Applying Genetic Algorithms with Speciation for Optimization of Grid Template Pattern Detection in Financial Markets

Tiago Mousinho Martins

Thesis to obtain the Master of Science Degree in Telecommunications and Informatics Engineering

Supervisor: Prof. Rui Fuentecilla Maia Ferreira Neves

Examination Committee
Chairperson: Prof. Ricardo Jorge Fernandes Chaves
Supervisor: Prof. Rui Fuentecilla Maia Ferreira Neves
Member of the Committee: Prof. Pável Pereira Calado

October 2018
Acknowledgments

I would like first to address special thanks to my supervisor, Professor Rui Neves, not only for his guidance, constant feedback and knowledge about the subject, which revealed crucial for the development of this work, but also for giving me freedom of choice to explore and decide over some nuances of the system being developed, not imposing strict deadlines for any specific task, responsibility which was delegated to me, and giving me some additional challenges along the path. All of these clearly enhanced some of my soft and technical skills, which I believe is a step forward into the professional journey ahead of me.

This document marks the end of my 17 year journey as a student so I take this opportunity to acknowledge some people not only for this report as a masters degree final project, but also as the report that ends that journey.

To start, I’d like to thank all the friends, teachers and other people that, each in their own way, helped me over these years to reach this stage.

Then, I also want to express my gratitude in particular to some people: (1) to my family, specially my parents that always put their efforts into helping me through all the stages I’ve been through and never took my academic course as a burden to our family budget, but as an important investment in my future; (2) to my best friends, from the ones I met at the university to the ones I know since I was a child, and particularly to my girlfriend; (3) to all my course mates and other people that I’ve worked with at university, specially the ones with whom I’ve shared a lot of hours solving and overcoming the challenges our course imposed us; (4) my course’s students branch members with whom I spent a great time making Telecommunications and Computer Science an even more referenced area to be studying in.

Finally, a compliment to Instituto Superior Tecnico and its teachers for giving us students many hard challenges along the path, allowing us to leave with a "tool box" filled of many skills and ourselves prepared to face the challenges of our professional life.
Abstract

This thesis presents a new computational finance approach. It combines a pattern recognition technique allied to an evolutionary computation optimization kernel based on Genetic Algorithms. For pattern matching, a template based approach using a fixed size grid of weights is adopted to describe the desired trading patterns, taking not only the closing price into consideration, but also the variation of price in each considered time interval of the time series. The scores assigned to the grid of weights will be optimized by the Genetic Algorithm and, at the same time, the genetic diversity of possible solutions will be preserved using a speciation technique, giving time for individuals to be optimized within their own niche. The adoption of this approach has the goal of reducing the investment risk and check if it outperforms similar approaches. This system was tested against state-of-the-art solutions, namely the existing adaptable grid of weights and a non speciated approach, considering real data from the stock market. The developed approach using the grid of weights had 21.3% of average return over the testing period against 10.9% of the existing approach and the use of speciation improved some of the training results as genetic diversity was taken into consideration.

Keywords: Genetic Algorithm, Speciation, Pattern Discovery, Grid of Weights, Optimization, Financial Markets
Resumo

Esta tese apresenta uma nova abordagem no âmbito da computação financeira através da combinação de uma técnica de reconhecimento de padrões aliada a um procedimento de otimização baseado na computação evolucionária, mais concretamente em Algoritmos Genéticos. Para a detecção de padrões, será adotada uma abordagem baseada em modelos e utilizando uma grelha de pesos fixa para descobrir os padrões de negociação desejados, que terá em conta não só o preço de fecho do mercado, como também as variações do mesmo no tempo em cada intervalo de tempo da série temporal. As pontuações atribuídas aos campos da grelha de pesos serão otimizadas pelo Algoritmo Genético e, ao mesmo tempo, a diversidade genética das diferentes possíveis soluções será preservada usando uma técnica de especiação, dando assim tempo para os indivíduos serem otimizados dentro do seu próprio nicho. A adoção desta abordagem tem como objetivo reduzir o risco de investimento e verificar se supera abordagens semelhantes. O sistema foi testado considerando dados reais do mercado de ações S&P 500 e contra outras soluções existentes, como a grelha de pesos adaptável e uma abordagem sem espécies. A solução desenvolvida usando a grelha de pesos obteve 21.3% de retorno médio sobre o período de teste comparado com os apenas 10.9% da solução existente, e o uso de especiação melhorou alguns dos resultados de treino visto que a diversidade genética foi tomada em consideração.

Palavras-chave: Algoritmo Genético, Especiação, Descoberta de Padrões, Grelha de Pesos, Otimização, Mercados Financeiros
# Contents

<table>
<thead>
<tr>
<th>Acknowledgments</th>
<th>iii</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>v</td>
</tr>
<tr>
<td>Resumo</td>
<td>vii</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xiv</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xv</td>
</tr>
<tr>
<td>Acronyms</td>
<td>xvii</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Background</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Proposed Solution</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Thesis Contribution</td>
<td>2</td>
</tr>
<tr>
<td>1.4 Outline</td>
<td>2</td>
</tr>
<tr>
<td>2 Related Work</td>
<td>3</td>
</tr>
<tr>
<td>2.1 Financial portfolio and portfolio optimization</td>
<td>3</td>
</tr>
<tr>
<td>2.2 Stock market analysis</td>
<td>3</td>
</tr>
<tr>
<td>2.2.1 Technical Indicators</td>
<td>3</td>
</tr>
<tr>
<td>2.3 Optimization Techniques</td>
<td>5</td>
</tr>
<tr>
<td>2.3.1 Genetic Algorithms</td>
<td>5</td>
</tr>
<tr>
<td>2.3.2 Artificial Neural Networks</td>
<td>6</td>
</tr>
<tr>
<td>2.3.3 NeuroEvolution</td>
<td>6</td>
</tr>
<tr>
<td>2.3.4 NeuroEvolution of Augmenting Topologies</td>
<td>7</td>
</tr>
<tr>
<td>2.3.5 Speciation</td>
<td>8</td>
</tr>
<tr>
<td>2.4 Pattern Detection</td>
<td>11</td>
</tr>
<tr>
<td>2.4.1 Chart Patterns</td>
<td>11</td>
</tr>
<tr>
<td>2.4.2 Detection of Perceptually Important Points</td>
<td>15</td>
</tr>
<tr>
<td>2.4.3 Application of Templates</td>
<td>16</td>
</tr>
<tr>
<td>2.4.4 Application of Rules</td>
<td>17</td>
</tr>
<tr>
<td>A</td>
<td>Installation Manual</td>
</tr>
<tr>
<td>B</td>
<td>Execution Manual</td>
</tr>
</tbody>
</table>
List of Figures

2.1 Candlestick representation. ........................................ 12
2.2 Graphic representation of Ascending and Descending Triangles charts. .......... 12
2.3 Graphic representation of Head and Shoulder Bottoms and Tops charts. ......... 13
2.4 Graphic representation of Rising and Falling Wedges charts. ....................... 13
2.5 Graphic representation of Flag and Pennant charts. ............................... 14
2.6 Graphic representation of Diamond Tops and Bottoms charts. ..................... 14
2.7 Graphic representation of Horn Tops and Bottoms charts. ......................... 15
2.8 Graphic representation of Horn Tops and Bottoms charts. ......................... 15
2.9 Bull Flag template from (Leigh, Paz, & Russell, 2002). .......................... 16
2.10 Bull Flag template from (Cervelló-Royo, Guijarro, & Michniuk, 2015). ........ 17

3.1 System’s architecture by components. ................................... 19
3.2 Commands issued to run the program or present results. ......................... 21
3.3 CSV file from NFLX stock. ........................................... 22
3.4 Flow of data chart during program execution. .................................. 23
3.5 Output files organization. ............................................. 24
3.6 Difference between an adaptable and a fixed size grid of weights. ............... 25
3.7 Constraint positioning of the first column of the grid. ............................ 26
3.8 Illustration of weighted scores procedure. ................................... 27
3.9 Genetic Algorithm schema. ............................................ 29
3.10 Genetic Encoding of the Grid and Weights: a) Normal grid representation; b) Chromosome’s genetic encoding representation. ......................... 29

4.1 Constraint positioning of the first column of the grid and grid offset. ............ 37
4.2 Performance comparison graphs of Test Scenario 1. ................................ 38
4.3 Extract from ADI stock during testing phase of FSGW. ............................ 39
4.4 Extract from ADI stock during testing phase of TSAG. ............................. 40
4.5 Extract from CHK stock during testing phase of FSGW. ............................ 40
4.6 Extract from CHK stock during testing phase of TSAG. ............................ 41
4.7 FE stock during testing phase of FSGW. .................................... 41
4.8 Extract from PNR stock during testing phase of FSGW. ............................ 42
## List of Tables

2.1 Results of some relevant studies for this report. ............................................. 18

3.1 Examples of crossover operation using the general rules for possible combinations. ... 32

4.1 General setup parameters for test scenarios. .................................................. 36

4.2 Evolutionary parameters of Test Scenario 1. .................................................. 38

4.3 Performance comparison metrics of Test Scenario 1. ...................................... 38

4.4 Evolutionary parameters of Test Scenario 2. .................................................. 43

4.5 Performance comparison metrics of Test Scenario 2. ...................................... 44

4.6 Comparison between FSGW 20-10’s worst positions against normal FSGW. ........ 45

4.7 Evolutionary parameters of Test Scenario 3. .................................................. 47

4.8 Performance comparison metrics of training set regarding Test Scenario 3. .......... 47

4.9 Performance comparison metrics of testing set regarding Test Scenario 3. .......... 47

4.10 Comparison between FSGW run chosen for test scenarios against the average of 10 executed runs. ................................................................. 51
List of Acronyms

**ANN**  Artificial Neural Networks

**ATR**  Average True Range

**API**  Application Programming Interface

**DEAP**  Distributed Evolutionary Algorithms in Python

**EMA**  Exponential Moving Average

**FSGW**  Fixed Size Grid of Weights

**GA**  Genetic Algorithm

**HMA**  Hull Moving Average

**MACD**  Moving Average Convergence Divergence

**NE**  NeuroEvolution

**NEAT**  NeuroEvolution of Augmenting Topologies

**OBV**  On Balance Volume

**PIPs**  Perceptually Important Points

**ROI**  Return on Investment

**RSI**  Relative Strength Index

**SMA**  Simple Moving Average

**TSAG**  Traditional Simple Adaptable Grid

**VIX**  Chicago Board Options Exchange Volatility Index

**SMTP**  Simple Mail Transfer Protocol
Chapter 1

Introduction

The domain of computational finance has been earning increased attention by people from both computational intelligence and finance domains. This happens because, when studying the market, determining the optimal timings to buy or sell stocks is crucial in achieving a trading strategy that will accomplish the financial performance requirements of the investor. However, this process tends to be very difficult as the stock market is noisy, non-stationary and non-linear, and at the same time influenced by many other factors, such as political, economic and even psychological variables.

1.1 Background

The efficient market hypothesis [1] presents a view in which is not possible beat the market because it already includes all available information at a specific moment, as well as defends that stock prices behave in a random walk, being the prediction of future prices impossible as they cannot be reflected by any trend or pattern. Besides that, it is also stated that the passive investing strategy Buy and Hold (B&H), where an investor buys stocks and holds them for a long amount of time, despite of market fluctuations, is the more adequate strategy.

Adversely to this vision, some investigators support the thesis that markets are not efficient, existing some variables that they do not reflect right away, which opens a window of opportunity to beat them and obtain returns using stock market forecasting techniques, such as technical analysis, fundamental analysis, and analysis of time series. To achieve this, investment strategies that are able to process large amounts of data and generate appropriate trading signals (buy or sell) must be defined, being the main challenges of the computational finance domain trying to predict future trend prices, in order to obtain the best correlation between risk (that should be as low as possible) and profit (that should be as high as possible).

One way of predicting the behavior of the market used by traders is through the analysis and identification of chart patterns in historical prices of financial assets. A problem arises with this approach regarding the way the market should match a specific chart pattern to make a buy or sell decision. Several works like [2] [3] [4] and [5] have been published where different approaches to deal with the chart
pattern identification are proposed and tested.

1.2 Proposed Solution

In this report a new approach to pattern identification is presented, combining the template based approach using a grid of weights, proposed by [3], and a GA [6]. The grid of weights assigned for a specific template and placed above the signal is going to generate a score that will be higher as close as the signal is from the pattern. The GA will be used to optimize the weights assigned to each cell of the grid so the pattern identification can be more precise, avoiding false positives and false negatives.

1.3 Thesis Contribution

The main contributions of the system proposed in this report are:

1. The creation of a new way to attribute the score to the signal, in which not only the closing price is taken into consideration, but also the price fluctuations over each time interval of the time series, in order to retrieve more information from it and improve the accuracy of the algorithm’s decision;

2. The fact that grid scores are not fixed but instead optimized by the GA in order to find the most profitable solution;

3. The elimination of the adaptability of the grid imposed by the previous approach, in which the range of prices of the sliding window would decide how the size of the grid would vary, into a scenario where fixed dimensions are attributed to the grid prior to its use in order to ensure that all detected patterns are identical, not only to the template, but also within each other;

4. The preservation of genetic diversity in the population being optimized by the GA, through the use of a speciation method, in order to give time for different genotypic solutions to be optimized within their own niche.

1.4 Outline

This document describes the research and work developed and it is organized as follows:

- **Chapter 1** presents the motivation, background and proposed solution;
- **Chapter 2** describes the previous work in the field;
- **Chapter 3** describes the system architecture and its requirements;
- **Chapter 4** describes the evaluation tests performed and the corresponding results;
- **Chapter 5** summarizes the work developed and future work.
Chapter 2

Related Work

2.1 Financial portfolio and portfolio optimization

A financial portfolio consists in allocating wealth among several financial assets, also called securities or investments, i.e. stocks. The portfolio optimization problem is one of the most important research themes in recent risk management studies, especially regarding its two most relevant parameters: risk, which is intended to be minimized as much as possible, and return, that should be as large as possible. Nevertheless, these two parameters are not inversely proportional as we would wish, but instead a high return is always associated with higher risk [7] [8].

2.2 Stock market analysis

Regarding stock market analysis, two main approaches arise: fundamental analysis and technical analysis. Fundamental analysis takes advantage of available data about companies to understand which ones are on the same competitive industry and measure the intrinsic value of a stock, in order to take conclusions on which is the best to invest, based on ratios that measure profitability, liquidity, debt and growth. Technical analysis does not attempt to measure the intrinsic value of a stock, but instead uses charts with historical data to identify trends and patterns and tries to predict what the stock may do in the future. This is done through the use of technical indicators that will define the behavior of the program [9].

2.2.1 Technical Indicators

A technical indicator is a metric usually extracted from the price of an asset and its main goal is to help predicting future prices or incoming trends. Some of the most used technical indicators are presented below [7] [10].
Trend Following Indicators

This type of indicators have the main goal of identify shifts in the tendency, namely if a trend has begun or finishing its cycle.

- Simple and Exponential Moving Average - The Simple Moving Average (SMA) is one of the oldest technical indicators, used to find the mean price over a time window. It is calculated simply by averaging the price data over a defined time interval in order to produce a smooth line for an easier analysis, instead of trying to understand the irregular signal of prices. The Exponential Moving Average (EMA), works in the same way as the SMA, but assigns more weight to the most recent data, thus making this average more sensitive to it. For this reason, EMA is much more used than SMA.

- Hull Moving Average - EMA's major problem is related to the buying and selling signals that it generates, as there is a lag between the time an investor should be making a move on the stock and the actual signaling from the average. The Hull Moving Average (HMA) fights that problem defining an even smoother curve that follow the price much more closely than its predecessor, making the generation of signals much more precise.

- Double Crossover - The Double Crossover method consists on combining two different moving averages to generate buy and sell signals i.e. each time they cross each other. It is usually used both a short and long moving averages to take advantage of their characteristics: the shorter is more sensible to the market signal and reacts faster, but can produce false signals; the longer has more delay but produces more accurate results.

Momentum Oscillators and Indicators

These indicators are important to identify turning points and are useful in markets that are not biased i.e. volatile and hardly predictable. The moving averages also identify and follow trends, but using them on these agitated markets might generate false signals.

- Relative Strength Index - The Relative Strength Index (RSI) is a momentum oscillator used to measure the stock’s recent gains against its recent losses and tries to identify overbought or oversold conditions. When calculated, the RSI signal has a range from 0 to 100 and there are specific levels that, if crossed by the RSI line, must generate buy or sell signals.

- Moving Average Convergence Divergence - Moving Average Convergence Divergence (MACD) is a trend following momentum indicator representing the relation between two moving averages. It is stated as one of the most reliable indicators and is defined by two lines: the MACD line (corresponding to the difference between a 12-week EMA and a 26-week EMA) and a trigger line (corresponding to a 9-day EMA applied to the MACD values). A histogram is obtained from the difference between the former lines, based on which buy and sell signals are generated.
• On Balance Volume - The On Balance Volume (OBV) is a momentum indicator that tries to relate the way volume is flowing in a security with the price, assuming that, if price changes alongside volume we are facing a tendency, otherwise it must be an indication that the tendency is about to finish.

Volatility Indicators

Volatility indicators are a reflection of investor sentiment and predictability of signal price. A stock is considered volatile if its signal movements are unstable and with constant changes in its amplitude. In this kind of stocks, the risk associated tends to be higher and the probability of having a big success or a big failure in the investment increases. On the other hand, a low volatile stock is characterized by a stable and predictable signal, reducing the probability of having a big success or failure, which reflects in lower risk for the investment.

• COBE Volatility Index – The Chicago Board Options Exchange Volatility Index (VIX) is commonly introduced as the “fear index” and measures the implied volatility of S&P 500. The VIX is quoted in percentage points and produces the expected range of movement in the previous index over the next year.

• Average True Range – Average True Range (ATR) was designed with commodities and daily prices in mind. These are more volatile than stocks and are subject to gaps and limit moves that would not be captured by volatility formulas based on high-low range. ATR is based on True Range which uses absolute price changes and reflects the enthusiasm behind a move or a breakout [11].

2.3 Optimization Techniques

2.3.1 Genetic Algorithms

In artificial intelligence we find the concept of Evolutionary Algorithms that use mechanisms inspired by biological evolution to find the best individuals in a population. One of the best known evolutionary techniques is the GA, which is based on Darwin’s principle that the fittest survive [6] [12] [13].

GAs start with a randomly generated population where each individual is defined by a set of chromosomes (also called genome) with encoded data in its genes. In the case of portfolio optimization, this data usually represents financial ratio weights or trading parameters. Despite the fact that GAs are randomized, it is not considered a random procedure, as it is follows an iterative process inspired by biological evolution and constituted by several steps, such as evaluation, selection, crossover and mutation, in order to search solutions with good performance. In the evaluation stage, each individual is appraised through the use of an evaluation or fitness function, so the best ones are able to be within the population of the next generation, in a process called elitism selection, which simulates the “survival of the fittest” convention. Once the new generation is set the process continues with a crossover operation,
where two previously chosen individuals, called parents, are merged to create a descendent with characteristics from both of them, with the purpose of combining their best attributes and generate a better performing individual. Finally a mutation is introduced to the chromosome in order to generate some diversity in the population, by randomly changing the genes in the individual. The GA process finishes when a termination condition is achieved, e.g. an established number of generations or a fitness value threshold.

Its capability of optimizing parameters in widely complex datasets makes it extremely indicated to be used in the financial markets to improve the investment strategy.

2.3.2 Artificial Neural Networks

Artificial Neural Networks (ANN) are computational models inspired by biological behaviour and adaptive capabilities of central nervous systems. GAs are used for supervised regression problems and are capable of solving non-linear problems with high complexity, being characterized for having a learning phase using existing datasets for predicting future results.

They are composed by interconnected processing units, called nodes or neurons, and they can be classified as input, output and hidden nodes. Input nodes receive the dataset of inputs of the model, output nodes play the role of outputting the predicted result, being their number usually the same as the classes to predict, and hidden nodes process information provided by input nodes, serving as an intermediary between input and output nodes, and their number is dependent of the task of the model [14].

The channels that connect neurons are called synapses and their job consists on taking a value from their input node, multiply it by a specific weight (called synaptic weight) and output the result to their output node. Each neuron has an activation function that is applied to the output of all its synapses and allows the neural network to introduce complex non-linear dynamic patterns into the model, like the ones we find in the stock markets.

In the learning phase, the backpropagation method [15] is used to train the synaptic weights of the network, minimizing the cost function by adapting weights. The cost function calculates the difference between the desired output and the network output.

2.3.3 NeuroEvolution

When trying to solve a particular task using ANNs, its architecture and the way the model is trained are important decisions to be made. NeuroEvolution (NE) tries to solve these problems using GAs to evolve the topology of ANNs [16]. Even though this solution offers several advantages over the backpropagation model, like achieving reasonable solutions in shorter time and being more flexible to factors that are not easily incorporated in the backpropagation method [17], it has two main drawbacks: the competing conventions problem and the topological innovation problem.
Competing Conventions Problem

A well-known problem of NE is the Competing Conventions Problem [18], in which two neural networks that order their hidden nodes differently might still be functionally equivalent. This represents a problem because these different representations can produce a flawed offspring during crossover. For example, crossing [A, B, C] with [C, B, A] using single-point crossover, in which one random point is chosen to split each genome and one of the parts is switched between both of them, will produce [C, B, C] and [A, B, A], both losing one third of important information to the final solution. Therefore these damaged solutions will be evaluated by the algorithm needlessly, increasing the computation time inherent to this process.

Topological Innovation Problem

Another problem must be taken into consideration if we need to adjust the existing structure, e.g. adding a new connection to the ANN. Such change will decrease fitness before the connection weight has time to optimize itself and reproduce in order for its children to develop a subsequent structural development that can be promising.

2.3.4 NeuroEvolution of Augmenting Topologies

The NeuroEvolution of Augmenting Topologies (NEAT) algorithm, developed by Ken Stanley, seeks to solve some of the drawbacks of previous NeuroEvolution approaches. NEAT proves to be effective in preserving structural innovation using historical markers, introducing speciation and by applying the principle of incremental growth from minimal structure, through a process called complexification [19].

During crossover, each genome is aligned based on its historical markers: matching genes if they share an ancestral origin, disjoint or excess genes if they exist, respectively, inside or outside the innovation numbers (unique numbers that are incremented for each new gene created) of their parents. Thus, two different parents will manage to find their structure similarities without expensive topological analysis and the extra genes are always inherited from the fittest parent. This process is focused on fighting the competing conventions problem as the functional parameters of each individual are preserved and offspring are undamaged.

Historical markers play another important role in speciation. Topological innovation can be protected this way, by ensuring genomes only compete within a population niche, protecting innovative structures. Two organisms are said to be in the same species if they represent topologically similar networks. The topological disparity between two neural networks can be quantified through the number of excess and disjoint nodes of both genomes and a compatibility function can be employed with a threshold to decide whether two fit individuals are similar enough. If not a new species can be created to give time for both topologies to evolve and compete with similar ones, avoiding being early discarded.

Stanley also defended that starting with random topologies does not lead to finding minimal solutions, since the population already starts with unnecessary nodes, thus starting with minimal structure allows
the algorithm to search for the solution in a low-dimensional search space, which improves performance (a process called complexification).

In finance, NEAT has already been applied to stock prediction [20] and, even though the results were not entirely conclusive, in three specific stocks (Apple, Microsoft and Yahoo) it outperformed backpropagation strategy.

2.3.5 Speciation

A GA seeks to find an optimal solution based on an optimization function, after multiple iterations over a procedure of selection, crossover and mutation, as explained before in this report. However this approach brings along a certain loss of genetic diversity, as inevitably the population will converge towards one fit individual and its close mutants, leading to that loss of genetic diversity. In nature, an ecosystem is composed by different physical spaces (niches) exhibiting different features, which allows both the formation and maintenance of different types of life that compete to survive (species). For each niche, the physical resources are finite and must be shared among the individuals belonging to it. This promotes genetic diversity by encouraging the emergence of species in different sub-spaces of the environment (niches) and we experience a wide variety of species co-existing within an ecosystem, evolving at the same time and competing using the environmental resources [21].

The main goal of carrying this concept to the Evolutionary Algorithms relies on the fact that, given some value calculated over the parameters of a particular genome, a threshold could be set to consider that two not looking alike individuals in the population must not be crossing over each other. This way, each individual will compete against individuals of the same species and each will have time evolve their fittest individuals for an optimal solution that took into account genetic diversity. A set of existing speciation methods is presented next.

NEAT and Historical Markers

Introduced in [19] and as explained before in this report, historical markers identify the original ancestor of each gene. When comparing a pair of genes, the disjoint genes (D), excess genes (E) and the average weight difference of matching genes (W) are taken into account to calculate a compatibility distance between the different structures, which is a simple linear combination of them:

\[
\delta = \frac{c_1 \times E}{N} + \frac{c_2 \times D}{N} + c_3 \times \overline{W}
\]  

(2.1)

where \(N\) is the number of genes in the larger genome of the population and the coefficients \(c_1, c_2\) and \(c_3\) allow to adjust the relative importance of the three factors (E, D and \(\overline{W}\)). By setting a compatibility threshold, the result of applying this measure will decide in which existing species it fits, or if a new one must be created.
**Taxon-exemplar scheme**

Brooker [22] used taxon and exemplars in his learning algorithm to create a restricted mating policy.  
A taxon is a string constructed with the alphabet 1, 0, #, where # matches both a 0 or a 1, and each individual is initialized in the population with it. In his policy, he wanted to restrict mating among similar taxon strings, which are identified by a score calculated between them and a given exemplar binary string. The score is calculated based on the correct matches and the number of #’s in the taxon.  
If a certain number of matching taxon strings (strings with a score matching a threshold score) are available within the population, parent strings are randomly chosen from this group, otherwise parent strings are chosen according to a probability distribution, calculated based on the matching score of taxon strings.

**The tag-template method**

Similar to the previous method, a template and tag strings are introduced [23]. The template string is constructed the same way as the taxon, but the tag string is binary, having both the same size. They are created at random in the initial population, along with the functional string, and have the same purpose as the taxon strings: before crossing a pair of individuals, tag and template strings are compared, a matching score is calculated and their mating depends on that score. The difference between this method and the one before stands for its purpose, as tag and template strings corresponding to the fittest individuals in early populations are highlighted and, later on, crossover is only performed between matching strings of each peak.

**Phenotypic and genotypic mating restriction**

Deb [23] has also developed two mating restriction schemes based on the phenotypic and genotypic distance, which are respectively calculated through Euclidean distance and Hamming distance. To choose a mate for an individual in the population, one of these distances is determined and if it is close to a given parameter they will engage in crossover, otherwise another individual is chosen and the distance is computed again. This process goes on until a fitting mate is found and, if this does not happen, the choice is made randomly. These mating restrictions were developed to be applied in multi-modal test problems.

**Fitness Sharing**

One of the most widely used niching methods that treats fitness as a resource to force all individuals within a niche to share a similar fitness between them. This is accomplished by using an adjusted fitness, called fitness sharing, which reduces the fitness of each individual by an amount approximately equal to the number of similar individuals in the population, using a defined set of parameters to measure the similarity between individuals. Formally, the shared fitness $\varphi_{sh,t}(\tau)$ of an individual $\tau$ is computed according to the following formula:
\[ \phi_{sh,t}(\tau) = \frac{\phi(\tau)}{m_t(\tau)} \]  

where \( \phi(\tau) \) is the raw fitness and \( m(\tau) \) is an estimate of the number of individuals belonging to the same niche.

However, this method comes with the drawback that setting proper values for these parameters implicates the need of some deep theoretical knowledge about the fitness landscape [24].

Dynamic Niche Sharing

First method to introduce a mechanism to dynamically classify species among the population during the evolution process [25]. At each generation, an algorithm called Dynamic Peak Identification pinpoints the existing niches in the population and classifies new individuals as belonging to one of those, or a special division called nonspecies. Then two fitness sharing methods are applied: one to update the fitness sharing value of the individuals in each niche and the standard one to be applied to the individuals of the nonspecies class. The main drawback of this approach relies on the fact that the number of peaks in the fitness landscape must be previously known.

Dynamic Niche Clustering

A method introduced by [26] that uses the concept of nicheset. Each niche is defined by some parameters that change during the evolution process and the method allows for an individual to be a member of more than one niche, which means that they may even overlap in the search space, although some parameters are delimited to avoid the excessive growth or decrease of the number of niches along the process. This overlapping of niches is a controlled process called striation that does not allow niches to converge by penalizing members belonging to both niches, even though an improved version of this method introduces a function to decide whether niches must be merged or split during evolutionary process [27]. Overall, the method relies on the assumption that the search space must have bounds of the same magnitude, otherwise niche radius can be wrongly calculated, and the initial population stands as an important aspect to define the bounds of the niche radius. The flexibility of not constraining individuals to the apex of the niche makes natural that their fitness is lightly reduced compared to other techniques.

Adaptive Species Discovery

A niching method that removes the premise of perfect discrimination which serves as base for niching techniques, like Fitness Sharing, and is more effective in handling with irregular spaced peaks in a multi-modal landscape. Like Dynamic Niche Sharing, this method does not require any a priori knowledge, but complements it as information about the fitness landscape is not needed as well. Each niche is identified by a representative individual, which is chosen among the members, and the explicit knowledge about the number of niches and their positions in the search space grants a way to preserve species masters between generations, in an elitist process that maintains the number of niches. The discovery process is
based upon a set of information dynamically acquired at each generation, through a Hill-Valley function [28], which allows the detection of genetic divergence within the population. The function was first adopted to verify whether two points lie or not in the same peak in the landscape, however it also grants the detection of genetic divergence phenomena within an artificial population, being the main idea of adopting this function to prevent individuals with very different genotypic features from being considered members of the same species [21].

2.4 Pattern Detection

One of the most difficult tasks for investors is to analyze financial charts to detect patterns. Three main concepts must be distinguished regarding this subject: time series, pattern recognition and pattern discovery.

A time series stands for a collection of observations chronologically made [2], while in pattern recognition the goal is to identify known patterns on the time series and pattern discovery consists in finding new patterns that occur in the time series and are unknown [29].

The appearance of chart patterns in financial time series is often a premeditation of a possible change in price trend. For the purpose of identifying these patterns, various approaches can be used to measure the similarities between chart pattern templates and sequences extracted from financial time series. Despite of the method chosen for pattern matching, there is not an ambiguous standard for how these patterns should be defined. Some well-known patterns are frequently used to evaluate the effectiveness pattern matching methods, but, once again, as there are no precise definitions for these chart patterns, researchers and financial analysts may specify them in the most suitable way for their applications.

2.4.1 Chart Patterns

In a financial time series, some similar graphic representations appear over time and, as these graphics are representative of behavior patterns of investors, as well as the market performance is influenced by them, some of these similarities are good warning signs on what the market will do next. Patterns can be characterized as variable or fixed fluctuation patterns, whenever they have an uncertain or fixed number of highs and lows respectively. They can also be identified over the continuous market signal line or in the candlestick graphic representation and most of them can be derived into two patterns of similar behavior, except for the fact that one represent an uptrend or ascending movement and the other a downtrend or descending movement, i.e. tops and bottoms, which are formations that represent a price level of resistance and support, indicating a turnabout in the trend.

Patterns can be found within the continuous signal of the stock market, which indicates the closing price in each time interval, or within the candlestick representation signal, which, as shown in Figure 2.1, also indicates the opening price, highest value and lowest value, besides the closing price.

Taking this into account, some the most known, used and reliable chart patterns are presented next [30] [31].
Figure 2.1: Candlestick representation.

**Ascending and Descending Triangle**

This pattern is characterized by having an horizontal top and an up-sloping bottom trend lines, if it is the ascending triangle, or a down-sloping top and horizontal bottom trend lines, in case of being a descending triangle. Ascending triangles usually confirm an uptrend and are more reliable when found in bullish market conditions, while descending triangles appear to support a downtrend and are more reliable in bearish market conditions. Figure 2.2 shows the graphic representation of *Ascending and Descending Triangles*.

Figure 2.2: Graphic representation of Ascending and Descending Triangles charts.

**Head and Shoulders Tops and Bottoms**

This pattern can have multiple heads or multiple shoulders (rarely both) when it represents bottoms, being the head(s) always lower than the shoulders. In the case of being a head and shoulders tops, it has multiple shoulders and they must mirror themselves related to the head, being this last one always higher than the shoulders. It is a reversal type and comes along with an indication of change in trend. Figure 2.3 shows the graphic representation of *Head and Shoulders Tops and Bottoms*.
Rising and Falling Wedges

This pattern is characterized by two trend lines that must both slope upward or downward, depending if we are considering rising or falling wedges respectively, where in the first case the bottom trend line must be steeper than the top trend line and otherwise in the second case. The minimum duration of these patterns is 3 weeks and usually don’t last more than 3 months. Rising wedges are associated with bearish market while falling wedges tend to appear more in a bullish market. Figure 2.4 shows the graphic representation of Rising and Falling Wedges.

Flags and Pennants

Both patterns are followed by ascent or descent burst, whether they represent a confirmation of an uptrend or downtrend, respectively. In the case of flags depending on whether we are considering a bullish or bearish market, the pattern oscillates between two horizontal lines with an upward or downward slope. The flags are often called Bull Flag or Bear Flag depending on the direction of the tendency. On the other hand, the pattern in the pennants oscillates between two intersecting trend lines, one sloping downward from the top and the other sloping upward from the bottom. Figure 2.5 shows the graphic representation of a Flag and a Pennant.
Diamond Tops and Bottoms

Both patterns are characterized by having higher top points and lower floor points in the first part of the pattern, followed by lower top points and higher floor points in the second part of the pattern. The difference between them relies on the start price trend. A diamond tops pattern has its start price in an upward trend and the diamond bottoms pattern is characterized by a starting price in a downward trend. Figure 2.6 shows the graphic representation of Diamond Tops and Bottoms.

Horn Tops and Bottoms

Both patterns are formed by five weekly candlesticks and the two upward or downward spikes, whether we are referring to tops or bottoms respectively, should be longer than similar spikes over the last 48 weeks and separated by a smaller stick between them. The pattern is confirmed when the lowest price of stick 5 closes below the lowest low, in case of horn tops, or above the highest high, in case of horn bottoms. Figure 2.7 shows the graphic representation of Horn Tops and Bottoms.
Pipe Tops and Bottoms

Both patterns are formed by four weekly candlesticks and have two adjacent upward or downward price spikes, whether we are referring to tops or bottoms respectively. They both should be longer than similar spikes over the last 48 weeks and the pattern is confirmed when the price closes below their lowest low, in case of pipe tops, or above their highest high, in case of pipe bottoms. Figure 2.8 shows the graphic representation of Pipe Tops and Bottoms.

2.4.2 Detection of Perceptually Important Points

One of time series pattern matching techniques is Perceptually Important Points (PIPs) [32], which allows a big reduction in the dimension of time series, maintaining the main characteristics of data. These points are important for the identification of patterns as they stand for the ones that characterize the time series. Pattern recognition after defining the PIPs can be done using two techniques: template-based or rule-based. The template-based approach allows a point-to-point comparison between time series and patterns, as the rule-based approach stands for the explicit relation between time series and patterns [4].
2.4.3 Application of Templates

The most straightforward way to achieve a comparison measure between two patterns is through template matching. By defining the shape of query patterns (pattern templates) visually, point-to-point direct comparisons can be performed [2]. However, different segments of a time series can have different amplitudes so, for instance, after identifying the PIPs a normalization is necessary in order to simplify the identification of identical patterns. Then, a direct point-to-point comparison can be done between the input sequence of points and the query sequence representative of the template.

Grid of Weights

One other way to identify patterns through the application of templates is using a grid of weights [3], which stands for a 10x10 grid, like the one in Figure 2.9, which allows the recognition of patterns by assigning higher classification to the cells of the grid matching the template, and by defining a minimum fit value threshold for which the sum of scores of the cells matching the close price signal must achieve.

This has the disadvantage of being a subjective process in which the threshold value must be adjusted, taking into consideration that higher values can miss price windows that fit the pattern and lower values can lead to the identification of false patterns. However, this subjectivity combined with the fact that the weights configuration can be changed, may grant the possibility of going further after the needs that each specific template require to be identified with more precision, even allowing to take into account not only closing prices but also highs, lows and opening prices.

![Figure 2.9: Bull Flag template from (Leigh, Paz, & Russell, 2002).](image)

In finance, some works have been developed relying on this representation for pattern detection. Cervelló-Royo [5] took what was said before into consideration and developed a grid with weights configured to support a scheme where the body of candlesticks were used to detect the Bull Flag pattern, as shown in Figure 2.10. The price should always visit the cell labeled 5 and if the threshold was set to 3, the body of the candlesticks should only visit negative cells twice. Parracho [33] used only the closing price, but tested this representation with 4 different patterns (uptrend, downtrend, breakout and bull
flag) combined between them. Both approaches showed results that outran the market and Parracho’s version of the grid also outperformed Buy & Hold.

<table>
<thead>
<tr>
<th>5</th>
<th>-1</th>
<th>-5</th>
<th>-5</th>
<th>-5</th>
<th>-5</th>
<th>-5</th>
<th>-5</th>
<th>-5</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>-5</td>
<td>-5</td>
<td>-5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>-5</td>
<td>-5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>-5</td>
</tr>
</tbody>
</table>

Figure 2.10: Bull Flag template from (Cervelló-Royo, Guijarro, & Michniuk, 2015).

### 2.4.4 Application of Rules

Through this approach, rules are defined to characterize the shape of wanted patterns. One of the disadvantages of template based approach is, as stated before, that the relationship between points is hard to define unequivocally and sometimes imprecise, so this methodology claims to overcome that problem specifying the relative positions of the points composing the pattern, e.g., for a Head and Shoulders pattern, both shoulders must be at a similar height and at a lower level than the head.

It is not possible to guarantee the reliability of these specifications with a template based approach because similar shaped patterns may be found that violate these rules. This way, given a certain number of points (e.g. PIPs) the matching pattern is depending on the validation of all rules.

Despite this technique has overall poor matching results for the majority of patterns, some specific ones like Head and Shoulders and Triple Top can be detected with distinction [2].

### 2.5 Chapter Conclusion

In Table 2.5 are presented the results of some relevant studies for the developed thesis, which apply some of the previous presented techniques to the stock market. Notice that presented results are not sorted in any particular order as it is difficult to rank them while being applied over different markets and testing periods. Although these comparisons are difficult to make, for each particular case, either using GA [29] [33], speciation [20], grid template detection approach [3] [5] [33] or other template detection approaches [4] [2], the results showed improvements against competitor methodologies. Based on these results, the main intent of this work is to develop an algorithm that aggregates and takes advantage of some aspects of all these approaches: combination of GA, speciation and pattern detection using grid template approach.
Table 2.1: Results of some relevant studies for this report.

<table>
<thead>
<tr>
<th>Ref &amp; Year</th>
<th>Method</th>
<th>Financial Data</th>
<th>Period</th>
<th>Algorithm Performance</th>
<th>Comparator Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>[20]</td>
<td>NEAT</td>
<td>Apple / Microsoft / Stock Price</td>
<td>2012 - 2014</td>
<td>36.22% / 19.33% / 20.02%</td>
<td>22.10% / 0.50% / 16.27%</td>
</tr>
<tr>
<td>[29]</td>
<td>SAX + GA</td>
<td>S&amp;P 500 Stock Price</td>
<td>1998 - 2010</td>
<td>6.7% / 63% / 4%</td>
<td>0.32% / 0.44% / 3.70%</td>
</tr>
<tr>
<td>[33]</td>
<td>Uptrend &amp; Downtrend pattern w/ Grid Template + GA</td>
<td>S&amp;P 500 Stock Price</td>
<td>1998 - 2010</td>
<td>36.92% / 16.33% / 10.02%</td>
<td>-4.69% / 0.12% / -1.41%</td>
</tr>
</tbody>
</table>

Note: The table includes studies with different methods and financial data, comparing algorithm performance to market performance.
Chapter 3

Architecture

3.1 System’s Architecture

To determine optimal timings for entering or leaving a specific market based on the analysis of template chart patterns optimized using a GA with speciation, while taking also into consideration the risk associated with the capital invested, Figure 3.1 presents the overall overview of the system’s architecture designed for that purpose.

![Diagram](image)

Figure 3.1: System’s architecture by components.

It follows the most common logical multi-layered architecture composed by three specific layers: the presentation layer, the application layer and the data layer. This gives the ability to develop and update the technology stack of one layer without impacting the remaining ones, providing flexibility to scale the application, as well ease of maintenance regarding the code base. The execution of the above system can be interpreted through the following steps:
1. User specifies the input parameters to be implanted in the system and used by the lower levels, such as financial data, timing constraints and Trading Algorithm parameters;

2. A component responsible for Pre-Processing the information provided by the user gathers all the needed data from the Financial Data component, normalizes it into the standard representations of the system and populates the adequate data structures to provide all the tools for proper operation of the Trading Algorithm in the next stage;

3. The Trading Algorithm component will be continuously generating a trading signal relying on a template-based pattern detection technique that is trained with GA with speciation and also outputting the most relevant results during its execution;

4. At the end of the system's execution, the results of the Trading Algorithm component will be presented to the user for analysis.

All these architectural components will be addressed with more detail in the following subsections.

3.2 User Interface

The interaction with the program provided for the user is mainly performed via command prompt, as the main goal of this project relies on proving a working trading algorithm and not a user friendly interface rigged with all its heuristics. The user can use it not only to issue inputs to setup and run the algorithm, as well as to choose between various ways of presenting results.

To run the algorithm a set of input parameters are displayed and editable in any way each user finds suitable depending on the analysis intended to be done. These inputs are written in a configuration file that will be read by the program upon starting, and are supported by a manual on prompt, helping users in understanding the meaning of each parameter. These parameters are mostly GA related, as the way some genetic operators act depend on these values, however one in particular is an exception and must be highlighted for the reason that its existence is merely for testing purposes and is related to the program capability of operating in 3 different ways: (1) to simulate the proposed system; (2) to simulate a system without speciation; (3) to simulate the traditional system proposed by [5].

To analyze results, at least one run of the program must have been executed prior to analysis and 3 options are available:

- Export the results into a .xlsx file where relevant information regarding each stock is provided and can be easily manipulated by programs such as Microsoft Excel;

- Display graphs containing the training and testing phases of the algorithm, for a specific run and a specific stock. These graphs contain the candlestick representation of every day in the periods related to each stage of the algorithm, along with indicators for market entries and exits, whether they are positive or negative. The frameworks matplotlib and mpl_finance were used for this purpose.
If a run of the algorithm using speciation was made, an horizontal graph bar can be displayed to visually illustrate the evolution of the fittest individual of each specie throughout all the generations. *Matplotlib* also supports this feature of the program.

Figure 3.2 presents a figurative example of running commands to execute the program or analyze results.

![Figure 3.2: Commands issued to run the program or present results.](image)

### 3.3 Financial Data

The program obtains and stores data from companies that is going to be used by the Trading Algorithm component. The process of collecting data uses a python module called *alphavantage*, a provider of APIs for real time and historical data on stocks. The set of companies used in this research belong to Standard and Poor’s 500 (S&P 500) and the API allows the download of daily data from the companies since the year 2000 up to the current day. Each data entry consists of a set of opening and closing prices, highs and lows, and volumes of transactions during each considered period.

Regarding the storage of data, it is made locally in Comma Separated Values (CVS) files, as the one shown in Figure 3.3 regarding Netflix (NFLX) stock, which are exported and imported to the program using another Python module named *pandas*, widely used for manipulating complex data collections within Python. Some mechanisms are implemented to extract specific subsets of data, which gives
flexibility for the program to explore various scenarios using different time intervals of data. Calls to the API are only made if the desired data is not yet in local storage.

<table>
<thead>
<tr>
<th>Date</th>
<th>Open</th>
<th>High</th>
<th>Low</th>
<th>Close</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002-05-23</td>
<td>16.19</td>
<td>17.1</td>
<td>16.04</td>
<td>16.75</td>
<td>7485900.0</td>
</tr>
<tr>
<td>2002-05-24</td>
<td>17.0</td>
<td>17.15</td>
<td>16.76</td>
<td>16.94</td>
<td>992200.0</td>
</tr>
<tr>
<td>2002-05-25</td>
<td>16.99</td>
<td>17.25</td>
<td>16.2</td>
<td>16.2</td>
<td>472000.0</td>
</tr>
<tr>
<td>2002-05-26</td>
<td>16.3</td>
<td>16.1</td>
<td>15.2</td>
<td>15.45</td>
<td>482700.0</td>
</tr>
<tr>
<td>2002-05-27</td>
<td>15.51</td>
<td>15.51</td>
<td>15.0</td>
<td>15.0</td>
<td>725300.0</td>
</tr>
<tr>
<td>2002-05-28</td>
<td>15.1</td>
<td>15.1</td>
<td>15.0</td>
<td>15.07</td>
<td>604600.0</td>
</tr>
<tr>
<td>2002-06-03</td>
<td>15.12</td>
<td>16.05</td>
<td>15.07</td>
<td>15.8</td>
<td>225100.0</td>
</tr>
<tr>
<td>2002-06-04</td>
<td>15.9</td>
<td>15.96</td>
<td>15.85</td>
<td>15.85</td>
<td>231800.0</td>
</tr>
<tr>
<td>2002-06-05</td>
<td>15.35</td>
<td>16.23</td>
<td>15.5</td>
<td>15.06</td>
<td>109400.0</td>
</tr>
<tr>
<td>2002-06-06</td>
<td>16.1</td>
<td>17.25</td>
<td>16.08</td>
<td>16.55</td>
<td>164700.0</td>
</tr>
<tr>
<td>2002-06-07</td>
<td>16.49</td>
<td>16.49</td>
<td>15.45</td>
<td>15.66</td>
<td>97800.0</td>
</tr>
<tr>
<td>2002-06-08</td>
<td>15.49</td>
<td>16.45</td>
<td>15.29</td>
<td>16.19</td>
<td>34600.0</td>
</tr>
<tr>
<td>2002-06-09</td>
<td>16.19</td>
<td>16.44</td>
<td>15.8</td>
<td>16.15</td>
<td>73100.0</td>
</tr>
<tr>
<td>2002-06-10</td>
<td>16.15</td>
<td>16.55</td>
<td>15.25</td>
<td>15.3</td>
<td>129500.0</td>
</tr>
<tr>
<td>2002-06-11</td>
<td>15.46</td>
<td>15.71</td>
<td>15.12</td>
<td>15.16</td>
<td>103499.0</td>
</tr>
<tr>
<td>2002-06-12</td>
<td>15.15</td>
<td>15.1</td>
<td>13.72</td>
<td>13.81</td>
<td>51700.0</td>
</tr>
<tr>
<td>2002-06-13</td>
<td>12.19</td>
<td>12.33</td>
<td>12.35</td>
<td>12.91</td>
<td>346300.0</td>
</tr>
<tr>
<td>2002-06-14</td>
<td>13.14</td>
<td>13.15</td>
<td>11.78</td>
<td>12.75</td>
<td>77200.0</td>
</tr>
<tr>
<td>2002-06-15</td>
<td>12.76</td>
<td>12.95</td>
<td>12.5</td>
<td>13.22</td>
<td>391800.0</td>
</tr>
<tr>
<td>2002-06-16</td>
<td>12.6</td>
<td>14.56</td>
<td>13.94</td>
<td>13.14</td>
<td>436500.0</td>
</tr>
<tr>
<td>2002-06-17</td>
<td>12.75</td>
<td>13.55</td>
<td>13.27</td>
<td>13.71</td>
<td>107600.0</td>
</tr>
<tr>
<td>2002-06-18</td>
<td>13.71</td>
<td>14.2</td>
<td>13.4</td>
<td>13.7</td>
<td>103400.0</td>
</tr>
<tr>
<td>2002-06-19</td>
<td>13.84</td>
<td>14.1</td>
<td>13.26</td>
<td>13.5</td>
<td>201500.0</td>
</tr>
<tr>
<td>2002-06-20</td>
<td>13.49</td>
<td>13.55</td>
<td>12.9</td>
<td>13.84</td>
<td>203900.0</td>
</tr>
</tbody>
</table>

Figure 3.3: CSV file from NFLX stock.

### 3.4 Data Flow and Data Processing

In terms of data being introduced, produced and extracted from the program, we can divide it in 3 types:

1. **Input Data**, that contains program variables with predefined values by the user via configuration file;

2. **Main Program**, containing training and testing phases. Training phase is where the GA finds the best individual to be tested in Testing phase, and both together rely on all the input data to generate relevant information that will be withdrawn from the program as output data;

3. **Output Data**, composed by every piece of information that is extracted from the program, either during the Training Phase of the program (to evaluate the outputs of the GA) or after the Testing Phase to reveal the final results.

Figure 3.4 illustrates the entire process described before.

Regarding the production of outputs, for an easier analysis of the results all retrieved data is organized in a folder hierarchy as follows:

- **First layer** - Composed by a folder per stock analyzed by the program;

- **Second layer** - For each analyzed stock, every program output of it is saved separately with dates on each folder, which facilitates subsequent results examinations;
Third layer - For each specific run involving a particular stock, 3 folders are created:

- plots - containing the file plots.txt where the records related to market entries of the best individual are saved. This information is used to plot output graphics and some of the information contained in excel output file is extracted from here as well.

- reports - can contain the file detailed.txt or simplified.txt, depending on the quantity of information the it has. This file contains a full report of the GA stating for each generation information about the live individuals and their returns, the species, the elitists and current best individual, as well as a final more specific insight of the fittest individual returned by the GA. The difference between both files relies on the fact that detailed.txt shows the state of the population between each stage of the generation (selection, crossover, mutation, elitism) and simplified.txt does not.

- returns - contains 3 files (testing.txt, training.txt and stats.txt) with information only used by the excel output file, and also species_evolution.txt containing information about the best individuals of each species in each generation used by the graph that shows the evolution of the species along the algorithm.
3.5 Trading Algorithm

This stands for the critical component of the system as it will contain the most needful mechanisms to detect chart patterns, i.e. the weighted grid alongside the GA, to train/optimize the detection procedure and support the decision of buying or selling positions in the market.

3.5.1 Pattern Detection

The pattern detection technique follows a heuristic based on templates, namely the 10x10 grid of weights approached in Section 2 [3]. Four conditions are being set for each column of the grid for a Bull Flag pattern:

1. the sum of the scores contained in each column must be 0;
2. there must be two cells with a score equal to 1, which is the maximum score allowed;
3. these two cells must serve as an axis of symmetry between the values of the remaining cells;
4. for each cell that is further away from that axis, the score must be lower or equal than the one that is closer.
As an example for this last condition, if in a certain column the third and fourth cells contain the score 1, the second and fifth cells must contain the same value, smaller than 1; first and sixth cells must also contain the same value, smaller than 1 and smaller or equal to the values contained in the second and fifth cells; and for the cells between seventh and tenth positions, the values of each cell must be lower than 1, lower or equal than all first 6 cells, and smaller between them as they are further away from the closest cell belonging to the axis of symmetry i.e. cell 4.

To detect patterns in the traditional way using a grid of weights, first it must be fitted into the price window we want to check and the closing price of each entry will coincide with one specific cell of the grid. The scores of each cell are assigned so the closer the signal is from the pattern, the higher its score is, and the goal of this process is to obtain a fit value (threshold) to show the level of matching between the pattern matrix and the price window. The choice of the threshold value must be done taking into account that low thresholds may lead to false positives and high ones may discard price windows that would fit the pattern, so a middle term must be found which is not always an easy task.

The pattern being detected is the Bull Flag pattern as shown in Figure 2.5. When trying to detect it using the traditional grid of weights, in addition to the issue of choosing the most suitable threshold, the fact that the grid adapts itself to the price window so each point of the signal is within it, i.e. each closing price of the price window being analyzed, is kept within the limits of the grid, may also lead to find false positives. A flag is only a continuation pattern if it forms an angle that is clearly against the current trend, but as it can be observed in Figure 3.6 an adaptable grid that varies in size according to the signal may mislead the correct pattern identification, e.g. in cases where the height of each row of the grid is small and the angle formed between the consolidation phase and continuation of the uptrend is very wide.

Figure 3.6: Difference between an adaptable and a fixed size grid of weights.

To overcome both these problems and increase precision while using a grid of weights to detect patterns in the time series, two modifications to the traditional grid of weights are being made, leading to the creation of the Fixed Size Grid of Weights (FSGW):

1. To ensure a choice of a suitable threshold, more information than just the closing price will be used, so that even using a low threshold value the matchings are more reliable. For that purpose, instead
of using the grid of weights against the continuous signal of the stock market, which indicates only the closing price in each time interval, the candlestick representation is adopted, which gives information about the highest value, lowest value, opening and closing prices for a specific time interval. The specific way this information is used will be explained in detail next.

2. To avoid the inaccurate detection of flags concerning its shape relatively to the ongoing trend, a fixed size grid is adopted, being the length of each grid row set as an input parameter. As shown in Figure 3.7, this grid does not depend on the relative size of the signal within the price window, being the positioning of the first column the only imposed constraint and it must be done in a way that fits the grid into the best possible position within the signal to find a Bull Flag, whether the following columns end up being Bull Flag match or not. This way, all pattern matches will be similar even in its relative size in the time series.

![Figure 3.7: Constraint positioning of the first column of the grid.](image)

Regarding the use of candlestick representation, even though we have more information than the closing price now, not everything is of the same importance for calculating the final score, which is why information must be divided into groups and each must be weighted differently according its importance in the grid column score for that specific time interval.

Formally, consider a cell named $A$, corresponding to where the closing price of the signal hits in that grid column; a set of cells named $B$, corresponding to the cells between where the opening and closing prices hit for a specific column of the grid (excluding the closing price cell); and a set of cells named $C$, corresponding to the cells visited by the signal during that time interval of the price window and excluding cells contained in $A$ and $B$. Consider also three weights $w_1$, $w_2$ and $w_3$, which are assigned to each the previous stated groups of cells, so the final score $\phi$ for that specific column is given as:

$$\phi = A \times w_1 + \sum_{i=0}^{\text{size}(B)} B_i \times w_2 + \sum_{i=0}^{\text{size}(C)} C_i \times w_3$$

(3.1)

being $w_1 > w_2 > w_3$ and $w_1 + w_2 + w_3 = 1$. If $B$ or $C$ are empty sets, their weights ($w_2$ and/or $w_3$) are added to $w_1$ and become 0. The weights $w_1$, $w_2$, and $w_3$ are randomly assigned to each individual by the
algorithm during its creation and optimized by the GA from then on, all this as long as the previous rules are met.

Figure 3.8 illustrates the process described above, where for the same column in the grid, three different candlesticks are presented with the same closing price and considering the following weights: \( w_1 = 0.5, w_2 = 0.3 \) and \( w_3 = 0.2 \).

![Figure 3.8: Illustration of weighted scores procedure.](image)

As it is observable, while taking only closing prices into account the score for each column would be 1 in the traditional way, but if we add to the balance more information than just the closing price, the risk associated with a momentary increase of market volatility (reflected by a big candlestick) is easily tracked as it results in a lower score.

### 3.5.2 API for Genetic Algorithm

The most commonly used Python framework for rapid prototyping and testing of ideas based on GA is called Distributed Evolutionary Algorithms in Python (DEAP) [34]. It has already implemented genetic operators, such as crossover, mutation and selection, and also supports the creation of specific genetic operators (in the scenario of this report a speciation operator would be needed). Besides this, DEAP also offers a wide range of representations for GA’s individuals (List, Array, Set, Dictionary, Tree, Numpy Array, etc).

When using a FSGW, some aspects regarding its representation can be already outlined:

- Each individual will necessarily be composed by a list of various sublists;
- Each sublist can vary in size, depending on the genetic encoding;
- Genetic operators such as crossover and mutation will involve the manipulation of entire sublists or their positions, and the ones chosen can be different in each iteration of the algorithm.
With these pointed out and although DEAP’s features allow adaptation to different representations, the level of complexity of trying to represent and manipulate a FSGW using this framework would be at least as challenging as creating an Application Programming Interface (API) from scratch, which ended up to be the solution found to address this issue.

In the following subsection the genetic encoding used and the way the API manipulates data in each genetic operator will be explained in detail.

Design Decisions

To effectively handle and manipulate all used data structures a hybrid approach using procedure-oriented and object-oriented programming paradigms were used. As individuals and species are the two structures that are the most used and manipulated, both represent objects in the program. On the other end, genetic operators are designed around functions that support them, representing the procedure-oriented part of the program.

Each individual contain an unique identifier, its genetic encoding structure, training Return on Investment (ROI) value, adjusted fitness and the generation in which it was created. Each specie has also an unique identifier, two values that limit which individuals’ compatibility values are accepted within this specie, two lists containing the entire species’ individuals and the elitists and a constant that holds the number of offsprings, i.e. number of individuals allowed to be carried into the next generation for that specific specie. Some of the attributes stated before are for GA purposes and their usefulness will be explained with more detail in the following subsections.

3.5.3 Genetic Algorithm and Speciation

The main goal of using a GA is to find scores for the grid matrix and values for weights $w_1$, $w_2$, and $w_3$ that maximize the ROI in the training period. The resulting grid from this optimization procedure will then be applied in the testing period. Figure 3.9 presents a schema of the GA information flow using system’s genetic encoding and genetic operator’s way of manipulating individuals.

Genetic Encoding

As the 10x10 grid being used to represent each individual of the population is composed by 100 cells, it would be appropriate to find a way of saving computational resources. Therefore, instead of having a data structure containing all the scores of each position of the grid, it is a better approach to take advantage of the symmetry property stated before.

The genetic encoding is then going to use the following representation: for each column of the grid, the first positions will contain two values corresponding to the number of cells above and below the ones that represent the axis of symmetry; the following positions will contain all different scores the respective column contains; the last three positions after all the columns have been handled will contain the weights $w_1$, $w_2$, and $w_3$. 
For the API, an individual is represented by a list with 11 positions, being the first 10 composed by elements of each column of the grid and the last a dictionary containing the 3 weights associated to it. Each column is also a list, being its first position a sublist of it. As stated before, it contains two values representing the number of cells above and below the axis of symmetry and the highest of this values determine the size of the list for that column. The remaining positions of the column list are filled with each different score that column holds.

Figure 3.10 serves as example to illustrate how, for the case of the Bull Flag pattern detection grid, we can build a genetic encoding representing a whole grid of 100 cells, saving 23 positions in the structure of each individual.

Figure 3.9: Genetic Algorithm schema.

Figure 3.10: Genetic Encoding of the Grid and Weights: a) Normal grid representation; b) Chromosome’s genetic encoding representation.

Speciation

The goal of using speciation in this work is to maintain diversity in the population and protect new individuals against biased solutions that optimize faster. For this purpose, a compatibility function will
be used taking into account the weights that have been assigned to each individual. The candidates will then be divided in species, according to a compatibility value calculated using the following equation:

$$\delta = w_1 \times A + w_2 \times B + w_3 \times C$$  \hspace{1cm} (3.2)$$

where $w_1$, $w_2$, and $w_3$ correspond to the weights of each individual and $A$, $B$ and $C$ are constants set to 0.65, 0.225 and 0.125 respectively. These constants are used to highlight variations in the weights that represent a significant impact in each individual scores, i.e. to $w_1$ is assigned the highest constant as this weight is the most representative of the three, and so on.

The compatibility values will be comprehended between 0 and 1 and bound values must be set to define each species. For example, if bound values are set to be multiples of 0.025, we can define a species $X$ that will comprehend all the individuals with compatibility values between 0.4 and 0.425, a species $Y$ with compatibility values between 0.425 and 0.45, and so on.

This bound values can determine the existence of too many species (closer bound values), leading to few individuals in each species which allows the progression of low performance individuals across generations, or very few species (further bound values), leading to the competition of individuals with genotyping differences among each other.

The speciation operator in the API executes the equation 3.2 in every new individual and assign it to its correspondent specie according to the compatibility value.

**Selection**

The number of offsprings that the system is allowed to generate must be established taking into account that a specified number of elements of each species must be selected. The computation of the amount of offsprings assigned to each species $i$ is represented in the following equations:

$$Offsprings_i = \sum_{j=0}^{N} \frac{AdjustedFitness_j}{AverageAdjustedFitness}$$  \hspace{1cm} (3.3)$$

where

$$AdjustedFitness_j = \frac{Fitness_j}{Size(Species_i)}$$  \hspace{1cm} (3.4)$$

and

$$AverageAdjustedFitness = \frac{TotalAdjustedFitness}{Size(Population)}$$  \hspace{1cm} (3.5)$$

After selecting $N$ individuals per species that will generate offsprings, a roulette procedure will be applied to select the mating combinations among the fittest $N$ individuals of that species.

As some rounding to values within these calculations are preformed, the total number of offsprings may be different than the population size. In this case a process called emigration is preformed in which the quantity of excess or missing offsprings are added or taken randomly from a specie.
Moreover, some species may not be able to produce as many offsprings as the previous equations results state that is suppose e.g. if a specie has 2 individuals and the calculations determine that this specie must generate 5 offsprings, this is impossible as each 2 parent individuals can only generate 2 child individuals, making a total of only 4 possible offsprings. For these cases, another process of emigration composed by two steps will take place: (1) a first iteration through all species will adjust their offspring amount to feasible values and collect the amount of emigrants needed to be redistributed; (2) the amount of emigrants will be appended to species that are able to generate more offsprings than the ones the equations settled them to do.

**Crossover**

The crossover process is made within each species and is divided in two steps: (1) a double point crossover in random positions of the parents grid, dividing them in 3 sets of columns and exchanging the middle set between them; (2) a single arithmetic recombination between the weights \( w_1 \), \( w_2 \), and \( w_3 \) of both parents.

The individuals involved in the crossover process as parents are chosen according to their ROI and it is implied that at least half of the offsprings have to be parents. The general rules for possible combinations between parents and children depending on the number of offsprings are the following:

1. 1 offspring - the parent with highest ROI proceed to the next generation.
2. 2 offsprings - the 2 parents with higher ROI proceed to the next generation.
3. 3 offsprings - the 2 parents with higher ROI and 1 of the children resulting from the crossover operation between them proceed to the next generation.
4. \( X \) offsprings that imply the existence of an even number of parents:
   (a) If \( X \) is odd, the \( \frac{X}{2} \) parents with higher ROI and the children resulting from the crossover operation between them, except one child, proceed to the next generation.
   (b) If \( X \) is even, the \( \frac{X}{2} \) parents with higher ROI and the children resulting from the crossover operation between them proceed to the next generation.
5. \( Y \) offsprings that imply the existence of an odd number of parents:
   (a) If \( Y \) is odd, the \( \frac{Y}{2} + 0.5 \) parents with higher ROI and the children resulting from the crossover operation between \( \frac{Y}{2} - 0.5 \) parents with higher ROI proceed to the next generation.
   (b) If \( Y \) is even, the \( \frac{Y}{2} + 0.5 \) parents with higher ROI and the children resulting from the crossover operation between \( \frac{Y}{2} - 0.5 \) parents with higher ROI, except one child, proceed to the next generation.

For a better understanding on how these rules work with real cases, table 3.1 contains 10 cases illustrating the relationship between parents and their children in terms of the crossover operation for each one, as well as indicates which of the previous rules applies in each case.
Table 3.1: Examples of crossover operation using the general rules for possible combinations.

<table>
<thead>
<tr>
<th>Offsprings</th>
<th>Parents</th>
<th>Children</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Rule 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 Rule 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 Rule 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 Rule 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 Rule 5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Mutation

The mutation operation has a chance $P$ of being triggered and will take place randomly over a number $N$ of columns of the grid. From each of the chosen columns, a number $M$ of cells will also be randomly chosen to be mutated, being their values affected according to a threshold $T$ that will define how big the change is.

These values are constrained as the four conditions for each column stated before in this report must be respected after the operation is completed. Therefore, $P$ is always set to 20%; $N$ can be set between 1 and 2; $M$ is fixed to 2 due to the condition stating that the sum of each column must be 0 (if a cell value is increased, other has to be reduced in the same column); $T$ can take the values 0.1 or 0.2.

The reason of all these constraints relies on the fact that the goal of the mutation operation is to introduce diversity in the population and not slow down the program. After choosing $N$ random columns and 2 different random cells of each, their values will be changed in $\pm T$. All these changes to the grid values are then random, so an auxiliary check function is then triggered to check whether the integrity of the grid regarding the conditions it must fulfill is kept despite the changes. If they are not, the process is repeated for same values $N$ and $T$, with another random combination of columns and cells and until the auxiliary function checks all four conditions. This is the reason why all the changes involved in the mutation operator must be narrowed into values that do not keep the program stalled for too long.

This operator will also rely on a method called Hyper-Mutation [35], which temporarily increases the mutation rate when a situation of local maximum occurs. This way, when the overall return of the GA does not improve over a predefined number of generations, the mutation rate increases inducing more genetic diversity and the possibility of finding a new better solution.

### Elitism

Some of the best fitting individuals are eligible to proceed to the next generation without passing through crossover and mutation operations in a process called elitism. The goal is to prevent changing in the performance of the best individuals of each generation, so the fittest individual or set of individuals of the current population, i.e. the ones with highest ROI, do not decrease in the following generation due
to crossover and mutation.

For that purpose one of the following conditions must be satisfied: (1) the most 2% fittest individuals of each species must proceed directly to the next generation; (2) if the value of previous condition is smaller than a predefined threshold value (e.g. one), the amount of fittest individuals to have a free pass into the next generation must be set to that value.

In order to track elitist individuals in each generation, the object specie has an attribute consisting of a list of elitists that is updated every generation before the mutation operation. Having this list aside the one containing the current individuals in the population allows to include even the fittest individuals into the process of crossover and mutation, which can open new windows of opportunity to increase the fitness of that species. If by any mean the overall ROI for that species decreases in the next generation, elitist individuals from the current will recover their positions in the current population list.

**Fitness Function**

The main objective of implementing this system is to maximize the return resulting from the investment that was made. So, in order to evaluate each individual of the population, ROI function is going to be applied to pick the best ones for reproduction over each generation of the GA.

Each time the pattern is detected by the algorithm, the program enters in market and only leaves once predefined percentages of profit or loss are achieved. The final ROI is, therefore, also a percentage calculated according to the formula:

\[
ROI = \#Wins \times \%\text{TakeProfit} - \#Losses \times \%\text{ExitWithProfit}
\]

where \#Wins and \#Losses represent the number of times the program left the market with profit or loss, respectively.

**Termination Condition**

The termination condition to be applied can affect both the quality and speed of the search, usually in an inverse proportionality relationship. Taking this into account, the program should avoid not only pointless computations, but also immature terminations of the program. Therefore, to fulfill these needs some conditions will be applied to ensure a correct termination of the algorithm:

1. If the algorithm reaches a predefined number of generations;

2. If the highest fitness score for the population does not improve for a predefined number of generations, always smaller than the ones in condition 1.

3. If the highest fitness score for a species population does not improve for a predefined number of generations, always smaller than the ones in condition 1, the fittest individual is kept but that species stops being evaluated.
Even though 1 ensures the program always ends and conditions 2 and 3 solve the problem of pointless computations, the generation numbers that will limit each condition must be carefully chosen in order to avoid the problem of immature terminations. Making sure program execution is properly bounded is key for unnecessary runtime of the system, as well as gives time for GA to improve existing solutions without over-fitting them.

3.5.4 System Optimizations

The main drawback of working with systems that rely on matrix calculations, such as a 10x10 grid of weights, in terms of computational purposes is that the whole process is slowed down by these computations.

Specifically about the FSGW being used in this project some tests were made in a ASUS laptop containing an Intel Core i5 with 2.50GHz of processor’s clock speed, where the average time for evaluating each individual in the GA was about 4 seconds. For an algorithm run with 25 generations, made over 100 different stocks and containing 100 individuals, the process would take almost 6 days of non-stop computation to finish.

With the goal of reducing this time, rather than heading for a machine with a faster CPU as the first priority, a python multiprocessing module was incorporated in the program. Multiprocessing is a package that supports spawning processes similarly to the threading module, offering local concurrency and the ability to make the most from each available core of the machine. The same test over these new conditions was almost 4 times faster.

Furthermore, the tool Anaconda Accelerate from NVIDIA for taking advantage of Graphics Processing Unit (GPU) of machines with the goal of increase computation speed came to prove unfeasible, as it is designed for common operations that repeat over lots of iterations contrarily to what happens in each iteration of the evaluation operator, which is an highly branched section of code whose behavior depends on the stock being analyzed.

3.6 Chapter Conclusion

This chapter presented an insight on the developed system’s architecture. It also contain all particular decisions regarding the API developed for the GA execution, all input parameters’ details and possible values, and the way data is represented and manipulated during the whole process, since it is fed by the user to its presentation as output data. The current solution is able to be tested and chapter 4 contains the description of all all preformed tests, their results and descriptive analysis.
Chapter 4

Evaluation

In this chapter the results of using FSGW (explained in the last chapter) applied over the stock market will be presented and analyzed. The following sections contain three test scenarios, a general setup of the system for every test, the reason why they were done and analysis of results.

4.1 Tests Scenarios and Objectives

To authenticate the effectiveness of the approach developed a strategy called Backtesting [36] was employed. Backtesting allows the simulation of a trading strategy using historical data to achieve results and evaluate risk and profitability before compromising real capital. If along with it come positive results then the strategy is solid enough and likely to yield profits when implemented. By applying this concept it is possible to test various scenarios without waiting for a long time to obtain results, detect flaws in the approach and analyze where it can be improved.

Over this idea the following three test scenarios were executed:

1. Using the FSGW to detect patterns, described in last chapter, against the TSAG, in order to prove if the new approach overcomes the previous one.

2. Using the FSGW with market exiting parameters (exit with profit and stop loss) that depend on the market volatility over the 3 months prior to the starting day of the test against the same approach using fixed exiting parameters, with the goal of understanding whether this additional measure can improve the results, by relying on it to decide if the risk of leaving a position open for more time expecting a higher return is worth it.

3. Using the FSGW with a speciated GA against the usual GA without species, in a way to verify if the introduction of genetic diversity in the population can act as a key to encounter better individuals in the training phase.

The choice of these scenarios were made taking into account they cover all main features of the system, which, all combined, make it different than existing ones, namely the combination of a GA
approach using speciation along with the dynamics involved over the market volatility, and also the FSGW approach never used before.

4.2 General Setup

Despite all test scenarios are independent of each other, some parameters and mechanisms are the same throughout the entire evaluation process. Regarding these predefined parameters, Table 4.1 displays their respective values.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Market</td>
<td>100 stocks from S&amp;P 500</td>
</tr>
<tr>
<td>Training Period</td>
<td>70%</td>
</tr>
<tr>
<td>Testing Period</td>
<td>30%</td>
</tr>
<tr>
<td>Minimum Training Time</td>
<td>1 year</td>
</tr>
<tr>
<td>Grid Offset</td>
<td>0.5</td>
</tr>
</tbody>
</table>

The portfolio chosen consist of 100 stocks from the S&P 500 index through a pre-selection process where 503 stocks were trained and the 100 most profitable were chosen as the ones to be analyzed in detail. The choice of this amount was made considering that in a real scenario the management of a bigger portfolio would be impracticable and, despite the optimizations, the algorithm still takes some time when running with a GA with lots of generations. In the choice of this portfolio none fundamental analysis was made, which, if done, could have improved even more the results.

Training and testing periods are treated as percentages because for not all S&P 500 stocks entered the index at the same time, so it is given the same amount of relative time to all stocks, having only the condition that training time percentage must be at least a whole year, otherwise the training period is not effective enough and the stock is discarded. Despite this, the majority of the stocks have 18 years of historical data available (between January 2000 and February 2018), which leads to approximately 12.5 years of training and 5.5 years of testing, being the point in between both in mid-2012.

The last parameter is related to the FSGW and decides the length of each grid level related to the stock quote. Figure 3.7 serves as an indication for the positioning of the first column of the grid, right in between the second and third cells of the grid and corresponding to the closing price. This Grid Offset indicates how much the stock quote varies between each cell of the grid, e.g., if the closing price for the first column of a specific grid is 15 and the Grid Offset value is 0.5, the price limits of the cell bellow that value are 15 and 14.5, the following one between 14.5 and 14, and so on. Figure 4.1 is an illustration of the given example.

Regarding the evaluation mechanisms to be used in every test scenario some considerations must be taken into account. Firstly, ROI will always be measured in percentages, being the program set to enter in the market when the pattern discovery condition is triggered and will only leave the market once a Take Profit or Stop Loss percentage is achieved, being the final ROI for each stock calculated accordingly to the formula 3.6.
Another aspect to take into consideration is related to the fact that for every test all parameter specifi-
cations will be the same except the specific one being addressed. For example, in the first test scenario
FSGW will be tested against TSAG that uses an adaptable grid and only the closing price for scoring
calculation, but all other parameters will be the same. One of the parameters that is always used, ex-
cept when is specifically indicated otherwise, is the choice of Take Profit and Stop Loss depending on
market volatility in the 3 months prior to the beginning of testing phase. Once these volatility values are
calculated for every stock, mean and standard deviation of them are obtained and 3 pairs of Take Profit
and Stop Loss are defined:

- Lower ones, if the volatility of a specific stock is lower than \( \text{Mean} - \text{StandardDeviation} \);
- Intermediate ones, if the volatility of a specific stock is higher than \( \text{Mean} - \text{StandardDeviation} \) but
  lower than \( \text{Mean} + \text{StandardDeviation} \);
- Higher ones, if the volatility of a specific stock is higher than \( \text{Mean} + \text{StandardDeviation} \).

Joining all previous, an anaesthetic aspect has also to be pointed for correct understanding of results
regarding price window graphs that will be presented. Each can contain three types of vertical lines
along them: a black vertical line that indicates a point of entry in the market, a green vertical line that
indicates a point of exit from the market with profit and a red vertical line that that indicates a point of
exit from the market with loss.

4.3 Test Scenario 1 - FSGW vs TSAG

The presented test scenario exhibits the results obtained when evaluating both these strategies over the
conditions of Table 4.1, except for the Grid Offset parameter in the TSAG, as the grid is adaptable.

4.3.1 Parameter Specification

The evolutionary strategy adopts the parameter specification of Table 4.2.

The three pairs in it refer to market exits with volatility influence explained in section 4.2; mutation
parameter refers to how the mutation operator works as explained in subsection 3.5.3, right in the same
place where hyper-mutation subject is addressed and it is triggered to double the mutation rate if, after
3 generations, the overall ROI of the population does not increase; species threshold refer to the size
of each species regarding their compatibility values as explained in the same subsection as before; and
finally when a grid hits the score of 7 units a market entry signal will be given.

4.3.2 Performance Measures

The following graphs show the performance of each strategy regarding the average return during testing
period, the average return for every year that the position stays opened in market during testing period
and the average return for every year of testing period, whether the position is opened or not in this last
case.

![Average Test Return](image1)

![Avg Return p/ Year w/ an opened position in Market](image2)

![Avg Return p/ Year in Test](image3)

Figure 4.2: Performance comparison graphs of Test Scenario 1.

On Table 4.3 each strategy performance is presented according to their entries and days in the
market, as well as the profitability of each position.

Table 4.3: Performance comparison metrics of Test Scenario 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>FSGW</th>
<th>Simple</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Market Entries</td>
<td>6.17</td>
<td>2.99</td>
</tr>
<tr>
<td>Wins (%)</td>
<td>48</td>
<td>50</td>
</tr>
<tr>
<td>Losses (%)</td>
<td>52</td>
<td>50</td>
</tr>
<tr>
<td>Average Days in Market</td>
<td>546.3</td>
<td>231.5</td>
</tr>
<tr>
<td>Profitable Positions</td>
<td>85</td>
<td>80</td>
</tr>
<tr>
<td>Non-Profitable Positions</td>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>Max. Profit (%)</td>
<td>120</td>
<td>75</td>
</tr>
<tr>
<td>Min. Profit (%)</td>
<td>-37.5</td>
<td>-15</td>
</tr>
<tr>
<td>Avg. Profit (%)</td>
<td>21.33</td>
<td>10.9</td>
</tr>
</tbody>
</table>
4.3.3 Discussion of results and Illustrative Examples

Testing whether the creation of FSGW to contrast with the traditional grid already in use reveals profitable is one of the main challenges of this project. From the graphs presented in Figure 4.2 we can observe that FSGW clearly surpasses TSAG in terms of average return, except for the average return per year in market, i.e. for each 365 days each position is hold in the market the average return is higher. The main reasons that contribute for this, as observable in Table 4.3, are the fact that the average number of days in market of each stock using TSAG approach is nearly 2.4 times inferior than in FSGW case, allied to the average number of market entries that are also nearly 2 times inferior and the fact that the percentage of wins is slightly higher. This, however, means that TSAG misses some of the pattern occurrences.

As an example, Figures 4.3 and 4.4, that display the candlestick representation of Analog Devices Inc. (ADI) stock between late 2015 and early 2017, with market entries and exits for FSGW and TSAG approaches respectively, show an uptrend scenario in which, using a FSGW approach, the pattern was identified three times successfully as the ROI for each entry was positive, but, on the other hand, while using the TSAG approach, none of these patterns were identified.

ADI’s stock using FSGW ended up being one of the portfolio’s best in testing phase, with a total return of 105%, while using TSAG was only the 47th best with a total return of 7.5%.

![Figure 4.3: Extract from ADI stock during testing phase of FSGW.](image)

Without taking into account any other of previous performance measures, the preceding example could suggest that the FSGW approach was lucky in identifying the pattern as the market was predominantly climbing, specially during the last part of the testing period.

To contradict this consideration, Figure 4.5, that displays the candlestick representation of Chesapeake Energy Corporation (CHK) stock between late 2014 and late 2015, with FSGW market entries and exits, show a downtrend scenario in which, using a FSGW approach, the pattern was identified three times successfully, as the ROI for each entry after the highest peak shown was positive and one time unsuccessfully as the ROI for that entry was negative.

On the other hand, Figure 4.6, that displays the candlestick representation of the same stock and
period of time, with TSAG market entries and exits, only the same unsuccessful pattern stated in the previous paragraph was identified and it remained out of the market during the rest of that period, missing at least two windows of opportunity after the late 2014 peak.

CHK’s stock using FSGW ended up being the best of 100 stocks in the testing phase, with a total return of 120%, while using TSAG ended up only as 43th best with a total return of 10%.

Both these examples along with already known performance metrics lead to the conclusion that using TSAG approach in situations of non linear uptrends and downtrends can lead to false positives or false negatives as an accurate identification cannot be achieved when it is possible to validate a pattern with a grid that adapts its size to the price window.

The fact the majority of successful entries in the market occur when the scale of the grid in a validation situation is similar to the following period in the price window not only explains why this approach stays much less days in the market, enters the market much less times, and is less profitable in the testing period compared to FSGW, but also why the yearly return, referred above in this subsection, is one of
the few metrics that improves with this approach.

Last paragraph observations ended up being some of the main reasons to create FSGW, as, if we cannot force every price window in the whole market to behave similarly in terms of scale like a specific price window, then let's fix a grid within predefined limits in that scale and find the best configuration of scores and weights for a specific market, so all identified patterns are similar within the scale of the grid and not the scale of the market.

Figure 4.7 displays the candlestick representation of First Energy Corp. (FE) stock during the whole testing period, between mid 2012 and early 2018, with market entries and exits for FSGW approach, and serves to illustrate why it was the least profitable with a return of -37.5%. As it is observable, the market is extremely volatile during this period and hence all entries are triggered by a pattern match technique, there are no other factors being taken into account. This is one situation where possibly using fundamental indicators and knowledge/experience of the market would help to blacklist this specific stock from the portfolio avoiding a negative ROI value, but this is beyond the purpose of this work and was not taken into consideration.

Figure 4.7: FE stock during testing phase of FSGW.
The last example of Test Scenario 1 stands as the perfect one to illustrate the main mechanisms of this novel approach in a real case, turning a negative ROI position of the portfolio into a positive one. Figures 4.8 and 4.9, that display the candlestick representation of Pentair PLC (PNR) during the whole testing period, between mid 2012 and early 2018, with market entries and exits for FSGW and TSAG approaches respectively, refer to one of the least profitable of TSAG’s portfolio, with a total return of -15%, while for the FSGW approach the total return is 7.5%.

As it is observable the market follows an uptrend in the beginning that is availed twice by the FSGW but never by TSAG, however it becomes extremely volatile at a certain point and TSAG only entered in action here with bad results. All these decisions are explained by the two main aspects that were pointed out to be improved using the FSGW, i.e. the adaptability of the grid and the fact that it only takes into account closing price of a stock, leaving some important information given by the candlestick representation regarding the volatility of the price window behind.

Figure 4.8: Extract from PNR stock during testing phase of FSGW.

Figure 4.9: Extract from PNR stock during testing phase of TSAG.
In sum, it is safe to say we are debating over two strategies that, for the given portfolio, present positive overall ROIs, but FSGW ended up improving them.

4.4 Test Scenario 2 - FSGW with market exiting parameters depending on volatility vs FSGW with fixed market exiting parameters

This test scenario exhibits results obtained when evaluating FSGW in two different modes: with market exiting parameters (exit with profit and stop loss) that depend on market volatility over the 3 months prior to the starting day of the testing and the same approach using fixed exiting parameters, always over the conditions of Table 4.1. As stated in subsection 4.2, 3 pairs of take profit and stop loss percentages are used while running FSGW normally, and in this test this approach will be compared with other 3 approaches with the following nomination:

- FSGW 10-5, where all stocks exit the market when having a take profit value of 10% and stop loss of 5%;
- FSGW 15-7.5, where all stocks exit the market when having a take profit value of 15% and stop loss of 7.5%;
- FSGW 20-10, where all stocks exit the market when having a take profit value of 20% and stop loss of 10%;

4.4.1 Parameter Specification

The evolutionary strategy adopts the following parameter specification. Each parameter serves the same purpose as stated in Test Scenario 1.

Table 4.4: Evolutionary parameters of Test Scenario 2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population</td>
<td>100</td>
</tr>
<tr>
<td>Number of Generations</td>
<td>25</td>
</tr>
<tr>
<td>Generation Limit for Improv</td>
<td>12</td>
</tr>
<tr>
<td>Mutation (% Chance / #Columns / Threshold)</td>
<td>10 / 1 / 0.1</td>
</tr>
<tr>
<td>Hipermutation (Gens. Without Improv / Increase Rate)</td>
<td>3 / x2</td>
</tr>
<tr>
<td>Species Threshold</td>
<td>0.02</td>
</tr>
<tr>
<td>Pattern Confirmation Score Threshold</td>
<td>7</td>
</tr>
</tbody>
</table>

4.4.2 Performance Measures

The following graphs show the performance of each strategy regarding the average return during testing period, the average return for every year that the position stays opened in market during testing period
and the average return for every year of testing period, whether the position is opened or not in this last case.

![Figure 4.10: Performance comparison graphs of Test Scenario 2.](image)

On Table 4.5 each strategy performance is presented according to their entries and days in the market, as well as the profitability of each position.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>FSGW</th>
<th>10% - 5%</th>
<th>15% - 7.5%</th>
<th>20% - 10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Market Entries</td>
<td>5,17</td>
<td>7,32</td>
<td>5,91</td>
<td>6,66</td>
</tr>
<tr>
<td>Wins [%]</td>
<td>48</td>
<td>42</td>
<td>49</td>
<td>53</td>
</tr>
<tr>
<td>Losses [%]</td>
<td>52</td>
<td>58</td>
<td>51</td>
<td>47</td>
</tr>
<tr>
<td>Average Days in Market</td>
<td>540,3</td>
<td>334,9</td>
<td>512,6</td>
<td>681,2</td>
</tr>
<tr>
<td>Profitable Positions</td>
<td>85</td>
<td>78</td>
<td>83</td>
<td>90</td>
</tr>
<tr>
<td>Non-Profitable Positions</td>
<td>15</td>
<td>22</td>
<td>17</td>
<td>10</td>
</tr>
<tr>
<td>Max. Profit [%]</td>
<td>120</td>
<td>90</td>
<td>112,5</td>
<td>120</td>
</tr>
<tr>
<td>Min. Profit [%]</td>
<td>-37,5</td>
<td>-35</td>
<td>-45</td>
<td>-40</td>
</tr>
<tr>
<td>Avg. Profit [%]</td>
<td>21,3</td>
<td>5,5</td>
<td>20,5</td>
<td>26,3</td>
</tr>
</tbody>
</table>

### 4.4.3 Discussion of results and Illustrative Examples

From the graphs presented in Figure 4.10 we can observe that, in what concerns the market we are evaluating FSGW, increasing percentages of take profit and stop loss is indeed more profitable. Table 4.5 indicates, as expected, that the higher exiting percentages are, more time opened positions stay in the market, and consequently they enter less times in the market.

The fallout of all this in a market that behaves mostly in an uptrend scenario is that low exiting percentages (e.g. FSGW 10-5) will have lower maximum/average profits and lower losses in terms of its portfolio’s worst positions, and on the other hand, high exiting percentages (e.g. FSGW 20-10) will have higher maximum/average profits and higher losses in terms of its portfolio’s worst positions, but as the percentage ratio between wins and losses is 2 for 1 respectively, for this type of market the losses being higher does not affect overall significantly.

As it is also observable, the hybrid FSGW that uses all 3 pairs of exiting depending on volatility does not overcome FSGW 20-10, but once again the market in which the scenario is being tested is what allows it to happen, as in a more sideways or sinusoidal market the quantity and severity of non-profitable positions of a FSGW 20-10 approach would reduce its effectiveness. However, even with this test conditions it is possible to prove this point. Table 4.6 shows all 10 non-profitable positions of FSGW 20-10 compared to the same positions in a portfolio using a normal FSGW approach.
Table 4.6: Comparison between FSGW 20-10’s worst positions against normal FSGW.

<table>
<thead>
<tr>
<th>Stock</th>
<th>20_10</th>
<th>Normal</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>-10</td>
<td>0</td>
</tr>
<tr>
<td>ALB</td>
<td>-10</td>
<td>7.5</td>
</tr>
<tr>
<td>FAST</td>
<td>-10</td>
<td>15</td>
</tr>
<tr>
<td>FTI</td>
<td>-10</td>
<td>-22.5</td>
</tr>
<tr>
<td>TGT</td>
<td>-10</td>
<td>7.5</td>
</tr>
<tr>
<td>UAA</td>
<td>-10</td>
<td>15</td>
</tr>
<tr>
<td>FE</td>
<td>-20</td>
<td>-37.5</td>
</tr>
<tr>
<td>MET</td>
<td>-30</td>
<td>-30</td>
</tr>
<tr>
<td>HBI</td>
<td>-40</td>
<td>22.5</td>
</tr>
<tr>
<td>SLG</td>
<td>-40</td>
<td>0</td>
</tr>
<tr>
<td>Avg. Return</td>
<td>-19</td>
<td>-6.75</td>
</tr>
</tbody>
</table>

For these positions of FSGW 20-10 to be negative means these stocks’ behaviour meet the conditions that reduce the effectiveness of using higher exiting percentages and, as it is observable, using a hybrid approach that reduces exiting percentages revealed a better option in 80% of the cases.

In a more concrete example, Figures 4.11 and 4.12 that display the candlestick representation of Under Armor Inc. Class A (UAA) along the year of 2016, with market entries and exits for FSGW and FSGW 20-10 approaches respectively, refer to the stock that benefited the most from using a FSGW hybrid approach, instead of FSGW 20-10, with an increase of 25% in total return.

Through observation of the graph itself, it is possible to understand that the stock price is not varying that much over time, and for each approach a signal for opening a position was given almost at the same time. However, as in FSGW the exiting percentages decided by the volatility of the graph are lower, the approach is capable of taking advantage of slighter uptrends and the position closes with profit right before the end of July in a small peak. On contrary, FSGW 20-10 approach is not capable of this and if the price does not climb that much once the pattern is discovered, the position will not be closed and takes the risk of ending up in a situation of stop loss. Only this specific case in one of the five years of testing period of UAA ended up being enough for turning a non-profitable stock into a profitable
one using FSGW’s hybrid approach.

In sum, what can be drawn from this test scenario is that one must know the market in which is putting his money to decide the better exiting configuration. If the market is climbing in overall it was proven that higher exiting percentages bring better results. Once again as all of the scenarios present positive overall ROIs, which enhances even more FSGW approach, but, as it is known, nothing can be taken from granted in what concerns the stock market, so the approach that benefits the risk/profit relationship and brings along more insurance is the hybrid FSGW with 3 pairs of exiting percentages.

### 4.5 Test Scenario 3 - FSGW using speciation in GA vs FSGW not using speciation in GA

The presented test scenario exhibits the results obtained when evaluating both these strategies over the conditions of Table 4.1. As the GA execution occurs on the training phase of the algorithm, beyond the usual testing phase data that is also presented in previous test scenarios it is also being taken into account data from training phase for this scenario.

#### 4.5.1 Parameter Specification

The evolutionary strategy adopts the parameter specification of Table 4.7. Each parameter serves the same purpose as stated in Test Scenario 1.

#### 4.5.2 Performance Measures

The following graphs show the performance of each strategy regarding the average return during testing period, the average return for every year that a position stays opened in market and the average return for every year (whether the position is opened or not in this last case) in both periods.
On Tables 4.8 and 4.9 each strategy performance is presented regarding their training and testing periods respectively, according to their entries and days in the market and also the profitability of each position.

Table 4.7: Evolutionary parameters of Test Scenario 3.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population</td>
<td>100</td>
</tr>
<tr>
<td>Number of Generations</td>
<td>25</td>
</tr>
<tr>
<td>Generation Limit for Improv</td>
<td>12</td>
</tr>
<tr>
<td>Lower Pair (Take Profit % - Stop Loss %)</td>
<td>15 - 7,5</td>
</tr>
<tr>
<td>Intermediate Pair (Take Profit % - Stop Loss %)</td>
<td>20 - 10</td>
</tr>
<tr>
<td>Higher Pair (Take Profit % - Stop Loss %)</td>
<td>25 - 12,5</td>
</tr>
<tr>
<td>Mutation (% Chance / #Columns / Threshold)</td>
<td>10 / 1 / 0,1</td>
</tr>
<tr>
<td>Hipermutation (Gens. Without Improv / Increase Rate)</td>
<td>3 / x2</td>
</tr>
<tr>
<td>Species Threshold</td>
<td>0,02</td>
</tr>
<tr>
<td>Pattern Confirmation Score Threshold</td>
<td>7</td>
</tr>
</tbody>
</table>

Figure 4.13: Performance comparison graphs of Test Scenario 3.

Table 4.8: Performance comparison metrics of training set regarding Test Scenario 3.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>FSGW</th>
<th>Single Specie</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Market Entries</td>
<td>18,72</td>
<td>18,65</td>
</tr>
<tr>
<td>Wins (%)</td>
<td>65</td>
<td>68</td>
</tr>
<tr>
<td>Losses (%)</td>
<td>35</td>
<td>32</td>
</tr>
<tr>
<td>Average Days in Market</td>
<td>1577,6</td>
<td>1586,9</td>
</tr>
<tr>
<td>Profitable Positions</td>
<td>89</td>
<td>89</td>
</tr>
<tr>
<td>Non-Profitable Positions</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>Max. Profit (%)</td>
<td>375</td>
<td>387,5</td>
</tr>
<tr>
<td>Min. Profit (%)</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>Avg. Profit (%)</td>
<td>193,6</td>
<td>200,9</td>
</tr>
</tbody>
</table>

Table 4.9: Performance comparison metrics of testing set regarding Test Scenario 3.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>FSGW</th>
<th>Single Specie</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Market Entries</td>
<td>4,65</td>
<td>4,88</td>
</tr>
<tr>
<td>Wins (%)</td>
<td>54</td>
<td>51</td>
</tr>
<tr>
<td>Losses (%)</td>
<td>46</td>
<td>49</td>
</tr>
<tr>
<td>Average Days in Market</td>
<td>673,3</td>
<td>681,9</td>
</tr>
<tr>
<td>Profitable Positions</td>
<td>89</td>
<td>89</td>
</tr>
<tr>
<td>Non-Profitable Positions</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>Max. Profit (%)</td>
<td>120</td>
<td>100</td>
</tr>
<tr>
<td>Min. Profit (%)</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>Avg. Profit (%)</td>
<td>25</td>
<td>26,5</td>
</tr>
</tbody>
</table>
4.5.3 Discussion of results and Illustrative Examples

The use of a speciated approach in this work had the goal of trying to improve the performance of the GA in what concerns the search for a most profitable grid. Observing the graphs of Figure 4.13 it is possible to realize that overall returns are similar to the point that average return per year in both training and testing phases are separated from each other by less than 1%.

The analysis of the performance metrics presented in Tables 4.8 and 4.9 lead us to a similar conclusion, as the most significant differences occur in maximum profit and percentage of market exits with profit metrics of each phases, which are values that, by themselves, do not have enough significance to conclude which approach is the best, as the maximum profit represents just the return of one stock of an entire portfolio and the percentage of wins does not indicate the magnitude of them. It is also visible that the training metric equivalence between both approaches is transposed to the testing scenario.

However, these similarities do not mean that one should discard the speciation approach for three main reasons: (1) the genetic encoding is composed of 57 grid positions and 3 weights whose values can vary, so the number of possible combinations is estimated to be in the order of trillions, which means that even a configuration between individuals and number of generations 1000 times higher than the “100 individuals and maximum of 25 generations” being used would cover less than 0.001% of possible configurations, so indeed the absolute best of these combinations may not be even close of being generated by the algorithm or achieved by genetic operator procedures even using speciation; (2) the weighted approach, the symmetry imposed to the grid that makes mandatory every change to be accompanied by another on the other side of the axis and the range of possible combinations can, all together, lead to a situation where even very different configurations may end up with same results in terms of final return; (3) the cost of using a speciated approach, in terms of time consumption, is roughly similar to not using one, so even if a speciated approach does not ensure that a bigger set of possible combinations is covered, it does cover a bigger area in the map of the possible sets, which is always a more optimistic scenario, even if it does not give any guarantees of a better return due to the reasons presented in the two previous points.

The following three figures address examples on how the use of a speciated strategy has improved the overall ROI of the GA in the training phase. As a recall from what was explained in subsection 3.5.3, the grid weight configuration is what determines in which species each individual ends up.

Figure 4.14 shows the evolution of each species, regarding the profitability of its best individual, during the GA execution of SL Green Realty Corp. (SLG) stock over 25 generations. In this figure are also presented relative percentages related to the ROI of each species in first and last generations for an easier visualization on how each species has evolved.

For SLG’s particular case, if this population was not divided into species the best individuals in the first generation were from the “zone” of individuals identified with the colors purple and orange, both having a return of 150% and representing 11% of the best individuals’ total returns, meaning that they would likely be dominant over less profitable individuals, such as the ones from the two green species located at each tip of the graph, as well as the red one. However, by letting each species improve their individuals constrained to its scope, the red one ended up being the most profitable with a total return of
190%, and the ones representing 11% of the best individuals’ total returns now only account for 8% of it.

Figure 4.14: Species profit evolution of SLG stock during testing phase of FSGW.

Figure 4.15 presents a similar case regarding the GA execution of Republic Services Inc. (RSG) stock over 20 generations, in which species marked in red and orange were two of the best in the first generation, representing 12% and 11%, respectively, of the best individuals’ total returns, but ended up decreasing 3% in the final generation. Contrarily, purple and pink species started with a mere quantity of 7% of the total scope and ended up as two of the most profitable with a growth of 4% in their relative profitability over the best individuals of each species.

Figure 4.15: Extract from RSG stock during testing phase of FSGW.
Finally, Figure 4.16 regarding the GA execution of Cabot Oil and Gas Corp. (COG) stock over 25 generations is one of the most extreme scenarios where it is visible that a species which was only considered the 6th best in first generation ended up having a profitability that represent 17% of the best individuals’ total returns. If the GA was not using speciation, this individual would eventually be crossed with a random one in the population instead of only ones with the same genetic compatibility, condition that was crucial to the creation of an extreme profitable individual.

In sum, it is proven that for training a FSGW using a GA speciated approach is possible to achieve at least as good results as a non-speciated approach, however there were presented cases where the use of species allowed a turnover in the profitability of individuals that were weaker in early generations, and as this approach is not more time consuming than using the regular GA, it is definitely to be considered. Also due to the huge quantity of different evolutionary parameter configuration more ways of trying to improve the speciated could have been addressed, but once again the main purpose of this system was not that.

4.6 Overall Analysis

This section serves as a summary of the results presented in previous test scenarios regarding all different approaches presented, where some overall comparisons and observations will be made. Recalling all different approaches being tested we have the main FSGW and all its variants (exit pairs 10%-5%, 15%-7.5% and 20%-10%, as well as the approach with a single species) and also TSAG.

Before heading to the main subject of this sub-section, it is worth to mention that while testing each of previous approaches it took approximately 1 day and 2 hours of non-stopping computation per execution. For that reason there was not enough time to execute sufficient tests so that an average run of each
approach could be presented. Nevertheless, as FSGW is the main system being tested, a series of 10 runs for that approach were made and Table 4.10 show the comparison of the run chosen for the previous test scenarios with the average results of the 10 runs, also including the highest and lowest result for each parameter.

Table 4.10: Comparison between FSGW run chosen for test scenarios against the average of 10 executed runs.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>FSGW</th>
<th>FSGW Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Market Entries</td>
<td>6,17</td>
<td>6.11 [5.29; 6.29]</td>
</tr>
<tr>
<td>Wins (%)</td>
<td>48</td>
<td>48 [47; 49]</td>
</tr>
<tr>
<td>Losses (%)</td>
<td>32</td>
<td>52 [51; 53]</td>
</tr>
<tr>
<td>Average Days in Market</td>
<td>546,3</td>
<td>558 [533.7; 572.1]</td>
</tr>
<tr>
<td>Profitable Positions</td>
<td>85</td>
<td>85 [82; 88]</td>
</tr>
<tr>
<td>Non-Profitable Positions</td>
<td>15</td>
<td>15 [12; 18]</td>
</tr>
<tr>
<td>Max. Profit (%)</td>
<td>120</td>
<td>115 [127.5; 105]</td>
</tr>
<tr>
<td>Min. Profit (%)</td>
<td>-37.5</td>
<td>-42.8 [-37.5; -32.5]</td>
</tr>
<tr>
<td>Avg. Profit (%)</td>
<td>21.3</td>
<td>20.8 [19.5; 22.4]</td>
</tr>
</tbody>
</table>

It is observable that the run chosen for the testing scenarios stays close to average values and even the disparity between highest and lowest result for each parameter is narrow, which leads to the conclusion that the chosen run of the algorithm can be accounted as well representative of what FSGW is able to achieve, as well as the approach is reliable since it always leads to similar results.

Heading to the overall analysis, Figure 4.17 joins all the different tested approaches together in a graph that shows the evolution of their average ROIs over the entire testing period. The lines highlighted with red and green glows are FSGW and TSAG approaches, respectively, and this serves not only to show the magnitude of the improvement that using a weighted grid with a fixed size approach over the traditional existing approach has, but also to show that from 5 FSGW approaches the only that does not overcome TSAG is the one in which the market exiting strategy is the most conservative one.

Figure 4.17: Average ROI evolution over the entire testing period.
Pursuing now to an yearly analysis of ROI’s evolution within the studied approaches, Figure 4.18 presents another clear proof of the FSGW approach dominance over TSAG in terms of average ROI. Excluding 2012 and 2018, which only account for roughly 2 month in the testing period each, only 2015’s average ROIs for FSGW’s approaches bumped more than TSAG. Yet, the overall yearly returns never drop more than -2.5% while in the most positive scenarios the traditional approach is always at least 2% less profitable than the main FSGW, and in general less profitable than at least 4 out of 5 FSGW variants.

![Yearly Evolution of ROI](image)

**Figure 4.18:** Average yearly ROI evolution over the entire testing period.

Another important detail important to point out is related to the duality between the percentage of exiting positions that won and lost. Watching out on the graphs presented in Figure 4.19, at first sight it might appear that FSGW strategy and all its variants are not that good, as the duality referred above tends more to higher cases with losing positions than winning ones.

![Percentage of Win/Loss](image)

**Figure 4.19:** Percentage of win/loss results from FSGW approach and its variants.
However, and as it has been said before in this report, the relationship between a win and a loss is always from 2 to 1, i.e., once a market exit strategy is defined it is always assured that, for every win, it is possible to have two losses and yet not being in a negative balance. Addressing the worst scenario in the following graphs, even with only 42% of wins against 58% of losses, if the total number of market exits totals 100 will end up with 84% of profit against 58% of loss. With this win-loss relationship settled, only in a scenario of less than 33% of wins the balance would start to become negative.

One last remark must be done regarding the overall results of these test scenarios and is related to the grids chosen by the GA to proceed from trading to testing phase of the algorithm. All Bull Flag grids have 2 main visible characteristics: the downtrend and consolidation zones of a perfect pattern market with the highest scores and the 3 most penalized zones located in both bottom corners and on top of the 5th, 6th, 7th an 8th columns where a high presence of the market signal is a huge indicator of a failed identification.

One particular detail stands out in the majority of the resulting grids of FSGW’s GA procedure as shown in the examples of Figure 4.20, in which, from the 3 penalization zones, the bottom right corner, right bellow the highest scoring positions of the consolidation zone of the pattern, is not that much punished with bad scores as the others. In fact if we treat the grid as a two dimensional referential centered in the middle of it, the whole 4th quadrant is the less penalized area of the entire grid.

Going even further on analyzing the way this penalization zones interfere within the market signal, if a “heat map” of the common negative cells of the grid was designed, it would look something like Figure 4.21.

The explanation stands as a relationship between the location of the penalization zone with the way the signal is supposed to behave in a Bull Flag pattern. A market signal that enters the bottom left penalization zone is likely to proceed into a steep downtrend and even if it recovers into what remains of the desired pattern, it won’t look like the template. Following the same line of thought, a market
signal that enters the top penalization zone is more an indicator of a sideways scenario than a Bull Flag, contradicting also the shape of the correct template. On the other hand, if a signal reaches the bottom right penalization zone with a good total score already from the previous positions, this means that it has behaved correctly more than half of the grid. Adding to this, two more reasons arise to allow the signal to be in that position without penalizing it too much: the consolidation phase may be a late one, or, even if it is not, confirming a pattern that ends in the lower zones of the grid is always safer than ending in the highest, as it has more room to climb if it indeed starts climbing as expected.

### 4.7 Chapter Conclusion

Within this chapter were presented all test scenarios that were made using a portfolio of 100 stocks from S&P 500, over training and testing periods representing 70% and 30% of each stock’s available data, which, in the majority of cases, consists on 12.5 years of training (between early 2000 and mid 2012) and 5.5 years of testing (between mid 2012 and beginning of 2018). The designed algorithm, using a fixed size grid and a new way of attributing scores, beat the existing adaptable and simple grid, having twice the average return on investment over the testing period. It was also proved that using exiting percentages based on market volatility avoids big losses in low volatile/sideways markets, as well as using speciation guarantees that the range of possible individuals to be trained by the GA is wide, even though that optimization did not improved significantly the overall return on investment.
Chapter 5

Conclusions and Future Work

5.1 Summary and Achievements

The presented work proposes a portfolio management system using a GA with speciation along with a new approach to pattern identification of the template based approach using a grid of weights.

To validate the implemented strategy it was compared against the already existing that using a grid of weights; against itself using different input parameters regarding the way market volatility affect the program; and again against itself using the same approach but without species. The period of training and test were 70% and 30% respectively, applied over the available data of a portfolio containing 100 stocks from S&P 500, which, in the majority of the cases stand for 18 years of data between January 2000 to the first semester of 2018, that can be divided into, approximately, 12.5 years of training and 5.5 years of testing, being the point in between both in mid-2012.

The developed algorithm is able to simulate all test scenarios and the quantity of parameters it gives for user to choose, specially the ones referring to the evolutionary algorithm, opens up the possibility of many to be tested depending on the needs of the user.

Achieved results of the proposed strategy compared with the one already existing under the same parameter specification were very promising, specially regarding the overall returns over the testing period. Various variants of the proposed strategy were also tested in order to draw some conclusions about how results react to changes in some parameters, yet the majority of them still beat the traditional approach.

5.2 Future Work

In this sub-section, several limitations of the algorithm are addressed, along with some improvements that could repair those limitations as well as the overall functioning of the system.

- The algorithm is not yet prepared to run in real time and be constantly updating information, analyzing it and notifying the user of what to do next, which would clearly be the next step as the validity of the system is completed;
• Extend the testing scenario from S&P 500 to other markets and understand how the developed approach behaves within new scenarios;

• Increase new patterns to the scope of FSGW analysis and allow the short selling mechanism to increase the time in the market and possible winnings;

• Allow a multi-pattern detection approach, in which various patterns would be analyzed at the same time and decide what to do depending on relative scores of each one;

• No kind of fundamental analysis or risk measurement is applied in order to decide if at some point the user wants a position closed because the stock is considered too risky, even if the algorithm says otherwise. For that reason we should enable the possibility for it to be removed from the portfolio, even if technical indicators are good;

• During testing phase (or in case of it being used in real time) there are no automatic measurements being done simultaneously, e.g. tracking volatility and moving averages, so that some parameters could be adjusted in real time depending on this analysis with the goal of reducing risk and increasing the profit;

• The amount of input parameters are so wide that some kind of information regarding the current configuration could be given, like creating a cache of already tested scenarios and use it to inform the user in future tests what happened when some subsets of parameters were configured in the exact same way in the past;
Bibliography


58


Appendix A

Installation Manual

The application code can be found within the Algorithm.zip file. The system can run in both UNIX base systems or Windows Systems. For each systems the following requirements are necessary:

**Programs**
- Python 3.6.4
- pip 9.0.1 (for Python 3.6)

**Packages**
Using the command "pip3 install <name of package>”, the following packages must be installed:
- alphavantage
- matplotlib
- mpl_finance
- numpy
- pandas
- shelve
- xlsxwriter

Once Algorithm.zip is extracted the application file tree will appear as in Figure A.1 and, if the previous requirements are fulfilled, the program will be ready to be executed. Each presented file and folder stand for the following:

- **outputs/** - predefined folder to where all outputs will be redirected by the program;
- **project/** - contains all program’s source code organized in packages/modules and separated between them as the system architecture indicates;
- **configfile.ini** - contains inputs to be "fed" into the program upon its start and must be set before running the program;
• **setup.py** - runs the program and contains the logic to communicate with the existing modules;

• **project/FinancialData/** - contains all financial data and logic to download/update it;

• **project/logging.py** - contains all the logic to generate output data and present messages on prompt;

• **project/main.py** - despite being part of the application layer, acts as a middleware between setup.py (the presentation layer) and the remaining application layer modules, deciding which ones to call depending on which commands receives from the presentation layer;

• **project/FinancialData/RetrievedData/** - stores all the financial data in three separated folders, depending on whether it is daily, weekly or monthly data;

• **project/FinancialData/retrieve_data.py** - contains the logic to download and store financial data;

• **project/FinancialData/api_key.txt** - contains the alpha_vantage api key associated to the user’s account needed to download financial data;

• **project/GeneticAlgorithm/ga.py** - contains all GA related logic;

• **project/GridOfWeights/grid.py** - contains all FSGW related logic;
Appendix B

Execution Manual

After the installation procedure, the user can start executing the program. The system execution can be performed by running the command "python3 setup.py <args>" if the current directory is the one where the installation files were set.

The field "args" can take one of six possible values depending on the operation the user wants to preform:

1. **python3 setup.py auto** - if the user wants to automatically update all financial data to the most current values;
2. **python3 setup.py <type> <stock>** - if the user wants to update the financial data from one specific stock and type (daily, weekly, monthly) of data;
3. **python3 setup.py plot** - if the user wants to consult the graphs (candlestick representation) of any particular stock, with entering and closing positions signalization;
4. **python3 setup.py gens** - if the user wants to consult a graph with the evolution of the species over the generations of the GA (only for speciated approaches);
5. **python3 setup.py excel** - if the user wants to generate an excel with detailed information regarding the results of an execution.
6. **python3 setup.py** (no args) - if the user wants to run the algorithm.

Regarding the plotting of graphs, once this option is selected a list of available stocks to be plotted is presented and the user must manually insert the stock he wants to observe. After this, he also has to choose the execution to be plotted (more than one can be stored in the same stock), and finally between one of the three types of data (daily, weekly and monthly). In result to this, two graphs will be shown corresponding to the training and testing data, respectively. Figure B.1 shows an example of instructions and Figure B.2 the outcome of its execution.

In respect to the exhibition of the evolution of the species over the generations the procedure is exactly the same as the previous: a list of available stocks will be presented and chosen, as well as the execution from the existing ones. Figure B.3 shows the outcome of the execution.
Figure B.1: Command prompt menu for plotting data.

Figure B.2: Results from the execution of plotting data functionality.

Figure B.3: Execution of the instruction to show evolution of species over the generations functionality.
Heading for the excel option, once the command is run there is no need of more user interaction with the program. It will by itself go through all available data of a run searching for the information it needs and create a .xlsx file as a result, like the one displayed in Figure B.4.

Finally, regarding the normal execution of the program, all the input parameters must be set in the configuration file configfile.ini that is presented in the main folder of the program. Once the user issues the command to execute the algorithm he will be advised to revise the parameters before proceeding or, alternatively, to consult a manual on prompt that explains in detail what every parameter in the configuration file stands for. Figures B.5 and B.6 show an example of the configuration file and user manual, respectively.

Figure B.4: Execution of the instruction to show an excel with data from a run of the algorithm.

Finally, regarding the normal execution of the program, all the input parameters must be set in the configuration file configfile.ini that is presented in the main folder of the program. Once the user issues the command to execute the algorithm he will be advised to revise the parameters before proceeding or, alternatively, to consult a manual on prompt that explains in detail what every parameter in the configuration file stands for. Figures B.5 and B.6 show an example of the configuration file and user manual, respectively.
Figure B.5: Example of configuration file.

```
[GA_main]
type: daily
training: 0.7
grid_offset: 0.5

exit_w_profit_margin_1: 0.10
stop_loss_margin_1: 0.05
exit_w_profit_margin_2: 0.15
stop_loss_margin_2: 0.075
exit_w_profit_margin_3: 0.20
stop_loss_margin_3: 0.10

generation_limit: 25
max_gen_for_improvement: 12
use_normal_species: 1
use_single_species: 0
use_simple_eval_species: 0
population_size: 100
threshold: 0.02
factor: 0.1
minimum: 1
chance: 0.2
n_columns: 1
t_threshold: 0.1

[Grid]

adaptable_grid: 0
```

Figure B.6: Configuration file manual in prompt.