On AIE-ASM: A Software to Simulate Artificial Stock Markets with Genetic Programming *

Shu-Heng Chen[1], Chia-Husan Yeh[2], and Chung-Chih Liao[3]

[1] AI-ECON Research Center, Department of Economics, National Chengchi University, Taipei, Taiwan
[2] Department of Information Management, I-Shou University, Kaohsiung, Taiwan
[3] Graduate Institute of International Business, National Taiwan University, Taipei, Taiwan

Abstract. Agent-based computational economic modeling requires demanding work on computer programming. Usually, the publications as outcomes of running these programs do not provide readers with adequate information to permit replication of the experiments reported in the papers. This may generally make the findings or conclusions from the agent-based simulations hard to verify. As a result, to facilitate the growth of this research area, it is necessary for the authors to make their source codes available in a public domain. This paper is a documentation accompanying the software AIE-ASM, which is available on the website. The software is designed for simulating the agent-based artificial stock market based on a standard asset pricing model. Genetic programming, as a part of the software, is used to drive the learning dynamics of traders. An example based on the version of single-population genetic programming is demonstrated in this paper.

1 Introduction

One of the earliest and the most productive application of John Holland's legacy to economics is the agent-based modeling of artificial stock markets. This research agenda originated from Brain Arthur and John Holland about a decade ago, when an economic research unit was established at the Santa Fe Institute in New Mexico([2]). This research line can be considered as the legacy of John Holland in economics. Mainly, it replaces the representative, perfect-foresight agent in the *standard asset pricing model* with Holland's *artificial adaptive agents* ([11]). The first journal publication on the agent-based artificial stock market is [14]. It progressed slowly at the very beginning, but then came to it burgeoning stage over the last few years. However, this research line experiences a problem which is generally not shared by the conventional economic approaches, but is widely shared by the agent-based computational approach, i.e. *replicatability*.

* A full version of this supplementary document is forthcoming in S.-H. Chen (ed.) *Evolutionary Computation in Economics and Finance*, to be published by the Springer Verlga in the year 2001.

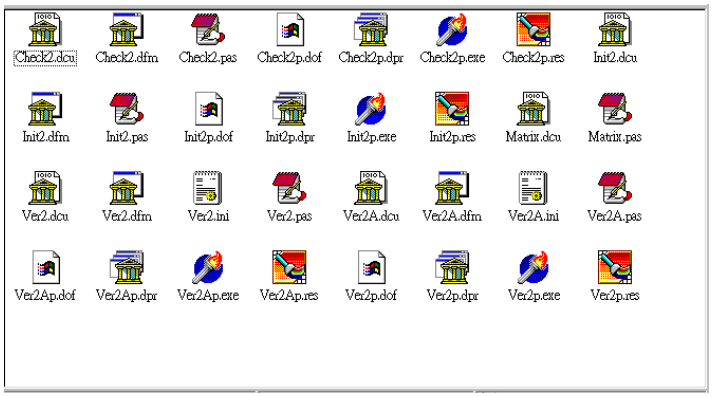

Fig. 1. An Overview of the Software: AIE-ASM

Agent-based computational economic modeling requires demanding work on computer programming. Usually, the publications as outcomes of running these programs do not provide readers with adequate information to permit replication of the experiments reported in the papers. This may generally make the findings or conclusions from the agent-based simulations hard to verify. As a result, to facilitate the growth of this research area, it is necessary for the authors to make their source codes available in a public domain. This paper provides documentation accompanying the software AIE-ASM, which is available on the website. The software is designed for simulating the agent-based artificial stock market based on a standard asset pricing model. *Genetic programming* (**GP**), as a part of the software, is used to drive the learning dynamics of traders. An example based on the version of *single-population genetic programming* is demonstrated in this paper. Other references related to these software can be found in [7] and [8].

2 AIE-ASM, Version 2: A User's Guide

One of the formidable tasks for the artificial stock market is the *design issue*. As [13] pointed out: "The computational realm has the advantages and disadvantages of a wide open space in which to design traders, and new researchers should be aware of the daunting design questions that they will face. Most of these questions still remain relatively unexploited at this time. (p.696)" Nevertheless, one should notice that this issue is not confined to the artificial stock market, and is widely shared by all research in *bounded rationality*. For example, [16] stated "This area is wilderness because the research faces so many choices after he decides to forgo the discipline provided by equilibrium theorizing. (p.2)"

```
BinarySet=+\-\*\%\
UnarySet=Exp\Log\Sin\Cos\Abs\Sqrt\
X_Var=X00\X01\X02\X03\X04\X05\X06\X07\X08\X09\X10\
Y_Var=Y00\Y01\Y02\Y03\Y04\Y05\Y06\Y07\Y08\Y09\Y10\
Extra_Var=Z00\Z01\Z02\Z03\Z04\
Const_Item=R

[Stock_Market]
H0=1.0
M0=100.0
Irate=0.1
Beta=0.001

[Traders]
Reproduct=50
Mutation=100
MutProb=3.3
MutMode=1
LeaveProb=5
DepthLimit=17
MaxExp=1700.0
LibraryProb=100
Lambda=0.5
G_Gap=1
TryNo=5
Theta1=0.5
Theta2=0.0001
Theta3=0.0133
Gen_No=20000
```

Fig. 2. The File "Ver2A.ini" in the Software "AIE-ASM"

Too many choices is a tough problem for us. There are *lots of parameters* in GP, and the results we obtained using the GP method might be sensitive to the particular parameter values chosen. We invite interested researchers to try what they may think sensitive, and provide a link below to the source code we used in [7].

The software **AIE-ASM** is available from the following web address:
http://www.aiecon.org/software.htm
The software is composed of 32 files as shown in Figure 1. Among them,

there are four executable files, namely, Ver2p.exe, Ver2Ap.exe, Init2p.exe and Check2p.exe. The parameters declared in the Table 1 of [8] are all stored in the file **Ver2A.ini** (Figure 2).

The dynamics of the stock market are initialized by using the initial values of the stock prices $\{P_{0-i}\}_{i=0}^{11}$ and the stock prices plus dividends $\{P_{0-i} + D_{0-i}\}_{i=0}^{11}$ specified in the file **p.txt**. To rerun a simulation with different initial values, one simply inputs a series of new data into the data file, p.txt. The time series of dividends $\{D_t\}$ follows a stochastic process which is exogenously given. Therefore, it is generated before the simulation and is stored in the file **Dstar.txt**. If the end user would like to test a different stochastic process, such as AR(1), she can simply generate a new series with the stated stochastic property and store it in the file Dstar.txt, and then rerun the program.

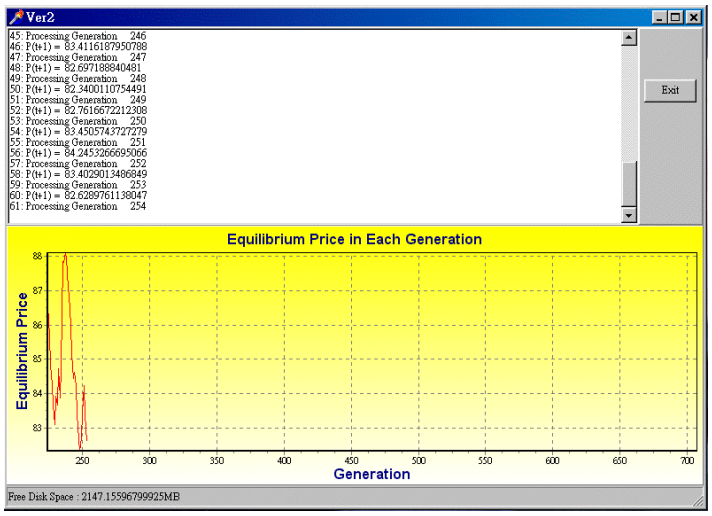

Fig. 3. The Results of a Sample Run Shown on the Screen

Before running the program, all executable files (.exe files) and parameter files (.ini files) have to be put under the same directory, and the place to store the data files (.txt files) should be declared in the parameter files. To run the program, simply use *mouse* to *click* the file Init2p.exe followed by clicking the file Ver2p.exe. One second after the execution of Ver2p.exe, Ver2Ap.exe will be executed automatically. Depending on the number of generations to evolve, the execution can take a lot of memory space. To see whether the execution is terminated unintendedly due to insufficient memory size, the user may like to click Check2p.exe after clicking Ver2p.exe. What Check2p.exe does is to check whether the execution is unintendedly terminated. If so, it will

Table 1. Time Series Outputs Generated from AIE-ASM: Data1.txt

Variables	Name in Data1.txt
Stock price	P_{t+1}
Dividends	D_{t+1}
Trading volumes	V_t
Number of buyers	BuyNo
Number of sellers	SellNo
Number of non-participants	BeSNo
Shares to buy	TotalBid
Shares to sell	TotalOffer
Average of depth of trees	AveDepth
Average of nodes of trees	AveNode
Average of shares held	AveHi
Mean of traders' expectations	AveVijs
Variance of depth of tress	VarDepth
Variance of nodes of trees	VarNode
Variance of shares held	VarHi
Variance of traders' expectations	VarVijs
Number of trades with successful search	SucceedNo
Number of martingale believers	MartingaleNo
Number of traders who decide to search	VisitGenNo
Number of traders who decide to search	VisitLibNo

VisitGenNo is available only in the case without the B-School, whereas No VisitlibNo is available only in the case with the B-school.

restart Ver2p.exe and continue the execution right from the point it was terminated so that you need not rerun the program, and the data generated before the termination will be kept. Generally speaking, we recommend the user to click this file if the number of generations to evolve is more than 2000. Once Ver2p.exe is executed, a time series plot of the stock price will be immediately shown on the screen (Figure 3).

In addition to the price series directly shown on the screen, there is a complete output file named **Data1.txt**. What has been stored in this file is the time series variables listed in Table 1. These variables are aggregate results. It is also possible to trace the whole population in an individual manner. The outputs about microstructure are stored in **Data2.txt**. To have a feeling of the evolution of traders' expectations and decisions, you can find in Figures 4 and 5 snapshots of the histogram of $E_{i,t}(P_{t+1} + D_{t+1})$ and $h_{i,t}$ of a single simulation by using the parameters in the Table 1 of [8].

Depending on the computing environment, running genetic programming can be very time-consuming. For example, for a typical run of SGP with business school, it takes 5 days for 20,000 generations on **AMD K6 233** and **Pentium II 233** with **128 MB RAM**. The computational time required is not a linear function of the number of iterations due to the fact that the

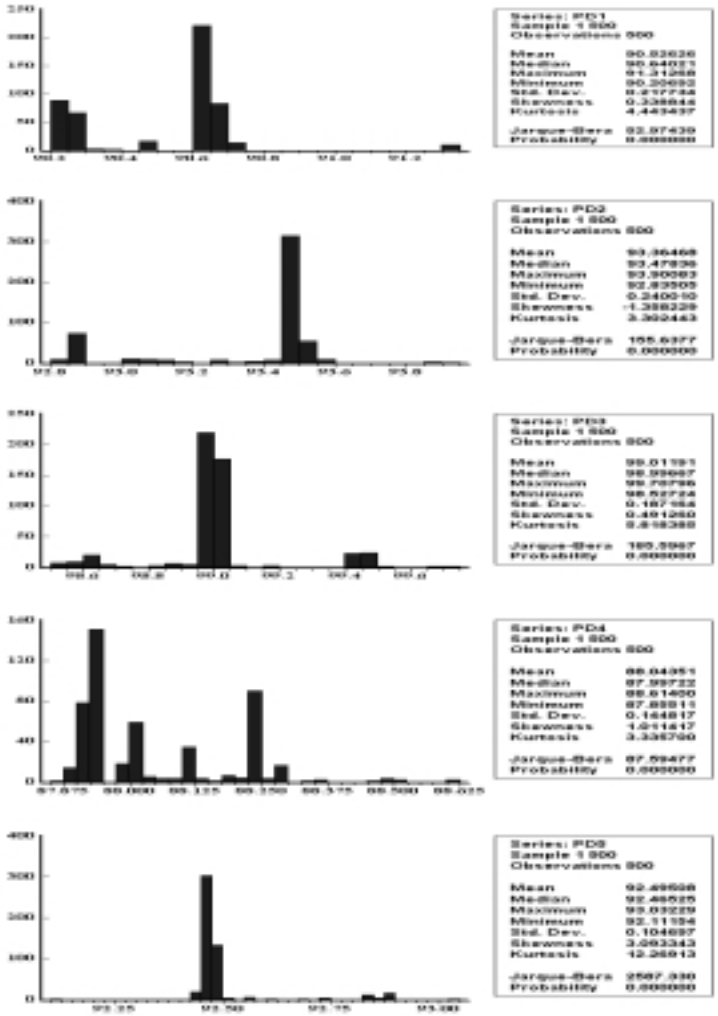

Fig. 4. Histogram of Traders' Expectations of the Price and Dividends

From the top to the bottom are the five snapshots of the histogram of traders' expectations on the 100th, 200th,...,500th trading day of a simulation based on the Table 1 of Chen and Yeh (2001b).

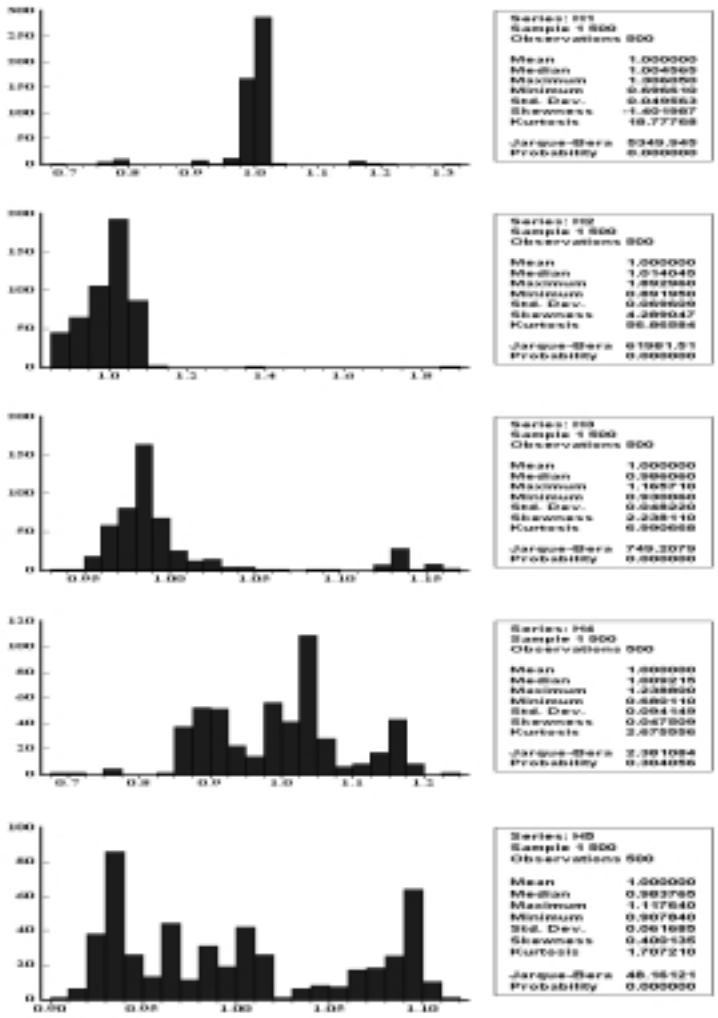

Fig. 5. Histogram of Traders' Portfolios

From the top to the bottom are the five snapshots of the histogram of traders' portfolios on the 100th, 200th,...,500th trading day of a simulation based on the Table 1 of Chen and Yeh (2001b).

tree size of the LISP program may increase with number of iterations (generations). This run-time speed limits the possibility of large-scale simulation. However, it is our belief that *one of the main contributions of this paper is to provide a public platform to facilitate more extensive research in this di-*

```
          op := GetOp(empty,Value);
        end;       (* end of while not empty *)

          if Top^.next <> nil then
        begin
          MessageDlg('Error: Data compute not match in T' ,mtWarning,[mbOK],0);
          exit;
        end;
        Vij := Top^.Value;

    if Abs(Vij) <= MaxExp then
        Vij := (Exp(Theta2*Vij)-Exp(-Theta2*Vij))/(Exp(Theta2*Vij)+Exp(-Theta2*Vij))
  //  if Abs(Vij) <= 10000.0 then
  //      Vij := Vij/10000.0
        else
        begin
          if Vij > 0.0 then
            Vij := 1.0
          else
            Vij := -1.0;
        end;
        Vij := DimC[u-1] * (1 + Theta1 * Vij);
     //   Vij := (DimB[u-1] + 10.0) * (1 + Theta1 * Vij);

        FreeStack(Top);
        except
          FreeStack(Top);
      end;
end;


procedure MakeFullTree(var t: TwoPoints;d,depth: integer);
var
    p          : TwoPoints;
    i,j,sign : byte;
    val        : real;
```

Fig. 6. Traders' Forecasting Equation in the File Ver2.pas

rection. Those interested in conducting further experiments are encouraged to download the software and give it a try.

AIE-ASM is quite user-friendly because most of the parameters can be directly modified through the file **Ver2A.ini**. For example, to simulate artificial stock market with and without the b-school[1], one can make a choice between these two versions, simply modify the parameter value of **LibraryProb** (Figure 2), which determines the probability of visiting the b-school if the trader decides to search. Setting this value at "0" means there is no chance of visiting the b-school, and hence the b-school is not included. Setting this value at "100" means that direct imitation from traders is infeasible, and that learning and adaptation is restricted to the b-school only. Anything between "0" and "100" makes a compromise between these two extremes.

The major files of the source code are **Ver2.pas** and **Ver2A.pas**. In some cases, the user needs to modify these two files to conduct her own experiments. For example, if the user wants to model traders' forecast in terms of level rather than increments (Equation (1)),

$$E_{i,t}(P_{t+1} + D_{t+1}) = (P_t + \mu)(1 + \theta_1 tanh(\theta_2 \cdot f_{i,t})). \tag{1}$$

[1] See "Search Processes" in the next section. For the details of the b-school, the interested reader is referred to [7].

then she needs to change the equation on line 796 contained in the procedure **ComputeVij** (line 680) and the one on line 631 in the procedure **ComputeHi** (line 511) (See Figure 6).[2]

3 Search Process without Business School

The search process to be detailed below will determine the outcome of the trader's search. Basically, it describes how *promising ideas* (*forecasting rules*) are popularized or how *new ideas* are discovered during traders' search and adaptive process, i.e., *the evolution of a population of ideas*. Recently, *genetic algorithms* (GAs) and *genetic programming* (GP) have been extensively used to substantiate processes like this. However, the straightforward application of single-population GAs or GP, which rests on the assumption that strategies are observable and imitable, has been seriously criticized by [10]. [7] considered a modified version of single-population GP by introducing a mechanism called *business school*. In the following, we will, however, describe the standard style of GP, i.e., the version without business school. For convenience, the former is called *a search process without business school* and the latter *a search process with business school*.

The search process without business school goes as follows. At each single search, with probabilities p_r, p_c, and $p_m (= 1 - p_r - p_c)$, the trader will *randomly* choose a *genetic operator* to search for an idea, where p_r is the probability of choosing the *reproduction* operator, p_c the probability of choosing *crossover*, and p_m the probability of choosing *mutation*. In the case where the reproduction operator is chosen, she will first randomly select two GP trees, say, $gp_{j,t}$ and $gp_{k,t}$ from GP_t, *the collection of all traders' forecasting rules at time t*. These rules are compared with each other based on their performance on accumulated profits over the last n_2 days. The one with higher increments is selected and is denoted by $gp^r_{i,t}$.[3] In the case of mutation, we

[2] For a lengthy discussion of using Equation (1), see [8]. Other possibilities are given as follows.

$$E_{i,t}(P_{t+1} + D_{t+1}) = P_t(1 + \theta_1 tanh(\theta_2 \cdot f_{i,t})) + \mu_d \tag{2}$$

$$\begin{aligned} E_{i,t}(P_{t+1} + D_{t+1}) = \; & P_t(1 + \theta_1 tanh(\theta_2 \cdot f^p_{i,t})) \\ + \; & D_t(1 + \theta_3 tanh(\theta_4 \cdot f^d_{i,t})) \end{aligned} \tag{3}$$

$$E_{i,t}(P_{t+1} + D_{t+1}) = \begin{cases} f_{i,t}, \text{if} \;\; (1-\theta_2)(P_t + D_t) \le f_{i,t} \le (1+\theta_1)(P_t + D_t), \\ (1+\theta_1)(P_t + D_t), \text{if} \;\; f_{i,t} > (1+\theta_1)(P_t + D_t), \\ (1-\theta_2)(P_t + D_t), \text{if} \;\; f_{i,t} < (1-\theta_2)(P_t + D_t) \end{cases} \tag{4}$$

[3] This is just the standard tournament selection style with tournament size 2.

follow the same procedure as reproduction except that $gp^r_{i,t}$ has a chance of being perturbed by *tree mutation* (See [12]), and the result is denoted by $gp^m_{i,t}$.

In the case of crossover, trader i first randomly selects two pairs of trees, say, $(gp_{j_1,t}, gp_{j_2,t})$ and $(gp_{k_1,t}, gp_{k_2,t})$ from GP_t. Then, as with the reproduction process, the better one of each pair is selected and further paired as parents. Two offspring, say $(gp_{j,t}, gp_{k,t})$, are born from the parents by the standard GP crossover. One of the two offspring is randomly selected and is denoted by $gp^c_{i,t}$. In sum, depending on the genetic operator used, $gp^*_{i,t}$ ($* = r, c$, or m) defines the outcome of a single search by trader i.

But, traders are not supposed to take whatever comes out of their search. Instead, trader i will first compare the performance of the new model with the old one, i.e., the one trader used at time t. If the accumulated profits of the new one are indeed better, then it will replace the old one.[4] In this case,

$$gp_{i,t+1} = gp^*_{i,t} \tag{5}$$

Otherwise, trader i will give up this newly-found rule and start another round of search, and she will do it again and again until either she finds a better one or she continuously fails I^* times. For the latter,

$$gp_{i,t+1} = gp_{i,t} \tag{6}$$

To wrap it up, a pseudo program is provided as follows (also see Figure 7).

Procedure [Trader's Search]

0. **begin**
   1. Calculate $\Delta W(gp_{i,t})$
   2. A = **Random**(R,C,M) with (p_r, p_c, p_m)
   3. If A = C, go to step (11).
   4. (gp_1, gp_2) = (**Random**(GP_t), **Random**(GP_t))
   5. Calculate $\Delta W(gp_1)$ and $\Delta W(gp_2)$.
   6. gp_{new} = **Tournament Selection** $(\Delta W(gp_1), \Delta W(gp_2))$
   7. If A = R, go to step (19).
   8. $gp_{new} \leftarrow$ **Mutation** (gp_{new})
   9. Calculate $MAPE(gp_{new})$
   10. Go to step (19)
   11. (gp^1_1, gp^1_2) = (**Random**(GP_t), **Random**(GP_t))
   12. (gp^2_1, gp^2_2) = (**Random**(GP_t), **Random**(GP_t))
   13. Calculate $\Delta W(gp^1_1)$ and $\Delta W(gp^1_2)$.
   14. Calculate $\Delta W(gp^2_1)$ and $\Delta W(gp^2_2)$.
   15. gp_1 = **Tournament Selection**$(\Delta W(gp^1_1), \Delta W(gp^1_2))$

[4] Such a procedure, known as the *election operator*, is first introduced by [1].

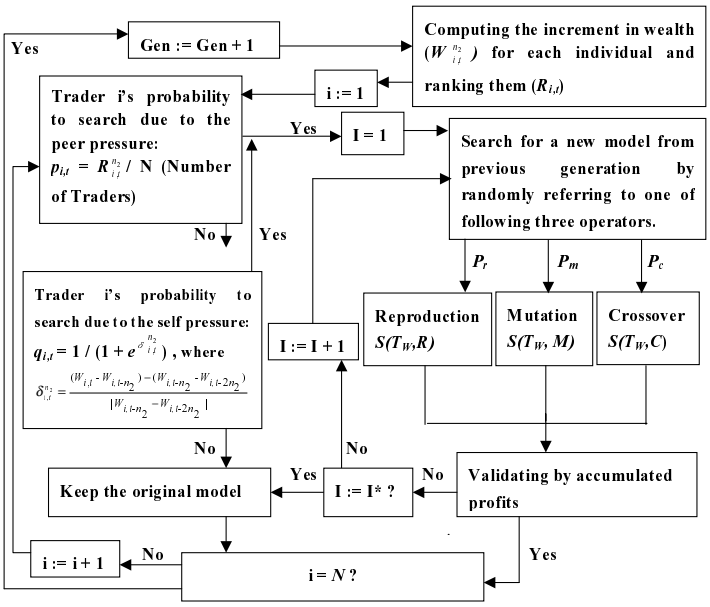

Fig. 7. The Flowchart of the GP-Driven Search Process

16. $gp_2 =$ **Tournament Selection**$(\Delta W(gp_1^2),\ \Delta W(gp_2^2))$
17. $(gp_1, gp_2) \leftarrow$ **Crossover** (gp_1, gp_2)
18. $gp_{new} =$ **Random**(gp_1, gp_2)
19. $gp_{i,t+1} =$ **Tournament Selection** $(\Delta W(gp_{i,t}), \Delta W(gp_{new}))$

20. **end**

4 An Example

In this section, we presents a few experimental designs based on the style of single-population genetic programming described in the previous section. In this case, the single-population genetic programming is directly operated on the society of traders. Two different designs are considered, which are identical in all aspects (control parameters) except for parameter n_2.

Parameter n_2 can be considered as *the time horizon of evolution*. In the evolutionary design proposed above, individual rules are assigned a fitness value at each round which is directly used to determine the probability of

their being passed down to the next generation, via reproduction, crossover, or mutation. When $n_2 = 1$, the rules are functions whose validity is estimated in a single shot (only one point). In this case, the number of generations is also the time scale of simulation. In other words, we are simultaneously evolving the population of traders while deriving the price P_t. Alternatively, the fitness may be collected over several rounds, rather than on one single trial, so that the "genetic" of the model is applied only after several rounds, during which the cumulative strength of the rules is the profits accumulated over several rounds. Here, for the single-shot case, n_2 is set at 1, while for the multi-shot case, it is set at 10. All parameters are summarized in Table 2.

For each value of n_2, we ran two experiments, and each experiment lasted for 20,000 periods. Figure 8 is time series plot of the stock price of each run. Under *full information* and *homogeneous expectations*, the *homogeneous rational expectations equilibrium price* (**HREE**) is[5]

$$P_t = \frac{1}{r}[\mu - \lambda\sigma_\xi^2(\frac{H}{N})] = \frac{1}{r}[\mu - \lambda\sigma_\xi^2 h]. \tag{7}$$

Since in our experiments (Table 1), $(\mu, \sigma_\xi^2, r, \lambda, h) = (10, 4, 0.1, 0.5, 1)$, the **HREE** price is 80. It seems difficult to predict from this reference price what actually happened in our artificial stock market. The dynamics of our markets are very wild. For example, in both experiments of $n_2 = 1$, we experienced a sequence of bubbles followed by crashes. The medians of these price series are 104.23, 102.88, 182.92, and 98.87 respectively, which are all higher than the **HREE** price.

To see whether these series satisfy *the efficient market hypothesis*, we first examined the *linear predictability* of the return series by using the Rissanen's *predictive stochastic complexity* (**PSC**) ([15]). The **PSC** criterion is a model selection criterion. It selects the model with the minimum PSC. By the PSC filtering algorithm, we can identify the linear ARMA model (p, q) of a series. If a series satisfies the **EMH**, then both p and q should be 0, i.e., there is no linear dependence, and hence the series is not linearly predictable. We applied the PSC algorithm to the ten equally-divided subseries of all four series, and the results are shown in Table 3.

From Table 3, almost all series are *linearly predictable* with a strictly positive p and q. Furthermore, if we run a regression based on the model with the selected order, the derived R^2 is non-trivial for most series. As a result, these series can hardly satisfy the *efficient market hypothesis* (**EMH**), and hence we cannot use these series to show the **EMH** as an emergent property. In fact, the feature of highly linear dependence is also rarely observed in real return series. Therefore, in addition to Harrald's criticism, the high R^2 of the return series also casts doubts on whether running single-population GP directly on a society of traders is a proper design for agent-based financial markets.

[5] See [3], pp. 40-41.

Table 2. Parameters of the GP-Based Artificial Stock Market: SGP without the B-School

The Stock Market	
Shares of the stock (H) per capita	1
Initial money supply (M_1) per capita	100
Interest rate (r)	0.1
Stochastic process (D_t)	IID $\sim$ Normal(10,4)
Price adjustment function	$tanh$
Price adjustment (β_1)	0.2×10^{-4}
Price adjustment (β_2)	0.2×10^{-4}
Traders	
Number of traders (N)	500
Degree of RRA (λ)	0.5
Criterion of fitness	Increment in wealth
Sample size of $\sigma^2_{t\|n_1}$ (n_1)	10
Evaluation cycle(n_2)	1, 10
Search intensity (I^*)	5
$(\theta_1, \theta_2, \theta_3)$	(0.5, 10^{-4}, 0.0133)
Genetic Programming	
Number of trees created by the full method	50
Number of trees created by the grow method	50
Function set	$\{+, -, \times, /, Sin, Cos, Exp, Rlog, Abs, Sqrt\}$
Terminal set	$\{P_t, P_{t-1}, ..., P_{t-10}, P_{t-1} + D_{t-1}, ..., P_{t-10} + D_{t-10}\}$
Selection scheme	Tournament selection
Tournament size	2
Probability of creating a tree by reproduction	0.10
Probability of creating a tree by crossover	0.70
Probability of creating a tree by mutation	0.20
Probability of mutation	0.0033
Probability of leaf selection under crossover	0.5
Mutation scheme	Tree Mutation
Maximum depth of tree	17
Number of generations	20,000
Number of trading days	20,000
Maximum number in the domain of Exp	1700
Criterion of fitness	Increment in Wealth

Earlier, we mentioned that SGP (GA) is not suitable for the architecture design due to Harrald's criticism. But, this does not rule out the attractiveness of this design because it may still generate series *behaving reasonably*

Table 3. Linear Dependence of the Series

Periods	Series 1-1		Series 1-2		Series 2-1		Series 2-2	
	PSC	R^2	PSC	R^2	PSC	R^2	PSC	R^2
1-2000	(0,2)	0.28	(1,0)	0.09	(0,0)	0.00	(0,0)	0.00
2001-4000	(1,0)	0.21	(1,0)	0.28	(2,3)	0.07	(1,0)	0.13
4001-6000	(1,0)	0.22	(1,0)	0.22	(2,2)	0.19	(0,1)	0.14
6001-8000	(1,0)	0.14	(1,0)	0.21	(2,2)	0.16	(1,0)	0.15
8001-10000	(1,0)	0.14	(1,0)	0.11	(3,3)	0.26	(1,0)	0.13
10001-12000	(1,0)	0.18	(1,0)	0.10	(3,3)	0.39	(1,0)	0.09
12001-14000	(1,0)	0.14	(1,0)	0.09	(6,6)	0.69	(1,0)	0.09
14001-16000	(1,0)	0.23	(1,0)	0.14	(0,0)	0.00	(1,0)	0.22
16001-18000	(1,0)	0.23	(1,0)	0.21	(3,4)	0.19	(0,2)	0.29
18001-20000	(0,1)	0.16	(1,0)	0.16	(3,3)	0.25	(2,2)	0.23

Each series has 20,000 observations. They are equally divided into ten non-overlapping periods, and the PSC algorithm is run for each period to select the linear ARMA order, i.e., (p, q). Once the order is selected, the R^2 is derived based on the model with the selected order.

close to the real return series. The evidence we have here, however, is not in favor of that possibility.[6]

5 A Summary of AIE-ASM Publications

In addition to the two journal articles ([5], [6]), simulations of this software can also be found in a few proceedings papers. They are available in the website:
http://econo.nccu.edu.tw/ai/staff/shc/vita.htm
or
http://aiecon.org/staff/shc/vita.htm

In addition to the two journal articles mentioned above there

[5] studied the effect of the search intensity resulting from *psychological pressure* on market efficiency. They found that increase in search intensity reduced the degree of heterogeneity of traders and made the the chance of successful search even more difficult. But, on the other hand, it also led to a reduction in market expectations' error.

[9] examined the possible explanations for the presence of the causal relation between stock returns and trading volume. They found that the bi-directional causality between trading stock returns and trading volume ubiquitously exists in all their four different experiments. The implication of this result is that the presence of the stock price-volume causal relation does not

[6] Of course, to show that *direct imitation* not only lacks behavioral foundation, but is also empirically implausible, one needs to run more experiments. We leave this as a future direction for research.

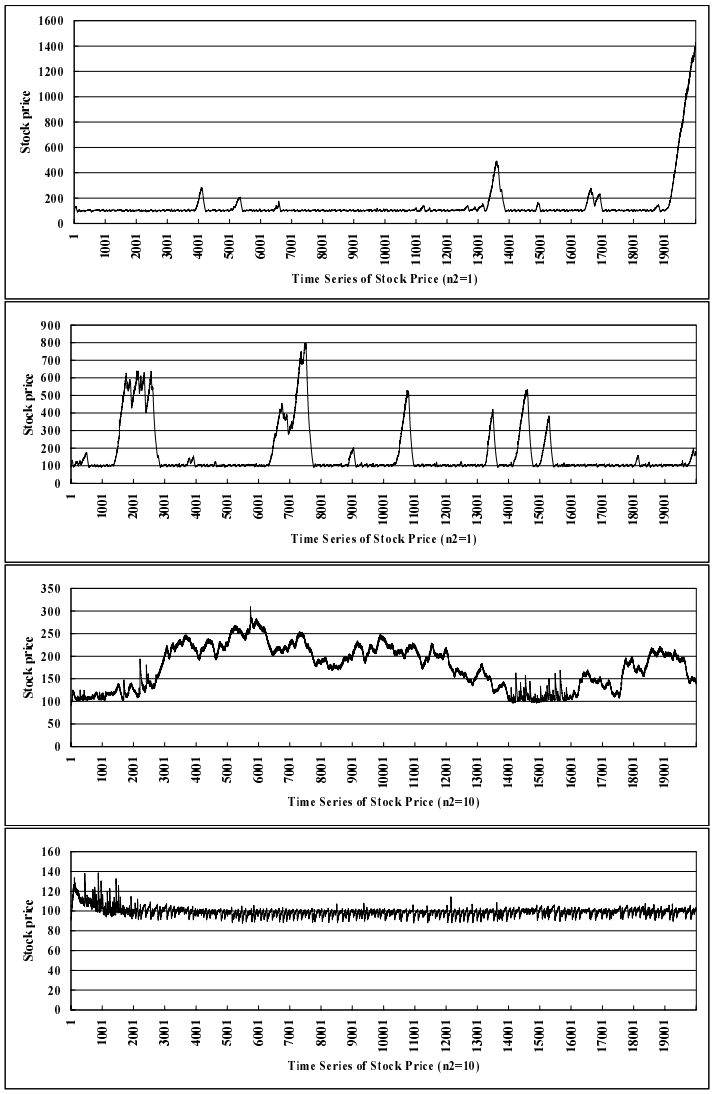

Fig. 8. Time Series Plot of the Stock Price: SGP without B-School

require any explicit assumptions like information asymmetry, reaction asymmetry, noise traders, or tax motives. It can be a generic property in a market modeled as evolving decentralized system of autonomous interacting agents.

[6] examined the significance of the *election operator*. By including or not including this operator, they considered two kinds of traders: the ones who

are "prudent" will validate a new idea before putting it into practice, and the ones who are "casual" will take whatever suggested (*follow the herd*). They found that markets with prudent traders and markets with casual traders could exhibit non-trivial differences in the size of speculative bubbles and price efficiency.

Finally, [4] studied the behavior of price discovery within a context of an *agent based stock market*. Via their agent based simulations, it was found that, except for some extreme cases, the mean prices generated from these artificial markets deviate from the *homogeneous rational expectation equilibrium* (**HREE**) *prices* no more than by 20%. Furthermore, while the HREE price should be a *deterministic* constant in all of our simulations, the artificial price series generated exhibited quite wild fluctuation, which may be connected to the well-known *excess volatility* in finance.

References

1. Arifovic J. (1994) Genetic Algorithm Learning and the Cobweb Model. Journal of Economic Dynamics and Control **18**, 3-28
2. Arthur B. (1992) On Learning and Adaptation in the Economy. SFI Working Paper, 92-07-038
3. Arthur W. B., Holland J., LeBaron B., Palmer R. and Tayler P. (1997) Asset Pricing under Endogenous Expectations in an Artificial Stock Market. In Arthur W. B., Durlauf S., Lane D. (Eds.), The Economy as an Evolving Complex System II, Addison-Wesley, 15-44
4. Chen S.-H., Liao C.-C. (2000) Price Discovery in Agent-Based Computational Modeling of Artificial Stock Markets. In Proceedings of the Second Asia-Pacific Conference on Genetic Algorithms and Applications. Global Link Publishing Company, Hong Kong, pp.380-387
5. Chen S.-H., Yeh C.-H. (2000a) On the Role of Intensive Search in stock Markets: Simulations Based on Agent-Based Computational Modeling of Artificial Stock Markets. In Proceedings of the Second Asia-Pacific Conference on Genetic Algorithms and Applications. Global Link Publishing Company, Hong Kong, 397-402
6. Chen S.-H., Yeh C.-H. (2000b) On the Consequence of "Following the Herd": Evidence from the Artificial Stock Market. In Arabnia H. R. (Ed.) Proceedings of the International Conference on Artificial Intelligence, Vol. II, CSREA Press, 388-394
7. Chen S.-H., Yeh C.-H. (2001a) Evolving Traders and the Business School with Genetic Programming: A New Architecture of the Agent-Based Artificial Stock Market. Journal of Economic Dynamics and Control **25**, 363-393
8. Chen S.-H., Yeh C.-H. (2001b) On the Emergent Properties of Artificial Stock Markets: The Efficient Market Hypothesis and the Rational Expectations Hypothesis. Forthcoming in *Journal of Economic Behavior and Organization*.
9. Chen S-H. Yeh C.-H. Liao C.-C. (2000), Testing for Granger Causality in the Stock-Price Volume Relation: A Perspective from the Agent-Based Model of Stock Markets. In Wang P. (Ed.) Proceedings of the Fifth Joint Conference on Information Sciences, Vol. II, 950-956

10. Harrald, P. (1998) Economics and Evolution. Panel Paper Given at the Seventh International Conference on Evolutionary Programming, March 25-27, San Diego, U.S.A.
11. Holland J. H., Miller J. H. (1991) Artificial Adaptive Agents in Economic Theory. American Economic Review: Papers and Proceedings **81(2)**, 365–370
12. Koza, J. (1992) Genetic Programming: On the Programming of Computers by Means of Natural Selection. The MIT Press.
13. LeBaron B. (2000) Agent-Based Computational Finance: Suggested Readings and Early Research. Journal of Economic Dynamics and Control **24**, pp. 679-702
14. Palmer R. G., Arthur W. B., Holland J. H., LeBaron B., Tayler P. (1994) Artificial Economic Life: A Simple Model of a Stockmarket. Physica D **75**, 264-274
15. Rissanen, J. (1989),"Stochastic Complexity in Statistical Inquiry". World Science, Singapore.
16. Sargent T. J. (1993) Bounded Rationality in Macroeconomics, Oxford.