Report group 8
Learning RTS AI

Gjermund K. Lie
Tom Sverre Hageland

Validation and Testing

Early testing: Our simulator test cases:

‘**Clear case’.** In this test, two simulators are pitted against each other, where one will focus on producing only one particular unit type (i.e: archer), while the other will build only the “anti-unit” (i.e: cavalry). The result shows that the simulator works as planned in such extreme cases: The simulator with the anti-units will win 100% of the simulated games.

‘**Combinations’.** In this test, two simulators are again pitted against each other, where one will focus on producing two different unit types (i.e: archer & infantry), the other will produce anti-units to both these (here: cavalry and archer). Again the results show that the simulator works as planned with a win rate of around 70% in favor of the simulator with the anti-units. Because the first simulator also has the anti-unit to half of the second simulators army - the assumed "underdog" in this test has a slightly better chance at winning than in the clear cases.

‘**Slight alterations’**. In this test, two simulators are once again pitted against each other, this time with only minute differences in unit-production priorities. The results are bland, and depend heavily on what kind of unit the priority difference influence (which is an indication of a game balance problem), and the results also depend on what resources are in the expansions that the simulators capture. Building slightly more cavalry, for example, usually results in a higher win rate, indicating a slight imbalance towards cavalry being a superior unit.

‘**Aggression – Low’.** In this test, one of the two simulators will have its aggression level (i.e., the chance that it will attack and capture enemy expansions or bases) significantly lowered, keeping the unit rates equal. The results coincide with the expectations in that the chance to win the game is greatly reduced. This is because the one who controls the most expansions controls the most resources, and also has increased unit production.

‘**Aggression – High’.** In this test, one of the two simulators will have its aggression level significantly heightened, while still keeping the unit rate equal. The win tendency of this strategy should not too certain, as the results should vary between whether the aggressive simulator attacks an expansion or a base. (Attacking a base causes higher troop-losses.) At the moment it performs a bit too well. The defender allocation algorithm has been improved to account for this, and is ready to be implemented.

’**Attack Choice’** . In this test, one of the two simulators is ‘coaxed’ into choosing a ‘smarter’ target. (I.e: it will attack expansions before bases). Unit rates are kept equal. This causes less troop-loss to base defenses and ensures that the simulator can gather more resources. As expected, this raises it’s win-rate considerably.

Additionally - Setting aggression to "high" and "attack choice" towards focusing on expansions logically (and correctly) leads to a vastly increased win ratio, even with equal unit rates.

Thoughts and interpretation

First impressions

According to the early testing - our simulator actually does a pretty good job at giving us the results we expect from two static opponents. In other words - the simulator is working as intended, or close to it. Some minor balancing changes, some of whom are already ready for implementation, and we have a solid base for testing our AI as it comes along. Next up is of course a simple AI which adapts to the situation. The main "problem" with the static simulators are that they don't work well with lacking a particular resource, or having little of a particular resource available for gathering - leading to reduced unit production. Adapting to these things alone would improve performance significantly. The next step - adapting to various unit combinations - would improve performance even more. In other words - a simple AI with adapting algorithms should be able to "learn" to beat a static opponent every time in the course of a few games. This is our logical next step for this project.

Further work

When the adapting AI is working as it should, we will try to improve its adaptability through learning - our intended target values for learning is the line between the actions of increasing and decreasing the rate values described above. Through several games played, a learning AI should improve the choices it makes to determine whether a value should be increased or decreased according to the results of both the final outcome of the game and the direct outcome of the changes made within the current game. The exact method to achieve this has not yet been determined, but we're working on a reward/punishment system designed to determine the "ultimate" strategy - if it exists - for this simulated game.