CIMSA 2009 - International Conference on Computational Intelligence for Measurement Systems and Applications
Hong Kong, China
May 11-13, 2009

A Decomposition Based Algorithm for Flexible Flow Shop Scheduling with Machine Breakdown

K. Wang and S.H. Choi
Department of Industrial and Manufacturing Systems Engineering
The University of Hong Kong
Hong Kong, China
shchoi@hkucc.hku.hk

Abstract — **Research on flow shop scheduling generally ignores uncertainties in real-world production because of the inherent difficulties of the problem. Scheduling problems with stochastic machine breakdown are difficult to solve optimally by a single approach. This paper considers makespan optimization of a flexible flow shop (FFS) scheduling problem with machine breakdown. It proposes a novel decomposition based approach (DBA) to decompose a problem into several sub-problems which can be solved more easily, while the neighbouring K-means clustering algorithm is employed to group the machines of an FFS into a few clusters. A back propagation network (BPN) is then adopted to assign either the shortest processing time (SPT) or the genetic algorithm (GA) to each cluster to solve the sub-problems. If two neighbouring clusters are allocated with the same approach, they are subsequently merged. After machine grouping and approach assignment, an overall schedule is generated by integrating the solutions to the sub-problems. Computation results reveal that the proposed approach is superior to SPT and GA alone for FFS scheduling with machine breakdown.**

Keywords - flexible flow shop; decomposition based approach; neighbouring K-means clustering algorithm; back propagation network; machine breakdown

I. INTRODUCTION

Uncertainties, such as machine breakdown, rush orders, job cancellations, and change of due date, are inherent yet difficult phenomena in manufacturing production. Research efforts on production scheduling generally ignore uncertainties in the real-world environment. To address such difficult problems of scheduling under uncertainty, an emerging research area has recently attracted much study effort.

This paper is primarily concerned with the scheduling problem of flexible flow shop with machine breakdown. The flexible flow shop (FFS) scheduling problem [1] is a multi-stage production process, where jobs have to pass all the stages in the same order. This problem is NP-hard in nature and difficult to solve [2, 3]. Consideration of uncertainties adds further complexity to it. As a research issue, it has indeed drawn considerable attention in recent years. Focusing on production scheduling with uncertainty, the completely reactive approach, the robust approach, and the predictive-reactive approach are three fundamental ways [4, 5] to tackle this issue.

The completely reactive approach changes decisions during execution when necessary. The dispatching rule is a typical reactive one, in which jobs are selected by sorting them according to predefined criteria. It is easy to understand, and its computation cost is low. However, it uses only local information to generate a schedule which may not be globally optimal in nature. Hunsucker and Shah [6] compared the performance of dispatching rules in the constrained multiprocessor flow shop and concluded that Shortest Processing Time (SPT) and Longest Processing Time (LPT) were superior for the makespan criterion.

The robust scheduling approach takes into account possible uncertainty to construct solutions. Uncertainty, known as a priori, is modelled by some random variables [7]. If the uncertainty is difficult to quantify, a range of scenarios will be considered and a solution is developed to optimize the performance under different scenarios [8]. In this case, the approach is viewed as a form of under-capacity scheduling in order to maintain the robustness under different scenarios.

The predictive-reactive approach is indeed a two-step process. First, a predictive schedule is generated over the time horizon considered. This schedule is then rescheduled during execution in response to unexpected disruption. This approach is by far the most studied. The most common rescheduling methods include the right-shift schedule repair, the partial schedule repair, and the completed scheduling [9]. The right-shift schedule repair postpones the remaining operations by the amount of machine breakdown time. The partial schedule repair only reschedules the operations that are affected by the disruption. The completed scheduling regenerates a completely new schedule for all the unprocessed operations. Although the completed scheduling may construct a better solution in theory, it is rarely applied in practice due to high computation burden and increasing scheduling instability for the shop [7]. Conversely, the right-shift schedule repair yields the least scheduling instability with the lowest computation effort, while the partial schedule repair is a moderate one for scheduling instability and computation time.

Since each of these three approaches to production scheduling with uncertainty has its own strength and weakness, some research work has been focused on comparison of their effectiveness. Lawrence and Sewell [10] studied the static and dynamic applications of heuristic approaches to job shop

978-1-4244-3820-4/09/$25.00 ©2009 IEEE

scheduling problems with uncertain processing times. Experiment results indicated that the predictive methods based on overall information were highly likely to perform better than completely reactive approaches in an environment with little uncertainty. However, the predictive methods might lead to poor result when the uncertainty exceeds a certain level.

In order to handle a complex environment with uncertainties, it is imperative to take advantage of mixing these three approaches. Matsuura et al. [11] developed a predictive approach on a periodic basis, called switching. The system switched to using a dispatching rule for the remaining operations when the deviation between the realized and predictive schedule exceeded a certain level. They concluded that the proposed approach dominated the dispatching rules when the frequency of disruption was low, but it yielded worse results than the dispatching rules when the disruption reached some level. A search of available literatures indicates not much research works have been attempted to integrate different approaches to take advantage of their specific strengths.

This paper studies the problem of scheduling an FFS with machine breakdown uncertainty. The objective is to minimize the makespan. Enlightened by the work of Lawrence [10], we aim to investigate how to integrate the completely reactive approach with the predictive-reactive approach to solve the FFS scheduling problem under uncertainty. The integration takes advantage of combining the strengths of these two approaches to deal with uncertainty. As the genetic algorithm (GA) is the most widely used heuristic and SPT gives better performance for FFS scheduling, they are adopted to solve the scheduling problem.

In order to obtain better results, a decomposition based approach (DBA) is proposed. In this approach, a neighbouring K-means clustering algorithm first groups the machines of an FFS into several clusters based on their stochastic levels. Then SPT or GA, determined by the process of approach assignment, is employed to generate a sub-schedule for each cluster. Finally, these sub-schedules are integrated to give an overall one. Due to the consideration of computation effort and shop stability, we apply right-shift scheduling repair to react to machine breakdown. The main contribution of this study lies in the development of a hybrid scheduling approach, which integrates the completely reactive approach with the predictive-reactive approach, to deal with the uncertainty. On the contrary, most studies in the reviewed references only focused on developing a single approach to handling uncertainty in scheduling. The proposed DBA explores a new direction for future research in the field of scheduling with uncertainty.

The remaining part of this paper is organized as follows. Section II is devoted to problem description. Section III describes the framework of DBA, which is explained in detail in Section IV. To evaluate the effectiveness of DBA, simulation is conducted and computation results are analyzed in Section V. Finally, conclusions are summarized and some directions of future work are discussed in Section VI.

II. PROBLEM DESCRIPTION

In the FFS discussed above, machines are arranged into stages in series. The jobs have to pass all the stages in the same order. In each stage, there are a number of identical machines in parallel, and a job is to be processed on one of these machines.

In order to simplify the typical FFS scheduling problem in consideration of machine breakdown, the following assumptions are made: (1) Preemption is not allowed for job processing; (2) Each machine can process at most one operation at a time; (3) All jobs are released at the same time for the first stage; (4) For the same job, the processing time at any parallel machine in a stage is identical; (5) There is no travel time between machines; (6) There is no setup time for job processing; (7) Infinite buffers exist for machines; and (8) Job processing stops when a machine suffers stochastic breakdown and resumes immediately after repair.

The scheduling objective under consideration is to determine the processing sequence of operations on each machine such that the makespan, which is equivalent to the completion time of the last job to leave the FFS, is minimized without violating any of the assumptions above. This FFS scheduling problem can also be described as follows.

$$\min\{\max[C_{tj}]\} \tag{1}$$

Subject to the following constraints:

$$C_{1j} = P_{1j},\ if\ \sum_{i=1}^{m_1} U_{1ij} > 0 \tag{2}$$

$$C_{1j_2} = \sum_{i=1}^{m_1}\sum_{j_2=1}^{n}\left(B_{1ij_1j_2}\times C_{1j_1}\right) + P_{1j_2},\ if\ \sum_{i=1}^{m_1} U_{1ij_2} = 0 \tag{3}$$

$$C_{kj} = C_{(k-1)j} + P_{kj},\quad if\quad k>1\ \&\ \sum_{i=1}^{m_k} U_{kij} > 0 \tag{4}$$

$$C_{kj_2} = \max\{\sum_{i=1}^{m_k}\sum_{j_2=1}^{n}\left(B_{kij_1j_2}\times C_{kj_1}\right),\ C_{(k-1)j_2}\} + P_{kj_2},\quad if\quad k>1\ \&\ \sum_{i=1}^{m_k} U_{kij_2} = 0 \tag{5}$$

$$ST_{kij} \geq 0 \tag{6}$$

$$P_{ki_1j} = P_{ki_2j} = P_{kj},\quad (i_1,i_2)\in M_k \tag{7}$$

$$ST_{(k+1)i_1j} - ST_{ki_2j} \geq P_{ki_2j} \tag{8}$$

$$[(ST_{kij_1} - ST_{kij_2}) \geq P_{kij_2}]\quad or\quad [(ST_{kij_2} - ST_{kij_1}) \geq P_{kij_1}] \tag{9}$$

Where

k:	stage index, $1 \leq k \leq t$
m_k:	number of parallel machines at stage k
M_k:	set of parallel machines at stage k
i, i_1, i_2:	machine index, $1 \leq i, i_1, i_2 \leq m_k$
j, j_1, j_2:	job index, $1 \leq j, j_1, j_2 \leq n$
C_{kj}:	completion time of Job j at stage k
$B_{kij_1j_2}$:	a Boolean variable, 1 if Job j_2 is scheduled immediately after Job j_1 on machine i at stage k, and 0 otherwise
U_{kij}:	a Boolean variable, 1 if Job j is the first job on machine i at stage k, and 0 otherwise
P_{kj}:	processing time of Job j at stage k
P_{kij}:	processing time of Job j on machine i at stage k
ST_{kij}:	start time of Job j on machine i at stage k

For the first stage, (2) and (3) give the completion time of the first job and that of each subsequent job on the machines, respectively. Similarly for all other stages, (4) and (5) determine the completion time of the first job and that of each subsequent job on the machines, respectively. While (6) ensures non-negative start time of job processing, (7) stipulates that each of the parallel machines at a stage takes equal time to process the same job. Lastly, (8) requires the processing sequence of each stage to satisfy the processing time, and (9) guarantees that each machine can process only one job at a time.

III. THE DBA FRAMEWORK

The DBA framework consists of three modules, as shown in Fig. 1. To improve the schedule performance, an FFS firstly is decomposed into machine clusters by a clustering algorithm according to the stochastic nature of machines. Clustering is the classification of objects into different groups, such that the objects in each group would share some common trait. Quite a few algorithms, such as K-means, fuzzy C-means, and self-organization maps etc., have been proposed to perform the classification. Since the K-means clustering algorithm is simple and widely used, a neighbouring K-means clustering algorithm is proposed and adopted in this paper.

After decomposition of the FFS, machines in the same cluster share the similar stochastic nature and can be scheduled by the same approach. Clusters with less machine breakdown are solved by the predictive-reactive method, while those with more machine breakdown are scheduled by the completely reactive method. Due to their better performance, SPT and GA are identified as the completely reactive approach and the predictive-reactive approach, respectively.

In order to assign an appropriate approach to a machine cluster, it is critical to establish an effective model to estimate the makespan difference (MDSG) when generating the schedule by both SPT and GA. Artificial neural networks (ANNs) are widely used in various areas due to its capability of identifying complex nonlinear relationships between input and output. The back propagation network (BPN) [12], is a simple, effective, and commonly used ANN structure. It is therefore adopted to determine the approach assigned to the cluster.

After approach assignment above, the sub-schedule for each of the clusters is generated by either SPT or GA, and subsequently integrated into an overall schedule.

Fig. 2 illustrates a typical decomposition result of an FFS with 7 stages and 3 parallel machines for each stage. Geometric figures with the same shape represent the parallel machines. One of the two approaches, the completely reactive approach or the predictive-reactive approach, is assigned to each cluster.

IV. DETAILED ALGORITHM

A. Neighbouring K-means Clustering Algorithm

Since this study aims to group machines into clusters of similar stochastic natures, the factors related to machine breakdown need to be identified to describe the stochastic nature of machines. Machine breakdown is modelled in terms of two parameters, mean-time-to-repair (MTTR) and machine breakdown level (BL) [13]. BL denotes the percentage of time the machines have failures; it is defined as $BL = MTTR/(MTTR + MTBF)$, where MTBF represents the mean-time-between-failures. Therefore, these two factors, MTTR and BL, are adopted to form the stochastic vector U_i of machine M_i, giving

$$U_i = [\alpha \times BL\_M_i, \beta \times MTTR\_M_i] \tag{10}$$

Where M_i is the set of machines in an FFS, MTTR_ M_i and BL_ M_i are the MTTR and BL of M_i respectively, and α, β are the coefficients. U_i represents the stochastic nature of machine M_i.

As the Euclidean distance is one of the most commonly used methods to measure the distance between a pair of data, it serves to define the machine distance, which indicates the difference of stochastic nature between two machines. It is calculated as follows:

$$D(U_i, U_j) = \|U_i - U_j\|_2 = \sqrt{\alpha^2 \times (BL\_M_i - BL\_M_j)^2 + \beta^2 \times (MTTR\_M_i - MTTR\_M_j)^2} \tag{11}$$

Considering $D(U_i, U_j)$, the FFS can be decomposed into machine clusters by K-means clustering algorithm. The major problem to apply K-means clustering algorithm is the choice of cluster number. Neither a small nor a large cluster number can offer a satisfactory classification of the data objects. Recently, cluster validity indices (CVIs), indicating how well the given data set are classified, have attracted much attention to determine the optimal cluster number. Dunn [14], DB [15], Vsv [16] and DVI [17], are some typical CVIs.

However, the FFS decomposition above is different from the traditional clustering problem. Since this study aims to schedule neighbouring clusters by different approaches, a good clustering algorithm should encourage large inter-cluster distances between neighbouring clusters rather than that between non-neighbouring clusters. For this purpose, a modified DB (MDB) is proposed as follows:

$$MDB = \frac{1}{n}\sum_{i=1}^{n} \max_{i \neq j} \left(\frac{S_i + S_j}{W_{ij} \times D_{ij}} \right) \tag{12}$$

$$w_{ij} = 1/(F_i + F_j) \tag{13}$$

Where n is the number of clusters; S_i denotes the average distance of all objects from the cluster to their cluster centre; D_{ij} represents the distance between cluster centre i and j; W_{ij} is the weight of D_{ij}; F_i is the first stage of i^{th} cluster.

In order to avoid specifying the cluster number, a neighbouring K-means clustering algorithm, incorporated with MDB, is established. Its procedure is shown in Fig. 3.

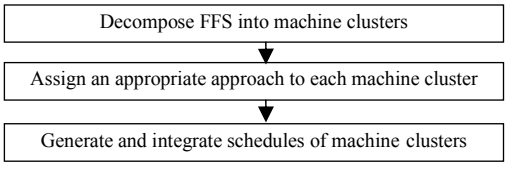

Figure 1. The DBA framework

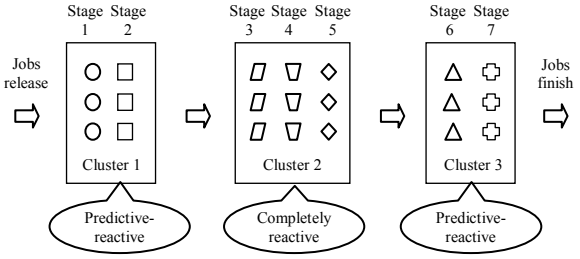

Figure 2. Machine clusters in an FFS

For k=2 to K_{max} (K_{max} = number of stages/2)
  For i=1 to I_{max} (I_{max} = 10)
    Apply the K-means clustering algorithm to decompose the FFS;
    Compute the weighted CVI for i^{th} iteration of partitioning objects into k clusters;
  End
End
Return machine clusters at k where the CVI is optimized over all i and all k.

Figure 3. Neighbouring K-means clustering algorithm

B. Back Propagation Network for Approach Assignment

Under the assumptions we made on the FFS scheduling problem, jobs are released simultaneously in the first stage. However, in subsequent stages, they are allocated by the FIFO rule and may arrive non-simultaneously. Therefore, two scenarios have to be considered when establishing models to predict the MDSG. The first scenario assumes the jobs to be released simultaneously, while the other allows the jobs arrive non-simultaneously.

Accordingly, two BPNs, each corresponds to a scenario, are generated. The configurations of these BPNs are established as follows: (1) Inputs: Five parameters, including MTTR, BL, stage size, job size and parallel machine size. These parameters significantly affect the performance of MDSG according to the experiment results in Section V; (2) Number of single hidden layers: Generally one hidden layer is capable of approximating any function with a finite number of discontinuities. Therefore, the BPN only consists of one hidden layer; (3) Number of neurons in the hidden layer: There is no concrete rule to find the optimal number. If inadequate neurons are adopted, there may be a greater risk of modelling the complex data poorly. However, if too many neurons are used, the network may fit the training data extremely well, but would perform poorly to new and unseen data. For these reasons, the optimum number with the minimum square error (MSE) is identified from the interval [2, 20] by setting a different number of neurons in the hidden layer for the same data set; (4) Output: MDSG; (5)Number of epochs per replication: 10000; (6) Number of replications: 100. The performance of a BPN is sensitive to the initial condition of network. The network with different initial conditions will be trained and evaluated respectively. Among the results, the best one is chosen.

After training, validation, and testing, the BPNs can estimate the MDSG. If the MDSG is predicted to be positive, GA is allocated to address the scheduling problem of the cluster. Otherwise, SPT is used to generate the schedule for the cluster. However, the neighbouring K-means clustering algorithm cannot avoid the possibility that two neighbouring clusters are to be suitably solved by the same approach. Therefore, it is reasonable to conduct a cluster merging process (CMP) to integrate neighbouring clusters if necessary. It consists of the following steps: (1) Identify the two neighbouring clusters which are to be addressed by the same approach; (2) Merge the two neighbouring clusters and determine the approach for the new cluster by BPN; (3) Repeat steps 1 and 2 until any two neighbouring clusters are allocated with different approaches.

Integrating with CMP, the complete process of approach assignment is summarized as follows: (1) Collect the data sets for both scenarios, including MTTR, BL, stage size, job size, parallel machine size, and the expected MDSG; (2) Train, validate and test the BPNs; (3) Estimate the MDSG for each machine cluster by BPN; (4) Assign either SPT or GA to each machine cluster according to the estimation of its MDSG; (5) Conduct CMP.

C. Cluster Scheduling

After FFS decomposition and approach assignment, schedules are generated by either SPT or GA for all clusters and then integrated into an overall one.

SPT performs better with low computation cost when the machines in the cluster have a high possibility of breakdown. It consists of the following steps: (1) Determine the job sequence based on the SPT rule for the first stage; (2) Allocate the finished job from the previous stage to the current stage by the FIFO rule until all the jobs are processed in each stage.

GA is used prior to the dispatching rules when scheduling the machine clusters with less machine breakdown. The right-shift scheduling repair is triggered to regenerate a new schedule whenever machine breakdown occurs. The overall structure of our GA is briefly described as follows: (1) Coding: The job sequence is used as the chromosome for the FFS scheduling problem. For example, job sequence [2,3,5,1,4,9,8,6,7,10] is a chromosome with ten jobs; (2) Fitness function: It is formulated as $fitness = C_{max}$, where C_{max} is the maximum completion time of jobs; (3) Selection strategy: Roulette wheel selection is applied to reproduce the next generation; (4) Crossover and mutation operation: Order preserved crossover (OPX) and shift move mutation (SM) are adopted. The crossover rate and mutation rate are analyzed by setting different values on the same FFS scheduling problem. A crossover rate of 0.8 and a mutation rate of 0.2 are found to give the best performance; (5) Termination criterion: The algorithm continues until 200 generations have been examined.

V. COMPUTATIONAL RESULTS AND ANALYSIS

The test-bed contains 27 problems with different stages, jobs and parallel machines, as shown in Table II. For each problem, ten instances with different processing times of operations are randomly generated and the simulation is iterated 50 times for each instance. Two parameters, MTTR and BL, are applied to describe the machine breakdown. Both repair times for the machines and times between failures are assumed to follow the exponential distribution with the rate λ = MTTR and λ = MTBF, respectively.

A. *Establishing BPNs for MDSG Estimation*

Corresponding to the two scenarios of simultaneous and non-simultaneous job arrivals, the examples for training and testing BPNs are generated for all the problems in the test-bed. The levels of BPN input used in the experiments are shown in Table I.

TABLE I. BPN INPUTS AND THEIR LEVELS

Factors	Levels
Breakdown level	10 levels (0.01, 0.02, …, 0.1)
Mean-time-to-repair	1P[a], 1.3P, 1.6P, 1.9P, 2.2P, 2.5P
Stage size	10 levels (1, 2, …, 10)
Job size	20, 25, 30, 35, 40, 45
Parallel machine size	2, 3, 4, 5, 6, 7

a. P is the average processing time of an operation.

Based on the data of the examples, scatter plots are generated to visualize the relationship of five factors, including BL, MTTR, stage size, job size and parallel machine size, on the MDSG. As shown in Fig. 4, circles and squares represent the results derived by the scenarios of simultaneous job arrivals and non-simultaneous job arrivals, respectively. Accordingly the following conclusions are drawn: (1) The MDSG decreases with the increasing of BL, MTTR, stage size and job size. Parallel machine size affects the MDSG as well. Therefore, it is reasonable to adopt these five factors as BPN inputs; (2) The MDSG is different for the two scenarios of simultaneous and non-simultaneous job arrivals. It implies that two BPNs are rational to estimate the MDSG as well.

The prediction accuracy of the two BPNs is measured by MSE. The minimal MSE with various numbers of neurons in the hidden layer are compared in Fig. 5. The optimum numbers of BPNs with simultaneous and non-simultaneous job arrivals are 8 and 12, respectively. The BPNs with the optimum numbers are adopted to estimate the MDSG.

B. *DBA Analysis*

To evaluate the effectiveness of the proposed approach, SPT, GA, and DBA are analyzed in a stochastic environment in which BL is uniformly distributed in the interval [0.01, 0.1] and MTTR are randomly selected either 1P or 2.5P. The experiment results of these three approaches with machine breakdown (denoted by SPT_BR, GA_BR, and DBA_BR respectively) are shown in Table II. The results of SPT and GA without machine breakdown (denoted by SPT, and GA respectively) are also given. It is clear that DBA_BR gives the best performance in most cases, decreasing the makespan by about 4% and 12% in comparison with SPT_BR and GA_BR, respectively.

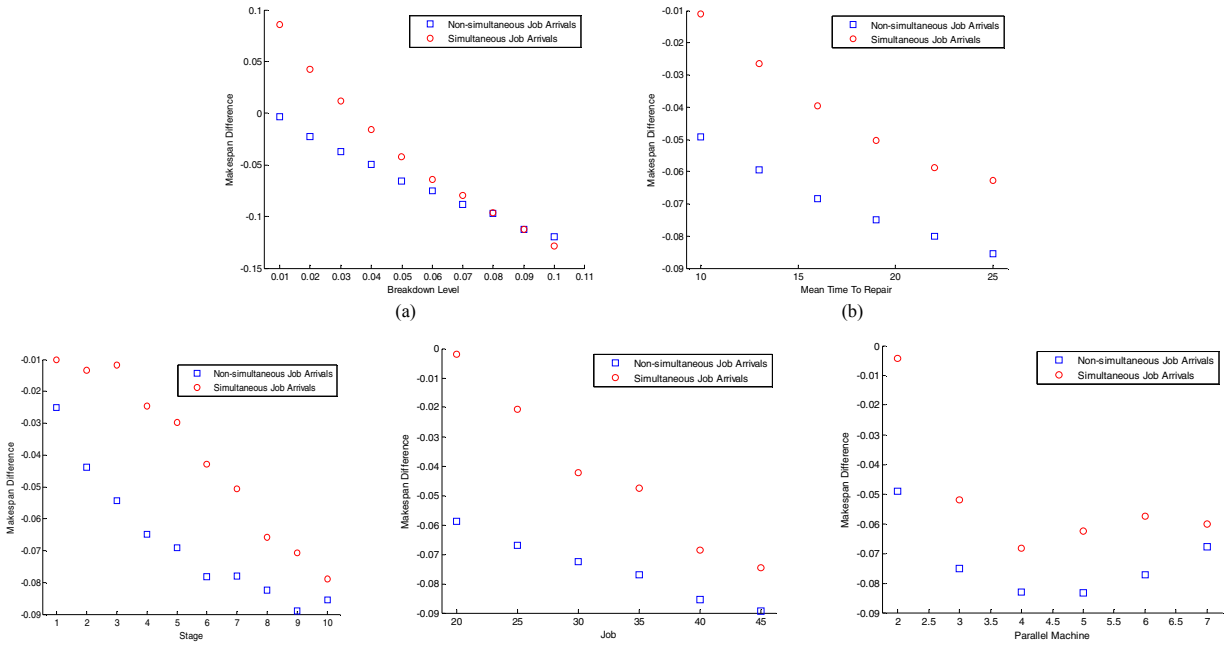

Figure 4. Scatter plots of makespan with (a) breakdown level, (b) mean-time-to-repair, (c) stage, (d) job, (e) parallel machine.

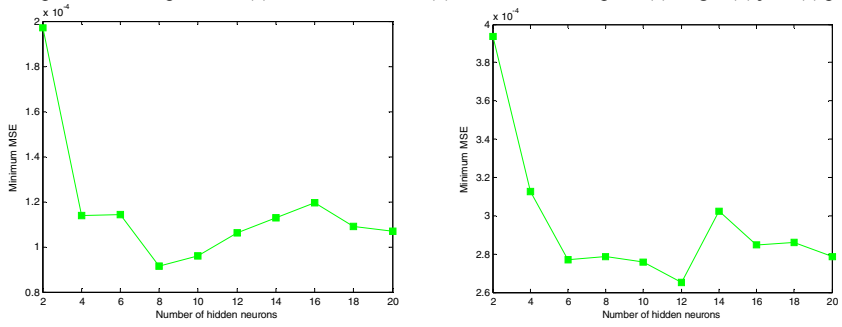

Figure 5. The minimal MSEs with various numbers of hidden neurons for (a) simultaneous job arrivals, (b) non-simultaneous job arrivals.

TABLE II. COMPARION OF MAKESPANS OF VARIOUS SCHEDULING ALGORITHMS

Problem size (No. of jobs x no. of stages)	No. of Parallel Machines in each stage	SPT	GA	SPT_BR	GA_BR	DBA_BR
20×6	2	1.110[a]	1.000	1.208	1.232	1.146
20×6	3	1.148	1.000	1.268	1.292	1.217
20×6	4	1.150	1.000	1.290	1.375	1.293
20×10	2	1.143	1.000	1.250	1.265	1.190
20×10	3	1.108	1.000	1.237	1.335	1.211
20×10	4	1.170	1.000	1.208	1.387	1.286
20×15	2	1.126	1.000	1.268	1.251	1.178
20×15	3	1.112	1.000	1.290	1.315	1.187
20×15	4	1.090	1.000	1.250	1.368	1.216
30×6	2	1.155	1.000	1.237	1.219	1.174
30×6	3	1.162	1.000	1.208	1.283	1.171
30×6	4	1.139	1.000	1.268	1.309	1.189
30×10	2	1.150	1.000	1.290	1.322	1.236
30×10	3	1.135	1.000	1.250	1.270	1.164
30×10	4	1.129	1.000	1.237	1.412	1.226
30×15	2	1.125	1.000	1.208	1.327	1.211
30×15	3	1.101	1.000	1.268	1.363	1.197
30×15	4	1.104	1.000	1.290	1.420	1.254
40×6	2	1.130	1.000	1.250	1.237	1.213
40×6	3	1.133	1.000	1.237	1.318	1.205
40×6	4	1.130	1.000	1.208	1.359	1.218
40×10	2	1.155	1.000	1.268	1.278	1.235
40×10	3	1.152	1.000	1.290	1.370	1.197
40×10	4	1.132	1.000	1.250	1.458	1.250
40×15	2	1.149	1.000	1.237	1.328	1.241
40×15	3	1.109	1.000	1.216	1.362	1.175
40×15	4	1.112	1.000	1.238	1.424	1.200
Average		**1.132**	**1.000**	**1.249**	**1.329**	**1.210**

a. Average makespan ratio of a scheduling algorithm over GA

VI. CONCLUSION

This paper proposed and studied the DBA to minimize the makespan for scheduling an FFS with machine breakdown. In the proposed algorithm, machines are grouped into several clusters by a neighbouring K-means clustering algorithm, and each cluster is scheduled by either SPT or GA. The effectiveness of DBA was measured with experiment results, from which we can draw the following conclusions: (1) breakdown level, mean-time-to- repair, stages size, job size, and parallel machine size all influence the MDSG significantly; (2) DBA is superior to SPT and GA for solving the FFS scheduling problem with machine breakdown. Its better performance results from the decomposition strategy – to schedule with GA in a less stochastic environment and with SPT in a more stochastic environment.

The proposed DBA provides a promising way to address the FFS scheduling problem with machine breakdown. Further research can focus on the development of new approaches based on DBA to deal with other types of uncertainties, such as non-deterministic processing times.

REFERENCES

[1] R. Linn and W. Zhang, "Hybrid flow shop scheduling: a survey," Computers and Industrial Engineering, vol. 37, pp. 57-61, 1999.

[2] M. R. Garey, Computers and intractability: a guide to the theory of NP-completeness. New York: W. H. Freeman, 1979.

[3] J. N. D. Gupta, "Two-stage, hybrid flowshop scheduling problem," Journal of the Operational Research Society, vol. 39, pp. 359-364, 1988.

[4] H. Aytug, M. A. Lawley, K. McKay, S. Mohan, and R. Uzsoy, "Executing production schedules in the face of uncertainties: A review and some future directions," European Journal of Operational Research, vol. 161, pp. 86-110, 2005.

[5] T. Vidal, "The many ways of facing temporal uncertainty in planning and scheduling," Tatihou, France, 2004, pp. 9-12.

[6] J. L. Hunsucker and J. R. Shah, "Comparative performance analysis of priority rules in a constrained flow shop with multiple processors environment," *European Journal of Operational Research,* vol. 72, pp. 102-114,1994.

[7] L. Liu, H.-y. Gu, and Y.-g. Xi, "Robust and stable scheduling of a single machine with random machine breakdowns," The International Journal of Advanced Manufacturing Technology, vol. 31, pp. 645-654, 2007.

[8] P. Kouvelis, R. L. Daniels, and G. Vairaktarakis, "Robust scheduling of a two-machine flow shop with uncertain processing times," IIE Transactions, vol. 32, pp. 421-432, 2000.

[9] G. E. Vieira, J. W. Herrmann, and E. Lin, "Rescheduling manufacturing systems: A framework of strategies, policies, and methods," *Journal of Scheduling,* vol. 6, pp. 39-62, 2003.

[10] S. R. Lawrence and E. C. Sewell, "Heuristic, optimal, static, and dynamic schedules when processing times are uncertain," *Journal of Operations Management,* vol. 15, pp. 71-82, 1997.

[11] H. Matsuura, H. Tsubone, and M. Kanezashi, "Sequencing, dispatching, and switching in a dynamic manufacturing environment," *International Journal of Production Research,* vol. 31, pp. 1671-1688, 1993.

[12] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature,* vol. 323, pp. 533-536, 1986.

[13] O. Holthaus, "Scheduling in job shops with machine breakdowns: an experimental study," *Computers & Industrial Engineering,* vol. 36, pp. 137-162, 1999.

[14] J. C. Dunn, "Fuzzy relative of iosdata process and its use in detecting compact well-separated clusters," *J Cybern,* vol. 3, pp. 32-57, 1973.

[15] D. L. Davies and D. W. Bouldin, "A Cluster Separation Measure," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. PAMI-1, pp. 224-227, 1979.

[16] D. J. Kim, Y. W. Park, and D. J. Park, "A novel validity index for determination of the optimal number of clusters," *IEICE Transactions on Information and Systems,* vol. E84-D, pp. 281-285, 2001.

[17] J. Shen, S. I. Chang, E. S. Lee, Y. Deng, and S. J. Brown, "Determination of cluster number in clustering microarray data," *Applied Mathematics and Computation,* vol. 169, pp. 1172-1185, 2005.