International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:9, 2008

A Subjective Scheduler Based on Backpropagation Neural Network for Formulating a Real-life Scheduling Situation

K. G. Anilkumar[1] and T. Tanprasert[2]

Abstract**—**This paper presents a subjective job scheduler based on a 3-layer Backpropagation Neural Network (BPNN) and a greedy alignment procedure in order formulates a real-life situation. The BPNN estimates critical values of jobs based on the given subjective criteria. The scheduler is formulated in such a way that, at each time period, the most critical job is selected from the job queue and is transferred into a single machine before the next periodic job arrives. If the selected job is one of the oldest jobs in the queue and its deadline is less than that of the arrival time of the current job, then there is an update of the deadline of the job is assigned in order to prevent the critical job from its elimination. The proposed satisfiability criteria indicates that the satisfaction of the scheduler with respect to performance of the BPNN, validity of the jobs and the feasibility of the scheduler.

Keywords**—**Backpropagation algorithm, Critical value, Greedy alignment procedure, Neural network, Subjective criteria, Satisfiability.

I. INTRODUCTION

THIS paper presents a subjective job scheduler based on a BPNN and a greedy alignment procedure. The application of the greedy algorithms [1], [2] along with the BPNN section of the scheduler always provides the best and immediate solutions without considering the details of all existing legal solutions for a scheduling problem. The scheduler concept that is implied in this paper is based on a real life situation; as per our daily life routines, certain jobs need to be finished on or before a given time. Eventually, it is possible to notice that, while handling a current job, the chances of the arrival of unexpected job(s) cannot be avoided and which may be either from a higher authority or from a normal level. In such a situation, the most critical job will be given the first priority. Similarly, in order to adopt such a real life situation in the form of a scheduler, it is essential to consider the subjective criteria as a major part of the scheduler which is nothing but, the solution views and plans towards a problem based on a particular human being.

[1]Lecturer, Department of Computer Science, Faculty of Science and Technology, Assumption University, (e-mail: anil@scitech.au.edu).
[2]Professor, Department of Computer Science, Faculty of Science and Technology, Assumption University, Soi 24, Ram Khamhaeng Road, Hua Mak, Bang Kapi, Bangkok 10240, Thailand, (e-mail: nui@scitech.au.edu).

Moreover, based on the nature of the mentioned real life situation, the scheduler needs to support only a single machine scheduling problem that resembles a humanly situation. Similarly, despite of the complexity of the real life problem, instead of sporadic jobs, the concept of the periodic jobs would be considered in this paper. Besides, this paper is based on our previous research works [3]-[7].

As per the concept of the proposed scheduler, initially, the job queue is maintained with a set of static jobs which initialize the scheduler with a set of static jobs before any periodic job. Thereafter, the job queue consists of periodic jobs that are arrived in a predefined time intervals. Soon after, a predefined time period is reached, the scheduler provides the *Finishing Time* (*FT)* of the entire job set. Even then the proposed cost evaluation shows that the feasibility of the scheduler, the validity test along with the scheduler will test the whether the periodic jobs are exactly based on the given subjective criteria or not.

The remainder of this paper is organized as follows: The structure of the proposed scheduler is described in section II. The description of the working of the scheduler is described in section III. In Section IV, details of the initial dataset of the scheduler are described. In section V, acceptance measure of the BPNN is described. In section VI, details of the input job validity test are described. In section VII, cost evaluation is described. In section VIII, satisfiability criteria of the scheduler are described. In section IX, details of the scheduler procedure are described. The simulation results are shown in section X. Finally, Section XI concludes this paper.

II. STRUCTURE OF THE SCHEDULER

This section describes certain notations which are used to formulate a single machine job scheduling problem in the following way: denotes $J = \{J_1, .. , J_n\}$ and M as a set of independent jobs and the machine. Each job is assumed to handle T sub tasks; $J_1 = \{T_{11}, .. , T_{1n}\}$ and each sub task of a job is represented as a set of attribute values, for example, a task, T_1 can be represented as the conjunction of its attributes, $\{a_{11} \wedge a_{12} \wedge \ldots . \wedge a_{1n}\}$, where a_{11}, a_{12},.., etc., are the of attributes of the task, T_1. The following attribute values are given to each task of a job;

a. *mood_factor* (*M*): Mood factor indicates mood of human mind while receiving or selecting a job/task. However this concept of human mood expression is incomplete and as per human psychology mood expressions are a non-linear set.
b. *Toughness* (*T*): Toughness indicates the toughness felt by a human while handling/managing a job/task.
c. *Duration* (*D*): Duration indicates the allotted execution time of a job/task and finally,
d. *Acceptance_level* (*A*): Acceptance level indicates the various decision level taken by humans in order to accept a job/task.

Let M_{ik}, T_{ik}, D_{ik} and A_{ik} are represent the *mood_factor*, *toughness, duration* and *acceptance_level* of task *k* of a job *i.* The uniqueness of this proposed scheduler is that which select the most critical job from the job queue at each time period and send to the machine without violating the task precedence order. It is due to the proposed deadline specification of the scheduler, there is no job preemption that is needed. The scheduler is formulated in such a way that the most critical job at a time is selected from a set of jobs and then sends each critical job to the machine before its deadline. In case of a new periodic job arrives, and then a new search is made by the scheduler for the next most critical job among the existing jobs. Similarly the scheduler will continue its critical job selection until a predefined time is reached. This process of the scheduler resembles a human like decision style; as per a human concept, jobs are processed by one machine and are always a low critical job is replaced by a high critical job. The structure of the scheduler is shown in Fig.1. The details of the initial dataset are described in section V. Similarly the details of the job validity test and cost evaluation are described in section VII and VIII respectively.

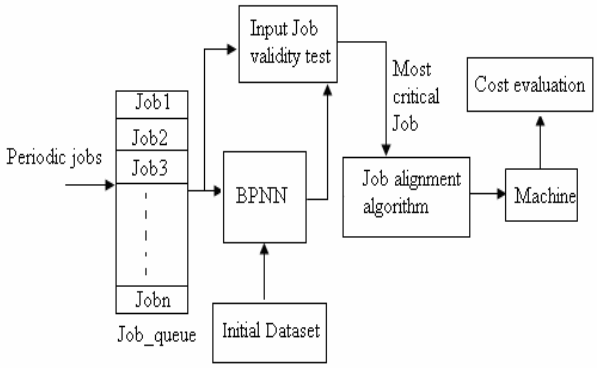

Fig. 1 Structure of the scheduler with input job

After testing with various possible network topologies, it is found that a 3-layer BPNN with topology 4 15 1 (one input layer with four inputs, one hidden layer with fifteen neurons, and one output layer with one output) is a suitable one with a learning rate 0.4. The BPNN with four input variables and one output variable is shown in Fig. 2. The BPNN is trained on examples which take the form of mapping $f: S \subset \Re^n \rightarrow \Re^m$, from some arbitrary bounded subset *S* of *n*-dimensional Euclidean space to *m*-dimensional Euclidean space. When an activity pattern is applied to the network, the error correcting rule adjusts the synaptic weights in accordance with the above mapping. The actual response of the network is subtracted from the desired target response to produce the error signal. Weights are adjusted so the actual network response moves closer to the desired response [8].

There are four parameters that are used as the inputs to the BPNN: (1) M_i, mood_factor related to task *i*, (2) T_i, toughness of task *i,* (3) D_i, duration of task *i,* and (4) A_i, acceptance_level of task *i*. The BPNN has one output: C_i, criticality of task, *i.*

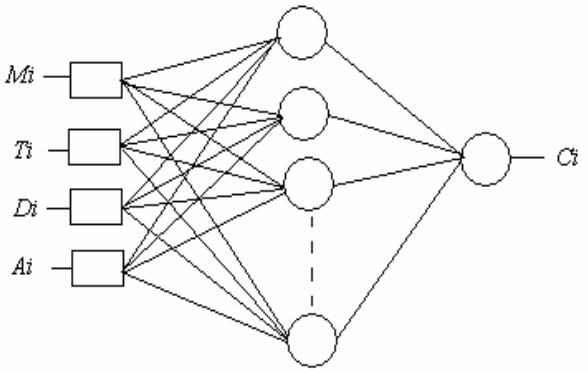

Fig. 2 A 3-layer BPNN with 4 input parameters and an output

III. DESCRIPTION OF THE WORKING OF THE SCHEDULER

Initially, the job queue is initialized with a set of static jobs (3 *jobs*) with their critical values (*it is assumed that a job has* 4 *subtasks*) by the scheduler. After the selection of the most critical job among the given static jobs, a periodic job is generated and then added to the job queue at each *time period* (T_a). In the beginning of the scheduler process, T_a is always zero. After each periodic job, the scheduler detects the most critical job from the job queue and sends it to the machine at each time period. Then the selected job will be removed from the queue. Besides, if the critical value of the newly arrived job is lower than that of existing jobs, then the scheduler will search into the queue for the most critical job. In the case that the next most critical job is one of the oldest jobs then the scheduler will automatically updates that job's deadline and passes to the given machine. That is, the scheduler updates the deadline of the selected job in order to prevent that job from elimination. In order to maintain the deadline of a critical job, there is an updating deadline rule that is introduced, which is defined as follow:

If the deadline of a critical job is less than the arrival time of the present job, modify the deadline of the critical job by adding the arrival time of the present job. As per the deadline modification rule, new deadline of a job can be given as;

$$J_{new\_deadline} = \text{Deadline of the selected job} + \text{arrival time of the current job} \quad (1)$$

IV. INITIAL DATASET GENERATION

In order to generate an initial training dataset for the scheduler, there are five numerical values with their linguistic terms are used along with each task attributes. The four task attributes with their numerical values are given below:

- M_i, mood_factor of task *i* have values: [0.1 (*very low*), 0.3 (*low*), 0.5 (*not low)*, 0.7 (*high*), 0.9 (*very high*)].
- T_i, toughness of task *i* have values: [0.1 (*very less*), 0.3 (*less*), 0.5 (*not less)*, 0.7 (*high*), 0.9 (*very high*)].
- D_i, duration of task *i* have values: [0.1 (*very low*), 0.3 (*low*), 0.5 (*not low)*, 0.7 (*high*), 0.9 (*very high*)].
- A_i, acceptance_level of task *i* have values: [0.1 (*very low*), 0.3 (*low*), 0.5 (*not low)*, 0.7 (*high*), 0.9 (*very high*)].

Based on the given attributes of seen job set, the following criteria are applied to the scheduler (the criteria may vary with a different human model) for finding the critical task:

-mood_factor $\propto$ 1/ *toughness*.
-mood_factor $\propto$ 1/*duration*.
-mood_factor $\propto$ *acceptance_level*.
-toughness $\propto$ *duration*.
-toughness $\propto$ 1/*acceptance_level*.
- duration $\propto$ 1/*acceptance_level*.

- A *very high/high* acceptance_level task with *very high/high* duration and *very high/high* toughness must hold *high/very high* criticality.
- A *very high/high* mood_factor can keep any tough task with any duration in a *very high/high* acceptance_level and hence the task's criticality can be *very high/high*.
- A *verylow/low* mood_factor can make any tough task with any duration in a *very low/low* acceptance_level and hence the task's criticality can be *very low/low.*
- A *very high/high* tough task with any duration can make mood_factor *very low/low* and hence the acceptance_level and the criticality.
- A very *high/high* duration task with *very high/high* toughness can make *very low/low* mood_factor and acceptance_level, hence the criticality can be *very low/low*.
- A *very low/low* acceptance_level task with *very low/low* mood_factor can make *very low/low* critical job.
- A task with *not low* mood_factor and with *not less* toughness and with any duration and with *not low* acceptance_level can make it as a *not low* critical one.
- A *very high/high* toughness task with *very high/high* duration can cause *low/very low* mood_factor and acceptance_level, hence that can cause *very low/low* critical value.
- A task with *not low* mood_factor and with *very less/ less* toughness and with *very low/low* duration can cause *not low /high* acceptance_level, hence the task criticality can be *not low/high*.

There are 100 different inputs and their respective output data patterns are created based on the mentioned criteria for the scheduler.

V. ACCEPTANCE MEASURE OF THE BPNN

The initial dataset of the problem is based on the given subjective criteria and that dependence on the views of a particular human model that wishes to solve the problem accordingly. Also the initial dataset training depends on the size of the dataset and the topology of the selected BPNN. Once the BPNN is trained properly (i.e. trained until its Mean Squared Error (MSE) is less than 0.01) with the given initial dataset, then it is possible to say that the scheduler is set for the view of a particular human model towards the solution plans of a problem. Even if, the initial dataset is trained by the selected BPNN as per the MSE value, it is essential to ensure that the performance of the selected BPNN is free from problems such as 'over-fitting' and local maxima. Therefore, there is an acceptance measure that is provided along with the BPNN section of the scheduler. The details of the acceptance criteria proposed for the BPNN are described as follows:

I. First, it is to train the given initial dataset with the proper network topology and training parameters such as *learning rate* (α) and *momentum term* (β) until its MSE is reduced to a value less than 0.01. The dataset used in the initial training is called as *seen data*.

a) *Seen dataset* is a set of data which is generated based on the adopted subjective criteria and is used for the initial training of the BPNN.
b) *Unseen dataset* is a data set which is not involved in the initial training of the BPNN (which is either from the user or from the random generator).

II. Select a set of seen outputs (say, *P*) from the initial dataset after its training.

III. Input the same dataset (without output data) to the scheduler and check the output given by the BPNN (say, *P'*).

IV. Based on the similarity measure (*S*) of *P* and *P'*, $S_{(P, P')}$; the performance measure of the BPNN can be defined as follows;

c) *If* $S_{(P, P')}$ *is above or equal to* +0.99, *then the performance of the* BPNN *is acceptable.*
d) *If* $S_{(P, P')}$ *is below* +0.99*, then the performance of the* BPNN *is not_accepatble.*

This paper uses a correlation coefficient method [9] in order to measure the similarity between two types of equal size datasets for a given problem. The correlation coefficient is a standardized form of angular separation by centering the coordinates to its mean value. Thus the higher value of the angular separation indicates that the two objects are similar. That is, the correlation coefficient results values between -1 and +1 based on the datasets. The details of the mathematical formulation of the correlation coefficient are described as:

Let $S_{i,j}$ is the normalized similarity between two sets of attribute values X_i and X_j of datasets *i* and *j*. The formulation of $S_{i,j}$ is given as;

$$S_{i,j} = \left(\sum_{k=1}^{n}(X_{i,k}-\overline{X_i})*(X_{j,k}-\overline{X_j})\right)/\left(\sum_{k=1}^{n}(X_{i,k}-\overline{X_i})^2 * \sum_{k=1}^{n}(X_{j,k}-\overline{X_j})^2\right]^{1/2} \quad (2)$$

Where $\overline{X_i} = 1/n * (\sum_{k=1}^{n} X_{i,k})$ and $\overline{X_j} = 1/n * (\sum_{k=1}^{n} X_{j,k})$

As per the conditions of the proposed scheduler, the scheduling process is based on the input dataset and which must be based on the given subjective criteria. In order to ensure whether the input dataset (which is either from the user or from the random generator) to the scheduler is exactly based on the subjective criteria or not, there is an input job validity test is proposed along with the scheduler**.** As like the mentioned BPNN acceptance test, the proposed input data validity test is based on the mentioned correlation coefficient and is described in the following section.

VI. VALIDITY TEST OF THE INPUT JOB

The input validity test (*V*) of the scheduler provides the degree of measure of the input (unseen) of the scheduler with respect to the original dataset (seen dataset) which is used in the initial training section of the BPNN. That is, the input job validity test of the scheduler depends on the inputs and output of the BPNN with seen and unseen job attributes for a given problem.

In this scheduler, each input job has a set of 4 sub tasks and each task has 4 attribute values. Hence a set of *n* jobs has a size of 2*n* x *n* (i.e., 2*n* rows and *n* columns) unseen data set. Similarly, a set of seen data with the size of the given unseen data is considered as an ideal subjective dataset for measuring the similarity of the given input dataset of the scheduler for a problem. Let *i*, is the seen dataset and *j*, is the unseen dataset for a problem and X_i, X_{i+1},…, X_{i+n}, are the *n* attributes of *i* and X_j, X_{j+1},…, X_{j+n}, are the *n* attributes of *j* (the sizes of *i* and *j* are same). The similarity $S_{i,j}$ of *i* and *j* based on the (2) can be given as;

$$S_{i,j} = (\sum_{k=1}^{n} (\overline{X_{i,k}} - \overline{Y_i}) * (\overline{X_{j,k}} - \overline{Y_j})) / [\sum_{k=1}^{n} (\overline{X_{i,k}} - \overline{Y_i})^2 * \sum_{k=1}^{n} (\overline{X_{j,k}} - \overline{Y_j})^2]^{1/2} \quad (3)$$

Where $\overline{X_{i,k}} = 1/n * (\sum_{i=1}^{n} X_{i,k})$, $\overline{X_{j,k}} = 1/n * (\sum_{j=1}^{n} X_{j,k})$,

$\overline{Y_i} = 1/n * (\sum_{k=1}^{n} \overline{X_{i,k}})$ and $\overline{Y_j} = 1/n * (\sum_{k=1}^{n} \overline{X_{j,k}})$.

Incase of an input job is not based on the given subjective criteria of the scheduler, and then obviously, that job is invalid. Based on this concept, the validity checking of the input of the scheduler is given as:

If (S > 0), then it is assumed that the unseen job is based on the adopted subjective criteria and is valid. Otherwise the unseen job is not matching with the adopted subjective criteria and is invalid. A Job validity scale based on (3) is shown in Fig. 3.

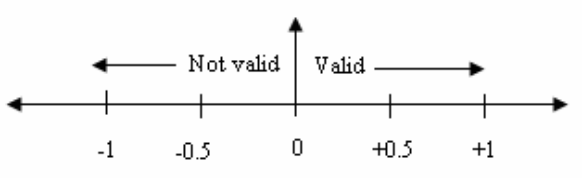

Fig. 3 Job validity scale

VII. COST EVALUATION

In a single machine scheduling problem, the given jobs/subtasks are processed by a machine in their precedence order. The total time duration of the tasks of *n* jobs before they are processed by the machine, *M* is given as, $\sum_{i=1}^{n} T_i$, where T_i is the duration of the job, *i*. Hence the *Approximate Finishing Time* (FT_a) of the given jobs before their execution can be given as,

$$FT_a = (total\_time\_period - \sum_{i=1}^{n} T_i) \quad (4)$$

Where, *total_ time_ period* is the sum of the periodic values and the *arrival time* of the last job into the scheduler. Similarly, $\sum_{i=1}^{n} T_{T1}$ is the total duration of tasks in job_1 and $\sum_{i=1}^{n} T_{Tn}$ is the total duration of tasks in job_n, respectively. Therefore, the *FT* of the *Entire Schedule* (FT_e) of *n* jobs on *M* is given as,

$$FT_e = (T_{a,1} + \sum_{i=1}^{n} T_{T1} + \ldots + T_{a,n} + \sum_{i=1}^{n} T_{Tn}) \quad (5)$$

Where $T_{a,1}$, $..,T_{a,n}$ are the arrival times of $job_1,\ldots,job_n$. From (4) and (5), the cost value (C_{cost}) of the entire schedule can be given as,

$$C_{cost} = [FT_e - FT_a] \quad (6)$$

C_{cost} based on (6) can be defined as:

$$C_{cost} \begin{cases} = 0 & good\_enough \\ > 0 & reasonable \\ < 0 & not\ reasonable \end{cases}$$

Let $X = FT_a$, then the *Relative Error* (*RE*) of the scheduler can be given as,

$$RE = (FT_e - X)/X \quad (7)$$

For a *good_enough* schedule, the *RE* is always zero. It is due to the nature of (7), the '*not reasonable*' cost result is not exists in the scheduler. Hence the feasibility of the scheduler is either *good_enough* or *reasonable* based on the given cost definition.

VIII. SATISFIABILITY DESCRIPTION

Satisfiability (*Sat*) of the scheduler indicates that the value of satisfaction (which is either *true* or *false*) of the scheduler for a given problem based on the given subjective criteria and the greedy procedure. In order to indicate the satisfiability of the scheduler for a given problem, there are three points considered in this paper; (a) Performance of the BPNN (b) Validity of the input job (with a threshold value above +0.5) and (c) the feasibility of the solutions (with a cost value 0 or below +0.5). The following model shows how to formulate *Sat* of the scheduler; Let O is the optimal performance of the BPNN, V is the validity of the input job set and C is the feasible solution from the scheduler. In order to find the *Sat* of the scheduler, it is considered that the given variables O, V and C are atomic in nature. That is, if O is *true* means the BPNN is optimal and *false* means not optimal. Similarly, the variables V and C have two values; either *true* or *false* depends on the scheduler. The logical combinations of *Sat* (which is either *true* or *false*) based on the given three atomic variables can be represented as;

(a). If (O is *false*) and (V is *false*) and (C is *false*) then (*Sat* is *false*).
(b). If (O is *false*) and (V is *false*) and (C is *true*) then (*Sat* is *false*).
(c). If (O is *false*) and (V is *true*) and (C is *false*) then (*Sat* is *false*).
(d). If (O is *false*) and (V is *true*) and (C is *true*) then (*Sat* is *false*).
(e). If (O is *true*) and (V is *false*) and (C is *false*) then (*Sat* is *false*).
(f). If (O is *true*) and (V is *false*) and (C is *true*) then (*Sat* is *false*).
(g). If (O is *true*) and (V is *true*) and (C is *false*) then (*Sat* is *false*).
(h). If (O is *true*) and (V is *true*) and (C is *true*) then (*Sat* is *true*).

The propositional logic representation of *Sat* with respect to the atomic variables O, V and C can be given as;

$$((O \wedge V \wedge C) \rightarrow Sat) \quad (8)$$

The given logic (8) can further be simplified into its disjunctive normal form (DNF) in propositional logic;

$$\sim(O \wedge V \wedge C) \vee Sat)$$
$$(\sim O \vee \sim V \vee \sim C \vee Sat) \quad (9)$$

Equation (8) means that, if O, V and C are *true*, then it is possible to claim that *Sat* is *true*. Else, it is not possible to claim that *Sat* is *true*. From this point, it is clear that the true *Sat* depends on the optimal performance of the BPNN, valid input job set and a feasible solution.

IX. DETAILS OF THE SCHEDULER PROCEDURE

Details of the procedure of the scheduler are given below:

I. Backpropagation algorithm is used to train the BPNN to get the critical values of tasks of each queued job (*it is assumed that a job has 4 sub tasks*).
II. Perform BPNN acceptance test.
III. Tasks precedence order of a job is estimated from the output of the BPNN by sorting their critical values.
IV. Estimate the critical value of a job, J_i by comparing critical values of tasks within a job and is given as; $J_{i,critical} = max\ (T_{i1,critical}, ..., T_{i4,critical})$, where $T_{i1,critical}, \ldots, T_{i4,critical}$ are the critical values of tasks, T_{i1}, T_{i2}, T_{i3} and T_{i4} of J_i.
V. Stores jobs with critical values back into the job queue.
VI. Select the most critical job, $J_{Most\_critical}$ from the job queue;

$J_{Most\_critical} = \max (J_{i,critical}, \ldots, J_{n,critical})$, where $i \in (1, n)$

VII. Estimate the duration $J_{duration}$ of the job, J_i can be estimated by adding durations of all its subtasks;

$$J_{i,duration} = \sum_{i=1}^{n} D_i$$, where D_i is the duration of task i.

VIII. while (T_a < *allotted time*) { start *while loop*;
(a). At start time, $T_a = t$, select the most critical job, $J_{i,critical}$ from the queue, FT of the job, J_i can be calculated as; $FT_{j,i} = T_a + J_{i,duration}$. Similarly, deadline $J_{i,deadline}$ of the job, J_i can be given as (in general, deadline of a job is always greater than its duration); $J_{i,deadline} = FT_{j,i} + \alpha$, where α is wait factor and is considered as 0.1.
(b). The over all finishing time (FT_{over}) of the scheduler is equal to the FT of the last job.
(c). At $T_a = t + 1$, a periodic job, J_{R1} arrives and estimate its critical value and send to the job queue.
i. Select the most critical job, $J_{Most\_critical}$ from the queue.
ii. If the deadline of the most critical job, J_i is less than T_a, then update the deadline of that job;

$$J_{i,new\_deadline} = J_{i,deadline} + T_a$$

iii. If J_{R1} is selected as the most critical job, then there is no need to update the deadline of that job.

IX. If (T_a is equal to *allotted time*), then stop the *while loop* and estimate the over all finishing time (FT_{over}) of the scheduler which is equal to the FT of the lastly selected job. Then, evaluate the cost of the scheduler and show the satisfiability. Initialize the job queue for the next job set.

X. SIMULATION RESULTS

In this simulation, the initial job set has 3 static jobs with a starting time (T_a) is 0 and each periodic job arrives at every 3 seconds. Also this simulation is restricted with maximum of 3 periodic jobs. Descriptions of the simulations carried out are given below:

A. *Job queue with* 3 *initial jobs* (JOB_1, JOB_2, and JOB_3) *at* $T_a = 0$

TABLE I
TASK ORDERS OF JOB_1, JOB_2 AND JOB_3 BASED ON THEIR CRITICAL VALUE

Job name	Task order	Critical value
JOB_1	T_{12}	0.40704
JOB_1	T_{14}	0.31789
JOB_1	T_{13}	0.24011
JOB_1	T_{11}	0.14208
JOB_2	T_{22}	0.64299
JOB_2	T_{23}	0.47487
JOB_2	T_{24}	0.38115
JOB_2	T_{21}	0.20675
JOB_3	T_{31}	0.47806
JOB_3	T_{32}	0.41785
JOB_3	T_{34}	0.32727
JOB_3	T_{33}	0.32252

Table 1 shows the most critical job in the job queue is JOB_2 with a critical value, 0.64299 at time, T_a is 0. Hence JOB_2 is selected by the scheduler for processing on machine, M. After that JOB_2 will be removed from the queue. The processing sequence of JOB_2 is shown in Fig. 4. The task durations are shown in brackets. Where $Start_1$ (0), FT_1 (2.7), and $Deadline_1$ (2.8) are the starting time (T_a), finishing time and deadline of the job, JOB_2.

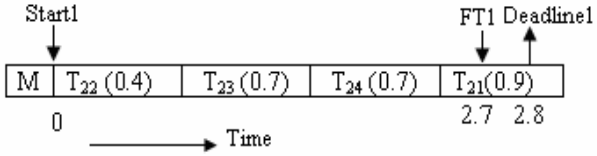

Fig. 4 Processing sequences of JOB_2 on machine, M

B. *Periodic job,* JOB_4 *enters into the job queue at* $T_a = 3$

Table 2 shows that the most critical job in the job queue is JOB_3 with a critical value, 0.47806 at T_a is 3. It is noticed that the T_a of job, JOB_3 is earlier than that of the current job, JOB_4. Therefore the scheduler needs to update the deadline of the job, JOB_3. Hence after the job, JOB_2 the scheduler selects JOB_3 for processing. After that, JOB_3 will be removed from the queue. The processing sequences of JOB_2 and JOB_3 are shown in Fig. 5 (task durations are shown in brackets). Where, $Start_2$ (3), FT_2 (3.5), and $Deadline_2$ (3.6) are the starting time (T_a), finishing time and deadline of JOB_3 are given respectively.

TABLE II
TASK ORDERS OF JOB_1, JOB_3, AND JOB_4 BASED ON THEIR CRITICAL VALUE

Job name	Task order	Critical value
JOB_1	T_{12}	0.40704
JOB_1	T_{14}	0.31789
JOB_1	T_{13}	0.24011
JOB_1	T_{11}	0.14208
JOB_3	T_{31}	0.47806
JOB_3	T_{32}	0.41785
JOB_3	T_{34}	0.32727
JOB_3	T_{33}	0.32252
JOB_4	T_{41}	0.41015
JOB_4	T_{43}	0.37528
JOB_4	T_{44}	0.35216
JOB_4	T_{42}	0.26620

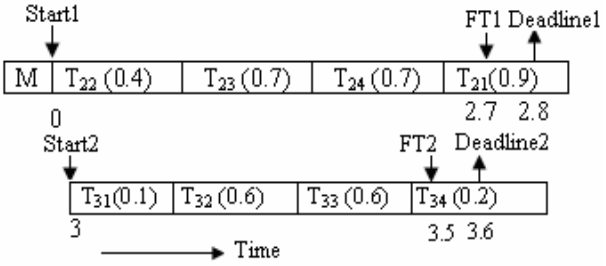

Fig. 5 Processing sequences of JOB_2 and JOB_3 on machine, M

C. *Periodic job,* JOB_5 *enters into the job queue at* $T_a = 6$

Table 3 shows that the most critical job is JOB_5 with a critical value, 0.80576 at time, T_a is 6 and it is noticed that *Ta* of the job, JOB_5 is later than that of the previous job, JOB_3. Hence there is no need to update the deadline of the job, JOB_5. After that, JOB_5 will be removed from the queue. The processing sequences of JOB_2, JOB_3 and JOB_5 are shown in Fig. 6, where, $Start_3$ (6), FT_3 (7.72), and $Deadline_3$ (7.82) are the starting time (T_a), finishing time and deadline of JOB_5 are given respectively.

TABLE III
TASK ORDERS OF JOB_1, JOB_4 AND JOB_5 BASED ON THEIR CRITICAL VALUE

Job name	Task order	Critical value
JOB_1	T_{12}	0.40704
JOB_1	T_{14}	0.31789
JOB_1	T_{13}	0.24011
JOB_1	T_{11}	0.14208
JOB_4	T_{41}	0.41015
JOB_4	T_{43}	0.37528
JOB_4	T_{44}	0.35216
JOB_4	T_{42}	0.26620
JOB_5	T_{51}	0.80576
JOB_5	T_{52}	0.58915
JOB_5	T_{54}	0..53737
JOB_5	T_{53}	0.14700

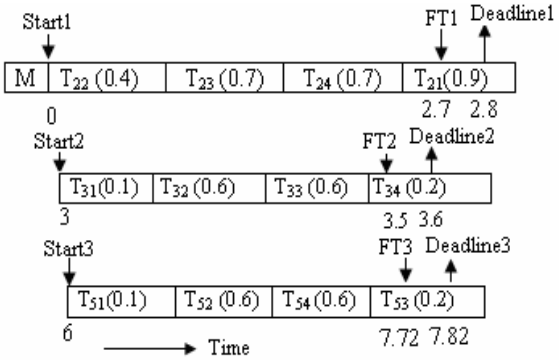

Fig. 6 Processing sequences of JOB_2, JOB_3 and JOB_5 on machine, M

D. Periodic job, JOB_6 enters into the job queue at $T_a = 9$

Table 4 shows that the most critical job is JOB_6 (with a critical value 0.81126) at time, T_a is 9. It noticed that T_a of the job, JOB_6 is later than that of the jobs, JOB_3 and JOB_5. Hence there is no need to update the deadline of the job, JOB_6. The processing sequences of JOB_2, JOB_3, JOB_5 and JOB_6 are shown in Fig. 7, where $Start_4$ (9), FT_4 (11.6), and $Deadline_4$ (11.7) are the starting time (T_a), finishing time and deadline of JOB_6 are given respectively. The cost evaluation indicates that the performance of the scheduler is *reasonable* (*RE* is not zero).

TABLE IV
TASK ORDERS OF JOB_1, JOB_4, AND JOB_6

Job name	Task order	Critical value
JOB_1	T_{12}	0.40704
JOB_1	T_{14}	0.31789
JOB_1	T_{13}	0.24011
JOB_1	T_{11}	0.14208
JOB_4	T_{41}	0.41015
JOB_4	T_{43}	0.37528
JOB_4	T_{44}	0.35216
JOB_4	T_{42}	0.26620
JOB_6	T_{61}	0.81126
JOB_6	T_{63}	0.43851
JOB_6	T_{62}	0.42691
JOB_6	T_{64}	0.29359

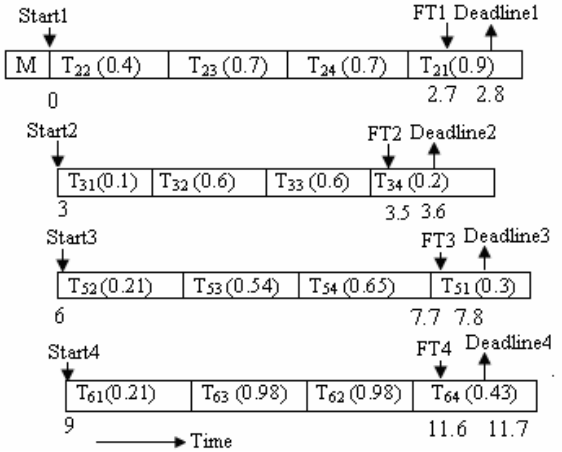

Fig. 7 Processing sequences of JOB_2, JOB_3, JOB_5 and JOB_6 on machine, M

The selected BPNN is an acceptable one with a similarity value, +0.9985. Similarly, JOB_1, JOB_2, JOB_3, JOB_4, JOB_5, and JOB_6 are valid due to their positive validation values, +0.567, +0.456, +0.651, +0.712, +0.451, and +0.721respectively. Hence the satisfiability of the scheduler is *true*. The deadline update specification for the selected critical jobs indicates that once a job is selected, it will never miss its deadline. This is similar to that of a human nature that tries to finish a critical job regardless of the indicated deadline. Fig. 8 shows the validity test results of 11 jobs (each job has 4 sub tasks). Where first 3 jobs are initial jobs and the left are periodic jobs. From the Figure, there are 6 jobs are valid, {1, 2, 4, 9, 10, 11} and 5 jobs are invalid, {3, 5, 6, 7, 8}.

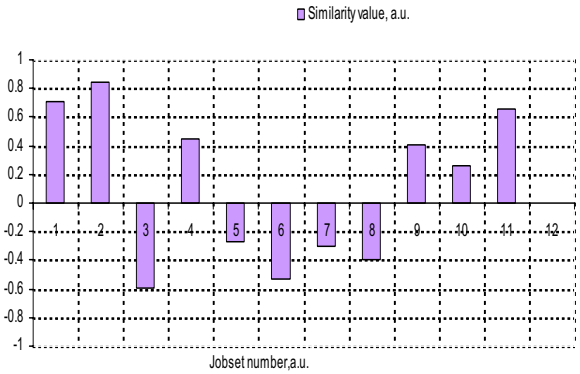

Fig. 8 Validation results of 11jobs

III. CONCLUSION

The significance of this scheduler approach is that which deals with subjective criteria for job selection where the exact criteria for finding a critical job are not formally described. That is, as per human nature, the critical job issue is a subjective one and therefore the criteria for finding a critical job depend on individual's view. The carried out simulations point out that the proposed scheduler approach is an efficient way for formulating a real life situation and is suitable for robotics application. Also a notable point from this research work is that with the help of backpropagation algorithm, and a proper greedy algorithm, it is possible to model a complex scheduling situation in an effective way.

REFERENCES

[1] W. Stinson, "An Introduction to the Design and Analysis of Algorithms", Cambridge University press, 1980, pp.70-103.
[2] T. H. Cormen, C.E. Leiserson, R. L. Rivest and C. Stein, "Introduction to Algorithms," The MIT Press: McGraw-Hill, 2001, pp. 370-399.
[3] K.G Anilkumar and T. Tanprasert, "Neural Network Based Priority Assigner for Job Scheduler", AU Journal of Technology, AU J.T.9 (3) 2006, pp. 181-186.
[4] K.G. Anilkumar and T. Tanprasert, "Neural Network Based Generalized Job-Shop Scheduler," in *Proc. 2nd IMT-GT Regional Conference on Mathematics, statistics and Applications*, Universiti Sains Malaysia, Penang, Malaysia, 2006, pp. 53 -58.
[5] K.G Anilkumar and T. Tanprasert, "Neural Network Bassed Greedy Job Scheduler," in *Proc. National Computer Science and Engineering Conference* (*NCSEC 2006*), Konkhean, Thailand, 2006, pp. 257-262.
[6] K. G. Anilkumar and T. Tanprasert, "Generalized Job-shop Scheduler Using Feed Forward Neural network and Greedy Alignment Procedure," in *Proc. IASTED Conference on Artificial Intelligence and Applications, AIA*-2007, Innsbruck, Austria, pp. 115-120.
[7] K. G Anilkumar and T. Tanprasert, "A Subjective Scheduler Based on Neural Network for Job Routing in a Generalized Job-Shop Problem", GESTS International Transactions on Computer Science and Engineering, vol.45, 2008, pp. 79-96.
[8] V. B. Rao and H. V. Rao, "Neural Networks & Fuzzy logic," BPB Publications, New Delhi, 1996, pp. 150-300.
[9] R. A. Johnson and D. W. Wichern, "Applied Multivariate Statistical Analysis," 5th edition, NJ: Prentice Hall, NJ, 2002, pp. 668-719.