Using Hyperheuristics under a GP framework for Financial Forecasting

Michael Kampouridis[1] and Edward Tsang[2]

[1] School of Computer Science and Electronic Engineering, University of Essex, Wivenhoe Park, CO4 3SQ, UK
mkampo@essex.ac.uk
http://kampouridis.net

[2] Centre for Computational Finance and Economic Agents, University of Essex, Wivenhoe Park, CO4 3SQ, UK
edward@essex.ac.uk
http://www.bracil.net/edward/

Abstract. Hyperheuristics have successfully been used in the past for a number of search and optimization problems. To the best of our knowledge, they have not been used for financial forecasting. In this paper we use a simple hyperheuristics framework to investigate whether we can improve the performance of a financial forecasting tool called EDDIE 8. EDDIE 8 allows the GP (Genetic Programming) to search in the search space of indicators for solutions, instead of using pre-specified ones; as a result, its search area is quite big and sometimes solutions can be missed due to ineffective search. We thus use two different heuristics and two different mutators combined under a simple hyperheuristics framework. We run experiments under five datasets from FTSE 100 and discover that on average, the new version can return improved solutions. In addition, the rate of missing opportunities reaches it's minimum value, under all datasets tested in this paper. This is a very important finding, because it indicates that thanks to the hyperheuristics EDDIE 8 has the potential of missing less forecasting opportunities. Finally, results suggest that thanks to the introduction of hyperheuristics, the search has become more effective and more areas of the space have been explored.

Key words: Hyperheuristics, Genetic Programming, Financial Forecasting

1 Introduction

Financial forecasting is an important area in computational finance [27]. There are numerous works that attempt to forecast the future price movements of a stock; several examples can be found in [10, 7]. A number of different methods have been used for forecasting. Such examples are for instance, Support Vector Machines [25], Fuzzy Logic [15] and Neural Networks [6]. Genetic Programming [18, 24] (GP) is an evolutionary technique that has widely been used for financial

forecasting. Some recent examples are [26, 6, 1, 12], where GP was used for time series forecasting.

In a previous paper [16], we presented EDDIE 8 (ED8), which was an extension of the financial forecasting tool EDDIE (Evolutionary Dynamic Data Investment Evaluator) [28, 29]. EDDIE is a machine learning tool that uses Genetic Programming to make its predictions. The novelty of ED8 was in its extended grammar, which allowed the GP to search in the space of indicators to form its Genetic Decision Trees. In this way, ED8 was not constrained in using pre-specified indicators, but it was left up to the GP to choose the optimal ones. We then proceeded to compare ED8 with its predecessor, which used indicators that were pre-specified by the user. Results showed that thanks to the new grammar, ED8 could find new and improved solutions. However, those results also suggested that ED8's performance could have been compromised by the enlarged search space. With the old grammar, which was also discussed in [16], EDDIE used 6 indicators from technical analysis with two pre-specified period lengths. For instance, if one of the indicators was Moving Average, then the two period lengths used would be 12 and 50 days. On the contrary, ED8 could use any period within a given parameterized range, which for our experiments was set to 2-65 days. Thus, the GP could come up with any indicator within that range, and not just with 12 and 50 days. As we can see, the search space of ED8 was much bigger than the one of its predecessor. With the old grammar, the GP would have to combine only 12 indicators (6 indicators with 2 periods each) to form trees; on the other hand, ED8 would have to combine $6 \times (65 - 1) = 384$ indicators. The difficulty of ED8 in making the appropriate indicators combinations was obvious. In addition, the search space of ED8 was also much bigger. For instance, let us assume that the training data consisted of 1000 data points. Then, with the old grammar, the GP would have to search in a space of $12 \times 1000 = 12,000$ points. On the other hand, ED8 would have to search in the much larger space of 384,000 data points (384×1000). It was therefore obvious that we needed to find new ways that would make the search more effective in such large search areas.

In this paper, we want to investigate if a hyperheuristics framework can address this issue. Hyperheuristics is a well-known method that has been used in a variety of search and optimization problems [23], such as transportation [14], scheduling [11], and timetabling [8]. To the best of our knowledge, *hyperheuristics have not been used before for a financial forecasting problem.* We thus use a simple framework that utilizes simple hill climbing, simulated annealing, random mutation, and weighted random mutation. We are interested in investigating whether hyperheuristics can improve ED8's performance, and whether the search space can be better explored. The rest of this paper is organized as follows: Section 2 presents the ED8 algorithm, Sect. 3 presents the hyperheuristics framework, along with its heuristics and operators, Sect. 4 presents the experimental setup, Sect. 5 presents and discusses the results, and finally, Sect. 6 concludes this paper and also discusses future work.

2 Presentation of EDDIE 8

EDDIE is a forecasting tool, which learns and extracts knowledge from a set of data. The kind of question ED8 tries to answer is 'will the price increase within the n following days by r%'? The user first feeds the system with a set of past data; EDDIE then uses this data and through a GP process, it produces and evolves Genetic Decision Trees (GDTs), which make recommendations of buy (1) or not-to-buy (0).

The set of data used is composed of three parts: daily closing price of a stock, a number of attributes and signals. Stocks' daily closing prices can be obtained online in websites such as `http://finance.yahoo.com` and also from financial statistics databases like *Datastream*. The attributes are indicators commonly used in technical analysis [13]; which indicators to use depends on the user and his belief of their relevance to the prediction. The technical indicators that we use in this work are: Moving Average (MA), Trade Break Out (TBR), Filter (FLR), Volatility (Vol), Momentum (Mom), and Momentum Moving Average (MomMA).[3]

The signals are calculated by looking ahead of the closing price for a time horizon of n days, trying to detect if there is an increase of the price by $r\%$ [28]. For this set of experiments, n was set to 20 and r to 4%. In other words, the GP is trying to use some of the above indicators to forecast whether the daily closing price iss going to increase by 4% within the following 20 days.

After we feed the data to the system, EDDIE creates and evolves a population of GDTs. Figure 1 presents the Backus Normal Form (BNF) [4] (grammar) of ED8. As we can see, the root of the tree is an If-Then-Else statement. The first branch is either a boolean (testing whether a technical indicator is greater than/less than/equal to a value), or a logic operator (and, or, not), which can hold multiple boolean conditions. The 'Then' and 'Else' branches can be a new GDT, or a decision, to buy or not-to-buy (denoted by 1 and 0).

As we can see from the grammar in Fig. 1, there is a function called *Var-Constructor*, which takes two children. The first one is the indicator, and the second one is the Period. Period is an integer within the parameterized range [MinP, MaxP] that the user specifies. As a result, ED8 can return decision trees with indicators like 15 days Moving Average, 17 days Volatility, etc. The period is not an issue and it is up to ED8, and as a consequence up to the GP and the evolutionary process, to decide which lengths are more valuable for the prediction. A sample GDT is presented in Fig. 2. As we can observe, the periods 12 and 50 are now in a leaf node, and thus are subject to genetic operators, such as crossover and mutation.

Depending on the classification of the predictions, we can have four cases: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative

[3] We use these indicators because they have been proved to be quite useful in developing GDTs in previous works like [21], [2] and [3]. Of course, there is no reason why not use other information like fundamentals or limit order book. However, the aim of this work is not to find the ultimate indicators for financial forecasting.

<Tree> ::= If-then-else <Condition> <Tree> <Tree> | Decision
<Condition> ::= <Condition> "And" <Condition> |
<Condition> "Or" <Condition> |
"Not" <Condition> |
VarConstructor <RelationOperation> Threshold
<VarConstructor> ::= MA period | TBR period | FLR period | Vol period | Mom period | MomMA period
<RelationOperation> ::= ">" | "<" | "="
Terminals:
MA, TBR, FLR, Vol, Mom, MomMA are function symbols
Period is an integer within a parameterized range, [MinP, MaxP]
Decision is an integer, Positive or Negative implemented
Threshold is a real number

Fig. 1. The Backus Normal Form of ED8

(FN). As a result, we can use the metrics presented in Equations (1), (2) and (3).

Rate of Correctness

$$RC = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Rate of Missing Chances

$$RMC = \frac{FN}{FN + TP} \tag{2}$$

Rate of Failure

$$RF = \frac{FP}{FP + TP} \tag{3}$$

The above metrics combined give the following fitness function, presented in Equation (4):

$$ff = w_1 * RC - w_2 * RMC - w_3 * RF \tag{4}$$

where w_1, w_2 and w_3 are the weights for RC, RMC and RF respectively. These weights are given in order to reflect the preferences of investors. For instance, a conservative investor would want to avoid failure; thus a higher weight for RF should be used. For our experiments, we chose to include strategies that mainly focus on correctness and reduced failure. Thus these weights have been set to 0.6, 0.1 and 0.3 respectively.

The fitness function is a constrained one, which allows EDDIE to achieve lower RF. The effectiveness of this constrained fitness function has been discussed in [29, 20]. The constraint is denoted by R, which consists of two elements represented by percentage, given by

$$R = [Cmin, Cmax],$$

where $C_{min} = \frac{P_{min}}{N_{tr}} \times 100\%$, $C_{max} = \frac{P_{max}}{N_{tr}} \times 100\%$, and $0 \leq C_{min} \leq C_{max} \leq 100\%$. N_{tr} is the total number of training data cases, P_{min} is the minimum

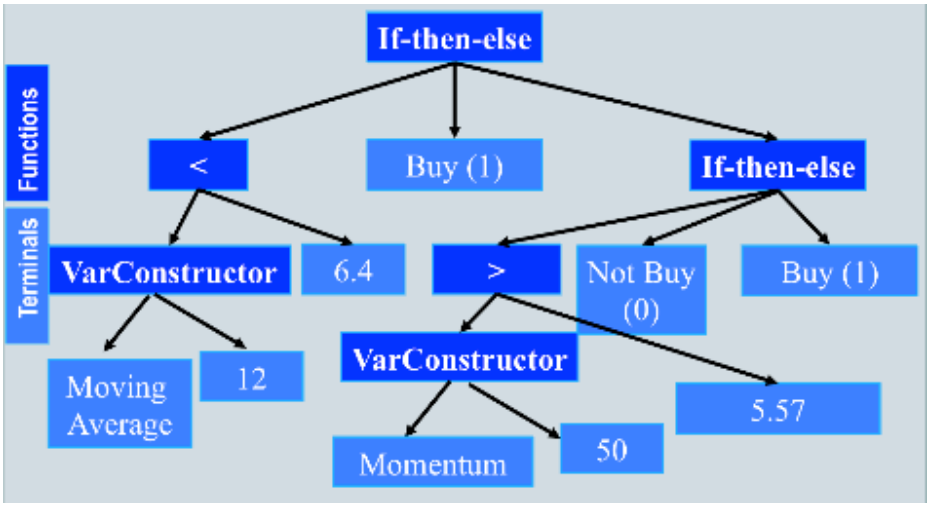


Fig. 2. Sample GDT generated by EDDIE 8.

number of positive position predictions required, and P_{max} is the maximum number of positive position predictions required.

Therefore, a constrained of $R = [50, 65]$ would mean that the percentage of positive signals that a GDT predicts[4] should fall into this range. When this happens, then w_1 remains as it is (i.e. 0.6 in our experiments). Otherwise, w_1 takes the value of zero.

During the evolutionary procedure, we allow three operators: crossover, mutation and reproduction. After reaching the last generation, the best-so-far GDT, in terms of fitness, is applied to the testing data.

This concludes this short presentation of ED8. In the next section we briefly present the heuristics and operators used in our framework, and then present the hyperheuristics framework itself.

3 Hyperheuristics Framework

3.1 Heuristics and Operators

We use two heuristics, namely simple hill climbing (SHC) and simulated annealing (SA), and two GP operators, namely random mutation (*Rnd Mut*) and weighted random mutation (*W. Rnd Mut*). However, we do not argue that the above techniques are the optimal ones for the purposes of our experiments. Other heuristics and operators could also be chosen. Nevertheless, the purpose of this

[4] As we have mentioned, each GDT makes recommendations of buy (1) or not-to-buy (0). The former denotes a positive signal and the latter a negative. Thus, within the range of the training period, which is t days, a GDT will have returned a number of positive signals

paper is not to look for the most effective heuristics or operators, but to investigate whether and how these heuristics combined under a hyperheuristics framework can be used to improve the performance of ED8. We leave it to future research to investigate for even more appropriate heuristics.

Let us now start by explaining how the first heuristic is used (SHC). First of all, a leaf which contains a period is randomly selected. Let us assume that this period is $p = 12$ (days). Then the simple hill climber increases the period by 1. If there is an improvement in the fitness, the process is completed and the period returned to the leaf is $p + 1$, which in this example is 13. If, on the other hand, there is no improvement, then the period p is reduced by 1. Again, if this has resulted to an improvement, then the process ends and the new period is $p - 1$, which in this example is 11. If again there is no improvement, then the initial period $p = 12$ is returned and the process is terminated. The purpose of using this heuristic is quite obvious: we are interested in investigating how a marginal change in the period can affect the performance of a tree.

The motivation behind the use of the remaining three heuristics/operators is to expand the search to other areas of the search space, and not just the neighborhood of a selected period. The second heuristic is the classic simulated annealing. A leaf from a tree is again randomly chosen. Then the standard algorithm of simulated annealing is used [17, 9]. We thus allow swaps among the different periods (2-65 days). The initial temperature t is set to a value such that around 65% of the inferior moves are accepted. The temperature is gradually reduced according to the following formula: $t = (1 - f) \times t$, where t denotes the temperature and f is the annealing temperature reduction factor, which is equal to 0.1. At each temperature a maximum of 10 iterations are executed. The acceptance criterion is the Metropolis criterion [22].

The two operators used in our framework are *Rnd Mut* and *W. Rnd Mut*. They both randomly select a leaf which contains a period. Random mutation then mutates this leaf to another period. This operator is the typical GP mutation operator and it allows us to randomly explore different areas of the search space. On the other hand, weighted random mutation offers a roulette-wheel-like selection, where the new period, which is going to replace a leaf, is probabilistically selected, based on its occurrence. However, it should be highlighted that *W. Rnd Mut* does not just focus on the period, but on the indicator itself. So when we select a period p for mutation, we also take into account the accompanying indicator (e.g. Moving Average). We then calculate the period occurrence under that specific indicator. The reason for doing this is that we aim to target indicators as a whole and not just periods. In this way, the new period p' that is going to replace of the old period p is a period that has a high occurrence under this specific indicator. *W. Rnd Mut* therefore introduces useful indicators to the population, rather than just introducing useful periods, like the other 3 heuristics/mutators do.

3.2 The Framework

In this simple framework, all low level heuristics are used simultaneously. The low level heuristics include both the heuristics and operators described in the previous section. Inspired by the Population Based Incremental Learning algorithm [5] and the alike Estimation of Distribution Algorithms [19, 30], all four low level heuristics are initially given a weight w of being selected, where $w = 25\%$. Then depending on the result on the performance of a tree after the implication of a heuristic, the following cases can occur:

1. Increase in performance
   (a) By using a new period
   (b) By using a pre-existing period
2. No change in performance
   (a) By using a new period
   (b) By using a pre-existing period
3. Decrease in performance

As we can see, we are not only interested in improving the performance of a GDT. We are also interested in whether this improvement comes from a new or from a pre-existing period. The reason for this is obvious: one of the goals of our experiments is to have better exploration of the search space. We thus want to reward a heuristic that allowed a new period to be invoked in the population.

Let us denote the reward/punishment after the implication of a heuristic by r. Then the weight w for each one of the above cases (1-3) is updated as follows:

1. Increase in performance
   (a) $w = w + r$ (new period)
   (b) $w = w + r/2$ (pre-existing period)
2. No change in performance
   (a) $w = w + r/5$ (new period)
   (b) $w = w - r/5$ (pre-existing period)
3. Decrease in performance
   $w = w - r$

The highest reward is offered when there is an increase in the performance and a new period has been used (1-a). When an improvement is caused by the use of a period that is already being used by other GDTs, half of the reward is offered (1-b). In the case of no change in performance, we still offer a small reward (equal to $\frac{r}{5}$) if a new period has been used (2-a), since a new area of the search space has been explored. If no new period has been used, then a small punishment is invoked (equal to $-\frac{r}{5}$) (2-b). Finally, there is also a punishment in the case of decrease in the performance (3).

Table 1. GP Parameters.

GP Parameters	
Max Initial Depth	6
Max Depth	8
Generations	50
Population size	500
Tournament size	2
Reproduction probability	0.1
Crossover probability	0.9
Mutation probability	0.01
Period (EDDIE 8)	[2,65]

4 Experimental Setup

The data we feed to ED8 consist of daily closing prices. These closing prices are from 5 arbitrary stocks from FTSE100. These stocks are: British Petroleum (BP), Carnival, Hammerson, Imperial Tobacco, and Xstrara. The training period is 1000 days and the testing period 300.

The GP parameters are presented in Table 1. For statistical purposes, we run the GP for 50 times. Thus, the process that is followed is that we create a population of 500 GDTs, which are evolved for 50 generations, over a training period of 1000 days. At the last generation, the best performing GDT in terms of fitness is saved and applied to the testing period. As we have already said, this procedure is done for 50 individual runs.

In addition, we should emphasize that we require that the datasets have a satisfactory number of actual positive signals. By this we mean that we are neither interested in datasets with a very low number of signals, neither with an extremely high one. Such cases would be categorized as chance discovery, where people are interested in predicting rare events, such as a stock market crash. Clearly this is not the case in our current work, where we use EDDIE for investment opportunities forecasting. We are thus interested in datasets that have opportunities around 50-70% (i.e. 50-70% of actual positive signals). Therefore, we need to calibrate the values of r and n (see Sect. 2) accordingly, so that we can obtain the above percentage from our data. For our experiments, the value of n is set to 20 days. The value of r varies, depending on the dataset. This is because one dataset might reach a percentage of 50-70% with $r = 4\%$, whereas another one might need a higher or lower r value. Accordingly, we need to calibrate the value of the R constraint, so that EDDIE produces GDTs that forecast positive signals in a range which includes the percentage of the *actual* positive signals of the dataset we are experimenting with. R thus takes values in the range of $[-5\%, +5\%]$ of the number of positive signals that the dataset has. For instance, if under $r = 4\%$ and $n = 20$ days, a dataset has 60% of actual positive signals, then R would be set to [55,65].

Finally, Table 2 presents the parameters of the hyperheuristics framework. The probability of applying hyperheuristics is set for this work at 35%. Thus, 35% of the GDTs' periods can be updated through hyperheuristics at each generation. We did not want to set a higher probability, because this could increase the computational times. At the moment, the initial weight and reward are set to 0.25 and 0.005, respectively. The former is set to 0.25 because we want all heuristics to have equal chances of being selected when the process starts. The reward is set arbitrarily. We leave it to future research to investigate whether the test results can be affected by different parameter values.

Table 2. Hyperheuristics Parameters.

Hyperheuristics Parameters	
Hyperheuristics probability	0.35
Initial Weight	0.25
Reward/Punishment	0.005

5 Results

We ran both the traditional version of EDDIE 8 (ED8) and the one that uses hyperheuristics (ED8-HH) for 50 individual times and present the results in this section. As we mentioned at the beginning of this paper, the goal of our experiments is twofold: (*a*) to investigate whether hyperheuristics can improve the performance of the trees that ED8 uses and (*b*) to investigate whether hyperheuristics can offer better exploration of the search space.

Let us begin with the performance of the GDTs. Table 3 presents the average and optimal results over the 50 runs for both ED8 and ED8-HH for the 5 stocks. The first row of each dataset presents the ED8 average results and the second row presents the ED8-HH ones. The third and fourth row of each dataset present the optimal values of the metrics for ED8 and ED8-HH, respectively. Optimal value can be either a maximum or a minimum, depending on the metric. Hence, because Fitness and RC are maximization problems, *optimal* refers to the *maximum* value of these metrics, over the 50 individual runs. On the other hand, because RMC and RF are minimization problems, *optimal* refers to the *minimum* value of these two metrics. In order to judge if the average results (rows 1 and 2) are significant, we also ran Kolmogorov-Smirnov tests at 5% significance level. The p-values of the tests are presented in Table 4. Thus, when there is a significantly better average value in Table 3, this is denoted by bold fonts. In addition, a higher value for the optimal results is underlined.

A first observation from Table 3 is that on average, there seems to be a 'tie' between ED8 and ED8-HH. ED8 is doing significantly better in the average Fitness and RF of BP, and in the average RMC of Hammerson. On the other hand,

Table 3. Average and optimal results over 50 runs for ED8 and ED8-HH for 5 FTSE 100 stocks. Each stock presents results in four rows: one for the average values of the metrics for ED8, one for the average values for ED8-HH, one for the optimal values of the metrics for ED8, and one for the optimal values for ED8-HH. A significantly better value (at 5% significance level) between ED8 and ED8-HH for the average results is presented in bold fonts, whereas a better value for the optimal results is underlined.

Stock		Fitness	RC	RMC	RF
BP	(ED8 Avg)	**0.2005**	0.5303	0.4756	**0.2338**
	(ED8-HH Avg)	0.1767	0.5299	0.4519	0.3203
	(ED8 Opt)	0.3341	0.6900	0.2523	0.1691
	(ED8-HH Opt)	0.2850	0.6500	0	0.1176
Carnival	(ED8 Avg)	0.1871	0.5607	0.3531	0.3801
	(ED8-HH Avg)	0.1900	0.5607	**0.2892**	0.3917
	(ED8 Opt)	0.2511	0.6300	0.1734	0.1728
	(ED8-HH Opt)	0.2470	0.6267	0	0.2414
Hammerson	(ED8 Avg)	0.2488	0.6071	**0.2331**	0.3073
	(ED8-HH Avg)	0.2164	0.5675	0.3507	0.2967
	(ED8 Opt)	0.3311	0.7033	0.0340	0.2472
	(ED8-HH Opt)	0.3450	0.7200	0	0.1818
Imp.Tobacco	(ED8 Avg)	0.1832	0.5245	0.6488	0.2222
	(ED8-HH Avg)	0.1946	0.5395	**0.5910**	0.2332
	(ED8 Opt)	0.2790	0.6533	0.2595	0.0270
	(ED8-HH Opt)	0.2959	0.6533	0	0.0222
Xstrata	(ED8 Avg)	0.2419	0.5807	0.3264	0.2462
	(ED8-HH Avg)	0.2359	0.5741	0.3330	0.2508
	(ED8 Opt)	0.3571	0.7267	0.0664	0.0400
	(ED8-HH Opt)	0.2510	0.7600	0	0.0714

ED8-HH is doing significantly better in the average values of RMC of Carnival and Imperial Tobacco. The remaining metrics of the other stocks have insignificant differences at the 5% level. It is not very easy to draw safe conclusions by just looking at the average results.

However, the picture gets clearer when we look at the optimal values of the metrics. ED8-HH is doing better in 12 cases (BP: RMC; Carnival: RMC; Hammerson: Fitness, RC, RMC, RF; Imp. Tobacco: Fitness, RMC, RF; Xstrata: RC, RMC), whereas ED8 is doing better only in 7 cases. What is even more interesting though, is the large and consistent improvement in the minimum value of RMC, for all 5 stocks. In fact, as we can see from Table 3, the minimum RMC under ED8-HH is always 0. This is a very important finding, because it indicates that ED8-HH has the potential of never missing any forecasting opportunities.

Moreover, Figure 3 presents the average over the 50 runs of the percentage of extinct indicators for each generation. To be more specific, the trees of each generation use a number of indicators from the available population. As we mentioned in Sect. 1, ED8 and ED8-HH use 384 indicators. An 'extinct indicator'

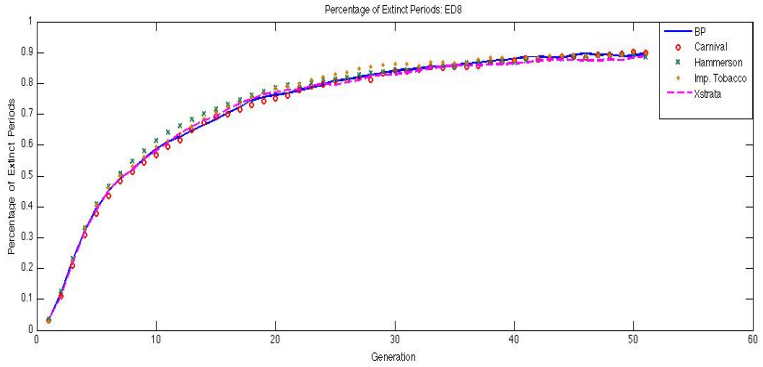
Table 4. p-values of the Kolmogorov-Smirnov non-parametric tests. A p-value that is less than 0.05 denotes significantly different distributions.

Stock	Fitness	RC	RMC	RF
BP	0.0171	0.9541	0.5077	0.0001
Carnival	0.1546	0.5077	0.0317	0.1546
Hammerson	0.1546	0.0560	0.0089	0.2408
Imp.Tobacco	0.6779	0.0560	0.0317	0.3584
Xstrata	0.3584	0.2408	0.3584	0.5077

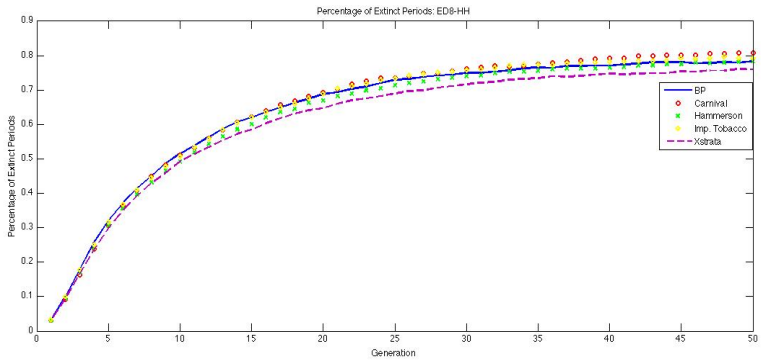
is therefore defined as the indicator that is not currently used by any of the GDTs in the population. From Fig. 3 we can see that at generation 0, the percentage of extinct indicators is very low for both ED8 and ED8-HH (around 2%) for all 5 datasets. This means that very few indicators are not being used by the trees at generation 0. As evolution proceeds, we can observe that the percentage of extinct indicators increases, and thus less and less indicators are used by the GDTs. This is normal, because it means that the GDTs have found some useful indicators and are focusing on them. However, by the end of the evolutionary process, we can see that the percentage of extinct indicators has increased to around 90% for ED8. This means that the GDTs are using only 10% of the available 384 indicators, i.e. around 38 indicators, and are thus only taking advantage of a very small area of the search space. So all 500 GDTs are using only these indicators, which indicates very low diversity in the population. When we look at ED8-HH's statistics, we see that throughout the evolutionary process, the percentage of extinct indicators is constantly lower by around 5-10%. This indicates that ED8-HH constantly explores more areas of the search space than ED8. By generation 50, this percentage is around 80%. ED8-HH has thus managed to keep 'alive' an extra 10% (in total 20%) of the indicators. This better exploration has of course led to more diversity in the population and could be a reason of ED8-HH's improved performance that we saw earlier in Table 3.

Furthermore, Table 5 presents information about the individual heuristics. For each stock, we present information in three rows. The first one presents the percentage of improvement caused by the relevant heuristic, on the GDTs' performance. Results are on average of 50 runs. Thus, the first element of the table informs us that *Rnd Mut* has on average improved the BP's GDTs performance by 5.72%. Similarly, *SA*, *SHC* and *W. Rnd Mut* have improved the performance by around 5%. Same rates of improvement apply to the remaining 4 stocks, too, varying in around 3-8%. Overall, all four heuristics seem to have equivalent contribution to the GDTs' improvement in performance. In addition, *SHC* seems to be the most consistent one, always having an improvement above 5%.

The second and third row of each stock present the percentage of improvements that was caused either by an existing (second row) or by a new period (third row). Therefore, BP's 88.25% in *Rnd Mut* means that 88.25% of the improvements came from periods that were already being used by the population of GDTs. In other words, already-successful periods were located by the heuris-



(a) Percentage of Extinct Indicators: ED8



(b) Percentage of Extinct Indicators: ED8-HH

Fig. 3. Percentage of Extinct Indicators for ED8 (a) and ED8-HH (b)

tics and re-used by the GDTs. The remaining 11.75% of the improvements came by new periods, which did not belong to the current GP population. As we explained earlier, we are especially interested in this figure, because it allows diversity in the population. From our experiments, SA and SHC seem to be more consistent in allowing new periods in the population, with percentages varying in the range of 10.22-29.32%. On the other hand, the two mutators seem to be better in re-using successful existing periods.

Finally, we should say that a single run of ED8 lasted approximately 3 minutes, whereas a single run of ED8-HH lasted approximately 13 minutes. It is obvious that the latter is more computationally intensive. However, we consider that the extra wait time is worthy, because of the improvements we saw in performance and search effectiveness. Future work could focus on improving the computational time of ED8-HH.

Table 5. Performance of each heuristic. First row presents the percentage of improvement to the GDTs' performance introduced by the specific heuristic. The second row presents the percentage of improvement that was caused by an existing indicator and the third row the percentage of improvement caused by a new indicator. Results are on average of 50 runs.

Stock		Rnd Mut	SA	SHC	W. Rnd Mut
BP	(Improvement)	0.0572	0.0560	0.0522	0.0552
	(Existing Ind.)	0.8825	0.7760	0.7068	0.8565
	(New Ind.)	0.1175	0.2240	0.2932	0.1435
Carnival	(Improvement)	0.0602	0.0748	0.0768	0.0536
	(Existing Ind.)	0.8967	0.7988	0.7887	0.8655
	(New Ind.)	0.1033	0.2012	0.2113	0.1345
Hammerson	(Improvement)	0.0541	0.0711	0.0777	0.0412
	(Existing Ind.)	0.8333	0.8311	0.8500	0.9062
	(New Ind.)	0.1667	0.1689	0.1500	0.0938
Imp. Tobacco	(Improvement)	0.0339	0.0298	0.0512	0.0297
	(Existing Ind.)	0.8930	0.8450	0.8978	0.9522
	(New Ind.)	0.1070	0.1550	0.1022	0.0478
Xstrata	(Improvement)	0.0508	0.0705	0.0728	0.0669
	(Existing Ind.)	0.9635	0.8121	0.8004	0.8778
	(New Ind.)	0.0365	0.1879	0.1996	0.1222

6 Conclusion

Hyperheuristics have been used in the past for several search and optimization problems, but not for financial forecasting. In this paper we used a simple hyperheuristics framework to investigate whether we could improve the performance of a financial forecasting tool called EDDIE 8 (ED8). ED8 allows the GP to search in the search space of indicators for solutions; as a result, its search space is quite big and sometimes solutions can be missed due to ineffective search. We thus used two different heuristics and two GP mutators combined under a simple hyperheuristics framework and discovered that hyperheuristics returned improved solutions. In addition, ED8-HH's minimum RMC reached it's minimum value of 0, under all 5 stocks tested in this paper. ED8-HH has thus the potential of never missing forecasting opportunities. Finally, results suggested that thanks to the introduction of hyperheuristics, more areas of the search space were explored, which led to higher diversity in the GP population.

Overall, we can characterize the results as encouraging. However, tests took place under a small sample (five stocks) and thus more experiments need to be done under more datasets. Furthermore, as our experiments took place under a simple hyperheuristics framework, we are interested in investigating the effects of more complex frameworks. At the same time, we want to examine the effects of different heuristics. Our goal is to show that under a more sophisticated framework, and with the use of more heuristics, the search can become even more effective, resulting to even higher performance of the GDTs.

Acknowledgments

The version has been revised in light of two anonymous referees' very helpful reviews, for which the authors are very grateful. The EPSRC grant with number EP/P563361/0 is also gratefully acknowledged.

References

1. Agapitos, A., O'Neill, M., Brabazon, A.: Evolutionary learning of technical trading rules without data-mining bias. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) Parallel Problem Solving from Nature – PPSN XI. Lecture Notes in Computer Science, vol. 6238, pp. 294–303. Springer (2010)
2. Allen, F., Karjalainen, R.: Using genetic algorithms to find technical trading rules. Journal of Financial Economics 51, 245–271 (1999)
3. Austin, M., Bates, G., Dempster, M., Leemans, V., Williams, S.: Adaptive systems for foreign exchange trading. Quantitative Finance 4(4), 37–45 (2004)
4. Backus, J.: The syntax and semantics of the proposed international algebraic language of Zurich. In: International Conference on Information Processing. pp. 125–132. UNESCO (1959)
5. Baluja, S.: Population-based incremental learning: a method for integrating genetic search based function optimisation and competitive learning (1994), technical Report, Carnegie Mellon University
6. Bernal-Urbina, M., Flores-Méndez, A.: Time series forecasting through polynomial artificial neural networks and genetic programming. In: Proceedings of the IEEE Congress on Evolutionary Computation. pp. 3324–3329. Hong Kong (June 2008)
7. Binner, J., Kendall, G., Chen, S.H. (eds.): Applications of Artificial Intelligence in Finance and Economics, Advances in Econometrics, vol. 19. Elsevier (2004)
8. Burke, E., MacCloumn, B., Meisels, A., Petrovic, S., Qu, R.: A graph-based hyper heuristic for timetabling problems. European Journal of Operational Research 176, 177–192 (2006)
9. Cerny, V.: A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. Journal of Optimization Theory and Applications 45, 41–51 (1985)
10. Chen, S.H.: Genetic Algorithms and Genetic Programming in Computational Finance. Springer-Verlag New York, LLC (2002)
11. Cowling, P., Chakhlevitch, K.: Hyperheuristics for managing a large collection of low level heuristics to schedule personnel. vol. 2, pp. 1214 – 1221 Vol.2 (dec 2003)
12. Dempsey, I., O'Neill, M., Brabazon, A.: Live trading with grammatical evolution. In: Proceedings of the Grammatical Evolution Workshop (2004)
13. Edwards, R., Magee, J.: Technical analysis of stock trends. New York Institute of Finance (1992)
14. Hart, E., Ross, P., Nelson, J.: Solving a real-world problem using an evolving heuristically driven schedule builder. Evol. Comput. 6(1), 61–80 (1998)
15. Kablan, A.: Adaptive neuro fuzzy inference systems for high frequency financial trading and forecasting. pp. 105 –110 (oct 2009)
16. Kampouridis, M., Tsang, E.: EDDIE for investment opportunities forecasting: Extending the search space of the GP. In: Proceedings of the IEEE Conference on Evolutionary Computation. pp. 2019–2026. Barcelona, Spain (2010)

17. Kirkpatrick, S., Gelatt Jr., C., Vecchi, M.: Optimization by simulated annealing. Science 220 (4598), 671–680 (1983)
18. Koza, J.: Genetic Programming: on the programming of computers by means of natural selection. Cambridge, MA: MIT Press (1992)
19. Larranaga, P., Lozano, J.: Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation. Norwell, MA: Kluwer (2001)
20. Li, J.: FGP: A Genetic Programming Based Financial Forecasting Tool. Ph.D. thesis, Department of Computer Science, University of Essex (2001)
21. Martinez-Jaramillo, S.: Artificial Financial Markets: An Agent Based Approach to Reproduce Stylized Facts and to study the Red Queen Efect. Ph.D. thesis, CFFEA, University of Essex (2007)
22. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of calculations by fast computing machines. Journal of Chemical Physics 21, 1087–1092 (1953)
23. Özcan, E., Bilgin, B., Korkmaz, E.E.: A comprehensive analysis of hyper-heuristics. Intell. Data Anal. 12(1), 3–23 (2008)
24. Poli, R., Langdon, W., McPhee, N.: A Field Guide to Genetic Programming. Lulu.com (2008)
25. Sapankevych, N., Sankar, R.: Time series prediction using support vector machines: A survey. Computational Intelligence Magazine, IEEE 4(2), 24 –38 (may 2009)
26. Sharma, V., Srinivasan, D.: Evolutionary computation and economic time series forecasting. In: Proceedings of the IEEE Conference on Evolutionary Computation. pp. 188–195. Singapore (25-28 September 2007)
27. Tsang, E., Martinez-Jaramillo, S.: Computational finance. IEEE Computational Intelligence Society Newsletter pp. 3–8 (2004)
28. Tsang, E., Li, J., Markose, S., Er, H., Salhi, A., Iori, G.: EDDIE in financial decision making. Journal of Management and Economics (2000)
29. Tsang, E., Markose, S., Er, H.: Chance discovery in stock index option and future arbitrage. New Mathematics and Natural Computation, World Scientific 1(3), 435–447 (2005)
30. Zhang, Q., Sun, J., Tsang, E.: Evolutionary algorithm with guided mutation for the maximum clique problem. IEEE Transactions on Evolutionary Computation 9(2), 192–200 (2005)