Dan Zhang
Jason Lee
Jimmy Liang
Sanae Mayer

Implementation of a Simple Bot for StarCraft II - Rushbot

Abstract

Imagine how humans and computers play games differently. Human dominates AI by quicker reactions to different situations, but human requires skillful experiences to play well. AI, once implemented, perform actions following a certain algorithm, therefore AI has a lower chance to make mistakes compared to a human, however, AI cannot handle unexpected circumstances.
While it is necessary to take every situation into account and come up with a counter-strategy for each of them. Scripting an AI countering every strategy is theoretically hard. A good way to achieve this is to build a machine/reinforcement learning model and learn from playing simulation games, but it is unrealistic to implement such algorithms in a limited time. We decided to go for the simpler the better – work towards building a strong AI with a simple strategy.

Motivation

StarCraft II is not a game that someone can understand easily. What we implemented was a method to help these beginners out by helping them understand and be more familiar with the simple strategy in any RTS (real time strategy) games.

A single match is divided into three different phases: early game, mid-game, late game, and strategies also fall under these categories. Late game strategies usually mean sacrificing early game, consolidate defense, and wait for the opponent's flaw to launch counterattacks. Mid-game strategies usually mean go through the early game by expanding/upgrading, then attack once fully prepared. Early game strategies usually mean go straight to launching attacks. We can clearly see that early game strategies are the simplest to follow – we call it rushing. Since rushing determines the winner in the early game if successful, strategies relying on going into mid/late game do not have any chances to carry out.

Approach

Once we decided to use rushing as a strategy, we must also choose a race to start with. Starcraft 2 has three different races: Terran, Protoss, Zerg. Each has different strengths and weaknesses. We will discuss them by means of the early game, mid-game, and late-game performances.

Terran is famous for its excellent defense, strongest damage vs cost efficiency, and easy to play by beginners. By being able to build cheap units with high attack, Terran is a very ideal race to do a rush. They have advantages in the early game and in mid-game. Terran's weaknesses are in the late game, when opponents fully developed their defensive architecture, Terran's offensive abilities are useless by then.

Protoss is a very flexible race to play. They can mine for resources and build simultaneously, which makes Protoss also ideal for early game. They also have strong units in late game. One main disadvantage is that while the other two races can do many things in the mid-game, Protoss spends their time heavily on upgrading. Once completed, Protoss can find their dominance back.

Zerg has the lowest unit defense and base defense, meaning their units are very fragile. During the early game, Zerg usually spends their time on building defense. Zerg also has lots of gas-dependent armies, which limits their availability in early games. So Zerg's playing style are usually going by counterattack. But once the game carried out towards late game, with Zerg's quick unit respawning power, enemies are crippled by numerous strong Zerg units.

To summarize, Terran is strong in early game and mid-game. Protoss is strong in the early game and late game, but very weak in mid-game. Zerg is weak in the early game and strong in late game. Since our idea is to dominate the early game, Terran and Protoss seem more ideal for our plan. We should also consider the possibility of rushing failure, and then turn our approach into mid-game. Consider the pros and cons listed above, we finally decided on doing a Terran rush.

One problem: Terran is a race requires multitasking and has low mistake tolerance? This only applies to human players, and we are talking about a scripting AI bot here!

Evaluation

We obviously want a bot that wins most of the times, that is also fast in performance, and doesn't take in more resources when performing the same outcome. By taking into consideration of win rates, the time it takes to finish, different races, and the lowest amount of troops needed to complete the object; we balance all of these aspects together by looking at all of the maximums only. For instance, if we notice that for the number of N_2 troops we can achieve higher win rate results than N_1 but takes more time while the differences between the two win percentage is not too much and they are both maximums, we can assume N_1 is more optimal than N_2. This way we make sure the cost efficiency is evenly distributed.

Dan Zhang
Jason Lee
Jimmy Liang
Sanae Mayer

Prototype Design

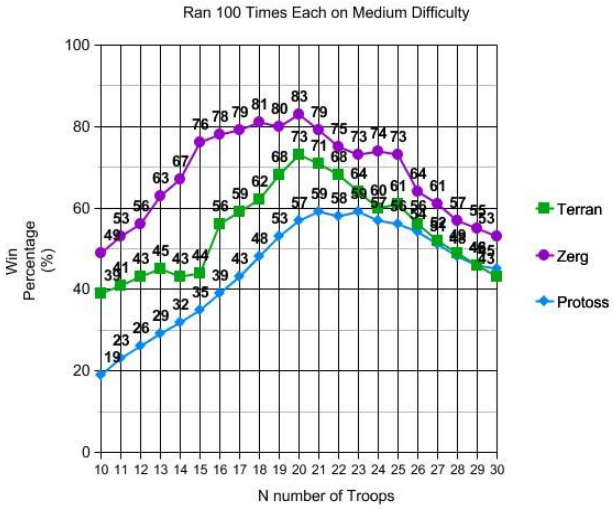
Our first approach was simply to wait until we have enough Marines, and attack the enemy base. Because “rushing” usually means to generate the weakest units of the race as quickly as possible and attack the enemy, our first approach was to generate N number of Marines as quickly as possible therefore attacking the enemy unexpectedly.

Here is the pseudocode for our first approach to the rushing technique:

```
#At every time step

#If the total count of all AI exceeds the max capacity
if (ai_count >= max_capacity)
    #Build a supply depot
    build_supply_depot()
else
  #If not enough workers are generated
  if (workers_count < 7 -> 15)
    #Focus on workers first
    generate_workers()
  else
    #If met quota

    #Try to make troops if possible
    try_to_make_troops()
    if (barracks < 2)
      #Make barracks
      build_barracks()
    else
      #If enough troops
      if (troops_count >= N)
        #Rush
        attack()
      else
        #Try to make troops if barrack exists
        try_to_make_troops()
```

As you can see, the pseudocode is very simple to understand which makes it not difficult to make a modification to the code.

Prototype Result



The graph displays the win percentage of the Rushbot when fought against the three different races when the program is run 100 times on medium difficulty over a different number of troops.

From this graph we can gather the information that around when N reaches 20, we have the highest rate of wins. While it is not the fastest, it is a maximum for both Zerg and Terran. Even though Protoss has a higher win rate when N is either 21 and 23, with the evaluation we're going with, we set N as 20 as being the most optimal.
We can also gain intel that while Zerg is commonly used for many rushing techniques, their defense is very flawed and can be beaten easily during an early game rush, whereas Protoss have high defense in the early games but slowly falls as the game progresses, and finally, Terran being the most balanced between the three races.

New Design

By evaluating the reason why we did not have a consistent win-rate against all three races, especially against Terran and Protoss, we noticed that if they had strong early units that were enough to fight against our Marines, such as Zealots, Marines would simply lose the fight therefore not having any way to come back from the defeat.

The second approach to the solution was to include a unit that was strong enough to fight against the strong early units. Tanks are one of the strongest ground units in StarCraft II because they have a strong attack-damage especially when they are in seized mode. Because Tanks have a range advantage against all units and due to the fact that all the early units across all races are ground units, Tanks fit the best to rush with the Marines so we can successfully defeat the enemy.

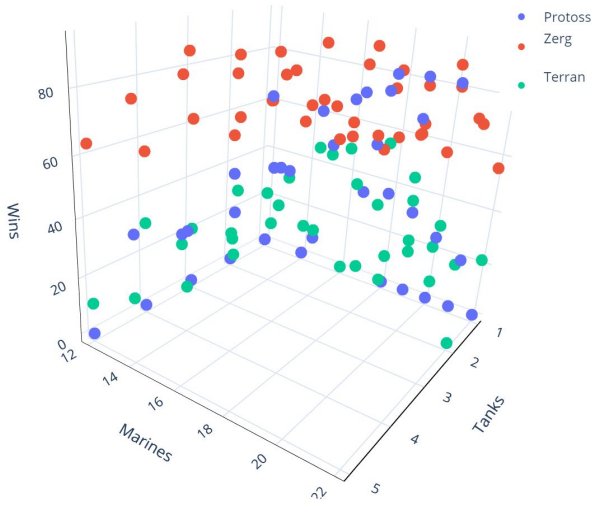
Here is the pseudocode of our new design:

```
#At every time step

#If the total count of all AI exceeds the max capacity
if (ai_count >= max_capacity)
    #Build a supply depot
    build_supply_depot()
else
  #If not enough workers are generated
  if (workers_count < 21)
    #Focus on workers first
    generate_workers()
  else
    #If met quota

    #Try to make troops if possible
    try_to_make_marines()
    try_to_make_tanks()

    #Try to make barracks if not enough
    if (barracks < 3)
      #Focus on barracks before factories
      build_barracks()
    else
      #Try to make factories if not enough
      if (factories < 2)
        build_factories()

    #If factories not built yet and at least 1 marine
    if (factories < 1 and marines_count > 0)
      scout_with_marine()

    try_to_make_marines()
    try_to_make_tanks()

    gather_up_troops()
    #If enough troops
    if (marines_count >= N and tanks_count >= M)
      #Rush
      attack()
    else
      #Try to make tanks and marines if building accessible
      try_to_make_marines()
      try_to_make_tanks()
```

Compared to the prototype pseudo, not much has changed since then. The only additional things added were the building of an extra barracks, factories, and tanks, and waiting on the number of tanks.

Now an experiment against StartCraft II bots will be needed to determine the combination of the number of Marines and the number of Tanks.

Final Result



With the experiments against StartCraft II bots that had a difficulty level of Very Hard, a total of 10 games for each set of units was performed to gain statistical evidence of the win rate percentage. Against Zerg, a win-rate above 50% was achieved of all sets of units. By having more than 2 tanks and 14 marines, a win-rate of more than 70% was achieved. Against Protoss, because their units consist of strong early units, if we did not have enough Tanks to fight against the units, the units ended up defeating the battle, therefore losing the game. By looking at the graph, once more than 3 tanks and 19 marines were gathered, a win-rate of more than 70% was achieved. Lastly, the average performance against Terran shows the lowest win-rate because of the fact that Terran has a strong defense. Again, once more than 3 tanks and 20 marines were gathered, the new design achieved more than 50% win-rate.

Because we are giving up time to make more Tanks, rather than to rush with Marines, the enemy had more time to create the defense, therefore, making the units harder to win the battle. However, the statistics show that the new design was able to defeat much harder difficulty than the early design proves that the newer design performs better.

Conclusions

There are a few advantages as well as disadvantages for using this rushing technique. Firstly, our rushing is easy to implement and understand since the strategy is very simple. All we need to do is build troops until it reaches a certain number of amounts and attacks. Secondly, we can get the result very fast. If the first wave of rushing hits the opponent hard, then we just keep attacking, in this case, we can beat the enemy very fast. But if the units for the first wave of attack got eliminated by our opponent. Then we are most likely to lose this game. Lastly, this strategy is effective when facing the opponent with poor defense at the early stage. In the early-game, most opponents might have a weak defense system. Thus siege tank can hit the opponent hard with its high damage. This makes the opponent hard to defend.

There are some issues for our rushing policy as well. The most obvious one would be the lack of mid-game, late-game strategy. We constantly use the same technique: when the number of the tanks reaches 3 or the number of marines reaches 20, then attack. This might be workable in the early-game to hit the opponent off guard, but when the opponent has more strong units to defend, especially in mid-game and late-game, it is impossible to beat it. Additionally, if the opponent rushes earlier than us, we have a low probability to stop it. Because our defensive strategy is fairly simple. We just gather the troop near our base for defending when the amount is not reached. For the strategy wise, we have a poor defense for air attacks. Since for our troops, marine is the only race who has the ability to attack air units. The other disadvantage is that there is no specific strategy for each race. Since each race is different and they all have their pros and cons. We have a relatively high win rate for Zerg but a low win rate for Terran. So we need to have more specific strategies for different races.

Future Work

In general, this rushing is a good strategy for beginners. But to make it more optimal,

there is a lot to work on. If we ever have the chance to improve this, we might want to consider doing the following.

The first thing that we can do is that we can add strategies for mid-game and late-game. Secondly, we also need to add a more advanced defense strategy to prevent opponent's rushing. Later on, we can also add different strategies for different enemies to make it more comprehensive. Lastly, it would be great if we add a few more units to support our rushing, like Medivac, Viking, etc.