Reachable Level of Stratego Using Genetic Algorithms

Author: R.M. de Boer, student in Artificial Intelligence

Supervisor: Prof. V. van Oostrom, professor at the Faculty of Humanities

University: Universiteit Utrecht

Bachelor Thesis for 7.5 ECTS

February, 2012

Contents

0 Preface **3**

1 Introduction **4**

2 Stratego **5**

3 Previous Work **6**

4 Genetic Algorithm **7**

5 Expected Results **10**

6 Experiments **13**
6.1 Experiment Level of Opponent 13
6.2 Experiment Increasing the Number of Generations 14
6.3 Experiment Increasing the Amount of Opponents for the Solutions per Generation 16
6.4 Experiment Increasing the Range of the Parameters 17
6.5 Experiment Increasing the Population Size 19

7 Best Configuration **20**

8 Evaluation on the Results **20**
8.1 Attacking Unmoved 21
8.2 Urge to Beat Weaker 22
8.3 Move to Own Side 22
8.4 Attacking Unknown but Moved 22
8.5 Tiles on Outer Columns of the Board 22

9 Discussion **23**

10 Conclusion **24**

11 References and Acknowledgements **25**

12 Reflection **26**

13 Access to Code **26**

0 Preface

I would like to thank the following people for assisting me in different ways: prof. V. van Oostrom for supervising, Vincent Tunru for the cooperation, Vincent de Boer for inspiring me for this subject, Sander Arts for supplying us with his StarBot, Ryan Albarelli for access to his bot, Pim Niemeijer for the deep conversations on Stratego, Chip McVey for login credentials on Metaforge, and Maarten Wiedenhof for his support. In addition I would like to thank the following people for reviewing and commenting on the draft of this thesis: Bas Zalmstra, Robert-Jan Drenth, Jeroen Wiedenhof, Sara Wegman, Maaike de Boer and Ben Wolf.

This bachelor thesis was written as part of a major in Artificial Intelligence at the Faculty of Humanities of Utrecht University. The research, in collaboration with Vincent Tunru, is about the board game Stratego. The thesises of Vincent Tunru and I have been written seperatedly according to the bachelor thesis procedure [1]. My thesis gives a thorough analysis on the evolved results of a Stratego Bot using a Genetic Algorithm. The emphasis of this thesis is the game play of the bot and the strategy of the bot is looked into in great detail. The algorithm has mainly been set up by Vincent Tunru, as the subject of his thesis. The Stratego-specific parts of the research are eloborated here.

1 Introduction

One of the goals of Artificial Intelligence is to question whether a computer can ever beat a human person in a game. The intelligence of a human is tested in the system of the game. A human has to deduct the best next move from the possible next moves. A program can be seen as intelligent if it can simulate these deductions. The first steps were made 15 years ago when the game of chess was mastered by computers [1]. Today the game Stratego still offers a challenge. The two-player board game features 40 pieces on each side. There are twelve different types of pieces, each having different characteristics: see figure 2. The objective is to capture one of these pieces: the flag. A higher ranked piece will defeat a lower ranked piece, destroying the defences for the flag [2]. Several aspects of the board game are quite intriguing. First, there is only partial knowledge of the current state. Each piece of the opponent will remain undiscovered before the first encounter, because each piece has its identity hidden from the opponent. The consequences of the partial knowledge are enormous: every piece can be of any type. This uncertainty is something a human player must adapt to. For a computer it means more computation time. Secondly, the amount of moves to achieve a sub-goal is high. The board is 10 by 10 tiles [2]. The player might need to get from one side to another with multiple pieces in order to attain a sub-goal in the game. The time required to process every move would be too long, thus a more intelligent approach than brute-force is desired [3].

In this thesis the aim is to give a Stratego Bot intelligence by using the idea of a Genetic Algorithm. This algorithm has its roots in biology, with its concept of evolution, and more specifically, survival of the fittest [4]. Only the most adequate parents may reproduce, their adequate characteristics are passed down to their children [5]. To simulate this principle on the computer, a few elements are required: a population, characteristics, a world, evaluation and reproduction [6]. In the case of Stratego, the population consists of the players. The characteristics are the heuristics of a player. One example of such a heuristic is the urge to move forward. Players may find it very important to enter the opponent's space or, rather, stay in their own territory. The world is the game board, on which players can test their strategies. The evaluation states whether a player was successful. The reproduction generates a new generation of the population. After some repeations of the algorithm, the Stratego Bot, ViCKI, will be well-playing.

The resulting Stratego Bot can now be analysed. Is it, from an experienced human player's point of view, intelligent? Do the characteristics make sense? Is it only successful against a specific type of opponent? Up until this point the Bot has little knowledge on the game, it only knows the rules: what it can do and what it can not. The Genetic Algorithm is fed by characteristics thought to be relevant. However, these characteristics have

no meaning to the Bot; it has only learnt whether a certain characteristic is successful or unsuccessful.

In the following chapters the game of Stratego is explained in more detail and previous work on Stratego Bots is analysed. After that the implementation of the Genetic Algorithm for Stratego, and the experiments with their results. The last partion will give an interpretation of the results.

2 Stratego

The game of Stratego was designed by Jacques Johan Mogendorff [7] and later produced by the company Jumbo. Stratego is a two-player game on a board with 10 x 10 tiles. Every player has 40 pieces and may arrange them freely in her starting area into a setup. The ranks are: Marshal, General, Colonel, Major, Captain, Lieutenant, Sergeant, Miner, Scout, Spy, Bomb, and Flag. The pieces can move one tile front, back, left, or right but cannot move to the blue lake-tiles. If the piece lands on an occupied tile a fight takes place, showing the identity of both pieces. The higher rank will destroy the piece with the lower rank. In case both pieces have the same rank, they will both be removed. Some pieces have special abilities: the Flag cannot move and a capture by the opponent will result in a loss. The Bomb cannot move either, and destroys all pieces when attacked. The Spy is the lowest ranked moveable piece and wins againt only the Marshal, provided that the spy has the initiative of the fight. The Scout is the exception on the rule that a piece can move one tile, as it can move multiple tiles in any of the fout directions. The Miner is the only piece that can take down bombs. The game can end in multiple ways: a player wins by conquering the opponent's flag, loses if no pieces can be moved, and both players can agree on a draw at any time. [8]

Figure 1: Stratego Board

Rank	Name Type	Occurences per player
10	Marshal	1
9	General	1
8	Colonel	2
7	Major	3
6	Captain	4
5	Lieutenant	4
4	Sergeant	4
3	Miner	5
2	Scout	8
1	Spy	1
-	Bomb	6
-	Flag	1

Figure 2: Ranks and occurences of the Stratego pieces

3 Previous Work

Stratego Bots have been created previously. Here is a selection of other bots and the approaches used by them. The first two bots figure as opponent for ViCKI, last three are mentioned for their unique approach. We picked RandomBot and StarBot as possible opponents as the code was available to us. Invincible and Probe are unique for their approach to solve Stratego. Ryan Albarelli has also chosen for a Genetic Algorithm to train his bot. The code we received of this bot was in C, opposed to the other bots which are written in Java.

- RandomBot serves as a test bot. The setup is random and a move is found by taking a random unit that can move to a random available direction.

- StarBot (2010) uses minimax with alpha-beta-pruning. By using minimax it searches a few steps ahead. For each turn all possible moves are retrieved and the assumption is that the opponent will do the most disadvantageous move in return. To evaluate whether a move is good or not, a simple evaluation function is used. To restrict the computation time alpha-beta-pruning is applied. This results in not considering every possible move each turn, since a search may be closed earlier if it has little chance to become the best. [9] The bot performance is reasonable. The level of the Bot is dependent on the search depth for the moves. If it looks 2 moves in advance, it has trouble winning over the RandomBot. However, 5 moves makes it a challenging Bot. For this research more than 5 moves was not practically possible on the hardware used.

- Invincible (2007) is focused on specific sub-goals or plans in Stratego. When it detects that the flag is open, it will move the necessary pieces. It uses mini-max with alpha-beta pruning to find these paths. [10]

- Ryan Albarelli's bot (2003) has an approach which relies on Genetic Algorithm (GA). It uses parameters as the number of the opponent's and its own generals. Minimax enables the GA to look a few turns ahead. The bot has been tested against itself. [11]

- Probe (1983 –2009) is a very complex Bot [12]. Minimax is used with a sophisticated evaluation function. It calculates extensive predictions about the opponent's units and detects patterns in the opponent's game play by remembering the setups used. This bot has won the World Championships for Stratego bots. [13]

4 Genetic Algorithm

The Genetic Algorithm takes a representation of the game and gives a set of new solutions. In this research the parameter approach for Genetic Algorithms has been applied[14]. There is a set of possible considerations for each move to be made. The setup or starting position of the pieces is given and consistent over all solutions. After a few generations the algorithm is able to determine how important an action is and whether it is preferable. The values of the parameters represent the strategy of the solution.

We have chosen two different kind of parameters. The first categories motivates certain actions, the second represent the value of certain tiles on the board. We have chosen to grant every type of piece in the game a own parameter value. This way, a move can be evaluated to be a good idea for a Miner, but a very bad idea for a Marshal. This is preferred as Stratego has different actors or pieces: the role of pieces differ highly per type as discussed later.

In the first category belong the parameters 1 to 8 and 11. The actions are motivated for the good underlying strategy, to trigger any move at all or to enable unpredictedness. The parameter 'Urge to attack unknown' represents the strategy not to hit pieces blindly. This is a risky maneouvre for a Marshal, which can get reflected by the parameter value for the Marshal for this parameter. The parameter 'Urge to the left' was not implemented for a higher strategy. We do not believe that it is a profitable action per se, however, it triggers a move to be selected. If no move can be selected based on sophisticated parameters, the algorithm will find a move to process. Previous to implementing this parameter, we found the bot to keep moving one piece up and down. The last parameter 'Random value' enables the bot to be unpredicting as there is no required context for this parameter to trigger.

In the second category belong the parameters 9 and 10. The tiles on the center of the board are expected to be important as this is where pieces of both places will frequently face eachother. A high value for the parameter 'value around the lakes' represent a high motivation for moves which land on those tiles.

We have chosen the following parameters: For each move all the values of the applicable parameters determine how important a move is. [15] The most important move will be executed, after which the whole process of selecting a move repeats. The solution is a String of values for each of the parameters. The best performers of the generation have a higher change to be the parent of a solution in the next generation. To determine the succesfullness of the operating bots there is the following strategy:

$$f(s) = \Sigma_{i=0}^{i \to 5} a * piecesleft(s) - b * pieces(0) + c * plys + d * won(s)[16]$$

The total fitness is the fitness summed over all the played games per bot in a generation. For each game the fitness is calculated and inceases over the amount of pieces left standing on the board. Deducted are the pieces of the opponent and the amount of moves needed for the game. It is good to have more pieces left, as it generally means the player had a strong position. A shorter game is preferred over a game which takes superfluous turns. As the goal is to win the game, doing so gives a great boost in fitness. The values of the variables a, b, c and d are changeable over the experiments and represent the importance of the factors.

At the reproduction stage of the program, this fitness will play a role to assemble a new set of parameter values. When initialising, all solutions get appointed a random value within a range. The parameters are fixed after that point, however, other values may be selected from the set. Both parents of a new solution will propagate the values. This way the solution will have a set from the created parameters at the initialization stage.

After a few generations, the solution will have learned as successful solutions are selected as parents. The good gene of the one parent mixed with the other parents gives a new collection of parameter values. The created solution will test the combination during its generation by playing matches against a fixed amount of opponents. One draw-back of this approach is that it may resolve to a local optimum. The bot has found a perfect approach to the game, however, it only works against a specific player. Stratego allows for different tactics. In general a player can be very aggressive or passive. In other words the player can be eager to move the pieces in the bot's setup or await a move. The goal is to create a bot to tackle both types of opponent's play.

To restrict the locality of the resulting paramters, the parents are selected with a specific strategy. It takes into account that the best option is not necessarily the most succesful solution. There is a random influence

Number	Name Parameter	Details
1	First move	the value for the urge to be the first move in the game.
2	Attacking an unmoved piece	the value for whether the pieces wishes to attack an unit that has not been moved by the opponent.
3	Urge to attack a weaker piece	When the rank of the target piece is known and is evaluated to be lower than itself, it may be a good idea to attack it.
4	Urge to walk to the opponent's side	represents how aggressive the unit acts.
5	Urge to move to the right	the piece may prefer to walk to the right side of the board.
6	Urge to move to one's own side	represents how passive the piece acts.
7	Urge to move to the left	the piece may prefer to walk to the left side of the board.
8	Urge to attack unknown, moved pieces	When the piece's rank is unknown, however, it is sure not to be a Bomb and it may be a good idea to attack it.
9	Value move around the lakes	tiles around the two lakes in the center of the board may be preferable.
10	Value of the outer columns of the rows that contain the lakes	those tiles may be preferable.
11	Random value	it may be profitable to something unpredictable random.

Figure 3: The parameters

to broaden the pool for parameters to choose from. The strategy favours parents with a higher fitness. It can be seen as a virual Wheel of Fortune. The strip of the wheel is made of all the fitnesses with a length equal to its value. The random number is the point where the wheel is put to a halt. The wider the solution is spread on the strip, the higher the chance for it to be picked. [17]

5 Expected Results

The resulting values of the parameters are not explained by the Bot. There is no way it will make clear why one strategy is effective for one piece and not for the other. Expectations help put the results in the right perspective. The global expectation is that the Bot's play may differ from regular play. The Bot is programmed and so is the competition. In human play, one factor is to remember the pieces of the opponent. Strategies focusing on the weak memory of the opponent may be very successful against a human player. In fact, by constantly moving the known pieces or placing a piece quietly on a spot with little attention are strategies often applied by humans. A piece is worth more when the rank is unknown for several reasons. Firstly the piece can be left unprotected as it will not get attacked without a second consideration. Secondly it may encounter a piece lower ranked, which otherwise would have avoided the piece.

In computer-versus-computer play, the bot can remember the rank of each piece and whether it has moved without any problem. So can the opponent, making it useless to bluff a piece is unknown. Computers do, however, have trouble in achieving a sub-goal. Coordinating an attack with two pieces towards the opponent's side is relatively light for a human. The player easily notices a spot to take. Then it quickly selects a high and a low piece to move up. The low piece can scout for any interesting pieces that the high piece can then take. For a bot the process would be more complex. To begin with, it has to spot the opportunity from all possible opportunities. After selecting a high piece and an available low piece to accompany close to the seen spot and calculate that both pieces have a path to the spot. This pattern matching is harder for the computer than the human. To make it even worse, the priority of each sub-goal has to be given. A human has the drive to win the game and capture the flag. The computer may find it equally interesting to take a scout as a flag.

In addition, humans may conduct information from the way the opponent plays. This is free information: there is no sacrifice required to notice the opponent is acting over-confident with a certain piece. Higher-level players will take advantage of this by giving fake signals, making the opponent believe it is another piece. Currently, this aspect of the game is not applicable in computer-versus-computer Stratego play. When a bot is facing

Parameter	Positive	Neutral	Negative
Attacking unmoved	Miner, Scout	-	Marshal, General
Urge to beat weaker	Scout, Spy	Marshal, General	Captain
Move to own side	Miner	Low ranked	high ranked
Attack moved, unknown pieces	Marshal, General	Major	Spy
Tiles on the outer columns	Captain, Major	-	Marshal

Figure 4: Predicted values per parameter

a human, it may retrieve information from turn speed. To conclude, by knowing the opponent is a bot, some strategies can be excluded. As ViCKI trains against a bot not nearly sophisticated to show this human behaviour, it will not take the psychology of the opponent into account.

With all this in mind, the expectations are formed as listed in figure 3. The pieces having a special ability will now appear to have a separate value. Then the more common pieces are clustered as one type, as the expectation is that the strategy for a Captain and Lieutenant will not differ significantly. The higher the value, the more eager a piece is to follow the corresponding strategy. If the value is positive, it is preferred for the piece to follow the strategy. If the value is negative, it should avoid the strategy. Around 0 gives the class of pieces for which the strategy is irrelevant or too fluctuating. For the piece the certain situation has no influence on the move to take.

The values for the parameter *attackingunmoved* are probably high for the Miner and Scout and low for the Marshal and General. The goal for the Miner is to find Bombs. The chance for the piece to be a Bomb is higher when it has not moved, as any moved piece cannot be a Bomb and there are 6 Bombs on 40 pieces. In starting position, the chance for a piece to be a Bomb is 6 / 40 = 15%. As soon as 10 pieces have moved, the chance is 20%. For a moved piece, the chance is 0%. The Scout's aim is to gain as much information as possible. As nothing is known about unmoved pieces, it may be valuable to attack it. The Marshal and General are too valuable to risk attacking unmoved pieces. They may face a Bomb or reveal their ranks to a low piece.

The urge to beat a weaker piece is expected to be high for the Scout

and Spy. The only piece beatable for the Spy is the Marshal; killing the Marshal is the sole purpose of the Spy. This makes it a valuable piece, but also the only piece a Scout can beat. Apart from the Flag which both units can defeat, which is the goal of the game. For the General and Marshal it can be a calculated risk to attack a weaker piece. The target piece can be protected by the Marshal or Spy on an adjacent tile. The gain is the piece gathered with the cost of losing the piece. The Captain has a higher chance of being taken by either directly the target piece or the defending piece.

Moving to one's own side is best for defensive pieces and those pieces with a designated goal. A Miner may want to await its prime moment when there is a Bomb to be taken. Lower ranked pieces such as a Lieutenant and Captain have a more complex situation. Their purpose is to gain information from the opponent, which they most likely have to do on enemy's terrain. However, they do not wish to walk right into a trap of the opponent. Marshal and General and Colonel, would like to pressure the opponent and making it harder to move. Walking back to own terrain would destroy this way of blocking the opponent. Attack moved, unknown pieces is a move only the Marshal can do without any direct risk. It is sure that the target piece cannot defeat the Marshal. For the other pieces it is only clear it is not a Bomb. For the General, it would make a nice move. A Spy has too little chance to be succesful.

Tiles on the outer columns on the board are defensive. A more complex rule of the game is the 3-move-rule. It restricts the amount of times a piece can walk over a line on the board. More concrete, a piece cannot walking back and forth over two tiles more than three times in a row. If a piece is now walking in from the middle, it is profitable to be at the other tile. The feared piece is not in the position to force the piece into a new direction or get beaten. [18] An example of a situation in which this rule has an impact, is shown in figure 4. The Blue piece is moving in on the Red piece along the arrow. The pieces will now be in opposed lines, giving the Blue piece no opportunity to hit or chase away the Red one.

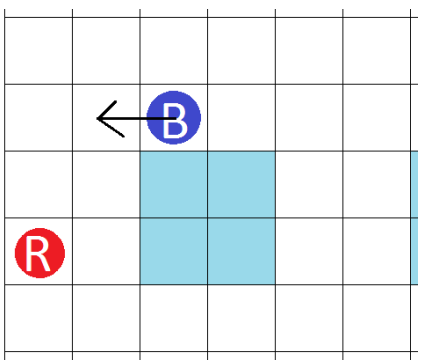

Figure 5: Snaplet of the board illustrating the defensive position for red

6 Experiments

Experiments with different properties for the Genetic Algorithm are executed to get the best solutions. Beforehand it is unknown which combination will work. We expectated the following cases to have an influence on the learning curve and the fitness reached:

1. Starbot or RandomBot as opponent
2. Increase in the number of generations
3. Increase in the amount of opponents for the solutions per generation
4. Increase in the range for the parameter values
5. Increase in population size

The factors which all experiments have in common, unless specified otherwise:

- Population size: 100
- Generations: 20
- Parents: 2
- Opponents: 5
- Range: 0.5

The following pages show the results of the experiments. The Genetic Algorithm has a chance to evolve on a sub-optimum. As the learning is based on search depth of 1 move, this happened in some rare cases. Graphs of learning curves can be found in this section. The next chapter focuses on the best configuration and reviews the results.

6.1 Experiment Level of Opponent

The created Stratego Bot based on Genetic Algorithms, ViCKI, had two possible bots to face: RandomBot and StarBot. As mentioned previously, the RandomBot does what it's name suggests: random moves. The StarBot uses minimax and a simple evaluation function.[15] For practical reasons, the Search Depth of the algorithm has been set to 2.

The figure shows the improvement of ViCKI. The fitness jumps up and down, however, there is a logarithmic increase in fitness. The highest point is around 1500, and the resulting bot has a fitness just below 1300.

There is a slight improvement in the fitness. It does learn over time, however, it is not preferred to have such a high range in the fitness. Its highest point is around 1130, and its lowest even 880, with a result of 1080.

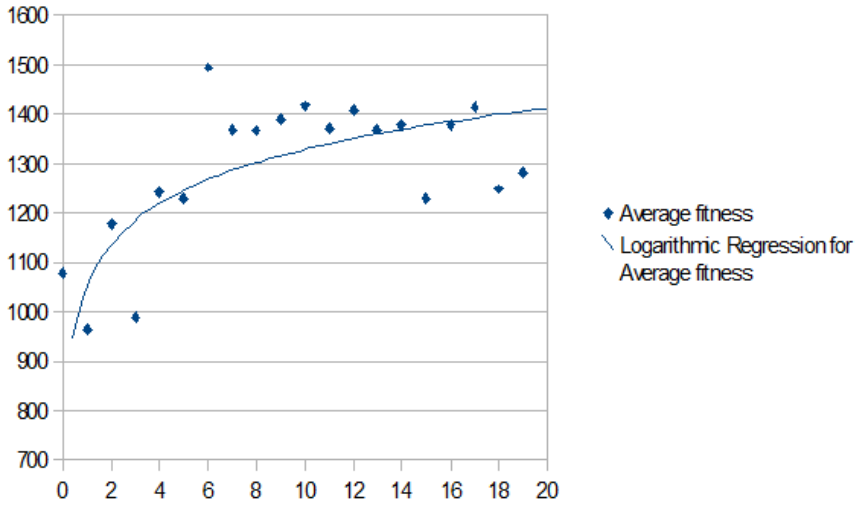

Figure 6: Average fitness against RandomBot (population 50)

6.2 Experiment Increasing the Number of Generations

The more generations of solutions the more time the bot will get to improve. The algorithm will have more time to evolve, so it may result in a better combination of parameter values. The experiment tests 60 generations, opposed to the 20 shown in figure 5.

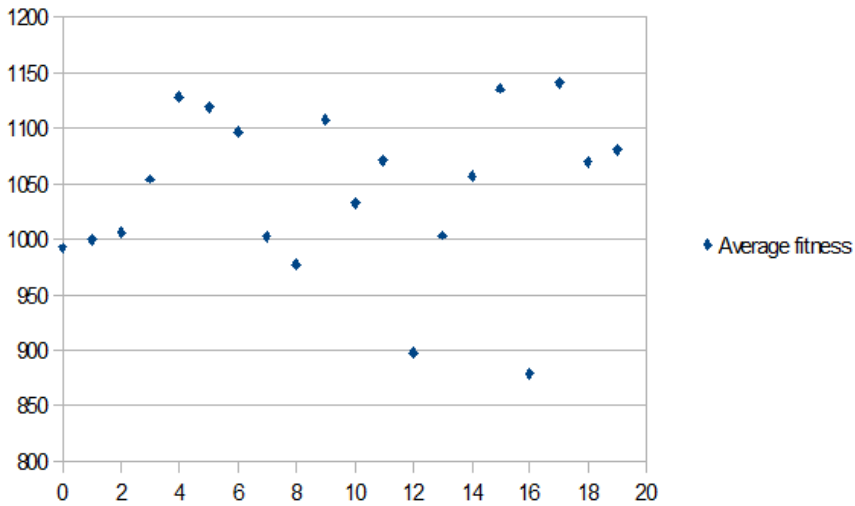

Figure 7: The average fitness against Starbot (population 50, SD 2)

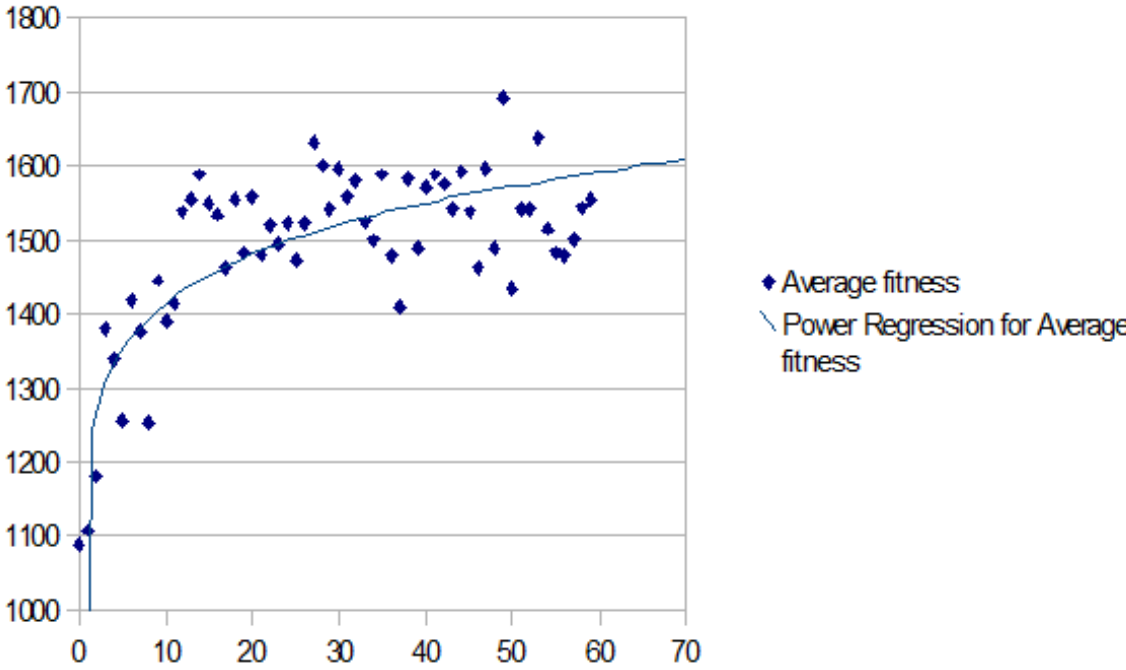

Figure 8: 60 generations instead of 20

The chart shows an improvement of the Bot. However, after 20 generations there is no more real gain. One possibility is that the best combination for the set of parameter values has been found. Another is that the current range for the parameter values is not sufficient. A solution may be to make the starting population or the range of the values bigger as both situations guarantee a more diverse starting pool.

6.3 Experiment Increasing the Amount of Opponents for the Solutions per Generation

Increasing the amount of opponents in each round aims to limit the factor of luck. Facing just the opponent Bot once a round is enabling a solution to win with only one working strategy. However, the Bot performs better when it is able to counter multiple types of games. The number of opponents was previously 5, figure 8 shows 50 opponents.

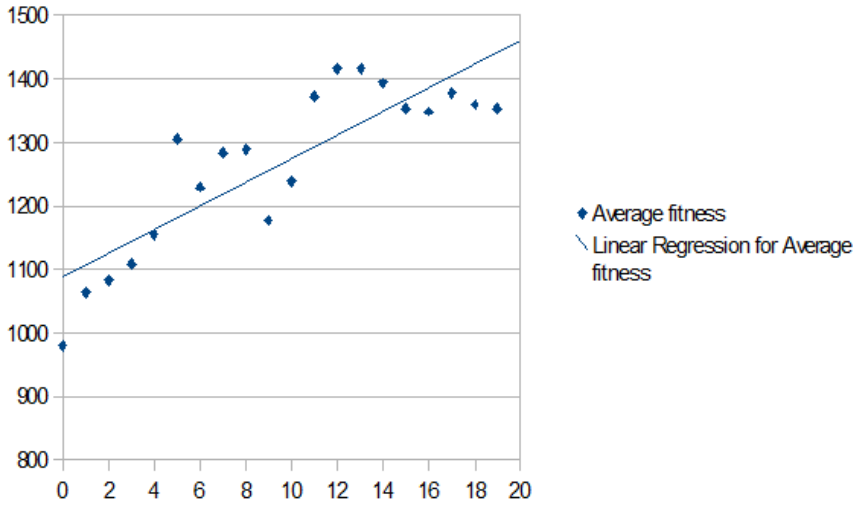

Figure 9: 50 opponents per solution each round

6.4 Experiment Increasing the Range of the Parameters

The range of parameters influences the diversity of the system. The more values are accepted for the parameter, the more possibilities there are for the starting pool. This allows a parameter to have even more impact compared to the fellow parameters. If the range is 2, such as the range the experiments started with, the values have a value from -1 to 1. The difference between two values can never be bigger than 2, while this may be desired. The sum of the 11 parameters decide whether the evaluated move gets chosen. With a range of 2 a value can at most be 2 higher than another value. When it is possible for the values to differ more, there is a higher chance for the values to have a real impact. The algorithm might be able to give a better focus on important parameters which result in a better performing bot.
The improvement for a range of 8 is as follows:

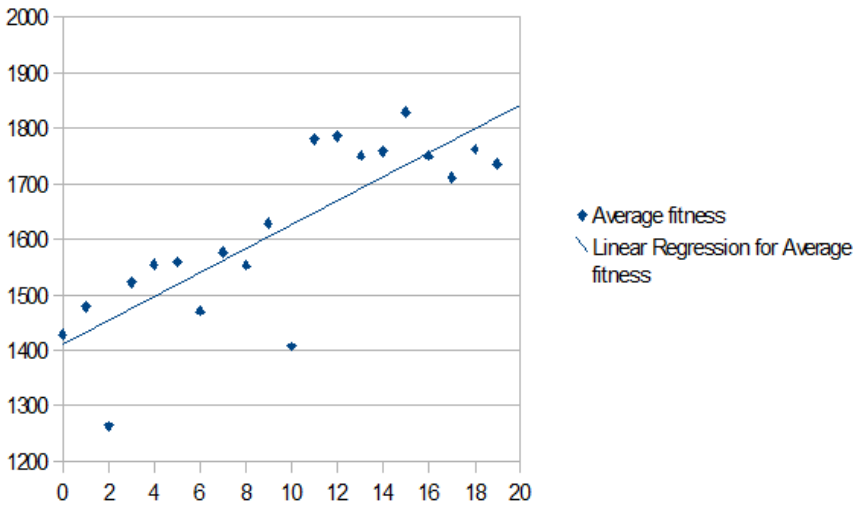

Figure 10: Range of 4 opposed to 1 for the parameter values

6.5 Experiment Increasing the Population Size

The more solutions randomly created at the starting stage, the more diverse the system is. The parameters have a desired value under which they are the most successful. As the amount of solutions increases, the number of appointed values to parameters is increased. If the value has not been chosen at the start, it cannot appear in the result.

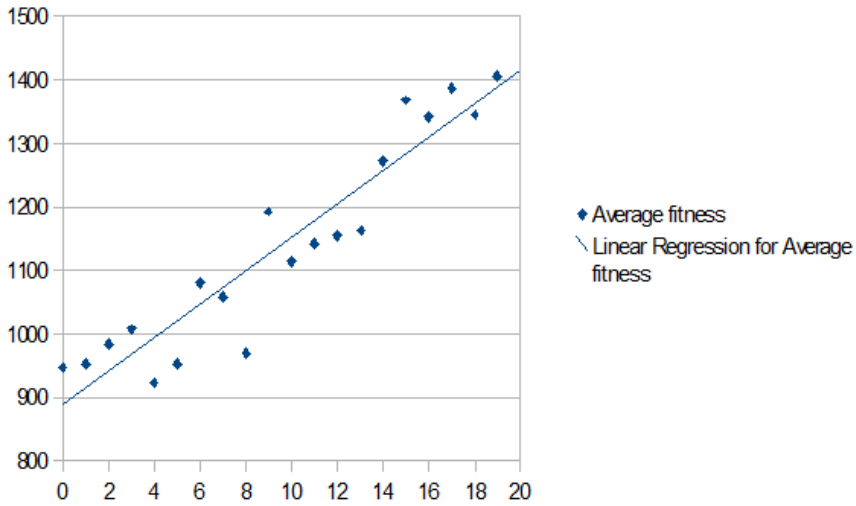

Figure 11: Population of 200 instead of 100

7 Best Configuration

The best combination can now be formed. The conclusions from the previous chapter combine to form for a good configuration. To conclude, the fitness has been proven to increase with the following settings:

- Bigger population: The population is set to 200.
- Wide range: The range is set to 4.
- More opponents: The solutions are tested 5 times against the opponent.
- Amount of generations equal: The number of generations (20) is kept the same. [15]

The result is the following learning curve for the bot. The fitness exceeds the 1900 mark easily.

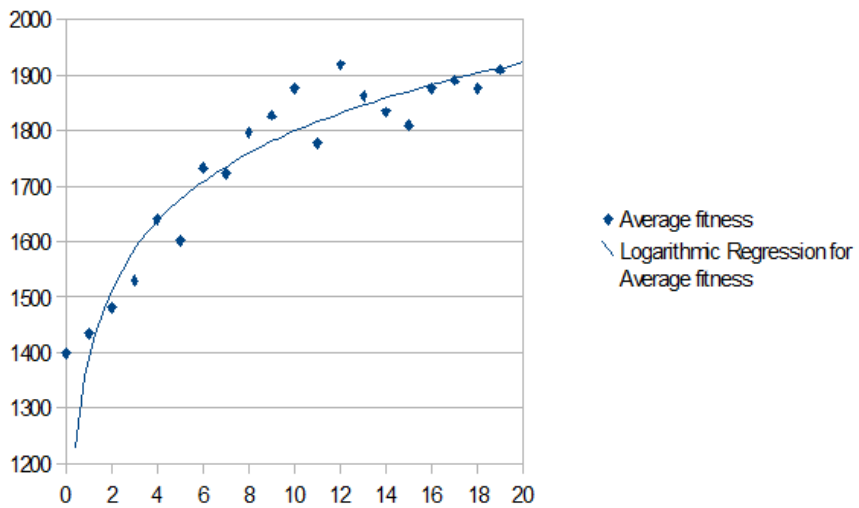

Figure 12: Average fitness with 20 generations, 200 population, 5 opponents and range of 4

8 Evaluation on the Results

An evaluation of the results is given in order to reach a more detailed view on the performance of ViCKI. To acquire an evaluation of the results and reach a more detailed view of ViCKI's performance, the expectations on

Parameter	Positive	Neutral	Negative
First move	Sergeant, General	Marshal, Lieut, Miner	Spy, Major
Attacking unmoved	Captain, Major	Lieut, Miner, Scout	Marshal, Colonel
Urge to beat weaker	Colonel, Marshal	General	Scout, Miner
Urge to walk to opponent's side	Marshal, General	Sergeant, Lieutenant	Spy, Captain
Urge to the right	Lieut, Marshal	Scout	Spy, Major
Move to own side	Captain, Spy	Major, Sergeant	General, Colonel
Urge to move to the left	Marshal, Captain	Sergeant	Spy, Colonel
Attack unknown moved	Marshal, Miner	Lieutenant	Sergeant, Spy
Next to lakes	Spy, Major	General, Captain	Miner, Lieut
Tiles on outer columns	Scout, General	Major, Spy	Marshal
Random influence	Lieut, Colonel	Miner, Major	Spy, Scout

Figure 13: Showing the results per parameter

the parameter values are tested against the resulting parameter values. The parameters have evolved over the generations and result in the values shown in Figure 12. These results can now be compared to the prediction and be explained.

8.1 Attacking Unmoved

Attacking unknown unmoved pieces is a risky move as the rank of the opponent is unknown. The Captain and Major have the chance to either hit the target piece, or explore a high unit. It is useful to know the location of the General and Marshal in the game. The player can avoid pieces that are too high in rank, and other places can be marked as safe. For a Lieutenant, the chance to defeat the target piece is lower, and the chance to be hit by one of the four Captains is added to the chance to be hit. The Miner both wants to hit Bombs and prevent a likely defeat. The Marshal and Colonel are too valuable to risk on an unknown piece as they may explode on a Bomb. The result matches the expectations from section 5, except for the Miner which was expected to be more valuable.

8.2 Urge to Beat Weaker

The Marshal and Colonel have higher values than expected. The General's value is more average, as in the previous parameter. The Scout and Miner do not prefer to beat weaker. Miners are expected to take down the Bombs and take the Flag if possible, however, should not focus on Scouts. The Spy has a lower value than expected, since it may not have been very important. There is a small chance that the opponent's Marshal is both known and standing next to the Spy. The Scout has a lower value, as it less important to take down the Spy. The result matches the expectations from section 5, except for the Spy.

8.3 Move to Own Side

Generally, the special pieces prefer to stay on their own side, the low in rank are neutral and the high reluctant to move back. The Spy prefers to walk to its own side along with the Miner, while not expressively positive. The high pieces have a strong urge to walk to the opponent's side, which translates into a low value for walking back. This version of the Bot can be said to have an aggressive play style, which corresponds to moving its pieces to the opponent's side. The resulting values match the expectations from section 5.

8.4 Attacking Unknown but Moved

If Marshal is likely to hit any moved piece, it can only turn out bad when it is protected by a Spy. The target piece will be a secure win, as it is excluded to be a Bomb and no other piece can defeat it. It will disappear when faced by the opponent's Marshal, in which case they are each immediately traded off. This can also be seen as a positive event, as there were no other scouting pieces necessairy to get to know the location of the Marshal. The Miner has lost some of its value, as the opponent's setup is random. The chance for the flag to be surrounded by Bombs, making the Miner vital to capture the Flag, is very small. The Spy is reluctant to hit, as explained by the fact that it is only strong against one piece. The chance for the opposed unit to be the Marshal is too low to make the move valuable for the Spy. The result matches the expectations from section 5.

8.5 Tiles on Outer Columns of the Board

The Scout and General prefer the defensive position the most, the Major is more restricted. The Scout can be hit by any piece, however, the need to protect it is low as the rank is low. On the other hand, General and Major are valuable. The result is for the Scout to prefer the tiles on the the outer columns very strongly, the General to Lieutenant to be moderate, and the

Marshal to be very reluctant. The result matches the expectations from section 5 for the neutral and negative values. The tiles are more profitable for the Scout and General.

9 Discussion

There is enough left to explore in this topic. The first steps for ViCKI have been set; however, there is more to do. During the research, several possible improvements were encountered. Here is a collection of subjects for further research:

- Adding minimax. This Stratego Bot can figure as a strong evaluation function as part of a minimax system. Currently, the bot has a Search Depth of 1 at which depth it can make a sophisticated move. Enabling the Bot to look ahead allows it to pick the move that proves to be the best in a few turns. The Stratego Bots 'Starbot', Ryan Albarelli's bot and 'Probe' seen in section 3 already use the minimax algorithm.

- Improve parameters. The parameters now reflect some situation on the game board in which a certain action is preferred or not. There are more situations that could be encoded into parameters, which may make a better Bot. For example the parameter 'hitting a piece when Marshal is 10 tiles away' evaluates a move when it is safer to hit. The Marshal will not be able to hit the piece directly after the attack, however, it is not a very general parameter. In addition, the current parameters do not take the stage of the game into account. One move may be good at the start of the game, but bad at the end game. The Miner is very valuable at the end of the game. At the start of the game, it has a high chance of being defeated when it has walked to the front.

- Other reference bots. The current opponent was completely random, which makes it harder to achieve advanced stratefies based on bluff. The bot will always recieves a random answer. The resulting Stratego Bot performs very well against random opponents, however, it will not learn to play against sophisticated opponents. In order to play against humans, it may be needed to test the Genetic Algorithm against an improved bot, or even better, against humans using crowd surfing. This last aspect has been explained on page 9 and 10 of this thesis.

- The setup. One part of the game is the setup made at the beginning. This stage has not been discussed, as a preset has been used. If another bot were to remember the setup discussed, ViCKI would not stand any chance. In order to become a real threat, the bot must tackle

this aspect as well. I strongly believe a good setup can be found by applying machine learning on human setups.

10 Conclusion

The conclusion of the research is that Genetic Algorithm applied on Stratego lead to a competing Bot. The set of parameter values evolved to such an extent that it could be explained by a human. The Bot has adapted to the opponent's playing style. In ViCKI's case it meant that it had to counter a random playing style. The result is that the Miners were evaluated to be less worth than expected in section 5. The lower value for the Miner can be explained by the decreased chance for the Miners to be vital to the game in a game against a random instead of human player. However, most resulting values for the parameter in section 8 match the expectations. The strategy not to attack unmoved pieces by high ranked pieces is confirmed by a low value for the parameter 2 evaluated in section 8.1.

The Stratego Bot is able to make sophisticated moves. It has learned to play Stratego, showing convincing wins against opponents. However, it is not in the scope of this thesis to answer the question proposed in section 1 on whether this system is intelligent. The parameters we have chosen were enough to evolve to a reasonable bot, I can imagine that this part can still be improved as mentioned in section 9. Selected actions and tile values are taken into account by the bot, which are useful as it leads to positive results.

11 References and Acknowledgements

1: Van Lith, J., 'Bacheloreindwerkstuk Cognitieve Kunstmatige Intelligentie'. 2009, page 6, URL: http://www.uu.nl/SiteCollectionDocuments/GW/GW_KunstmatigeIntelligentie/CKI_eindwerkstuk.pdf 1: Campbell, M., Hoane Jr., A. J., Feng-hsiung Hsu, 'Deep Blue'. 2002, Elsevier, Volume 134 issues 1-2, pages 57-83, DOI: http://dx.doi.org/10.1016/S0004-3702(01)00129-1
2: Jumbo International, 'Stratego original, val aan en verover de vlag!'. 2008, page 4, URL: http://www.jumbo.eu/media/products/assets/manuals/99a69525-ed0c-503c-b2cf-fa2ecc3625fb.pdf
3: Arts, S., 'Competive Play in Stratego'. 2005, page 11, URL: http://www.unimaas.nl/games/files/msc/Arts_thesis.pdf
4: Obitko, M., 'Biological Background'. In: Obitko, M., ' Introduction to Genetic Algorithms', 1998, 1, URL: http://www.obitko.com/tutorials/genetic-algorithms/biological-background.php
5: Van Doorn, S., 'Survival of the fittest?'. 2003, Nederlands Instituut voor Biologie. URL: http://www.kennislink.nl/publicaties/survival-of-the-fittest
6: Obitko, M., 'Genetic Algorithm'. In: Obitko, M., ' Introduction to Genetic Algorithms', 1998, 1, URL: http://www.obitko.com/tutorials/genetic-algorithms/ga-basic-description.php
7: Mogendorf, J. J., Stratego, #695.583, 'Official Gazette' 1960.
8: Jumbo International, 'Stratego original, val aan en verover de vlag!'. 2008, page 5-7, URL: http://www.jumbo.eu/media/products/assets/manuals/99a69525-ed0c-503c-b2cf-fa2ecc3625fb.pdf
9: Arts, S, 'Competive Play in Stratego'. 2005, URL: http://www.unimaas.nl/games/files/msc/Arts_thesis.pdf
10: De Boer, V., 'Invincible A Stratego Bot'. 2007, URL:http://www.kbs.twi.tudelft.nl/docs/MSc/2007/deBoer/thesis.pdf
11: Albarelli, R., 'Optimizing Stratego Heuristics With Genetic Algorithms'. 2003, DOI: 10.1.1.113.5389
12: Satz, I., 'Probe'. 2011, URL: http://www.probe.imersatz.com/
13: Unknown, '2010 Computer Stratego World Championship'. 2010, URL: http://www.strategousa.org/wiki/index.php/2010_Computer_Stratego_World_Championship
14: Koza, J.R., 'Breeding Populations of Computer Programs to Solve Problems'. 1990, page 8, URL: ftp://reports.stanford.edu/public_html/cstr.old/reports/cs/tr/90/1314/CS-TR-90-1314.pdf
15: Tunru, V., 'Feasibility of Applying a Genetic Algorithm to Playing Stratego'. 2012, page 5, URL: http://ubuntuone.com/1LLSZIgDNkLbmuDmakWrvl
16: Tunru, V., 'Feasibility of Applying a Genetic Algorithm to Playing Stratego'. 2012, page 5, URL: http://ubuntuone.com/1LLSZIgDNkLbmuDmakWrvl
17: Tunru, V., 'Feasibility of Applying a Genetic Algorithm to Playing Stratego'. 2012, page 6, URL: http://ubuntuone.com/1LLSZIgDNkLbmuDmakWrvl

18: Jumbo International, 'Stratego original, val aan en verover de vlag!'. 2008, page 6, URL: http://www.jumbo.eu/media/products/assets/manuals/99a69525-ed0c-503c-b2cf-fa2ecc3625fb.pdf

12 Reflection

Along with ViCKI, I have learnt a lot during this research. Before the project, knowledge on Stratego was thoroughly acquired. Since receiving the title of Female World Champion, I have not been challenged to this extent. When designing a parametric approach, one must explicitly know what the underlying decisions are that make up one's intuition. Knowledge on Stratego was supplemented by the experience in Genetic Algorithms. While the concept has come to my attention during my studies, no concrete project had focused on it. I have learnt to test the algorithm and to tweak to achieve better results. In general I have learnt to set up a research project and experienced the joy of sitting in front of the computer typing scarily much.

13 Access to Code

The programmed code is distributed under the GNU General Public license (http://www.gnu.org/licenses/gpl-2.0.html). Currently, the code can be retrieved from: https://gitorious.org/stratego.