Grammar-Mediated Time-Series Prediction

Anthony Brabazon[1], Katrina Meagher[1], Edward Carty[1], Michael O'Neill[2], and Peter Keenan[1]

[1] Faculty of Commerce
University College Dublin, Ireland
[2] Biocomputing and Developmental Systems Group
University of Limerick, Ireland

Abstract. Grammatical Evolution is a data-driven, model-induction tool, inspired by the biological gene-to-protein mapping process. This study examines the potential of Grammatical Evolution to uncover useful technical trading rule-sets for intra-day equity trading. The form of these rule-sets is not specified *ex-ante*, but emerges by means of an evolutionary process. High-frequency price data drawn from US stock markets is used to train and test the model. The findings suggest that the developed rules earn positive returns in hold-out test periods, and that the size of these returns are critically impacted by the choice of position exit-strategy.

Keywords: Grammatical evolution, High-frequency finance, Intra-day stock trading.

1 Introduction

Grammatical Evolution (GE) [21, 19, 20, 23] is an evolutionary automatic programming methodology which can be used to evolve rule-sets. These rule-sets can be as general as a functional expression which produces a good mapping between a series of known input-output data vectors. A particular strength of the methodology is that the structure of the functional expression need not be specified *a priori* by the modeler. This is of particular utility in cases where the modeler has a weak understanding of the functional relationship between the explanatory and the dependent variables. A key element of the methodology is the concept of a *Grammar*, which governs the creation of the rule-sets. This paper applies a GE methodology to evolve intra-day trading systems for equities, based on technical indicators.

1.1 Motivation for study

As noted by [12], there are a number of reasons to suppose that the use of an evolutionary automatic programming (EAP) approach such as GE can prove fruitful in the financial prediction domain. EAP can conduct an efficient exploration of a search space, and can uncover dependencies between variables leading to the selection of a good subset for inclusion in the final model. The use of an EAP approach facilitates the implementation of complex fitness functions, including those which are discontinuous or non-differentiable. This is of particular importance in a financial domain where fitness criterion can be complex, usually requiring a balancing of risk and return.

Despite the potential for EAP for the construction of useful financial trading systems, there have been relatively few studies applying Genetic Programming (GP) [13] or GE for this purpose. Notable exceptions are [1, 3, 16, 18, 24, 25]. Most prior work uses financial data sampled on a daily basis, and adopts simple exit strategies for closing out trading positions.

In contrast, this study utilizes high-frequency time-series data, sampled at five-minute intervals during each trading day. This permits the construction of intra-day trading system. This study also examines the affect on trading system performance of adopting three distinct trade-exit strategies.

The rest of this paper is organized as follows. The next section provides a short discussion of the literature on technical analysis and discusses the construction of a trading system using technical indicators. This is followed by a section which describes Grammatical Evolution. Then we outline the data set and methodology utilized. Finally, we provide the results of the study, followed by a number of conclusions.

2 Technical Analysis

Technical analysis is defined as the attempt to identify regularities in the time-series of price and volume information from a financial market, by extracting patterns from noisy data [14]. Some market traders, known as technical analysts, believe that prices move in trends and that price patterns repeat themselves [15]. Technicians argue that trends are more likely to continue than to reverse, giving rise to market lore such as *'the trend is your friend'* and *'never buck the trend'* [15] (p. 49). The cornerstone of the technical philosophy is that all of the relevant data that an analyst needs to forecast market direction for a financial asset, is recorded on its price chart. Technical analysts believe that market price, which results from the forces of supply and demand for a financial asset, discounts (or reflects) all of the information that can affect the price of a financial asset

[15]. Technical analysis studies the relative strength of these forces, as evidenced by share price movements, to gain insight into the likely future trading range and direction of price movement for a financial asset.

There is much debate amongst financial theorists as to whether technical analysis can actually provide information on the future direction of price movement a financial asset. Research in a variety of financial markets has provided at least tentative support for the potential effectiveness of technical analysis. A number of studies have suggested that it may indeed be possible to uncover patterns of predictability in price behavior. [4] found that simple technical trading rules had predictive power and suggested that the conclusions of earlier studies that technical trading rules did not have such power were *premature*. Other studies which indicated that there may be predictable patterns in share price movements include those which suggest that markets do not always impound new information instantaneously [6, 11, 7], and studies which suggest that stock markets can overreact as a result of excessive investor optimism or pessimism [10, 9]. Although controversy exists amongst financial theorists regarding the veracity of the claims of technical analysts, the methods are widely applied in practice.

2.1 Technical indicators

The development of trading rules based on current and historic market price / volume information has a long history [5]. The process entails the selection of one or more technical indicators and the development of a trading system based on these indicators. These indicators are formed from various combinations of current and historic price and trading volume information, and in essence the indicators serve to pre-process or transform the underlying time-series. Although there are an infinite number of such indicators which could be calculated, the financial literature [4, 15, 22] suggests that certain indicators are widely used by investors.

The objective of trading systems developed using technical indicators is to identify the current pervasive trend and to trade in that direction [15]. If the objective is to trend-follow, identification of trend reversals is clearly important. Concepts which are relevant to this task include those of *support* and *resistance*. A zone of support arises at prices where there is a concentration of demand, a zone of resistance arises when there is a concentration of supply. Technical analysts suggest that down-trends in a price series tend to reverse at zones of support, whereas up-trends tend to reverse at resistance zones. If these zones are breached (called *a breakout*), perhaps because of a significant new information flow to the market, then the trend in price movement accelerates and new support and resistance levels are established [15]. Four groupings of indicators are given prominence in the technical analysis literature:

i. Moving average indicators
ii. Momentum indicators
iii. Trading range indicators
iv. Oscillators

In this paper, we have limited our attention to the first three groupings of indicators. The simplest moving average systems compare the current price of a share with a moving average of the share price over a lagged period, to determine how far the current price has moved from an underlying trend. As they smooth out daily price fluctuations, moving averages can heighten the visibility of an underlying trend. A trading signal can be generated when a share price moves above or below its moving average. A variation on simple moving average systems is to use a moving average convergence-divergence (MACD) oscillator. A simple version of this indicator is to generate a trading signal when a short run and a long run moving average cross. In a recursive fashion, more complex combinations of moving averages of values calculated from a MACD oscillator can themselves be used to generate trading rules. The momentum of share price is the ratio of a time-lagged price to the current price $\left(\frac{Price_t}{Price_{t-x}}\right)$. The belief underlying this indicator is that a strong tend is likely to persist for a period of time. Trading Range Breakout indicators seek to determine when the price of a share moves out of a range defined by its maximum or minimum value in a lagged time-period. A simple example of a trading rule derived from a breakout indicator would be to buy a share when it exceeds its previous high in the last four weeks, and conversely to sell if it falls below its previous four week low.

2.2 Using Technical Indicators in a Trading System

Technical indicators can be incorporated as an input in a trading model in two ways. Individual indicators can be used directly as model inputs, or alternatively, individual indicators can be preprocessed to produce a model input, for example by using the ratio of two individual indicators, or through the use of IF-THEN statements. As an example of the latter, a 0 or 1 could be output from a compound 'IF-THEN' rule, such as IF (indicator $a > 2$) AND (indicator $b < 4$) THEN 'trading signal' is buy / sell. A trader who wishes to construct a trading system using technical indicators as inputs faces several decisions:

i. Which indicators will be used?
ii. What parameter values (lag periods / trigger values) should be used?
iii. How should the indicators be combined to produce a trading signal?

This presents a combinatorial problem. The huge number of possibilities open to the modeler suggests that an evolutionary automatic programming methodology such as GE, in which model structure and inputs are not fixed *a-priori*, will have particular potential.

2.3 High-frequency data

High-frequency financial data is data which is sampled at small time intervals (or at high-frequency) during the trading day. Therefore, high-frequency financial data can contain the price and volume history of a financial asset for every minute of trading (or in the limit, every single transaction) during the day. Virtually all studies of the utility of technical analysis use low-frequency time-series, typically end-of-day price and volume information. There are approximately 250 trading days per year, so using a daily sampling period results in a significant reduction in the density of data. For example, a heavily-traded share such as Microsoft may generate 20 million transactions annually. Using end-of-day data for

Microsoft implies that each data point represents on average, 80,000 transactions. Substantial volumes of trading in financial markets are intra-day. In the case of foreign-exchange markets, [8] estimate that approximately 90% of all trading is accounted for by intra-day traders. Studies of technical analysis using daily price data cannot provide insight into the possible utility of technical analysis for intra-day trading. In this study we are interested in intra-day trading, hence we utilize high-frequency financial data.

3 Grammatical Evolution

This section provides a short description of Grammatical Evolution (GE), a more detailed description of GE can be found in [21, 20, 19] or [23]. GE implements a population-based search for 'good' rule-sets for the problem of interest, in this study trading systems. The population of solutions is modified from one iteration of the algorithm to the next, biasing the adaptation process towards good solutions in the current population. In GE, as in biology, a strong distinction is drawn between the genotype and the phenotype. The latter corresponds to the created rule-set (trading system), and the former corresponds to a 'genetic encoding' of this system. The core of the GE system is the genotype-phenotype mapping process used to generate the rule-set. Each genotype, represented as a variable length binary string, contains in its codons (groups of bits on the binary string) the information to select rule-set (trading system) production rules from a Backus Naur Form (BNF) grammar. BNF is a notation that represents a language in the form of production rules. It is comprised of a set of non-terminals that can be mapped to elements of a set of terminals, according to the production rules. An overview of the mapping process in GE is provided in Figure 3. Each member of the population is represented as a binary string, comprised of multiple codons, each of which are translated into an integer value. These integers are mapped onto trading rules, using a mediating BNF Grammar.

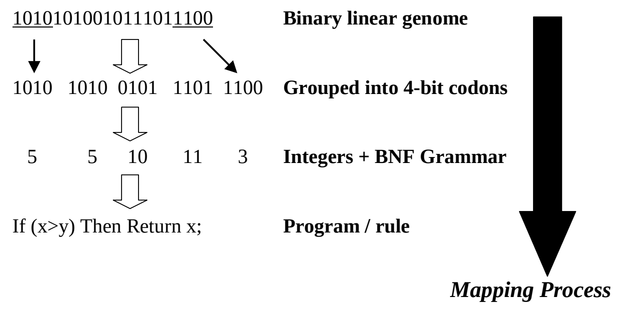


Fig. 1. Binary string to Rule Mapping of GE.

To demonstrate the mapping of integers to trading rules, a example excerpt from a simple BNF Grammar is given below. These productions state that S can be replaced with either one of the non-terminals $ma(< int >, day)$, $momentum(< int >, day)$, or $trb(< int >, day)$.

```
S ::=
 | ma( <int> , day )          (0)
 | momentum( <int> , day)     (1)
 | trb( <int> , day)          (2)
```

The Grammar is used in a generative process to construct a program by applying production rules, selected by the genome, beginning from the start symbol of the Grammar. In order to select a rule in GE, the integer corresponding to the next codon value on the genome is obtained and placed in the following formula:

$$ChosenRule = Codon\ Value\ MOD\ Num.\ Rules$$

In the above example, given that we have three rules to select from, if the next codon integer value was 4, we get $4\ MOD\ 3\ =\ 1$. S will therefore be replaced with the non-terminal $momentum(< int >, day)$. Beginning from the left hand side of the genome, codon integer values are generated and used to select rules from the BNF Grammar, until one of the following situations arise:

i. A complete program is generated. This occurs when all the non-terminals in the expression being mapped, are transformed into elements from the terminal set of the BNF Grammar.

ii. The end of the genome is reached, in which case a *wrapping* operator is invoked. This results in the return of the genome reading frame to the left hand side of the genome. The reading of codons continues unless an upper threshold representing the maximum number of wrapping events has occurred during this individual's mapping process.

iii. In the event that a threshold on the number of wrapping events is exceeded and the individual is still incompletely mapped, the mapping process is halted, and the individual assigned the lowest possible fitness value.

After each individual genotype is mapped to a trading rule or when the mapping fails because the wrapping threshold is exceeded, the profitability of each rule is assessed, and this measure of fitness drives a genetic algorithm search engine, operating on the population of binary genotypes in an effort to uncover ever better trading rules in successive iterations of the GE algorithm. A critical feature of GE is that the search and solution spaces are separated, through the genotype-phenotype mapping, and this allows GE to generate output in an arbitrary language according to the specification in the BNF grammar. Therefore, GE can be used to generate any form of program / rule-set as defined in the Grammar.

The GE system is summarized in Figure 3. A GE system requires the selection by the modeler of a search engine, a suitable domain Grammar, and a domain-specific objective function which is used to assess the quality of the evolved rule-sets / programs (the output). In this study the Grammar includes pre-defined technical indicators, the objective function is a measure of the return generated by a trading rule and the search engine operating on the population of binary strings is a genetic algorithm. Each of these elements is modular, can can be altered at will by the modeler. For example, any search algorithm with the ability to operate over binary or integer strings can be used in place of the genetic algorithm as the search engine [21, 17].

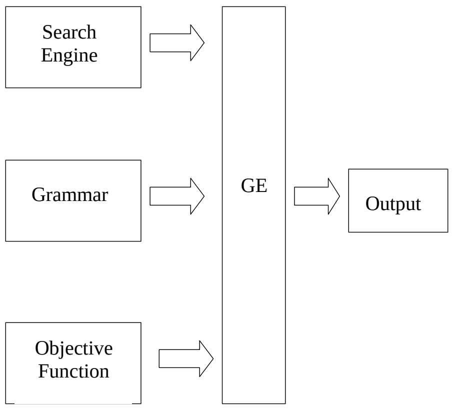


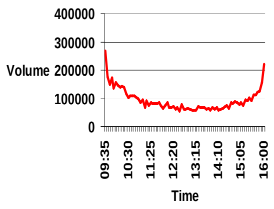
Fig. 2. Summary of GE system, the search engine, the grammar, and the objective function.

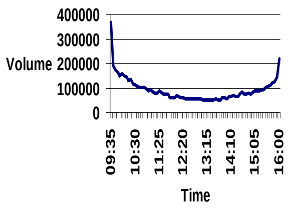
4 Experimental Approach

This study uses high-frequency price data for Ford and IBM, both of which are quoted on the NYSE. The trading systems were developed using data for the period 1/2/02 to 4/15/02, and were tested out-of-sample using data from 4/16/02 to 7/2/02. Data is sampled at five-minute intervals from these periods, producing 9,828 data vectors for each stock. Each data vector included the opening and closing prices, the high and low price and volume traded, for each five-minute interval. The average of the open and closing price of each interval is used as the input data for the trading system. The price data is normalized into the range [0,1].

It is well-known that high-frequency financial data exhibits intra-day seasonality. In the case of equity and bond markets which are open for fixed trading hours, activity measures display a distorted U shape over the trading day [2]. Typically, intra-tick duration (the time between one transaction and the next) is lowest in the opening minutes of daily trading, is highest during lunch hour, and decreases again towards the end of the trading day. Figure 3 provides a graph of the average trading volume of Ford and IBM for each five-minute interval during the period January-July 2002. Prices tend to be most volatile in early and late trading, due to the queueing of pre-opening trades in the morning, and the activities of traders who do (do not) want to carry inventory of a stock overnight in the case of late trading. In developing our trading system, we do not trade in either the first or last half hour each day. All open positions are closed out before the last half hour of each day, resulting in the system not holding any positions overnight. This reduces the price risk to which the system is exposed, as new information that is brought to the market pre-opening the next day, could affect the next morning's opening price causing it to 'gap' upwards or downwards from the closing price of the previous evening.

Fig. 3. Average intra-day trading volume for Ford (left) and IBM (right) at five-minute intervals, Jan-July 2002.





4.1 Trading system

After the initial half hour period each day, the trading system considers whether or not to trade at the end of every 10-minute interval during the day. At each possible trading time the system calculates its prediction. The prediction is calculated by evaluating the evolved trading rule. The rule returns a value in the range 0 to 1, and this value is post-processed using:

$$Sell = Value < .33$$

$$Do - Nothing = .33 >= Value < .66$$

$$Buy = .66 >= Value$$

Permitting the model to output a 'Do-Nothing' signal reduces the hard threshold problem associated with production of a binary output. Markets do not invariably trend upwards or downwards, but may trade sideways in a relatively narrow band of prices, and trend-following trading systems should be allowed to 'opt-out' of trading these markets for which they are not well-suited. A variant on the trading methodology developed in [4] is then applied.

If the prediction is to go 'Long' the system will buy \$1,000 of the stock, if it is to go 'Short' the system will sell \$1,000 of the stock. When the trade is closed out a profit or loss is evaluated, and a cumulative total of the profits or losses of the trading rule is maintained. The maximum amount that the system can have invested at any one time at \$10,000. If the total trading capital is invested at any time, no further positions are open until pre-existing positions are closed.

Once a trading position is open, a variety of *exit strategies* could be employed to decide when this position is to be closed out. In order to examine the significance of the choice of exit strategy on the results obtained by a trading system, this study evolves trading systems which use three different exit strategies, 'standard close', 'extended close' and 'stop-loss, take-profit close'.

The standard and extended close strategies are examined for both stocks, and the stop-loss, take-profit close strategy is examined for Ford. In the simplest system, the *standard close*, the evolved systems automatically close out all trading positions 30 minutes after they are opened. In the *extended close*, the system rechecks after 30 minutes whether the prediction is unchanged from the initial prediction and if it is, the trade is extended for a further 30 minutes. In the *stop-loss, take-profit* close, the position is initially held for 30 minutes, and thereafter, if the position generates a loss $\geq 0.1\%$ it is closed immediately. Any position which makes a profit of $\geq 0.8\%$ is also closed automatically (a take-profit trigger). If any position is still open 30 minutes from the end of the trading day, it is closed out.

4.2 Parameter choices for GE algorithm

In this study, the GE algorithm uses a steady-state replacement mechanism such that two parents produce two children, the best of which replaces the worst individual in the current population, if the child has greater fitness. The standard genetic operators of bit mutation (probability of 0.01), and crossover (probability of 0.9) are adopted in the genetic algorithm search engine. At the end of each run, the best individual is stored, along with that individual's trading rule, fitness and drawdown. The Grammar adopted is defined as follows:

```
N={<code>,<expr>,<fopbi>,<fopun>,<matbi>, <relbi>,<var>,<int>}
T={p,=,(,),f_and,f_or,f_not,+,-,*,>,
<,>=,<=,scale,ma,day,1,2,3,4,5,10}
S=<code> P={ <code> ::=  p
=<expr> ;<expr>::=<fopbi>(<expr>,<expr>)|<fopun>(<expr>)|
<expr><matbi><expr>|<expr><relbi><expr>|<var>

<fopbi> ::= f_and | f_or
<fopun> ::= f_not <matbi> ::= + | - | *
<relbi> ::= > | < | >= | <=

<var> ::=<int>|day|ma(<int>,day)|momentum(<int>,day)
|trb(<int>,day) <int> ::= 1 | 2 | 3 | 4 | 5 | 10 }
```

In addition to the technical indicators the grammar also allows the use of the binary operators `f_and`, `f_or`, the standard arithmetic operators, and the unary operator `f_not`, and the current day's index value `day`. The operations `f_and`, `f_or`, and `f_not` return the minimum, maximum, of the arguments, and 1 - the argument, respectively. A series of functions, in this study, technical indicators, are pre-defined as are a series of mathematical operators. A population of initial trading rule-sets (programs) are randomly generated, and by means of an evolutionary process, these are improved. No explicit model specification is assumed *ex-ante*, although the choice of mathematical operators defined in the Grammar do place implicit limitations on the model specifications amongst which GE can search.

5 Results

The results from our experiments are now provided for both the training period (Table 1) and the out-of-sample test period (Table 2). Neither stock displayed a distinct monotonic price trend during the train or test periods (see Figure 4 for a graph of Ford's share price over the train/test period).

Table 1. Trading profit in $ (maximum drawdown) during Training Period

	Ford	IBM
Standard Close	1,280.73 (1.02)	1,221.67 (12.08)
Extended Close	2,436.67 (29.52)	2,431.51 (68.29)
Stop-Loss	1,965.07 (1.01)	n/a
Buy-and-hold	96.92	-2,890.28

Table 2. Trading profit in $ (maximum drawdown) during Test Period

	Ford	IBM
Standard Close	823.86 (14.47)	892.50 (20.42)
Extended Close	1,257.71 (61.96)	1,761.33 (2.21)
Stop-Loss	1,291.10 (14.47)	n/a
Buy-and-hold	286.55	-1,905.94

In both the training and test periods, the extended close exit strategy notably outperforms the standard close strategy, without exhibiting significantly higher drawdowns (maximum cumulative loss since commencing trading with the system). This result highlights the impact that the choice of exit strategy can have on the results produced by a trading system. In each case, the trading systems were developed on the same data, using the same GE algorithm, with only the exit strategy differing. The result is also notable in the context of evaluating prior work, as it suggests that comparing the performance of differing studies of financial prediction may be problematic, unless they have employed identical exit strategies. In considering the utility of the stop-loss, take-profit exit strategy (which has only been examined for Ford in this study), it is noted that it outperforms the standard close exit mechanism in both the training and test periods. It does not clearly dominate the extended close exit strategy, under-performing in the training period, and outperforming in the test period.

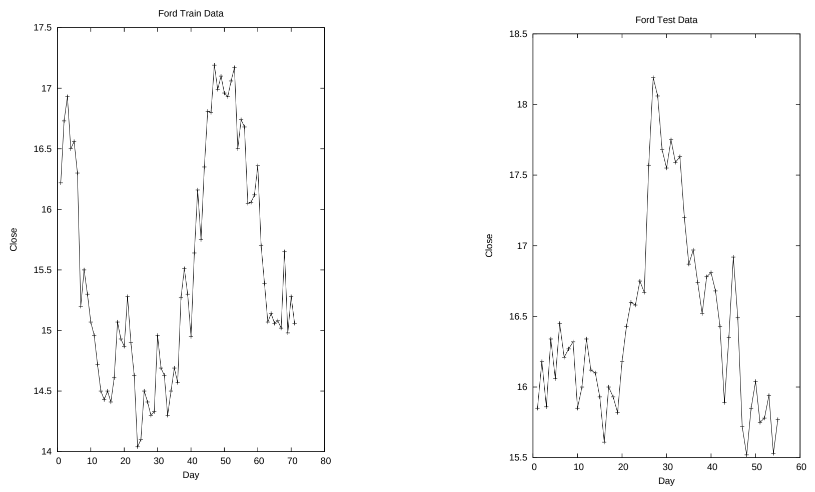


Fig. 4. Ford share price during train and test periods.

Table 3 provides information on the percentage of profitable trades under each exit strategy for the test period. Under the standard close approximately 55% of trades are winning trades, but this percentage drops under the other two strategies, notably under the stop-loss, take-profit strategy. Despite the lower percentage success under the latter two strategies, they substantially outperform the standard close strategy in terms of $ profit generated. This suggests a danger in using percentage success as a fitness measure in evolving trading systems. In the case of the stop-loss, take-profit strategy, the low percentage of profitable trades is explained by the tight take-loss criterion, where positions losing more than 0.1% were closed out. This has the affect of closing out positions which have incurred a small loss, increasing the percentage of trades closed at a loss, but simultaneously reducing the price-risk of the trading system. Although not undertaken in this study, the trigger points for the stop-loss and take-profit exit strategy could themselves be evolved as part of the trading system.

Table 3. Percentage of Profitable Trades in Test Period

	Ford	IBM
Standard Close	55	54
Extended Close	45	55
Stop-Loss	32	n/a

To provide additional insight into the trading characteristics of the evolved systems, the equity curves for Ford, in and out-of-sample, for both the standard and stop-loss, take-profit exit strategies are provided in Figures 5 and 6. An equity curve is a graph of the cumulative profits generated by the trading system over a period of time. Generally, successful trading strategies should produce good returns and a smooth increase in the equity curve. Examining the equity curves produced by the evolved trading systems suggested that profits were accumulated gradually over both the training and test periods, and were not the result of a few very successful trades.

Fig. 5. Equity curve for Ford from standard close system during train and test periods.

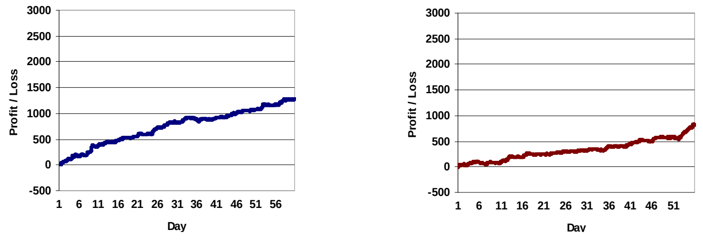
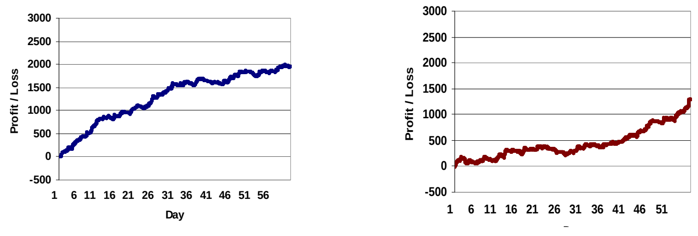


Fig. 6. Equity curve for Ford from stop-loss, take-profit system during train and test periods.



5.1 Benchmark for trading system

Before the returns generated by the trading system can be assessed, an appropriate benchmark must be defined. A wide variety of benchmarks could be utilized including a 'No-change model' and simple buy-and-hold. The no-change model assumes that prices follow a random-walk, therefore the current price is the best predictor of price in the future. Under this perspective, the expected rate of return from investment is zero. A simple buy-and-hold strategy results when an investor buys an equity and closes out the position at the end of the trading period. Although this is a simple metric, it is not risk-comparable with the trading model, as it maintains a fully-invested position in the equity market at all times. To the extent that the buy-and-hold strategy maintains a greater investment in equity markets than the trading model, it has more capital at risk. The buy-and-hold performance for Ford and IBM during the training and test period under-performs the results from all the evolved trading systems.

6 Conclusions & Future Work

In this paper GE was novelly applied for the purposes of evolving intra-day trading systems for equity markets. It is noted that the methodology has general utility for rule-induction, data-mining applications. GE was shown to evolve profitable trading rules for both training and test periods. The evolved trading rules did not exhibit large drawdowns. The significance of the choice of exit strategy for the profitability of the evolved trading systems was also noted.

A number of extensions of this study are left for future development. Our methodology has included a number of simplifications, for example we only considered a small set of technical indicators, we have ignored trading costs and slippage, and we have not explicitly incorporated a risk penalty into the fitness function. There is scope to develop more sophisticated money-management strategies than those employed in this study. For example, rather than investing a fixed amount on each trading signal, the size of the investment amount could be linked to the strength of the trading signal, with strong signals resulting in the assumption of a bigger position than weaker signals. Another avenue of exploration is the adoption of multi-stage models which could further improve predictive quality. A possible implementation of this could be to develop a series of GE models, using non-homogeneous inputs. The predictions of the individual models could then be used as inputs to a second stage model which produces the final trading signal. This second stage model could be developed using GE or a different methodology such as neural networks, in order to integrate these signals to produce the final trading signal.

References

1. Allen, F., Karjalainen, R. (1999). 'Using genetic algorithms to find technical trading rules', *Journal of Financial Economics*, 51:245-271.
2. Anderson, T. Bollerslev, T. and Das, A. (2001). 'Variance-ratio Statistics and High-Frequency Data: Testing for Changes in Intraday Volatility Patterns', *Journal of Finance*, LVI(1):305-327.
3. Bauer R. (1994). *Genetic Algorithms and Investment Strategies*, New York: John Wiley & Sons Inc.

4. Brock, W., Lakonishok, J. and LeBaron B. (1992). 'Simple Technical Trading Rules and the Stochastic Properties of Stock Returns', *Journal of Finance*, 47(5):1731-1764.

5. Brown, S., Goetzmann W. and Kumar A. (1998). 'The Dow Theory: William Peter Hamilton's Track Record Reconsidered', *Journal of Finance*, 53(4):1311-1333.

6. Chan, L. K. C., Jegadeesh, N. and Lakonishok, J. (1996). 'Momentum strategies', *Journal of Finance*, 51(5):1681-1714.

7. Cross, F. (1973). 'The Behaviour of Stock prices on Friday and Monday', *Financial Analysts' Journal*, 29(6):67-74.

8. Dacorogna, M., Gencay, R., Muller, U. Olsen, R. and Pictet, O. (2001). *An introduction to high-frequency finance*, New York: Academic Press.

9. DeBondt, W. and Thaler, R. (1987). 'Further Evidence on Investor Overreaction and Stock Market Seasonality', *Journal of Finance*, 42(3):557-581.

10. Dissanaike, G. (1997). 'Do stock market investors overreact?', *Journal of Business Finance & Accounting (UK)*, 24(1):27-50.

11. Hong, H., Lim, T. and Stein, J. (1999). 'Bad News Travels Slowly: Size, Analyst Coverage and the Profitability of Momentum Strategies', *Research Paper No. 1490, Graduate School of Business, Stanford University.*

12. Iba H. and Nikolaev N. (2000). 'Genetic Programming Polynomial Models of Financial Data Series', In *Proc. of CEC 2000*, 1459-1466, IEEE Press.

13. Koza, J. (1992). *Genetic Programming*, Massachusetts: MIT Press.

14. Lo, A. W., Mamaysky, H. and Wang, J. (2000). 'Foundations of Technical Analysis: Computational Algorithms, Statistical Inference, and Empirical Implementation', *Journal of Finance*, 55(4):1705-1765.

15. Murphy, John J. (1999). *Technical Analysis of the Financial Markets*, New York: New York Institute of Finance.

16. Neely, C., Weller P. and Dittmar, R. (1997). 'Is technical analysis in the foreign exchange market profitable? A genetic programming approach', *Journal of Financial and Quantitative Analysis*, 32(4):405-428.

17. O'Neill, M., Brabazon, A. (2004). Grammatical Swarm. In *Proceedings of the Genetic and Evolutionary Computation Conference GECCO 2004, Lecture Notes in Computer Science (3102)*, Deb et. al. Eds., 1:163-174, Berlin: Springer-Verlag.

18. O'Neill M., Brabazon, A., Ryan C., & Collins J.J. (2001). Evolving Market Index Trading Rules Using Grammatical Evolution. *LNCS 2037, Applications of Evolutionary Computation: EvoWorkshops 2001*, pp. 343-352. Springer.

19. O'Neill M. (2001). Automatic Programming in an Arbitrary Language: Evolving Programs with Grammatical Evolution, PhD thesis, University of Limerick, Ireland, 2001.

20. O'Neill M., Ryan C. (2001).'Grammatical Evolution', *IEEE Trans. Evolutionary Computation*, 5(4):349-358.

21. O'Neill M., Ryan C. (2003). *Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language*, Boston:Kluwer Academic Publishers.

22. Pring, M. (1991). *Technical analysis explained: the successful investor's guide to spotting investment trends and turning points*, New York:Mc Graw-Hill Inc.

23. Ryan C., Collins J.J., O'Neill M. (1998). Grammatical Evolution: Evolving Programs for an Arbitrary Language. *Lecture Notes in Computer Science 1391, Proceedings of the First European Workshop on Genetic Programming*, pp. 83-95, Springer-Verlag.

24. Svangard, N., Nordin, P., Llyod, S. and Wihlborg, C. (2002). Evolving short-term trading strategies using Genetic Programming, in Proceedings of CEC 2002, 2006-2010, IEEE Press.

25. Tsang, E. and Li, J. (1999). Improving technical analysis predictions: An application of Genetic Programming, in *Proceedings of 12th Annual FLAIRS conference.*