a great diversity of strategy profiles.

1 Problem Definition

The computer game Starcraft has enjoyed such widespread commercial success that South Korea employs 12 professional Starcraft teams whose matches are broadcast on Korean television. In the game of Starcraft players choose to command the armies of one of three races: Terran, Zerg, or Protoss. Unlike chess pieces, which are the same for both sides, the army compositions of each Starcraft race are entirely unique. A game between two players of different races contains many asymmetries. Despite the many asymmetric factors influencing the game, Starcraft is considered to be a balanced game in that all three races have successful professional players. Starcraft games are played on many different "maps" which change the fundamental board layout. Maps differ from each other in attributes such as the ease of acquiring resources, distance between bases, and the openness of the terrain. Figure 1 gives an example of the characteristics of two commonly played Starcraft maps. In professional televised games, most maps favor one race's chances of winning over another by a $60\% - 40\%$ split [1]. The most common bias is a cyclical Terran $>$ Zerg $>$ Protoss $>$ Terran relationship that prevents one race from dominating the others. We wish to investigate whether the overall winning percentage in a match between two given races on a given map can be predicted from the early game decisions made by the players.

One of the most important decisions in Starcraft consists of choosing which buildings a player constructs. Buildings either allow the player to collect more income, to produce more units for the player's army, or to allow high tech units into the player's army. We consider only the construction of buildings and not individual troops, since the number of possible build orders quickly becomes combinatorially intractable otherwise. The important information we can learn from buildings lies in the order they are built. Attempting to define a game state based on the mere presence of buildings would not be valid (given that troops and money reserves are not accounted for in the state). The time at which a building is built matters greatly because this information conveys intention to either invest in long term resource gathering or raise an army quickly for a decisive all-in push. For example, a Terran player's first two buildings usually will be either a Supply Depot then a Barracks or a Barracks then a Supply Depot. The order matters greatly. Building the Supply Depot before the Barracks is more common and allows the main base to produce resource gathering units without ceasing. Building a Barracks before a Supply Depot gives faster access to military units but requires a pause in the production of resource gathering units and thus implies unusually early aggression. Most buildings have prerequisites, so the early build decisions are combinatorially limited and can be represented cleanly as a tree. We only need to concern ourselves with analyzing the earliest few buildings, since after the opening minutes of the game the player will have scouted the opponent and further decisions will be reactionary.

We hypothesize that a mixed strategy Nash equilibrium exists between the many possible opening build orders and that experienced players choose their opening builds accordingly as a function of the map characteristics. We expect this condition to hold at high levels of play because if it were not true then there should exist a strategy profile that beats the current top players, and any player using this strategy profile would then become a top player. Initially, the player does not have any information on the opponent's actions, so in the field of game theory one could compare Starcraft's early building decisions to

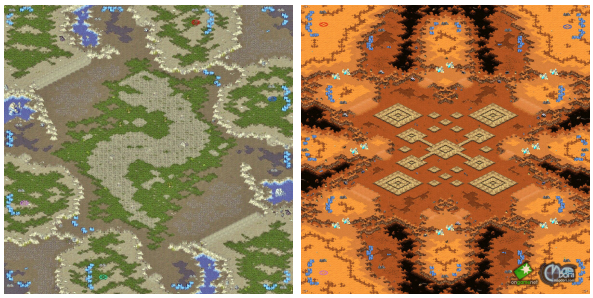

Figure 1: Left: Python, a "standard" map with nearby expansions and an open middle. Right: Troy, a tricky map whose connectivity can change when neutral buildings are destroyed. Source: [1]

Map	Total Games	TvZ	ZvP	PvT
Blue Storm	1354	446	593	315
Gaia	694	238	328	128
Katrina	391	114	156	121
Python	2248	608	922	718
Tau Cross	439	89	252	98
Troy	356	89	113	154
Zodiac	179	52	71	56

Table 1: Number of replays in data set for each map. Races abbreviated as $T = Terran$, $Z = Zerg$, $P = Protoss$.

a probabilistic simultaneous decision game. Guided by this intuition we test the claim that the overall win rate of a race on a map can be expressed as a function of the distribution of opening build orders, so that maps with similar winning percentages exhibit similar distributions of build orders.

2 Approach

Starcraft allows players to record replays of games for future viewing. To compare the win rate and the build order distribution, we collected approximately six thousand replays and used a computer to analyze them. This section details the process of obtaining and parsing the data. Details of the learning techniques applied to the data are described in Section 3.

Note that we consider data for each of the three asymmetric racial matchups differently. For example, our analyses of Zerg vs Terran matches and Zerg vs Protoss matches are kept completely separate since we expect inherently different build order distributions from the Zerg player in those two matchups.

2.1 Obtaining the Data Set

Data was taken from the ladder stage replay packs of the top 48 playoff finalists in the 2008 Teamliquid Starleague tournament. An archive was downloaded containing the finalists' 10400 automatically collected replays over the course of the three week ladder [2]. Replay files deemed to be invalid due to not being 1v1 games or being under 2 minutes (indicative of an internet disconnection) were discarded. Mirror matchups (*i.e.* Protoss vs Protoss) were not considered. Of the remaining files, 7 maps were considered to have sufficient data, a total of 5661 replays. The number of replays for each map and matchup is summarized in Table 1. The map Longinus had 837 additional replays, but was not considered due to our parsing techniques failing on those replays.

2.2 Analyzing Replay Files

We need to parse the replay files to determine who won each game and the buildings built by both players. Replay files contain neither video recordings of the games nor copies of the game's state at different time periods. Instead, the replays are a history of all the users' mouse clicks and keyboard actions. Starcraft uses these recorded actions to deterministically recreate the course of the game. The amount of money each player has and the deaths of units and buildings are the results of actions, not actions themselves, and thus not explicitly listed.

The replay file does not consist of plain text, but its binary format can be parsed with a C++ library [3]. To determine the winner of the game, we first identify which player recorded the replay file. If the game contains a "CMD_LEAVEGAME" action from the player's opponent, the player recording the file is considered to have won, otherwise this player lost. Attempting to compute the last player to make an action will not determine the winner as this does not uniquely determine whether the replay file was recorded on the winner's or loser's computer. The C++ library is a fan project that is not officially supported Starcraft's creators Blizzard. Starcraft has undergone several patches since the replay library was released, so parts of the library appear to be broken. Even if the data we wish to collect is conceptually straightforward, the C++ library functions frequently encountered pointer errors that hindered our ability to do a deeper analysis.

The C++ library comes with a standalone statistics tool called BWChart [3] that already contains functionality for tabulating statistics in a collection of game replays. Unfortunately, BWChart is not open source so we cannot modify it to collect data and export in the exact format we want. However, the functionality for computing build order was still

useful. We define "build order" to be an ordered list of the first n objects that a player builds. We must now determine the value of n and which objects are informative. Displaying the build order for a game is straightforward, but knowing which parts of the build order are relevant requires making a decision. Technically a player's army units, buildings, and upgrades could all be considered part of a build order. Since the number of entries in Table 1 is finite, including unit information into the build order would lead to a distribution of all unique build order instances yielding no information.

Using our intuition of the game, we can safely discard all information on unit production to restrict the number of build orders to a feasible amount. Units are usually built continuously with any available funds, so their specific order is implied by the existing buildings. We decided to look at the first 3 buildings built, and performed the analysis again using the first 2 buildings not counting Supply Depots or Pylons. Due to the way BWChart parses the data, Zerg Overlords are considered units instead of supply buildings and thus not counted. Attempting to use more buildings has several disadvantages: a long tail of infrequent builds is present, some games end before enough buildings are built, and by this time players will have scouted their opponents and begun making building decisions that are reactionary instead of simultaneous.

3 Predicting Win Rates

The number of times each build order was used on each map in each racial matchup was tabulated and normalized. Winning percentages were predicted from the probability distribution of build orders using leave-one-map-out cross validation. For each map, data on all of the other maps was used as a training set, then the distribution of build orders was used to predict the fraction of games won by a particular race. Splitting the maps into training and testing sets is undesirable since there are only 7 maps and the rank of the probability distribution matrix is a limit on the effective number of "useful" build orders in predicting the winning percentage.

The model was a linear regression on the build order distributions with the winning percentages in the training data set mapped into logistic space. A logistic map was used for the winning percentages because it complements the ladder system's assumption that skill is distributed according to an ELO system and guarantees that the predicted winning rate lies between 0% and 100%. This analysis was performed for each possible racial matchup and predictions were made using the first two non-supply buildings and again using the first three buildings.

Our goal was to design an analysis that had no prior knowledge of the relation between different build orders. Although one could attempt to define a metric on the build order space measuring how "close" one build order is to another and then kernelize the build order distributions appropriately, there is no such natural accurate metric that does not require a biased existing knowledge of the game.

4 Results

Tables 2 through 7 on the next page show the winning percentages in each matchup observed in our data set, predicted based on the first two non-supply buildings, and predicted based on the first three buildings.

We also looked at the SVD decomposition of the build order distribution matrices to identify the singular vectors for principal component analysis. The singular vectors purportedly represent tradeoffs in preferring some build orders over others and the magnitudes of the elements of the singular vectors corresponding to the largest eigenvalues of the matrix are correlated with the more commonly played build orders. In practice, attempting to extract Starcraft insight from these vectors was concluded to be nonsensical in the context of the game so this data is not presented.

5 Discussion

5.1 A surprisingly balanced game

The most striking observation about this data is that all matchups on all maps but Gaia are within just a percentage point or two of being exactly 50% winning percentage for each race. The overall winning percentages in our data for each race were not known at the time of the project's start. Gaia was developed as a World Cyber Games map, the remaining maps originated in the Korean professional tournaments. It is interesting to note that in our data set Gaia actually exhibits reverse cyclical bias from the canonical imbalance.

The observed winning percentages in our data set were compared with those taken from Korean professional Starcraft games [1] and against all ladder games played on the ICCUP ladder [4] and no correlation or trend was found (data not shown). Although the players in our data set are not full time professionals, they represent the best talent outside of Korea or China and the replay collection is a complete set of ladder history rather than only "notable" games.

Table 2: Terran vs. Zerg

Map	Actual	2 Buildings	3 Buildings
Blue Storm	48.32%	46.86%	48.55%
Gaia	48.32%	49.25%	45.10%
Katrina	52.60%	46.34%	48.22%
Python	49.92%	51.10%	49.87%
Tau Cross	50.19%	48.16%	50.26%
Troy	49.44%	47.98%	45.24%
Zodiac	48.09%	52.22%	52.91%

Table 3: Zerg vs. Protoss

Map	Actual	2 Buildings	3 Buildings
Blue Storm	49.99%	49.95%	53.69%
Gaia	43.22%	42.84%	49.68%
Katrina	49.50%	49.54%	53.34%
Python	49.52%	49.47%	46.29%
Tau Cross	50.71%	50.77%	53.94%
Troy	50.44%	50.34%	44.84%
Zodiac	48.10%	48.19%	42.65%

Table 4: Protoss vs. Terran

Map	Actual	2 Buildings	3 Buildings
Blue Storm	50.02%	97.47%	75.71%
Gaia	41.41%	51.11%	50.47%
Katrina	48.79%	0.49%	4.40%
Python	51.12%	40.91%	46.68%
Tau Cross	51.68%	35.5%	70.27%
Troy	51.95%	99.97%	19.03%
Zodiac	52.28%	37.98%	72.82%

Table 5: Zerg vs. Terran

Map	Actual	2 Buildings	3 Buildings
Blue Storm	51.68%	48.12%	48.34%
Gaia	51.68%	45.52%	49.77%
Katrina	47.40%	73.25%	52.17%
Python	50.08%	52.14%	50.22%
Tau Cross	49.81%	50.87%	49.58%
Troy	50.56%	78.77%	49.13%
Zodiac	51.91%	45.87%	47.41%

Table 6: Protoss vs. Zerg

Map	Actual	2 Buildings	3 Buildings
Blue Storm	50.01%	57.26%	51.61%
Gaia	56.78%	50.36%	51.34%
Katrina	50.50%	56.76%	47.31%
Python	50.48%	54.01%	52.45%
Tau Cross	49.29%	31.29%	47.53%
Troy	49.56%	26.53%	41.10%
Zodiac	51.90%	44.12%	59.45%

Table 7: Terran vs. Protoss

Map	Actual	2 Buildings	3 Buildings
Blue Storm	49.98%	72.26%	54.68%
Gaia	58.59%	48.23%	47.95%
Katrina	51.21%	100.00%	45.10%
Python	48.88%	99.99%	50.20%
Tau Cross	48.32%	62.08%	44.47%
Troy	48.05%	0.00%	41.69%
Zodiac	47.72%	86.35%	44.30%

5.2 Why some matchups are predicted better than others

Given that all but a couple of the win rate data points are statistically indistinguishable from 50% in our data set, it is little surprise that our predictions do not always succeed. Our model predicts some of the matchups much better than others. Looking at the details of the build order distributions and regression coefficients gives us insight into why this is the case.

The predictions on both sides of the Terran vs Protoss matchup are outright terrible. If we were not operating in logistic space, these predictions would not be bounded in the $[0\%, 100\%]$ interval. The model predicts heavily lopsided results but the maps are in fact balanced. The reason for this is all of the build order distributions are heavily skewed towards one very common build. With such a high percentage of the games having one path, the data for the remaining build orders becomes noisy. Both of the 2 non-supply building predictions and the Terran's 3 building prediction each contain a single build order that accounts for over 80% of the games. The predictions for 3 building Protoss vs Terran are still poor but less atrocious. This 3 building distribution has a 75% chosen most common build. There are a still a wide range of possible build orders in this matchup, but intuitively the important decisions in Terran vs Protoss are made a couple buildings later. Our approach of looking at just the earliest couple buildings separates build orders into archetypes of Fast Expand, Rush, and Tech. Both sides of the Protoss vs Terran matchup usually prefer to Tech. There are many variants of tech builds, but these nuances simply aren't captured by our categorization.

The best predictions occur when using 2 buildings to predict a Zerg player versus a Protoss opponent. One of the reasons that this model achieves such great accuracy is that the Zerg has three common openings (Hatchery/Spawning Pool, Spawning Pool/Extractor, and Spawning Pool/Hatchery) which each occur about 1/3 of the time. Going to three buildings actually hurts these predictions. In the three building build order distribution there are still three dominant build orders, but now the 4th and 6th most common build orders are branches from the original 3 that place a Creep Colony. Such an unusually early

defensive structure only makes sense as a reaction to an opponent's early rush, which violates the assumption that decisions are made simultaneously with no knowledge of the opponent.

The Terran vs Zerg and Zerg vs Terran predictions are not horrible, but naively guessing 50% balance would be better. The concentration of build order distributions in these matchups are somewhere in between the Terran vs Protoss and Zerg vs Protoss matchups.

The build order distributions for Protoss vs Zerg are fairly balanced like in Zerg vs Protoss, but the predictions are not as good. In particular, the model underpredicts Protoss's chances on the map Troy. The three most common early builds are two variants of Forge-first fast expand and double Gateway. Traditionally, double Gateway was the most common Protoss vs Zerg build order but ever since the televised 2007 MSL finals of Bisu vs Savior, fast expand has become popular and double Gateway is seen as an uncomfortable build forcing short-term gambits. On Troy the double Gateway build allows Protoss to threaten to change the connectivity of the map by destroying neutral buildings so this build order is not weak.

In principle, the results for the 2 non-supply and 3 building analyses should be nearly identical for the Protoss matchups. Since Protoss must use a Pylon as the first building, the only information added in the 3 building analysis is the Pylon/Gateway/Pylon build. The fact that this control test fails suggests that our method is fundamentally nonrobust despite ostensibly working well for Zerg vs Protoss.

Many of our predictions struggle on the map Katrina. Katrina has a resource cluster behind each player's base that allows unusually early fast expansions. The map shows an excess of 70% win rate for Protoss over Terran in professional games [1] but favors Terran slightly in our data set.

Our analysis does not take into account any knowledge of which build orders are conceptually similar. It is possible that better results would have been obtained by looking at build orders out to 4 or 5 buildings then combining the many rare build orders into the more common variations to reduce the number of possible build orders to a reasonable amount. Such manual classification requires biased judgment on the part of the researcher. Furthermore, the same early build order can convey multiple intentions. Sometimes instead of building in his own base the player builds in close proximity to his opponent's base. The most common "proxy" tactic is to build Barracks or Gateways in an all-in gamble to end the game in the first few minutes. Proxy buildings can also be used economically to block a Zerg opponent's early expansion or prevent an opponent from harvesting his gas geyser. In particular, Protoss double Gateway is frequently played as both a proxy and not a proxy and we cannot distinguish the two. For Terran, proxy buildings are usually signaled by building a Barracks before Supply Depot, but there are exceptions such as the non-committal fake Bunker rush designed to make Zerg lose mining time by sending resource gatherers to defend before military units can be created.

Even if the winning percentages lined up perfectly, the point of our analysis is not to tell one how to be a better Starcraft player. By analyzing a distribution of chosen build orders, we cannot conclude which build order is the "best" since we already assumed the existence of a Nash equilibrium. The conceptual idea is that when players feel that a map encourages them to Rush or Fast Expand more frequently than normal then this is reflected in the winning percentages. However, the Protoss vs Zerg results show that radically different build order distributions can still lead to balanced results.

6 Conclusion

We have taken data on thousands of replay files from a high level tournament. Except for the case of a Zerg player versus a Protoss opponent, our model fails to convincingly predict winning percentages better than a naive coin toss. Rather than being disappointed, this is great news in the context of the game. The fact that build order distributions have little correlation with win rate means that the game is robust and allows many viable opening moves instead of one forced line of optimal play. It is amazing that each race's winning percentage is so close to 50% across all of the maps. The game would not still be avidly played after 11 years if it had devolved into a single dominant strategy.

References

[1] Teamliquid Progaming Database. `www.teamliquid.net/tlpd/`

[2] Teamliquid Starleague Ladder Replay pack. `http://www.teamliquid.net/forum/viewmessage.php?topic_id=72552`

[3] BWChart 1.04. Software program. Created by 'jca.' `http://bwchart.teamliquid.net/`

[4] ICCUP ladder map statistics. `www.iccup.com/starcraft/sc_ladder_maps.html`