

An easy and formal method for generating structural high leverage policy in system dynamics models

Showing H. Young, Associate Professor

Chia Ping Chen, Doctoral Student

Department of Business Management

National Sun Yat-Sen University, Kaohsiung, Taiwan

Fax: 886-7-5252367; E-mail: syoung@mail.nsysu.edu.tw

Abstract

Designing high leverage policy is a very crucial and challenging step in system dynamics approach. However, very limited formal methods were developed in this area. Literature showed three kinds of these methods: the algorithm method, the mathematical method, and the guideline method. The algorithm method is easy to use and suitable for nonlinear models, but can only obtain “parameter policy”, not “structural policy”. The mathematical method can obtain “structural policy”, but is not easy to use and not suitable for nonlinear models. The guideline method uses guidelines induced from some special cases to design policy; it is easy to use, but its generality is very weak. The objective of this research is to develop an easy and formal method for generating structural high leverage policy in system dynamics models. The idea of the method came from our experimental studies of microworlds. In which we observed that if the subjects repeatedly play a microworld by trial and error, they could often implicitly learn how to control the microworld even when they did not know the underlying structure. This kind of cognitive behavior is useful for controlling system dynamics models. So we imitate it to develop a conceptual framework for generating structural high leverage policy. And then we follow the conceptual framework to direct the development of our method. In short, this kind of cognitive behavior has two major activities: selecting information and organizing information. We adopt the genetic algorithm as the mechanism for selecting information. This algorithm is suitable for searching huge solution domain; probabilistically searching with natural selection, but without blind search; being able to obtain global satisfactory, not local optimal solution. We adopt the back-propagation algorithm as the mechanism for organizing information. This algorithm is a kind of artificial neural network with learning capability, and has showed its effectiveness in system control. The actual operation of this method is as follows. First, we use the conventional optimization algorithm to obtain the optimal trajectory of decision output, so called the open-loop solution. Then we use a hybrid software of genetic algorithm and back-propagation algorithm developed in this research to find out a decision function that produces approximation to the optimal trajectory, what is called close-loop solution. In the hybrid software, genetic algorithm is used to find out the independent variables of the decision function from the observable level variables in the system dynamics model, and back-propagation algorithm is used to generate the functional relationship among the independent variables. We have applied the method to the model developed by Forrester in his paper “Market Growth as Influenced by Capital Investment”. The result showed that the performance of the policy obtained by our method is better than that of Forrester’s policy.

Introduction

It seems crucial and challenging to design a high leverage policy for improving the system behavior in system dynamics area. However, very limited formal methods were developed in this area. This study developed a formal and easy method combining the genetic algorithms and the artificial neural network to obtain a high leverage closed loop policy.

Table 1 shows the position of the literature about the formal methods obtaining the high leverage solution in system dynamics. We positioned these according to two dimensions: one is the solution level, the other is the approach.

Table1: The category of the formal methods about designing the high leverage policy

Approach		Optimal algorithm	Mathematical method from the control theory	Guideline from simulation experiments
Solution level				
Open loop		*Burns & Malone(1974) *Bradford Group(e.g. Coyle, 1985)		
Closed loop	Wire	This study s method	*Talavage(1980) *Mohapatra & Sharma(1985) * Macedo(1989) *Ozveren & Stermann(1989)	
	Wire and flow			*Graham(1977) *Franco(1990)
System's boundary				

The three solution levels are open loop solution, closed loop solution and system's boundary. The open loop solution means that the solution function has not the variables from the system. If the system is fluctuated by some little impact, the open loop solution without information feedback can not adjust itself to the new state. So it's not robust. The closed loop solution means that the solution function has the variables from the system. The system's boundary solution means that the solution may lie outside the structure, or be generated from reconstructing the structure.

There are three kinds of approaches: the algorithm methods, the mathematical methods and the guideline methods. The algorithm methods are most friendly for users, and they can deal with the nonlinear system, but they can not obtain the closed loop solution. The mathematical methods almost originated from the control theory, they are difficult for the users without advanced mathematical background. They can

not directly deal with the nonlinear system, however they can obtain the closed loop solution. The guideline methods induce guidelines from some special cases; they are easy to use, but their generality is very weak.

From the users' viewpoint, the algorithms methods are easy to use, so this study's method will base on these, but can obtain the closed loop solution, that is, positioning on the gray cell in Table 1.

Method

Imitating human brain

According to our past research about human beings making decision in the microworld, we found the subjects' performance increased through many times practice in the microworld experiments, although they may not know the underlying structure of the microworld (e.g., Young et al., 1991; Wang and Young, 1992; Young et al., 1993; Young et al, 1994; Young and Wang, 1995; 1996). The finding shows that human brain seems to be an effective controller for the microworld with the characteristics of dynamics complexity. So this study's method will imitate the human brain.

Working hypothesis of human brain

We now construct a working hypothesis about the human brain how to operate in the microworld according to our observation in experiments and some literature (Forrester, 1961; 1964; 1968; Sterman, 1989; Morecroft, 1988). Then we will develop the study's method based on the working hypothesis.

Fig. 1-(a) shows the working hypothesis of human brain. There are two layers in human brain: one is the superficial cognitive activity; the other is the deep cognitive mechanism. Both layers are influenced by the whole system objectives, for example, the growth of system, the stable of system, or making a profit, and so on. We can identify and observe at least three superficial cognitive activities in the microworld experiments. They are screening, interpreting, and organizing in sequence. When human being interacted with the microworld, his brain screened, interpreted, and organized the information of the system state of the microworld, and then made decision, which led to changes of the system state of the microworld. The changed system state produced new information and make the interaction continue.

The underlying layer of the superficial cognitive activities is the deep cognitive mechanism. The behavior that the subjects adopted to improve performance in the microworld is a kind of implicit learning. The operation of neural network of human brain may account for the implicit learning by some research (e.g. Cleeremans, 1993). We take this hypothesis to develop this study's method.

The conceptual framework of the method

Fig. 1-(b) shows the conceptual framework of the method, which is based on the working hypothesis of human brain. In the conceptual framework, we use the genetic algorithm and the artificial neural network to serve as the underlying mechanism of the screening activity and the organizing activity respectively.

The genetic algorithm is a kind of the optimal search algorithms. It imitates the natural evolution and hereditary to search the potential solution space and obtain the optimal solution. The potential solution space is analogous to the population of some being. The objective function is analogous to the natural environment. According to the principle of evolution- survival of the fittest, the survival individuals can produce their child generation through mating, crossover, and mutating. After some number of generations, the process converges. The best individual hopefully represents the optimal solution. The advantage of the genetic algorithm is that it is suitable for searching huge solution domain; probability searching with natural selection, but without blind search; being able to obtain global satisfactory, not local optimal solution (the reader is referred to Goldberg (1989) for further details of the genetic algorithm).

The artificial neural network imitating the neural network of the human brain consists of numbers of the processing elements. The processing element imitating the neuron is the most basic operator in the artificial neural network. Its operation is to sum up all inputs value weighting by each connecting weight, and then transfer the sum value to an output value. This study adopted the back-propagation algorithm--a kind of artificial neural network. The network architecture of this algorithm is that the network consists of numbers of layers, and the layer consists of numbers of the processing elements. The layers between input and output of the network are called the hidden layers. Every processing elements of the same layer can not connect each other. The stimulus information propagates forward element by element through the network, and emerges at the output layer of the network as an output value. The error information is the gap between the output value and the desired value, and propagates backward layer by layer through the network, and adjusts every connecting weight. The advantage of the back-propagation algorithm is that it can serve as a substitute of human brain; its effectiveness for system control is supported by some evidences; and it can approximate any continuous function to map input value to desired output value (the reader is referred to Haykin (1994) for further details of the artificial neural network).

The operational framework of the method

The learning of back-propagation algorithm is a kind of supervised learning. The supervised learning must provide the neural network with a set of desired sample for adjusting output error. However, the conceptual framework does not have the mechanism to produce the target decision value. So we add the optimal algorithm into the conceptual framework to resolve above problem. Fig. 1-(c) shows the operational framework of the method, which is modified from the conceptual framework.

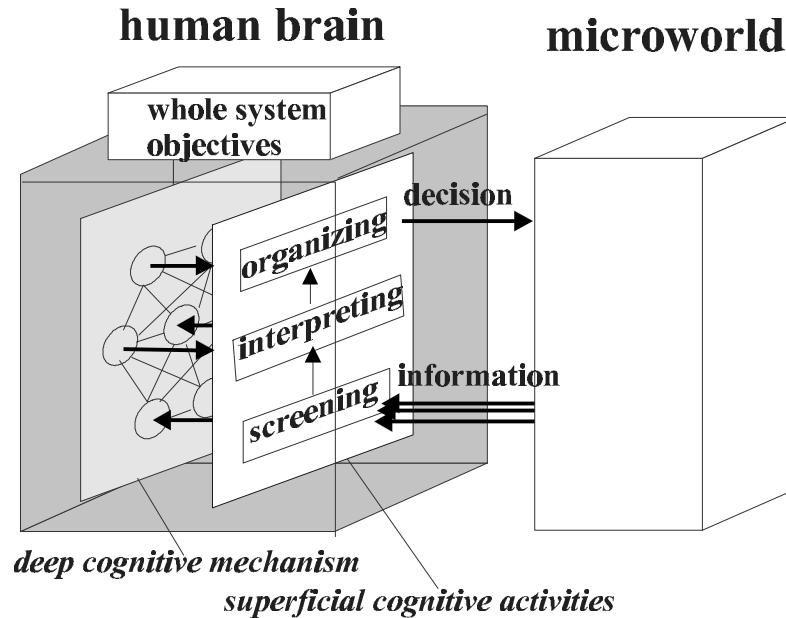


Fig.1-(a): The working hypothesis about human brain interacting with microworld

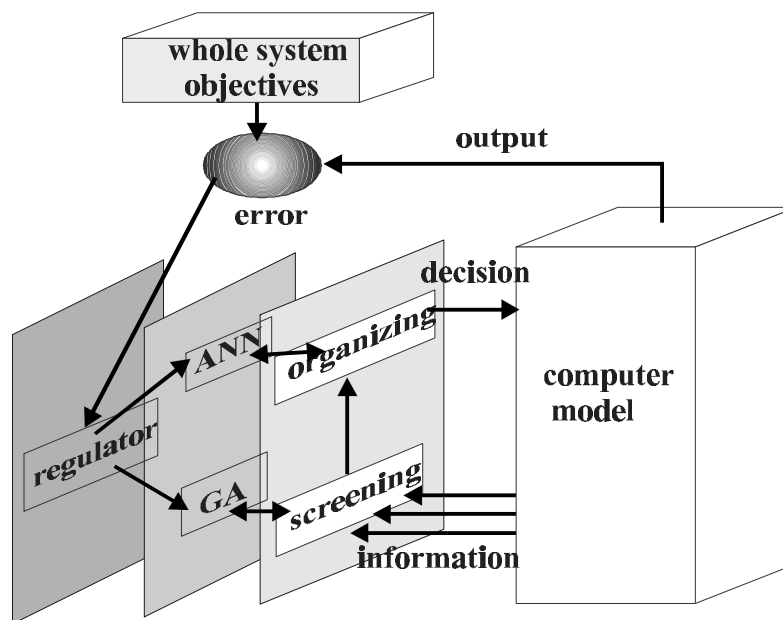


Fig.1-(b): the conceptual framework of the method

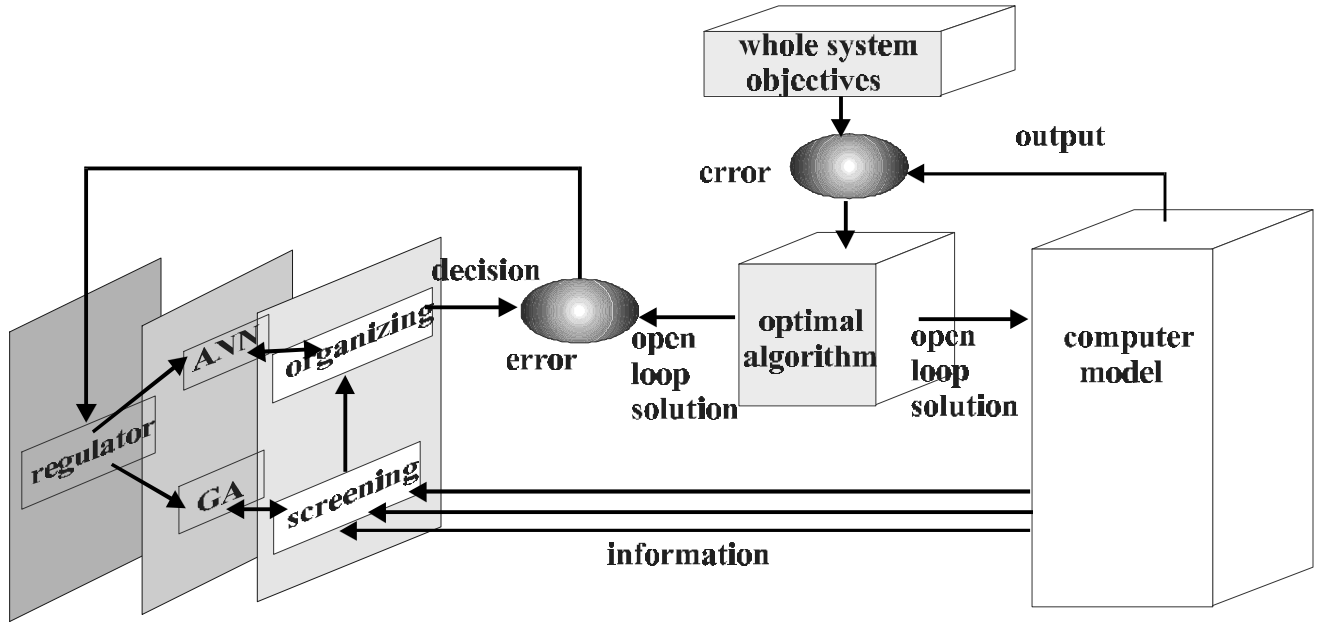


Fig.1-(c): the operational framework of the method

The operational process--two stage

According to the operational framework, we can further develop the operational process. There are two stages in the process. The first stage is to obtain the open loop solution of the system by the optimal algorithm. The second stage is to obtain the closed loop solution based on the open loop solution by GNN (Genetic Neural Network software, combining genetic algorithm and artificial neural network. We will discuss more about it later).

How to obtain the open loop solution

There are two kind of the open loop solution: the trajectory solution, and the constant value. In order to obtain the trajectory open loop solution, we adopt Fourier series to substitute original policy acting on the sensitive variable. The Fourier series can approximate any function. We use Powell algorithm (Vensim provides the function) to estimate the optimal parameter value of the Fourier series.

As to the objective function, we use L_2 -gain as the objective function of the stable oriented models. L_2 -gain is defined as Eq.(3), in which the numerator represents the norm of output variable, and the denominator represents the norm of input

variable. If L_2 -gain is less than 1, then the system between input and output dissipates energy (Schaft, 1992). According to Lyapunov's direct method, if a system dissipates energy, then the system will approach stable. We use conventional norm H_2 as the objective function of the growing oriented models. Eq.(1) is the objective function, the first term of the right-hand side represents the profit, the second represents the cost, and the third represents the growth potentiality.

How to obtain closed loop solution

We use GNN based on the open loop solution to obtain the closed loop solution, which mathematical implication is shown in Fig.2. Given the desired trajectory of the decision point and the optimal trajectory of the other variables in the system, the problem is to obtain some variables and the functional relationship between these variables and the decision variable. So we develop GNN to solve the problem.

The illustration of GNN

We combined the genetic algorithm and the artificial neural network to develop GNN software. Fig.3 shows the calculating process. Firstly, a group of sets of variables are randomly produced to be the variables of potential closed loop solution function by the genetic algorithm. Then the artificial neural network adjusts the functional relationship among the variables until the error between the function output and the desired decision value can not be decreased. Last, the genetic algorithm selects some sets of variables from original group according to the final error of each function. Meanwhile, the genetic algorithm uses these sets of variables to produce another sets of variables through mating, crossover, and mutating. Then GNN continues to run next cycle until the genetic algorithm converged.

We can use an imaginary experiment to illustrate the calculating process of GNN. Let us imagine that a group of subjects play the same microworld, but they are restricted to look at different sets of information. These subjects play repeatedly until their performance can not be improved. The researcher selects some sets of information based on the subject's performance, and changes these to new sets of information through the operations of mating, crossover, and mutating. Then the experiment uses new subjects and new sets of information to run next iteration.

When we obtain the variables of the closed loop solution function by GNN, we can base on the variables and the open loop solution to obtain the connecting weights of the variables by any artificial neural network software. With the variables, the connecting weights, and the network architecture, we can construct a high leverage policy.

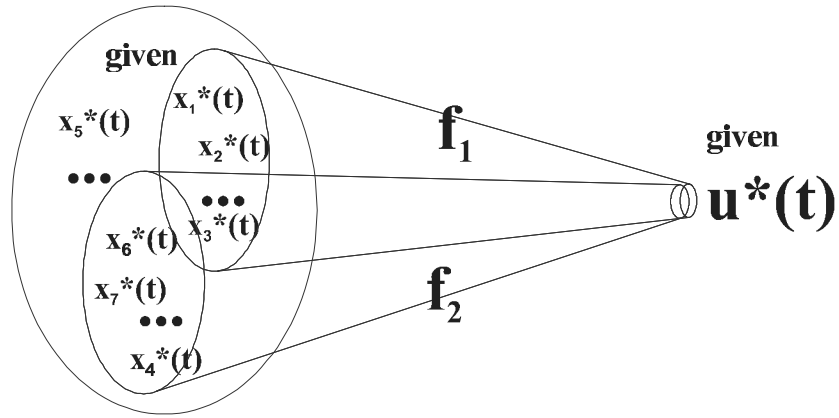


Fig.2: The mathematical implication of the method about the second stage

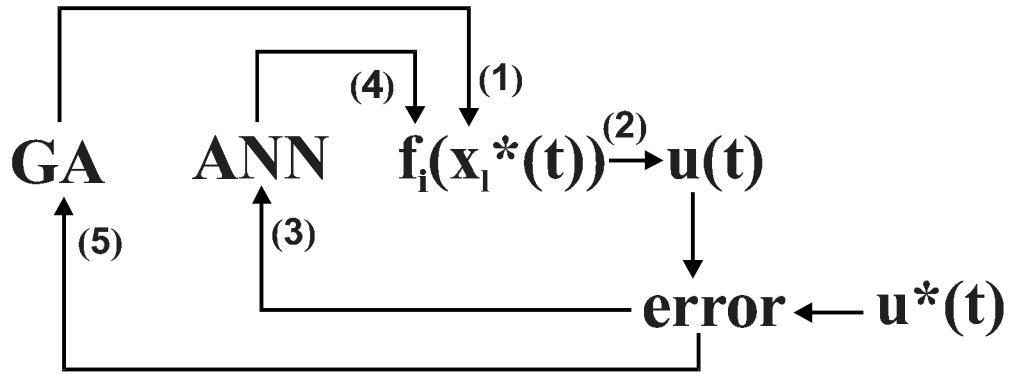


Fig.3: The calculating process of GNN

Result

In the following we apply the method to two different types of model: the growing oriented model--Forrester's market growth model (Forrester, 1968), and the stable oriented model--Forrester's customer-producer-employment model (Forrester, 1961). The operational process and every step results are summarized in Table 2.

Table 2: Operational process and each step results

Stage	General operational process	Growth Model: Forrester's market growth model	Stable Model: Forrester's customer-producer-employment model
Open Loop	Select one sensitive decision variable. Cancel the policy acting on the decision variable.	PCR: production capacity ordering	LDF: labor desired at factory
	Set up the objective function.	Eq.(1)	Eq.(3) or Eq.(4)
	Use Fourier series to replace original policy acting on the variable.	Eq.(2) and T=150	Eq.(2) and T=350
	Use Powell algorithm to estimate the optimal parameter value of the Fourier series.	Powell and Multiple_start =Vector $A_0, \dots, A_7=0 \in [0, 1e+5]$ $B_1, \dots, B_7=0 \in [-1e+5, 0]$	Powell $A_0=750 \in [400, 1200]$ $A_1, \dots, A_7, B_1, \dots, B_7=0 \in [-20, 20]$
	Obtain some open loop solutions by different methods or different objective functions.	Fig.4	Table 4
	Select a satisfactory open loop solution.	The simulation output of the heuristic mathematical method (Young and Chen, 1998) is best.	The simulation output of the objective function: "min gain(IAFPC)" is best.
Closed Loop	Preprocess the data of the selected variables from open loop solution for GNN.	Selecting 24 variables Variable transformation: $y_i(t)=\ln(x_i(t))$ $z_i(t)=(y_i(t)-y_i\text{mean})/y_i\text{s.d.}$ $w_i(t)=1/(1+e^{-z_i(t)})$ the training data set: 73 patterns the testing data set: 145 patterns delete former 5 patterns	Selecting 47 variables Variable transformation: $z_i(t)=(x_i(t)-x_i\text{mean})/x_i\text{s.d.}$ $w_i(t)=0.1+[(z_i(t)-z_i\text{min})/(z_i\text{max}-z_i\text{min})](0.9-0.1)$ the training data set: 176 patterns the testing data set: 351 patterns
	Set up the parameter value of GNN. Run GNN.	Table 5	Table 5
	Select the set of variables with best performance from the result of GNN.	GNN converged at the 11 th generation. From the 11 th to 80 th , total 5600 sets, we selected the best set of variables: BL, DRA, DRC, SEDM, and revenue.	GNN converged at the 8 th generation, and again diverged at the 21 th . From the 8 th to 20 th , total 1222 sets, we selected the best set of variables: DFOF, EDPC, FRFIF, LLF, and RMPAF.
	Obtain the connecting weights of the variables by any ANN software. Construct a potential high leverage policy with the variables, the connecting weights, and network architecture.	Fig. 5	Fig. 6
Test	Reproducing test	Fig. 7: good	Table 6: good
	Robustness test	Fig. 8: good	Table 7: not good

Table 3: Equations indicated in Table 2

$$\max J = \int_0^{150} (DRA \times 100)^2 dt - \int_0^{150} (PCO \times 1000)^2 dt - \int_0^{150} (SEM - SE)^2 dt \quad (1)$$

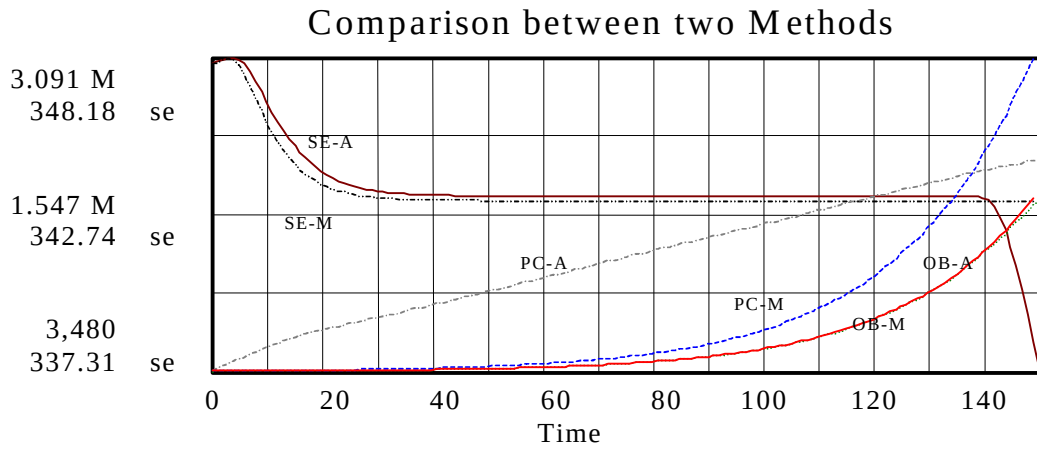
$$\begin{aligned} \text{Fourier function} = & \frac{A_0}{2} + A_1 \times \cos\left(\frac{t \times 2 \times p}{T}\right) + A_2 \times \cos\left(\frac{t \times 4 \times p}{T}\right) \\ & + A_3 \times \cos\left(\frac{t \times 6 \times p}{T}\right) + A_4 \times \cos\left(\frac{t \times 8 \times p}{T}\right) + A_5 \times \cos\left(\frac{t \times 10 \times p}{T}\right) \\ & + A_6 \times \cos\left(\frac{t \times 12 \times p}{T}\right) + A_7 \times \cos\left(\frac{t \times 14 \times p}{T}\right) + B_1 \times \sin\left(\frac{t \times 2 \times p}{T}\right) \\ & + B_2 \times \sin\left(\frac{t \times 4 \times p}{T}\right) + B_3 \times \sin\left(\frac{t \times 6 \times p}{T}\right) + B_4 \times \sin\left(\frac{t \times 8 \times p}{T}\right) \\ & + B_5 \times \sin\left(\frac{t \times 10 \times p}{T}\right) + B_6 \times \sin\left(\frac{t \times 12 \times p}{T}\right) + B_7 \times \sin\left(\frac{t \times 14 \times p}{T}\right) \end{aligned} \quad (2)$$

$$\min \text{gain}(x) = \sqrt{\frac{\int_0^{350} (x(t) - 100)^2 dt}{\int_0^{350} (INPPC(t) - 100)^2 dt}} \quad (3)$$

$$x(t) = \text{BLCPC}(t) \text{ or } \text{CASPC}(t) \text{ or } \text{DQDFPC}(t) \text{ or } \text{IAFPC}(t) \text{ or } \text{MENPC}(t)$$

or

$$\max NPTDF = \int_0^{350} NPRF(t) dt \quad (4)$$



-M: the heuristic mathematical method; -A: this study's method

Fig.4: Comparison of the growth model's open loop solutions between this study's method and the heuristic mathematical method developed by Young and Chen (1998).

Table 4: Comparison of the stable model's open loop solutions with different objective functions

result objective	NPTDF	gain BLCPC	gain CASPC	gain DQDFPC	gain IAFPC	gain MENPC
max NPTDF	4710010.5	4.5281	1112.3	2.5858	567.34	11.633
min gainBLCPC	3508531.5	0.2828	3.1322	0.2506	0.6646	0.4694
min gainCASPC	3526868.5	1.1696	0.9483	0.428	1.1432	0.5872
min gainDQDFPC	3546074	0.8504	1.9827	0.259	0.7031	0.5127
min gainIAFPC	3521606.3	0.7356	1.9707	0.2096	0.5277	0.4281
min gainMENPC	3560326.5	9.2369	14.112	5.1947	6.8034	0.0036

Table 5: The parameter value of GNN

GNN parameter		models	The growth model	The stable model
the parameter of genetic algorithms				
population size			80	94
chromosome length			24	47
generation number			80	25
mating probability *crossover probability			0.7*0.6	0.9*0.9
mutation probability			0.01	0.03
chromosome type	fixed or not?		fixed	fixed
	fixed number (fixed)		7	7
	penalty multiplier (fixed)		1.2	1.2
reproduction methods	Proportional: convergent pressure		3	1
	rank-based: no.1 number			
the parameter of neural network				
learning rate			0.05	0.1
momentum parameter			0.01	0
noise factor			0	0
stop criteria	error tolerance		0.001	0.001
	maximum learning cycles		100	500
initialized number			3	1
network architecture-layer number			3	3
neurons number of input layer			24	47
neurons number of hidden layers			3	3
neurons number of output layer			1	1

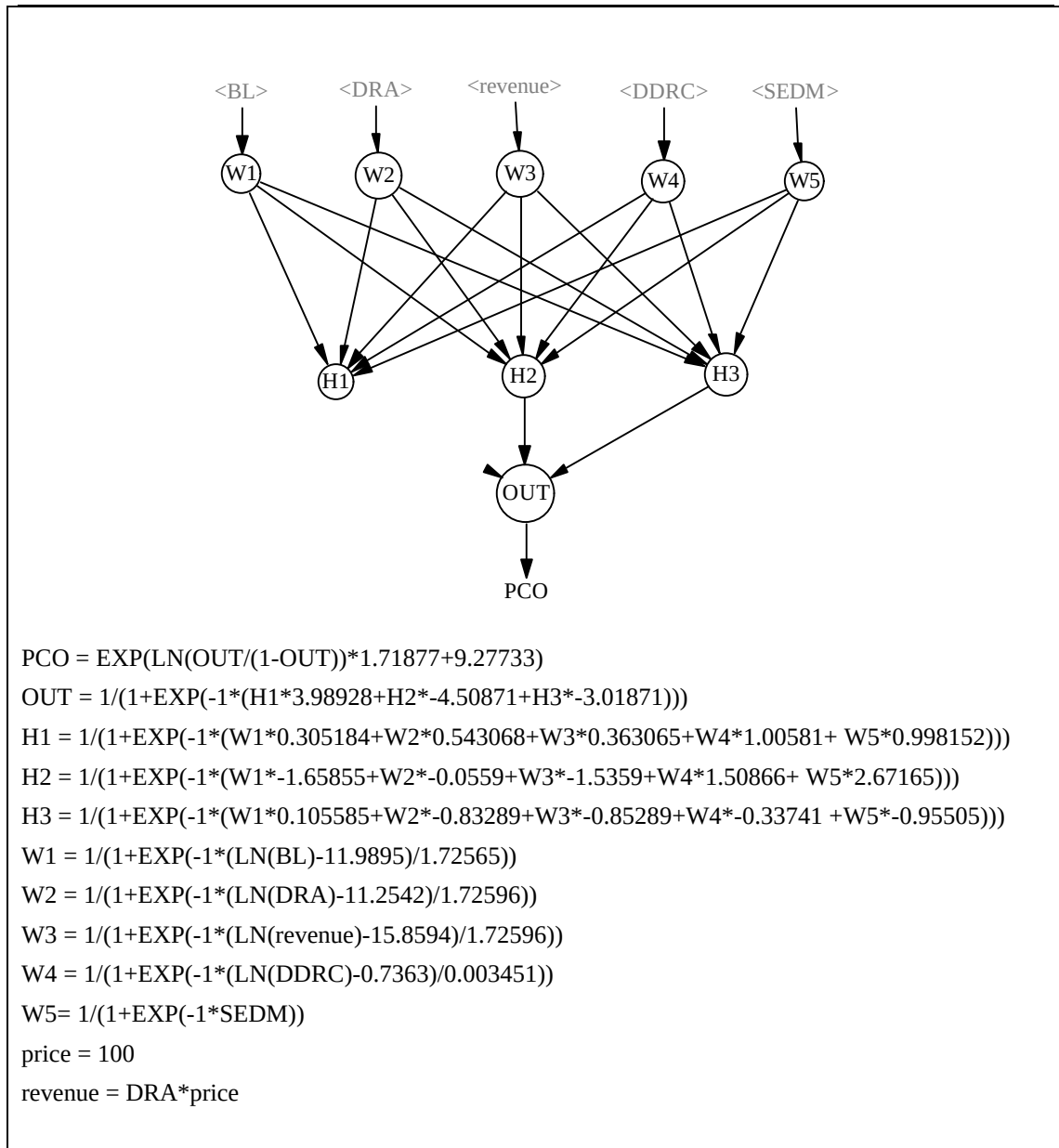
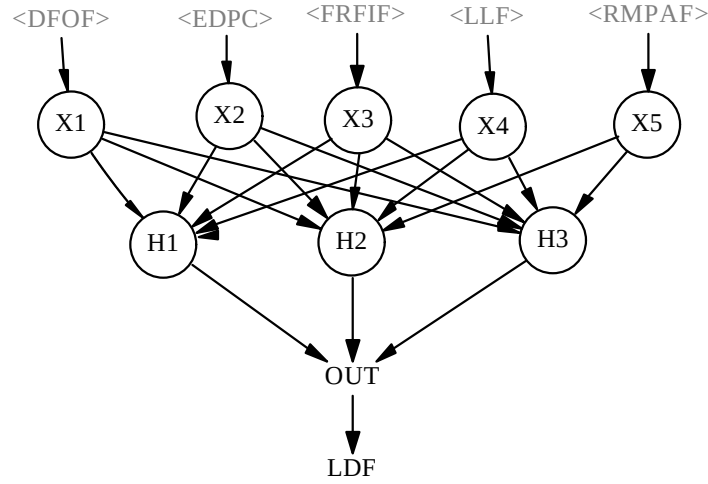


Fig.5: Diagram and equations of the growth model's high leverage policy



$$H1 = 1/(1+\text{EXP}(-1*(X1*w11+X2*w12+X3*w13+X4*w14+X5*w15)))$$

$$H2 = 1/(1+\text{EXP}(-1*(X1*w21+X2*w22+X3*w23+X4*w24+X5*w25)))$$

$$H3 = 1/(1+\text{EXP}(-1*(X1*w31+X2*w32+X3*w33+X4*w34+X5*w35)))$$

$$\text{OUT} = 1/(1+\text{EXP}(-1*(H1*wh1+H2*wh2+H3*wh3)))$$

$$X1 = 0.8*((\text{DFOF}-x1m)/x1s-y1min)/(y1mm)+0.1$$

$$X2 = 0.8*((\text{EDPC}-x2m)/x2s-y2min)/(y2mm)+0.1$$

$$X3 = 0.8*((\text{FRFIF}-x3m)/x3s-y3min)/(y3mm)+0.1$$

$$X4 = 0.8*((\text{LLF}-x4m)/x4s-y4min)/(y4mm)+0.1$$

$$X5 = 0.8*((\text{RMPAF}-x5m)/x5s-y5min)/(y5mm)+0.1$$

$$x1m=4.71389282, x2m=30093.22159, x3m=0.698456352, x4m=1.116144058, x5m=3005.067463;$$

$$x1s=0.070924391, x2s=1068.491835, x3s=0.007880488, x4s=1.428607458, x5s=113.762688;$$

$$y1min=-1.92222158, y2min=-1.524071469, y3min=-1.936169338, y4min=-0.781281136,$$

$$y5min=-1.807770392;$$

$$y1mm=3.858390322, y2mm=3.101063413, y3mm=3.858390113, y4mm=3.732034352,$$

$$y5mm=3.495289642;$$

$$w11= 0.171587, w21=-4.985481, w31= 1.218813$$

$$w12= 2.758465, w22=-0.780286, w32= 4.973371$$

$$w13=-0.724769, w23=-5.785361, w33= 1.145750$$

$$w14=-4.233764, w24=-3.310918, w34= 1.346264$$

$$w15=-1.145976, w25= 6.089832, w35=-8.820408$$

$$wh1= 4.745557$$

$$wh2=-6.000803$$

$$wh3=-3.972231$$

Fig.6: Diagram and equations of the stable model's high leverage policy

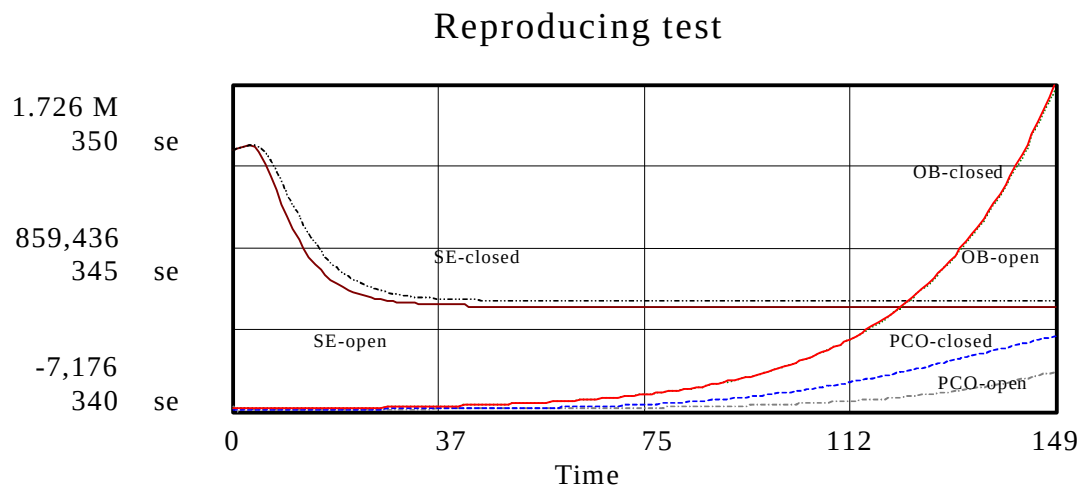


Fig.7: Result of reproducing test of the growth model

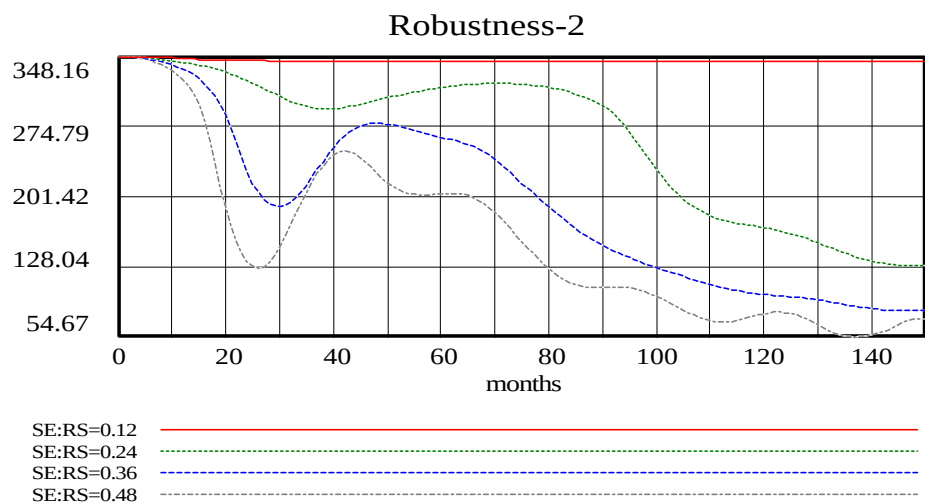
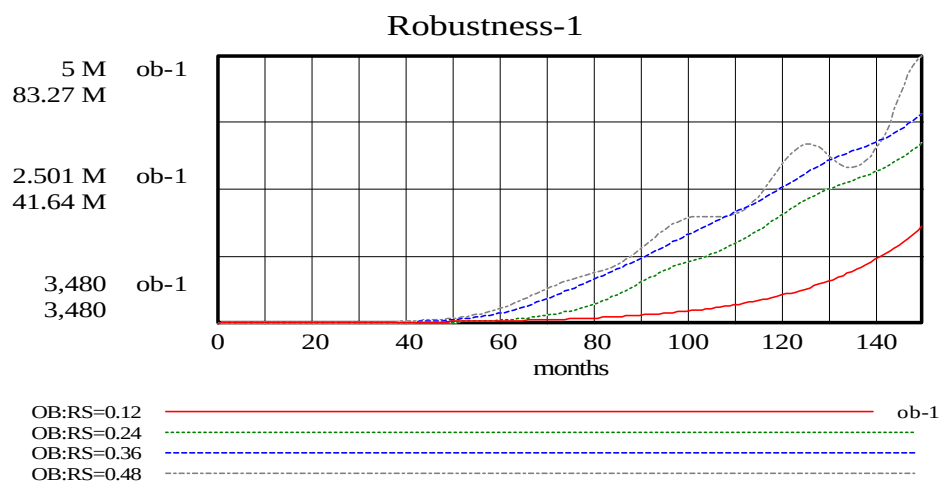


Fig.8: Result of robust test of the growth model

Table 6: Result of the reproducing test of the stable model

scenario: 2-year sine curve

	NPTDF	gainB	gainC	gainD	gainI	gainM
original model base run	3345853.5	10.579	7.7413	3.13	2.674	2.0087
original model new policy	3501079.8	1.7019	2.5091	0.7566	1.8962	0.5125
open-loop solution	3521606.3	0.7356	1.9707	0.2096	0.5277	0.4281
closed-loop policy	3552401.3	1.2381	3.3604	0.5904	1.419	0.5208

Table 7: the result of the robust test of the stable model

scenario: 4-year sine curve

	NPTDF	gainB	gainC	gainD	gainI	gainM
original model base run	3616565.3	4.4418	3.9684	1.2668	1.2496	1.0473
original model new policy	3622099.3	3.2196	2.726	1.3318	3.0075	1.2072
closed-loop policy	3618647.3	4.9186	10.134	2.4356	4.3962	0.5423

Discussions

In the simulation experiments, we found some interesting phenomena worth discussing.

1. Besides the policies of the system, the system performance is influenced by the objective function. Table 3 shows that if we take gain (IAFPC) as objective function, then the system performance is best. However, if we take NPTDF as objective function, then the system performance is worst.
2. In the second stage, we use the best set of variables by GNN to construct a high leverage policy. We found that the best sets of variables by GNN are different every time. However, these policies from different sets of variables made similar good performance. From above simulation experiments, we infer that there exist more than one high leverage policy in nonlinear dynamic system.
3. It is hard to explain the result of the method, because the method includes two nonlinear algorithms. The reason for the difference of the robustness between the two the model simulation is that, we guess, the artificial neural network has the capability of pattern recognition. It is very similar among the growth model's pattern of behaviors in different situations, but it is very different among the stable model's. So the growth model is robust, but the stable model isn't.
4. The policy constructed by the best set of variables from GNN may not pass the testing stage. In this time, we suggest to add one cycle loop process between the

second stage and the testing stage, that is to select next best set of variables from the output of GNN to construct another policy, and then test it.

Summary and further study

This study's method imitates human brain. According to the working hypothesis of human brain, we construct the conceptual framework of the method, and then base on it to develop the operational framework and process. There are two stages in the operational process: the first is to obtain open loop solution, the second is to obtain closed loop solution by our GNN software.

The result of the method's test shows that the reproducing is good, and the robustness of is half good and half bad.

In the users' view, this study's method is friendly and easy. Users without enough mathematical background can use the method by following the procedure. We will replace the routine of the procedure with a computer program in the future. So the method will be easier for users.

However, the method has two flaws. One is that it is hard to explain the result of the method. The reason results from two nonlinear algorithms constituting the method. Maybe the flaw can be resolved until the two algorithms have well-developed theorem. The other flaw is that GNN takes much time to run in PC. If we use higher speed computer, the flaw will be resolved.

As to the management implication of the method, in addition to derive high leverage policies, there are two themes: how to screening decision relevant information, and how to construct the objective hierarchy of organization. Now we are studying these themes.

Reference

1. Burns, J. R. and D. W. Malone, 1974, "Optimization Techniques Applied to the Forrester Model of the World," IEEE Transaction on Systems, Man, and Cybernetics, Vol. SMC-4, No. 2, pp. 164-171.
2. Cleeremans, Axel, 1993, Mechanisms of Implicit Learning: Connectionist Models of Sequence Processing, Cambridge, MA.: The MIT Press.
3. Coyle, R. G. 1985, "The Use of Optimisation Methods for Policy Design in a System Dynamics Model," System Dynamics Reviews, Vol. 1, pp. 81-92.
4. Forrester, J. W., 1961, Industrial Dynamics, Cambridge, MA, USA: Productivity Press.
5. Forrester, J. W., 1964, "Modeling the Dynamic Processes of Corporate Growth," In Collected Papers of Jay W. Forrester, Cambridge, MA, USA: Wright-Allen Press, pp. 172-174.

6. Forrester, J. W., 1968, "Market Growth as Influenced by Capital Investment", In Collected Papers of Jay W. Forrester, Cambridge, MA, USA: Wright-Allen Press, pp. 111-132.
7. Franco, D., 1990, "Policy Design in Oscillating Systems," Proceedings of the 1990 International Conference of the System Dynamics Society, pp. 381-394.
8. Goldberg, D. E., 1989, Genetic Algorithms in Search, Optimization, and Machine Learning, Reading, MA., USA: Addison-Wesley Publishing Company, Inc.
9. Graham, Alan K., 1977, Principles of Relationship Between Structure and Behavior, Ph. D. Dissertation, M.I.T.
10. Haykin, Simon, 1994, Neural Networks: A Comprehensive Foundation, New York, USA: Macmillan College Publishing Company, Inc.
11. Macedo, J., 1989, "A Reference Approach for Policy Optimization in System Dynamics Models," System Dynamics Reviews, Vol. 5, pp. 148-175.
12. Mohapatra, P. K. J. and S. K. Sharma, 1985, "Synthetic Design of Policy Decisions in System Dynamics Models: A Modal Control Theoretical Approach," System Dynamics Reviews, Vol. 1, pp. 63-80.
13. Morecroft, John D. W., 1988, "System Dynamics and Microworlds for Policymakers," European Journal of Operational Research, 35, pp. 301-320.
14. Ozveren, C. M. and John D. Sterman, 1989, "Control Theory Heuristics for Improving the Behavior of Economic Models," System Dynamics Reviews, Vol. 5, pp. 130-147.
15. Schaft A.J. van der, 1992, "L2-Gain Analysis of Nonlinear Systems and Nonlinear State Feedback H_∞ control," IEEE Trans. on Automatic Control, Vol.37, No.6, pp.770-784.
16. Sterman, J. D., 1989, "Modeling Managerial Behavior: Misperceptions of Feedback in a Dynamics Decision Making Experiment," Management Science, Vol. 35, pp. 321-339.
17. Talavage, J. J., 1980, "Modal Analysis to Aid System Dynamics Simulation," In A. A. Legasto et al., eds., System Dynamics,
18. Wang, Sy-Feng and Showing H. Young, 1992, "A Preliminary Experiment on Examining thinking in a Meta-Dynamic Decision Making Environment" , Proceedings of the 1992 International System Dynamics Conference, 757-766, Utrecht, The Netherlands.
19. Young, Showing H. and Chia Pin Chen, 1998, A Heuristic Mathematical Method for Improving the Behavior of Forrester's Market Growth Model, The 16th International Conference of System Dynamics Society, forthcoming, Quebec, Canada.
20. Young, Showing H. and Sy-Feng Wang, 1995, Designing the Learning

Environment of Learning Laboratories: Cognitive Strategy, Learning and Transfer, Proceedings of the 1995 International System Dynamics Conference, 976-985, Tokyo, Japan.

21. Young, Showing H. and Sy-Feng Wang, 1996, Measuring the Learning of Systems Thinking: Theory and Method, Proceedings of the 1996 International System Dynamics Conference, 605-608, Cambridge, MA., U.S.A.
22. Young, Showing H., Shih Hui Lo, and Sy-Feng Wang, 1993, "A Preliminary Design of CSS Production-Distribution Board-Type Simulation Game," The Proceedings of the 1993 System Dynamics Conference, 603-610, Cancun, Mexico.
23. Young, Showing H., Sy-Feng Wang, and Jen-shou Yang, 1991," Designing Successful Learning Program with the Tool of Dynamic Decision Simulation Game-An Experimental Study," Proceedings of the Fourth International Conference on Comparative Management, 276-281, Kaohsiung, Taiwan, R.O.C.
24. Young, Showing H., Sy-Feng Wang, and Jen-shou Yang, 1994, "Overcoming the Learning Barriers of Management Flight Simulators". Proceedings of the 1994 International Conference of the System Dynamics Society :Microworld, 101-108. Stirling, Scotland.