Sixth International Conference on
Computing in Economics and Finance
Barcelona, Spain, 6-8th July 2000

Reducing Failures in Investment Recommendations using Genetic Programming

Jin LI AND Edward P.K. TSANG

Department of Computer Science,
University of Essex, Wivenhoe Park, Colchester, United Kingdom
{jli, edward}@essex.ac.uk

ABSTRACT

FGP (Financial Genetic Programming) is a genetic programming based system that specialises in financial forecasting. In the past, we have reported that FGP-1 (the first version of FGP) is capable of producing accurate predictions in a variety of data sets. It can accurately predict whether a required rate of return can be achieved within a user-specified period. This paper reports further development of FGP, which is motivated by realistic needs as described below: a recommendation "not to invest" is often less interesting than a recommendation "to invest". The former leads to no action. If it is wrong, the user loses an investment opportunity, which may not be serious if other investment opportunities are available. On the other hand, a recommendation to invest leads to commitment of funds. If it is wrong, the user fails to achieve the target rate of return. Our objective is to reduce the rate of failure when FGP recommends to invest. In this paper, we present a method of tuning the rate of failure by FGP to reflect the user's preference. This is achieved by introducing a constraint-directed fitness function to FGP. The new system, FGP-2, was tested on historical Dow Jones Industrial Average (DJIA) Index. Trained with data from a seven-and-a-half-years period, decision trees generated by FGP-2 were tested on data from a three-and-a-half-years out-of-sample period. Results confirmed that one can, to a certain extent, tune the rate of failure by adjusting a constraint parameter in FGP-2. Lower failure rate can be achieved at the cost of higher missing opportunities, without affecting the overall accuracy of the system. Similar results were achieved in forecasting individual shares. Results were compared with those produced by neural networks.

1. INTRODUCTION

Genetic Programming (GP) (Koza 1992, 1994; Koza et al 1996) is a promising variant of genetic algorithms (Holland 1975, Goldberg 1989, Mitchell 1996) that uses tree representations instead of chromosomes. Genetic algorithms have been studied in financial markets for quite a few years. Bauer (1994) reported his GAs intelligent systems which aimed at finding tactic market timing strategies; Allen & Karjalainen (1995) applied Genetic Programming technique and intended to find profitable technical trading rules by trading over S&P 500 index; Chen & Yeh (1996) attempted to formalize the notion of unpredictability in the efficient market hypothesis in terms of search intensity and chance of success in the search conducted by genetic programming; Mahfoud & Mani (1996) presented a new genetic-algorithm-based system and applied it to the task of predicting the future performances of individual stocks; Neely *et al.* (1997) and Oussaidene *et al.* (1997) applied genetic programming to foreign exchange forecasting and reported some success.

Our earlier work (Tsang *et al.* 1998, Li & Tsang 1999a and Li & Tsang 1999b) reported some of preliminary but promising results by using a tool called FGP (which stands for Financial Genetic Programming). FGP was used specifically to predict whether a price series will increase by *r*% or more within the next *n* periods. FGP was found to compare favourably with random rules, commonly used individual technical rules and C4.5 rulesets with respect to prediction accuracy and average annualised rate of return.

However, for financial prediction problems, prediction accuracy is not a sole issue that one may concern. In practice, one may be more concerned with the false alarm rate (or *Rate of Failure* (RF)). A false alarm is a predicted opportunity that turns out to be a failure. If the system is employed for investment decisions, a high false alarm rate means severe losses. In this paper, study a method to reduce false alarms.

2. BACKGROUND OF FGP

Like other standard GAs, FGP maintains a population (set) of candidate solutions, each of which is a decision tree for financial forecasting. Candidate solutions are selected randomly, biased by their fitness, for involvement in generating members of the next generation. General mechanisms (referred to as *genetic operators*, e.g. reproduction, crossover, mutation) are used to combine or change the selected candidate solutions to generate offspring, which will form the population in the next generation. For details of GA and GP, readers are referred to Holland (1975), Goldberg (1989) and Koza (1992).

In FGP, a candidate solution is represented by a *genetic decision tree* (GDT). The basic elements of GDTs are *rules* and *forecast values*. A single rule is consisted of one useful indicator for prediction, one relational operator such as "greater than", or "less than", etc, and a threshold (real value). Such a single rule interacts with other rules in one GTD through logic operators such as "Or", "And", "Not", and "If-Then-Else". Forecast values in this example are either a *positive position* (i.e. *r*% return within *n* days can be achievable) or *negative position* (i.e. *r*% return within *n* days can not be achievable).

In this paper, we follow our earlier work by adopting the indictors that were derived from finance literature (see, e.g., Alexander 1964, Sweeney 1988, Brock et al. 1992, and Fama & Blume 1966). They include three types of technical analysis rules (i.e. moving average rules, filter rules, trade range break rules) as follows.

(1) MV_12 = Today's price – the average price of the previous 12 trading days
(2) MV_50 = Today's price – the average price of the previous 50 trading days
(3) Filter_5 = Today's price – the minimum price of the previous 5 trading days
(4) Filter_63 = Today's price – the minimum price of the previous 63 trading days
(5) TRB_5 = Today's price – the maximum price of the previous 5 trading days (based on the Trading Range Breakout rule [Brock et al., 1992]).
(6) TRB_50 = Today's price – the maximum price of the previous 50 trading days

Figure 1 shows an example of a simple GDT built by using the above grammar. A useful GDT in the real world may be a lot more sophisticated than this.

```
(IF  (PMV_50  <  -18.45)
                THEN  Positive
                ELSE (IF  ((TRB_5  >  -19.48) AND (Filter_63 < 36.24))
                        THEN Negative
                        ELSE Positive))
```

Figure 1. A (simplistic) GDT concerning the prediction (2.2% return within one months)

This example rule suggests that if today's price is more than 18.45 below the average price of last 50 days, then today is very likely a positive position (i.e., one could achieve a return of 2.2% or more within the next one months); otherwise one should depends on the values of TRB_5 and Filter_63 to make decisions. If today's price is no more than 19.48 above the maximum price of the previous 5 trading days or today's price is more than 36.24 above the minimum price in the last 63 days, then it is also an alternative good opportunity to make a buy decision.

FGP uses pretty standard GA and GP operators. In this paper, we focus on the fitness function. By using different fitness functions, the user is allowed to use FGP to achieve certain investment objectives.

3. MODIFYING FITNESS FUNCTION

3.1. PERFORMANCE CRITERIA AND EXPERIMENTAL DATA

Before presenting the procedure of modifying fitness function, in this subsection, we need divert to describe some criteria to assess performances of GDTs. Since GDTs are used to predict whether an increase 2.2% or more within one month can be achievable at any given day, the prediction actually can be categorised as a two-class classification problem. Each day can be classified into either a *positive position* or a *negative position*. For each GDT, we define RC, RMC, and RF as its prediction performance criteria. Formula for each criterion is given through a contingency table (Table 1) as follows:

Predicted negative positions (N-)	Predicted positive positions (N_+)	
*# of True Negative (**TN**)*	*# of False Positive (**FP**)*	**Actual negative positions (O_-)**
*# of False Negative (**FN**)*	*# of True Positive (**TP**)*	**Actual positive positions (O_+)**

$$\textbf{RC} = \frac{TP+TN}{O_+ + O_-} = \frac{TP+TN}{N_+ + N_-}; \qquad \textbf{RMC} = \frac{FN}{O_+}; \qquad \textbf{RF} = \frac{FP}{N_+};$$

Where $O_+ = FN+TP$; $O_- = TN+FP$; $N_- = TN+FN$; $N_+ = FP+TP$.

Table 1. A contingency table for two-class classification prediction problem

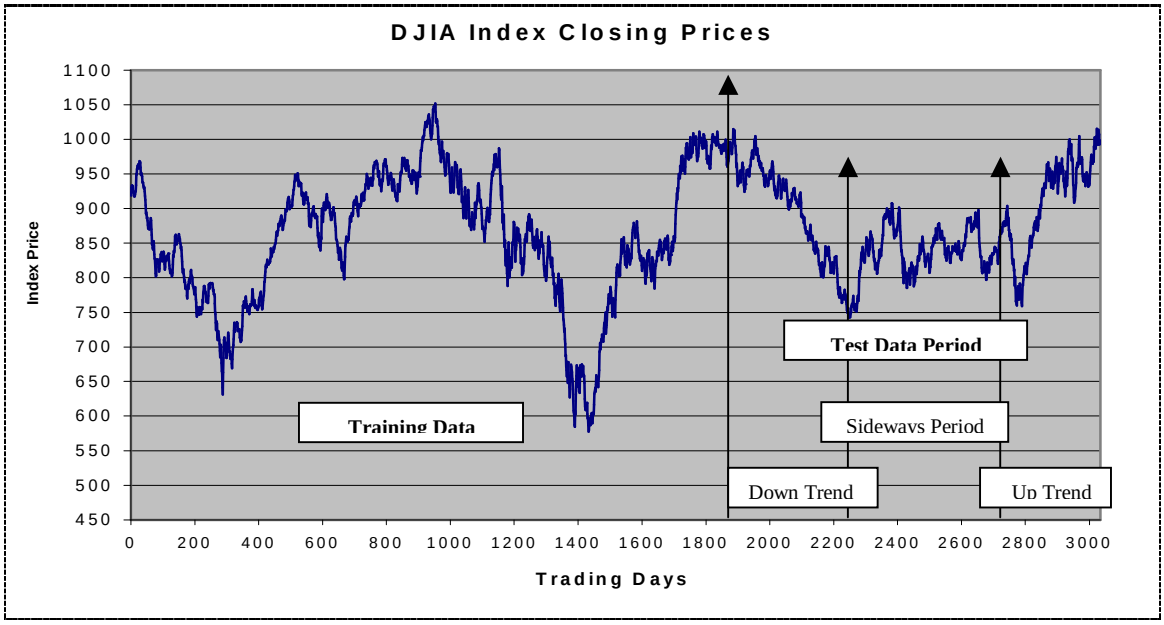

Figure 2. Experimental DJIA index data form 07/04/1969 to 09/04/1980 (3035 trading days)

The goal for modifying fitness function is to achieve a lower RF without significantly affecting RF and RMC. Experiments in this paper were first carried out on DJIA closing index from 07/04/1969 to 09/04/1981, a total of 3,035 trading days, as illustrated in Figure 2. We took the data from 07/04/1969 to 11/10/1976 (1,900 trading days) as training data, and the period from 12/10/1976 to 09/04/1981 (1135 trading days) as testing data. For the purpose of analysis, we chose r = 2.2 and n = 21 days, which give roughly 50% of positive positions in both the training and test periods. We partitioned the whole test period into three partitions: (a) down-trend period from 12/10/1976 to 12/04/78 (378 trading days); (b) side-way-trend period from 13/04/1978 to 27/03/1980 (496 trading days); and (c) up-trend period from 28/03/1980 to

09/04/81 (261 trading days). FGP was run on a Pentium PC (200MHz) using a population size of 1,200. The termination condition was 30 generations or maximum of 2 hours running.

3.2. A LINEAR FITNESS FUNCTION

In our earlier work, the fitness function mainly used in FGP was RC. By using the following fitness function, the user may satisfy individual objectives by adjusting the weights w_rc, w_rmc and w_rf:

$$f_{(1)} = w\_rc * RC - w\_rmc * RMC - w\_rf * RF \qquad (1)$$

It involves three performance values, i.e. RC, RMC and RF, each of which is assigned a different weight. Obviously, the performance of a GDT is no longer assessed by RC only, but by a synthetical value, which is the weighted sum of its three performance rates. By proper adjusting sizes of three weights, one is able to put his emphasis on one performance than on the others. In order to achieve a low RF, one may assign a higher value to *w_rc* and *w_rf* and a smaller or zero value to *w_rmc*.

To a certain extent, the fitness function $f_{(1)}$ does allow us to reduce RF. However, it has two drawbacks: 1) FGP's performance is very sensitive to the three weights; 2) results are unstable. For example, in one of our series of preliminary experiments in which we used the following three weights:

$$w\_rc=1; \quad w\_rmc=0 \quad \text{and} \quad w\_rf=\alpha \qquad \text{where } 0 < \alpha \leq 1$$

We found that a slightly bigger α almost always resulted in a GDT that did achieve a lower RF (even zero over training period) by making no positive recommendations over the test period. This was probably due to *over-fitting*. In contrast, a slightly smaller α usually resulted in generating GDT that it did not show any improvement on RF. We refer to this as the *no-effect problem*. Even though a plausible α was found (e.g. α = 0.62), it does not generate effective GDTs reliably. For example, among 10 runs, only two runs generated a GDT that predicted a few correct positive positions on the test period. The remaining 8 runs showed either the over-fitting or no-effect problem.

3.3. PUTTING CONSTRAINTS INTO FGP

We can further improve the linear fitness function $f_{(1)}$ by introducing constraints into it. We introduce a new parameter to FPG, $\Re = [P_{min}, P_{max}]$, which defines the minimum and maximum percentage of recommendations that we instruct FGP to make in the training data (with the assumption that the test data exhibits similar characteristics, as most machine learning methods do). We call the new fitness function $f_{(2)}$.

Choosing appropriate values for $\Re$ and the weights for $f_{(2)}$ remains a non-trivial task, which we approached by trial and error. When appropriate parameters were chosen, FGP managed to reduce RF and avoid the over-fitting and no-effect problems. Efficacy of the constraint in fitness function is first demonstrated by the following experiment.

RULES	RF	RMC	RC	AARR	RPR	# of Recommendation
GDT 1	0.4034	0.6402	0.5392	0.6068	0.7059	357
GDT 2	0.4122	0.6267	0.5366	0.6383	0.6755	376
GDT 3	0.4012	0.6622	0.5366	0.6198	0.7096	334
GDT 4	0.4006	0.6639	0.5366	0.6260	0.7078	332
GDT 5	0.4025	0.6740	0.5339	0.6402	0.6966	323
GDT 6	0.4103	0.5946	0.5427	0.5826	0.6929	407
GDT 7	0.4147	0.6639	0.5295	0.6299	0.6735	340
GDT 8	0.3994	0.6875	0.5330	0.6398	0.6916	308
GDT 9	0.3982	0.6655	0.5374	0.6341	0.7173	329
GDT 10	0.3640	0.6959	0.5463	0.7202	0.7244	283
MEAN	0.4006	0.6574	0.5372	0.6338	0.6995	338.9
STD	0.0141	0.0299	0.0048	0.0353	0.0167	34.7

Table 2. FGP results on test data using $f_{(2)}$ with $\Re = [35\%, 50\%]$

In this experiment, we took $\Re = [35\%, 50\%]$, *w_rmc* = 0 and *w_rf* = 1, and run FGP 10 times. Results are showed in Table 2. The performances on each criterion are rather stable and no over-fitting problem occurred again. For reference, we have included the AARR (Average Annualised Rate of Return) and RPR (Ratio of Positive Returns) in Table 2. RPR measures the proportion of times when FGP's recommendation gives a positive return, even when the target *r*% has not been achieved. Both AARR and RPR are used as reference.

RULES	RF	RMC	RC	AARR	RPR	Number of Recommendatio
GDT 1	0.4111	0.4459	0.5656	0.5782	0.6661	557
GDT 2	0.4389	0.4493	0.5410	0.5240	0.6609	581
GDT 3	0.4235	0.5355	0.5427	0.5504	0.6897	477
GDT 4	0.4502	0.4307	0.5322	0.5496	0.6460	613
GDT 5	0.4409	0.4409	0.5401	0.5233	0.6368	592
GDT 6	0.4458	0.5253	0.5269	0.5402	0.6588	507
GDT 7	0.4333	0.5051	0.5392	0.5490	0.6557	517
GDT 8	0.4361	0.3885	0.5507	0.6034	0.6636	642
GDT 9	0.4336	0.4527	0.5454	0.5382	0.6521	572
GDT 10	0.4379	0.5034	0.5357	0.5509	0.6558	523
MEAN	0.4351	0.4677	0.5419	0.5507	0.6586	558.1
SD	0.0111	0.0471	0.0107	0.0242	0.0139	51.7

Table 3. FGP results on test data using the general fitness function (RC)

To see the effect of the constrained fitness function, we compare the above results with those generated by FGP using RC as the fitness function. Results are listed in Table 3. From Table 3, we can see that by using $f_{(2)}$, the mean RF is reduced from 43.51% to 40.06%. Consequently, the mean AARR increases from 55.07% to 63.38% whilst the mean RPR rises from 65.86% to 69.95%. The price to pay for a lower RF is that more opportunities was missed: the mean RMC increased from 46.77% to 65.74%. The mean RC only slightly decreases from 54.19% to 53.72%. To determine whether result differences are statistically significant, the statistical two-tailed paired t-test can be applied on the null hypothesis that mean performances of two groups are not statistically different under each of the five criteria. Shown in Table 4 are *t*-values and their corresponding *p*-values under each criterion. The results indicated that by using the constrained fitness function, the GDTs generated statistically exhibit better RF, AARR and RPR at a significant level of $\alpha = 0.001$, though they statistically significantly get worse under RMC. However, it is important to note that they do not show statistical significance for RC (*p*-value is 0.2612). That is, RC has not been compromised as RF is reduced.

Criteria	For RF	For RMC	For RC	For AARR	For RPR
t values	-4.64	6.33	-1.16	4.69	4.71
p values	0.000205	0.000005	0.261247	0.000182	0.000175

Table 4. *t*-statistics for comparing mean performances of two groups
(Results using RC versus results using the constrained fitness function with $\Re$ = [35%, 50%])

4. IMPACT OF THE CONSTRAINT

To further explore the impact of the constraint ℜ on reducing RF, we took additional 5 non-overlapped ℜs in the fitness function respectively. Five mutually exclusive ℜs are: ℜ1=[5%, 10%], ℜ2=[10%, 15%], ℜ3=[15%, 20%], ℜ4=[20%, 35%], ℜ5= [50%, 60%]. For each ℜ, we run FGP 10 times using all the same parameters (i.e., *w_rmc*=0 and *w_rf*=1). We calculated its mean performances on test data with respect to RF, RMC, RC, RPR, AARR and mean number of positive positions recommended (which is also referred to as the mean number of recommendations), respectively. Here we simply list the means of 10 runs under each criterion. All experimental results are showed in Table 5, including preceding results using ℜ=[35%, 50%]. We visualise the results in Figure 3.

ℜ [%, %]		RF	RMC	RC	AARR	RPR	Number of Recommendations
[5, 10]	Mean	0.1348	0.9914	0.4819	2.2403	0.9222	6.2
	SD	0.1485	0.0063	0.0026	2.2924	0.1086	4.8
[10, 15]	Mean	0.2860	0.9405	0.4970	1.3681	0.8295	49.3
	SD	0.0622	0.0165	0.0076	0.3052	0.0440	13.1
[15, 20]	Mean	0.3102	0.8569	0.5174	0.9958	0.7902	125.1
	SD	0.0521	0.0641	0.0167	0.2550	0.0547	62.7
[20, 35]	Mean	0.3600	0.7525	0.5341	0.7568	0.7361	229.8
	SD	0.0259	0.0550	0.0119	0.0955	0.0341	55.1
[35, 50]	Mean	0.4006	0.6574	0.5372	0.6338	0.6995	338.9
	SD	0.0141	0.0299	0.0048	0.0353	0.0167	34.7
[50, 65]	Mean	0.4673	0.4547	0.5131	0.5226	0.6257	606.2
	SD	0.0137	0.1040	0.0164	0.0163	0.0167	115.4

Table 5. The mean performances on test data using six different constraint values of ℜ

Figure 3 shows that RF decreases gradually as ℜ is reduced. The lowest RF (13.48%) is obtained by using the smallest ℜ [5%, 10%] whereas the highest RF (46.73%) is obtained by using the biggest ℜ [50%, 65%]. The six mean RFs in the graph suggest that taking a reduced ℜ in the fitness function may result in a lower RF. Reduction in RF obviously benefited RPR and AARR. RPR rises from 57.16% to 92.85%. AARR increases dramatically from 40.32% to 300.33%. The GDT obtained by using the tightest constraints [5%, 10%] provides the most reliable recommendations, as the rate of failure is only 13.48%. The only drawback of using constraint of this tightness is that it makes fewer positive recommendations. If we reduce ℜ beyond a certain point, no positive recommendations will be made by FGP. So it is crucial to choose a proper ℜ,

which is a nontrivial task. The point that we are trying to make here is that $\Re$ is a useful handle for tuning RF in FGP.

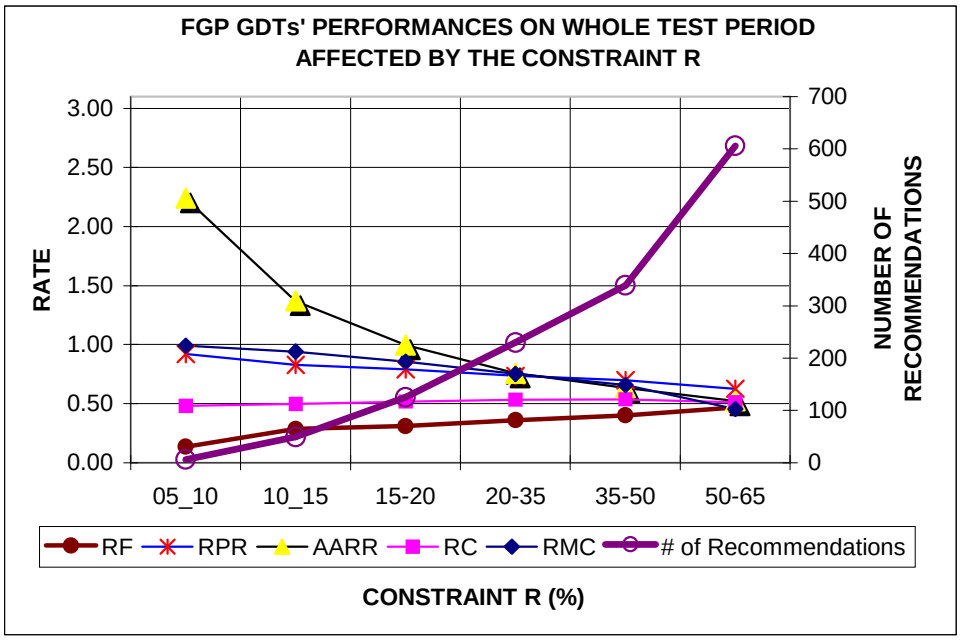

Figure 3. GDT mean performances affected by the constraint $\Re$

Next we examined the properties of the GDTs that FGP generated in different market conditions, namely, *down-trend*, *side-way-trend* and *up-trend* markets as illustrated in Figure 2. Results obtained were consistent with those shown above. To further verify FGP's reliability, we tested the same GDTs on a different test period, from 10/04/81 to 29/10/1984, which includes 900 trading days following the first test period. Consistent results were found. For simplicity, details of values are not presented here.

5. COMPARISON STUDY

Up to this point, we only tested the constrained fitness function on financial index data. Should it still be effective and applicable to individual stock data? How does FGP compare with other methods? To partially answer these questions, we referred to Saad *et al* (1998) in which three specially developed Neural Networks, (i.e. Time Delay (TDNN), Recurrent (RNN) and Probabilistic (PNN)), and a linear classifier were employed to address a similar prediction problem. They also have the goal of achieving low false

alarm. Here, we compared performances based on predictions with r = 2% and n = 22 (i.e. daily predict ions on whether a return of 2% or more can be achievable within the next 22 trading days). We obtained from Saad the 10 stocks:

- Apple (AAPL), IBM(IBM), Motorola(MOT), Microsoft (MSFT): representing the technology group which generally has high volatility
- American Express(AXP), Well Fargo (WFC): representing the banks
- Walt Disney Co. (DIS), McDonald (MCD): representing consumer stocks
- Public Svc New Mexico (PNM), Energras (V.EEG): representing cyclical stocks

All data series were ended at 06/03/1997, but with different starting dates. Following (Saad et. al 1998), for each stock, the last 100 days were chosen as the test data.

In the experiments, we ran FGP 10 times for each data set. For each run, we took 500 trading days just before the final 100 days as training data, and took a constraint $\Re$ = [20%, 30%] for most data sets except for AAPL, PNM and V.EEG, for which we took a constraint $\Re$ = [10%, 20%]. The $\Re$ value is chosen to reflect the percentage of positive positions in the data. The termination condition was 50 generations. Since FGP is a probabilistic technique, it is run ten times for each share. For each share, we picked the best GDT from the ten FGP runs for the purpose of comparison, as only the same is done for the three different neural networks reported in (Saad et. al 1998).

Table 8 lists the performance of the three different NNs, a linear classifier and FGP on 10 stocks. The "Total" column summarises the total number of predicted positive positions on all 10 stocks. The last column, "Ave.", reports the average rate of failure over 10 stocks. Like the NNs, FGP out-performed the linear classifier for any stocks. The best found GDT found 385 positive signals totally, which is slightly more than 372 found by the linear classifier. However, the average RF of the GDTs found, 5.08% is much better than 18.62%, of the linear classifier. The average RF by the best GDT found (1.29%) is better than

those of NN (3.61 to 7.56%)[1]. On individual shares, the RF by the best GDT found is as good as or better than the RF found by the NNs in 8 out of the 10 shares.

Stocks			AAPL	IBM	MOT	MSFT	AXP	WFC	DIS	MCD	PNM	V.EEG	Total	Ave.
Profit Opp. (r=2%;n=22)			62	72	81	87	92	85	74	73	50	70	746	74.6
PPN	Total N+		51	25	48	49	20	45	19	4	63	14	338	
	RF (%)		7.84	4.00	18.75	4.08	0.00	4.44	0.00	0.00	36.50	0.00		7.56
TDNN	Total N+		10	9	27	61	17	19	7	6	22	8	186	
	RF (%)		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	18.00	12.50		3.05
RNN	Total N+		16	22	33	46	49	29	48	53	35	37	368	
	RF (%)		0.00	0.00	3.03	2.17	0.00	0.00	0.00	5.66	17.14	8.11		3.61
The Linear Classifier	Total N+		82	24	87	17	10	22	2	32	20	76	372	
	RF (%)		31.71	20.83	18.39	0.00	0.00	13.64	0.00	21.88	60.00	19.74		18.62
Mean and STD of 10 GDTs	Total N+	Mean	18.5	68.7	20.7	26.8	38.3	66.6	20.1	40.2	23.4	49.4	373	
		STD	9.9	3.9	5.1	6.2	9.9	11.1	3.1	1.8	5.9	9.6		
	RF (%)	Mean	9.16	10.15	1.33	3.10	3.72	8.20	0.40	0.00	13.07	4.83		5.08
		STD	5.66	1.13	2.82	2.47	3.10	2.33	1.30	0.00	12.30	3.90		
The Best GDT	Total N+		4	70	28	33	39	69	22	43	28	49	385	
	RF (%)		0.00	8.57	0.00	0.00	0.00	4.35	0.00	0.00	0.00	0.00		1.29

Table 8**.** Performance comparisons among NNs., a linear classifier and FGP in terms of RF and N_+ (the total number of predicted position positions)

6: DISCUSSIONS AND CONCLUSIONS

The work presented in this paper involves technical indicators only. In other applications, FGP has used indicators generated by experts or mathematical models. In historical DJIA index, FGP succeeded in finding patterns that repeated themselves in the test periods. Whether repeated patterns happen by chance or not in general is the subject of the efficiency market debate, which is beyond the scope of this paper. Our position is: if such patterns exists, FGP stands a chance of finding them.

We have shown above successfully experiments of FGP. We do not wish to give the false impression that FGP can find patterns in every series. In fact, even after carefully tuning of the parameters, FGP found no repeated pattern in many share prices (more tests are needed to statistically establish FGP's success rate). The point is: one does not have to find patterns in every series to benefit from one's forecasting.

[1] The favourable results by FGP may be partly due to the rather bullish market over test period in which over 50% of the positions are positive for all the shares; e.g. 87% of the positions were positive for MSFT and 92% for AXP.

Finally, we would like to emphasize that FGP is only a tool, not a replacement of experts. Success of FGP depends on the user's choice of indicators. Besides, the users are given the responsibility to verify the rules that FGP generates (an advantage of genetic programming over neural networks is that decisions can be explained). By finding interactions between the input indicators and finding appropriate thresholds, FGP can be seen as a tool that extends the user's capability and give the user an edge over other investors of the same calibre.

ACKNOWLEDGEMENTS

This work is partly supported by the Research Promotion Fund, 1997(DDP540), University of Essex. Jin Li is supported by Overseas Research Students Awards(ORS) and University of Essex Studentship. The DJIA index data was generously provided by Professor Blake LeBaron in University of Wisconsin-Madison. Ten stock data were kindly supplied by Emad W. Saad in the Texas Tech University.

REFERNECES:

Allen, F. & Karjalainen, R. (1995), Using Genetic Algorithms to find Technical Trading Rules. Working paper at Rodney L. White Center for Financial Research.

Alexander, S.S., (1964), Price movement in speculative markets: trend or random walks, No. 2, in Cootner, P. (ed.), The random character of stock market prices, MIT Press, Cambridge, MA, 338-372.

Backus, J.W., (1959), "The syntax and semantics of the proposed international algebraic language of Zurich", ACM-GAMM conference, ICIP, Paris, June.

Bauer, R. J. Jr., (1994), *Genetic Algorithms and Investment Strategies.* New York, John Wiley & Sons, Inc.

Brock, W., Lakonishok, J. & LeBaron, B., (1992), Simple technical trading rules and the stochastic properties of stock returns, *Journal of Finance*, 47, 1731-1764.

Chen, S-H & Yeh, C-H. , (1996), Toward a computable approach to the efficient market hypothesis: An application of genetic programming, *Journal of Economic Dynamics and Control*, 21, 1043-1063.

Fama, E.F. & Blume, M.E., (1966), Filter rules and stock-market trading, *Journal of Business* 39(1), 226-241.

Goldberg, D.E., (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley.

Holland, J.H., (1975), *Adaptation in natural and artificial system*, University of Michigan Press.

Koza, J.R., (1992), *Genetic Programming: on the programming of computers by means of natural selection*. MIT Press.

Koza, J.R., (1994), *Genetic Programming II: Automatic Discovery of Reusable Programs.* MIT Press.

Koza, J., Goldberg, D., Fogel, D. & Riolo, R. (ed.), *Proceedings of the First Annual Conference on Genetic programming*, MIT Press, 1996.

Li, J. & Tsang, E.P.K, (1999a), Improving technical analysis predictions: an application of genetic programming, *Proceedings of The 12th International Florida AI Research Society Conference*, Orlando, Florida, May 1-5, 1999, 108-112.

Li, J. & Tsang, E.P.K, (1999b), Investment decision making using FGP: a case study, *Proceedings of Congress on Evolutionary Computation (CEC'99)*, Washington DC, USA, July 6-9 1999.

Mahfoud, S. & Mani, G., (1997), Financial Forecasting Using Genetic Algorithms, *Journal of Applied Artificial Intelligence* Vol.10, Num 6, 543-565.

Mitchell, M., (1996), *An Introduction to Genetic Algorithms.* MIT Press.

Neely, C., Weller, P. & Ditmar, R., (1997), Is technical analysis in the foreign exchange market profitable? A genetic programming approach, *Journal of Financial and Quantitative Analysis*, 32, 405-26.

Oussaidene, M., Chopard, B., Pictet, O. & Tomassini, M., (1997), Practical aspects and experiences - Parallel genetic programming and its application to trading model induction, *Journal of Parallel Computing* Vol. 23, No. 8, 1183-1198.

Saad, E., Prokhorov, D., and Wunsch, D., (1998), Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks, *IEEE Transactions on Neural Networks*, vol. 9. 1456-1470.

Sweeney, R.J., (1988), "Some new filter rule tests: Methods and results," Journal of Financial and Quantitative Analysis, 23, 285-300.

Tsang, E.P.K., Li, J. & Butler, J.M., (1998), EDDIE beats the bookies, *International Journal of Software, Practice & Experience*, Wiley, Vol.28 (10), 1033-1043