Journal of Industrial and Systems Engineering
Vol. 4, No. 3, pp 183-192
Fall 2010

Learning Curve Consideration in Makespan Computation Using Artificial Neural Network Approach

Suresh Kumar[1*], A. Arunagiri[2]

Department of Electrical and Electronics Engineering Technology, Yanbu Industrial College, Yanbu Al Sinaiyah, Kingdom of Saudi Arabia
[1]vpsureshkumar@yahoo.com, [2]a.arunagiri@yahoo.co.in

ABSTRACT

This paper presents an alternative method using artificial neural network (ANN) to develop a scheduling scheme which is used to determine the makespan or cycle time of a group of jobs going through a series of stages or workstations. The common conventional method uses mathematical programming techniques and presented in Gantt charts forms. The contribution of this paper is in three fold. Firstly, the learning curve which is characterized by a coefficient is considered in the computation work. Secondly, this work is limited to small number of jobs and is useful for project based pilot runs which involve learning. Lastly, the scheduling scheme is developed in ANN as an alternate method. Extensive and successful training using the input and output vector pairs were done to establish the proposed method. Comparison was done for the tested outputs and results produced seem reliable.

***Keywords*:** Learning curve, Scheduling, Back Propagation Network, Gantt chart

1. INTRODUCTION

Designing work systems require many considerations such as volume of jobs to done, number of people to be stationed, equipment procurement, sustaining raw material inventories and storage, learning of the work to be done by the operators and controlling the required time to complete the jobs.

Learning through human training is a very important consideration when humans are involved in the work design. Thus, it is important to be able to predict how learning affects the job times and costs. In human performances, when activities are done repetitively, the time required to perform a job decreases. The degree of improvement and the number of jobs needed to significantly materialize improvement is a function of jobs being done. This is characterized as learning curve defined by Stevenson (2006). However, learning factors have very little relevance for scheduling routine activities or mass production lots but they are significant for new or complex repetitive activities.

[*] Corresponding Author

ISSN: 1735-8272,

Typically, new products or pilot runs require small number jobs to be done. In this scenario, operators that are assigned to respective stages or workstations have to be trained or guided. Once the training is done, then they have to produce the first job and the time is measured. Obviously, the time taken is bound to be longer as they have very little experience in performing the job. As they move on to perform subsequent similar jobs, they will gain experience and produce the jobs at a faster rate. Realistically, the processing times for jobs do not stay constant throughout. The general relationship is alternatively referred as experience curve and is quantified by factor called the learning curve coefficient.

This research focuses on developing an alternative method using ANN to determine the makespan of scheduling scheme for a tandem system. In this work, the learning curve coefficient is considered in the computation work. It is an important consideration in predicting the cycletime or makespan. Since the learning curves are considered, the proposed method is limited to small number jobs.

This paper is organized as follows: Section 2 includes some previous work done. Section 3 considers the analytical algorithm to develop the schedule scheme. Section 4 describes the Back Propagation Network (BPN) technique. Section 5 deals with a computational example and section 6 concludes this paper.

2. LITERATURE REVIEW

Artificial neural networks (ANNs) apply a different way from traditional computing methods to solve problems. Basically, conventional computers use algorithmic approach which means that specific steps have to be defined for the computer in order to solve a specific problem. That means, traditional computing methods can only solve the problems that we have already understood and knew how to solve. Due to ANNs ability to adapt, learn, generalize, cluster or organize data, it has in some way become a powerful tool to solve problems that we do not exactly know how to solve. There are so many structures of ANNs including, percepton, adaline, madaline, kohonen, back propagation and many others. Back propagation network (BPN) is the most commonly used, as it is relatively very simple to implement and effective. In this work, BPN approach is adopted to develop the proposed solution method.

A mathematical model was developed by Bohlen and Barany (1976) to predict a specific learning curve for a specific operator performing a specific operation. Learning curves of operators performing industrial operations vary according to the characteristics of the operator and the characteristics of the operation being performed. Large increases in productivity are typically realized as organizations gain experience in production. As explained by Argote and Epple (1990), learning curves have been found in many organizations and the rates vary considerably. Some organizations show remarkable productivity gains and others show little or no learning. This is due to organizational "forgetting," employee turnover, transfer of knowledge from other products and other organizations, and economies of scale.

In Guo et al. (2009) study, a production control problem on a flexible assembly line (FAL) with flexible operation assignment and variable operative efficiencies is investigated. They have formulated a mathematical model of the production control problem with the consideration of the time-constant learning curve to deal with the change of operative efficiency in real-life production. In Stafford and Tseng’s (2002) paper, they analyzed two models for a family of four different M-machine, N-job flowshop sequencing problems. They used mixed-integer linear programming to produce optimal job sequence by achieving a positional assignment approach. Two evolutionary algorithms which are the genetic algorithm and memetic algorithm were propose for a scheduling problem for manufacturing cells by Moghaddam et al. (2006). The intention was to find an optimal permutation schedules.

Kumar and Omar (2005a, 2005b, 2007) have developed an analytical method for a 5 stage probabilistic re-entrant line using the MVA technique for an environmental stress testing (EST) operation. The model was based on MVA and traffic equation. Their objective was to compute the mean waiting time and mean throughput rate. In their work, they found that as the number of lots in the system increased beyond a certain number, analytical and simulation results comparison produces large errors. A correction factor was introduced by Kumar and Omar (2005a) into the model to adjust the mean waiting time beyond a certain number of lots in the system. However the correction factor formulation obtained was case dependant. Further improvement was done on the formulation by Kumar (2008) and limited hour of operation usage is considered. Thus, a modified analytical method based on MVA and lot clustering technique is proposed to be used for the probabilistic re-entrant line. It is a method developed to allow the operation manager to measure the cycle time in terms of lot clusters.

3. ANALYTICAL SCHEDULING SCHEME

For a tandem network of n stages, the time taken to process a job is given by t_{kn} where T_{11} to T_{1n} are the times taken to process the first lot. Thus the cycle time for the first lot is given by

$$M(1) = \sum_{n=1}^{n} T_n \tag{1}$$

However in order to complete a group of k jobs the makespan M equation is given by

$$M(k) = \sum_{k=1}^{k} \sum_{n=1}^{n} t_{kn} \tag{2}$$

where k is the number of jobs and n is the stage index. In order to accommodate the learning curve coefficient, the jobs times will follow an exponential relationship which is given by

$$t_{kn} = T_{1n} \times k^{(\ln\phi_n/\ln 2)} \tag{3}$$

where T_{1n} is the time for first lot in the n stage. The obtained values are framed in a Gantt chart and the makespan can be determined. The values of the makespan with respect to the number of jobs are generated using this method and are used to train the BPN. This becomes a baseline for the training and once the BPN is trained, other output values can be obtained for a given set of inputs as demonstrated by Shihab (2006). The input parameters are k, ϕ_n and T_{1n}. ϕ_n should be determined from a real system by measuring ratio of time taken to produce k and ($k \times 2$) jobs.

$$\phi_n = \frac{t_{(2k)n}}{t_{kn}} \tag{4}$$

3.1 System Assumptions

In order to facilitate the learning curve into the computation work for both analytical and BPN approach, several assumptions and limitations are considered as follows:

a) Projections based on learning curves are regarded as average or approximation of actual times.

b) The rate of improvement or learning is assumed to be constant

Since the learning rates differ from stage to stage due different job requirement and operator's capability, the empirical studies of their learning rate is desired. However due to the limitation of these work, assumed coefficients are used to demonstrate this application.

4. BACK PROPAGATION NETWORK (BPN)

Back propagation was created by generalizing the Widrow-Hoff learning rule by Mcollum (1997) into multiple-layer networks and nonlinear differentiable transfer functions. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined by the user.

Standard back propagation is a gradient descent algorithm defined by Widrow-Hoff learning rule, in which the network weights are moved along the negative of the gradient of the performance function. The term back propagation refers to the manner in which the gradient is computed for nonlinear multilayer networks. There are a number of variations on the basic algorithm that are based on other standard optimization techniques, such as conjugate gradient and Newton methods.

Properly trained back propagation networks tend to give reasonable answers when presented with inputs that they have never seen. Typically, a new input leads to an output similar to the correct output for input vectors used in training that are similar to the new input being presented. This generalization property makes it possible to train a network on a representative set of input/target pairs and get good results without training the network on all possible input/output pairs.

4.1 BPN architecture

The most common BPN architecture is presented in Figure 1. It is shown to have three layers, namely, input, hidden and output layers. During the training, several sets of the input and their corresponding output vectors are considered. The training phase is used to determine the weights between the input, hidden and output layers.

The neurons used in the study use the sigmoid activation function defined by the following equation:

$$\begin{bmatrix} \text{Neuron} \\ \text{output} \end{bmatrix} = \frac{1.0}{1.0 + e^{-\alpha v}} \tag{5}$$

where α is the abruptness of the sigmoid function and the y is the total input to the neuron. Let the vector **X** represent an input to the input layer as shown in the Figure 1. The net input at the hidden layers is computed by the matrix equation. Let the vector **X** represent an input to the input layer as shown in the Figure 1.

The net input at the hidden layers is computed by the matrix equation as below:

$$V_H = [WH]\ \mathbf{X} \tag{6}$$

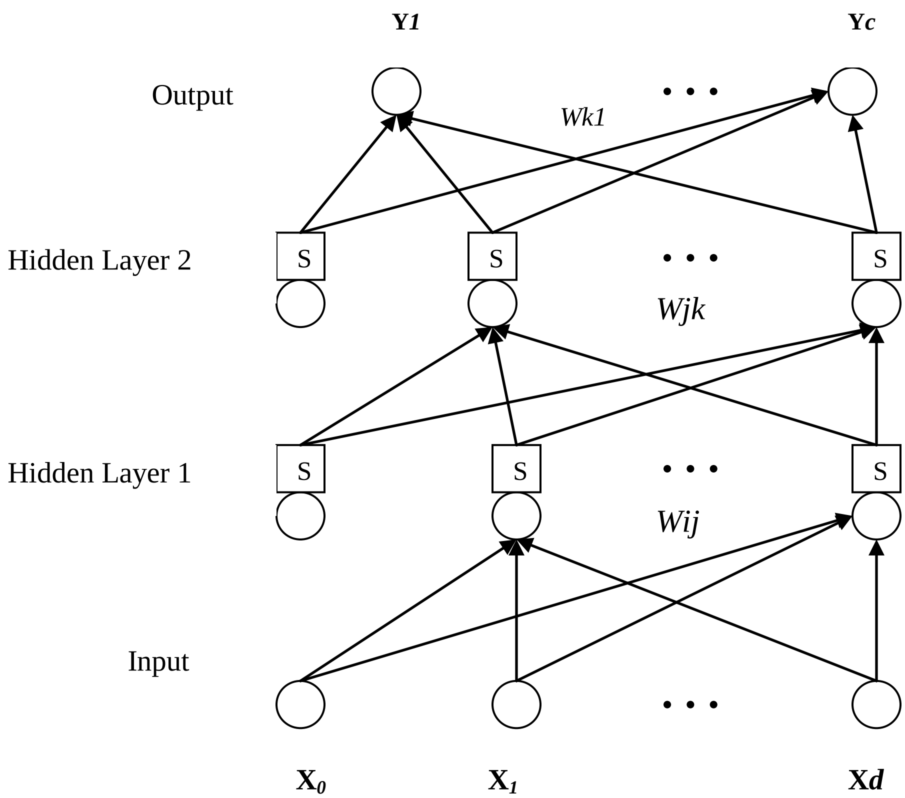

Figure 1 Basic BPN Architecture

where WH_{ij} denotes the weight between i^{th} input layer node and j^{th} hidden layer node and WH_{jk} denotes the weight between j^{th} hidden layer layer node and k^{th} hidden layer node The output of the hidden layer nodes are given by

$$\mathbf{Y}_H = \Phi\ (V_H) \tag{7}$$

where Φ is the appropriate activation function. In a similar manner, the total input at the output layer is given by the following equation:

$$V_O = [WO]\ \mathbf{Y}_H \tag{8}$$

The output of the output layer node is given by

$$\mathbf{Y} = \Phi\ (V_O) \tag{9}$$

The steps of the well-established training algorithm based upon Newton's steepest descent technique are given below:

1. Read in the training set and randomly initialize the weights. Set iteration index $n = 1$.
2. Set training set index $p = 1$.
3. Propagate $\mathbf{X}^{\mathbf{p}}$ through the network.

4. Determine the error vector of the p^{th} training set
   $\mathbf{E}^p = \mathbf{O}^p - \mathbf{Y}^p$ where $\mathbf{O}^P$ is the vector of expected output.
5. Correct the weights using Newton's steepest descent technique.
6. If $p <$ number training sets P, set $p = p+1$ and go to step 3.
7. If $\sum_{p=1}^{P} \left| E^p \right|^2 >$ tolerance ε, increment the iteration index n and go to step 2.

The above method works well and has been well documented. The method requires that the input and output to be from a continuous domain. Further, it also requires that the input and output set of vectors are non-contradictory for a successful training and operational function.

The next section explains the use of BPN to determine the makespan for a given job (k), the individual stage learning curve coefficient (ϕ_n), and the time taken by the individual stage to complete the first job (T_{1n}). n is stage index (1,2,3,..).

4.2. Implementation of BPN to determine the makespan, *M*(*k*)

The input vector for the BPN comprises the total number of jobs, the learning coefficient for the individual stage, and the time taken by the individual stage to complete the first job. Thus the input vector **X** is

$$\mathbf{X} = [k,\ \phi_1,\ \phi_2,\ \phi_3, \ldots\ldots, \phi_n, T_{11},\ T_{12},\ T_{13,\ldots\ldots\ldots}\ T_{kn}\ ] \tag{10}$$

where n represents the stage index (1, 2, 3,..)

- k represents the total number of jobs
- ϕ_n represents the learning curve coefficient of the individual stage and
- $T_{1n,}$ represents the time taken by the individual stage for the first job in hours.

The output vector for the BPN consists of makes span. $M(k)$ for the corresponding input Vector **X**. Thus the output Vector **Y** is

$$\mathbf{Y} = [\ M(1)_,\ M(2)_,\ M(3)_{\ldots\ldots\ldots}\ M(k)] \tag{11}$$

For example, $M(2)$ is defined as the cycle time to complete two jobs in the system.

Several sets of input vectors were created by the following scheme:

(a) Varying the total number of jobs (k)

(b) Varying the learning coefficient of the individual stage.

(c) Varying the time taken by the individual stage for the first job.

The N^{th} such input generated is referred to by the vector $\mathbf{X_N}$. Similarly the corresponding output vector for this N^{th} input is referred to by $M(k)$.

Summarizing, several of these input and output vector pairs are generated by the analytical method explained in section 3 and are stored for the BPN training. After the successful training of the BPN model, it should able to produce the makespan *M* for any number of the jobs (*k*), the individual learning curve coefficient (ϕ_n), and the time taken by the individual stage to complete the first job (T_{1n}) with minimum time and maximum accuracy.

5. COMPUTATIONAL EXAMPLE AND RESULTS

A seven stage tandem system as shown in Figure 2 is designed and tested using the proposed method. This system is useful in developing a small number of similar new products. Each stage has it own job and processing time. Since each stage has different job function, the learning rate also differs with different initial time. In addition the number of jobs is limited to 20 as this represents significantly small number of jobs.

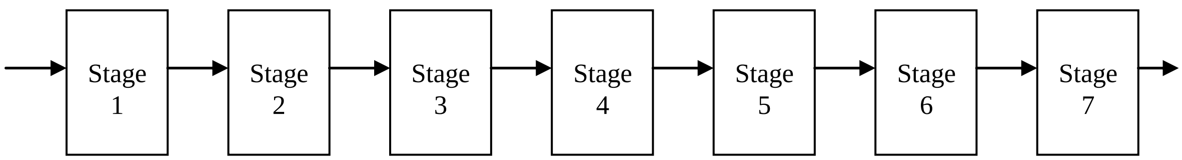

Figure 2 A seven stage tandem network

The BPN was trained in MATLAB® environment unit for different combinations of *k*, ϕ_n and T_{1n}. The training iteration is illustrated in Figure 3 using sets of 75 combinational data. For this case, the training seemed to require 1169 iterations in order to achieve the goal.

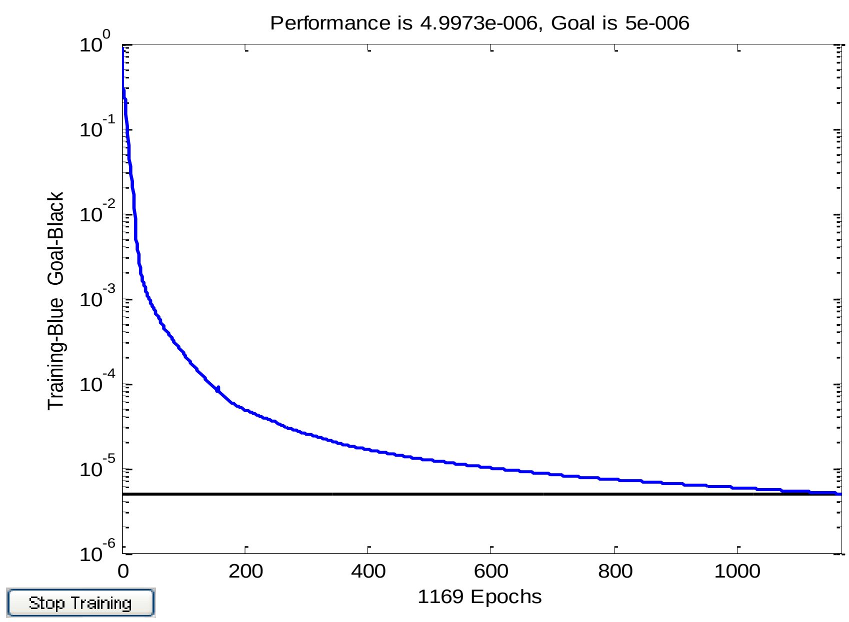

Figure 3 BPN training

Table 1 Results from the analytical method and from the trained BPN

Job (k)	Learning curve coefficient (ϕ_n)							Time for the first job at each stage (hours)							M(k) (hours)	BPN output (hours)	Accuracy %
	ϕ_1	ϕ_2	ϕ_3	ϕ_4	ϕ_5	ϕ_6	ϕ_7	T_{11}	T_{12}	T_{13}	T_{14}	T_{15}	T_{16}	T_{17}			
3	0.8	0.85	0.8	0.9	0.75	0.8	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.5082	0.5075	99.86
5	0.8	0.85	0.8	0.9	0.75	0.8	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.6194	0.6188	99.90
7	0.8	0.85	0.8	0.9	0.75	0.8	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.7360	0.7358	99.97
11	0.8	0.85	0.8	0.9	0.75	0.8	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.9626	0.9634	99.92
5	0.75	0.8	0.9	0.7	0.8	0.8	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.6020	0.6049	99.52
7	0.75	0.8	0.9	0.7	0.8	0.8	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.6881	0.6852	99.58
9	0.75	0.8	0.9	0.7	0.8	0.8	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.7688	0.7656	99.58
13	0.75	0.8	0.9	0.7	0.8	0.8	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.9194	0.9192	99.98
15	0.75	0.8	0.9	0.7	0.8	0.8	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.9907	0.9908	99.99
5	0.8	0.7	0.85	0.8	0.7	0.75	0.7	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.6021	0.6027	99.90
9	0.8	0.7	0.85	0.8	0.7	0.75	0.7	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.7688	0.7693	99.93
13	0.8	0.7	0.85	0.8	0.7	0.75	0.7	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.9195	0.9209	99.85
15	0.8	0.7	0.85	0.8	0.7	0.75	0.7	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.9907	0.9909	99.98
5	0.8	0.9	0.9	0.75	0.8	0.7	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.6021	0.6020	99.99
9	0.8	0.9	0.9	0.75	0.8	0.7	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.7689	0.7681	99.90
13	0.8	0.9	0.9	0.75	0.8	0.7	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.9192	0.9195	99.97
2	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.4500	0.4503	99.93
4	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.5564	0.5451	97.98
6	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.6200	0.6196	99.94
8	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.6538	0.6511	99.58
11	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.8008	0.8005	99.97
16	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.9715	0.9927	97.81
17	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.8749	0.8766	99.81
18	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.067	0.05	0.05	0.083	0.05	0.033	0.067	0.8830	0.8836	99.93

Once the training is complete, the BPN model is ready for use. Several sets of inputs were tested and documented. In order to demonstrate the application some of the results are reported and presented. The results from the analytical method and from the trained BPN are tabulated in Table 1. Comparative results indicate that there is a close agreement between both methods. The proposed BPN method seems to work well and provides fast and reliable results and can be used as an alternate method to determine the makespan *M*. However, the *k* is limited up to 20 for this BPN model. Having more lots into the system, the accuracy of the BPN model will be lost. This is due to the fact that the learning curve being a nonlinear function creates greater distortion on time computation when *k* increases significantly.

6. CONCLUSION

This paper presents an alternate method and a well defined BPN approach to compute the makespan of set of *k* lots applied to a tandem network. In order to illustrate further, the BPN model is applied to a 7 stage tandem system. The contribution of this paper is that, learning experience is considered in the computation work. The learning curve is quantified using a coefficient and each stage has its own learning experience. Several combinations of k, ϕ_n and T_{1n} in a defined job are considered and their solutions are computed using the analytical method. Then using these sets of input and output vector pairs, the Back Propagation Network is trained. Thereafter, the BPN is ready for use wherein, given number of lots, it gives out the makespan as a solution with minimum time and maximum accuracy. Results show that there is a close agreement between the analytical and BPN method and number of jobs has to be limited to below 20. Results indicate that the BPN approach can be used as an alternate method apart for the conventional analytical method. The benefit of this approach is that it provides a fast and reliable solution after a good set of training being done.

REFERENCES

[1] Argote L., Epple D. (1990), Learning curves in manufacturing; *Science* 247(4945); 920-924.

[2] Bohlen G.A., Barany J.W. (1976), A learning curve prediction model for operators performing industrial bench assembly operations; *International Journal of Production Research* 14(2); 295-303.

[3] Guo Z.X., Wong W.K, Leung S.Y.S, Fan F.T (2009), Intelligent production control decision support system for flexible assembly lines; *Expert Systems with Applications: An International Journal* 36(8); 4268- 4277.

[4] Kumar S., Omar M.K. (2005a), Stochastic re-entrant line modeling for an environmental stress testing in a semiconductor assembly industry; *Applied Mathematics and Computation* 173(1); 603-615.

[5] Kumar S., Omar M.K. (2005b), Performance measure in a probabilistic reflow screening line using mean value analysis; *AIUB Journal of Science and Engineering* 4(1); 53-58.

[6] Kumar S. (2007), Performance analysis of a probabilistic re-entrant line in an environmental stress testing operation; *Doctoral Thesis*; Multimedia University, Malaysia.

[7] Kumar S. (2008), Modeling of a probabilistic re-entrant line bounded by limited operation utilization time; *Journal of Industrial and Systems Engineering* 2(1); 28-40.

[8] McCollum P. (1997), An introduction to back-propagation neural networks; *The Newsletter of the Seattle Robotics Society*; November.

[9] Moghaddam R.T., Kanani Y.G., Cheraghalizadeh R. (2008), A genetic algorithm and memetic algorithm to sequencing and scheduling of cellular manufacturing systems; *International Journal of Management Science and Engineering Management* 3(2); 119-130.

[10] Shihab K. (2006), A back propagation neural network for computer network security; *Journal of Computer Science* 2(9); 1549-3636.

[11] Stafford E.F., Tseng F.T. (2002), Two models for a family of flowshop sequencing Problems; *European Journal of Operational Research* 142; 282-293.

[12] Stevenson W.J. (2006), Operations Management; 9th ed. McGraw-Hill; New York, 349-357.