Genetic Network Programming with Sarsa Learning and Its Application to Creating Stock Trading Rules

Yan Chen, Shingo Mabu, *Member, IEEE* Kotaro Hirasawa, *Member, IEEE* Jinglu Hu

***Abstract*— In this paper, trading rules on stock market using the Genetic Network Programming (GNP) with Sarsa learning is described. GNP is an evolutionary computation, which represents its solutions using graph structures and has some useful features inherently. It has been clarified that GNP works well especially in dynamic environments since GNP can create quite compact programs and has an implicit memory function. In this paper, GNP is applied to creating a stock trading model. There are three important points: The first important point is to combine GNP with Sarsa Learning which is one of the reinforcement learning algorithms. Evolution-based methods evolve their programs after task execution because they must calculate fitness values, while reinforcement learning can change programs during task execution, therefore the programs can be created efficiently. The second important point is that GNP uses candlestick chart and selects appropriate technical indices to judge the timing of the buying and selling stocks. The third important point is that sub-nodes are used in each node to determine appropriate actions (buying/selling) and to select appropriate stock price information depending on the situation. In the simulations, the trading model is trained using the stock prices of 16 brands in 2001, 2002 and 2003. Then the generalization ability is tested using the stock prices in 2004. From the simulation results, it is clarified that the trading rules of the proposed method obtain much higher profits than Buy&Hold method and its effectiveness has been confirmed.**

I. Introduction

Evolutionary Computation aims at producing the solutions in optimization problems or useful programs based on change, composition and selection. We have proposed Genetic Network Programming (GNP) [1], [2] as an extended method of Genetic Algorithm (GA) [3], [4] and Genetic Programming (GP) [5], [6]. It has been clarified that GNP is an effective method mainly for dynamic problems since GNP represents its solutions using graph structures, which contributes to creating quite compact programs and implicitly memorizing past action sequences in the network flows. Moreover, we proposed an extended algorithm of GNP which combines evolution and reinforcement learning [7] (GNP-RL). One of the advantages of GNP-RL is online learning. Original GNP is based on evolution only, therefore the programs are evolved mainly after task execution or enough trial, i.e., offline learning. On the other hand, GNP-RL can change its programs incrementally based on rewards obtained during task execution, i.e., online learning. For example, when an agent takes a good action with a positive reward at a certain state, the action is reinforced and therefore the same action will be adopted with higher probability when visiting the state again. Another advantage of GNP-RL is a combination of a diversified search of GNP and an intensified search of RL. The role of evolution is to make rough structures, i.e., plural paths of node transition, through selection, crossover and mutation. The role of RL is to determine one appropriate path in a structure made by evolution. Because RL is executed based on immediate rewards obtained after taking actions, intensified search, i.e., local search, can be executed efficiently. We call evolution as a diversified search just because evolution could change programs largely than RL with which the programs could escape from local minima.

In recent years, research on stock price prediction and trading model using evolutionary computation and neural networks has been done [8]-[10]. Generally speaking, methods for predicting stock prices and determining the timing of buying or selling stocks are divided into two groups: one is fundamental analysis which analyzes stock prices using the financial statement of each company, the economic trend and movements of the exchange rate; the other is technical analysis which analyzes numerically the past movement of stock prices. Generally, the research on stock price prediction and trading model using soft computing methods belong to technical analysis, so it determines the timing of buying and selling stocks based on the technical indices such as Relative Strength Index, Rate of Change, Golden Cross and so on. The proposed method also belongs to technical analysis.

In this paper, there are three important points. First, we combine GNP and Sarsa Learning [11] which is one of the reinforcement learning methods. Also Importance Index (IMX) and Candlestick Charts [12]-[13] are introduced for efficient stock trading decision making. Concretely speaking, Sarsa selects appropriate actions (buying/selling) and stock price information obtained from IMX and candlestick charts through the experiences during the trading. IMX and candlestick charts tell GNP whether or not the buying or selling signals are likely to appear at the current day. Second, although there are so many technical indices in the technical analysis, GNP with Sarsa can select appropriate indices and also select candlestick charts to judge the buying and selling timing of stocks. In other words, GNP with Sarsa could optimize the combinations of the information obtained by technical indices and candlestick charts. The third important point is that sub-nodes are introduced in each node to determine appropriate actions (buying/selling) and to select appropriate stock price information depending on the

Yan Chen, Shingo Mabu, Kotaro Hirasawa and Jinglu Hu are with the Graduate school of Information, Production, and Systems, Waseda University, 2-7, Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka(phone: 093-692-5261; fax: 093-692-5261; email: chenyan15153@akane.waseda.jp, mabu@waseda.jp,hirasawa@waseda.jp,jinglu@waseda.jp).

1-4244-1340-0/07/$25.00 ©2007 IEEE

situation.

This paper is organized as follows: In Section 2, the algorithm of the proposed method is described. Section 3 shows simulation environments, conditions and results. Section 4 is devoted to conclusions.

II. GNP with Sarsa (GNP-Sarsa) and Its Trading Algorithm

A. Basic structure of GNP-Sarsa

Fig.1 shows a basic structure of GNP-Sarsa and Fig. 2 shows judgment node and processing node structures. GNP-Sarsa consists of judgment nodes and processing nodes, which are connected to each other. Judgment nodes have if-then type branch decision functions. They return judgment results for assigned inputs and determine the next node. Processing nodes take actions (buying or selling stocks). While judgment nodes have conditional branches, processing nodes have no conditional branches. The role of a start node is to determine the first node to be executed. The graph structure of GNP has some inherent characteristics such as compact structures and an implicit memory function that contributes to create effective action rules. GNP-Sarsa has two kinds of time delays: time delays GNP-Sarsa spend on judgment or processing, and the ones it spends on node transitions. In this paper, the role of time delays is to determine the maximum number of technical indices and candlestick information to be considered when GNP-Sarsa determines buying or selling at a certain day. In detail, d_{ip} ($1 \leq p \leq m_i$, m_i is the number of subnodes in judgment and processing nodes) is the time delay spent on the judgment or processing at node i, while d_{ip}^A, d_{ip}^B,c are time delays spent on the node transition from node i to the next node. In this paper, d_{ip}^A, d_{ip}^B,c are set at zero time unit, d_{ip} of each judgment node is set at one time unit, d_{ip} of each processing node is set at five time units. In addition, we suppose that the trade in one day ends when GNP uses five or more time units. Therefore, the trade in one day ends when GNP executes fewer than five judgment nodes and one processing node, or five judgment nodes.

K_i represents the node type, K_i =0 means start node, K_i =1 means judgment node and K_i =2 means processing node. ID_i represents an identification number of the node function, e.g., K_i =1 and ID_i =2 mean the node is J_2. a_{ip} is a parameter which represents the threshold for determining buying or selling stocks in a processing node. Q_{ip} means Q value which is assigned to each state and action pair. In this method, "state" means a current node, and "action" means a selection of a sub-node (node function). In general reinforcement learning framework, the current state is determined by the combination of the current information, and action is an actual action an agent takes, e.g., buying or selling stocks. However, in GNP-Sarsa, the current node is defined as the current state, and a selection of a sub-node is defined as an action. d_{ip} is the time delay spent on judgment or processing. C_{ip}^A, C_{ip}^B,cshow the node number of the next node. d_{ip}^A, d_{ip}^B,c mean time delays spent on the transition from node i to node C_{ip}^A, C_{ip}^B, c respectively. Judgment node determines the upper suffix of the connection genes to refer to depending on the judgment result. If the judgment result is "B", GNP-Sarsa refers to C_{ip}^B and d_{ip}^B. Processing nodes always refer to C_{ip}^A and d_{ip}^A because processing nodes have no conditional branch.

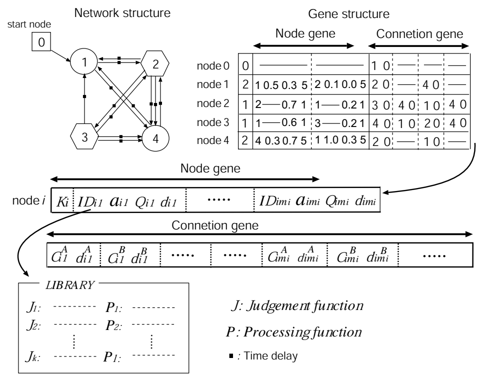

Fig. 1. Basic structure of GNP with Sarsa

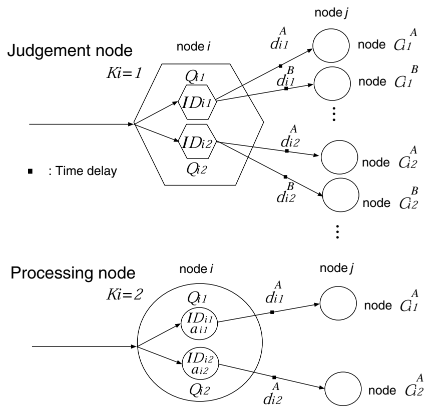

Fig. 2. Node structure

B. Judgment and Processing Functions of GNP-Sarsa

The node transition of GNP-Sarsa starts from a start node and continues depending on the node connections and judgment results. Fig. 2 shows node structures of a judgment node and a processing node.

(1) Judgment node: When a current node i is a judgment node, first, one Q value is selected from Q_{i1},c, Q_{im_i} based on ϵ -greedy policy. That is, a maximum Q value among Q_{i1},c, Q_{im_i} is selected with the probability of 1-ϵ, or a random one is selected with the probability of ϵ. Then corresponding function (ID_{ip}) is selected. The gene ID_{ip} shows a technical index or a candlestick GNP judges at node i. Each technical index has its own IMX function shown in Fig.3. x axis shows the value of each technical index,

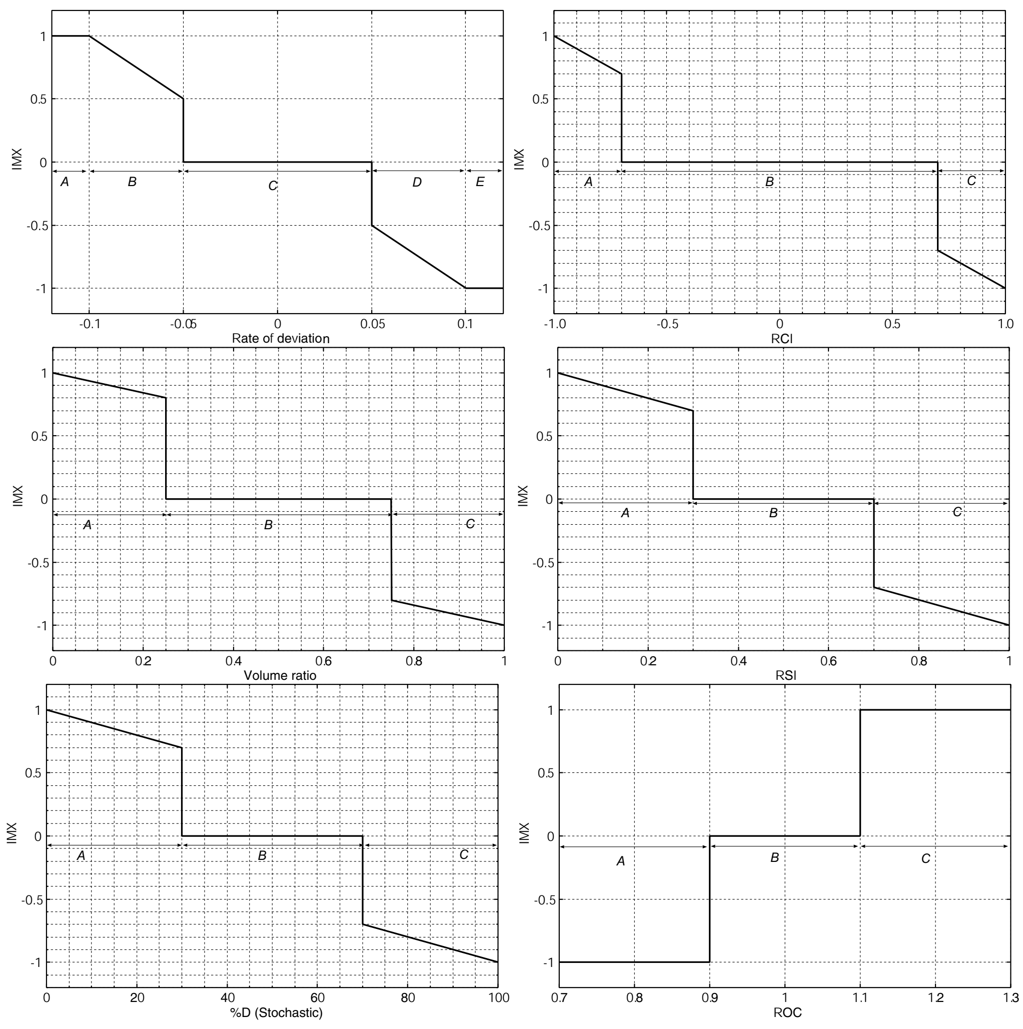

Fig. 3. IMX functions in Judgment nodes

and the sections A, B, C c correspond to judgment results. Suppose Q_{i1} and the corresponding ID_{i1} =1 (judgment of rate of deviation) are selected, and if the rate is more than 0.1, the judgment result becomes E, and the next node number becomes C_{i1}^{E}. y axis shows the output of the IMX function and it is used at a processing node. However, the IMX output of golden cross, dead cross and MACD could be 1, 0 or -1 based on the cross of the lines, and the values correspond to judgment results A, B and C, respectively. Concretely speaking, for three days after a golden cross appears, the IMX output becomes 1, and for three days after a dead cross appears, it becomes -1, otherwise 0. Furthermore, for three days after MACD passes through the signal from the lower side to the upper side, the IMX output becomes 1, and for three days after it does from the upper to the lower, the IMX output becomes -1, otherwise it becomes 0. Generally, golden cross indicates buying signals and dead cross indicates selling signals, therefore, buying signals become stronger as the IMX output is close to 1, and selling signals become stronger as it is close to -1.

In this paper, candlestick chart is used as one of judgment functions. As we know, candlestick chart has been winning international recognition for its good indication of stock prices, and it has been widely used as the means of indicating the fluctuations of the stocks. The proposed method has judgment nodes which check candlestick chart patterns. The judgment function of candlestick chart is executed as follows. When the selected sub-node has a judgment function of candlestick chart, GNP judges yesterday's candlestick and the candlestick of the day before yesterday. There are eight candlestick chart patterns as shown in Fig.4 according to two kinds of rules: (A) Judge whether there is a gap or not between yesterday's lowest price and the highest price of the day before yesterday, or between yesterday's highest price

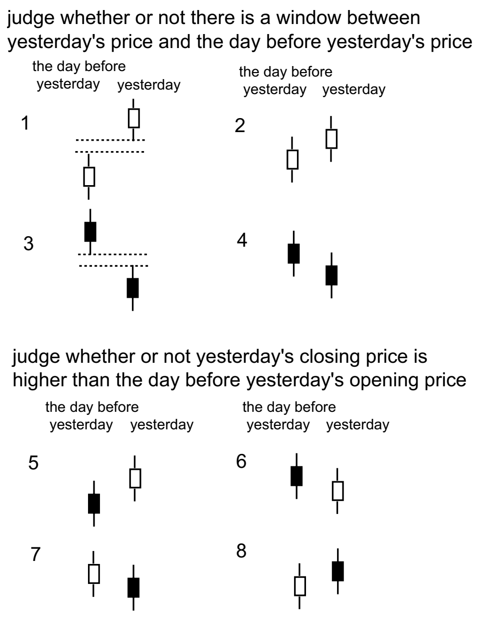
judge whether or not there is a window between yesterday's price and the day before yesterday's price
the day before yesterday
yesterday
1
2
3
4
judge whether or not yesterday's closing price is higher than the day before yesterday's opening price
5
6
7
8

Fig. 4. Candlestick Chart Patterns

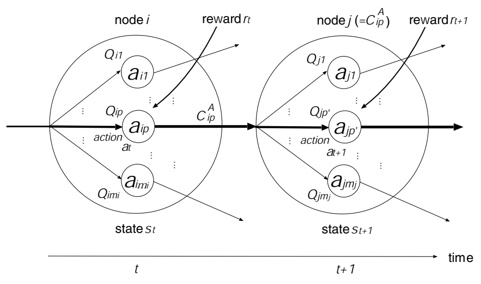

Fig. 5. An example of node transition

and the lowest price of the day before yesterday. (B) Judge whether or not yesterday's closing price is higher than the opening price of the day before yesterday. Especially, when the opening price equals to the closing price, the case is treated as black body candlestick. As an example, when the candlestick pattern is "3", GNP-Sarsa selects third branch to transfer to the next node. However, judgment nodes of candlestick chart do not have IMX function.

(2) Processing node: When a current node is a processing node, Q_{ip}, the corresponding ID_{ip} and a_{ip} are selected based on ϵ -greedy policy. The selected a_{ip} is a threshold for determining buying or selling stocks. We explain the procedure of buying and selling stocks using Fig. 5, where the current node at time t is a processing node.

1) First, one Q value is selected from Q_{i1} ,c, Q_{im_i} based on ϵ -greedy policy. That is, a maximum Q value among Q_{i1} ,c, Q_{imi} is selected with the probability of 1-ϵ , or a random one is selected with the probability of ϵ. Then the corresponding a_{ip} is selected.

2) Calculate an average of the IMXs obtained at the judgment nodes executed in the node transition from the previous processing node to the current processing node.

$$A_t = \frac{1}{|I'|}\sum_{i' \in I'} IMX(i')$$

where, I' shows a set of the judgment node numbers executed in the node transition from the previous processing node to the current processing node. IMX(i') shows an IMX output at node i' I'. However, when a judgment node of candlestick chart was executed or an IMX output is zero at a judgment node of golden cross, deadcross and MACD, the node number is excluded from I' for calculating A_t.

3) determine buying or selling:

In the case of ID_{ip} =0 (buy): if $A_t \geq a_{ip}$ and we do not have any stocks, GNP buys as much stocks as possible. Otherwise, GNP takes no action.

In the case of ID_{ip} =1 (sell): if $A_t < a_{ip}$ and we have stocks, GNP sells all the stocks. Otherwise, GNP takes no action.

4) The current node is transferred to the next node. If a_{ip} is selected, the next node number becomes C^A_{ip} .

The above procedure puts the information of the technical indices together into A_t, and GNP-Sarsa determines buying or selling stocks by comparing A_t with a_{ip}. Therefore, the points of this paper are 1) to find appropriate a_{ip} in the processing nodes by evolution and Sarsa, and 2) to determine I' by evolution, in other words, what kinds of judgments (technical indices and candlestick charts) should be considered is determined automatically.

C. Learning phase

First we explain Sarsa algorithm briefly. Sarsa can obtain Q values which estimate the sum of the discounted rewards obtained in the future. Suppose an agent selects an action a_t at state s_t at time t, a reward r_t is obtained and an action a_{t+1} is taken at the next state s_{t+1}. Then $Q(s_t, a_t)$ is updated as follows.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

α is a step size parameter, and is a discount rate which determines the present value of future rewards: a reward received k time steps later is worth only γ^{k-1} times of the reward supposed to receive at the current step.

As described before, a state means the current node and an action means the selection of a sub-node. Here, we explain the procedure for updating Q value in this paper.

1) At time t, GNP refers to Q_{i1},..., Q_{im_i} and selects one of them based on ϵ-greedy. Suppose that GNP selects Q_{ip} and the corresponding function ID_{ip}.

2) GNP executes the function ID_{ip}, gets the reward r_t and suppose the next node j becomes C^A_{ip}.

3) At time t + 1, GNP selects one Q value in the same way

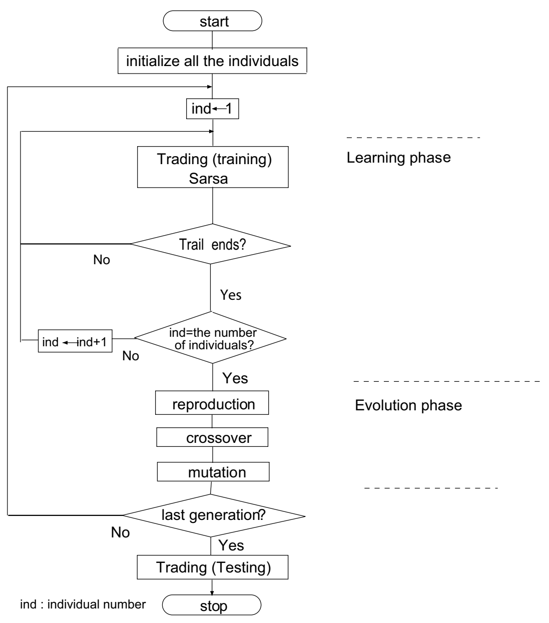

Fig. 6. Flowchart of GNP-Sarsa

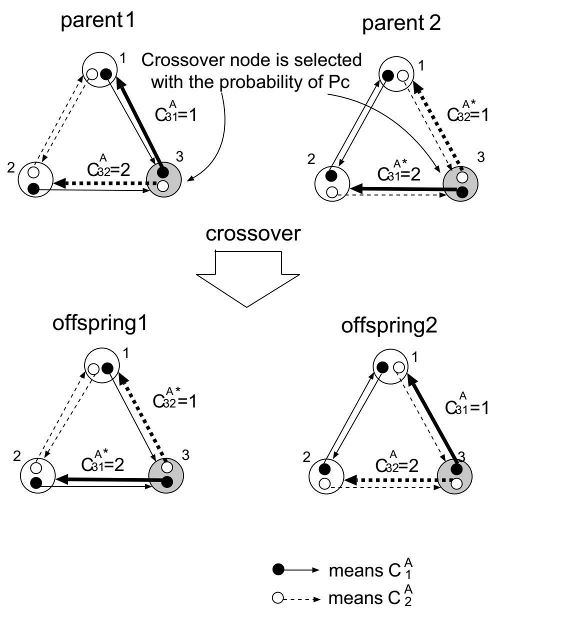

Fig. 7. Crossover

as step1. Suppose that $Q_{jp'}$ is selected.
4) Q value is updated as follows.

$$Q_{ip} \leftarrow Q_{ip} + \alpha[r_t + \gamma Q_{jp'} - Q_{ip}]$$

5) t $t + 1$, i j, p p' then return step 2.

D. Evolution phase

Fig.6 shows the whole flowchart of GNP-Sarsa. In this sub-section, the genetic operators in the evolution phase are introduced. The role of evolution is to change graph structures and randomly change node parameters a_{ip}.

1) Crossover: Crossover is executed between two parents and generates two offspring [Fig. 7]. The procedure of crossover is as follows.

1) Select two individuals using tournament selection twice and reproduce them as parents.

2) Each node is selected as a crossover node with the probability of P_c.

3) Two parents exchange the genes of the corresponding crossover nodes, i.e., the nodes with the same node number.

4) Generated new individuals become the new ones of the next generation.

Fig. 7 shows a crossover example of the graph structure with three processing nodes for simplicity. If GNP exchanges the genes of judgment nodes, it must exchange all the genes with suffix A,B,C,... simultaneously.

2) Mutation: Mutation is executed in one individual and a new one is generated [Fig. 8]. The procedure of mutation is as follows.

1) Select one individual using tournament selection and reproduce it as a parent.

2) Mutation operation

a) change connection: Each node branch (C^A_{ip}, C^B_{ip}, c) is selected with the probability of P_m, and the selected branch is reconnected to another node.

b) change parameters (a_{ip}): each a_{ip} is changed to other value with the probability of P_m.

c) change node function: Each node function (ID_{ip}) is selected with the probability of P_m, and the selected function is changed to another one.

3) Generated new individual becomes the new one of the next generation.

III. SIMULATION

To confirm the effectiveness of GNP-Sarsa, we carried out the trading simulations using 16 brands selected from the companies listed in the first section of Tokyo stock market in Japan (see Table 3). The simulation period is divided into two periods; one is used for training and the other is used for testing simulation.
Training: January 4, 2001—December 30, 2003 (737 days)
Testing: January 5, 2004—December 30, 2004 (246 days)
We suppose that the initial funds is 5,000,000 Japanese yen in both periods, and the order of buy or sell is executed at the opening of the trading day, i.e., we can buy and sell stocks with the opening price.

A. Fitness and Reward

Reward shows a capital gain of one trade (one set of buying and selling) and is used for learning. Fitness is the sum of the rewards obtained in the trading period.

Reward=selling price - purchase price

Fitness=Σ Reward

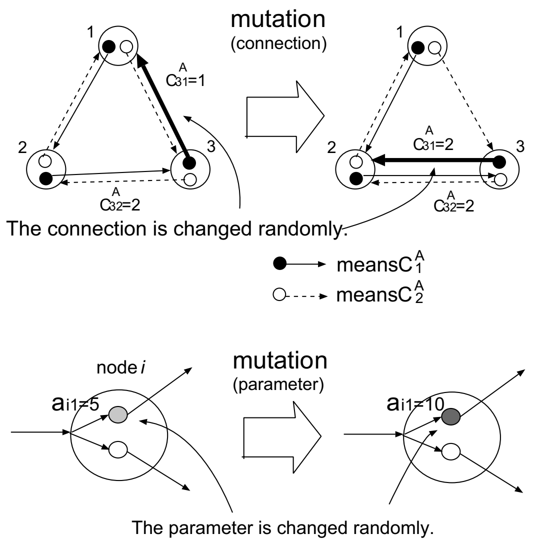

Fig. 8. Mutation

B. Conditions of GNP-Sarsa

GNP-Sarsa uses judgment nodes which judge the technical indices shown in Table I and candlestick chart. The technical indices are calculated using three kinds of calculation periods except Golden/Dead cross and MACD. Therefore, the number of kinds of judgment nodes is 21 (including one candlestick judgment). The number of processing functions is two: buying and selling. Table 2 shows simulation conditions. The total number of nodes in each individual is 31 including 10 processing nodes, 20 judgment nodes and one start node. However, the functions ID_{ip} in sub-nodes are determined randomly at the beginning of the first generation, and changed appropriately by evolution.

The initial connections between nodes are also determined randomly at the first generation. At the end of each generation, 179 new individuals are produced by mutation, 120 new individuals are produced by crossover, and the best individual is preserved. The other parameters are the ones showing good results in the simulations. The initial Q values are set at zero.

C. Simulation results

First, 300 individuals are evolved for 300 generations using the training data. Fig. 9 shows the fitness curve of the best individual at each generation in the training term using the data of Toyota motor, and the line is the average over 30 independent simulations. From the Figure, we can see that GNP-Sarsa can obtain larger profits for the training data as the generation goes on. The fitness curves of the other companies have almost the same tendency as that of Toyota Motor.

Next, the test simulation is carried out using the best individual at the last generation in the training term. Table 3 shows the profits and losses in the testing term. The values in Table 3 are the average of the 30 independent simulations with different random seeds. For the comparison, the table also shows the results of Buy&Hold which is often considered to be a benchmark in trading stocks simulations. Buy&Hold buys as much stocks as possible at the opening of the market on the first day in the simulations, and sells all the stocks at the opening on the last day. From the table, the proposed method can obtain larger profits than Buy&Hold in the trade of 12 brands out of 16. Especially, the stock prices of NEC, Fuji Heavy Ind., KDDI, Nomura Holdings, Shin-Etsu Chemical Co., Ltd. are down trend, so Buy&Hold always makes a loss, however the proposed method can obtain profits in five all brands.

Fig. 10 shows the change of the price of Toyota motor in the testing term and also shows typical buying and selling points by the proposed method. Fig. 11 shows the change of the funds as a result of the trading. From these figures, we can see that GNP-Sarsa can buy stocks at the lower points and sell at the higher points.

Fig. 12 shows the average ratio of the nodes used in the test period over 30 independent simulations in order to see which nodes are used and which are most efficient for stock trading model. The total number of node function is 23, while each processing node has a node number (0-1), and each judgment node has a node number (2-22). The x-axis shows the kinds of the nodes while the y-axis shows the average ratio of the used nodes. From the figure, we can see that the processing nodes are used to determine buying and selling stocks, and the judgment nodes of “Rate of deviation1” and “Volume ratio3” are frequently used.

Thus it can be said that GNP-Sarsa judges that these nodes are important to determine stock trading. GNP-Sarsa can automatically determine which nodes should be used in the current situation by evolving node functions and connections between nodes, in other words, GNP-Sarsa can optimize the combination of technical indices and candlestick charts used

TABLE I
CACULATION PERIODS OF THE TECHNICAL INDEX [DAY]

Technical index	period1	period2	period3
Rate of deviation	5	13	26
RSI	5	13	26
ROC	5	13	26
Volume ratio	5	13	26
RCI	9	18	27
Stochastics	12	20	30
Golden/Dead cross	5(short term)	26(long term)	
MACD	5(short term)	26(long term)	9(signal)

TABLE II
SIMULATION CONDITIONS

Number of individuals=300 (mutation:179, crossover:120, elite:1)
Number of nodes=31(Judgement node=20, Processing node=10, start node=1)
Number of sub-node in each node=2
P_c=0.1, P_m=0.02, α =0.1, γ =0.4, ϵ =0.1

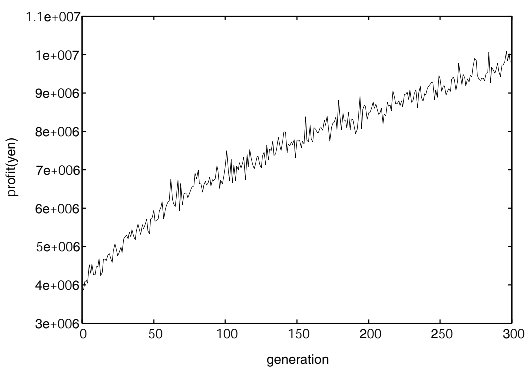

Fig. 9. Fitness curve in the training period (Toyota Motor)

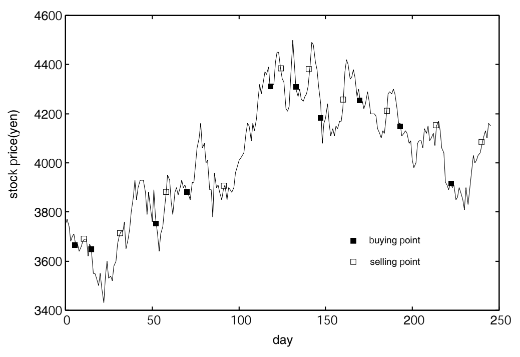

Fig. 10. Stock price of Toyota Motor and typical buying and selling point in 2004(test period)

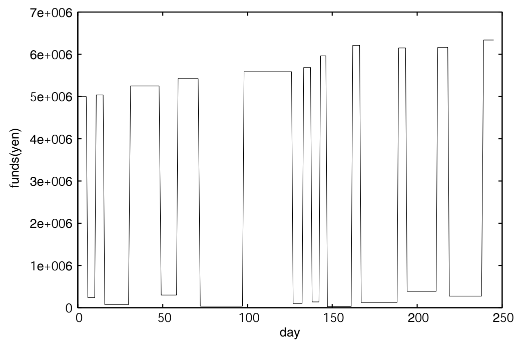

Fig. 11. Change of funds in the test simulation (Toyota Motor)

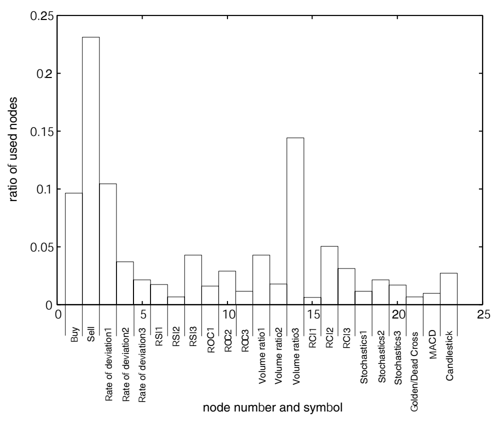

Fig. 12. Ratio of nodes used by GNP-Sarsa in the test period(Toyota Motor)

for stock trading model.

IV. CONCLUSIONS

In this paper, a stock trading model using GNP-Sarsa with important index and candlestick charts is proposed. First, a newly defined IMX function is assigned to each technical index to tell GNP-Sarsa whether buying or selling stocks is recommended or not. Second, Sarsa learns Q values to select appropriate sub-nodes/functions used to judge the current stock price information and determine buying and selling timing. We carried out simulations using stock price data of 16 brands for four years. From the simulation results, it is clarified that the fitness becomes larger as the generation goes on and the profits obtained in the testing term are better than Buy&Hold in the simulations of 12 brands out of 16. When there is downtrend, Buy&Hold makes a loss in five brands, but the proposed method can obtain profits in five all brands.

There remain some problems to be solved. First, in this paper, the calculation period of each technical index is fixed in advance. However, to improve the performance of the proposed method, we should develop a new method that can learn appropriate calculation periods. Next, it is necessary to consider the way of classifying the candlestick chart body type, and create more efficient judgment functions to judge current stock price appropriately. Also, we will evaluate the proposed method comparing with other methods using many

TABLE III
PROFITS IN THE TEST SIMULATIONS

Brand	Profit(profit rate[%])	
	GNP-Sarsa	Buy&Hold
Toyota Motor	522,333(10.4)	520,000(10.4)
Mitsubishi Estate	444,733(8.9)	664,000(13.3)
Showa Sell Sekiyu K.K.	263,100(5.3)	319,200(6.4)
East Japan Railway	413,833(8.3)	477,000(9.5)
NEC Corporation	36,600(0.7)	-1,026,000(-20.5)
Fuji Heavy Ind.	217,133(4.3)	-189,000(-3.8)
Sekisui House, Ltd.	582,466(11.6)	264,000(5.3)
Mitsu & Co.	473,033(9.5)	240,000(4.8)
Sony	148,733(3.0)	150,000(3.0)
Tokyo Gas	669,733(13.4)	372,000(7.4)
KDDI	199,400(4.0)	-576,000(-11.5)
Tokyo Electric Power	570,266(11.4)	262,500(5.3)
Daiwa House Industry	612,633(12.3)	32,000(0.6)
Nomura Holdings, Inc.	366,033(7.3)	-985,500(-19.7)
Shin-Etsu Chemical	562,700(11.3)	-264,000(-5.3)
Nippon Steel Cooperation	469,866(9.4)	399,000(8.0)

data of other brands.

REFERENCES

[1] S. Mabu, K. Hirasawa and J. Hu: *A graph-based evolutionary algorithm: Genetic network programming and its extension using reinforcement learning*, Evolutionary Computation, MIT Press, (to appear).

[2] T. Eguchi, K. Hirasawa, J. Hu and N. Ota: *study of evolutionary multiagent models based on symbiosis*, IEEE Trans. Syst., Man and Cybern. B, Vol. 36, No. 1, pp. 179-193 (2006)

[3] J. H. Holland: *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press (1975)

[4] D. E. Goldberg: *Genetic Algorithm in search, optimization and machine learning.* Addison-Wesley (1989)

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

[6] –: *Genetic Programming II, Automatic Discovery of Reusable Programs*, Cambridge, Mass.: MIT Press (1994)

[7] R. S. Sutton and A. G. Barto: *Reinforcement Learning - An Introduction*, Cambridge, Massachusetts, London, England: MIT Press (1998)

[8] N. Baba, N. Inoue and Y. Yanjun: *Utilization of soft computing techniques for constructing reliable decision support systems for dealing stocks*, in Proc. of International Joint Conference on Neural Networks (2002)

[9] J.-Y. Potvin, P. Soriano and M. Vallee: *Generating trading rules on the stock markets with genetic programming*, Computers & Operations Research, Vol. 31, pp. 1033-1047 (2004)

[10] K. J. Oh, T. Y. Kim, S.-H.Min and H. Y. Lee: *Portfolio algorithm based on portfolio beta using genetic algorithm*, Expert Systems with Application, Vol. 30, pp. 527-534 (2006)

[11] S. Mabu, H.Hatakeyama, M. T. Thu, K. Hirasawa and J. Hu: *Genetic Network Programming with Reinforcement Learning and Its Application to Making Mobile Robot Behavior*. IEEJ Trans. EIS, Vol.126, No.8, pp.1009-1015 (2006)

[12] K. H. Lee and G.S. Jo: *Expert system for predicting stock market timing using a candlestick chart, Expert Systems with Applications*, Vol.16, pp. 357-364 (1999)

[13] Y. Izumi, T. Yamaguchi, S. Mabu, K. Hirasawa and J. Hu: *Trading Rules on the Stock Market using Genetic Network Programming with Candlestick Chart*, 2006 IEEE Congress on Evolutionary Computation, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada, July 16-21, pp. 8531-8536 (2006)