AUTOMATED STOCK TRADING: A MULTI-AGENT, EVOLUTIONARY APPROACH

by

KURT KRUGER

submitted in fulfilment of the requirements for the degree of

MASTER OF SCIENCE

in the subject

COMPUTER SCIENCE

at the

UNIVERSITY OF CAPE TOWN

Supervisor: **Dr Anet Potgieter**

August 2008

ABSTRACT

Stock market trading has garnered much interest over the past few decades as it has been made easier for the general public to trade. It is certainly an avenue for wealth growth, but like all risky undertakings, it must be understood for one to be consistently successful. There are, however, too many factors that influence it for one to make completely confident predictions. Automated computer trading has therefore been championed as a potential solution to this problem and is used in major brokerage houses world-wide. In fact, a third of all EU and US stock trades in 2006 were driven by computer algorithms.

In this thesis we look at the challenges posed by the automatic generation of stock trading rules and portfolio management. We explore the viability of evolutionary algorithms, including genetic algorithms and genetic programming, for this problem and introduce an agent-based learning framework for individual and social intelligence that is applicable to general stock markets.

Statistical tests were applied to determine whether or not there was a significant difference between the evolutionary trading approach and an accepted benchmark. It was found that while the evolutionary trading agents comfortably realised higher portfolio values than the ALSI, there was insufficient evidence to suggest that the agents outperformed the ALSI in terms of portfolio performance. Additionally, it was observed that while the traders combined knowledge from the expert traders to form complex trading models, these models did not result in any statistically significant positive returns. It must be said, however, that there was overwhelming evidence to suggest that the traders learned rules that were highly successful in predicting stock movement.

ACKNOWLEDGEMENTS

I would like to express my eternal gratitude to following people without whose influence this thesis would never have seen the light of day:

- Dr Anet Potgieter for her guidance and constant motivation;
- My girlfriend, Kristine, for her love and for putting up with my moods during stressful times;
- My parents, Gavin and Lesley, for their support and love;
- Ryan, my brother, for all his help with my research and access to all his finance resources.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

"...We simply attempt to be fearful when others are greedy and to be greedy only when others are fearful..." – **Warren Buffet**, American Billionaire

The layman's interest in the stock market has grown substantially over the last few decades due to the potential earning power of stock trading. Whereas it was once only really accessible to the rich, it has now turned into the investment opportunity of choice for growing wealth. In addition, the advances in trading technologies have opened up the market to nearly everyone who wishes to chance their arm at playing the stock market.

Despite the popularity of stock trading, most people do not fully understand stocks and the stock market. Investment in stocks can generate great wealth, but as with all inexact sciences, there is also great risk attached to this commitment. The key to turning a profit on the market is to understand where to invest money.

Alternatively, the hassle of studying historical data can be avoided by employing computer systems to automate stock trading [1]. The disadvantage of using many of these systems, however, is their inherent black-box-like characteristics [2]. Traders generally have an intuitive understanding of and feel for the stock market. These automated systems simply accept masses of data and output a result without any clear indication of the internal processes that produced it.

According to Andrew Lo [2], the director of Massachusetts Institute of Technology's Laboratory for Financial Engineering, "Now it's an arms race. Everyone is building more sophisticated algorithms, and the more competition exists, the smaller the profits."

1.1 MOTIVATION

The primary objective for this dissertation is to develop a dynamic system that models trader behaviour on the Johannesburg Stock Exchange and enables the model to learn without human intervention by using evolutionary techniques.

The motivation for this work is to investigate the feasibility of using evolutionary and technical trading models to deal with dynamic data and fluctuations in the stock market. It is hoped that the experimental results will increase our knowledge of how and why technical trading works and present effective means of maximising portfolio returns.

1.2 OBJECTIVE AND REQUIREMENTS

The aim of this work is to investigate the performance and adaptability of an evolutionary model compared to that of a baseline performance model. Rules were evolved using genetic programming where the tree structures are composed of technical trading indicators. Evolutionary agents were also employed to simulate stock trading on historical JSE data as it would occur in the real world. In addition, these agents shared a collective memory whereby they could share their trading strategies. To achieve this objective the following research questions were posed at the inception of the project:

- Can rules be evolved to adapt to the market conditions?
- Do the evolutionary trading agents realise higher portfolio values than that realised by the JSE ALSI benchmark?
- Do the evolutionary trading agents outperform the JSE ALSI benchmark?
- Do trading personalities exchange ideas and combine their trading strategies successfully?

1.3 EVALUATION CRITERIA

To evaluate our approach we need to determine how well each of the research questions was answered. Each question can be answered directly and with a high degree of confidence by applying the appropriate statistical methods. These statistical tests are introduced and discussed in more detail in Chapter 6.

1.4 DISSERTATION OUTLINE

Chapter 2 – Portfolio Theory and Technical Analysis: In this Chapter, basic portfolio theory is introduced and technical analysis and the related technical indicators are described and analysed.

Chapter 3 – Evolutionary Algorithms: In Chapter 3 the concepts of evolutionary algorithms are discussed with specific reference to genetic algorithms and genetic programming. The underlying theories are discussed and important stumbling blocks are analysed. Solutions to these problems are examined.

Chapter 4 – Agent Architectures: Background theory regarding agent-based systems is discussed in Chapter 4 with specific emphasis placed on blackboard communication and knowledge sharing.

Chapter 5 – Literature Review: Chapter 4 presents a literature review of previous work that has been done in the field of applications of evolutionary computation and agent-based reasoning to stock market trading and stock portfolio optimisation.

Chapter 6 – Design and Implementation: In Chapter 5 the system design for the evolutionary trading agents is proposed and described in detail later. This includes in-depth information about implementation details regarding the system components and the various technical indicators and their associated parameters.

Chapter 7 – Research Methodology and Results: The experiments used to test the effectiveness of the evolutionary system are presented in Chapter 6. The results thereof are discussed and conclusions drawn regarding the reasons for the observations.

Chapter 8 – Conclusions and Future Work: Chapter 7 concludes the dissertation with a summary of the experimental results from the system and discusses possible extensions for future work.

2. PORTFOLIO THEORY AND TECHNICAL ANALYSIS

"...Look at market fluctuations as your friend rather than your enemy; profit from folly rather than participate in it..." – **Warren Buffet**

This chapter provides an introduction to portfolio theory and technical analysis, with an emphasis on risk-return measurement and portfolio composition tools. It describes the major portfolio performance measurements and provides an in-depth discussion of the most popular technical analysis methods.

2.1 PORTFOLIO THEORY

Portfolio theory was introduced by Markowitz [3] in his paper "Portfolio Selection" published in the 1952 Journal of Finance. A portfolio is defined as the collection of stocks held by a trader at any given time. Previous work on the subject focused on assessing the risks and resulting rewards of individual securities when constructing portfolios [4]. It was common practice to identify securities that offered the best gains with the least amount of risk. By applying these notions, an investor might conclude that fast-food stocks all offer good risk-reward ratios and create a portfolio consisting entirely of these. Common sense deigns that this approach would be foolish. Instead, Markowitz [4] proposed the principle of diversification within portfolios. In fact he was the first to quantify risk and demonstrate quantitatively the value of diversification. His findings illustrated the way in which diversification works to reduce risk and optimise returns.

This section deals with the underlying theories of portfolio management and introduces the concepts of pricing models, active versus passive management and technical analysis as well as some of the more popular technical indicators.

2.1.1 CAPITAL ASSET PRICING MODEL

Sharpe [3] later furthered Markowitz's work and went on to formalise the capital asset pricing model (CAPM). According to the CAPM all investors should hold the market portfolio and leverage or un-leverage it with positions in the risk-free asset.

The CAPM predicts the expected return of any security given the expected return on the market, the security's beta and risk-free rate [5]. The prediction is given by and is based on the security-market line which is a trade-off between expected return and the beta risk of the security relative to the market portfolio:

$$E(r_i) = r_f + \beta_i [E(r_m) - r_f] \tag{2.1}$$

where:

- $E(r_i)$ = expected return of security
- r_f = risk-free rate
- β_i = security beta (referred to as beta henceforth)
- $E(r_m)$ = expected return on market

The underlying theory is based on the belief that the value of an asset reflects the risk associated with it given the investors' combination of risk-free asset and market portfolio [6]. The market portfolio refers to a hypothetical portfolio that contains every available security in the market in amounts proportional to their market values. It is assumed that a risky asset has no effect on the risk-free rate, but will instead affect the portfolio as they are correlated.

Several simplifying assumptions are made with regards to the validity of the predictions. It is assumed that [5]:

- CAPM is a one period model.
- All investors have the same information at the same time.

- Asset returns are normally distributed.
- Investors can borrow or lend unlimited amounts of a risk-free asset at a constant rate.
- There are a fixed number of assets and their quantities are fixed within the model world.
- All assets are perfectly divisible and are priced as if there is perfect competition. Perfect competition is a hypothetical economic model in which no producer or consumer has market power to influence prices.
- The borrowing rate equals the lending rate.
- There are no market imperfections such as taxes.

While the CAPM is popular for its simplicity, its main weakness is the assumptions that are made. Questions can be raised as to how best to estimate beta and whether or not beta can sufficiently summarise all possible risk factors [7].

2.1.2 ACTIVE vs. PASSIVE MANAGEMENT

There are two schools of thought regarding portfolio management: active and passive. Whereas active management revolves around stock picking and market timing[1], passive management refers to a buy-and-hold[2] approach [8].

2.1.2.1 ACTIVE MANAGEMENT

The primary objective of active portfolio management is to identify securities that are considered under-priced [9]. These securities are considered to be under-priced as market managers' forecasts differ from that predicted for the market. This concept of under-pricing, however, implies a disregard of the CAPM belief that all securities are priced accurately. In addition, it is only worthwhile given that the returns on under-priced securities outweigh the maintenance cost of portfolio management. Manager fees and analyst compensation are examples of the types of costs incurred using this management technique [9].

[1] The act of predicting the future direction of the market
[2] Stocks are bought at the start of a holding period and sold at the end thereof

Active portfolio managers are divided into three distinct groups, namely market timers, sector selectors and security selectors [9].

Market timers aim to change their portfolio beta[3] according to market forecasts [9]. If the forecast is bullish, they increase their beta above that of the market portfolio. Securities with a higher beta than that of the market will result in higher appreciation. The reverse holds true for bearish predictions.

Sectors are classified by industries, products or market-related characteristics for example size, growth and cyclical nature. Sector selectors try to increase their exposure to a certain sector when they believe the sector will perform above average and vice versa [9].

Security selectors employ the most common form of active management. They try to identify securities with the highest expected returns [9]. By identifying such securities and increasing their exposure to them, they realise higher returns if their predictions are correct. Once again, though, the idea of higher expected returns rejects the concept of accurate pricing in the CAPM.

2.1.2.2 PASSIVE MANAGEMENT

Passive traders attempt to construct a portfolio that replicates the return pattern of a specified index [9]. The simplest method to achieve this is to exactly replicate the index. Replication is often difficult and expensive as there must be a trade-off between accuracy and turnover cost. Tracking error, however, is low relative to other methods and is defined as [9]:

$$TE = \sigma(\overline{r_p} - \overline{r_m}) \tag{2.2}$$

where:

---

[3] The weighted average of the individual betas of the securities held

- TE = tracking error
- $\overline{r_p}$ = return on portfolio p
- $\overline{r_m}$ = return on market portfolio m
- σ = standard deviation

The primary issues involved in indexing are the selection of securities for the tracking portfolio and the number of securities to include [8]. The best securities to include are those that have low residual risk, low bid-ask spreads and high liquidity [9]. The number of securities to include is dominated by transaction costs as the more securities included, the lower the tracking error and the larger the transaction costs.

The general procedure followed by active and passive managers is:

- Risk analysis
- Portfolio selection
- Performance measurement

2.1.3 RISK ANALYSIS

To some degree both active and passive managers are interested in the risk-return ratio of securities. The returns on securities consist of two components [8]:

- the risk-free interest rate
- the risk premium

The risk-free interest rate is the interest rate that is assumed to be obtainable by investing in financial instruments with no default risk. In practice most professionals use short-dated government bonds for this purpose. The risk premium is the minimum difference between the expected value of the security and its realised value. An important assumption in portfolio theory is that risk premiums are predictable and can therefore be measured by the average historical risk premiums. In light of the above, one can differentiate between two types of risk [5]: non-systematic and systematic. Non-systematic risk refers to the risk that appears in the portfolio construction process when there is diversification among assets that are correlated. Systematic risk is the risk that remains after a portfolio, which is assumed to contain all risky assets, has been constructed and which cannot be reduced through diversification.

Variance is a commonly used measure of portfolio risk and measures the variability of the realised returns around an average level [9]. Larger values of variance imply higher levels of risk.

Covariance is a measure that depends on the way in which securities interact and identifies whether or not the returns of two securities are correlated [8]. It cannot, however, determine the degree of correlation. To this effect, it is standardised by dividing the covariance by the product of the standard deviations of the securities, resulting in a correlation coefficient in the range of -1 to 1. A coefficient of +1 signifies that the securities are moving in the same direction whereas a value of -1 indicates that they are moving opposite directions. Finally, a value 0 means that there is no correlation between the securities.

2.1.4 PORTFOLIO SELECTION

Active and passive management are related in that portfolio selection, in both, attempts to maximise the returns on the portfolio while minimising the risk incurred.

2.1.4.1 ACTIVE

Active portfolio management is reliant on stock picking techniques [9]. This method assumes that the manager has knowledge above and beyond the common information available regarding expected returns. Typically, managers select stocks that they consider as under-priced and then construct a well-diversified portfolio from them. This risk-averse strategy is valid as the pricing equation is only approximate and is subject to noise and individual stocks still have associated company-specific risk despite being mispriced (mispriced securities are securities with non-zero alpha's) [8].

The most common risk-averse strategy is the Treynor-Black model. The steps in stock selection for this method are as follows [8]:

- Select a small set of securities (Active portfolio A) that are thought to be mispriced.
- Combine the active portfolio with the passive benchmark portfolio *m* to diversify.
- Calculate a new capital allocation line (CAL). The CAL is a line graph of all possible combinations of risky and risk-free assets.
- Use a utility function to determine an optimal portfolio.

The model can be expressed mathematically as follows:

$$\max_{w} SR_p = \frac{E(r_p) - r_f}{\sigma_p}$$

$$r_p = w \times r_A + (1-w) r_m \tag{2.3}$$

where:

- SR_p = Sharpe ratio of portfolio p
- $E(r_p)$ = expected return on portfolio p
- r_f = risk-free rate of return
- r_p = return on portfolio p
- w = weighting in active portfolio A
- r_A = return on active portfolio A
- r_m = return on market

The optimal amount to put into the active portfolio is then determined by:

$$w^* = \frac{w_0}{1+(1-\beta_A)w_0}$$

$$w_0 = \frac{\dfrac{\alpha_A}{\sigma^2(e_A)}}{\dfrac{E[r_m - r_f]}{\sigma^2{}_m}} \tag{2.4}$$

where:

- w^* = optimal weight to put into active portfolio A
- w_0 = optimal weight to put into active portfolio A without adjustment for β
- β_A = beta of active portfolio A
- α_A = alpha of active portfolio A
- $\sigma^2(e_A)$ = standard deviation of residual returns on active portfolio A
- r_m = return on market
- r_f = risk-free rate of return
- $\sigma^2{}_m$ = standard deviation of market

From the equation above, it follows that a weight of (1-w*) should be put into the benchmark (market) portfolio [8].

2.1.4.2 PASSIVE

Most index-fund managers utilise performance enhancements as they would perform below the index otherwise due to transaction costs [9]. In order to maximise the probability of outperforming the index, the portfolio returns relative to that of the benchmark must be optimised. This inequality is defined as:

$$\max_{x_1,\ldots,x_n} E(r_p - r_m) = E([\sum_{i=1}^{N} x_i r_i] - r_m) \tag{2.5}$$

where:

- r_p = return on tracking portfolio p
- r_m = return on market portfolio m
- x_i = weight of security i
- r_i = return on security i

The inequality is subject to the constraint that the variance must be below some threshold:

$$\sigma(r_p - r_m) = \sigma([\sum_{i=1}^{N} x_i r_i] - r_m) \leq \overline{\sigma} \tag{2.6}$$

where:

- r_p = return on tracking portfolio p
- r_m = return on market portfolio m
- x_i = weight of security i
- r_i = return on security i
- $\overline{\sigma}$ = threshold value

The problem can be worded instead as minimising the variance of (r_p-r_m) subject to realising the specified expected returns. Since the portfolio weights must sum to 1 the problem can be rewritten as:

$$\min_{w_1,\ldots,w_n} \sigma(\sum_{i=1}^{N} w_i(\overline{r_i} - \overline{r_m})) \tag{2.7}$$

where:

- N = number of securities
- w_i = weight of security i

- $\overline{r_i}$ = average return on security i
- $\overline{r_m}$ = average return on market m

subject to:

$$E(r_p - r_m) = E(\sum_{i=1}^{N} w_i(\overline{r_i} - \overline{r_m})) \geq \overline{R} \qquad (2.8)$$

where:

- N = number of securities
- r_p = return on tracking portfolio p
- r_m = return on market portfolio m
- w_i = weight of security i
- $\overline{r_i}$ = average return on security i
- $\overline{r_m}$ = average return on market m

The solution to this problem is then a set of portfolio weights, $w_1,\ldots,w_n$, that denotes the optimal tracking portfolio.

2.1.5 PERFORMANCE MEASUREMENTS

The theory supporting portfolio performance measurement is that the return of the portfolio is adjusted for the associated risk in the holding period. These adjustments are based on either the security-market line (Treynor Index, Jensen's Alpha) or capital-market line (Sharpe Ratio, Risk-adjusted-performance Ratio).

2.1.5.1 SECURITY-MARKET LINE BASED MEASUREMENTS

The security-market line (SML) graphs the systematic risk against the return of the market at a certain time and indicates all risky securities [5]. It plots the result from the CAPM formula, where the x-axis represents the risk and the y-axis the expected return. The market risk premium can then be determined from the slope of the SML. If the security's risk is above the SML, it is considered undervalued as the investor can expect greater returns for the same risk [5]. Similarly, if the risk is below the SML, it is considered overvalued since the returns would be lower for the same risk.

2.1.5.1.1 TREYNOR INDEX

The Treynor Index is based on the assumptions of the CAPM using the SML as a benchmark [7]. It relates the expected rate of return to the rate of return of the market average. When it is plotted on a graph, the line that represents it tends to be stationery despite fluctuation in the short-term rate of return. The slope of the resulting line measures volatility and therefore provides a measure of risk. It then follows that the steeper the slope, the more sensitive the rate of return is to fluctuations and vice versa [5]. In addition, portfolios can then be ranked based on their respective slopes.

$$T_p = \frac{(\bar{r}_p - r_f)}{\beta_p} \tag{2.9}$$

where:

- T_p = Treynor Index for portfolio *p*
- $\bar{r}_p$ = average return of portfolio *p*
- r_f = average risk-free rate
- β_p = beta for portfolio *p* (sensitivity of portfolio to market return changes)

2.1.5.1.2 JENSEN'S ALPHA

Like the Treynor Index, Jensen's Alpha also uses the assumptions of the CAPM using the SML as a benchmark. It essentially measures the difference between the actual returns on the portfolio in any holding period and the expected returns on the portfolio dependent on the risk-free rate and the actual returns on the market portfolio [7]. In other words, it measures the difference of a portfolio's return from the return predicted by the CAPM adjusted for the systematic risk of the portfolio. Jensen's Alpha is calculated by finding the intercept in the regression:

$$E(r_p) = \alpha_p + \beta_p(E(r_m) - r_f) + \varepsilon_p \tag{2.10}$$

where:

- $E(r_p)$ = estimated returns on portfolio p
- α_p = Jensen's Alpha for portfolio p
- β_p = beta for portfolio p
- $E(r_m)$ = estimated returns on market m
- r_f = risk-free rate of return
- ε_p = systematic risk of portfolio p

Giving the following equation for the Jensen Alpha (also known as Jensen Index):

$$\alpha_p = \bar{r}_p - r_f - [\bar{r}_m - r_f]\beta_p \tag{2.11}$$

where:

- α_p = Jensen's Alpha for portfolio p
- $\bar{r}_p$ = average return of portfolio p
- r_f = average risk-free rate
- $\bar{r}_m$ = average return of market m
- β_p = beta for portfolio p

According to Jensen, portfolio performance is said to be neutral if the portfolio's historical returns equals the returns implied by the CAPM ($E(\alpha_p)=0$)[4] [11]. Similarly, superior portfolio performance can identified by $E(\alpha_p)>0$ (the reverse also holds).

[4] Calculated by substituting α_p into the CAPM equation (3)

This implies that mean returns on the portfolio are consistently greater than that implied by its level of systematic risk.

2.1.5.2 CAPITAL-MARKET LINE BASED MEASUREMENTS

The capital-market line (CML) is used to illustrate the rates of return for efficient portfolios depending on the risk-free rate of return and the associated level of risk [5]. According to the CAPM, the market portfolio is the efficient frontier. The CML can therefore be derived by drawing a line from the intercept point of the efficient frontier to the point where the expected return equals the risk-free rate of return [5].

2.1.5.2.1 SHARPE RATIO

The Sharpe Ratio uses the assumptions of the CAPM, but uses the CML as benchmark instead of the SML. It measures the risk premium earned per unit of risk exposure and is given by the following equation [7]:

$$S_p = \frac{(\overline{r_p} - \overline{r_F})}{\sigma_{r_p}} \tag{2.12}$$

where:

- S_p = Sharpe ratio for portfolio p
- $\overline{r_p}$ = average return on portfolio p
- $\overline{r_F}$ = average return on risk-free asset
- σ_{r_p} = standard deviation of return on portfolio p

Higher values of S_p indicate superior performance and imply that the investor receives more compensation for the same increase in risk.

2.1.5.2.2 RISK-ADJUSTED PERFORMANCE

Modigliani [9] proposed a modified version of the Sharpe Ratio, the Risk-Adjusted Performance (RAP) ratio. Whereas Sharpe ranks funds according to the slope of the CML, Modigliani levers or un-levers the portfolio's risk to match the market risk and use the risk-adjusted return as a ranking variable. Sharpe and RAP produce the same ranking.

$$RAP_p = \frac{\sigma_m}{\sigma_p}(\bar{r}_p - r_f) + r_f \tag{2.13}$$

where:

- RAP_p = risk-adjusted performance ratio for portfolio p
- $\bar{r}_p$ = average return of portfolio p
- r_f = average return of risk-free rate
- σ_m = ex-post standard deviation of market m
- σ_p = ex-post standard deviation of portfolio p

The relationship between Sharpe and RAP is given by:

$$RAP_p = SR_p \times \sigma_m + r_f \tag{2.14}$$

where:

- RAP_p = risk-adjust performance ratio for portfolio p
- SR_p = Sharpe ratio for portfolio p
- σ_m = standard deviation of market m
- r_f = average return of risk-free rate

Similar to the Sharpe Ratio, higher values of RAP indicate stronger portfolio performance.

2.1.6 PROBLEMS

The main problem associated with these performance measures stems from the use of an estimated benchmark. If the SML is incorrectly estimated, the market index will be inefficient and this will negatively impact the Treynor Index and Jensen's Alpha. This is primarily caused by two factors [9]:

1. The true risk-free rate is not that the same as the risk-free rate used in the model. This situation can arise if an investor cannot borrow at the assumed risk-free rate used in the model.
2. A non-optimised marked index is used. An index is non-optimised if the expected return differs from the expected return of the optimised index.

These factors cause the SML to be positioned incorrectly. The Sharpe Ratio measures portfolio performance by assuming that a linear relationship exists between total risk and excess return over the risk-free rate [9]. Accordingly, if an investor is forced to pay higher interest rates, the higher assumed level of risk will lead to the misclassification of funds.

2.2 TECHNICAL ANALYSIS

Market analysis can be categorised into two approaches: fundamental analysis and technical analysis [5]. The former focuses on analysing the attributes of a company to estimate its value, while the latter ignores the intrinsic value of securities and instead relies on the study of technical indicators to determine trends that are likely to continue in future.

According to Skiena [10], technical analysis covers a class of investment strategies that "analyze patterns of past behaviour for future predictions". It is based on the following assumptions:

- Market value is determined by supply and demand *only*
- Stock prices tend to move in trends for long periods of time
- Shifts in supply and demand cause reversals in trends
- Shifts can be detected in charts/graphs
- Chart patterns repeat themselves

Technical analysis has not been proven to be a definite science and it still attracts many sceptics. Much of the criticism levelled at technical analysis stems directly from the efficient market hypothesis [5]. According to this theory, the market price is the correct one for any security and therefore any analysis to discover undervalued securities is useless.

There are an immense number of charts and graphs available that can be manipulated to form technical indicators. These indicators can, however, be reasonably grouped into five categories [11]:

- **RELATIVE STRENGTH INDEX**

  Relative strength index indicators compare the recent and historical gains and losses in relation to its historical strength and weakness.

- **TRADING RANGES**

  Support and resistance lines are drawn to border the trading range. A breakout above or below the range occurs when the price sustains movement above or below the range.

- **PATTERN ANALYSIS**

  Charts are analysed for a series of accepted patterns, including head and shoulders, triangle up or down, rounded tops or bottoms and cup-and-handle formations.

- **TREND ANALYSIS**

The most common form of trend analysis involves looking for crossovers of multiple trend lines, the most of common of which are the moving average indicators.

- **GAP ANALYSIS**

  A gap occurs when the opening price of a stock significantly differs from the closing price of the previous period.

2.2.1 MOVING AVERAGE CONVERGENCE/DIVERGENCE

The Moving Average Convergence/Divergence (MACD) technical indicator is one of the simplest and more reliable indicators used by traders. It utilises moving averages that are inherently lagging indicators and as such, incorporates trend-related characteristics [12]. These lagging indicators are subsequently transformed into momentum oscillators by subtracting indicators for long periods from that for short periods [13]. Once plotted, the resulting graph oscillates above and below the zero-line without bounds.

$$MA = \frac{\sum_i (n-i+1)p}{\sum_i (n-i+1)}$$

$$MACD = MA_{slow} - MA_{fast} \tag{2.15}$$

where:

- n = number of periods
- p = closing price
- MA_{slow} = slow MA
- MA_{fast} = fast MA

The standard MACD calculation is simply the difference between the 26-day (slower as the measurement changes slower due to the increased number of data points) and 12-day (faster) exponential moving averages. These values can be tailored, however, to better suit a faster of slower security. The shorter average implies a quicker more responsive indicator, while the longer average results in a slower indicator less prone to anomalies like whipsaws. An example of these calculations is given in graph form in figure 2.1.

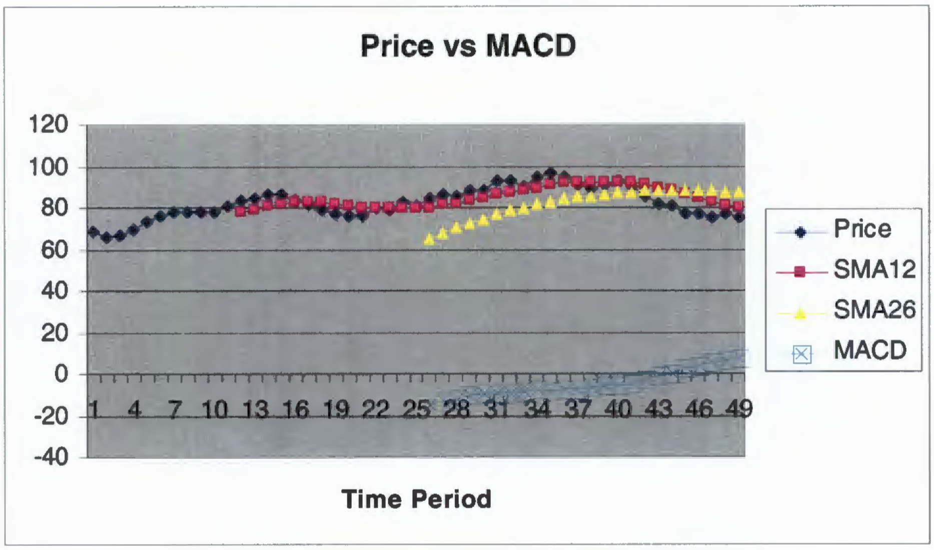

Figure 2.1: Comparison of price vs. MACD

A positive-valued MACD implies that the slower MA is greater than the faster MA, while the converse also holds. In the event that the MACD is positive and rising, the gap between the two MA's is also necessarily increasing. This widening gap, in turn, implies that the rate of change of the faster MA is greater than that of the slower MA and therefore the increasingly positive momentum implies a bullish market. In figure 2.1 above, this phenomenon can be observed as the yellow line crossing above the purple line at around the 45^{th} time period. The converse also holds true and an increasingly negative momentum implies a bearish market. The three trend lines (slow, fast and convergence-divergence) can be clearly seen in figure 2.1 above. There is also a clear indication of the positive crossing over of the convergence-divergence line, indicating that the slower MA is greater than the faster MA.

2.2.2 RELATIVE STRENGTH INDEX

The relative strength index (RSI) indicator compares the magnitude of a stock's recent gains to the magnitude of its recent losses and normalises the result into the range 0 to

100 [14]. A high RSI is obtained when the market has rallied sharply and, conversely, a low RSI occurs when the market has fallen sharply [15].

$$RSI = 100 - (\frac{100}{1 + RS})$$

$$Ave_{GAIN} = \frac{Total_{GAIN}}{n}$$

$$Ave_{LOSS} = \frac{Total_{LOSS}}{n} \qquad (2.16)$$

$$RS = \frac{Ave_{GAIN}}{Ave_{LOSS}}$$

where:

- RSI = relative strength index
- RS = ratio of average gains to average losses
- Ave_{GAIN} = average gains over n periods
- Ave_{LOSS} = average losses over n periods
- n = number of periods

An example of RSI for a fictitious stock is presented in figure 2.2. Of importance is the crossing over of the 70-level, indicating an overbought level.

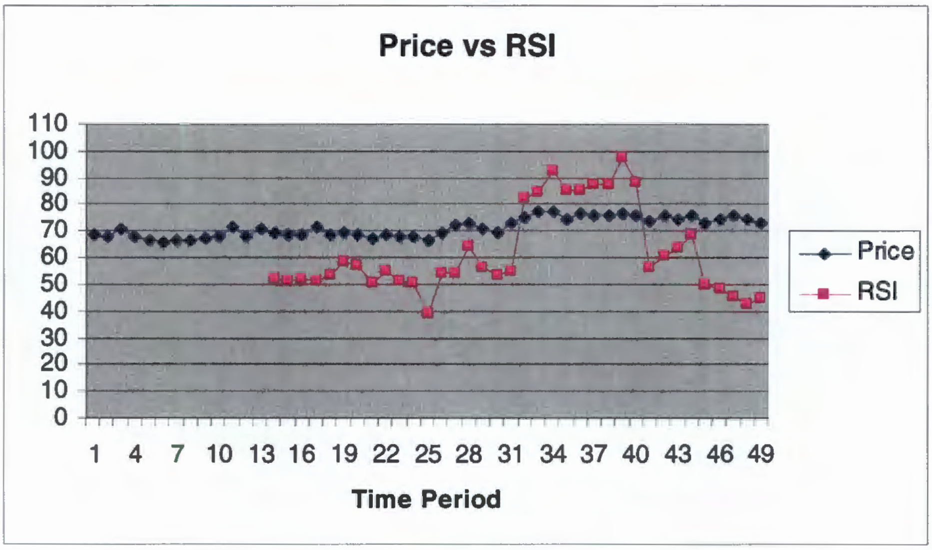

Figure 2.2: Comparison of price vs. RSI (n=14)

An important attribute of the RSI indicator is that it slows down when it reaches very overbought or oversold conditions and then jumps back when the market corrects even slightly [19]. This results in the RSI returning to more neutral levels and indicates that the price trend might continue.

It is recommended to use 70 and 30 respectively as overbought and oversold levels [13]. If RSI is greater than 30 it implies a bullish signal for the stock, whereas if RSI less than 70 it implies a bearish signal. Some traders prefer to identify long-term trend and use extreme readings as entry points, e.g. if a long-term trend is bullish, oversold readings serve as entry points.

Similar to overbought and oversold levels, the standard centreline for RSI is 50 [13]. A value greater than 50 implies that the average gains are higher than the respective average losses and vice versa. Some investors accept a move above the centreline as a bullish signal and below the line as a bearish signal.

2.2.3 RATE OF CHANGE

Rate of change (ROC) is a very simple but effective momentum oscillator. It measures the percentage change in price from one period to the next [16].

$$ROC = \frac{p_t - p_{t-1}}{p_{t-1}} \tag{2.17}$$

where:

- t = current time period
- p_i = price at time t

A graph plot of the indicator forms an oscillator that fluctuates above and below the zero line as the ROC changes from positive to negative. The longer the time span used to calculate it, the greater the fluctuation in its magnitude and duration. In the event that the ROC crosses up through the zero line, a buy signal is generated, whereas a downward crossing results in the generation of a sell signal. Both cases of crossing over can be observed in figure 2.3 below as the ROC crosses positively and negatively through the zero-line on a number of occasions.

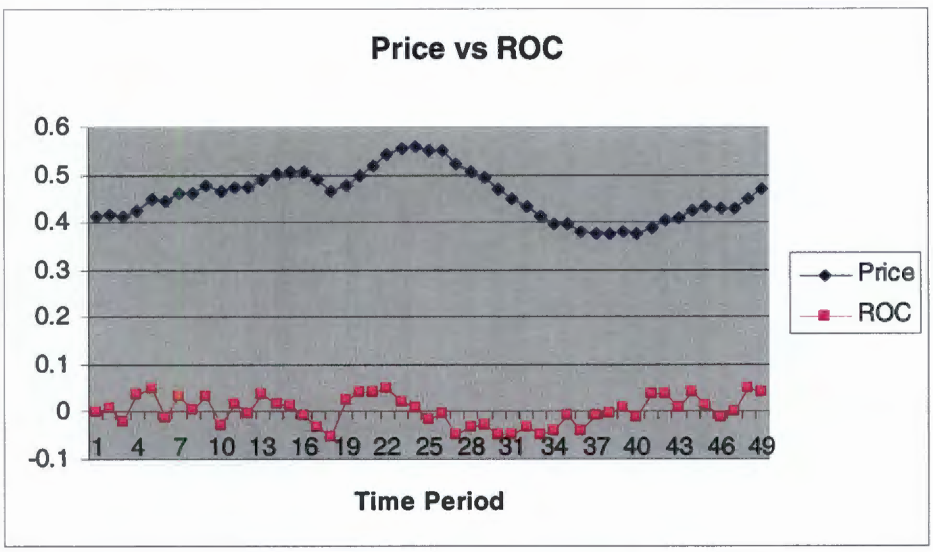

Figure 2.3: Comparison of price vs. ROC

2.2.4 STOCHASTIC OSCILLATOR

The stochastic oscillator indicator is a momentum indicator that shows the location of the current close relative to the high or low range over a set number of periods [17]. If the closing levels are close to the top of the range, it implies accumulation (buying pressure) whereas if it is near the bottom of the range, it implies distribution (selling pressure) [17]. An example of such a graph is provided in figure 2.4.

$$\%K = 100 * (p_{close} - p_{lowest}) / (p_{highest} - p_{lowest})$$
$$\%D = MA_3(\%K)_3 \tag{2.18}$$

where:

- %K = fast oscillator
- %D = slow oscillator
- p_{close} = recent close
- p_{lowest} = lowest low
- $p_{highest}$ = highest high
- MA_3(%K) = 3-period moving average of %K

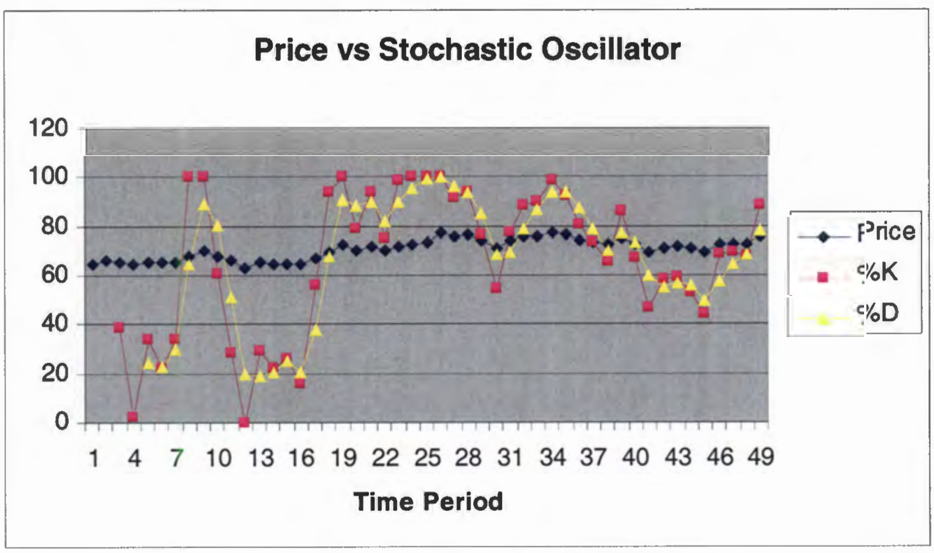

Figure 2.4: Comparison of price vs. stochastic oscillator

One can distinguish between three distinct types of oscillators: fast, slow and full [17].

2.2.4.1. FAST

The fast stochastic oscillator requires two parameters that determine the number of periods used to create the %K and %D lines: (number of periods to use, number of periods to smooth %K) [17]. The greater the value of the parameter for %D, the smoother the %D line will be as it represents a slower MA of %K. Figure 2.5 below serves as an example plot of the fast stochastic oscillator.

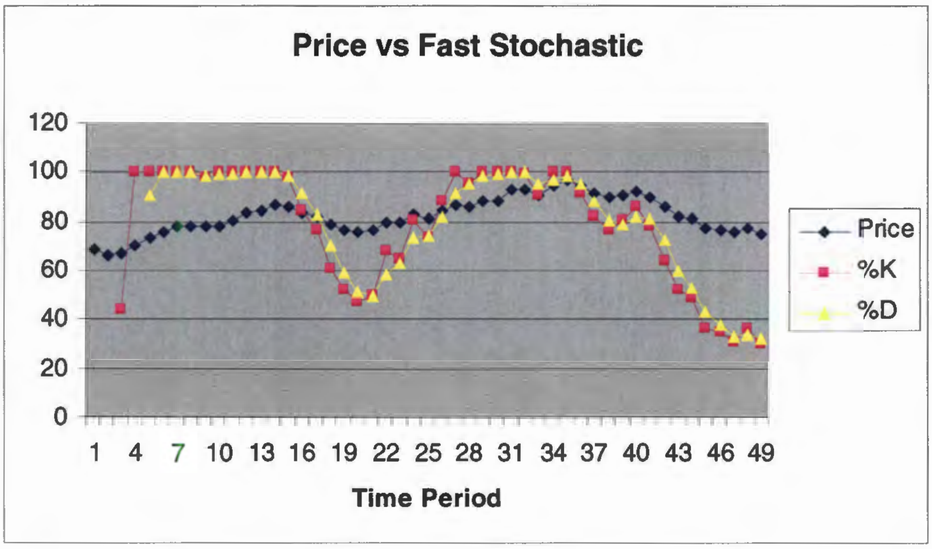

Figure 2.5: Comparison of price vs. fast stochastic oscillator

2.2.4.2. SLOW

Similar to the fast stochastic, the slow version requires only the two parameters mentioned previously: (number of periods to use, number of periods to smooth %K). It differs, however, by virtue of the fact that the slow %K line is actually a 3-period MA of the fast %K line [17]. Similarly, the %D line will be an n-period MA of the slow %K line. The differences between the fast and slow stochastic oscillator are clearly visible from figures 2.5 (above) and 2.6 (below). The stochastic is presented as a smoother line and this results in fewer crossings and hence, fewer false signals.

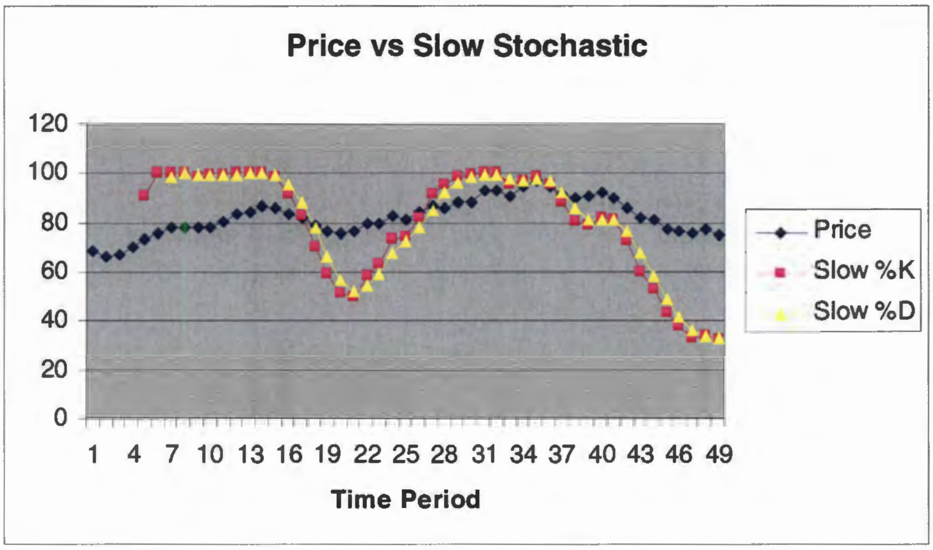

Figure 2.6: Comparison of price vs. slow stochastic oscillator

2.2.4.3 FULL

The full stochastic oscillator requires three parameters [17]:

- o number of periods used to create the %K line
- o number of periods used to create the %D line
- o smoothing factor for the %K line (%K is then plotted as an n-period MA of the initial %K)

The presence of the smoothing factor makes the full stochastic much more flexible than either the fast or slow versions. To illustrate this fact, it can be used to duplicate the other versions, e.g. a (14, 3) fast stochastic is the same as a (14, 1, 3) full stochastic and a (12, 2) slow stochastic is the same as a (12, 3, 2) full stochastic.

There are a number of ways to interpret the stochastic oscillator, of which the most popular are to [13]:

- o buy when the oscillator falls below a specific level and then rises above that level (vice versa) – 20 is usually thought to be oversold and 80 is considered overbought
- o buy when the %K line rises above the %D line (vice versa)

2.2.5 SUPPORT/RESISTANCE LEVELS

Support/resistance levels refer to the price levels at which price movement should stop and then reverse [13]. It, therefore, acts as a floor or ceiling to future movements.

The support level is the price below the current market price at which buying interest should overcome the selling pressure and prevent the price from falling further [13]. Accordingly, the resistance level is the price level above the current market price at which selling pressure is sufficiently strong to overcome the buying pressure and keep the price from rising.
Two important characteristics of these levels are [13]:

- Support/resistance levels reverse roles once the price level crosses through it.
- Support/resistance levels vary in strength such that certain levels are considered major and other minor, e.g. a 10-year high on a weekly chart is considered major, whereas a 2-week trend line intersection point is considered relatively minor

Support/resistance levels can be identified by applying either technical or fundamental analysis. Those levels that can be identified and are based on technical analysis include the following [17]:

- Recent major highs or lows
- Moving averages
- Retracement levels
- Pivot points
- Gaps
- Trend line intersections
- Pattern areas
- Congestion or high-volume areas

In general, support/resistance levels can be leveraged in one of two ways. It can either be used to assess the market direction or it can be used for scalping and order placement [17].

If one considers support/resistance levels to be prizes which buyers and sellers compete for, it can be used to assess the market direction [17]. The "winner" of the competition temporarily holds a stronger position, has control of the market and can ultimately drive the market in a certain direction. If, for example, the market can rally above a certain resistance level, one can assume that buyers have the upper hand and it can be expected that the market will climb higher.

These levels can also be used to identify scalping points and price levels at which to place orders [17]. As an example, a trader might want to sell short just under a certain level as the price will stop and sell off from that level. This results in the trader realising a small profit when his/her short position[5] is covered.

Long-term traders can use support/resistance levels for order placement to enter new positions or place protective stops [17]. In the case that the trader wants to buy into a market, it is prudent to wait until it falls to a level just above the support level. The trader will then place a protective sell-stop one or two levels lower just in case the market shows weakness by falling further.

The trader can also enter a new position by placing a sell-stop just beyond the support/resistance level [17]. The market would be expected to move more quickly in the direction once the level has been penetrated.

2.3 CONCLUSION

This chapter presented a broad overview of portfolio theory and technical analysis. It introduced the concepts of active and passive trading, systematic and non-systematic risk, portfolio performance measurements and various technical indicators that are

[5] Sale of a borrowed security expecting that it will fall in value

used to identify trends in securities data. Finally, the problems and possible workarounds associated with measurements and predictions were discussed.

3. EVOLUTIONARY ALGORITHMS

"...It is not the strongest of the species that survives, nor the most intelligent that survives. It is the one that is the most adaptable to change..." – **Charles Darwin**

This chapter gives a brief overview of evolutionary algorithms, more specifically genetic algorithms (GA's) and genetic programming (GP). It describes the processes involved in each algorithm as well as the stumbling blocks presented by each and established solutions to circumvent these. Background theory on these algorithms is important as both are used in the framework design to solve different problems.

3.1 GENETIC ALGORITHMS

"Ignorance more frequently begets confidence than does knowledge: it is those who know little, not those who know much, who so positively assert that this or that problem will never be solved by science." - **Charles Darwin** (Introduction to The Descent of Man, 1871)

Some problems simply cannot be solved *exactly* in polynomial time and include such problems as job shop scheduling, bin packing and travelling salesman problems. These problems generally have state spaces that are too large to search exhaustively in reasonable time. Though no optimal solution currently exists for them, several approximations exist, few more appropriate and more general than GA's.

This section introduces the concept of GA's and describes in detail the analogy of evolutionary processes with regards to problem solving.

GA's belong to a family of guided random search algorithms based on the evolutionary ideas posed by Darwin of natural selection and survival of the fittest [18]. Given time, it directs the search into regions providing better performance within the search space and offers result-oriented advantages over typical search optimisation algorithms within large state spaces [19].

GA's are based on the following fundamental foundations [20]:

- Individuals in the population will compete for resources and mating partners
- Successful individuals will produce more offspring than poor-performing individuals
- "Good" genes will propagate throughout the population as well-performing parents will sometimes breed offspring that are better than themselves
- Based on the above, successive generations will adapt to better suit the environment

Each individual of the population in the search space represents a candidate solution and is encoded as a finite length vector of variables usually in binary {0, 1} format [18] as can be seen in figure 3.1 below. The individuals are equivalent to chromosomes, while the variables are analogous to genes associated with the chromosomes.

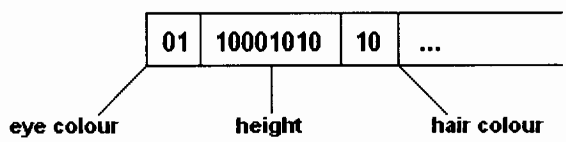

Figure 3.1: Binary encoding of sample chromosome

The steps performed in GA's are listed below [18]:

- Generate the initial population of random individuals
- Iteratively perform the following steps until the termination criterion are met:
    - Evaluate individuals in the population and assign a fitness value to each
    - Select parents from the population based on fitness
    - Perform crossover on the parents to form new a population
    - Perform mutation on the new population
    - Copy the new population to the current population

3.1.1 EVOLUTION

Chromosomes are modified via stochastic evolutionary operators that are synonymous with Darwin's theories. The search optimisation process therefore introduces two concepts: exploration and exploitation [20]. Exploration refers to the discovery of promising areas in the search space, while exploitation refers to optimisation within a promising area [20]. Whereas crossover is an example of an exploratory process, mutation is an example of an exploitative process.

3.1.1.2 MUTATION

The mutation operator introduces random modifications into individuals from the population. Portions of the individuals' genes are inverted with low probability to maintain diversity within the population and hence, prevent premature converge due to the search being trapped in a local minima [18].

Mutation alone would result in the algorithm performing a random walk through the search space, whereas utilising both selection and mutation would result in a parallel, noise-tolerant, hill-climbing algorithm [20].

3.1.1.3 CROSSOVER

The crossover operator represents mating between individuals in the population. The theory underlying the crossover operator is that recombining parts of successful individuals is likely to result in the creation of more successful individuals over time. Crossover, however, cannot be employed by itself as the use of crossover (and selection by implication) alone will cause the algorithm to converge on sub-optimal solutions [20].

3.1.1.3.1 ONE-POINT CROSSOVER

A single crossover point is selected in both parents. All data beyond the point in the parents is swapped between the children. The operation is presented in figure 3.2 below.

Figure 3.2: One-point crossover in two parents A & B

3.1.1.3.2 TWO-POINT CROSSOVER

Two identical crossover points are selected in both parents. All the data between these points in the parents are swapped between the children. The resulting children can be seen in figure 3.3.

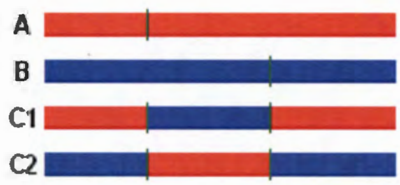

Figure 3.3: Two-point crossover in two parents A & B

3.1.1.3.3 CUT & SPLICE

Unlike the one- and two-point crossovers, the cut and splice method changes the length of the children. Different crossover points are selected in each parent and the data beyond each point in the parent is swapped in the children, resulting in the formation in figure 3.4.

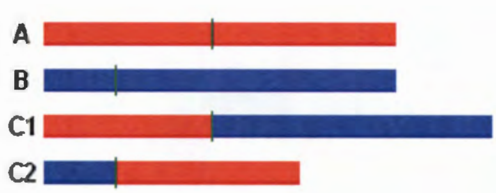

Figure 3.4: Cut and splice crossover in two parents A & B

3.1.1.4 RANDOM-ASSORTMENT RECOMBINATION OPERATOR

Genes are defined as pieces of genetic information that define certain properties within an individual. Alleles, then, are the values that a gene can assume. According to Radcliffe [21], the ideal crossover operator should have both respect and assortment. Whereas respect refers to the fact if two parents share alleles, those alleles should be present in their children, while assortment refers to the fact that any alleles of both parents should be available to the children as long as they are consistent and do not lead to the formation of illegal chromosomes [22].

This is possible for variable size sets as the optimum would be achieved when the subset of genes was the same as the complete set under full replication. A fixed size set would result in incompatibility between respect and assortment. Consider the example of two parents, each having five genes of which four are common to both. Respect demands that the four shared genes must be present in the child, meaning that there is only one available gene to be filled in the child. In this situation, a gene must be selected from one parent and the gene in the other excluded. This action, however, violates the assortment property.

Instead, an approximation can be used for fixed size sets using the Random Assorting Recombination operator (RAR) [22]. It takes as input two parents p and q and a positive integer w (weight) that represents the amount of importance given to respect over assortment. The higher the value of w the greater the importance of respect over assortment and the more likely shared alleles are to be selected for the resulting children.

Given the necessary inputs, the algorithm proceeds as follows [22]:

- Place w copies of each allele present in both parents in a bag
- Add one copy of each allele present in only one parent to the bag
- Repeatedly draw alleles from bag without replacement
    - When it is possible to add an allele to a child add it
    - If not discard the allele

    - Continue until the bag is empty or the child is fully specified (a basic allele has been chosen for each basic gene)
- If the child is not fully specified, assign alleles to the remaining genes at random from the set of legal values

3.2 GENETIC PROGRAMMING

". . . If we are interested in getting computers to solve problems without being explicitly programmed, the structures that we really need are computer programs..."
- **John Koza** (GP: On the Programming of Computers by Means of Natural Selection, 1992)

GP is an extension of GA's and uses a similar procedure to search and optimise a function based on the selective recombination of candidate solutions from the total population [23]. The major difference between GP and GA is that they differ in representation. GP's generally make use of a tree structure that is essentially a hierarchical model of interconnected nodes that resembles Lisp or Scheme program code [41]. Each node can have several connections to nodes in lower layers, but is only allowed one connection to nodes in upper layers (parents).

GP's represent functions in the form of program syntax trees. Nodes that branch to lower layers (branch nodes) are functions and take arguments passed by their immediate descendants (children) as input and return the resultant output to their parents [24]. Terminal nodes represent input arguments. In addition, branching between nodes denotes the ordering of evaluation.

Unlike GA's, GP trees can generate candidate solutions of variable size and complexity and there is no specific mapping of portions of the trees to parts of the candidate solution.

The general GP algorithm as formulated by Koza [25] is as follows:

- Generate the initial population of random trees consisting of functions/terminals
- Iteratively perform the following steps until the termination criterion are met:
    - Execute each program in the population and assign fitness values according to the accuracy of the solution
    - Create a new population by applying two primary operations. Operations applied to the population are chosen with probability based on fitness
    - Copy existing programs to the new population
    - Create a new program by genetically recombining randomly chosen parts of the two parent programs
- The best program that appears in any generation may be solution to problem

3.2.1 TREE GENERATION

The set of all possible trees is given by the set of all possible combinations of functions that can be composed from the set of n functions from $F=\{f_1, f_2, \ldots, f_n\}$ and the set of m terminals from $T=\{t_1, t_2, \ldots, t_m\}$ [26] . The functions and terminals thus selected are required to satisfy the requirements of closure and sufficiency [26].

According to the closure property, any function should be well-defined and closed for any combination of arguments [26]. Accordingly, boundary functions are important and functions should be well-defined for illegal inputs. Illegal inputs can be dealt with by associating a type with each node as well the types it can legally call. This enables the calling function to cast arguments to the appropriate type.

Sufficiency refers to the fact that the solution to the problem must be expressible in terms of the set of all terminals and functions [26].

Good random tree-generation algorithms are required to create the initial population of individuals and to create sub-trees during mutation. Tree creation also plays an important part in preventing code bloat, which is the tendency of trees to grow unbounded in the evolutionary process [27]. The most common methods for creating trees and sub-trees are the GROW algorithm, its full-tree variant, the FULL algorithm, and a combination of the two approaches, the RAMPED HALF-AND-HALF algorithm.

This section compares these tree creation algorithms and provides a basic overview of the steps involved in each.

3.2.1.1 GROW

The GROW algorithm starts with a set of functions F (terminals and non-terminals) with which to populate the tree [27]. The algorithm randomly selects a root from the full set and then proceeds to fill the root's parameters and then the children's parameters recursively. A common implementation is given below [26]:

Given:

    Maximum depth bound D
    Function set F consisting of non-terminal set N and terminal set T

Do:

    New tree T = GROW(0)

GROW(depth *d*)

    Returns: tree depth <= D - *d*

    If *d* = D, return random terminal
    Else
        Choose random function *f* from F
        If f is a terminal, return *f*
        Else
                For each argument *a* of *f*
                        Fill a with GROW(d+1)
                Return f with filled arguments

While the GROW algorithm is easy to implement and runs in linear time, it has a few disadvantages. It picks all functions with equal likelihood and hence, does not give the user much control over the tree structures that are generated [26]. In addition, while D is used as an upper bound on the maximal tree depth, there is no appropriate way to create trees with either a fixed or average depth [27].

Some variations of the GROW algorithm allow the user to specify the maximum tree size S. This can be achieved by continuously producing trees until one of size less than S is created [26]. Alternatively it can be enforced by keeping track of the number of created nodes and the number of unfilled arguments and when the total exceeds S, only terminals can be used to fill arguments.

3.2.1.2 FULL

The FULL algorithm is simply a full-tree variant of the GROW algorithm that *always* produces a full tree of depth D [26]. It is this property that results in the algorithm producing a very narrow range of structures with relatively less applicability than the GROW algorithm. The general algorithm is given below [27]:

Given:

    Maximum depth bound D
    Function set F consisting of non-terminal set N and terminal set T

Do:

    New tree T = FULL(0)

FULL(depth *d*)

    Returns: tree depth <= D - d

    If d = D, return random terminal
    Else
        Choose random function f from N
        For each argument a of f
                Fill a with FULL(d+1)
        Return f with filled arguments

3.2.1.3 RAMPED HALF-AND-HALF

The RAMPED HALF-AND-HALF algorithm generates trees by using the GROW or FULL algorithm, both with 50% probability. As there is no fixed size parameter, the RAMPED-HALF-AND-HALF algorithm does not have well-defined computational complexity in terms of size. The algorithm is provided below [26]:

Given:

    Maximum depth bound D
    Function set F consisting of non-terminal set N and terminal set T

Do:

    New tree T = RAMPED(0)

```
RAMPED()
    Returns: tree depth <= D

    Choose random value d in range 1-D
    D = d
    If random value p > 0.5
        Return GROW(0)
    Else
        Return FULL(0)
```

3.2.2 EVOLUTION

This section introduces the concepts of fitness and selection and provides a comparison of the effectiveness of the two most common evolutionary operators: mutation and crossover.

3.2.2.1 FITNESS

The fitness function of a GP describes the way in which the system evolves. While individuals are rated on their associated performance, it is necessary to rate these individuals on a similar scale so that they can be compared and the more successful individuals distinguished from the less successful.

The raw fitness is the value assigned directly and is also described as the error measure. The adjusted fitness is the fitness of the individual normalised to the range 0 to 1 (with 1 being ideal) and is given by the following formula [28]:

$$f_{adjusted} = \frac{1}{1 + f_{raw}} \tag{3.1}$$

where:

- $f_{adjusted}$ = adjusted fitness value
- f_{raw} = raw fitness value

The addition of 1 to the denominator prevents a divide-by-zero error and increases the difference between two individuals at the higher end of the scale.

3.2.2.2 SELECTION

Selection is the evolutionary process that determines which individuals (parents) will be chosen for reproduction and how many children each parent will produce. This process not only determines which individuals will undergo crossover and mutation, but it also helps to guide the stochastic search towards more promising areas of the state space.

Bias and spread are two important concepts in selection. The former refers to the absolute difference between an individual's normalised fitness and its expected probability of reproduction while the latter refers to the range of values for the number of children of an individual.

3.2.2.2.1 FITNESS PROPORTIONATE (ROULETTE WHEEL)

Fitness proportionate selection, also known as stochastic sampling with replacement, is the simplest selection scheme [29]. If one considers the total sum of the individuals' fitness as the circumference of a circle, each individual can be represented by a sector of the circle equal in size to its fitness value. The probability of being in a given sector is then proportional to the fitness of the individual (hence the alternative name) as presented in figure 3.5 below.

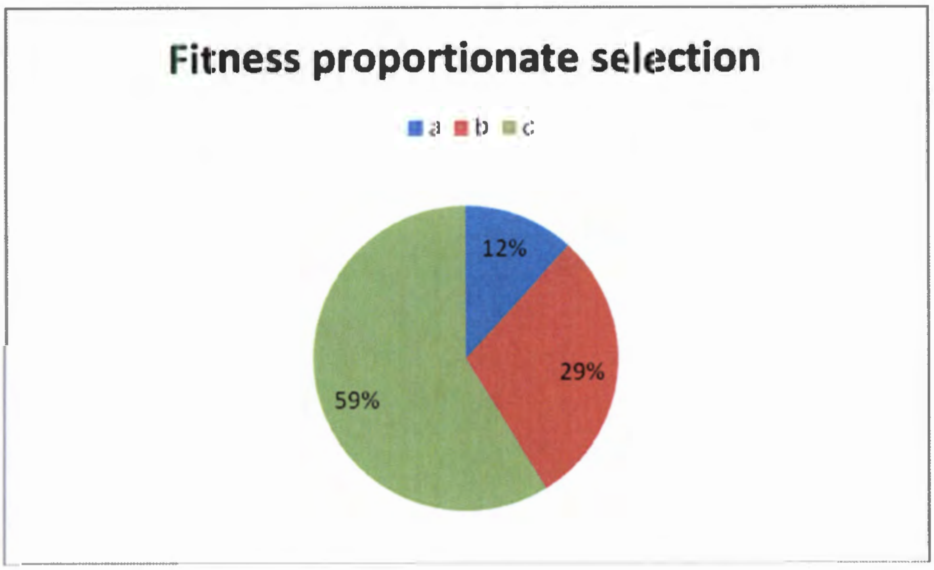

Figure 3.5: Individual fitness represented as a pie chart

Fitness proportionate selection provides zero bias, but does not guarantee minimum spread. The algorithm is given below:

Given:

Total fitness f_{total}

Individual set P of size *n*

Do:

Individual p = FITNESS_PROPORTIONATE(f_{total})

FITNESS_PROPORTIONATE(f_{total})

Returns individual P_i

R = random(0 - f_{total})

F = 0

For each i in n

F = F + f_i

If (F>R) return P_i

3.2.2.2.2 RANK

Ranked selection is similar to the fitness proportionate method, but whereas the proportion in the latter is dependent on the absolute fitness of individuals, the former depends on their ranked positions [24]. A degree of bias can be included by using the rank position of the individual raised by a specified polynomial factor [24]. As the individuals must be sorted first in order to rank them, this method is the most computationally expensive of the alternatives.

3.2.2.2.3 TOURNAMENT

Tournament selection involves the random selection of groups of individuals from the population with the fittest member declared the winner [29]. The degree of selection bias is proportional to the size of the group. Hence, the larger the size of the group, the greater the relative weight of fitter individuals is. As only the selected individuals need to be evaluated, this method is computationally efficient.

3.2.2.3 MUTATION

There are several mutation operations that can be performed on GP trees. The most common of which are briefly described below.

3.2.2.3.1 SUB-TREE SWAP

Entire sub-trees can be swapped within an individual as presented in the Figure Celow [30]. Note that the sub-trees with roots coloured red and green are swapped. The nodes that are to be swapped must be of the same type to ensure that the tree remains valid.

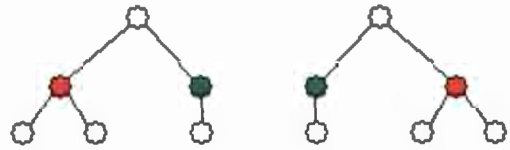

Figure 3.6: Sub-tree swap mutation

3.2.2.3.2 TREE-NODE SWAP

Similar to the sub-tree swap mutation, single nodes can be swapped in the individual as depicted in the Figure Celow [30]. Again, care must be taken to ensure that the nodes are of the same type to ensure consistency in the tree structure.

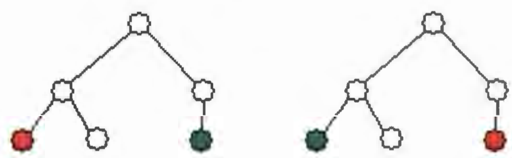

Figure 3.7: Tree-node swap mutation

3.2.2.3.3 SUB-TREE DESTRUCTIVE

Alternatively, an entire sub-tree in the individual can be destroyed and regenerated [30]. Again it must be ensured that the root node of the newly generated sub-tree is of the same type as that which was destroyed. Figure 3.8 below serves as an example of this type of mutation.

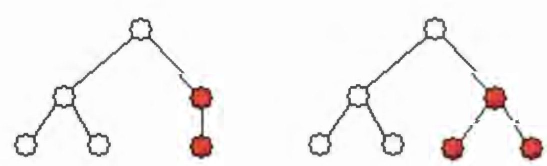

Figure 3.8: Sub-tree destructive mutation

3.2.2.3.4 NODE REPLACEMENT

Lastly, a single node within the individual can be replaced with a different node of the same type [30]. In this case, the sub-tree attached to the node to be replaced is retained as can be seen in figure 3.9.

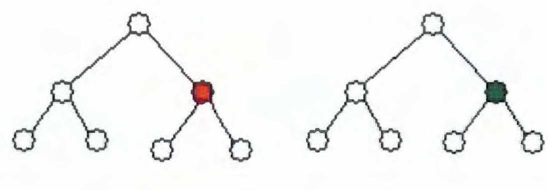

Figure 3.9: Node replacement mutation

3.2.2.4 CROSSOVER

A crossover point (function or terminal node) of the same type is selected randomly in each parent. These points and the entire attached sub-trees are swapped in the specified trees [30]. The crossover point selection usually favours the selection of functions rather terminals. This process is depicted in the Figure Celow.

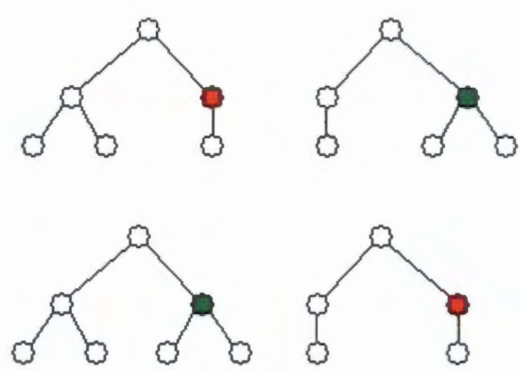

Figure 3.10: Sub-tree crossover

3.2.2.5 CROSSOVER vs. MUTATION

It is assumed that individuals transfer important information via sub-tree crossover, but mutation sometimes provides a combination of better fitness and smaller trees

[31]. Genetic programs are essentially computer programs and hence suffer from the "linkage problem" [31]: tree nodes in the GP are not independent of each other; instead they are linked through functional, control and data dependencies.

Functional dependency exists between children and their parents when data that is passed from one to the other affects the operation or result of either [31]. A change in the child could result in a change in the parent's operation and vice versa.

Control dependency manifests itself when a change in a sub-tree changes the flow of control thereby affecting whether or not a child is executed [31].

According to Luke [31] data (domain) dependency exists when functions or terminals take turns in manipulating the global environment in a specific order. A change in a function or terminal can affect the operations of other functions or terminals in the individual. Crossover between individuals can sever the relationship between sub-trees and. Consequently, this can modify or even introduce new global dependencies in the second individual on the addition of the sub-tree.

3.2.2.6 PROBLEMS

Despite the advantages of using GP's to solve non-trivial problems, there are two phenomenons that can manifest during the evolutionary process, namely overfitting and code bloat. This section introduces these concepts and describes methods to reduce their effects.

3.2.2.6.1 OVERFITTING

Overfitting occurs when a learned or evolved model fits the training data too well and does not generalize to out-of-sample data [30][32]. Three major approaches to solve the overfitting problem are described briefly below.

To order the search from simple to complex or search the solution space from general to specific models must be pruned and stopping criterion used. This biases the search in favour of simpler models. According to Occam's Razor [33], simpler models often have greater predictive power than more complex models and tend to result in less generalisation error. However, Domingos [34] regards the number of models, not the underlying complexity, as leading to overfitting.

Alternatively one can either limit the number of generations evaluated or reduce the population size [33] as this, too, reduces the number of models to be evaluated. Since fewer models are evaluated there is less chance of overfitting occurring.

Lastly, a validation data set can be used to test generalisation errors. This in turn can be used to terminate a search thereby limiting the number of models evaluated and reducing the possibility of overfitting [33].

3.2.2.6.2 CODE BLOAT

Code bloat is defined as "the uncontrolled growth in individuals over time, independent of equivalent changes in fitness" [31]. Once can either improve breeding, selection and generation to ensure that searches are more efficient so that fitter individuals can be found before bloat occurs or other techniques can be used to stave off bloat for as long as possible, effectively increasing the search period.

One of the most common causes of bloat is hitchhiking [31]. Many trees contain useless sub-trees known as introns. Introns are sub-trees that have no effect on the final result returned by the tree [31]. These sub-trees piggyback on important pieces of code as they are transferred from individual to individual resulting in the trees growing continuously [35]. One cannot simply make the system more selective about what code is transferred as this could still result in tree growth. As selectivity is increased, the value of important pieces of code increases, allowing these code blocks to replicate faster throughout the population. This, in turn, could allow introns to hitchhike and spread faster too.

Introns increase the number of crossover points in trees that are unlikely to modify the individual. Therefore, the more introns that are present the more likely they are to be used as crossover points [31]. This results in the child structure being identical in function and fitness to the parent and is a phenomenon known as neutral crossover.

The expected number of children n of an individual i that have fitness at least equalling that of i is given by:

$$n = s_i(1 - p_c((\frac{C_{ei}}{C_{ai}})d_i + \frac{(1 - C_{ei})}{C_{ai}}0)) = s_i(1 - p_c(\frac{C_{ei}}{C_{ai}})di) \tag{3.2}$$

where:

- s_i = expected number of times individual i is selected for crossover/reproduction
- p_c = probability that crossover will be breeding mechanism
- C_{ai} = number of crossover points in an individual (absolute complexity)
- C_{ei} = number of crossover points that might result in neutral crossover
- d_i = probability that individual i will be damaged by crossing over in non-neutral crossover points (in this case, probability is 0 as crossing over in neutral points causes no damage)

From the above formula it follows that initially crossover is equally likely to be constructive as it will be destructive [31]. As the evolutionary process progresses, however, individuals become fitter and finding better solutions becomes more difficult. The individuals with the highest survival rate are the ones with a low ratio of effective to absolute complexity [31]. Due to this, crossovers have little effect and high fitness values are maintained.

Similar to introns, inviable code refers to an area in an individual's tree where crossover can change neither the function nor fitness [31]. The more inviable code present, the more likely the associated sub-trees are to crossover. To overcome this problem and preserve the individual, the sub-trees removed during modification must be smaller than the inviable area [31]. This leads to a bias towards removing smaller sub-trees from the individual, leaving the larger sub-trees to grow indefinitely.

The three most commonly used ad-hoc techniques to rectify this problem are to [31]:

- Depth limit GROW/FULL algorithms to 2 to 6 ply for the initial generation
- Depth limit GROW algorithm to 4 ply for the generation of sub-trees for mutation points picked from non-terminals with 90% probability
- Depth limit sub-tree mutation and crossover to 17 ply

Without depth-limiting common function sets can cause tree sizes to approach infinity. However, depth-limiting does not guarantee that bloat will never occur as sub-tree mutation can result in bloating. Near-full trees that are at maximal depth will dominate the initial structures and sub-trees generated (for a given depth limit) and will bias the search space towards a small set of trees [31].

Tree size penalties have been put forward as an alternative solution to code bloat [31]. Linear or constant functions are commonly used as a function of size penalty, but an adaptive approach can be implemented that changes in response to tree growth metrics.

Yet another method involves code editing [31], where the introns in programs are physically removed. This can, however, lead to premature convergence in the evolutionary system [31]. Conversely, one can allow the inclusion of special nodes that increase the probability of crossover occurring at a specific position in trees.

3.3 CONCLUSION

This chapter presented a brief overview of GA's and GP. Various variations of evolutionary operators were introduced and their effects discussed. Finally, the problems regarding overfitting and bloat were discussed and generally accepted solutions proffered.

4. AGENT ARCHITECTURES

"...Agent-based computing is likely to be the next significant breakthrough in software development..." – **The Guardian, 12 March 1992**

This chapter provides a basic introduction to agent architectures and learning in agent-based systems. It discusses background theory on agents and the belief-desire-intention architecture and describes social learning and knowledge sharing primarily through the use of blackboard designs and a collective memory approach.

4.1 AGENTS

There is no general consensus on the definition of an agent. According to Smith et al. [42] an agent is a "persistent software entity dedicated to a specific purpose", whereas Selker [43] defines agents as "computer programs that simulate a human relationship by doing something that another person could do for you". Yet another definition for agents is given by Riecken [43] as "integrated reasoning processes". Perhaps the best definition is provided by Franklin and Graesser [42]: "An autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future." Their extended explanation goes on to describe the notion of situatedness within the environment. Here, situatedness refers to the ability of the agent to sense its surrounding and act upon it to make changes.

According to Wooldridge and Jennings [42], autonomous agents have the following characteristics:

- **AUTONOMY**

  Agents operate without the intervention of humans and have control over their actions and states.

- **SOCIAL ABILITY**

  Agents interact with each other via a communication language.

- **REACTIVITY**

  Agents sense their environment and respond to changes.

- **PRO-ACTIVENESS**

  Agents are able to exhibit goal-directed behaviour by taking the initiative.

4.2 AGENT ARCHITECTURES

Agent architectures serve as a bridge between agent theories and the implementation of the latter. These architectures describe the internal information structures, processing mechanisms and specification of information flow in agents. One can distinguish between three types of agent architectures namely: reactive, deliberative and hybrid [44].

Reactive architectures do not include an internal representation of the world and do not require any kind of complex reasoning mechanisms [44]. Instead, processing is performed in real-time using behaviour rules, presented as (situation → action), and limited sensory information as input.

Unlike the reactive architecture, deliberative architectures incorporate an internal representation of the agents' world and the agents' explicit mental states [44]. Such agents make decisions through logic reasoning. Several problems plague this architecture however, namely [44]:

- It is difficult to translate information from the agents' sensors to their internal structure.
- It is difficult to model the agents' internal information regarding the environment and develop effect reasoning mechanisms to reason on this information.

Lastly, hybrid architectures attempt to combine the two previous architectures by using a layered approach where the reactive layer has higher priority than the deliberative layer in order to quickly answer the changing events in the environment [44].

4.2.1 BELIEF-DESIRE-INTENTION

The Belief-Desire-Intention (BDI) architecture is probably the best example of a deliberative agent architecture. It attempts to provide an approximation of practical reasoning. Practical reasoning consists of two distinct activities [45]. The first is deliberation (what one wishes to achieve) and the second is reasoning (how one wishes to achieve these ends). This section introduces the concepts of BDI agents and the BDI architecture.

4.2.1.1 BDI AGENTS

In the BDI approach, agents incorporate the following high-level specifications [44]:

- Structures that mimic the mental states of belief, desire and intention
- An internal processing system that interprets and updates structures, and determines the actions for execution

Furthermore, according to Rao and Georgeff [46], BDI agents may exhibit three particular kinds of behaviour: strong realism, realism and weak realism.

When an agent exhibits strong realism, the agent's intentions are a subset of its desires that are in turn influenced by the agent's beliefs. In other words, if the agent does not believe in something, it will never be intended.

Realism behaviour states that an agent's beliefs are a subset of its desires that are influenced by its intentions. Worded otherwise, if an agent believes in something it is intended.

Finally, an agent behaving under weak realism does not desire something if its negation is believed. If its negation is desired it does not intend something. If its negation is believed, it does not intend something

BDI agents contain an internal representation of their mental states namely [44]: beliefs, desires and intentions. These are briefly discussed below.

- **BELIEFS**

  Beliefs are those concepts that agents believe to be true. They describe the agents' perspective of the world in which they operate and serve as the basis of the information held by the agents.

- **DESIRES**

  Desires refer to the concepts that agents believe to be false *but* that agents would like to see become true in future. Agents need not know how this might come to pass and, indeed, it might be inconsistent with the agents' beliefs and intentions at any given time. In other words, desires can be likened to an agent's motivation.

- **INTENTIONS**

  Intentions are analogous to the set of actions or tasks that an agent is committed to in order to make a concept true. It is the result of deliberation processes and must be internally consistent within the agent.

4.2.1.2 BDI ARCHITECTURE

The general abstract BDI architecture consists of beliefs, a belief revision function, an option generation function, current options, a filter function, current intentions and an action selection function. The interoperation of these components is described in the figure below.

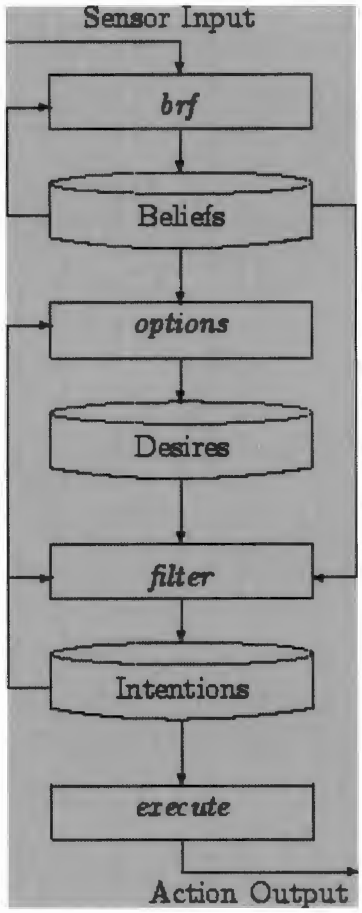

Figure 4.1: Interoperation of components in BDI architecture [44]

- **BELIEFS**

  As mentioned previously, beliefs refer to the information that agents have regarding the surrounding environment.

- **BELIEF REVISION FUNCTION**

  The belief revision function produces a new set of beliefs based on the agent's current sensor perceptions and beliefs.

- **OPTION GENERATION FUNCTION**

  The option generation function defines an abstract hierarchical plan of action that is, at first, coarse-grained. The plan is then subdivided and reconsidered until only a set of atomic actions remains. Options generated in this way must be consistent with the agent's current beliefs and intentions. In addition, the agent must be clever enough to perceive environmental changes favourable to achieving goal and those that are likely to lead to impossible goals.

- **CURRENT OPTIONS**

  The current options refer to all possible actions for the agent.

- **FILTER FUNCTION**

  The filter function handles the deliberation processes for the agents and has two primary roles. It must discard intentions that are either not achievable or of which the execution cost is excessively high compared to the expected gain. It must retain those intentions that are expected to benefit the overall performance of the agent.

- **CURRENT INTENTIONS**

  The current intentions are those states that the agents are committed to.

- **ACTION SELECTION FUNCTION**

  This function determines the action the agent should execute based on the available list of intentions.

4.3 BLACKBOARD THEORY

Blackboard theory is based on a brainstorming analogy. People with expert knowledge about a given problem work together as a team to brainstorm a possible solution to the problem. A blackboard is used as a workspace in which the agents work cooperatively towards solving the problem. During this process, all experts continuously monitor the blackboard and contribute to the solution as appropriate.

Blackboards were originally designed as a means of dealing with ill-defined, complex applications [47]. The first blackboard system was used in the Hearsay-II speech understanding system [48]. The basic features thereof are still found in implementations today, but numerous advances and enhancements have been made as blackboards have gained recognition in a wide variety of applications.

4.3.1 FRAMEWORK

Blackboard systems consist of three components:

- Knowledge Sources
- Blackboard
- Control Component

The inter-relationship between these components is presented in the Figure Celow.

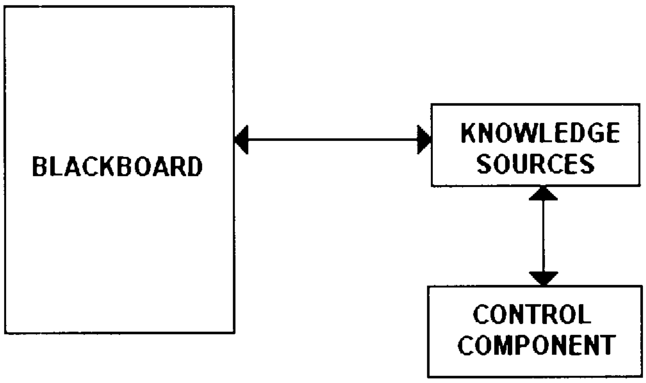

Figure 4.2: Blackboard system architecture

4.3.1.1 KNOWLEDGE SOURCES

Each knowledge source (KS) is separate and independent from all other knowledge sources. Its purpose is to contribute information to the blackboard that will ultimately lead to the solution of the problem.

KSs are generally represented as procedures, sets of rules or logic assertions [49]. They modify only the blackboard and control data structures and only the KSs are allowed to modify the blackboard. Each KS knows the conditions under which it can contribute to the solution and this knowledge is known as the triggering condition. There are associated preconditions that indicate the condition on the blackboard that must exist before the body of knowledge can be activated.

4.3.1.2 BLACKBOARD

The blackboard is a structure that is globally available to all KSs. It holds computation and solution-state data that are needed by and produced by the KSs (KSs therefore use the blackboard data to interact with each other directly) and consists of objects from the solution space including raw input data, partial, alternative and final solutions [49]. These objects are organised hierarchically. Generally, information on one level serves as input to a set of KSs that in turn place new information on the same or other levels.

4.3.1.3 CONTROL COMPONENT

The control component consists of a set of control modules that constantly monitor changes on the blackboard and decide on the actions to perform [49]. Various kinds of information are made globally available to these modules as it is used to determine the next focus of attention. The focus of attention determines the next object to be processed and can be either a KS or a blackboard object or a combination thereof. This flow of processing results in the solution being built incrementally. Any type of reasoning data can be applied at each stage and therefore the sequence of KS invocations is dynamic, rather than fixed.

4.3.2 APPROACH

The general approach to solving a problem using blackboards is to divide the problem into loosely coupled subtasks. Each subtask corresponds to an area of specialisation in the task. The designer defines the solution space and the knowledge required to find a solution. The solution space is then divided into multiple levels of intermediate solutions and the knowledge to solve the problem is divided into specialised knowledge areas to perform subtasks. Information in the levels is globally accessible on the blackboard and as such provides a medium of communication between the KSs.

The procedure followed by the blackboard approach is as follows [48]:

- KS makes a change to the blackboard and record thereof is kept in a global structure holding control information
- Each KS indicates the contribution it can make to the solution
- The control module selects the focus of attention based on the above
- The control module prepares for execution:
    - If it is a KS then a blackboard object is the context of invocation (knowledge-scheduling approach)
    - If it is a blackboard object then the KS is chosen which will process that object (event-scheduling approach)
    - If it is both a KS and blackboard object, the KS is ready for execution and is executed together with the context

Usually a KS indicates when the problem-solving should be terminated either due to the generation of an acceptable solution or the system cannot continue for lack of knowledge or data.

4.4 KNOWLEDGE SHARING

Agent behaviour and coordination techniques are often predefined and fixed by the designer. Agents with fixed and limited knowledge and behaviours, however, will be unsuited to a dynamic, changing environment. It is important that these agents learn to cope with new and changing situations to improve their effectiveness.

It is not necessary that agents only learn from their own experiences; they can instead, observe other agents and learn from their situations and behaviours. There are two approaches to learning between agents [50]:

- Agents can consult with or request advice from more expert agents
- Agents can explicitly share information and learn from this

This cooperative learning approach results in more knowledge and information resources and can greatly increase agent efficiency compared to that obtained from individual learning. Coordination and integration of knowledge is problematic however. The following questions must be answered before a cooperative system can be implemented [50]:

- How is newly acquired information and knowledge evaluated?
- How are the behaviour and intelligence levels of other agents assessed?
- How does one combine the knowledge of one agent with that of others?

4.4.1 COLLECTIVE MEMORY

Collective memory is a resource that is made available to agents through experience and can be used to improve performance when interacting to solve collaborative problems [51]. It is generally stored in either centralised (blackboards) or distributed memory of agents.

Given an environment with multiple adaptive agents of differing abilities and limited knowledge of each other, the activity cycle of the system is given by [51]:

- Give the community of agents a set of goals
- Allocate these goals among agents
- Repeat until all agents have achieved their goals
  - Active agents use collective memory to create or adapt plans
  - Agents with plans attempt an operator
  - Make on-line adjustments to collective memory
- Make off-line adjustments to collective memory

The use of collective memory can impact the performance of agents in two ways [51]:

- Agents can use past successful problem solving experience to guide them in similar situations rather than planning from scratch to find solution.
- When an agent has to plan from scratch its experience in interacting with the domain allows it to construct a plan that is more likely to succeed as it has a better idea of its and other related agents' capabilities.

It is impossible to guarantee that agents will find efficient solutions without using prior experience. Agents have common top-level goals, but they all have their own idea of how to best proceed towards those goals. Instead, by employing collective memory, agents share a common viewpoint on how best to approach the problem and work towards a solution. In other words, agents remember successful patters of cooperation from the past and can use this as a basis for proceeding in future. Even this, however, cannot guarantee the derivation of efficient solutions. In new situation where agents do not share a common view, it is possible for agents to refuse to help when they should or agree to help when they should not.

4.5 CONCLUSION

An overview of agent-based architectures was presented in this chapter. The concepts of agents and agent architectures were introduced and discussed as well communication methods via blackboards and knowledge sharing through the use of collective memory. The problems with these approaches were examined, but these were mitigated as the advantages and simplicity of these techniques outweighs the disadvantages. Social learning was presented in the form of knowledge sharing through which agents share local knowledge with other agents in order to allow agents to adapt to dynamic environments. The problems with this learning approach were also discussed and it was decided that a general solution is not applicable and must be treated on a case-by-case basis.

5. LITERATURE REVIEW

This chapter presents an analysis and critical discussion of related work published in the fields of:

- Genetic programming for trading rules
- Genetic algorithms for portfolio optimisation
- Agent-based systems for trading

5.1 GENETIC PROGRAMMING FOR TRADING RULES

GP solutions have proven to be adept at computing approximate solutions for problems for which no general efficient solution is known. One such problem is forecasting in stock markets. Here, forecasting is meant to be the process of deriving trading rules that indicate when to buy a particular share and when to sell it. The universe of all possible trading rules and the list of factors that can potentially affect the results thereof are simply too vast for the problem to be solved effectively and efficiently. It is this type of problem to which GA's are geared.

In the past few years, much research has been conducted on the field of the application of GP to the production of effective and successful trading rules. One such approach was taken by Potvin, Soriano et al. [28]. They proposed a GP system that generates trading rules encoded as computer programs. These rules were Boolean in nature and returned either a buy (true) or sell (false) signal.

The trees generated by their GP system incorporated a wide-range of function nodes including [28]:

- Arithmetic operators: +, -, ÷, ×
- Boolean operators: and, or, not
- Relational operators: <, >
- Boolean functions: if-then-else

- Real functions:
  - Norm(r1, r2) = absolute value of the difference between r1 and r2
  - Avg(s, n) = average of price or volume over the past n days
  - Max(s, n) = maximum value of price or volume over the past n days
  - Min(s, n) = minimum value of price or volume over the past n days
  - Lag(s, n) = price or volume lagged by n days
  - Volatility(n) = variance in daily returns over past n days
  - RSI(n) = RSI
  - ROC(n) = ROC

In addition, the following terminal nodes were used:

Real constants: number in interval [0, 250]
Boolean constants: true, false
Other constants: price, volume

Potvin, Soriano et al. [28] conducted a study on Canadian companies selected from the TSE300 index for the period June 1992 to June 2000 spanning 14 different sectors, with one company being selected from each sector. Both long and short training periods were tested and the results compared. It was shown that the system achieved impressive performance on the training data for the short training period (131.17%) , whereas training performance for long training periods was less impressive (22.93%). The results were however very similar and equally disappointing for the test period using both short (-4.85%) and long training (-3.59%) periods. In addition, the rules generated by the GP were often intricate and difficult to comprehend.

The problems encountered in their work could be attributed to the way in which the GP trees were structured. The combination of real and Boolean valued nodes in the tree could have resulted in the generation of complex trees. It could also be attributed to the fact that different types of values could be compared in the tree, for example the following node layout is entirely legal:

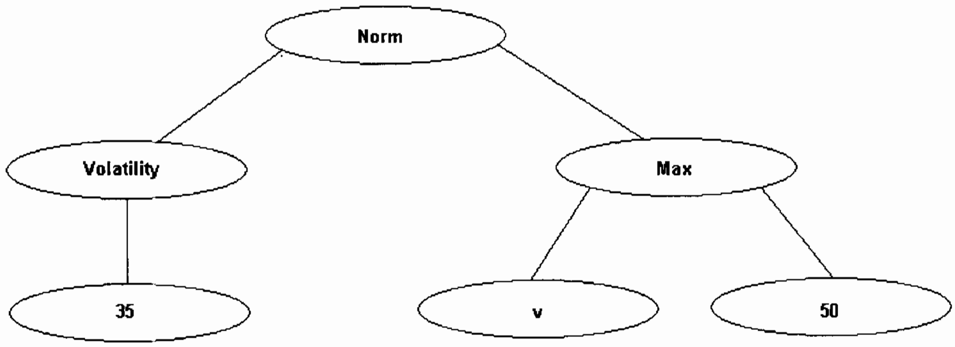

Figure 5.1: Sample tree generated by GP

The rule in the figure above can be interpreted as: generate a buy signal if the absolute difference between the volatility in price returns for the last 35 days and the maximum value of the volume in the last 50 days is positive. Even a seasoned investor might have trouble making sense of such a rule.

A slightly different approach was taken by Seshadri [30]. Instead of mixing real and Boolean node types, only the following node types were implemented:

- Boolean operators: and, or, not
- Comparison operators: <, >
- Technical indicators: price, moving averages

Although this reduces the expressive power of the GP trees, it results in more understandable trading rules and through judicious selection of technical indicators more powerful rules *could* be developed [30]. Whereas Potvin, Soriano et al. [28] used daily stock data, Seshadri [30] maintained that using monthly instead would result in fewer trades and therefore smaller transaction costs. Seshadri [30] also proposed the use of data from the S&P500 index. As portfolios are often evaluated by their ability to outperform the return of this index, it would be simple to compare it to the profits generated by the rules.

Three experiments were performed:

- The effect of a complexity-penalising factor
- The effect of different evaluation functions

- The effect of pairing buy and sell rules

In the first test, the mean out-of-sample performance for the rules without a penalising factor was 2141. The smallest tree had a depth of 11 and size of 41 while the largest tree was comprised of almost 1000 nodes and a depth of approximately 90. In addition, only 2 of the rules generated managed to outperform the buy-and-hold returns. The mean out-of-sample performance for rules with the penalising factor was much more compelling at 2812. In fact, the generated rules outperformed the buy-and-hold returns at the 95% significance level. Similar results were obtained for the final two tests, with the paired buy and sell rules outperforming the buy-and-hold returns at the 99% significance level [30].

Yan [36] proposed an alternative GP approach whereby the trees had the following structure:

CONFIDENCE VALUE = BUY TREE – SELL TREE

$$CS = (BI1*W1+BI2*W2+\ldots+Bin*Wn) - (SI1*W1+SI2*W2+\ldots+Sin*Wn) \qquad (5.1)$$

where:

- BI_i = the *i*-th buy indicator
- W_i = the *i*-th weight
- SI_i = the *i*-th sell indicator

In tests conducted, the in-sample performance of the system was 7.2% compared to the benchmark return of 6.67% for the same period. Similar results were obtained for out-of-sample performance. The system return was 5.622% versus the benchmark of 4.79% [4].

The lack of an accepted base-line performance was a common thread that ran through all the previous examples except that of Seshadri [30]. His approach included a comparison of the performance of the evolved rules and a buy-and-hold strategy. Yan

[36] included some form of benchmark performance, but did not elaborate on what it was and how it was obtained.

5.2 GENETIC ALGORITHMS FOR PORTFOLIO OPTIMISATION

Portfolio optimisation, like forecasting in stock markets, is another problem that is difficult to solve efficiently and optimally. Several evolutionary approaches have been suggested in the past alongside the more traditional quadratic programming techniques.

One such approach was proposed by Shapcott [22]. He used a modified GA to represent the composition of the portfolios and the weighting of the stocks that comprised it. The traditional crossover operator would be invalid when used with this representation as there is no well-defined behaviour for exchanging stocks (and weightings) between individuals. Instead, a modified crossover operator is introduced in the form of random assortment recombination [22]. This method generates child portfolios by selecting stocks from the parents to incorporate and gives preference to stocks that are present in both parents.

A related but slightly different approach was taken by Chang, Mead et al. [37]. The individuals in their evolutionary system were comprised of two distinct parts, a set Q of K distinct assets and K real numbers si ($0 \leq si \leq 1$) $i \in Q$. Given Q a fraction of the total portfolio is already accounted for and si can be interpreted as the share of the free portfolio portion associated with asset i. Similar to the RAR operator used by Shapcott [22], Chang, Mead et al. [37] use a modified uniform crossover operator. According to their strategy, an asset is definitely present in the child if it is present in both parents. If an asset is not present in both parents, it has a 50% probability of being present in the child. The children portfolios are also subjected to mutation. The weight of a randomly selected asset is either increased or decreased by 10% with equal probability.

A robust testing strategy was proposed with data from the Hang Seng, DAX100, FTSE100, S&P100 and Nikkei225 indices for the period March 1992 to September

1997 [37]. Tests were run to determine how well the unconstrained and cardinality constrained efficient frontiers could be calculated using a GA, tablu search and simulated annealing.

The GA heuristic was best able to approximate the UEF with an average mean error of 0.0114% [37]. Similarly, the GA provided the best approximation for the CCEF, however the differences were less marked than in the case of the UEF.

Yet another evolutionary method is proposed Korczak and Lipinski [38]. They encode portfolios as a set N of n real numbers where N is the number of stocks in the portfolio and N is a set of asset weights. To evaluate the portfolios thus generated, various objective functions were proposed:

$$
\begin{aligned}
F_1(x) &= \frac{1}{1+\varepsilon_1 \times SVar(R_x)} \\
F_2(x) &= \frac{1}{1+\varepsilon_1 \times SVar(R_x)+\varepsilon_2 \times \left|\beta_x - \beta_{x_0}\right|} \\
F_3(x) &= \frac{1}{1+\varepsilon_1 \times Cov(R_x, R_i)+\varepsilon_2 \times \left|\beta_x - \beta_{x_0}\right|} \\
F_4(x) &= \frac{1}{1+\varepsilon_1 \times SVar(R_x)+\varepsilon_2 \times Cov(R_x, R_i)+\varepsilon_3 \times \left|\beta_x - \beta_{x_0}\right|}
\end{aligned}
\tag{5.2}
$$

where:

- $F_k(x)$ = objective function k
- x_0 = initial portfolio
- R_i = market return
- $\beta_x = \beta$ coefficient of portfolio x
- $\beta_{x0} = \beta$ coefficient of initial portfolio x_0

Two tests were performed on the CAC40 index; the first with 10 randomly chosen stocks the other with all 40 stocks comprising the index [38]. All experiments used a test period of 20 or 60 days. In most of the tests, the evolutionary algorithm outperformed the buy-and-hold approach for the same time period, while only a few of the tests resulted in better performance than the index benchmark.

The results from these approaches suggest that evolutionary optimisation of portfolios can result in superior performance under satisfactory conditions and finely tuned control parameters.

5.3 AGENT-BASED SYSTEMS FOR TRADING

Kendall and Su [52] employed an agent-based approach to stock trading whereby artificial stock traders coevolved by means of individual and social learning and learned to trade stock profitably. It was only tested on single stocks, however, and therefore did not include a portfolio management component.

In their system, agents used multi-layer feedforward neural networks (NN) to represent trading models. The nodes forming the NN were randomly selected from a set of indicators that included: moving averages, relative strength, rate of change and stochastics. During the individual learning phase agents evolve their NNs by means of GAs.

In this system individual learning occurs during every 125-day trading period. At the start of the period, each trader selects a set of indicators from which to generate their predictive models. On the first day of this trading period, each agent selects a model with which to trade for the following five days after which the models are evolved by GAs.

Social learning occurs at the end of every individual learning period. Each agent assesses its own performance and then compares this value to that of the other agents. The traders who have performed consistently well compared to the rest can publish their predictive models to a central model pool that is visible to all other agents. Those who performed poorly can request predictive models from the pool with which to trade in the next individual learning period.

In tests performed on five different stocks from different market sectors, two different trading strategies were identified:

- **AGGRESSIVE**
  These traders followed the trend of the stock price closely and accumulated wealth in frequent trading. This type of strategy works well in a bull market, but during a bear market the key is the adaptability of the agents.

- **CONSERVATIVE**
  These traders are more cautious about trading in the market. They trade less frequently and usually have lower growth lines during a bull market. However, during a bear market, these traders managed to adapt to the new environment faster and transferred their assets from the market to the bank.

Additionally, it was shown that between 22% and 36% (across all five stocks) of the traders were able to evolve profitable trading strategies that beat the market. In early work pioneered by Kendall and Su [52], 80% of the agents managed to beat the market, trading with only one stock. While the results for the above tests are commendable, their research did little to further knowledge regarding the evolution of trading strategies for multiple stocks – indeed, their research focused solely on predicting stock movement for a single stock. As stocks present very different behavioural patterns, their solution might perform poorly in a multi-stock environment.

6. DESIGN AND IMPLEMENTATION

"...A complex system that works is invariably found to have evolved from a simple system that worked..." – **John Gall**

This chapter presents the design and implementation decisions taken for the software component during the course of this thesis. It gives a broad overview of the design followed by more in-depth analysis of the implementation details.

6.1 DESIGN OVERVIEW

The key idea for the software component was to simulate a stock market and intelligent traders entirely in software without the need for human intervention. To this effect the following entities were identified for inclusion in this simulated world:

- A data source
- Traders
- Trading Strategies
- A semblance of retention of "knowledge"

The following actions were extracted from these entities as they would occur in the real world:

- Develop trading strategies
- Trade stocks
- Compile a portfolio of stocks
- Update "knowledge"

The following diagram represents a mock-up of the system design:

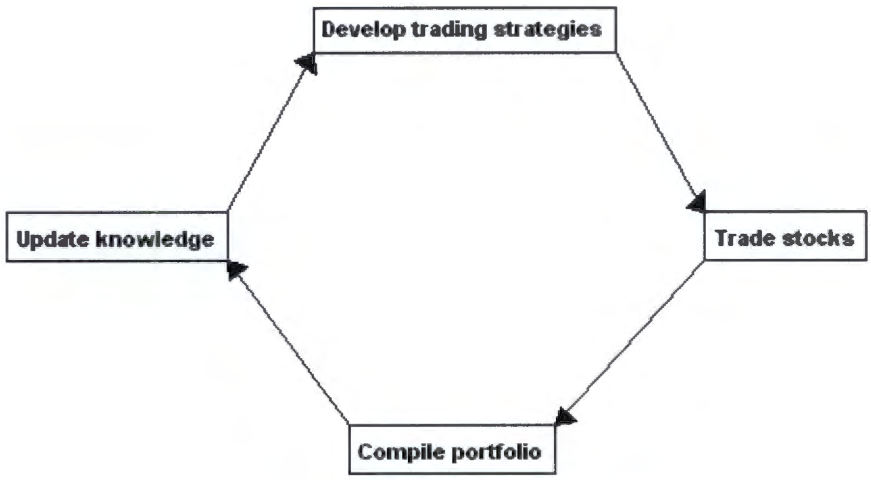

Figure 6.1: Mock-up of system design

The mock-up was then used to draw up a list of entities that would appear in the software solution:

MOCK-UP COMPONENT	FINAL COMPONENT
Data Source	Data Source
Traders	Traders
Trading Strategies	Rules
Source of "knowledge"	Central Model Pool

Table 6.1: List of components in final solution

Accordingly, the data source, central model pool and traders form the key components of the proposed trading system. Figure 6.2 provides a conceptual high-level overview of the interaction between these components. This section outlines the resulting system designs.

Previous attempts [28][30][36] at evolutionary trading systems have used in-sample data to evolve the GP trading rules and out-of-sample data to test its effectiveness. The result is a static rule that was "trained" from past data and is expected to realise significant returns for all future time periods. The major shortcoming with this approach is that it does not take into account the dynamicity of the stock market environment. A rule that was successful 6 months ago might not achieve the same success in future time periods. To adapt to a constantly changing environment the trading rules must change constantly too. The learning framework design therefore revolves around a form of knowledge feed-back loop where trading rules that performed well in the past are kept in a knowledge base and updated in future if they are expected to perform well.

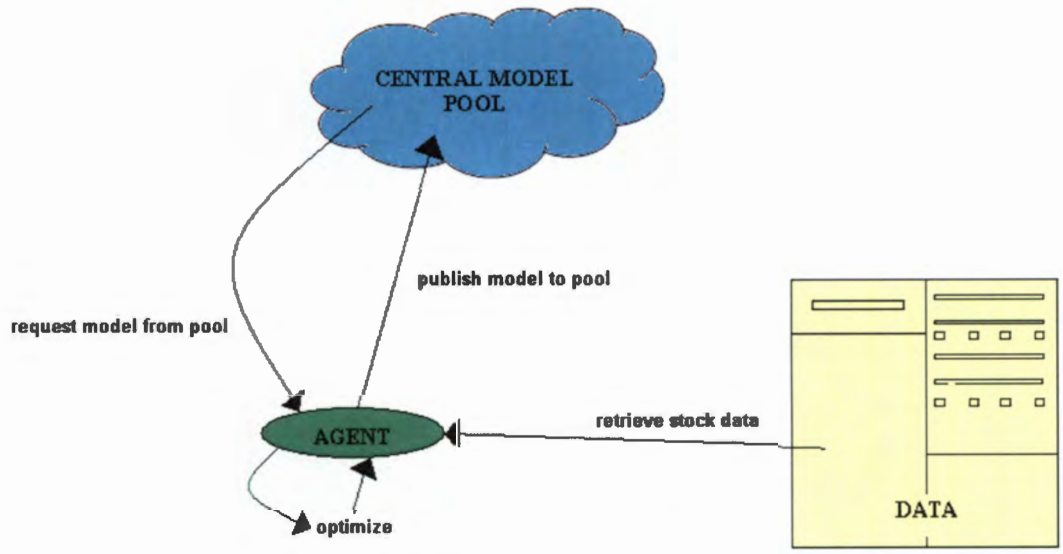

Figure 6.2: High-level overview of the trading system

The system is logically separated into three phases of learning: conditioning, individual and social.

During the conditioning and individual phases each trader uses its unique rule sets to make predictions based on historical stock data. The only difference between these two phases is that traders trade stocks and maintain a stock portfolio during the individual phase as well. Said rule sets are then evolved based on the strength of the traders' predictions and the cumulative performance of each trader is updated accordingly.

The condition phase was included to prevent the traders from trading with completely random rules initially. Therefore a conditioning period of two years was granted to each trading agent to evolve initial rule sets before trading.

The steps in the individual phase are summarised below in the flowchart in figure 6.3.

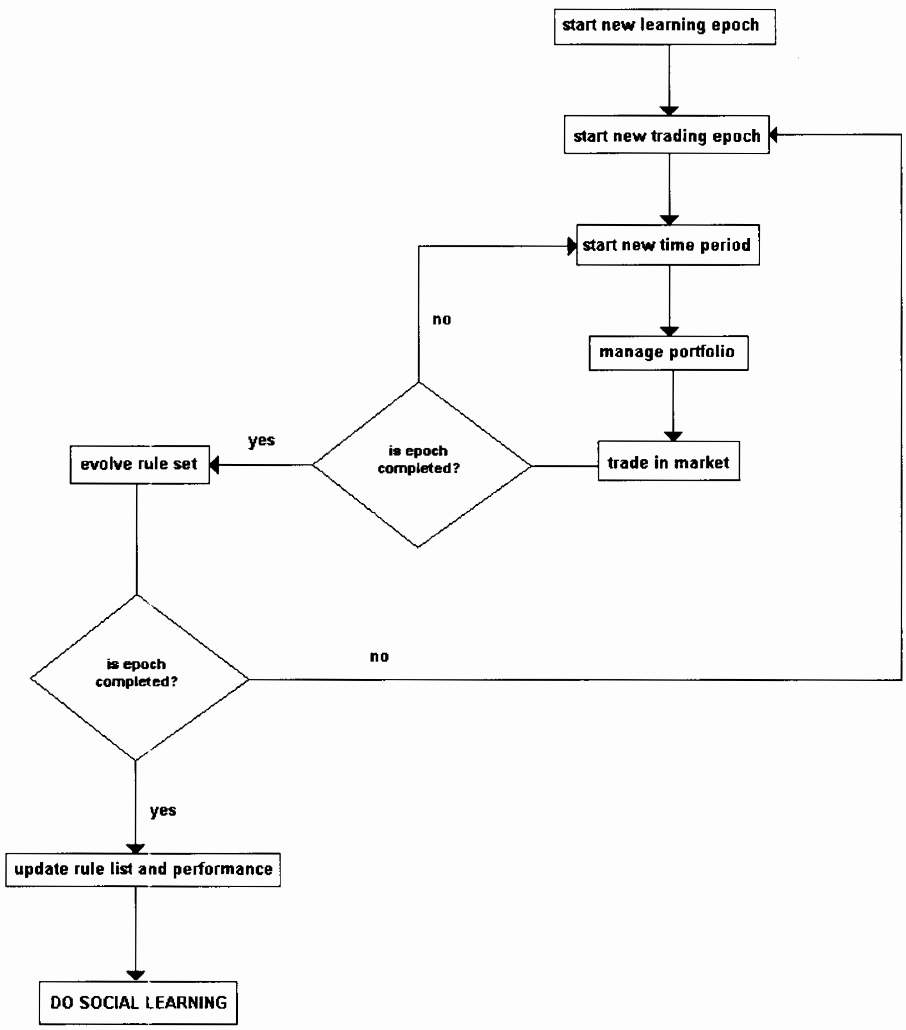

Figure 6.3: Flow-control diagram for the individual learning phase

In figure 6.4 below, a flow chart is given of the social learning phase. During this phase, the traders compare their respective performances to that of their peers for the recently completed learning phase. Based on the strength of their individual gains (or losses) on their initial portfolio values for the time period, the best-performing traders publish their rule sets to the model pool, while poor performance results in the corresponding trader subscribing to the pool for better rules with which to trade and evolve in the next period.

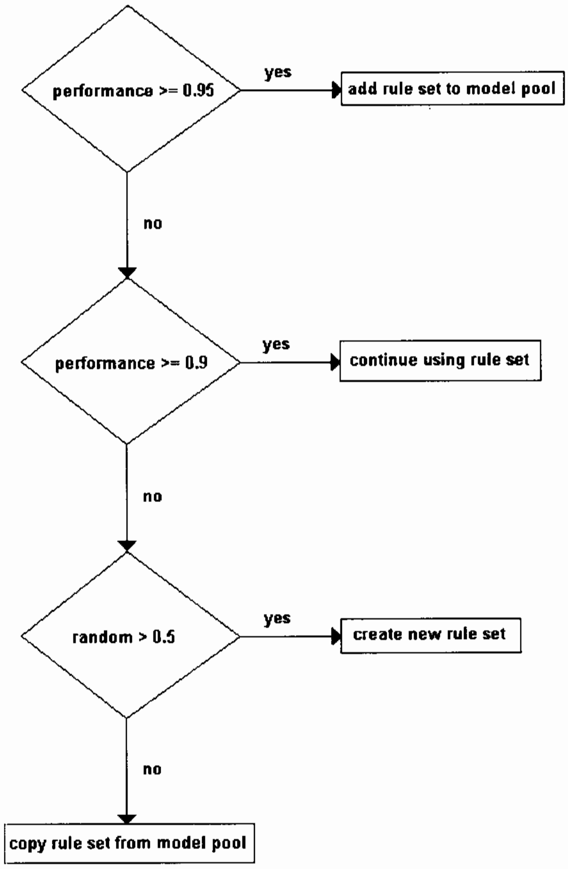

Figure 6.4: Flow-control diagram for the social learning phase

The threshold values for determining when rules are submitted and requested were derived from the research of [52] and extensive system testing.

6.1.1 TRADERS

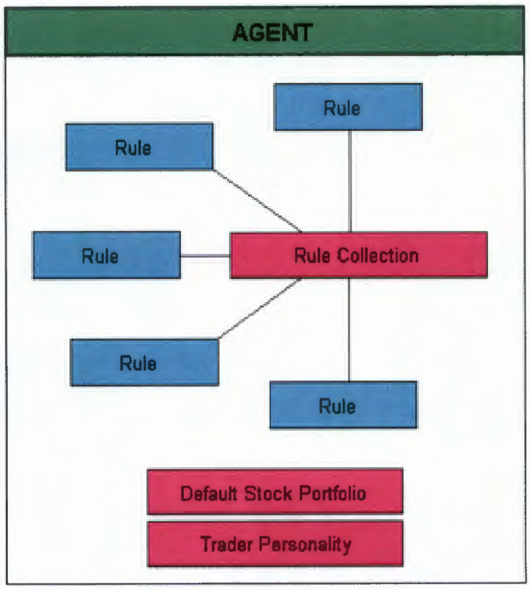

Figure 6.5: Overview of the trader characteristics

Each trader maintains a rule collection, which is evolved during each learning phase, and an active rule from the collection which is used to trade stocks on the market. In addition, a portfolio of stocks is optimized at the completion of a round of trading. These details can be seen in figure 6.5 above.

To simulate an expert knowledge system while avoiding the rigidity that accompanies the resultant models, each trader is assigned a trader personality. The trader personalities, in turn, determine the composition of the models. The personalities implemented are [39]:

- Dreman
- Graham
- Lynch
- O'Neil

The primary reason for the inclusion of well-known traders like Dreman and O'Neil is that each takes a different approach to trading on the stock market. It was hoped that given sufficient time, the different traders would modify rules from other traders and publish the modified rule to the central model pool, resulting in a combination of ideas from the different trader personalities. The process if presented in figure 6.6 below.

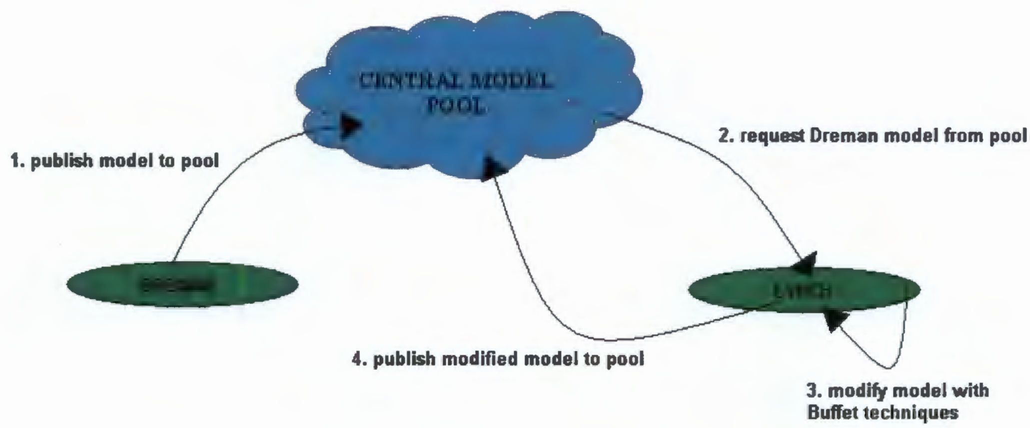

Figure 6.6: Sharing of knowledge between Dreman and Lynch traders

This exchange of expert knowledge can be seen as a form of emergence as each trader only has limited knowledge of the doings of other traders. The only indication of the actions of the other traders is the models that have already been submitted to the central model pool. One can regard the models generated by each trader as a being simple entities as they only incorporate the domain knowledge specific to the trader. By combining their respective knowledge, however, more complex trading strategies can be derived than could have been generated from the individual knowledge. The process is depicted in the figure above.

It was also hoped that the resulting shared knowledge would lead to more robust trading rules and therefore better trading performance.

6.1.2 RULES

Each trading rule generated by the traders consists of separate buy and sell rules, each corresponding to a genetic program, depicted in figure 6.7 below. Each program is interpreted bottom-up and returns a Boolean value representing either a buy or a sell signal depending on which tree was interpreted and whether or not the trader should be in the market.

SELL SIGNAL	IN MARKET	OUT OF MARKET
True	***SELL***	HOLD
False	HOLD	HOLD

Table 6.2: Sell trading rule logic table

BUY SIGNAL	IN MARKET	OUT OF MARKET
True	HOLD	***BUY***
False	HOLD	*HOLD*

Table 6.3: Buy trading rule logic table

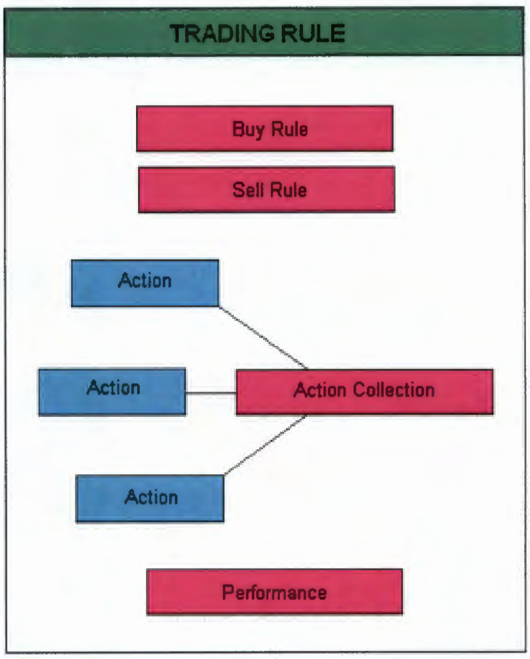

Figure 6.7: Overview of the trading rule characteristics

The genetic programs are binary trees that consist of function and terminal nodes. Function nodes are equivalent to the subset of the most common Boolean operators (AND, NOT, OR, XOR) and the IF-THEN-ELSE construct. Terminal nodes correspond to a subset of simple technical indicators including subsets of rules used by Dreman, Graham et al. [39]. A sample tree generated using these nodes is presented in figure 6.8 below. It can be interpreted as (assuming it is a buy tree): buy shares in a stock if either the MA or RSI indicate a buy signal or the rate of change does not indicate a buy signal.

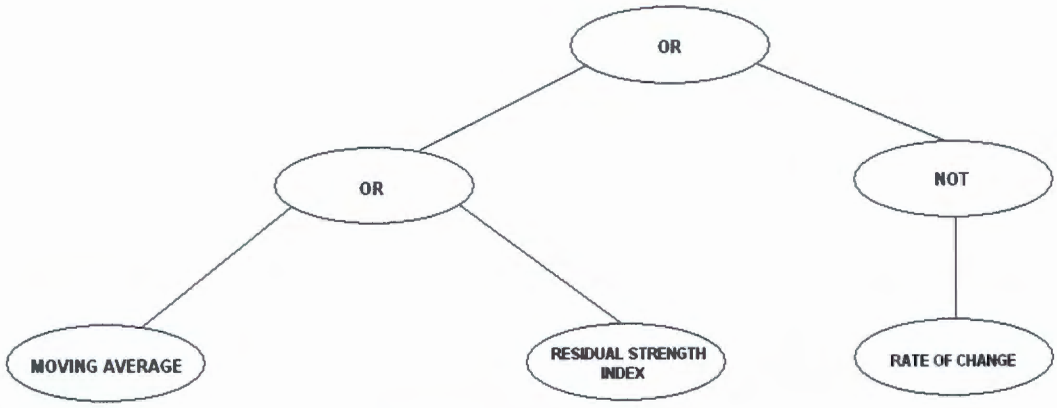

Figure 6.8: Sample GP tree

Trading rules are generated randomly during the initialisation procedure of each trader. The trees are created using either the FULL or GROW algorithm (50-50 chance).

6.1.3 PORTFOLIO

Each portfolio maintained by the trading agents is simply a collection of stocks that are held at a specific period in time and their associated quantities. The genetic algorithm encoding thereof is described in the Figure Celow.

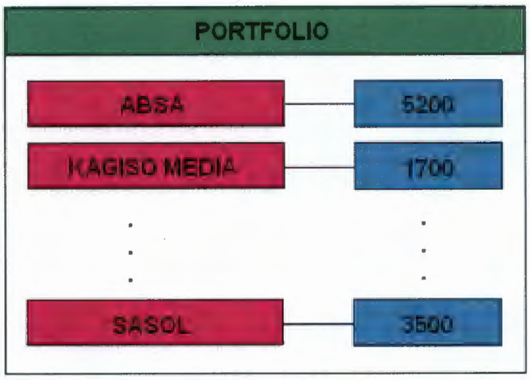

Figure 6.9: Sample portfolio

Each stock is analogous to a chromosome and the genes are represented by the associated quantity of shares.

6.1.4 MODEL POOL

The model pool is a globally accessible structure, available to all traders, that contains an ordered list of trading rules and their associated cumulative performances. Trading rule performance in this context refers to the cumulative normalized returns over all lapsed time periods. Its main purpose is to provide the traders with a centralized blackboard through which they can share successful trading strategies in a social learning environment. At the end of every individual learning period, the traders that achieved the best performance relative to that of their peers publish their findings to the model pool, thus making their successful strategies available for other traders to copy and evolve.

6.2 IMPLEMENTATION OVERVIEW

This section provides an in-depth discussion of the characteristics and implementation details of each of the aforementioned components. An overview is also given of the technical indicators that have been included in the final system.

6.2.1 RULE EVOLUTION

Rules are evolved by using the traditional mutation, crossover and reproduction operators. The strongest half of the current population is used to generate a new batch of offspring that then replace the weakest half of the population. Offspring are generated by selecting one of the three operators with their associated probabilities. The rules to be replaced are simply overwritten with the new children.

At every time step the performance levels of the current rules are updated. Each rule is evaluated using the previous time step's stock data and compared with the actual data from the current time step. The numbers of correct and incorrect decisions for each rule are accumulated and a trading strategy can be formulated for the next time step.

Fitness values are assigned to each trading rule (not to individual buy and sell rules) according to the following formula:

$$f(i) = \left(\frac{g(i)}{g(i) + b(i)} \right)^{p(i)+1} \tag{6.1}$$

where:

- $f(i)$ = fitness value
- $g(i)$ = number of correct predictions
- $b(i)$ = number of incorrect predictions
- $p(i)$ = penalty for large code trees

and:

$$p(i) = p_{buy}(i) + p_{sell}(i) \tag{6.2}$$

where:

$$p_{buy}(i) = \frac{\max(0, depth_{buy} - depth_{optimal})}{depth_{optimal}}$$

$$p_{sell}(i) = \frac{\max(0, depth_{sell} - depth_{optimal})}{depth_{optimal}} \tag{6.3}$$

where:

- $p_{buy}(i)$ = penalty of buy rule tree
- $p_{sell}(i)$ = penalty of sell rule tree
- $depth_{buy}$ = depth of buy rule tree
- $depth_{sell}$ = depth of sell rule tree
- $depth_{optimal}$ = depth of the optimal rule tree

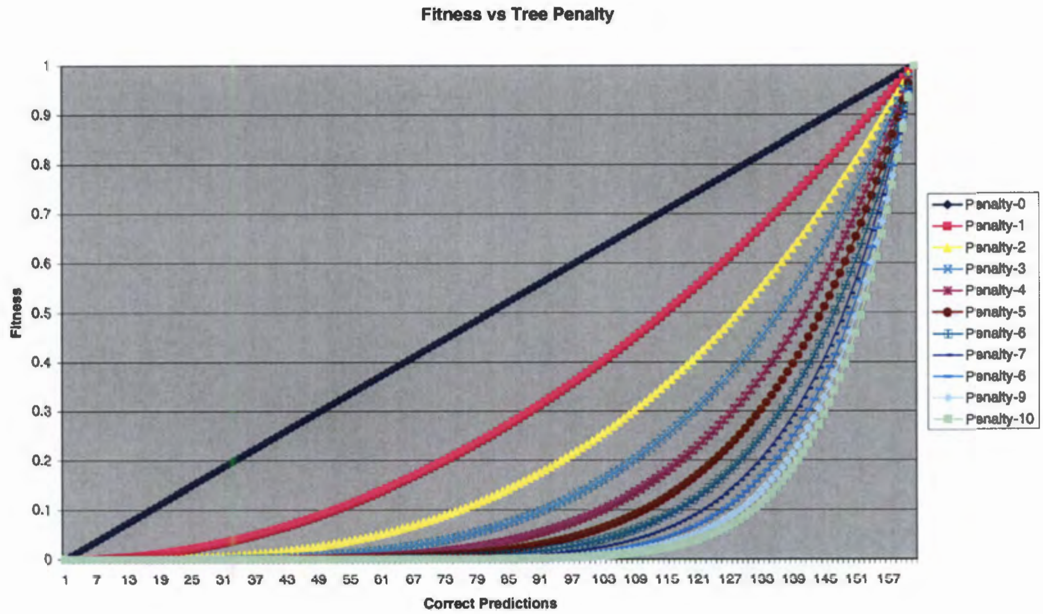

Figure 6.10: Rule fitness vs. Tree penalty

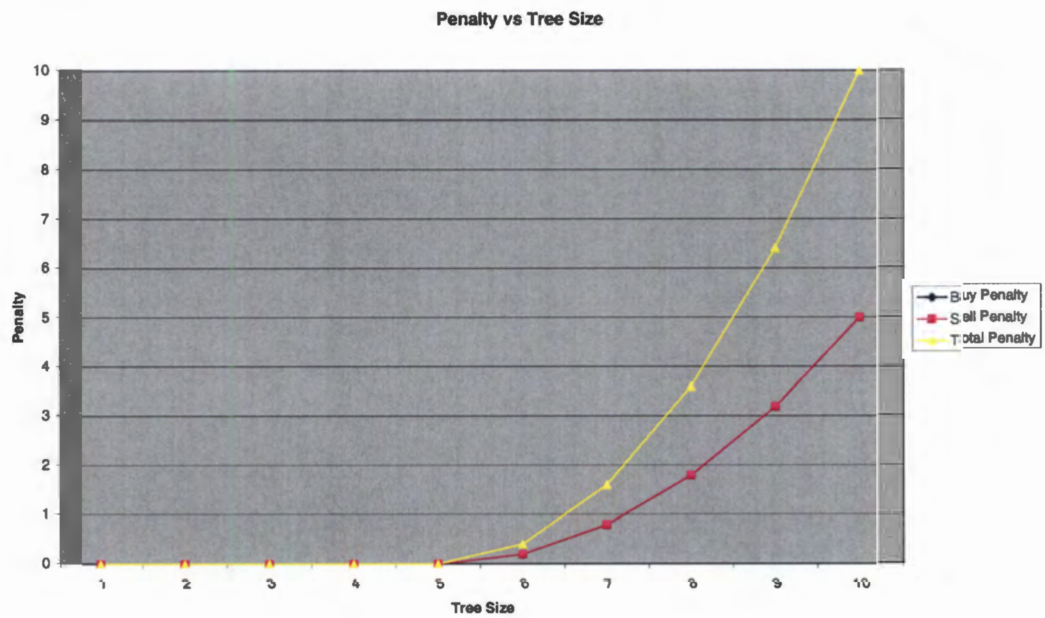

Figure 6.11: Tree penalty vs. Tree size

The trading rules are then ranked based on these fitness values (in descending order) and rules are selected for evolution based on their associated probability of selection, $p(i)$.

The rule fitness graph (figure 6.10) above clearly indicates the dynamic relationship between rule fitness and individual tree penalty. In the ideal case where the tree is within the penalty-less tolerance level, the plot results in a straight line graph, indicating a linear relationship between fitness and penalty. As the tree penalty grows, the relationship becomes more inverse logarithmic. This methodology ensures that the selection of rules for evolution is biased in favour of low penalty rules with high success rates.

Fitness rules were penalised on the basis of tree length (figure 6.11). As discussed earlier in chapter 3, longer trees tend to promote the spread of introns that reduce rule effectiveness. The introduction of penalties will reduce the size of the trees generated, reduce the effect of introns and therefore increase the overall rule fitness.

6.2.2 PORTFOLIO EVOLUTION

At the end of every training period, each trader activates the fittest trading rule with which to trade in the following learning period. Each such rule generates a set of signals that correspond to buy, sell or hold actions for each stock in the market. An initial collection of portfolios is created based on this array of actions and fitness values are assigned to each. The fitness values are calculated as follows:

$$f(i) = \sum s(i) - \sum p(i) + cash \tag{6.4}$$

where:

- $f(i)$ = fitness value
- $s(i)$ = total value of shares sold
- $p(i)$ = total value of shares purchased
- $cash$ = total amount of cash available

$$s(i) = [price(i) * (1 + E(i)] * shares(i) \tag{6.5}$$

where:

- *price(i)* = closing price of the previous day
- *E(i)* = estimated return on the stock calculated using the CAPM
- *shares(i)* = number of shares sold

$$E(i) = rfr + \beta(mr - rfr) \tag{6.6}$$

where:

- rfr = risk free return
- mr = average market return
- β = security beta

$$p(i) = shares(i) * price(i) * (1 + c) \tag{6.7}$$

where:

- *shares(i)* = number of shares bought
- *price(i)* = opening price of the current day
- c = average transaction cost (usually between 0.1-0.5%)

6.2.2.1 CROSSOVER

At each step of the mutation process, a population of portfolios is created. The crossover operator selects two portfolios at random from the big population which will form the parents in the crossover operation. A bag of stocks (and respective shares held) is created from which the offspring will select stocks randomly. To increase the probability of the offspring selecting a stock that is common to *both* parents, the stock is added to the bag in multiples of the respect value as defined in chapter 3. All other stocks are added to the bag only once. Stocks are then selected from the bag to generate a new portfolio. Additional constraints have been implemented to ensure that the generated portfolio is valid and can be generated given the current holding of stocks, amount of cash available with which to trade and total transaction cost for the changes. If the constraints are not met, the newly generated portfolio is discarded and replaced by the better-performing parent.

6.2.2.2 MUTATION

During the training period, actions that correspond to buy and sell signals are generated and assigned to each stock. At the end of each training period, the weakest half of the population is replaced by new offspring generated by random assortment recombination. A modified version of the mutation operator is then performed on the entire population of portfolios to derive the optimal solution for the given buy and sell actions. The best resulting solution is then flagged as the active portfolio.

Every iteration of the mutation process results in batches of stocks being modified randomly. Stocks that are to be bought are increased in random multiples of the specified lot size. If the trader is already in the market for a given share, it ignores a corresponding buy signal. Similarly, if the trader is out of the market for a given share and a sell signal occurs, it is ignored. Conversely, if a sell signal occurs and the trader is *in* the market, the stock is *immediately* sold.

6.2.3 MODEL POOL PUBLICATION/SUBSCRIPTION

Traders that performed poorly during a trading period will select models from the pool with which to trade in the following period. The normalized rule performance is calculated as follows:

$$P_{normalized}(i) = \frac{P_{current}(i)}{P_{total}(i)} \tag{6.8}$$

where:

- $P_{current}(i)$ = current performance for a rule
- $P_{total}(i)$ = total trader performance
- $P_{normalized}(i)$ = normalised performance of a rule

The rule performance as reflected in the model pool is updated as follows:

$$P_{pool}(i) = P_{pool}(i-1) + P_{normalized}(i) \tag{6.9}$$

where:

- $P_{pool}(i)$ = cumulative performance of a rule

The selection of rules is skewed in favour of those rules with the best cumulative performance. A probability of selection is assigned to each rule according to the following formula:

$$\Pr(i) = \left(\frac{2 - c + \dfrac{(2 * c - 2) * rank}{n}}{n}\right) \tag{6.10}$$

where:

- $Pr(i)$ = probability of selecting a given rule
- c = bias constant (higher values will favour higher a rank)
- $rank$ = ranked position of the rule
- n = total number of rules in the pool

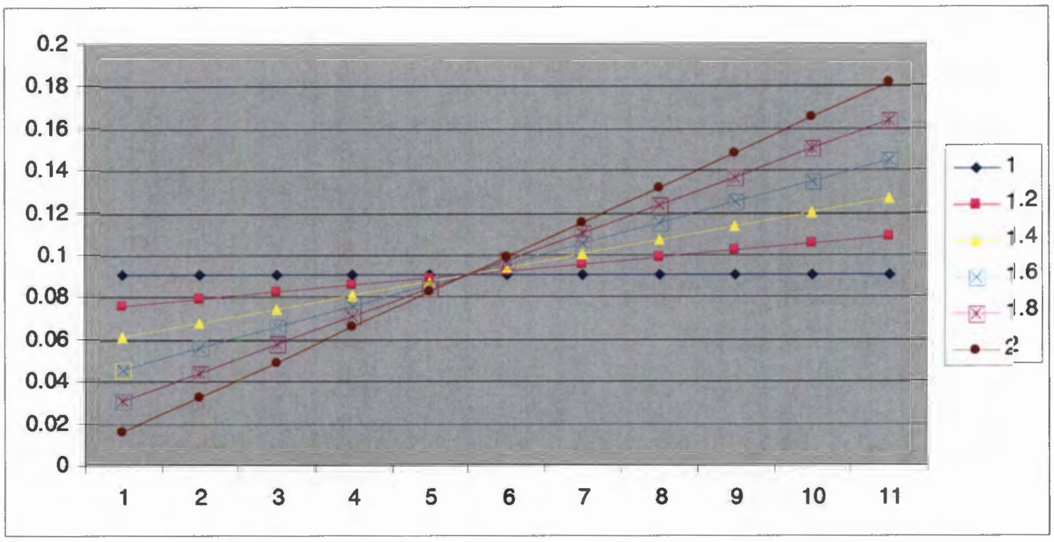

Figure 6.12: Selection probability vs. Bias constant

The linear relationship between selection probability and the bias constant is presented in figure 6.12 above. One can clearly observe the increase in slope and decrease in y-intercept as the bias constant increases in value from 1 to 2. A steeper slope is indicative of the fact that rules with higher probability are more likely to be selected than those with lower probability.

6.2.4 TRADING RULES

Contrary to what was mentioned in Chapter 2, all periods are measured in terms of weeks instead of days. It was decided that weeks be used instead of days as there would be excessive fluctuation in daily stock data. These fluctuations would probably result in the traders making trades too often, not only increasing the chance of making poor decisions, but also increasing the total transaction costs and thereby reducing the average rate of return on investment per time period.

6.2.4.1 GENERIC PERSONALITY

It was decided that the generic trader personality would use the most common technical indicators, including MACD, RSI, ROC and the stochastic oscillator.

6.2.4.1.1 MACD

PERIOD	DURATION
short term	2 weeks
medium term	4 weeks
long term	8 weeks

MACD	BUY RULE	SELL RULE
short term > long term	*BUY*	*HOLD*
short term < long term	*HOLD*	*SELL*

A simple version of the MACD was implemented. Essentially, this implementation of the MACD generates a buy signal when the fast indicator crosses above the slow indicator. Conversely, it generates a sell signal when the slow indicator crosses above the fast indicator.

6.2.4.1.2 RSI

PERIOD	DURATION
short term	2 weeks
medium term	4 weeks

BOUNDARY	THRESHOLD	
	short term	long term
lower	20	30
upper	80	70

RSI	BUY RULE	SELL RULE
RSI > lower bound	*BUY*	*HOLD*
RSI < upper bound	*HOLD*	*SELL*

A buy signal is generated when the RSI crosses above the specified lower bound, while a sell signal is generated when the RSI crosses below the upper bound.

6.2.4.1.3 ROC

PERIOD	DURATION
short term	2 weeks
medium term	4 weeks
long term	8 weeks

ROC	BUY RULE	SELL RULE
ROC(i)>0	*BUY*	*HOLD*
ROC(i)<0	*HOLD*	*SELL*

A buy signal is generated when the ROC crosses from negative to positive. Accordingly, a sell signal is generated when the ROC crosses from positive to negative.

6.2.4.1.4 STOCHASTIC OSCILLATOR

PERIOD	DURATION
short term	4 weeks
medium term	8 weeks
long term	16 weeks

STOCHASTIC	BUY RULE	SELL RULE
K > lower bound OR K > D	*BUY*	*HOLD*
K < upper bound OR K < D	*HOLD*	*SELL*

A buy signal is generated when K% crosses above the lower bound and a sell signal is generated when K% crosses below the upper bound. Similarly, a buy signal is generated when the K% line crosses above the D% line and a sell signal is generated when the opposite occurs.

6.2.4.2 PERSONALITIES

The various technical indicators used by the trader personalities are discussed in this section as well as the reasons for their use. It should be noted that not all the technical indicators used by these traders could be implemented in the system as the data that was required for them could not be obtained. These indicators have not been discussed and are instead put forward as future extensions for similar projects in Chapter 8.

6.2.4.2.1 DREMAN

Dreman is considered a contrarian in investment circles [39]. Contrarians generally refute conventional trading wisdom and so have the edge over other traders. His modus operandi is to seek out-of-favour stocks that are considered undervalued by the market and increase his exposure in them. He then sells the stock as soon as the market values the stock at the same level it values other stocks. The price-earnings ratio is a common means for Dreman to determine whether a company is in favour or out of favour.

Dreman is convinced that his strategy works, but the question can then be raised as to why it is not used more often. His answer to this question is "investor psychology" [39]. According to his theories, investors overreact to events and under well-defined circumstances do it predictably and systematically. Traders in general just cannot persevere with his trading strategies. Dreman prefers to buy out-of-favour stocks as surprises are common in the stock market. Holding in-favour stocks will incur big losses in the event of negative surprises and very little profit on positive surprises. Conversely, out-of-favour stocks will results in minute losses on negative surprises, but can result in large profits on positive surprises.

Dreman proposes the use of the following indicators:

- Return on Equity (ROE)
- Pre-tax profit margin

- Debt-equity ratio
- Price Earnings (PE)
- Price-to-Book Value (PTBV)
- Current ratio
- Payout
- Earnings per Share (EPS)

6.2.4.2.1.1 ROE >= 10%

	BUY RULE	SELL RULE
ROE >= 10%	*BUY*	*HOLD*
RSI < 10%	*HOLD*	*SELL*

Dreman [39] assumes that a high return on equity (ROE) helps to ensure that there are no structural flaws in the company. According to his philosophies, ROE should be greater than the ROE earned from the top third of stocks and that any ROE greater than approximately 27% is "staggering".

6.2.4.2.1.2 PRE-TAX PROFIT MARGIN >= 8%

	BUY RULE	SELL RULE
PRE-TAX PROFIT MARGIN >= 10%	*BUY*	*HOLD*
PRE-TAX PROFIT MARGIN < 10%	*HOLD*	*SELL*

6.2.4.2.1.3 DEBT-EQUITY RATIO < 20%

(also used by: GRAHAM - DEBT-EQUITY RATIO < 10%)

	BUY RULE	SELL RULE
DEBT-EQUITY RATIO < 20%	*BUY*	*HOLD*
DEBT-EQUITY RATIO >= 20%	*HOLD*	*SELL*

Similar to his first trading guideline, Dreman [39] believes that a low debt-equity ratio is a good indicator of a strong balance sheet.

6.2.4.2.1.4 PE IN BOTTOM 20%

	BUY RULE	SELL RULE
PE RATIO IN BOTTOM 20%	*BUY*	*HOLD*
PE RATIO *NOT* IN BOTTOM 20%	*HOLD*	*SELL*

Dreman [39] conducted empirical studies from 1970 to 1996 and his work revealed that stocks with price-equity ratios in the bottom 20% of the market had on average approximately 4% higher annual returns over the market (19% versus 15.3%). His studies further showed that stocks with low PE ratios had higher returns than stocks with higher PE ratios while exposing investors to less risk.

6.2.4.2.1.5 PTBV IN BOTTOM 20%

	BUY RULE	SELL RULE
PTBV RATIO IN BOTTOM 20%	*BUY*	*HOLD*
PTBV RATIO *NOT* IN BOTTOM 20%	*HOLD*	*SELL*

According to Dreman, book-value is the value of a company's common stocks less all its liabilities and preferred shares. The results stemming from Dreman's work [39] also indicated that stocks with a price-book ratio in the bottom 20% of the market had an average annual return of 18.8% compared to 15.1% for the market – a significant difference of over 3%. It further elicited the fact that, while stocks with a low PB ratio had greater returns than stocks with higher PB, the trader also incurred less risk by trading them.

6.2.4.2.1.6 CURRENT RATIO >= 2
(also used by: GRAHAM)

	BUY RULE	SELL RULE
CURRENT RATIO >= 2	*BUY*	*HOLD*
CURRENT RATIO < 2	*HOLD*	*SELL*

The current ratio refers to the relationship between a company's assets and its current liabilities. It is said to be a good indicator a company's ability to pay current debts and a good identifier of a financially strong company as it implies that finances are highly liquid [39].

It can, however, be a red herring as receivables are regarded as current assets and when there is a large amount of receivables, the company may have difficulty in collecting it [39]. Inventory levels can also be high, but be difficult to sell. In these cases, current assets would appear inflated and result in misguided signals.

6.2.4.2.1.7 PAYOUT TESTS < HISTORICAL AVERAGE

	BUY RULE	SELL RULE
CURRENT RATIO >= 2	*BUY*	*HOLD*
CURRENT RATIO < 2	*HOLD*	*SELL*

Dreman [39] believes that a low payout ratio is a good indicator that a company is able to raise dividends. Furthermore, if the recent payout ratio is less than the historical average, there is sufficient room to increase the dividend where the payout ratio is the percentage of the company's earnings paid out as dividends.

6.2.4.2.1.8 EPS PREVIOUS < EPS CURRENT

	BUY RULE	SELL RULE
EPS PREVIOUS < EPS CURRENT	*BUY*	*HOLD*
EPS PREVIOUS < EPS CURRENT	*HOLD*	*SELL*

6.2.4.2.1.9 PRICE-DIVIDENDS PER SHARE IN BOTTOM 20%

According to Dreman [39], if the price-dividends per share is in the bottom 20% of the market, the yield should, correspondingly, be in the top 20%. The studies mentioned earlier also showed that stocks with price-dividend ratios in the bottom

20% of the market had an average annual return of 16.1% versus 14.9% for the market [39].

	BUY RULE	SELL RULE
PD RATIO IN BOTTOM 20%	*BUY*	*HOLD*
PD RATIO *NOT* IN BOTTOM 20%	*HOLD*	*SELL*

6.2.4.2.2 O'NEIL

O'Neil does not believe in the age-old adage "buy low, sell high", instead he believes that if one wants the best, one must be prepared to pay for it [39]. In fact his favourite phrase reads: "Buy high, sell even higher" [39]. He is convinced that so-called bargain stocks are price as low as they are as they are usually inferior and investing in them is unlikely to realise long-term success. To quote from his book (How to Make Money in Stocks): "The hard-to-accept paradox in the stock market is that what seems too high and risky to the majority usually goes higher and what seems low and cheap usually goes lower." [39]

His approach requires one to monitor stocks constantly as decisions are sometimes made based on the movement of stocks *during* the day. It is fairly risky and but this risk controlled by a disciplined approach to buying and selling. Traders using his approach must base buys and sells on his indicators and must adhere to his rules religiously.

The following technical indicators are used by O'Neil:

- RSI
- ROE
- EPS growth rate
- Price

6.2.4.2.2.1 RSI > 80

	BUY RULE	SELL RULE
RSI >= 80	*BUY*	*HOLD*
RSI < 80	*HOLD*	*SELL*

To quote O'Neil [39]: "Relative strength measures the cold, realistic auction marketplaces appraisal of a stock, in spite of the theoretical value of the company on its past popularity, name and image. How did the stock's price behave in the market in the last year? Its running 12 month performance is updated daily, compared to all other stocks and then placed on the same easy-to-use scale."

O'Neil continues to elaborate and states that a leader is defined as a company with a relative strength greater than 70. This implies that its stock has outperformed 70% of the stocks in the comparison group during a given period. Researchers have determined that the 500 best-performing stocks on the NYSE had an average RS of 87 just before their prices increased substantially. Therefore, according to O'Neil's theories, one should neither buy nor hold stocks with a relative strength of less than 70.

6.2.4.2.2.2 ROE < 17%

	BUY RULE	SELL RULE
ROE <= 17%	*BUY*	*HOLD*
ROE > 17%	*HOLD*	*SELL*

O'Neil [39] prefers to invest in companies that show a return on equity of at least 17% over the last year. Essentially, this means that its net income over the last four financial quarters must be 17% greater than its total equity value.

6.2.4.2.2.3 EPS GROWTH RATE > 18%

	BUY RULE	SELL RULE
EPS GROWTH RATE <= 17%	*BUY*	*HOLD*
EPS GROWTH RATE > 17%	*HOLD*	*SELL*

According to O'Neil's analysis [39], it is advisable to avoid holding shares in a stock whose earnings per share for the most recent quarter is less than 18% than that in previous years. Applying his methodology, the greater the percentage increase, the more attractive the stock. In addition, it is considered a bonus if the current quarterly EPS growth rate is greater than or equal to 18% *and* the current quarterly EPS is 25% greater than that estimated by analysts.

6.2.4.2.2.4 PRICE WITHIN 15% OF A 52 WEEK HIGH

	BUY RULE	SELL RULE
PRICE WITHIN 15% OF 52 WEEK HIGH	*BUY*	*HOLD*
PRICE NOT WITHIN 15% OF 52 WEEK HIGH	*HOLD*	*SELL*

Traders would, ideally, want to purchase shares in a stock that dips down in price, comes back and has been below the high for at least six weeks as this gives it sufficient time to consolidate and "build a base for breakout" [39]. One method of determining this behaviour is the identification of "cup and handle" patterns (a dip and recovery after a major consolidation period). In light of this, stocks that are not trading within 15% of their 52 week high should be avoided and those that do meet this requirement should pique interest.

6.2.4.2.3 GRAHAM

Graham believed in studying the fundamental characteristics of a company, namely the financial statements and performance [39]. This does not mean that he was entirely oblivious to the changes in environment surround the stock, but he thought that the stock of a solid company with good prospects, that is undervalued, would be successful over the long-term.

His techniques focused more on the defensive investor who is conservative and fairly passive as he was sceptical that active investors could beat the market consistently. To quote Graham [39]: "The defensive (or passive) investor will place his chief emphasis on the avoidance of serious mistakes or losses. His second aim will be freedom from effort, annoyance, and the need for making frequent decisions. The determining trait of the enterprising (or active, or aggressive) investor is his willingness to devote time and care to the selection of securities that are both sound and more attractive than the average."

The following indicators are used by Graham:

- PE
- Price-to-book (PB)
- Long-term liabilities : net current assets

6.2.4.2.3.1 PE <= 15

	BUY RULE	SELL RULE
PE <= 15	*BUY*	*HOLD*
PE > 15	*HOLD*	*SELL*

Dreman [39] prefers a moderate price-equity ratio and gives preference to stocks whose price does not exceed 15 times the amount the company earned per share. A stock with a moderate PE ratio equates to a more defensive stock, whereas a higher PE ratio equates to a more speculative stock as there is less profit per share supporting the price.

6.2.4.2.3.2 PB <= 1.5 *or* PE*PB <= 22

	BUY RULE	SELL RULE
PB <= 1.5 or PE*PB < = 22	*BUY*	*HOLD*
PB > 1.5 and PE*PB > 22	*HOLD*	*SELL*

Price-to-book ratio refers to the ratio between stock price and the book value per share (book value is total assets less intangible assets less liabilities). This ratio is therefore an indicator of the price of the stock versus the tangible assets. Since Graham [39] prefers tangible assets, preference is given to stocks with reasonable PB ratios.

6.2.4.2.3.3 LONG-TERM LIABILITIES < NET CURRENT ASSETS

	BUY RULE	SELL RULE
LONG-TERM LIABILITIES < NET CURRENT ASSETS	*BUY*	*HOLD*
LONG-TERM LIABILITIES >= NET CURRENT ASSETS	*HOLD*	*SELL*

The ratio, long-term liabilities to net current assets, is a liquidity measure mainly aimed at industrial companies (but holds for other as well) [39]. If a company's long-term debt is lower than its net current assets, it is said to be in a strong financial position to meet its long-term obligations. This display of financial security is very important in Graham's stock selection process.

6.2.4.2.4 LYNCH

Lynch's theories are aimed at the general public who are not schooled in finance and do not have access to the financial information and resources that are available to professional investors. His main premise is that one can have the edge over other investors if one has personal knowledge of something positive about a company (hence invest in what you know). He also advises to look for opportunities that have not yet been discovered by other investors.

In an introduction to his book "One Up on Wall Street" he provides the following disclaimer [39]: "Peter Lynch doesn't advise you to buy stock in your favourite store just because you like shopping in the store, nor should you buy stock in a manufacturer because it makes your favourite product or a restaurant because you like the food. Liking a store, a product, or a restaurant is a good reason to get interested in a company and put it on your research list, but it's not enough of a reason to own the stock! Never invest in any company before you've done the homework on the company's earnings prospects, financial condition, competitive position, plans for expansion, and so forth."

The technical indicators used by Lynch include:

- EPS
- Inventory : Sales
- PE growth ratio

6.2.4.2.4.1 EPS > 0

Lynch uses this simple indicator to determine whether or not a company is profitable and requires that the earnings per share of the company in question be positive over the last year.

	BUY RULE	SELL RULE
EPS > 0	*BUY*	*HOLD*
EPS <= 0	*HOLD*	*SELL*

6.2.4.2.4.2 CHANGE IN INVENTORY : SALES

It is generally considered a red flag when inventory increases faster than sales. Lynch [39] gives an allowance of up to 5% points, i.e. if sales are up 10%, inventory cannot be allowed to rise more than 15%. Lynch therefore considers a company attractive if inventory increases less than or equal to sales or increases less than 5% faster that sales.

	BUY RULE	SELL RULE
CHANGE IN INVENTORY:SALES <= 5%	*BUY*	*HOLD*
CHANGE IN INVENTORY:SALES > 5%	*HOLD*	*SELL*

6.2.4.2.4.3 PE GROWTH RATIO

The price-earnings growth ratio is a determinant of whether or not a stock is fairly priced. According to Lynch's model [39], stocks are considered fairly priced if the price-earnings ratio is less than or equal to the stock's historical growth rate. If PE is less than half the historical growth rate, it is considered a bargain [39].

	BUY RULE	SELL RULE
PEG RATIO : HISTORICAL GROWTH RATE > 0 *AND* < 1.8	*BUY*	*HOLD*
PEG RATIO : HISTORICAL GROWTH RATE < 0 *OR* > 1.8	*HOLD*	*SELL*

6.3 DATA

Weekly stock market data was obtained for the Johannesburg Stock Exchange for the period January 1999 to December 2002 by using the REUTERS Excel application. The following stocks were included:

ABSA GROUP	CAXTON CTP PUBLISH PRINT	KAGISO MEDIA	RAINBOW CHICKEN
ACUCAP PROPERTIES	CERAMIC INDUSTRIES	KAP INTL.	REAL AFRICA
ADCORP	CITY LODGE HOTELS	KUMBA RESOURCES	REDEFINE INCOME FD.
ADVTECH	CLIENTELE LF.ASR.	LEWIS GROUP	REMGRO
AECI	COMBINED MOTOR	LIBERTY GROUP	RESILIENT PR.FD.
AFGRI	CONSOL	LIBERTY INTL. (JSE)	REUNERT
AFRICAN BANK INVS.	CORONATION FD.MGRS.	MADISON PROP FD.MNGRS	RICHEMONT SECS. (JSE)
AFRICAN OXYGEN	DATATEC	MAKALANI INVS.	RMB
AFN.RAINBOW MRLS.	DELTA ELECT.INDS.	MARTPROP PR.FUND	SABMILLER (JSE)
ALEXANDER FORBES	DIMENSION DATA HDG.(JSE)	MASSMART	SANLAM

ALLAN GRAY PR.TRUST	DISCOVERY	MEDI CLINIC	SANTAM
ALLIED ELECTRONICS	DS.& WHSG.NETWORK	MERAFE RESOURCES	SAPPI
ALLIED ELTN.PTG.PREF.	DRD GOLD	METAIR INVESTMENTS	SASOL
ALLIED TECHNOLOGIES	EDGARS CONS.STORES	METOREX	SCHARRIG MINING
AMAL.APPC.	ELAND PLATINUM	METROPOLITAN HDG.	SHOPRITE
ANGLO AMERICAN (JSE)	ELLERINE	MITTAL STEEL SA.	SOUTH AF.RET.PROPS.
ANGLO PLATINUM	EMIRA PROPERTY FUND	MR PRICE GROUP	SPAR GROUP
ANGLOGOLD ASHANTI	FAMOUS BRANDS	MTN GROUP	SPEARHEAD PROPS.
APHEXI PROPERTIES 'A'	FIRSTRAND	MURRAY & ROBERTS	SPUR
APHEXI PROPERTIES 'B'	FOSCHINI	MUSTEK	STANDARD BK.GP.
ARGENT INDUSTRIAL	FREESTONE PR.HDG.	MVELAPHANDA GP.	STEINHOFF INTL.
ASPEN PHMCR.	GOLD FIELDS	MVELAPHANDA RES.	SUN INTERNATIONAL
ASTRAL FOODS	GOLD REEF RESORTS	NAMPAK	SUPER GROUP
ASTRAPAK	GRINDROD	NASPERS	SYCOM PROPERTY FUND
ATLAS PROPERTIES	GROUP FIVE	NEDBANK GROUP	TELKOM
AVENG	GROWTHPOINT PROPS.	NETWORK HEALTHCARE	TIGER BRANDS
AVI	HARMONY GOLD MNG.	NEW CLICKS HDG.	TIGER WHEELS
BARLOWORLD	HIGHVELD STL.& VNM.	NORTHAM PLATINUM	TONGAAT HLT.GP.
BARPLATS INVS.	HUDACO	OCEANA GROUP	TOURISM INV.
BELL EQUIPMENT	HYPROP INVESTMENTS	OLD MUTUAL (JSE)	TRANS HEX GROUP
BHP BILLITON (JSE)	IFOUR PROPERTIES	OMNIA	TRENCOR
BIDVEST GROUP	ILIAD AFRICA	PALABORA MINING	TRUWORTHS INTL.
BRAIT SA. (JSE)	ILLOVO SUGAR	PANGBOURNE PROPS.	UNITRANS
BRANDCORP	IMPALA PLATINUM	PEERMONT GLOBAL	VUKILE PR.FUND
BRIMSTONE INV.'N'	IMPERIAL	PEREGRINE	WESCO INVESTMENTS
BUSINESS CONNEXION GROUP	INVESTEC	PICK N PAY STORES	WESTERN AREAS
BYTES TECH.GP.	INVESTEC (JSE)	PREMIUM PROPERTIES	WILSON BAY HLM OVC
CADIZ	INVICTA	PRETORIA POR.CMT.	WITS.CONS GD.RES.
CAPITAL PROPERTY FD.	JD GROUP	PRIMEDIA	
CAPITEC BANK	JOHNNIC COMMS.	PRIMEDIA 'N'	
CASHBUILD	JSE	PSG GROUP	

Table 6.4: List of stocks used in testing

Stock data was retrieved in the following CSV tabular format:

COLUMN	DESCRIPTION
Date	Date data was recorded on
Type	REUTERS internal number
NAME	Stock name

MNEM	Stock abbreviated name
P	Closing stock price
RI	Returns index
PE	Price Earnings
DY	Dividend Yield
MV	Market Value
NOSH	Number of Shares
EPS	Earnings Per Share
PTBV	Price to Book Value
MTBV	Market to Book Value
POUT	Payout
DPS	Dividends Per Share
ICBT	Interest Cover
BETA	Measured Risk
PC	
TOTAL SALES	
ROE	Return on Equity
TOTAL DEBT	
EQUITY CAPITAL AND RESERVES	
TOTAL CURRENT ASSETS	
TOTAL CURRENT LIABILITIES	
NET CURRENT ASSETS	
ORDINARY DIVIDENDS - GROSS	
OPERATING PROFIT MARGIN	
PRE-TAX PROFIT MARGIN	
NET PROFIT MARGIN	
TRADE DEBTORS	
CASH EARNINGS PER SHARE	
TOTAL STOCK AND WORK IN PROGRESS	

Table 6.5: List of data columns and corresponding meanings

This data was then converted into a chunked format that was easier and faster to stream directly from a binary file. The chunked format also allowed for easier caching as the chunks could be cached directly without the need for post-processing. For the purposes of this thesis and future expansion, a least-recently used cache system was implemented. The data was chunked hierarchically by attribute:

- Stock ID (synonymous with stock name)
- Date (converted to contiguous integer)
- Column (analogous to columns from CSV)

The binary file not only resulted in faster loading times, but also reduced file size.

6.4 CONCLUSION

The agent-based learning framework was introduced and discussed in this chapter. The design decisions and issues were explained and the resulting implementation details were examined and presented. The main features of the system are evolutionary learning, knowledge retention and knowledge sharing between agents. The combination of these ideas enables the system to deal with a constantly changing environment.

7. RESEARCH METHODOLOGY AND RESULTS

In the previous chapters the evolutionary trading system was introduced and explained and the theories that lead to its development described. This chapter investigates the effectiveness of the system with regards to its performance on the stock market and the ability to combine ideas from different expert traders.

The aims of the experiments are discussed in Section 6.1. The evaluation method and the reasons for its use are discussed in Section 6.2. The parameters used to set up and run the experiments are stipulated in Section 6.3. Finally, the results and conclusions are presented in Section 6.4.

7.1 AIMS

An evolutionary trading system has been implemented using a technical trading methodology as its basis. It was decided that the trading strategy be investigated to determine whether or not it results in profitable returns and whether or not the returns thus observed are greater than the returns that would have been realised had the capital been invested in the ALSI.

One cannot, however, base performance solely on the magnitude of the returns generated. It is possible for a trader to strike it lucky in one period, underperform in subsequent periods and still realise great returns. To this extent, portfolio performance measures must be applied to determine the true difference between the two approaches.

Finally, technical indicators from well-known traders were included in the trading rules alongside the more common indicators (MACD, RSI, ROC, stochastic oscillators). Their inclusion was mean to simulate an expert knowledge system[6]. It was decided that the models generated using these indicators should be tested to determine whether or not their inclusion resulted in positive performance.

[6] A computer program that has subject-specific knowledge and analytical skills of a human expert [53]

7.2 METHOD

In order to make sound conclusions and ensure the reliability thereof it was decided that statistical analysis in the form of the student t-test would be performed on the results obtained from the tests runs [40]. In order to test the magnitude of the difference in the means of two populations, one must test the hypothesis that there exists a difference in the population means obtained from these strategies. Similarly, to test the value of the mean of a population, one must test the hypothesis that there exists a difference between the two. By obtaining a sample of the performance from a specified strategy one can draw reliable conclusions by inferring patterns that are exhibited by the entire strategy.

The test procedure requires that the following conditions be met [40]:

- the sampling method for each sample must be random
- each sample is drawn from a normal (or a close approximation thereof) population
- sample size is less than 30

While the first two criteria are easily met, the third has its own set of requirements [40]:

- the sample data are symmetric, unimodal, of size 15 or less and void of any outliers
- the sample data are slightly skewed, of size of between 16 and 40 and void of any outliers

The testing procedure consists of two steps, namely: defining hypotheses, determining decision rules and conclusions. These are presented in more detail below.

7.2.1 DEFINING HYPOTHESES

The null hypothesis for the one-sample case can be formulated as:

$H_0 : \mu_1 = x$

H_1: $\mu_1 \neq x$

or

H_1: $\mu_1 < x$

or

H_1: $\mu_1 > x$

where:

- μ_1 = sample mean of the population
- x = value to compare mean against

7.2.2 DETERMINING DECISION RULES AND CONCLUSIONS

The t-statistic is calculated as follows for the one-sample test [40]:

$$t = \frac{\overline{X} - \mu}{\frac{S_d}{\sqrt{n}}} \text{ with degrees of freedom} = n - 1 \tag{7.1}$$

where:

- t = t-statistic
- $\overline{X}$ = sample mean for population
- μ = hypothesised value for sample mean
- S_d = sample standard deviation for sample
- n = sample size

To determine whether or not H_0 can be rejected given the calculated t-statistic, one must first determine a threshold value against which to test the significance of the t-statistic. The critical t-value can be obtained from most statistics textbooks and is determine by the number of degrees of freedom for the test and the specified significance (α) level. Once the t-value has been determined, the decision rules can be formulated as follows [40]:

- reject H_0 if $|t| > |t\text{-}Critical|$
- accept H_0 if $|t| <= |t\text{-}Critical|$

7.3 TEST PARAMETERS

Due to the initially random nature of the algorithms presented, it was decided that 10 runs would return sufficiently comprehensive results. The parameters used to obtain these results were kept constant across all tests. A brief overview of these parameters is given below:

PARAMETER	VALUE	EXPLANATION
LEARNING_PERIOD	104	Period before trading during which to evolve models only
SOCIAL_PERIOD	16	Period during which to do social learning
TRAINING_PERIOD	4	Period during which to analyse, evolve models and select a new trading strategy
TOTAL_PERIODS	101	Total number of periods to test
REBALANCE_PERIOD	32	Period during which to sell off all stocks
LOT_SIZE	100	Minimum multiple of stocks to purchase
TRADERS	10	Number of traders in simulation
RULES	20	Number of rules each trader maintains
MIN_TREE_DEPTH	2	Minimum depth of GP trees
MAX_TREE_DEPTH	8	Maximum depth of GP trees
OPTIMAL_TREE_DEPTH	5	Optimal tree depth
GP_MUTATION_RATE	0.14	Probability of performing mutation
GP_MUTATION_NODES	10	Maximum number of nodes to mutate
GP_CROSSOVER_RATE	0.85	Probability of performing crossover
GP_REPRODUCTION_RATE	0.01	Probability of performing reproduction

GA_POPULATION	200	Number of individuals to evolve
GA_MUTATION_RATE	0.95	Probability of performing mutation
GA_MUTATION_NODES	4	Number of stocks to mutate
GA_ITERATIONS	100	Number of iterations to evolve over
GA_RESPECT	2	Respect value for the RAR operator
PORTFOLIO_SIZE	161	Maximum size of the portfolio
STOCKS	161	Number of stocks to analyse
TRANSACTION_COST	0.005	Broker fee for buying & selling stocks (0.5%)
RISK_FREE_RETURN	0.06	Guaranteed return on investment
MARKET_RETURN	n/a	Market return is automatically by application
SELECTION_BIAS	1.4	Bias to ranked selection
PROB_TERMINAL	0.35	Probability of selecting a terminal
PROB_OPERATOR	0.65	Probability of selecting an operator

Table 7.1: List of parameters used in testing and corresponding values

The following research questions were thus posed to identify key results from the tests:

- **Do the evolved rules adapt to the market conditions?**
- **Do the evolutionary trading agents realise higher portfolio values than that realised by the JSE ALSI benchmark?**
- **Do the evolutionary trading agents outperform the JSE ALSI benchmark?**
- **Do trading personalities exchange ideas and combine their trading strategies successfully?**

7.4 RESULTS

This section details the results from the research questions that were posed for the project.

7.4.1 Do the evolved rules adapt to the market conditions?

To test whether or not the evolved rules adapt sufficiently well to changing market conditions, it was decided that a simple linear regression be performed on the average fitness of the evolved rules at end of every trading period. This results in a slope-intercept pair that approximates the fitness values at every time step. A one-sample t-test was then used to test whether or not the slope indicated a steady increase in correct trading decisions. These hypotheses were set out as:

- $H_0 : \mu_1 = 0$
- $H_1 : \mu_1 > 0$

where:

- μ_1 = average fitness for the evolved rules
- 0 indicates that there is no growth

Two tests were performed per sample: first at a significance level of 90% ($\alpha_1 = 0.10$) and then at a more significant level of 95% ($\alpha_2 = 0.05$). The following critical values were obtained from the standard t-distribution tables for three degrees of freedom [140]:

- $\alpha = 10\%$, critical t-value = 1.53
- $\alpha = 5\%$, critical t-value = 2.13

The following decision rules were then formulated:

- reject H_0 if $|t| > 1.53$ (similarly for the 95% level)
- accept H_0 if $|t| <= 2.13$ (similarly for the 95% level)

The following results were obtained:

One-sample t-Test	
Sample Mean	0.03
Hypothetical Mean	0
t-Value	4.68

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
t-Value	4.54

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
t-Value	4.48

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
t-Value	3.71

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
t-Value	2.95

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
t-Value	3.86

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
t-Value	3.28

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
t-Value	4.33

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
t-Value	3.72

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
t-Value	5.46

In all the above tests, $t >$ t-critical for both the 10% and 5% significance levels and so we reject H_0. We conclude that there is overwhelming evidence to suggest that the slope of the regression line is positive and therefore the average fitness of the evolved rules increases every time step.

It is important to note that there is a dip in the average fitness of the rules in the second trading period across all the test runs. This could be attributed to a variety of factors. The most probable reason is that the market performed very poorly, very quickly during this period and hence the trading agents were unable to cope with the drastic change. The addition of trading agents and the use of more trading rules could perhaps improve this situation as it would be more likely that one of the additional agents might present a rule that would take itself out of the market during such periods and reintroduce itself when the market situation has been restored. It is also more likely then that this rule would be added to the model pool and selected by poorly performing agents.

7.4.2 Do the evolutionary trading agents realise higher portfolio values than that realised by the JSE ALSI benchmark?

The one-sample t-test was used to test this research question. According to the testing procedure set out in 7.2, the hypotheses were specified as follows:

- $H_0 : \mu_1 = 12{,}203.23$
- $H_1 : \mu_1 > 12{,}203.23$

where:

- μ_1 = final portfolio value of evolutionary trading agents

- 12,203.23 is the Rand value realised by the ALSI given starting capital of 10,000

It was decided that two tests would be performed per sample run: first at a significance level of 90% ($\alpha_1 = 0.10$) and, if the null hypothesis is rejected, at a more significant level of 95% ($\alpha_2 = 0.05$). The following critical values were obtained from the standard t-distribution tables [140]:

- $\alpha = 10\%$, critical t-value = 1.383
- $\alpha = 5\%$, critical t-value = 1.833

From the form of the null hypothesis, the following decision rules were formulated:

- reject H_0 if $|t| > 1.383$ (similarly for the 95% level)
- accept H_0 if $|t| <= 1.383$ (similarly for the 95% level)

The following results were obtained:

Test Run 1

One-sample t-Test	
Sample Mean	17270.76
Hypothetical Mean	12203.23
t-value	2.92

Test Run 3

One-sample t-Test	
Sample Mean	15758.31
Hypothetical Mean	12203.23
t-value	1.93

Test Run 4

One-sample t-Test	
Sample Mean	15363.17
Hypothetical Mean	12203.23

t-value	2.05

Test Run 5

One-sample t-Test	
Sample Mean	23337.26
Hypothetical Mean	12203.23
t-value	2.44

Test Run 6

One-sample t-Test	
Sample Mean	24434.72
Hypothetical Mean	12203.23
t-value	3.07

Test Run 7

One-sample t-Test	
Sample Mean	22466.60
Hypothetical Mean	12203.23
t-value	2.56

Test Run 8

One-sample t-Test	
Sample Mean	17837.47
Hypothetical Mean	12203.23
t-value	2.07

For the above tests, $t >$ t-critical for both the 10% and 5% significance levels and so we reject H_0. We conclude that there is overwhelming evidence to suggest that the mean final portfolio value of evolutionary trading agents is greater than that of the ALSI for tests 1, 3, 4, 5, 6, 7 and 8.

Test Run 2

One-sample t-Test	
Sample Mean	16966.24
Hypothetical Mean	12203.23

t-value	1.46

For test 2, $t = 1.46 >$ t-critical for only the 10% significance level and so we reject H_0 at the 10% level but not at the 5% level. We conclude that there is sufficient evidence to suggest that the mean final portfolio value of evolutionary trading agents is greater than that of the ALSI for test 2.

Test Run 9

One-sample t-Test	
Sample Mean	12399.85
Hypothetical Mean	12203.23
t-value	0.15

Test Run 10

One-sample t-Test	
Sample Mean	13510.07
Hypothetical Mean	12203.23
t-value	0.73

For tests 9 and 10, $t <$ t-critical for the 10% significance level and so we accept H_0 and perform no further tests. Therefore, we conclude that there is insufficient evidence to suggest that the mean final portfolio value of evolutionary trading agents is greater than that of the ALSI for tests 9 and 10.

CONCLUSION

From the above it is apparent that the evolutionary trading agents realise significantly higher portfolio values than that realised by the ALSI in 80% (8 out of 10) of the tests, while only 20% (2 out of 10) of the tests resulted in the ALSI performing better. We can safely conclude that the evolutionary trading approach trades more profitably than the ALSI does.

In the cases where the evolutionary trading agents performed better than the ALSI, their average portfolio value was significantly higher than that of the ALSI (19179.32

vs. 12203.23). This represents an average abnormal return over the ALSI portfolio value of 57% over a 101 week period.

7.4.3 Do the evolutionary trading agents outperform the JSE ALSI benchmark?

The fact that the evolutionary trading agents realised higher returns than that of the ALSI does not necessarily mean that the agents outperformed the ALSI in terms of portfolio performance. It is possible that the agents could make large profits from very few trades, followed by very small profits or even losses from subsequent trades and still realise higher portfolio values than that of the ALSI. In order to test this research question, we decided to compare our approach to the ALSI benchmark in terms of the Sharpe ratio. Earlier it was stated that the Sharpe ratio allows one to directly test the difference in performance by simply comparing the values thus obtained. The higher of the two values is said to have resulted in superior performance.

Similar to the test in question in 7.4.1, the one-sample t-test was used to test this research question. The hypotheses were set out as follows:

- $H_0 : \mu_1 = 0.035$
- $H_1 : \mu_1 > 0.035$

where:

- μ_1 = Sharpe ratio for evolutionary trading agents
- 0.035 is Sharpe ratio obtained by the ALSI

The null hypothesis can be worded as: the mean Sharpe ratio for evolutionary trading agents is not significantly different from that of the ALSI. Conversely, the alternative hypothesis can be read as: the mean Sharpe ratio for evolutionary trading agents is larger than that of the ALSI.

Again, it was decided that two tests would be performed per sample run: first at a significance level of 90% ($\alpha_1 = 0.10$) and then at a more significant level of 95% ($\alpha_2 =$

0.05). As the numbers of degrees of freedom for these tests are the same as for the previous, the critical values are also the same:

- $\alpha = 10\%$, critical t-value = 1.383
- $\alpha = 5\%$, critical t-value = 1.833

For the same reason, the decision rules are the same too:

- reject H_0 if $|t| > 1.383$ (similarly for the 95% level)
- accept H_0 if $|t| <= 1.383$ (similarly for the 95% level)

The following results were obtained:

Test 1

One-sample t-Test	
Sample Mean	0.209
Hypothetical Mean	0.035
t-value	2.97

Test 4

One-sample t-Test	
Sample Mean	0.115
Hypothetical Mean	0.035
t-value	2.71

Test 6

One-sample t-Test	
Sample Mean	0.231
Hypothetical Mean	0.035
t-value	3.54

Test 7

One-sample t-Test	
Sample Mean	0.155

Hypothetical Mean	0.035
t-value	2.03

For the above tests, $t >$ t-critical for both the 10% and 5% significance levels and so we reject H_0. We conclude that there is overwhelming evidence to suggest that the mean Sharpe ratio of the evolutionary trading agents is greater than that of the ALSI for tests 1, 4, 6 and 7.

Test 8

One-sample t-Test	
Sample Mean	0.138
Hypothetical Mean	0.035
t-value	1.56

For test 8, $t = 1.56 >$ t-critical for only the 10% significance level and so we reject H_0 at the 10% level but not at the 5% level. We conclude that there is sufficient evidence to suggest that the mean Sharpe ratio of the evolutionary trading agents is greater than that of the ALSI.

Test 2

One-sample t-Test	
Sample Mean	0.072
Hypothetical Mean	0.035
t-value	0.51

Test 3

One-sample t-Test	
Sample Mean	0.075
Hypothetical Mean	0.035
t-value	0.67

Test 5

One-sample t-Test	
Sample Mean	0.136
Hypothetical Mean	0.035

t-value	1.26

Test 9

One-sample t-Test	
Sample Mean	0.042
Hypothetical Mean	0.035
t-value	0.18

Test 10

One-sample t-Test	
Sample Mean	-0.156
Hypothetical Mean	0.035
t-value	-1.12

For tests 2, 3, 5, 9 and 10, t < t-critical for the 10% significance level and so we accept H_0 and perform no further tests. Therefore, we conclude that there is insufficient evidence to suggest that the mean Sharpe ratio of the evolutionary trading agents is greater than that of the ALSI.

CONCLUSION

In light of these statistical tests, we cannot, with absolute certainty, conclude that the evolutionary traders' portfolios outperformed that of the ALSI. In 50% (5 out of 10) of the tests, the results indicated a significant difference between the two, in favour of the evolutionary trading agents, but the other 50% (5 out of 10) of the tests indicated less than stellar performance on their part.

It should be noted however, that in the tests where the ALSI equalled or outperformed the evolutionary agents, the agents' performance was largely skewed by a few poor performing agents, while the rest of the agents' outperformed the ALSI. A case in point is test 5, pictured in figure E5 in Appendix E, in which only two agents underperformed, but the statistical tests still showed no significant difference between the agents' and the ALSI results.

7.4.4 Do trading personalities exchange ideas and combine their trading strategies successfully?

The null and alternative hypotheses were formulated as follows:

$H_0: \mu_1 = 0$

$H_1: \mu_1 > 0$

where:

μ_1 = cumulative trading performance of mixed trading rules

In other words, the null hypothesis can be worded as: the mean cumulative trading performance of mixed rules is zero. The alternative hypothesis can be worded as: the mean cumulative trading performance of mixed rules is positive.

It was decided that two tests would be performed per sample run: first at a significance level of 90% ($\alpha_1 = 0.10$) and, if the null hypothesis is rejected, at a more significant level of 95% ($\alpha_2 = 0.05$). From the form of the null hypothesis, we will reject H_0 if t > T-critical. As there is no guarantee that each test run will result in the generation of the same number of mixed trading rules, it was highly likely that each test would yield a different number of degrees of freedom. The critical t-values were therefore included in each test described below:

Test 7

One-sample t-Test	
Sample Mean	4690.03
Hypothetical Mean	0
Degrees of Freedom	5
t-Value	2.99
Critical t-value (10%)	1.48
Critical t-value (5%)	2.02

Test 8

One-sample t-Test	
Sample Mean	6838.42
Hypothetical Mean	0
Degrees of Freedom	4
t-Value	2.35
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Test 9

One-sample t-Test	
Sample Mean	3760.63
Hypothetical Mean	0
Degrees of Freedom	2
t-Value	6.13
Critical t-value (10%)	1.89
Critical t-value (5%)	2.92

Test 10

One-sample t-Test	
Sample Mean	1638.30
Hypothetical Mean	0
Degrees of Freedom	2
t-Value	2.93
Critical t-value (10%)	1.89
Critical t-value (5%)	2.92

$t >$ t-critical for both the 10% and 5% significance levels for tests 7, 8, 9 and 10, and so we reject H_0. We conclude that there is overwhelming evidence to suggest that the mean returns of the mixed trading rules is positive.

Test 1

One-sample t-Test	
Sample Mean	1441.05
Hypothetical Mean	0
Degrees of Freedom	11
t-Value	1.22

Critical t-value (10%)	1.36
Critical t-value (5%)	1.80

Test 2

One-sample t-Test	
Sample Mean	1101.46
Hypothetical Mean	0
Degrees of Freedom	5
t-Value	0.93
Critical t-value (10%)	1.48
Critical t-value (5%)	2.02

Test 4

One-sample t-Test	
Sample Mean	-2938.00
Hypothetical Mean	0
Degrees of Freedom	5
t-Value	-3.21
Critical t-value (10%)	1.48
Critical t-value (5%)	2.02

$t <$ t-critical for the 10% significance level for tests 1, 2 and 4, and so we accept H_0 and perform no further tests. Therefore, we conclude that there is insufficient evidence to suggest that the mixed trading rules result in positive returns.

CONCLUSION:

The tests performed above indicate that the mixed rules result in positive returns on average 57% (4 out of 7) of the time, whereas negative or neutral returns were realised 43% (3 out of 7) of the time. This can be attributed to the fact that either the mixed rules were tested during a period where the market fell markedly in general or where there were simply insufficient mixed rules applied by the trading agent. Additionally, an initially large drop in price would have made it extremely difficult for the mixed rule to recoup the losses in subsequent trading periods.

Despite the average performance of the mixed rules, the fact stands that the rules *were* generated in 70% of the tests (7 out of 10). Given more time to trade or a different market environment, the rules might have resulted in better performance.

8. CONCLUSIONS AND FUTURE WORK

8.1 CONCLUSIONS

In this thesis a dynamic system for stock trading and portfolio optimisation with individual and social learning has been presented. It should be noted that the system is general enough to be applied to stock markets other than the JSE and can easily incorporate other technical indicators to form new trading strategies.

The research in this thesis can be categorised into three interdependent subsections:

- stock trading
- portfolio optimisation
- individual and social learning

The results from the test runs indicate that the trading strategies generated for stock trading results in profitable rules. In 80% of the tests, the evolutionary strategies realised greater returns than the ALSI and in 50% of the tests, the agents outperformed the ALSI in terms of portfolio performance. Finally, social learning was implemented via a central model pool. Superior models were continually posted to and update from the pool, resulting in the generation of sequentially better trading strategies. Mixed trading rules were generated in 70% of the tests and in 53% of these tests the mixed rules were profitable and resulted in positive returns.

Some of the insights gained from the research include:

- there is sufficient evidence to suggest that an evolutionary trading approach can realise greater returns than that of the benchmark index
- there is insufficient evidence to suggest that the evolutionary trading agents outperform the benchmark index in terms of portfolio performance
- there is sufficient evidence to suggest that social learning can result in the combination of ideas from several expert knowledge systems

- there is insufficient evidence to suggest that models with components from different trader personalities result in positive returns

8.2 FUTURE WORK

There are a number of areas in which the system could be extended.

1. Fine-tune the evolutionary operator parameters

The lack of statistical evidence in answering the research questions suggests that more time should be spent in fine-tuning the parameters used in the evolutionary algorithms. A finer balance must be struck between the probabilities associated with mutation, crossover and reproduction. The relatively high probability set for mutation in this research might have resulted in the generation of sub-par models – however, this cannot simply be taken at face value. Instead, more rigorous testing should be done by varying the probabilities of the evolutionary operators.

2. Use more stable and established stock markets (FTSE/NYSE)

For the purpose of this thesis, it was deemed that all the stocks in the JSE should be incorporated into the analysis. Smaller, more volatile securities were therefore included and this might have resulted in the presence of bias in the results. Alternative approaches could include the use of only the top-performing securities, or the use of an entirely different stock market. Perhaps more stable and established markets could be introduced, for example the FTSE100.

3. Use more data

Relatively few data points (104 for conditioning and 101 for dynamic learning) were used in this research. Since the models are updated continuously, it would make sense to test the system with a large number of data points. This would result in a longer period during which the models could evolve and perhaps converge towards more successful trading strategies.

4. Incorporate more complex and varied technical indicators

The technical indicators used in the generation of the trading strategies were simplified versions of those used by real-life traders. Not only could more complex variants thereof be implemented, but completely different indicators could be implemented to increase variety and reduce the likelihood of early convergence and overfitting. In addition, several technical indicators used by the trader personalities could not be implemented in the final version of the system as the information that was required could not be obtained. The inclusion of these indicators might have resulted in improved performance.

5. Dynamically switch between weekly trading and a buy-and-hold approach

The performance levels achieved by weekly trading and that of the buy-and-hold approach are very similar. It can be expected that each would be more appropriate in certain market conditions. A possible future extension to this work could include the traders being able to switch between the two approaches depending on which has performed better in recent time periods. This should result in better overall portfolio performance as the trader will be better equipped to adapt to the changing market environment.

8. BIBLIOGRAPHY

[1] Anonymous 2007. The Ultimate Money Machine, In *Iran Daily*, May 2007

[2] Duhigg, C. 2006. Artificial intelligence applied heavily to picking stocks, In *The New York Times*, November 2006

[3] Rubenstein, M. 2002. Markowitz's "Portfolio Selection": A Fifty-Year Retrospective, In *Journal of Finance,* vol. 57, no. 3, pp 1041-1045

[4] Markowitz, H. 1952. Portfolio Selection, In *The Journal of Finance*, vol. 7, no. 1 (1952), p77-91

[5] Reilly, F., Brown, K. 2000. Investment Analysis and Portfolio Management, Sixth Edition

[6] Busetti, F. 2000, *Metaheuristic approaches to realistic portfolio optimisation.* Thesis (MSc). University of South Africa

[7] Kugi, V. 1999. *Performance Evaluation.* Thesis (MSc). University of Wien

[8] Daniel, K. November 1999. *Lecture Notes 7: Active and Passive Portfolio Management.* Department of Finance, Kellogg School of Management, Northwestern University

[9] Aldrian, J. 2000, *Portfolio Performance Evaluation.* Thesis (MSc). University of Wien

[10] Skiena, S. 2004, *Lecture Notes 11: Technical Analysis,* State University of New York, Stony Brook

[11] Technical Analysis Revisited. Retrieved February 8, 2007, from trading24.org: http://www.trading24.org/

[12] Moving Average Convergence/Divergence (MACD). Retrieved April 13, 2006, from StockCharts.com: http://www.stockcharts.com/

[13] Interpreting Technical Indicators. Retrieved May 3, 2006, from Optima Investment Research, Inc.: http://www.oir.com/

[14] Relative Strength Indicator (RSI), Retrieved April 13, 2006, from StockCharts.com: http://www.stockcharts.com/

[15] Achelis, S., Market Indicator Interpretation Guide: Using The Technician, 1986

[16] Rate of Change (ROC). Retrieved April 13, 2006, from StockCharts.com: http://www.stockcharts.com/

[17] Stochastic Oscillator. Retrieved April 13, 2006, from StockCharts.com: http://www.stockcharts.com/

[18] Whitley, D. 1993. *A Genetic Algorithm Tutorial.* Colorado State AI Lab, Colorado State University

[19] Back, T. 1991. Optimization by Means of Genetic Algorithms. In *36th International Scientific Colloquium*, Technical University of Ilmenau, pp 163-169

[20] Anonymous. 1996. Genetic Algorithms. In *SURPRISE 96 Journal*, vol. 1, Department of Computing, Imperial College of Science Technology and Medicine

[21] Radcliffe, N. 1992. Genetic Set Recombination, In *Foundations of Genetic Algorithms 2*, Morgan Kaufmann

[22] Shapcott, J. 1992. *Index Tracking: GA's for Investment Portfolio Selection.* Report EPCC-SS92-24, Edinburgh Parallel Computing Centre, University of Edinburgh

[23] Koza, J. 1998. Genetic Programming. In *Encyclopaedia of Computer Science and Technology*, Marcel-Dekker

[24] Sulejmanovic, F. 2005. *Stock Price Predictions and Hedge Fund Optimisations using Genetic Programming*. Thesis (MSc). University College of London

[25] Koza, J. 1992. Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press

[26] Luke, S. 2000. Two Fast Tree-Creation Algorithms for Genetic Programming, In *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 3, pp 274-283

[27] Luke, S., Panait, L. 2001. A Survey and Comparison of Tree Generation Algorithms, In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, (San Francisco, California, USA), pp 81-88

[28] Potvin, J., Soriano. P., Vallee, M. 2004, Generating trading rules on the stock markets with GP, In *Computers and Operations Research*, vol. 31, no. 7

[29] Hackett, P. 1995. *A Comparison of Selection Methods Based on the Performance of a Genetic Program Applied to the Cart-pole Problem*. Thesis (Hons). Griffith University

[30] Seshadri, M., Comprehensibility 2003. *Overfitting and Co-Evolution in Genetic Programming for Technical Trading Rules*. Thesis (MSc). Worcester Polytechnic Institute

[31] Luke, S. 2000. *Issues in Scaling GP: Breeding Strategies, Tree Generation, and Code Bloat*. Thesis (PhD). University of Maryland

[32] Whitley, D. 2001. An Overview of Evolutionary Algorithms: Practical Issues and Common Pitfalls, In *Information and Software Technology*, vol. 43, no. 14, pp 817-831

[33] Blumer, A., Ehrenfeucht, A., Haussler, A., Warmuth, D., Warmuth, M. 1987. Occam's Razor, In *Information Processing Letters*, vol. 24, pp 377-380

[34] Domingos, P. 1999. The Role of Occam's Razor in Knowledge Discovery, In *Data Mining and Knowledge Discovery*, vol. 3, pp 409-425

[35] Bauer, R. 2000, Using Genetic Programming to Design a Generalized Trading System, In *Managerial Finance*, vol. 26, no. 6, pp 1-15

[36] Yan, W. 2003. *"Profitable, Return Enhancing" Portfolio Adjustments - An Application of Genetic Programming with Constrained Syntactic Structure*. Thesis (MSc). University College of London

[37] Chang, T., Meade, N., Beasley, J., Sharaiha, Y. 1999, Heuristics for cardinality constrained portfolio optimisation, In *Computers and Operations Research*, vol. 27, pp 1271-1302

[38] Korczak, J., Lipinski, P. 2001. Evolutionary Approach to Portfolio Optimization, In *Proceedings of Workshop on Artificial Intelligence for Financial Time Series Analysis*

[39] Reese, T., Glassman, T. 2002. The Market Gurus, Dearborn Trade

[40] Underhill, L., Bradfield, D. 1994. IntroSTAT 5.0, Juta Academic

[41] Arslanov, I., Kolosovska, K. 2004. Active Portfolio Management with the Application of Adaptive Artificial Intelligence Tools in the Context of the Baltic Stock Market. Thesis (Hons). Stockholm School of Economics

[42] Bradshaw, J. M. 1997. Software Agents. The Mit Press

[43] Luck, M. Introduction to Autonomous Agents and Multi-Agent Systems, Department of Electronics and Computer Science, University of Southampton, Retrieved August 28, 2007, from: http://www.ecs.soton.ac.uk/~mml

[44] Pereira, D., Moreira, N. 2005. The Role of Emotions in BDI Agents. Retrieved August 28, 2007, from: http://www.ncc.up.pt/%7Edpereira/tese/relatorios/a_fazer/techreport/techreport.html

[45] Guerra-Hernandez, A., El Fallah-Seghrouchni, A., Soldano, H. Learning in BDI Multi-agent Systems. Retrieved August 28, 2007, from: http://www.-ia.di.fct.unl.pt/~jleite/climalV/12.pdf

[46] Rao, A., Georgeff, M. 1995. BDI Agents: From Theory to Practice, In *Proceedings of the First International Conference on Multi-Agent Systems*

[47] Corkill, D. Blackboard Systems. Blackboard Technology Group, Inc. Retrieved August, 24, 2007, from: http://www.bbtech.com/papers/ai-expert.pdf

[48] Nii, H. 1986. Blackboard Systems, In *AI Magazine*, vol. 7-2, 7-3

[49] Pfleger, K., Hayes-Roth, B. An introduction to Blackboard-Style Systems Organisation. In *Stanford University Technical Report KSL-98-03*

[50] Ahmadabadi, M., Asadpour, M., Nakano, E. 2001. Cooperative Q-learning: the knowledge sharing issue, In *Advanced Robotics*, vol. 15, no. 8, pp 815-832

[51] Garland, A., Alterman, R.1996. Multiagent Learning through Collective Memory. In *Symposium on Adaptation, Co-evolution and Learning in Multi-agent Systems*, pp 33-38

[52] Kendall, G., Su, Y. 2003. A Multi-agent Based Simulated Stock Market – Testing on Different Types of Stocks. School of Computer Science and IT, University of Nottingham

[53] Expert Systems, Association for the Advancement of Artificial Intelligence, Retrieved August 28, 2007, from: http://www.aaai.org/AITopics/html/expert.html

APPENDIX A: RULE FITNESS RESULTS

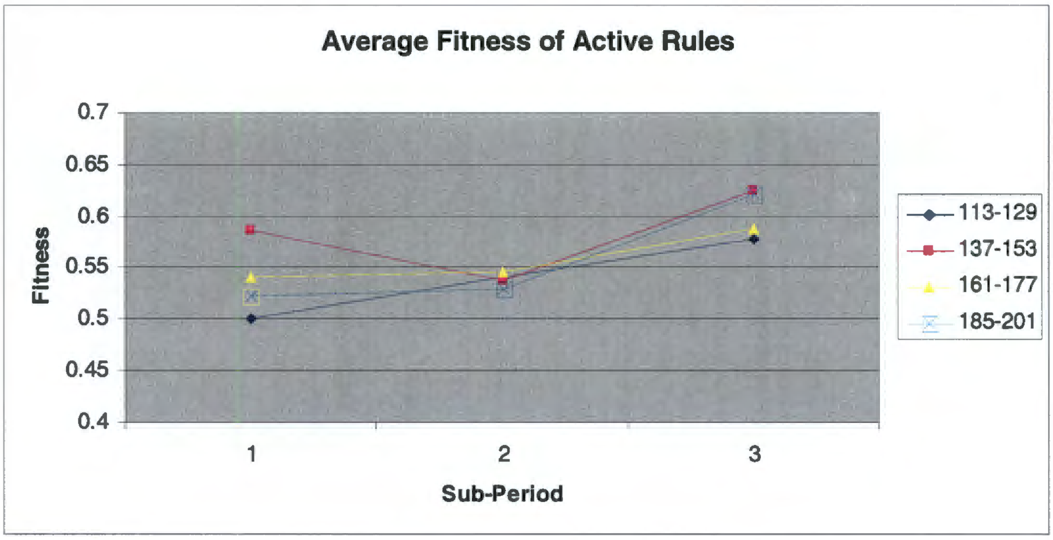

Figure A1: Average fitness of evolved rules for test 1

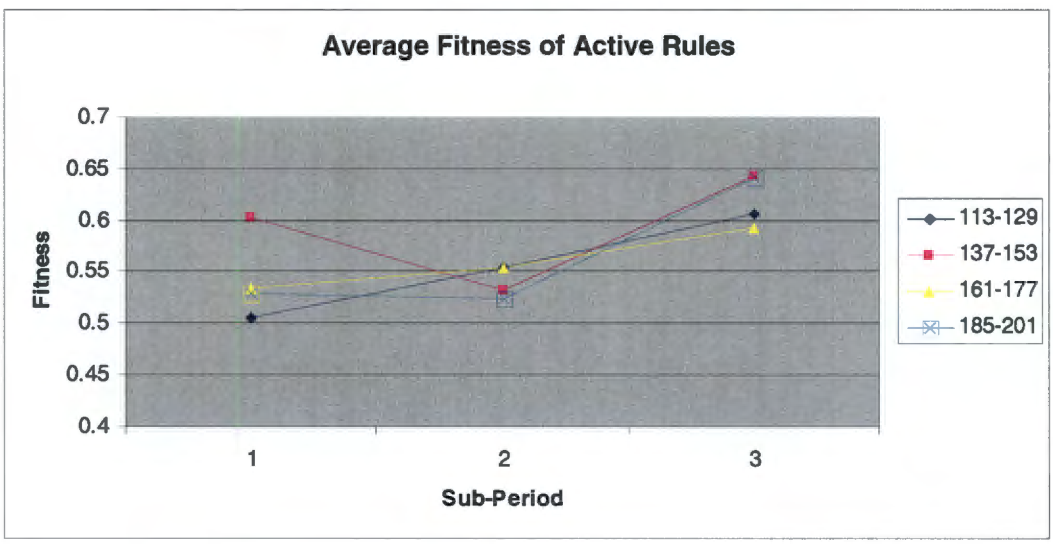

Figure A2: Average fitness of evolved rules for test 2

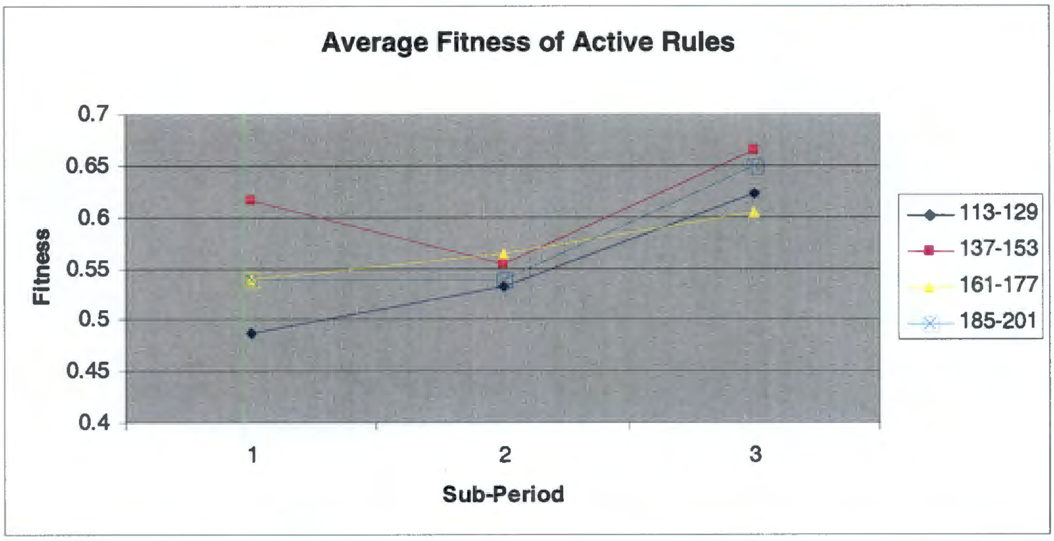

Figure A3: Average fitness of evolved rules for test 3

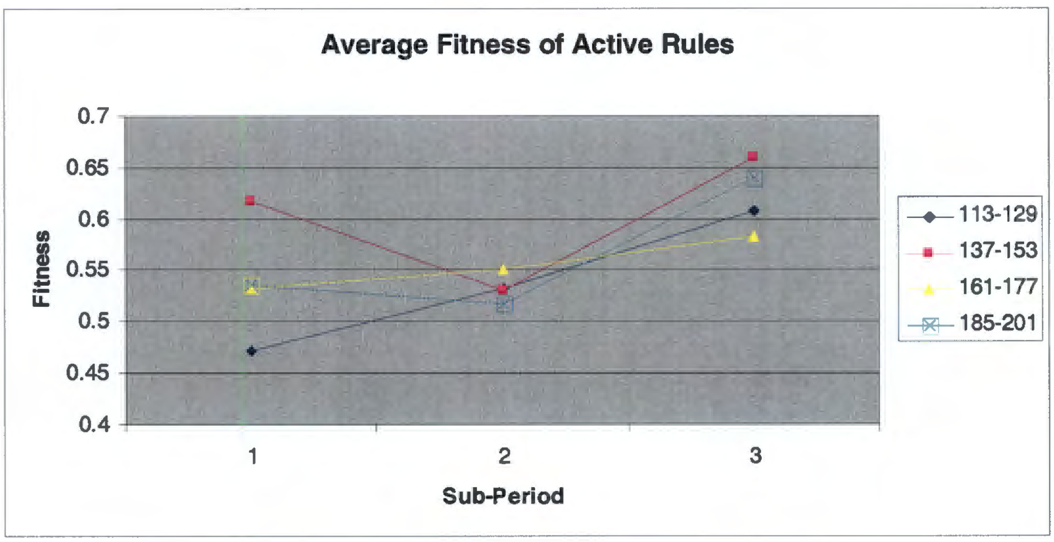

Figure A4: Average fitness of evolved rules for test 4

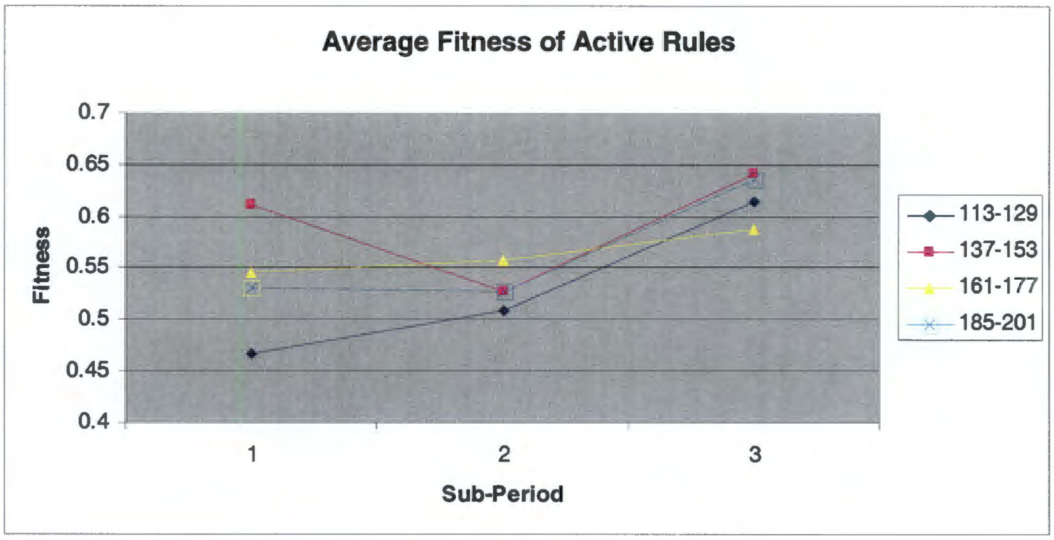

Figure A5: Average fitness of evolved rules for test 5

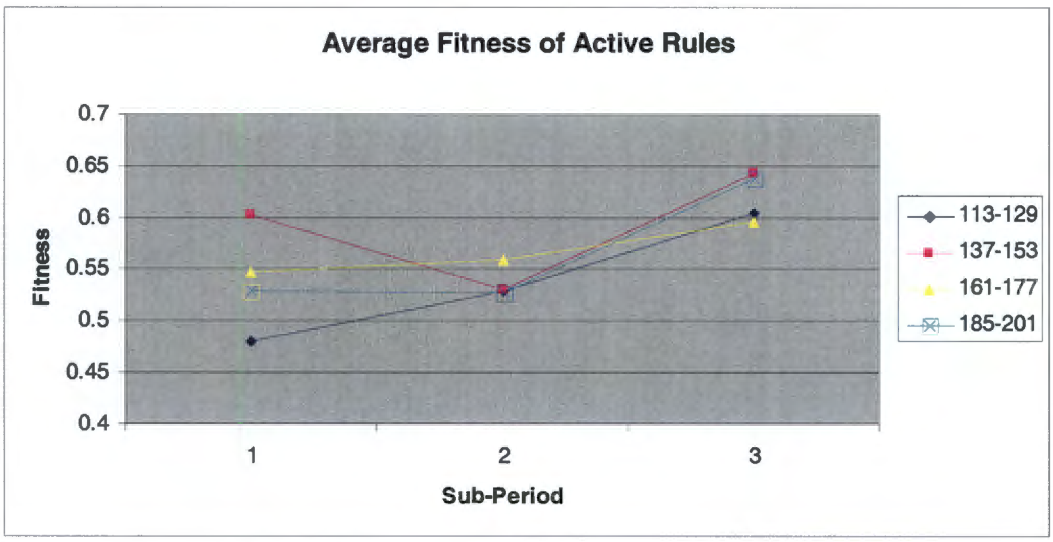

Figure A6: Average fitness of evolved rules for test 6

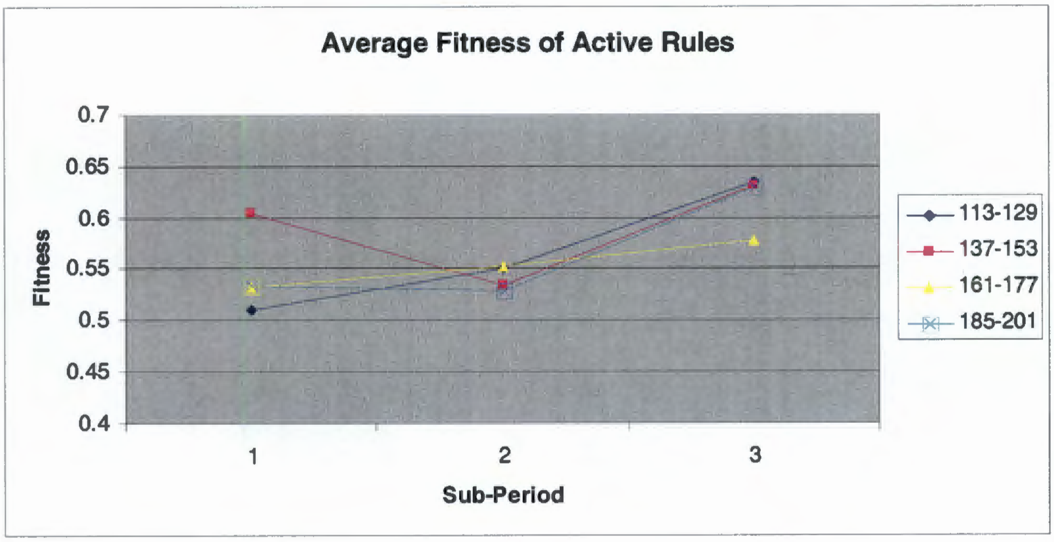

Figure A7: Average fitness of evolved rules for test 7

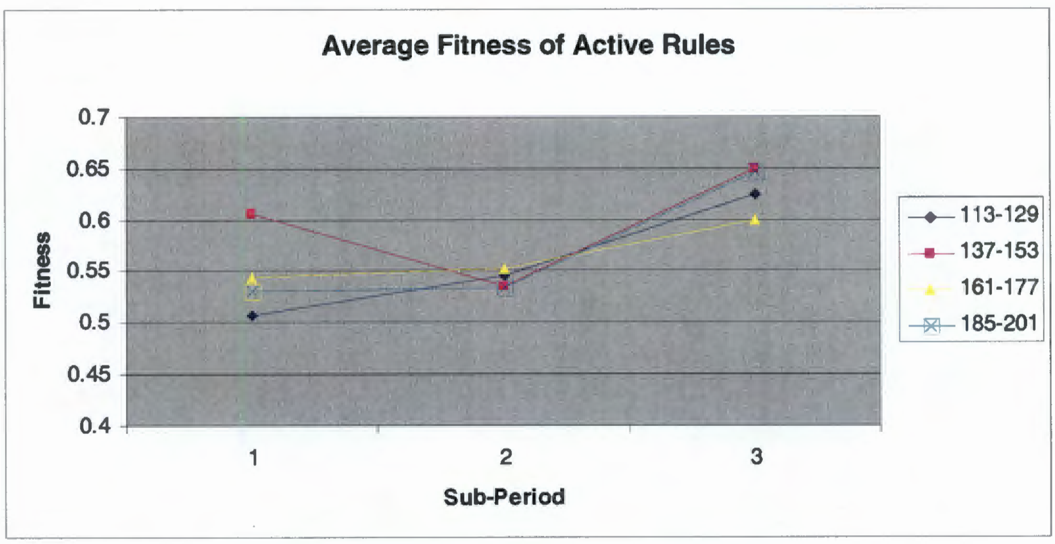

Figure A8: Average fitness of evolved rules for test 8

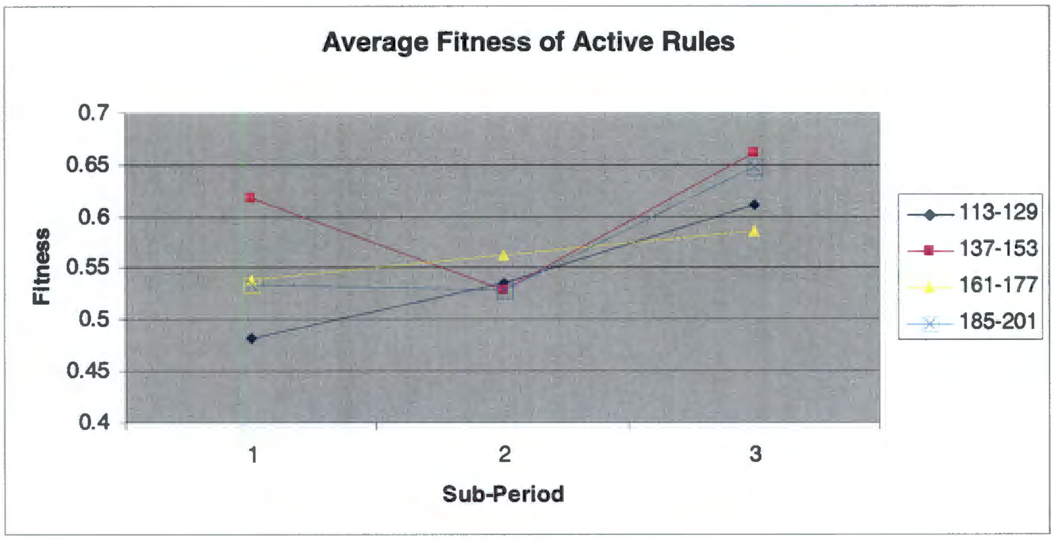

Figure A9: Average fitness of evolved rules for test 9

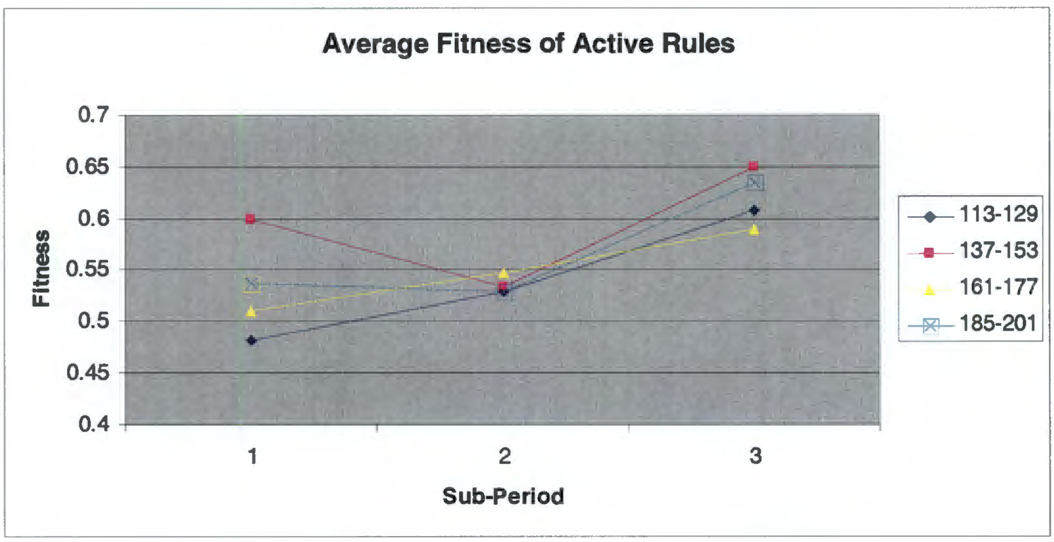

Figure A10: Average fitness of evolved rules for test 10

APPENDIX B: RULE FITNESS TESTS

One-sample t-Test	
Sample Mean	0.03
Hypothetical Mean	0
Degrees of Freedom	3
t-Value	4.68
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table B1: t-Test for average fitness of evolved rules for test 1

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
Degrees of Freedom	3
t-Value	4.54
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table B2: t-Test for average fitness of evolved rules for test 2

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
Degrees of Freedom	3
t-Value	4.48
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table B3: t-Test for average fitness of evolved rules for test 3

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
Degrees of Freedom	3
t-Value	3.71
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table B4: t-Test for average fitness of evolved rules for test 4

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
Degrees of Freedom	3
t-Value	2.95
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table B5: t-Test for average fitness of evolved rules for test 5

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
Degrees of Freedom	3
t-Value	3.86
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table B6: t-Test for average fitness of evolved rules for test 6

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
Degrees of Freedom	3
t-Value	3.28
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table B7: t-Test for average fitness of evolved rules for test 7

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
Degrees of Freedom	3
t-Value	4.33
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table B8: t-Test for average fitness of evolved rules for test 8

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
Degrees of Freedom	3
t-Value	3.72
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table B9: t-Test for average fitness of evolved rules for test 9

One-sample t-Test	
Sample Mean	0.04
Hypothetical Mean	0
Degrees of Freedom	3
t-Value	5.46
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table B10: t-Test for average fitness of evolved rules for test 10

APPENDIX C: PORTFOLIO VALUE RESULTS

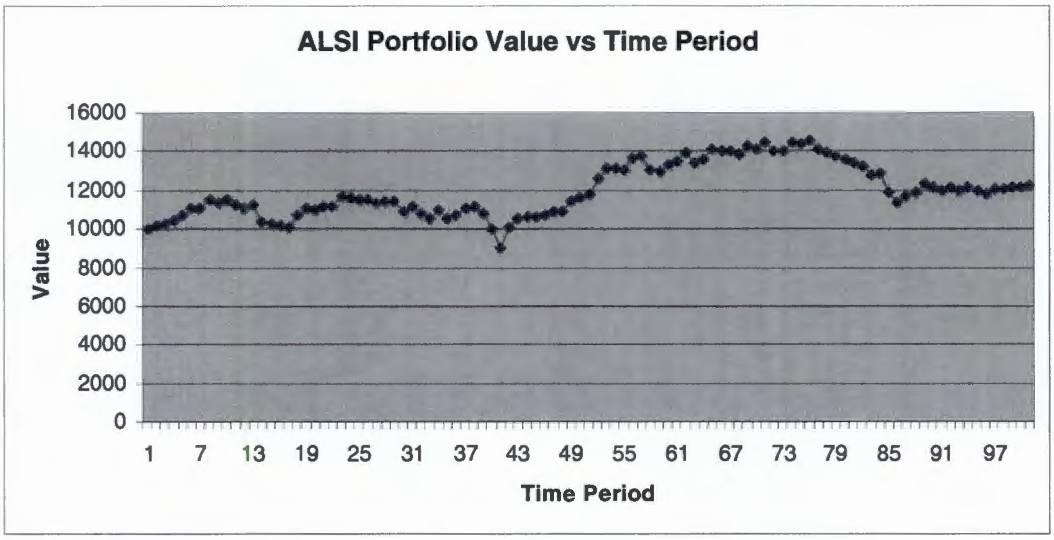

Figure C1: ALSI portfolio values

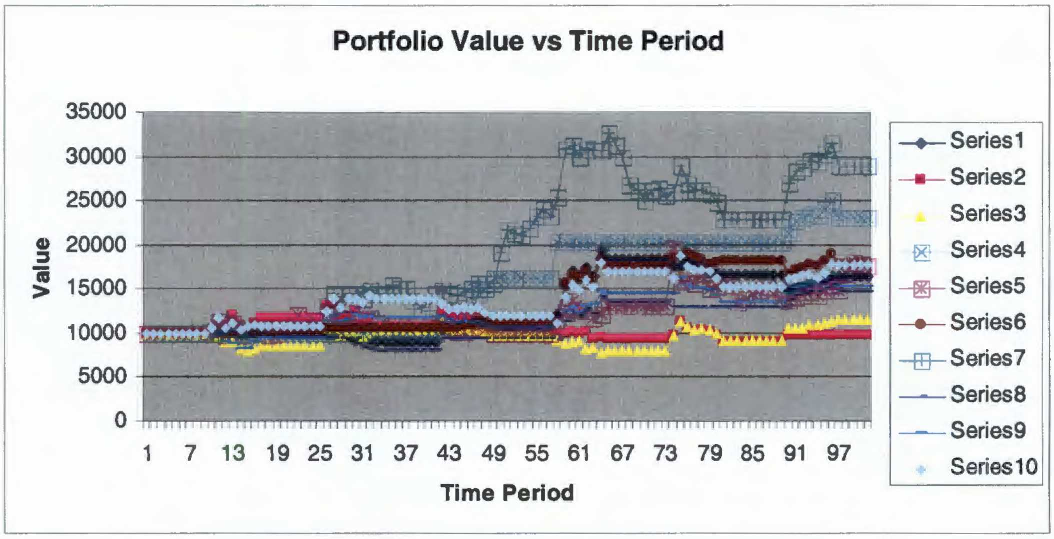

Figure C2: Evolutionary trader portfolio values for test 1

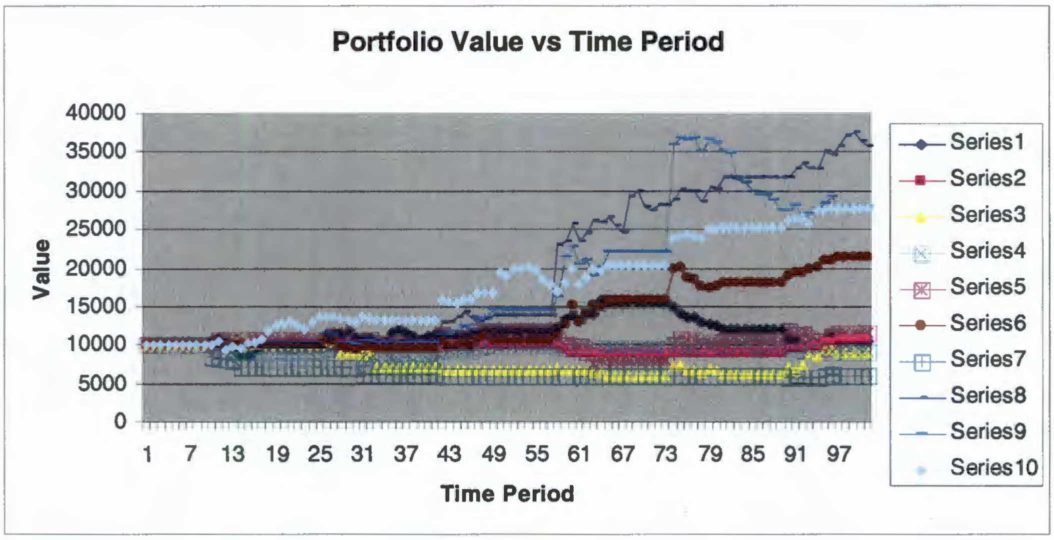

Figure C3: Evolutionary trader portfolio values for test 2

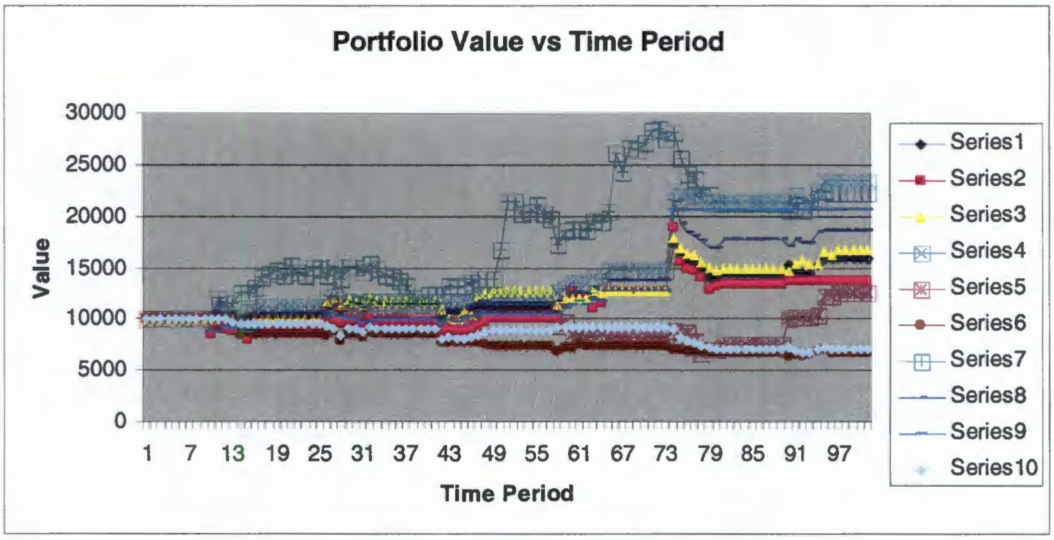

Figure C4: Evolutionary trader portfolio values for test 3

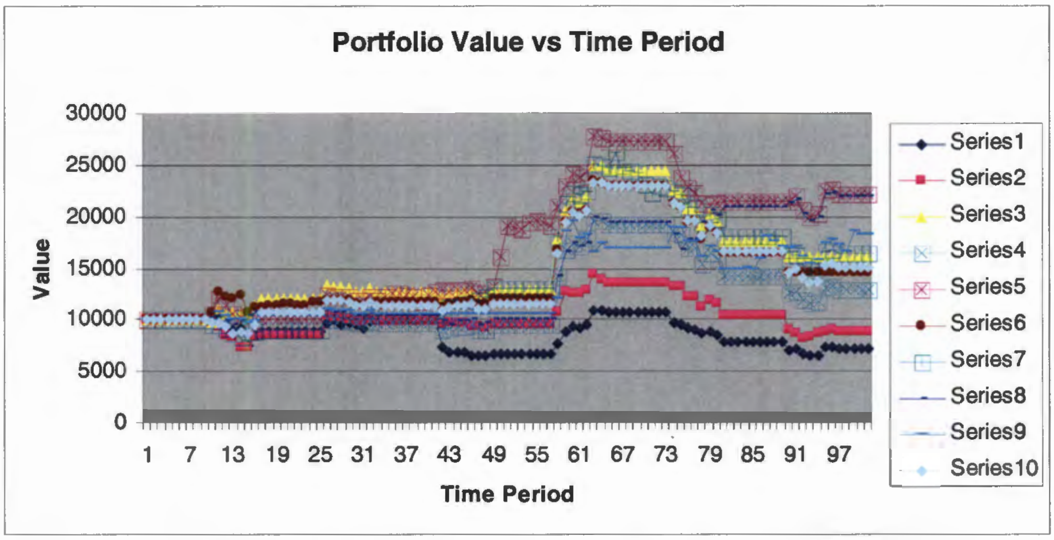

Figure C5: Evolutionary trader portfolio values for test 4

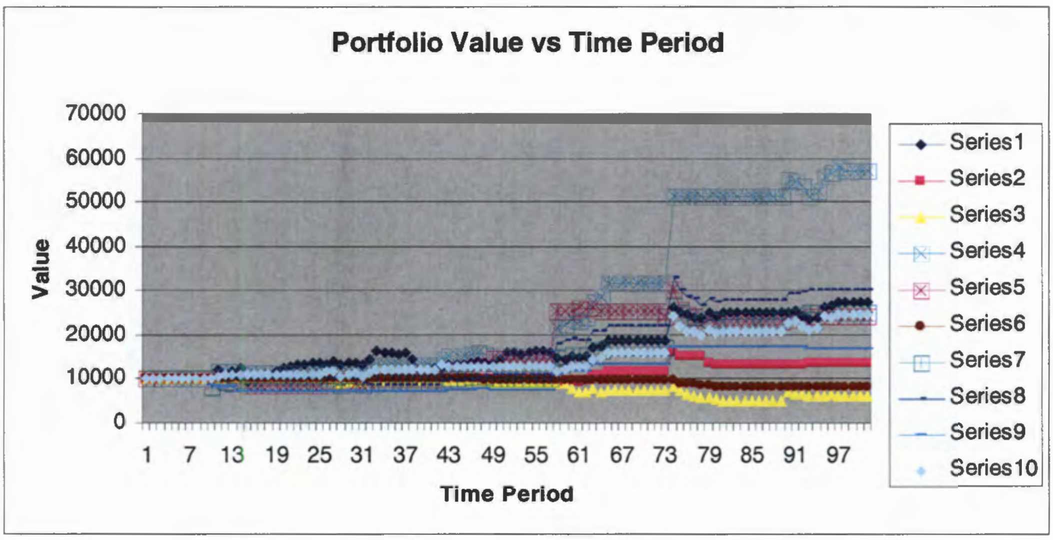

Figure C6: Evolutionary trader portfolio values for test 5

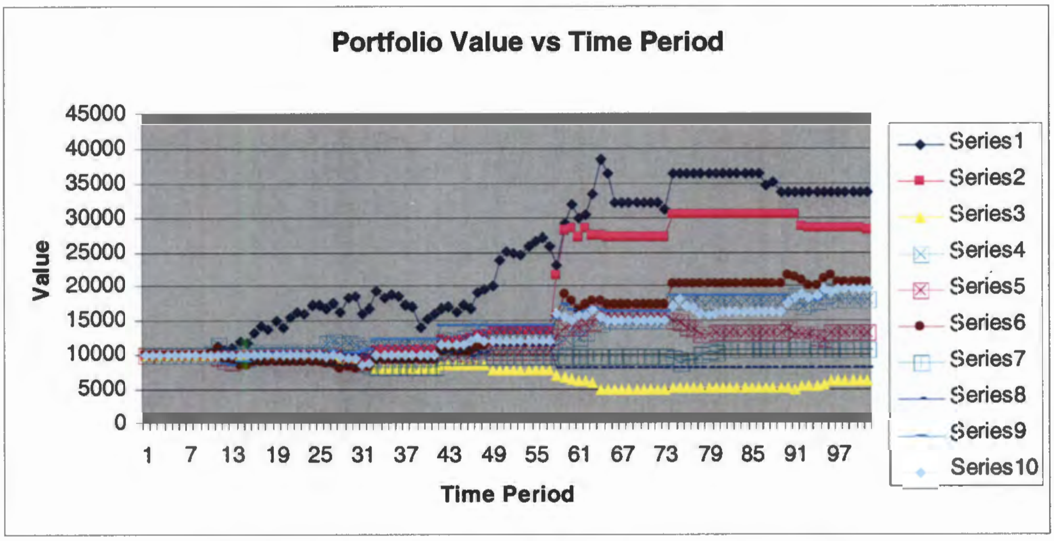

Figure C9: Evolutionary trader portfolio values for test 8

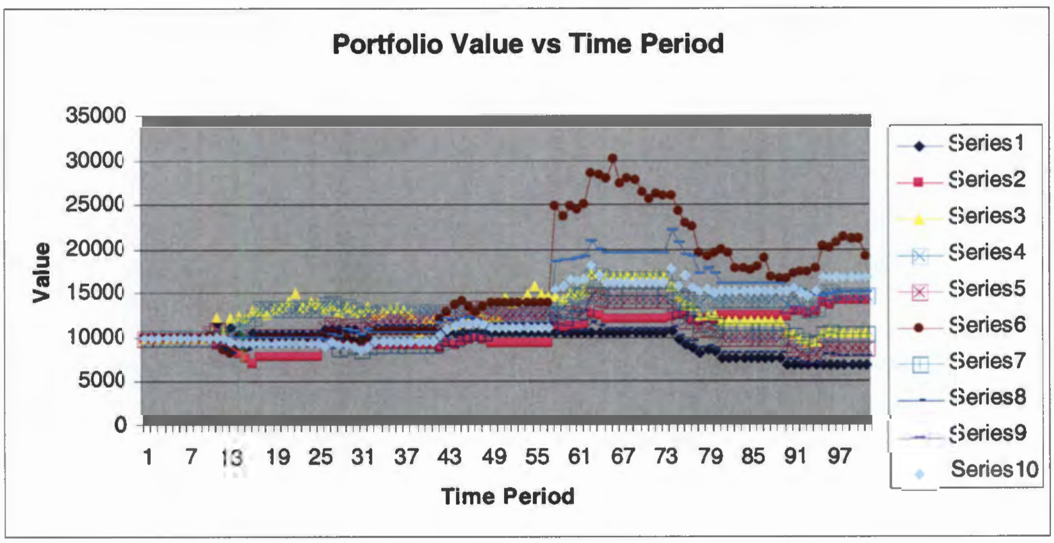

Figure C10: Evolutionary trader portfolio values for test 9

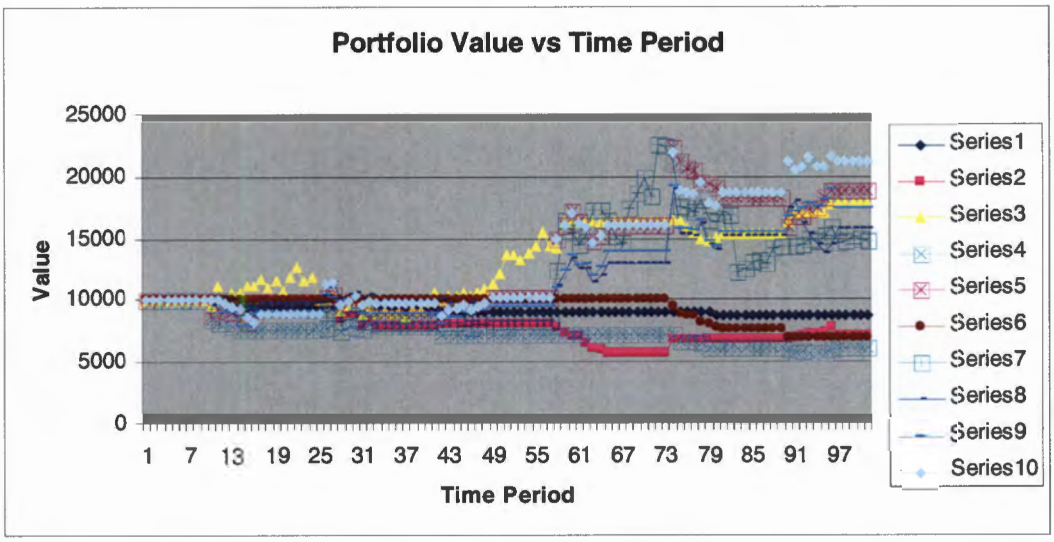

Figure C11: Evolutionary trader portfolio values for test 10

APPENDIX D: PORTFOLIO VALUE TESTS

One-sample t-Test	
Sample Mean	17270.76
Hypothetical Mean	12203.23
Degrees of Freedom	9
t-value	2.92
Critical t-value (10%)	1.383
Critical t-value (5%)	1.833

Table D1: t-Test for returns portfolio value for test 1

One-sample t-Test	
Sample Mean	16966.24
Hypothetical Mean	12203.23
Degrees of Freedom	9
t-value	1.46
Critical t-value (10%)	1.383
Critical t-value (5%)	1.833

Table D2: t-Test for returns portfolio value for test 2

One-sample t-Test	
Sample Mean	15758.31
Hypothetical Mean	12203.23
Degrees of Freedom	9
t-value	1.93
Critical t-value (10%)	1.383
Critical t-value (5%)	1.833

Table D3: t-Test for returns portfolio value for test 3

One-sample t-Test	
Sample Mean	15363.17
Hypothetical Mean	12203.23
Degrees of Freedom	9
t-value	2.05
Critical t-value (10%)	1.383
Critical t-value (5%)	1.833

Table D4: t-Test for returns portfolio value for test 4

One-sample t-Test	
Sample Mean	23337.26
Hypothetical Mean	12203.23
Degrees of Freedom	9
t-value	2.44
Critical t-value (10%)	1.383
Critical t-value (5%)	1.833

Table D5: t-Test for returns portfolio value for test 5

One-sample t-Test	
Sample Mean	24434.72
Hypothetical Mean	12203.23
Degrees of Freedom	9
t-value	3.07
Critical t-value (10%)	1.383
Critical t-value (5%)	1.833

Table D6: t-Test for returns portfolio value for test 6

One-sample t-Test	
Sample Mean	22466.6
Hypothetical Mean	12203.23
Degrees of Freedom	9
t-value	2.56
Critical t-value (10%)	1.383
Critical t-value (5%)	1.833

Table D7: t-Test for returns portfolio value for test 7

One-sample t-Test	
Sample Mean	17837.47
Hypothetical Mean	12203.23
Degrees of Freedom	9
t-value	2.07
Critical t-value (10%)	1.383
Critical t-value (5%)	1.833

Table D8: t-Test for returns portfolio value for test 8

One-sample t-Test	
Sample Mean	12399.85
Hypothetical Mean	12203.23
Degrees of Freedom	9
t-value	0.15
Critical t-value (10%)	1.383
Critical t-value (5%)	1.833

Table D9: t-Test for returns portfolio value for test 9

One-sample t-Test	
Sample Mean	13510.07
Hypothetical Mean	12203.23
Degrees of Freedom	9
t-value	0.73
Critical t-value (10%)	1.383
Critical t-value (5%)	1.833

Table D10: t-Test for returns portfolio value for test 10

APPENDIX E: PORTFOLIO PERFORMANCE RESULTS

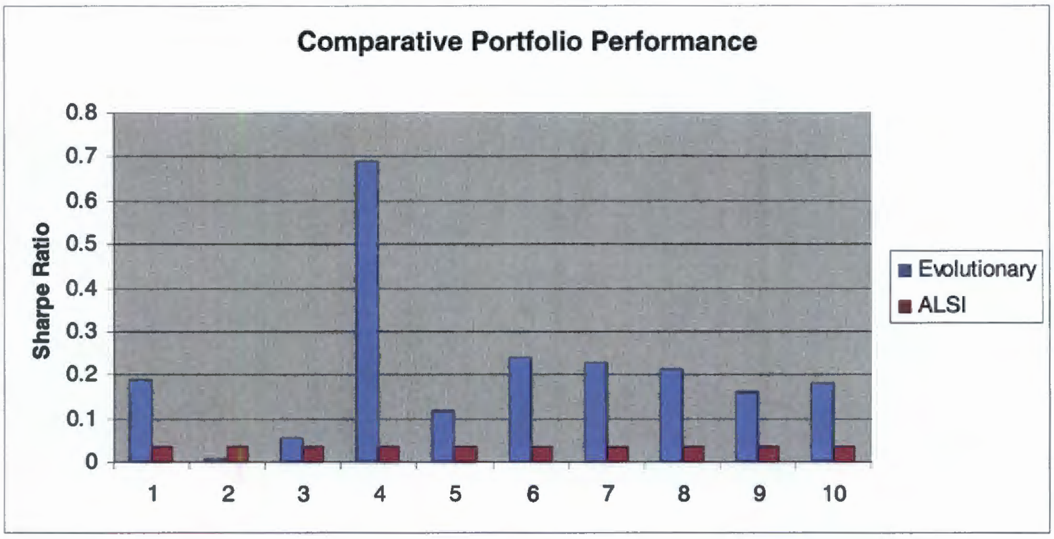

Figure E1: Comparative portfolio performance for test 1

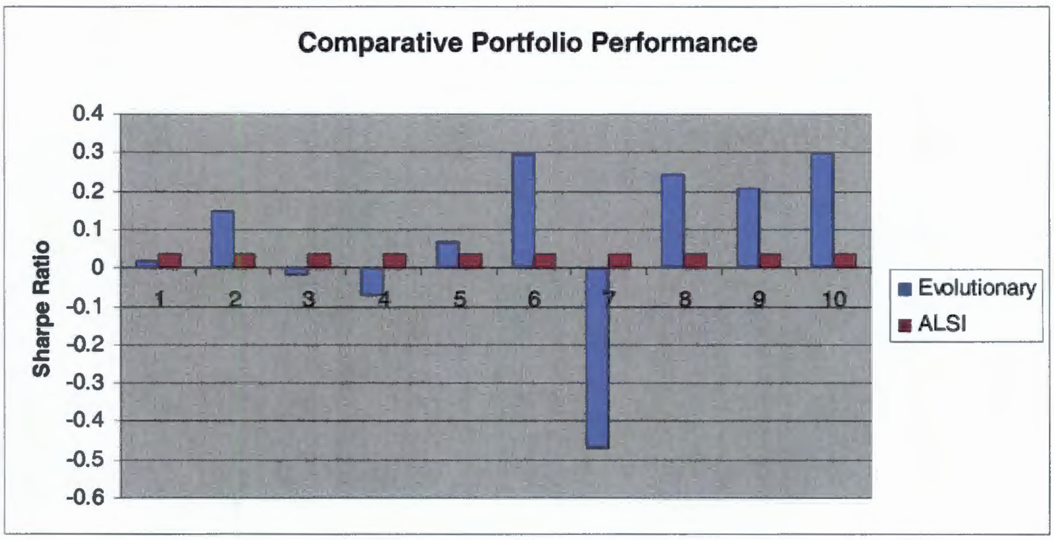

Figure E2: Comparative portfolio performance for test 2

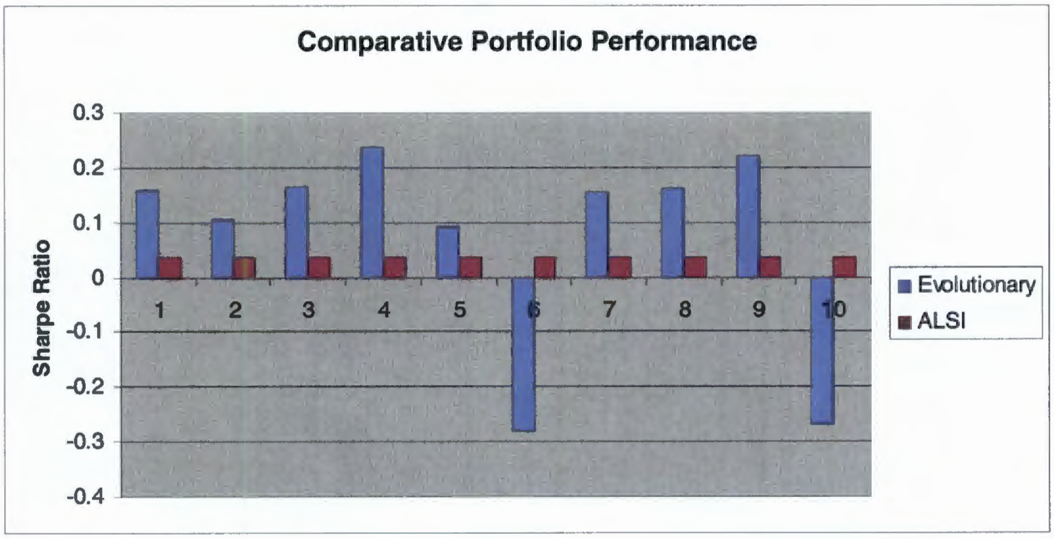

Figure E3: Comparative portfolio performance for test 3

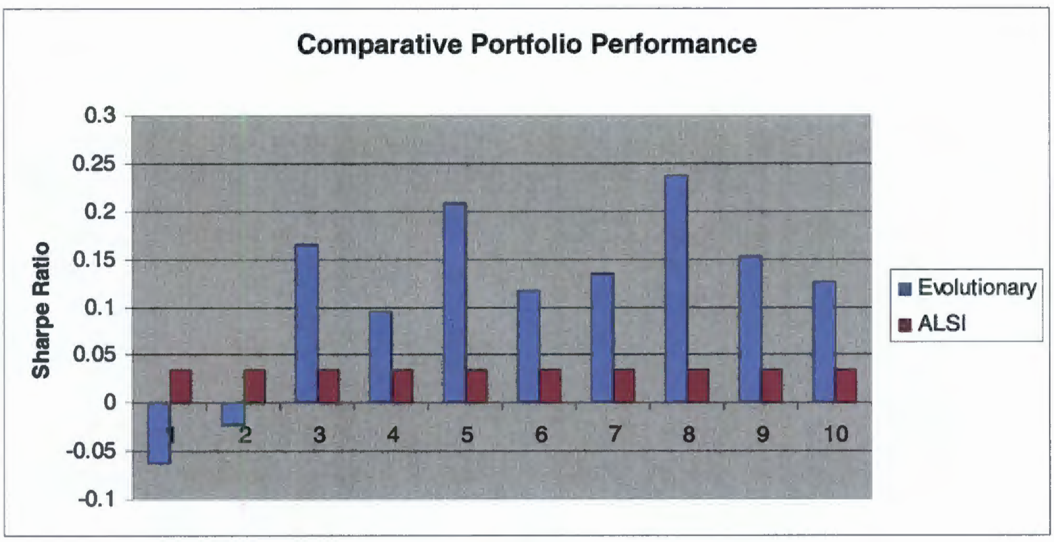

Figure E4: Comparative portfolio performance for test 4

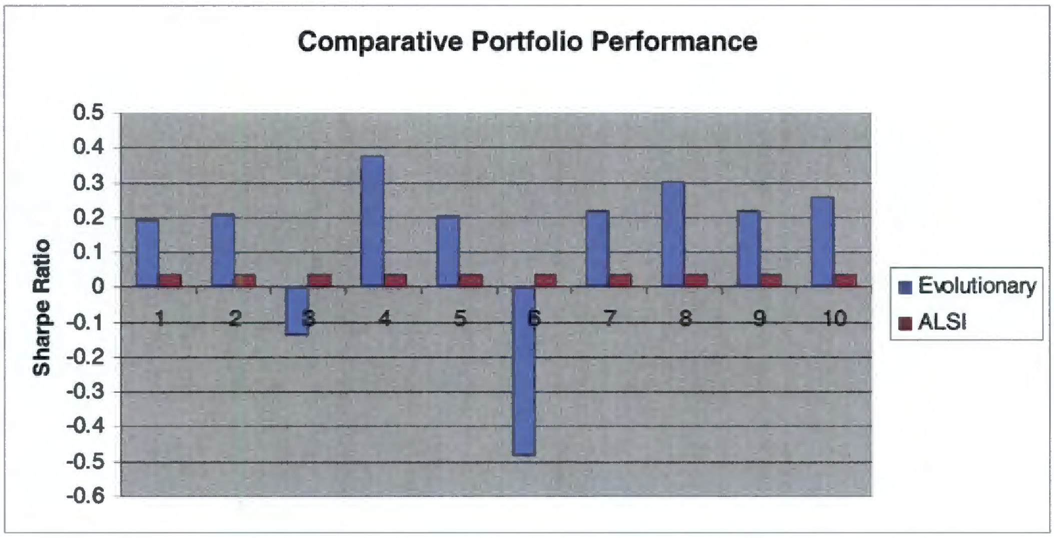

Figure E5: Comparative portfolio performance for test 5

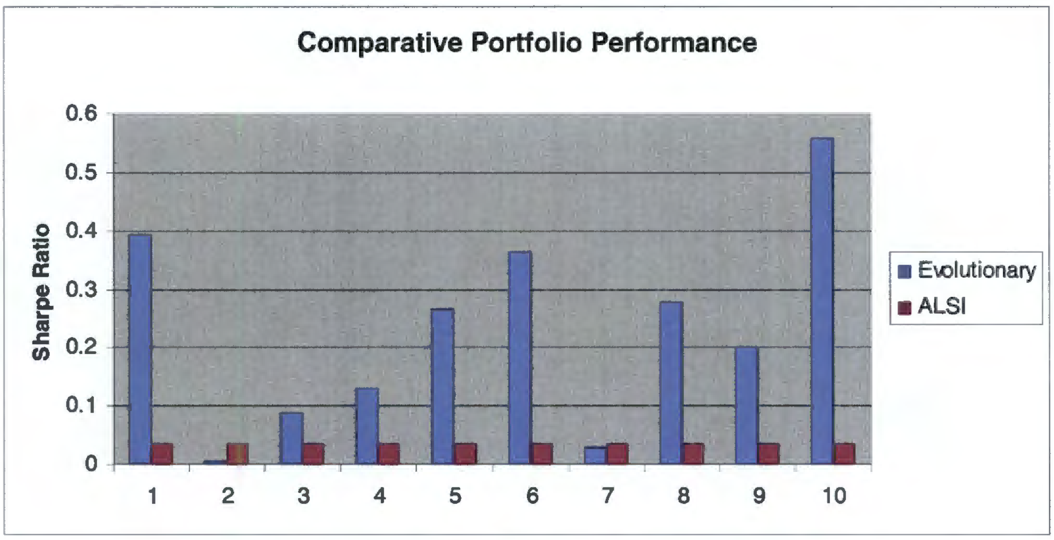

Figure E6: Comparative portfolio performance for test 6

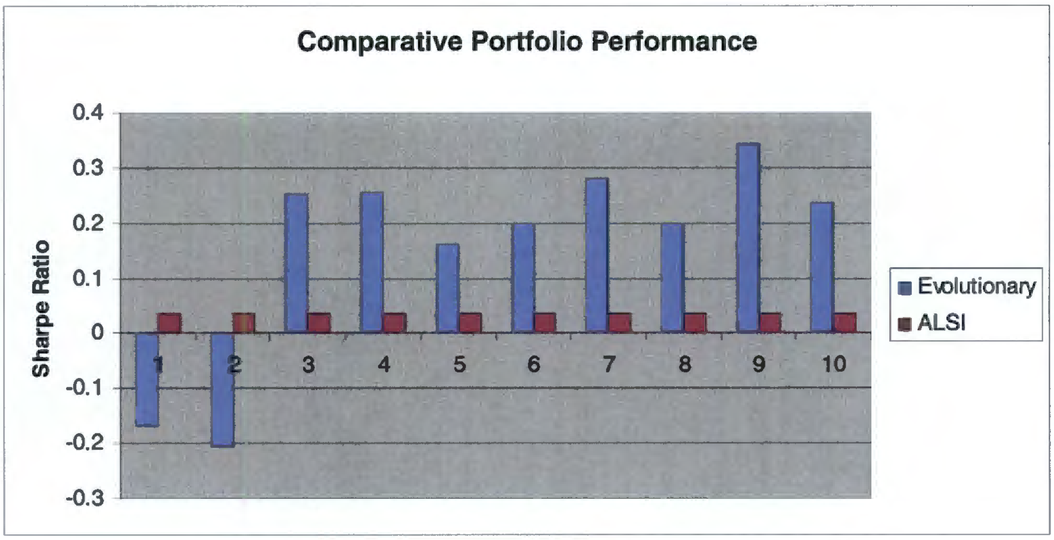

Figure E7: Comparative portfolio performance for test 7

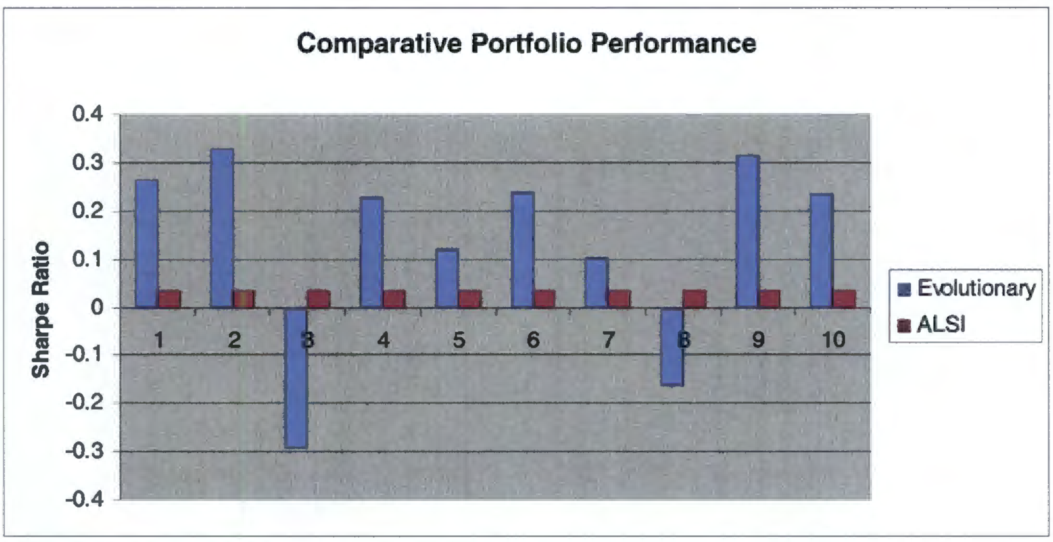

Figure E8: Comparative portfolio performance for test 8

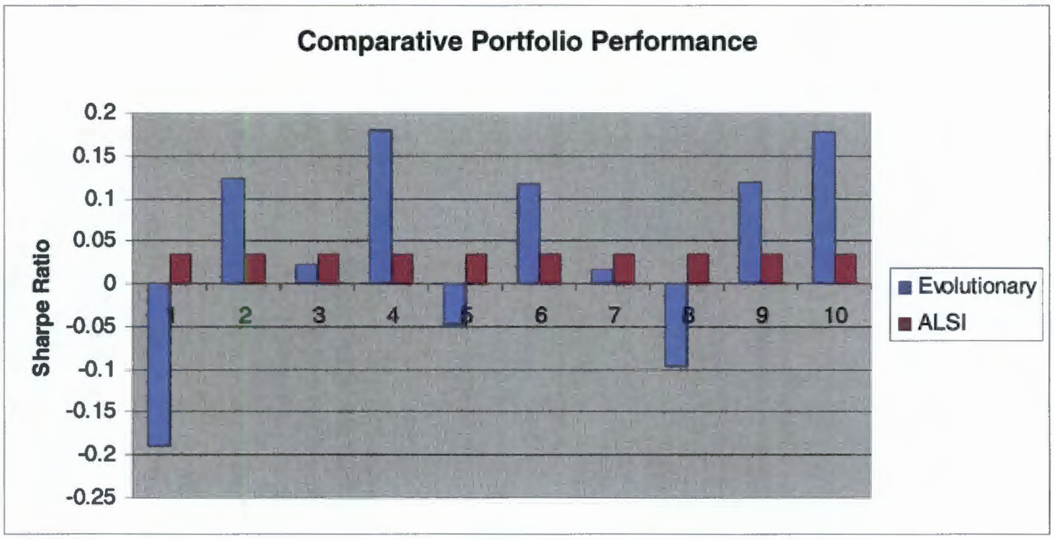

Figure E9: Comparative portfolio performance for test 9

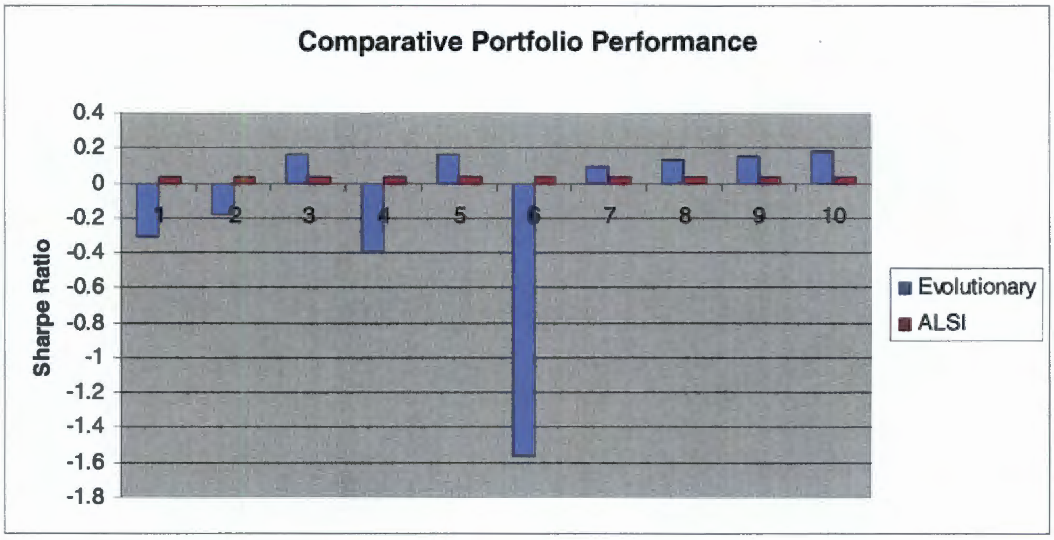

Figure E10: Comparative portfolio performance for test 10

APPENDIX F: PORTFOLIO PERFORMANCE RESULTS

One-sample t-Test	
Sample Mean	0.209
Hypothetical Mean	0.035
Degrees of Freedom	9
t-value	2.97
Critical t-value (10%)	1.38
Critical t-value (5%)	1.83

Table F1: t-Test for portfolio performance for test 1

One-sample t-Test	
Sample Mean	0.072
Hypothetical Mean	0.035
Degrees of Freedom	9
t-value	0.51
Critical t-value (10%)	1.38
Critical t-value (5%)	1.83

Table F2: t-Test for portfolio performance for test 2

One-sample t-Test	
Sample Mean	0.075
Hypothetical Mean	0.035
Degrees of Freedom	9
t-value	0.67
Critical t-value (10%)	1.38
Critical t-value (5%)	1.83

Table F3: t-Test for portfolio performance for test 3

One-sample t-Test	
Sample Mean	0.115
Hypothetical Mean	0.035
Degrees of Freedom	9
t-value	2.71
Critical t-value (10%)	1.38
Critical t-value (5%)	1.83

Table F4: t-Test for portfolio performance for test 4

One-sample t-Test	
Sample Mean	0.136
Hypothetical Mean	0.035
Degrees of Freedom	9
t-value	1.26
Critical t-value (10%)	1.38
Critical t-value (5%)	1.83

Table F5: t-Test for portfolio performance for test 5

One-sample t-Test	
Sample Mean	0.231
Hypothetical Mean	0.035
Degrees of Freedom	9
t-value	3.54
Critical t-value (10%)	1.38
Critical t-value (5%)	1.83

Table F6: t-Test for portfolio performance for test 6

One-sample t-Test	
Sample Mean	0.155
Hypothetical Mean	0.035
Degrees of Freedom	9
t-value	2.03
Critical t-value (10%)	1.38
Critical t-value (5%)	1.83

Table F7: t-Test for portfolio performance for test 7

One-sample t-Test	
Sample Mean	0.138
Hypothetical Mean	0.035
Degrees of Freedom	9
t-value	1.56
Critical t-value (10%)	1.38
Critical t-value (5%)	1.83

Table F8: t-Test for portfolio performance for test 8

One-sample t-Test	
Sample Mean	0.042
Hypothetical Mean	0.035
Degrees of Freedom	9
t-value	0.18
Critical t-value (10%)	1.38
Critical t-value (5%)	1.83

Table F9: t-Test for portfolio performance for test 9

One-sample t-Test	
Sample Mean	-0.156
Hypothetical Mean	0.035
Degrees of Freedom	9
t-value	-1.12
Critical t-value (10%)	1.38
Critical t-value (5%)	1.83

Table F10: t-Test for portfolio performance for test 10

APPENDIX G: COMBINED RULE RETURNS RESULTS

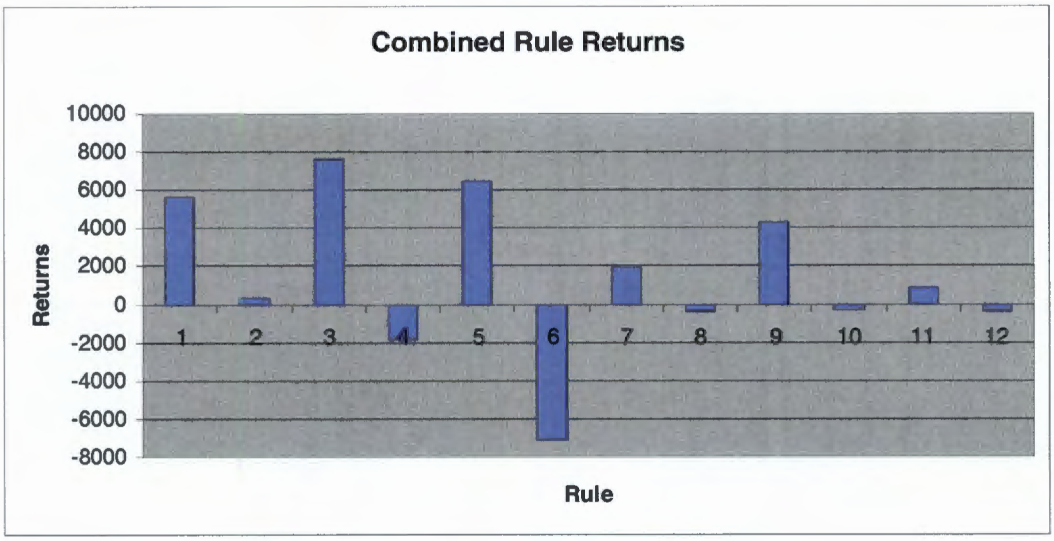

Figure G1: Combine rule returns for test 1

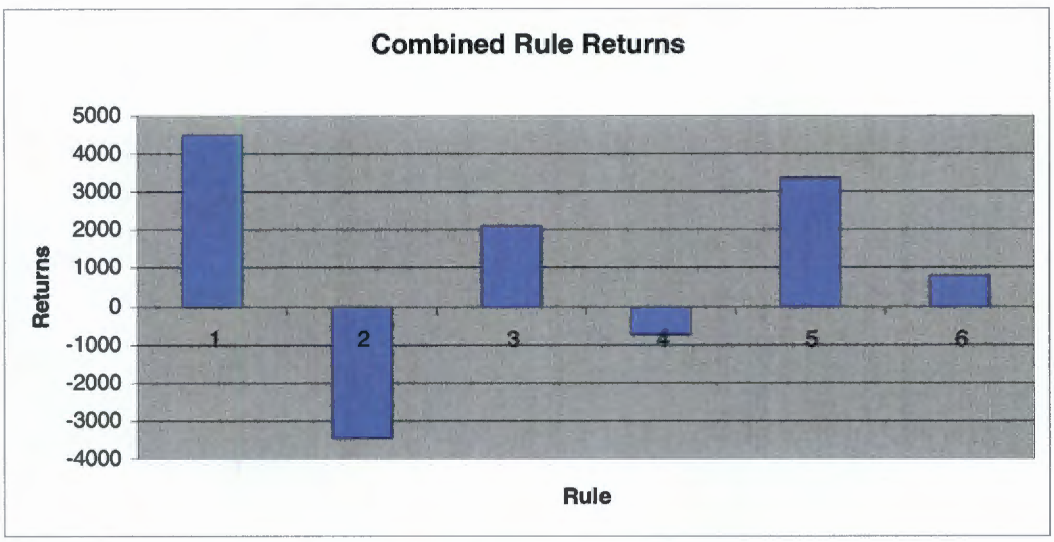

Figure G2: Combine rule returns for test 2

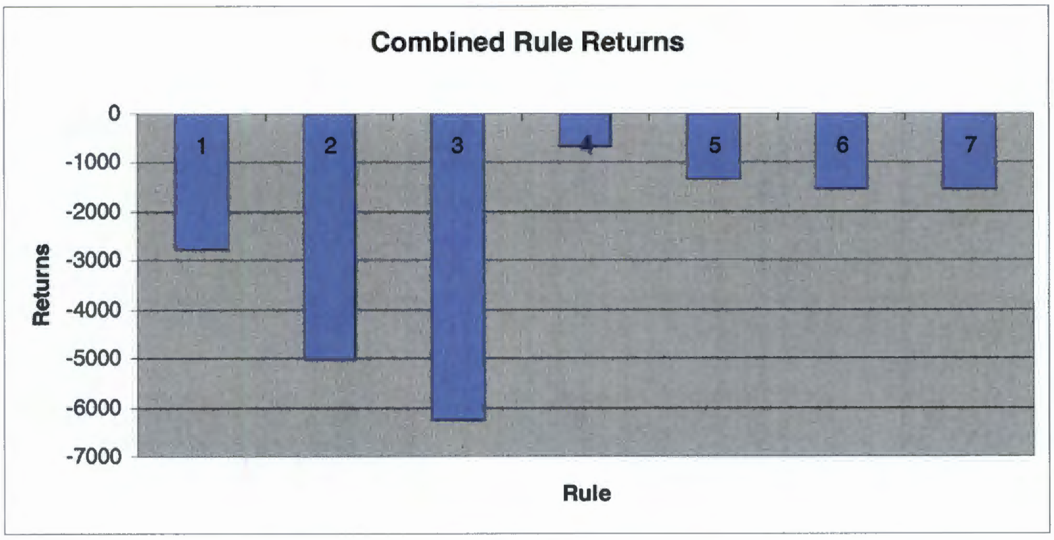

Figure G3: Combine rule returns for test 3

Figure G4: Combine rule returns for test 4

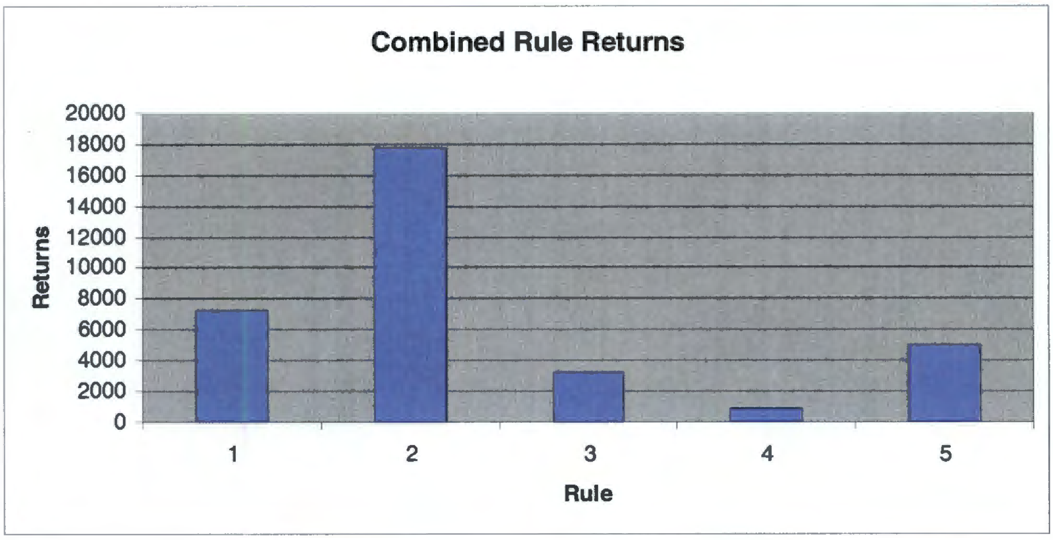

Figure G5: Combine rule returns for test 5

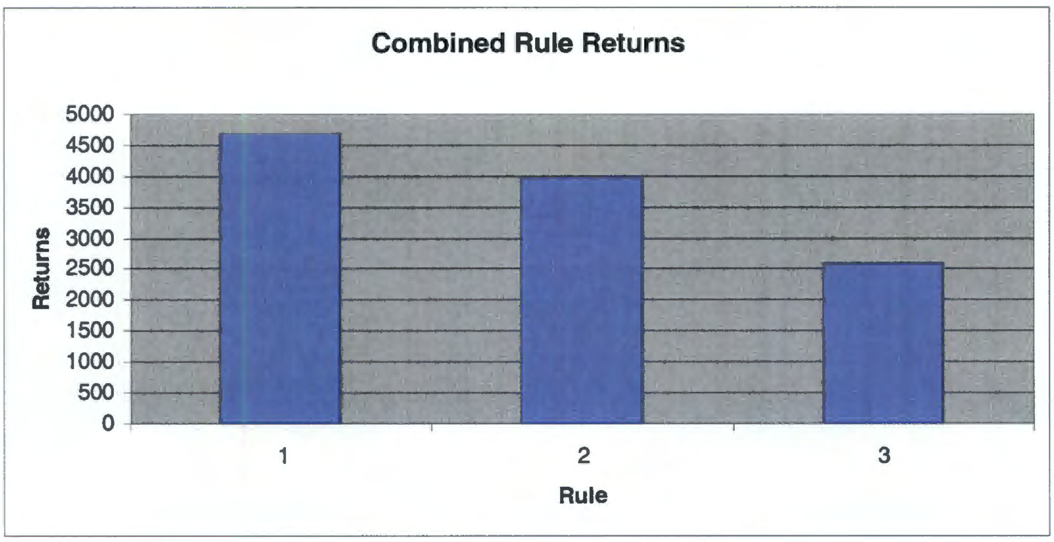

Figure G6: Combine rule returns for test 6

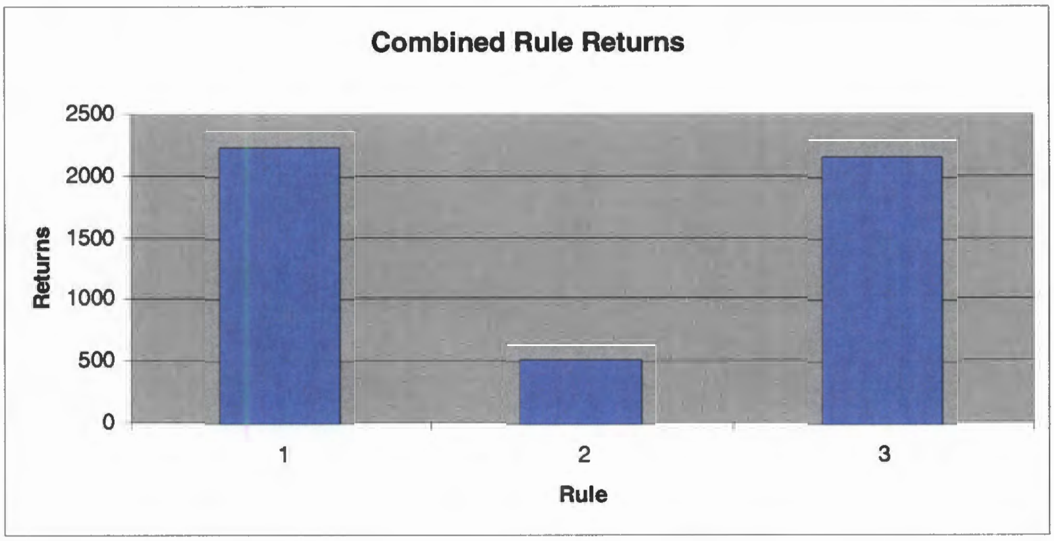

Figure G7: Combine rule returns for test 7

APPENDIX H: COMBINED RULE RETURNS TESTS

One-sample t-Test	
Sample Mean	1441.05
Hypothetical Mean	0
Degrees of Freedom	11
t-Value	1.22
Critical t-value (10%)	1.36
Critical t-value (5%)	1.80

Table H1: t-Test for combined rule returns for test 1

One-sample t-Test	
Sample Mean	1101.46
Hypothetical Mean	0
Degrees of Freedom	5
t-Value	0.93
Critical t-value (10%)	1.48
Critical t-value (5%)	2.02

Table H2: t-Test for combined rule returns for test 2

One-sample t-Test	
Sample Mean	-2938.00
Hypothetical Mean	0
Degrees of Freedom	5
t-Value	-3.21
Critical t-value (10%)	1.48
Critical t-value (5%)	2.02

Table H3: t-Test for combined rule returns for test 3

One-sample t-Test	
Sample Mean	4690.03
Hypothetical Mean	0
Degrees of Freedom	5
t-Value	2.99
Critical t-value (10%)	1.48
Critical t-value (5%)	2.02

Table H4: t-Test for combined rule returns for test 4

One-sample t-Test	
Sample Mean	6838.42
Hypothetical Mean	0
Degrees of Freedom	4
t-Value	2.35
Critical t-value (10%)	1.53
Critical t-value (5%)	2.13

Table H5: t-Test for combined rule returns for test 5

One-sample t-Test	
Sample Mean	3760.63
Hypothetical Mean	0
Degrees of Freedom	2
t-Value	6.13
Critical t-value (10%)	1.89
Critical t-value (5%)	2.92

Table H6: t-Test for combined rule returns for test 6

One-sample t-Test	
Sample Mean	1638.30
Hypothetical Mean	0
Degrees of Freedom	2
t-Value	2.93
Critical t-value (10%)	1.89
Critical t-value (5%)	2.92

Table H7: t-Test for combined rule returns for test 7