Journal of International Information Management

Volume 12 | Issue 2

Article 7

2003

An Artificial Neural Network Approach to Learning from Factory Performance in a Kanban-Based System

Barry A. Wray
University of North Carolina at Washington

Ina S. Markham
James Madison University

Richard G. Mathieu
St. Louis University

An Artificial Neural Network Approach to Learning from Factory Performance in a Kanban-Based System

Barry A. Wray
University of North Carolina at Wilmington

Ina S. Markham
James Madison University

Richard G. Mathieu
St. Louis University

ABSTRACT

Many Just-In-Time (JIT) manufacturing environments generate operational data reflecting both efficient and inefficient factory performance. Frequently data for inefficient performance is lost or discarded for fear of replicating poor performance. The purpose of this paper is two fold. First, historical JIT shop data is analyzed using a genetic algorithm (GA) to determine which shop factors are important determinants of factory performance. Second, subsequent to these important factors being identified by a GA, an artificial neural network (ANN) is used to learn the relationships between these factors and factory performance. The ANN can then be used to predict factory performance for future shop conditions and enhance shop performance. While ANN learning techniques have previously been applied to JIT production systems (Wray, Rakes, and Rees, 1997) (Markham, Mathieu, and Wray, 2000), these techniques have only been trained on data sets that reflect an efficient factory. Mathieu, Wray, and Markham (2002) investigated inefficient and efficient JIT factory performance but did not deploy either ANNs or a GA. In this paper an example application is presented using a GA to specify important shop factors and to predict saturated, starved or efficient factory performance based on dynamic shop floor data.

INTRODUCTION

There have been many benefits for firms that have employed the Just-In-Time with Kanban philosophy. Some of these benefits are the reduction of work-in-progress inventories, level production schedules, manual control systems, and high levels of quality. A kanban (Japanese for sign) is a manual/visual cue that is used to signal the replenishment of goods at each stage in the production process. The number of circulating kanbans is a critical issue for effective operation of a JIT production system. Too many kanbans result in excess work-in-progress inventory, while too few lead to production-floor stoppages. Excessive inventories, storage problems, machine idle times, long lead times, and output shortages can be caused by inappropriate levels of kanbans in a shop.

Published by CSUSB ScholarWorks, 2003

The purpose of this paper is two fold. First, both efficient and inefficient historical JIT shop data is analyzed using a genetic algorithm (GA) to determine which shop factors are important determinants of factory performance. Second, once the GA identifies these important factors an artificial neural network (ANN) learns the relationships between these factors and factory performance. The ANN is then used to determine how to adjust shop factors to ensure efficient factory performance for future shop conditions. A computer-based approach is presented that supports learning in a JIT manufacturing environment to improve factory performance. While it is possible for an ANN to learn from successes (positive data) and failures (negative data), conventional modeling dictates that negative information be eliminated from the training data set. However, in many environments it may be desirable to learn from an archived history of data that contains negative information (Triantaphyllou and Soyster, 1996) (Hall, Hansen and Lang, 1997).

Markham et al. (2000) compare artificial neural networks and CART on the kanban setting problem. The approach presented in this paper advances prior research by demonstrating a technique that can identify the importance of relationships between shop factors in determining good or bad factory performance. Dynamic factors (over two production periods) both endogenous and exogenous to the production function are examined. A methodology is presented that will allow a shop floor manager to identify relationships between shop factors that need to be monitored closely to operate a JIT factory at peak production performance.

The identification of important relationships between shop factors in determining good or bad factory performance in a dynamic setting is sometimes difficult when less than optimal shop conditions exist. As a result of this difficulty, an inefficient number of kanbans may exist during normal operations, causing sub-optimal performance. The development of a model for the JIT system must ensure that such sub-optimal performance is not repeated if possible. Traditional wisdom is to discard any data representing poor shop performance (negative information). However, there are a wide variety of reasons for wanting to learn from data representing less than optimal conditions. First, there is knowledge to be gained from each mistake so that the mistake is not repeated. The precise conditions that caused poor performance can be identified and steps can be taken to rectify the situation in the future (Minsky, 1994). Second, by combining negative data with positive data the number of observations in the training and validation data sets is increased. As a result, the sometimes data hungry artificial neural network can be successfully applied in cases where the quantity of only positive data was insufficient. Finally, by analyzing both good and poor performance it is possible for the model to uncover predictive structures that would otherwise be hidden. This learned information can reveal relationships causing poor (negative) performance so that measures to assure good (positive) performance can be taken.

LITERATURE REVIEW

The necessary internal and external conditions to the production function that must be met for a successful application of JIT have been identified as a constant (or near-constant) product demand, production process, and vendor supply (Fukukawa and Hong, 1993) (Huang, Rees and Taylor, 1983) (Price, Gravel and Nsakanda, 1994). Wray et al. (1997) found that even with constant distribution means for demand, machine processing, and vendor supply a dynamic variance could cause disruptions and inefficiencies in a JIT shop. They also found that certain combinations of 2-period dynamic factors are important for efficient factory performance. Markham, Mathieu, and Wray (1998) used a classification tree based approach to identify critical dynamic shop factors and then predicted the number of kanbans necessary for efficient factory performance. While the potential for genetic algorithm (GA) to find optimal or near optimal solutions to large, complex problems has been demonstrated by researchers, and real-world applications of GA are becoming increasingly common (Davis, 1991) (Goldberg, 1994), little research has been conducted in applying GA to the study of dynamic JIT factory performance. Artificial neural networks have been applied to JIT production systems (Wray et al., 1997) (Markham et al., 2000), but these techniques are typically used to learn by training on data sets that contain only efficient factory data. The classification tree based approach developed by Markham et al. (1998) was extended by Mathieu et al. (2002) to investigate inefficient and efficient factory conditions.

The performance of kanban-based production systems is a topic of continued interest in the academic community. Both Tardif and Maaseidvaag (2001) and Shahabudeen, Gopinath, and Krishnaiah (2002) present JIT production models which vary the number of kanbans in an attempt to improve factory performance. Haslett and Osborne (2000) modeled the local rules used by managers in the operation of a kanban system and report on the success as well as the unintended consequences of applying these decision rules. Takahashi and Nakamura (2002) developed and tested a decentralized reactive kanban system that improves factory performance in periods of unstable product demand.

RESEARCH DESIGN: AN OVERVIEW OF GENETIC ALGORITHMS AND ARTIFICIAL NEURAL NETWORKS

Genetic Algorithm

During the process of natural evolution individuals of a species are created by decoding information stored in sequential codes called chromosomes. These individuals are evaluated by the environment, with the fittest having a higher probability of surviving long enough to successfully reproduce. Reproduction, in essence, combines the existing chromosomes to create a new set of chromosomes. New chromosomes are also created as a result of mutation. Mutation occurs with a small probability. The success of the evolutionary process prompted John Holland (1986) to develop the genetic algorithm.

In the GA, every solution is represented as a chromosome, or string of values. Each value on the chromosome, or token, is restricted to a set of legal values. The set of problem solutions, or population, available at any point in time is the current generation. The fitness of each member of a generation is evaluated through a mathematical function. A new set of chromosomes is formed by operating on members of the current population. Although the chromosomes selected for use in forming the next generation are selected stochastically, those which are most fit have the highest probability of being selected.

Implementation of the genetic algorithm includes several specific decisions. The representation scheme for the chromosome should allow problem solutions to be represented as a fixed-length, string of tokens. The selection of an initial population of solutions and the determination of the size of the initial population are important. A method for evaluating the fitness of each member of the population must be determined. This evaluation must be efficient and must represent the goal of the optimization problem. In addition, the feasibility of each solution must also be evaluated. One or more operations must be defined for transforming existing chromosomes to a new generation of chromosomes. Finally, a criterion for stopping the process must be selected.

There are several issues to consider when using GAs: chromosome length, population size, fitness measure, crossover site, gap factor (if using elitist reproduction), probability of crossover, and probability of mutation. GAs are increasingly being considered as a possible method for solving combinatorial problems because their ability to hop randomly from solution to solution allows them to escape from local optima in which other algorithms might land.

Genetic Algorithm For Variable Selection

The selection of important shop variables is a critical step in understanding the true nature of the underlying relationships in a JIT shop. In the presence of noise, certain variables can mask the power of other variables. Variable selection, therefore, requires the ability to identify small synergistic subsets of variables that solve the problem at least as well as the full set of variables. NeuralWorks' NeuralSIM incorporates GA as the variable selection technique. The individuals are treated as sets of input variables. The algorithm starts with small sets of variables and uses successful groups of variables in the population to select larger sets of variables if necessary (smaller variable sets being preferred to larger sets). NueralSIM uses a multiple regression method incorporating a logistic function to select the model's input variables. This method avoids over-dependence on a test set by using cross-validation procedure. This study used three cross-validation sets. The objective function for determining fitness is the Root Mean Power Error function. The parameter is the value of the exponent in this objective function. A SoftMax output function is used to shape the variable selection model. A threshold, called the Ill Condition Threshold, is used to calculate fitness in a manner as to remove the problems associated with inverting ill-conditioned matrices.

The convergence conditions for the GA were set at a maximum of 50 generations, with halt occurring if the population's average fitness does not improve within 7 generations (referred to as a patience of 7). The improvement in the fitness, to be meaningful, is given by tolerance, set at 0.001 per generation. Parents are selected with a probability based on a normalized fitness value. NeuralSIM allows three ways to select a parent. It is recommended that the first parent be selected randomly from a ranked linear distribution while the second parent be selected randomly according to a uniform distribution. In the case of the parents ranked in order of fitness, two individuals are considered to have the same fitness if both values are within a quantization level, where a quantization level of 100 is equivalent to a tolerance of 1/100 = 0.01. For two individuals with equivalent fitness as measured by quantization, the smaller sized variable set is considered fitter whereas if the two individuals have equivalent fitness and are the same size, the most recently established individual is considered to be fitter.

A crossover probability of 0.7 and a mutation factor of 1 are used in this study. Crossover probability is the probability that an individual in a new generation will be created by combining variables from two parent individuals from the previous generation rather than just replicated from a single parent. Mutation is used as a mechanism to re-introduce lost variables back into the population. The elitist factor of 0.05 is used in the non-linear (neural net) variable selection methods. This specifies the proportion of the population that is replicated in the new generation. The multiple regression method selects only the fittest individual for reproduction.

Classification using an Artificial Neural Network

A key feature of this research is the use of NeuralWare's NeuralSIM artificial neural network package to model the relationships between JIT shop conditions and shop performance from past data that includes both good and poor factory performance. Important relationships between shop floor factors and factory performance will also be identified using a NeuralSIM built-in genetic algorithm. The ANN utilized in this research is a proprietary non-linear feed-forward general-purpose algorithm based on an Adaptive Gradient learning rule. Since the purpose of this research is not the development of ANN models please see Markham et al. (2000) for a complete description of ANN modeling.

METHODS AND PROCEDURES

A four-step methodology is presented that uses training data collected from a JIT factory exhibiting both good and poor performance. The four steps in the methodology are: (1) data generation, (2) identification of important shop factors, (3) identification of factory performance, and (4) identify the conditions for an efficient factory.

Data Generation

A simulation model of a JIT factory with endogenous and exogenous factors such as inventory levels, the number of kanbans, demand variability, vendor supply variability etc. is utilized in this research. SLAM II, developed by Pritsker (1986), is used as the simulation language to model the shop. The simulation model used in this research for data generation expands on the basic model developed by Huang, Rees, and Taylor (1983) and was used to generate data for Wray et al. (1997). Since the basic purpose of this research is not the development of a JIT shop model further discussion of modeling details is not included. For a complete description of the SLAM II model used in this research (including the code) see Wray (1992a).

The JIT shop simulated is dynamically changing over 2 periods of operation. The factors chosen for study were determined by reviewing the JIT literature and by analyzing a simulated, 'static' (i.e., one-period) shop (Rakes, Rees, Siochi, and Wray, 1994). In particular, the studies reported by Philipoom, Rees, Taylor, and Huang (1987), Gupta and Gupta (1989), Deleersnyder, Hodgson, Muller, and O'Grady (1989), and Ragatz and Mabert (1984) were influential in our selection. Not all factors deemed relevant by previous studies (e.g., shop configuration and product structure) were included because of the additional complexity they would have added to an already involved methodology. Such factors are left for examination in further research.

The 14 factors chosen for study are as follows:

Previous Period t-1 factors

1. demand variability during period t-1
2. vendor supply variability during period t-1
3. machine processing variability during period t-1
4. number of kanbans circulating in period t-1
5. finished goods inventory at the end of period t-1
6. work-in-progress inventory at the end of period t-1
7. average lead-time during period t-1 (the time between a production kanban arrives at a workcenter and the item is produced)
8. overtime for period t-1
9. average kanban circulation rate during period t-1
10. average kanban waiting time during period t-1,

Present Period t factors:

11. forecast of demand variability for period t
12. forecast of vendor supply variability for period t
13. forecast of machine processing variability for period t
14. number of kanbans to use in the present period at this workcenter

The previous research cited above suggested the inclusion of factors 1-8 and 11-14. Factor 9, the kanban circulation rate, and factor 10, kanban waiting time, were not found in previous studies but are included in this study because of their potential to add important explicative information on kanban levels and shop conditions. The GA will determine if each independent factor is important and thus included in the ANN model.

A simulation model of a JIT shop was used to collect the data for our study. The multi-period data generation design is given in Figure 1. The simulation model generates a data set of 560 possible combinations of inputs. For a more complete description of the simulation model of the JIT shop used in this research see Mathieu et al. (2002), Wray (1992a) and Wray (1992b).

The data represent 560 different combinations of shop factors and the resulting shop performance. For instance, a typical data tuple consists of values for each factor listed above (the independent variables) and the resulting condition of the factory (the dependent variable). The condition of the factory will be starved, efficient or saturated with inventory. A starved factory is distinguished by a shortage of inventory. These shortages cause excessive variations in processes resulting in costly starting and stopping of operations. Similarly, a saturated factory is distinguished by excessive inventories that can hide problems in the production process and result in excessive holding costs as well as problems with scheduling, queuing, storage and space. The factory condition (starved, efficient, or saturated) is determined by a total cost function with three components: overtime cost, work-in-process inventory cost, and finished goods inventory cost. Basically, if too few kanbans are used there would be excessive overtime costs. If too many kanbans are used there would be excessive holding costs for both work-in-progress inventory and finished goods inventory at the target workcenter.

It is not our intent to propose that companies wishing to duplicate our study within their shop construct a simulation model to generate shop data. The simulation model used in this research was necessary to generate dynamic shop data that was otherwise unavailable. Each individual application of our methodology should use actual historical information about shop factors, the number of kanbans used, and the resultant performance from the JIT shop under consideration. This shop data would require a screening process to determine if shop performance is starved, efficient, or saturated for the number of kanbans used for a particular set of shop factors. In practice, data for all possible shop scenarios would be unavailable for training the ANN. A major strength of an ANN is that it can learn relationships in existing data and successfully interpolate or extrapolate to estimate unknown points in the decision space. The ANN model would learn relationships based on the limited historical data from a JIT shop and make future recommendations on the number of kanbans needed in shop scenarios that the ANN model has never seen before.

Identification of Important Shop Factors

The JIT shop data were analyzed using a genetic algorithm within NeuralSIM. Of the 14 independent shop factors included, 9 were found to be important in predicting whether the JIT factory will be starving for inventory, running efficiently, or saturated with too much inventory. Table 1 lists the important shop factors that were chosen as the "fittest" by the GA in the shop factor selection process. It is important to note that dynamic factors from both period t-1 and period t were selected by the GA. Period t-1 factors are from the previous period's production and provide valuable contributions as determinants of the factory performance in period t.

Vendor supply variability from period t-1 was found to be important. This is consistent with previous research (Wray et al., 1997) (Markham et al., 1998). A grouping of the number of kanbans, ending finished goods inventory, and ending work-in-progress inventory from period t-1 were all determined to be important. This is a somewhat interesting combination because these factors are highly correlated (see Table 2). Also, a second grouping of period t-1 leadtime, kanban circulation rate, and kanban waiting time are included as important factors. The factors in the second grouping are also highly correlated (table 2). The inclusion of such highly correlated combinations of shop factors is evidence that the GA was able to identify the incremental contribution of each individual shop factor.

The forecast of demand variability for period t was found to be an important shop factor by the GA. This is evidence that the forecast of demand variability for period t is a good indicator of factory performance. Such evidence is not surprising since previous studies by Huang et al. (1983), Philipoom et al. (1987), Wray et al. (1997), Markham et al. (1998), and others have also reported the same finding. However, the GA did not include period t-1 demand variability in the set of important shop factors. This seemingly contradictory information may be the result of other factors "covering up" for the affect of demand variability in period t-1. An examination of Table 2 shows a high correlation of period t-1 demand variability with period t-1 leadtime, kanban circulation rate, and kanban waiting time. Thus eliminating period t-1 demand variability does not automatically exclude its influence.

Perhaps the most significant shop factor included by the GA is the number of kanbans in period t. This shop factor must be significant in order for our methodology be valid. This shop factor is the one factor that we can control for period t production. Thus, in the fourth and final phase of our methodology trial numbers of kanbans for period t production can be used to find a level that will allow the JIT factory to run efficiently.

Identification of Factory Performance

The classification results for the ANN are given in Tables 3 and 4. The results are in the form of probability tables for both the learning and test samples used. Each table displays the classification probabilities for the ANN for classifying a starved, efficient or saturated factory. Based on the probabilities tables, the ANN was able to correctly classify an efficient factory scenario for 76% of the learning samples and 69% of the test samples. It is worth noting that the ANN misclassified an efficient factory scenario as starved (18%) three times more often than an efficient factory was misclassified as saturated (6%) in the learning sample. In the test sample an efficient factory was misclassified as starved (26%) about 5 times as often as an efficient factory was misclassified as saturated (5%). This is a poor result for JIT factory performance because incorrectly identifying an efficient factory as starved would result in more inventory being added to an already efficiently running factory, thus resulting in excess inventory (saturated factory). This is counter to the JIT philosophy of reduced inventories. However, this may be due to backorders costs having a greater influence than excessive inventory costs in the cost function used.

When the ANN was presented with a starved factory it correctly classified 82% for the learning sample and 82% correct for the test sample. It is important to point out that the ANN incorrectly classified only 18% of the starved factories as efficient and 1% of the starved factories as saturated in the learning sample and only 17% of the starved factories as efficient and 1% as saturated in the test sample. Any misclassification is not desirable, however classifying a starved factory as saturated would result in less inventory in an already starved JIT factory. This is consistent with the JIT philosophy of reducing inventory to uncover hidden problems in a shop.

The ANN's classification performance was best for the saturated factory condition. The ANN correctly classified a saturated scenario for 93% of the learning sample and 88% of the test samples. Only 6% and 11% of the saturated factories were misclassified as efficient in the learning sample and test sample respectfully. An important result is that only 1% of the saturated factories were misclassified as starved in both learning and test samples. This is an intuitive result for a JIT factory. A misclassified saturated factory would result in more inventory being introduced in an already saturated factory. The JIT philosophy emphasizes the reduction of inventory and the ANN classification results are consistent with this philosophy.

Enhancing JIT Factory Performance

The final step of our methodology is to use the concepts learned from efficient and non-efficient historical data to improve shop performance. There are two ways a shop manager can control period t production. First, the one factor in the JIT shop that management has control over for period t production is the number of kanbans used at a workcenter. This factor was found by the GA to be an important factor and included in the ANN model. The trained ANN model could be used to alternatively evaluate different trial levels of kanbans until the ANN predicts an efficient shop. This approach can be used to assist in the decision making process of a manager when it is necessary to determine the number of kanbans in period t to ensure efficient factory performance.

Second, the forecast for period t demand variability was found to be important, therefore the shop manager should closely monitor the actual level of demand variability (an exogenous variable) to ensure it is the same as the forecasted variability for period t. If this forecast proves invalid the forecast must be updated and the ANN model should be updated to reflect the new level of demand variability.

CONCLUSIONS

The results presented in this paper demonstrate the successful application of a genetic algorithm (GA) to identify the important JIT shop factors together with the application of an artificial neural network (ANN) to learn the relationships between these shop factors and factory performance. These results extend the findings of Wray et al. (1997), Markham et al. (2000) and Mathieu et al. (2002) by applying GA and ANN techniques to a non-optimal JIT production environment in which starved, saturated and/or efficient factory conditions are represented. Ultimately, the ANN was successfully used to predict factory performance for future shop conditions and to enhance JIT shop performance.

This paper suggests further research in applying artificial intelligence to JIT production planning. Recent developments in information technology and manufacturing planning and control (MPC) systems have resulted in manufacturing information systems that supply real-time shop floor data to data warehousing applications that support management decision making. Data mining tools have been developed to enable the exploration and analysis of large quantities of data to discover meaningful patterns and rules, which in turn allows an organization to improve its manufacturing operations through better understanding of its processes. A promising area of future research is in the design of information architectures that support rule-based decision-making in a manufacturing environment.

REFERENCES

Davis, L., (1991). *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold.

Deleersnyder, J., Hodgson, T., Muller, H., & O'Grady, P. (1989). Kanban controlled pull systems: an analytic approach. *Management Science*, 35(9), 1079-1091.

Fukukawa, T., & Hong, S.C. (1993). The determination of the optimal number of kanbans in a just-in-time production system. *Computers and Industrial Engineering*, 24(4), 551-559.

Goldberg, (1994). Genetic and Evolutionary Algorithms Come of Age, *Communications of the ACM*. 37 (3), 113-118.

Gupta, Y. & Gupta, M. (1989). A system dynamics model for a multi-stage multi-line dual-card JIT Kanban system, *International Journal of Production Research*. 27(2), 309-352.

Hall, D., Hansen, R., & Lang, D., (1997). The negative information problem in mechanical diagnostics, *Journal of Engineering for Gas Turbines and Power*. 119, 370-377.

Haslett, T. and Osborne, C. (2000). Local rules: their application in a kanban system. *International Journal of Operations & Production Management*, 20(9), 1078-1092.

Holland, J.H., Holyoak, K.J., Nisbett, R.E., & Thagard, P.R. (1986). *Induction: process of inference, learning, and discovery*. The MIT Press.

Huang, P., Rees, L. & Taylor, B. (1983). A simulation analysis of the Japanese just-in-time technique (with kanbans) for a multi-line, multi-stage production system. *Decision Sciences*. 14(3), 326-344.

Mathieu, R., Wray, B. & Markham, I. (2002). An approach to learning from both good and poor factory performance in a Kanban-based just-in-time production system. *Production Planning and Control*. 13(8), 715-724.

Markham, I., Mathieu, R. & Wray, B. (1998). A rule induction approach for determining the number of kanbans in a just-in-time production system. *Computers and Industrial Engineering*. 34, 717-726.

Markham, I., Mathieu, R. & Wray, B. (2000). Kanban setting through artificial intelligence: a comparative study of artificial neural networks and decision trees. *Integrated Manufacturing Systems: the International Journal of Manufacturing Technology Management*, 11(4), 239-246.

Minsky, M. (1994). Negative expertise. *International Journal of Expert Systems*. 7(1), 13-19.

Philipoom, P., Rees, B., Taylor, B., & Huang, P. (1987). An investigation of the factors influencing the number of kanbans required in the implementation of the JIT technique with Kanbans. *International Journal of Production Research*. 25(3), 457-472.

Price, W., Gravel, M., & Nsakanda, A.L. (1994). A review of optimization models of kanban-based production systems. *European Journal of Operational Research*. 75, 1-12.

Pritsker, A., Sigal, C., & Hammesfahr, R., 1989, *SLAM II: network models for decision support* Prentice-Hall.

Ragatz, G. & Mabert, V. (1984). A simulation analysis of due date assignment rules. *Journal of Operations Management*. 5(1), 27-39.

Rakes, T., Rees, L., Siochi, F., & Wray, B. (1994). Estimating the number of kanbans using neural networks. *Advances in Artificial Intelligence in Economics, Finance, and Management*. 1, 125-139.

Shahabudeen, P., Gopinath, R. and Krishnaiah, K. (2002). Design of bi-criteria kanban system using simulated annealing technique. *Computers & Industrial Engineering*, 41(4), 355-372.

Takahashi, K. and Nakamura, N. (2002). Decentralized reactive kanban system, *European Journal of Operational Research*, 139(2), 262-274.

Tardif, V., and Maaseidvaag, L. (2001). An adaptive approach to controlling kanban systems. *European Journal of Operational Research*, 132(2), 411-423.

Triantaphyllou, E. & Soyster, A. (1996). On the minimum number of logical clauses inferred from examples. *Computers & Operations Research*. 23(8), 783-800.

Wray, B., Rakes, T., and Rees, L. (1997). Identifying critical factors in a dynamic JIT environment using neural networks. *Journal of Intelligent Manufacturing*. 8, 83-96.

Wray, B. A. (1992a). Prediction and control in a Just-In-Time environment using neural networks, Dissertation, Department of Management Science, Virginia Tech, Blacksburg, Va.

Wray, B. A. (1992b). Dynamically adjusting the number of Kanbans in a stochastic JIT environment: an application of neural networks. Proceedings of the Twenty-Fourth Annual Meeting of the Decision Sciences Institute. San Francisco, 735-737.

Figure 1. Multi-Period Data Generation Design where:

DVT1 – Demand variability for period t-1 (possible values are Low, High)
PVT1 – Machine processing variability for period t-1 (possible values are Low, High)
VVT1 – Vendor supply variability for period t-1 (possible values are Low, High)
DVT – Forecast of Demand variability for the present period t (possible values are Low, High)
PVT – Forecast of Machine processing variability for the present period t (possible values are Low, High)
VVT – Forecast of Vendor supply variability for the present period t (possible values are Low, High)

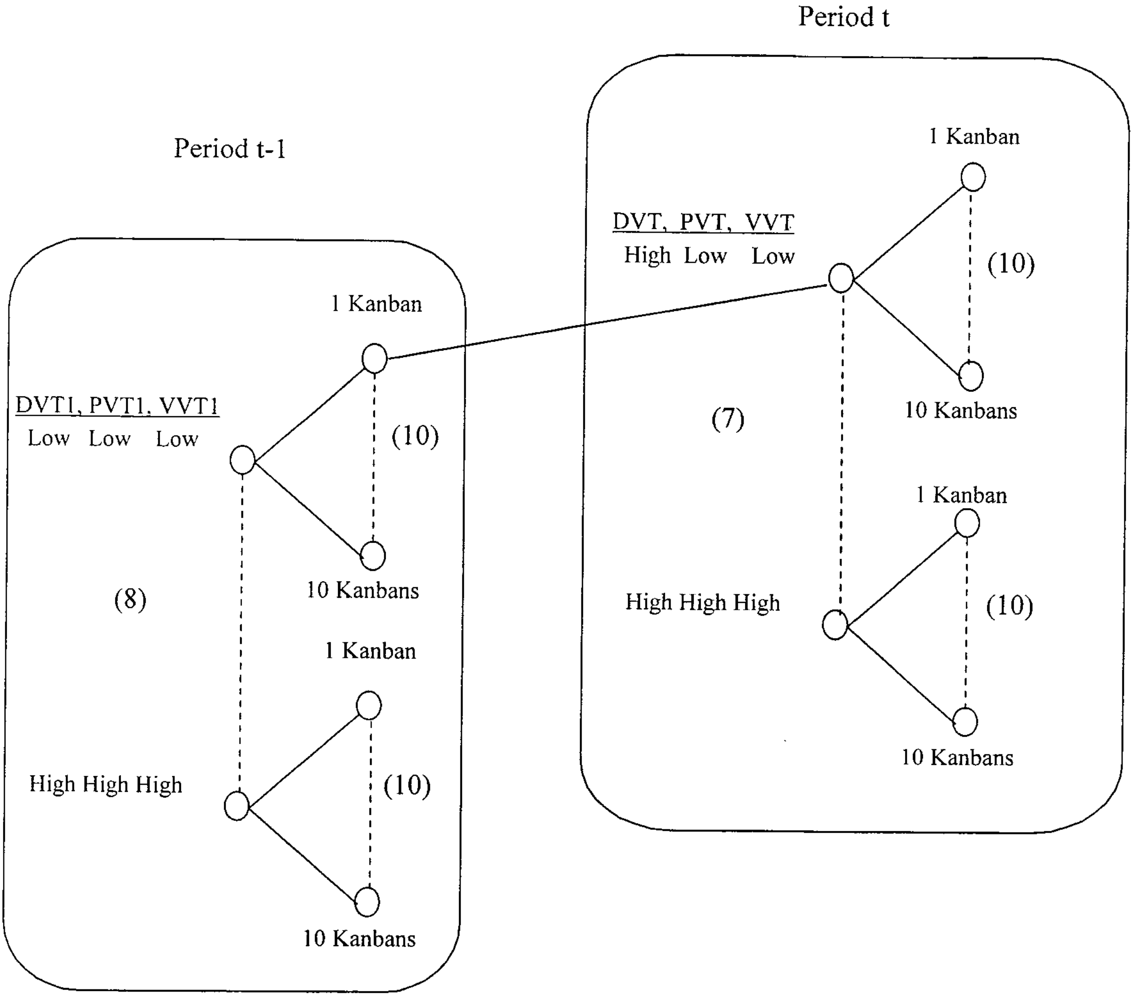

Table 1: Results of the GA Variable Selection procedure

Important Factors
Vendor supply variability for period t-1
Number of kanbans from period t-1
Finished goods inventory at the end of period t-1
work-in-progress inventory at the end of period t-1
average lead-time during period t-1
average kanban circulation rate during period t-1
average kanban waiting time during period t-1
Number of kanbans from period t
Demand variability for period t

Table 2. Correlation matrix of period t-1 resultant variables

	Demand Variability t-1	**# Kanbans period t-1**	**Ending Inv from t-1**	**Ending WIP from t-1**	**Avg Leadtime from t-1**	**Circulation rate t-1**	**Waiting time t-1**
Demand Variability t-1	1						
# Kanbans period t-1	0.034492	1					
Ending Inv from t-1	-0.16574	0.952057	1				
Ending WIP from t-1	-0.35363	0.855968	0.915944	1			
Avg Leadtime from t-1	0.897998	-0.27146	-0.41581	-0.53422	1		
Circulation rate t-1	-0.81093	0.319555	0.418558	0.566411	-0.91208	1	
Waiting time t-1	0.770365	-0.51134	-0.60791	-0.6995	0.960394	-0.90357	1

Table 3. Learning Sample Classification Probability Table

ACTUAL CLASS	PREDICTED CLASS		
	Starved	Efficient	Saturated
Starved	0.82	0.18	0.01
Efficient	0.18	0.76	0.06
Saturated	0.01	0.06	0.93

Table 4. Test Sample Classification Probability Table

ACTUAL CLASS	PREDICTED CLASS		
	Starved	Efficient	Saturated
Starved	0.82	0.17	0.01
Efficient	0.26	0.69	0.05
Saturated	0.01	0.11	0.88