A Decomposition-Based Approach to Flexible Flow Shop Scheduling under Stochastic Setup Times

K. Wang and S.H. Choi

Department of Industrial and Manufacturing Systems Engineering
The University of Hong Kong
Hong Kong, China

Abstract—**Research on production scheduling under uncertainty has recently received much attention. This paper presents a novel decomposition-based approach (DBA) to flexible flow shop (FFS) scheduling under stochastic setup times. In comparison with traditional methods using a single approach, the proposed DBA combines and takes advantage of two different approaches, namely the Genetic Algorithm (GA) and the Shortest Processing Time Algorithm (SPT), to deal with uncertainty. A neighbouring K-means clustering algorithm is developed to firstly decompose an FFS into an appropriate number of machine clusters. A back propagation network (BPN) is then adopted to assign either GA or SPT to generate a sub-schedule for each machine cluster. Finally, an overall schedule is generated by integrating the sub-schedules of the machine clusters. Computation results reveal that the DBA is superior to SPT and GA alone for FFS scheduling under stochastic setup times.**

Keywords-back propagation network; decomposition; flexible flow shop; neighbouring K-means clustering algorithm; stochastic setup times.

I. INTRODUCTION

A flexible flow shop (FFS) [1] consists of a series of production stages, each of which has several functionally identical machines operating in parallel. All the jobs released to an FFS have to visit all the stages in the same order. The FFS scheduling problem has attracted considerable attention during the past decades; research efforts generally consider a static environment with no unexpected events during the job processing. Real manufacturing, however, is dynamic and tends to suffer a wide range of uncertainties, such as stochastic processing times, machine breakdown, and rush orders, etc. The FFS scheduling problem is NP-hard in nature [2], and consideration of uncertainties aggravates its complexity.

This paper is primarily concerned with the scheduling problem of FFS with stochastic setup times. The setup in this study is non-anticipatory and sequence-independent. A non-anticipatory setup can only be started when the corresponding job becomes available on the machine. For a sequence-independent setup, the setup time only depends on the job to be processed [3].

As a research issue, scheduling under uncertainty has recently drawn considerable attention. The completely reactive approach, the robust approach, and the predictive-reactive approach are three fundamental ways [4] 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 can find a reasonably good solution relatively quickly. However, it uses only local information to generate a schedule, which may not be globally optimal in nature [5].

The robust scheduling approach takes into account possible uncertainties to construct solutions. Uncertainties, known as a priori, can be modelled by some random variables [6]. If such uncertainties are difficult to quantify, a range of scenarios will be considered and a solution is developed to optimise the performance under different scenarios [7]. In this case, the approach is viewed as a form of under-capacity scheduling to maintain robustness under different scenarios.

The predictive-reactive approach is a two-step process. First, a predictive schedule is generated over the time horizon in question. 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 [8]. The right-shift schedule repair postpones the remaining operations by the amount of time needed to make the schedule feasible. 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 [6]. 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 in this regard.

Most of research work on scheduling under uncertainty employs only one of these three approaches. In comparison, there are some early studies on comparing the effectiveness of different approaches or integrating them to deal with uncertainty. Lawrence and Sewell [9] studied the job shop 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 under little uncertainty. However, the predictive methods might lead to

poor result when the uncertainty exceeded a certain level. In order to handle a complex environment, Matsuura et al. [10] developed a predictive approach on a periodic basis. 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. A search of available literature indicates not much research works have been attempted to address the combination of different approaches.

This paper studies the problem of FFS scheduling under the uncertainty of stochastic setup times, with the objective to minimise the makespan. Enlightened by the work of Lawrence's [9], 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 machine clusters based on their stochastic nature during job setup. Then the completely reactive approach or the predictive-reactive approach, determined by the process of approach assignment, is used to generate a sub-schedule for each machine cluster. Finally these sub-schedules are integrated into an overall solution. The proposed DBA explores a new direction in the field of scheduling under uncertainty. Instead of using a single approach, the DBA attempts to combine and take advantage of the completely reactive approach with the predictive-reactive approach to deal with the uncertainty.

The remaining part of this chapter will first describe the problem. Then, the framework and the details of the DBA will be explained. Subsequently, computation results will be analysed to evaluate the effectiveness of the DBA. Finally, conclusion and future work will be discussed.

II. PROBLEM DESCRIPTION

In the FFS discussed above, machines sharing a similar characteristic are arranged into stages in series. Jobs have to pass all the stages in the same order. In each stage, there are a number of functionally identical machines in parallel, and a job is to be processed on one of these machines. In this paper, a job needs to be set up before processing and setup times are not negligible.

Setup includes work to prepare the machine, process, or bench for product parts or the cycle [11]. Setup times are uncertain due to the variation in skill levels of setup crews, temporary shortage of equipment, tools and setup crews, and unexpected breakdowns of fixtures and tools [12]. The actual setup time can be described by the expected setup time E[S] and the standard deviation σ. The coefficient of setup time variation (CSTV), defined as $CSTV = \sigma / E(S)$, can be used as an indicator to setup time uncertainty; it equals 0 when setup times are deterministic, and increases as the uncertainty increases.

In order to simplify the typical FFS scheduling problem with stochastic setup times, the following assumptions are made: (1) Preemption is not allowed for job processing; (2) All jobs are released at the same time for the first stage; (3) All machines are available when jobs are released to the FFS. Each machine can process at most one operation at a time; (4) There is no travel time between machines; (5) Infinite buffers exist for machines; (6) Job setups are non-anticipatory and sequence-independent; (7) The actual setup time of a job on a machine is uncertain; (8) For the same job, the expected setup time at any parallel machine at a stage is identical; (9) Job setups on machines at a stage share the same CSTV, but the CSTV may be different for the job setups at other stages.

The scheduling objective 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 minimised without violating any of the assumptions above. This FFS scheduling problem can also be described as follows.

$$\min\{\max[C_{tj}]\} \qquad (1)$$

Subject to the following constraints:

$$C_{1j} = S_{1j} + P_{1j}, \quad if \quad \sum_{i=1}^{m_1} U_{1ij} > 0 \qquad (2)$$

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

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

$$C_{kj_2} = \max\{ \sum_{i=1}^{m_k} \sum_{j_2=1}^{n} \left(B_{kij_1j_2} \times C_{kj_1} \right), \ C_{(k-1)j_2} \} + S_{kj_2} + P_{kj_2}, \qquad (5)$$

$$if \quad k > 1 \ \& \ \sum_{i=1}^{m_k} U_{kij_2} = 0$$

$$E\left(S_{ki_1j} \right) = E\left(S_{ki_2j} \right) = E\left(S_{kj} \right), \quad (i_1, i_2) \in M_k \qquad (6)$$

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

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

$$[(ST_{kij_1} - ST_{kij_2}) \geq S_{kij_2} + P_{kij_2}] \ or \ [(ST_{kij_2} - ST_{kij_1}) \geq S_{kij_1} + P_{kij_1}] \qquad (9)$$

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
$E(S_{kj})$:	expected setup time of Job j at stage k
$E(S_{kij})$:	expected setup time of Job j on machine i at stage k
S_{kj}:	stochastic setup time of Job j at stage k
S_{kij}:	stochastic setup time of Job j on machine i at stage k
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. (6) and (7) stipulates that the expected setup time and processing time of a job is equal on any parallel machines at a stage. 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 FRAMEWORK OF THE PROPOSED DECOMPOSITION-BASED APPROACH (DBA)

The DBA framework consists of three modules, including FFS decomposition, approach assignment, and sub-schedule generation and integration.

A. FFS Decomposition

An FFS is firstly decomposed by a clustering algorithm into machine clusters, each of which contains a number of machines sharing a similar stochastic nature. The stochastic nature of a machine results from the uncertainties which occur during job setup and processing. The high stochastic nature of a machine usually leads to a large difference between the actual and the planned schedule. As the actual setup times of jobs on a machine may be non-deterministic, the setup time uncertainty is used to describe the stochastic nature of a machine.

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 widely used, a neighbouring K-means clustering algorithm is proposed to decompose an FFS in this paper.

B. Approach Assignment

After FFS decomposition, machines in the same machine cluster share the similar stochastic nature and can be scheduled by the same approach. Machine clusters with low stochastic natures are solved by the predictive-reactive approach, while those with high stochastic natures are scheduled by the completely reactive approach. Due to their better performance, the Genetic Algorithm (GA) and the Shortest Processing Time (SPT) algorithm are identified as the predictive-reactive approach and the completely 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. The back propagation network (BPN) has been successfully applied for system modelling, prediction, and classification due to its capability of identifying complex nonlinear relationships between input and output [13]. It is therefore adopted to estimate the MDSG for each machine cluster, and the positive or negative sign of the MDSG determines the approach to be assigned to the machine cluster.

C. Sub-schedule Generation and Integration

After approach assignment above, the sub-schedule for each machine cluster is generated by either GA or SPT, and subsequently integrated into an overall schedule. Fig. 1 shows the decomposition result of an FFS with 7 stages and 3 parallel machines at each stage. Geometric figures with the same shape represent the parallel machines. One of the two approaches, GA or SPT, is assigned to each machine cluster.
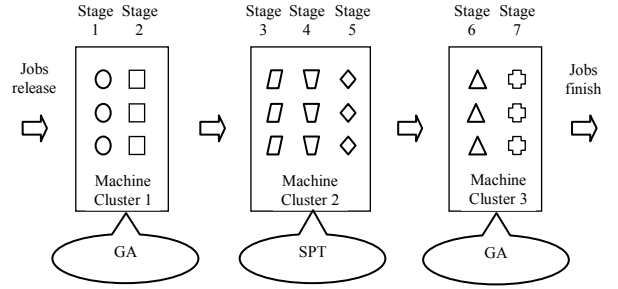

Figure 1. Machine clusters in an FFS

IV. DETAILED ALGORITHM

A. Neighbouring K-means Clustering Algorithm

For the purpose of decomposing an FFS, the machines of an FFS are grouped into a few machine clusters in which machines share a similar stochastic nature. Since the CSTV represents setup time uncertainty, it is adopted to form the stochastic vector U_i to group machines into machine clusters of similar stochastic natures, giving

$$U_i = [CSTV_i] \qquad (10)$$

Where $CSTV_i$ is the CSTV of the parallel machines at stage i. U_i represents the stochastic nature of machines at stage i. A machine with a large CSTV indicates a high stochastic nature of setup time uncertainty.

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 $D(U_i, U_j)$, which represents not the physical distance but the difference of stochastic nature between the parallel machines at stages i and j. The machine distance is calculated as follows.

$$D(U_i, U_j) = \|U_i - U_j\|_2 = \sqrt{(CSTV_i - CSTV_j)^2} \qquad (11)$$

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 machine cluster number. Neither a small nor a large machine cluster number can offer a satisfactory classification.

Recently, cluster validity indices (CVIs) have attracted much attention as an approach to determining the optimal cluster number. Most CVIs are defined by combining the intra-cluster distances and inter-cluster distances [14]. The former one measures the distances of objects within a cluster to represent its compactness, while the latter one computes the distance between two different clusters and is an indicator of cluster separability. A good clustering algorithm should have small intra-cluster distances and large inter-cluster distances. Dunn, DB, Vsv and SD are some typical CVIs [14].

However, the FFS decomposition above is different from the traditional clustering problem. Since this study aims to schedule neighbouring machine clusters by different approaches, a good clustering algorithm should encourage

large inter-cluster distances between neighbouring machine clusters rather than that between non-neighbouring machine clusters. For this purpose, a modified DB (MDB), giving the weight to the inter-cluster distance, is proposed as Table I.

TABLE I. DB AND MDB

DB	MDB
$MDB = \dfrac{1}{n}\sum_{i=1}^{n}\max_{i\neq j}\left(\dfrac{S_i + S_j}{D_{ij}}\right)$	$MDB = \dfrac{1}{n}\sum_{i=1}^{n}\max_{i\neq j}\left(\dfrac{S_i + S_j}{W_{ij}\times D_{ij}}\right),$ $w_{ij} = 1/\left(F_i + F_j\right)$

Where n is the number of machine clusters; S_i denotes the intra-cluster distance, which measures the average machine distance of all objects from the machine cluster i to their cluster centre; D_{ij} represents the inter-cluster distance, which measures the machine distance between machine cluster centre i and j; W_{ij} is the weight of D_{ij}; F_i is the first stage of ith machine cluster.

In order to avoid specifying the machine cluster number, a neighbouring K-means clustering algorithm, incorporated with MDB, is established. Its procedure is shown in Fig. 2.

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 an FFS into k machine clusters;
    Compute the MDB for i^{th} iteration of decomposing an FFS into k machine clusters;
  End
End
Return machine clusters where the MDB is minimal over all i and all k.

Figure 2.   The proposed neighbouring K-means clustering algorithm

B. Back Propagation Network for Approach Assignment

After FFS decomposition, the machine clusters can be scheduled by either SPT or GA. The assigned approach for a machine cluster can be determined by the makespan difference of the schedules generated by SPT and GA (MDSG), giving

$$MDSG = \left(M_{SPT\_S} - M_{GA\_S}\right)/M_{GA} \qquad (14)$$

Where M_{SPT_S} and M_{GA_S} are the makespans generated respectively by SPT and GA with stochastic setup times, while M_{GA} is the makespan generated by GA with deterministic setup times. For a machine cluster, if the MDSG is predicted to be positive, GA is allocated to address the scheduling problem of the machine cluster. Otherwise, SPT is used to generate the schedule for the machine cluster.

The back propagation network (BPN) is adopted to estimate the MDSG for each machine cluster in this study. 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 BPNs. 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 details of BPN establishment for each scenario are as follows: (1) Inputs: Four parameters, namely CSTV, stage size, job size, and parallel machine size. These parameters are found to affect the performance of MDSG significantly according to the experiment results in Section V; (2) Number of 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 hidden neurons: As there is no concrete rule to find the optimal number, the number of hidden neurons is intentionally selected from the interval {2, 20}. For each scenario, the BPNs with different number of hidden neurons are generated and evaluated by the mean square error (MSE), and the one that corresponds to the number of hidden neurons that has the least minimal MSE is termed the optimal BPN; (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. Therefore, for a specific number of hidden neurons, 100 BPNs with different initial conditions will be trained and evaluated respectively. Among these BPNs, only the one with minimal MSE is kept for the purpose to further identify the optimal BPN. (7) Training examples: The generation of training examples is described in Section V.

C. Machine Cluster Scheduling

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

SPT performs better when the machines in a machine cluster with a high stochastic nature. It consists of the following two main 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 at each stage.

GA is used prior to the dispatching rules when scheduling a machine cluster with a low stochastic nature. The right-shift scheduling repair is triggered to regenerate a schedule whenever job processing has to be postponed due to the stochastic setup times. 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 in an FFS; (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. A crossover rate of 0.8 and a mutation rate of 0.2 are found to give good performance; (5) Termination criterion: The algorithm continues until 200 generations have been examined.

V. COMPUTATION RESULTS AND ANALYSIS

A. Experiment Design

Two experiments are designed and conducted to evaluate the proposed DBA. The first experiment aims at establishing
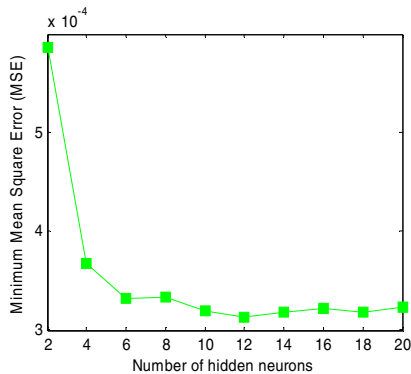
the BPNs for the MDSG estimation. The second one focuses on analysing the performance of DBA on a test-bed shown in Table III.

In the two experiments, the processing times of operations are generated from the uniform distribution U (1, 20), while two levels of expected setup times are obtained from the uniform distribution U (1, 20η), where η is setup time severity and set at 1.0 and 2.0. The actual setup times are uncertain and follow the gamma distribution with the expected setup time E(S) and standard deviation $\sigma = E(S) \times CSTV_j$, where CSTV$_j$ is the coefficient of setup time variation at stage j.
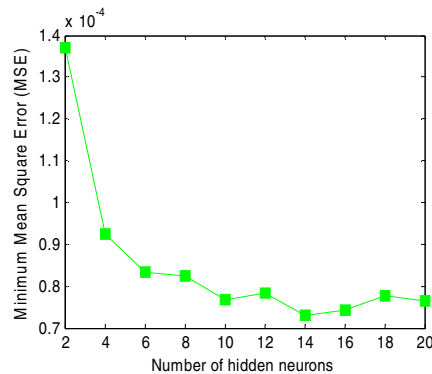
B. Experiment I: Generation of BPNs for MDSG Estimation

For a specific η, two sets of training examples, corresponding to the scenario of simultaneous and non-simultaneous job arrivals, need to be generated, respectively. The levels of BPN input used are shown in Table II. For each scenario, by exploring all possible combinations (10×10×6×6 = 3,600) of BPN inputs, the experimental FFS scheduling problems to minimise makespans with stochastic setup times are firstly generated, in each of which all the parallel machines share the same CSTV. Subsequently, these problems are solved by GA and SPT, respectively. Lastly, the MDSG, which is the output of BPN, can be obtained by (14) for each problem. Thus, this procedure results in a total of 3,600 training examples for each scenario.

In order to identify the optimal BPNs, BPNs with different number of hidden neurons are established based on training examples and their prediction accuracy is measured by MSE.
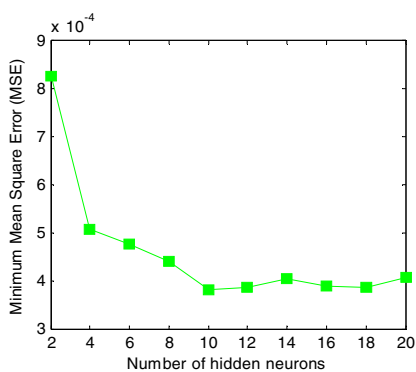
Fig. 3 and Fig. 4 show the relationship of the minimal MSE with various numbers of hidden neurons when η = 1.0 and 2.0, respectively. The optimal BPNs for simultaneous and non-simultaneous job arrivals have 12 and 14 hidden neurons when η =1.0, and have 10 and 14 hidden neurons when η = 2.0. Accordingly, such four optimal BPNs are used to estimate the MDSGs.

TABLE II. BPN INPUTS AND THEIR LEVELS

Factors	Levels
CSTV	10 levels {0.1, 0.2, …, 1}
Stage size	10 levels {1, 2, …, 10}
Job size	20, 25, 30, 35, 40, 45
Parallel machine size	2, 3, 4, 5, 6, 7

C. Experiment II: DBA Analysis

In order to evaluate the effectiveness of the proposed algorithm, SPT, GA, and DBA are analyzed in a stochastic environment in which CSTV is uniformly distributed in the interval {0.1, 1}. The experiment results of these three algorithms with stochastic setup times (denoted by SPT_S, GA_S, and DBA_S respectively) are shown in Table III.

All the results are the ratios of the average makespan of various scheduling algorithms to that of SPT_S. From the experiment results, it can be seen that DBA_S gives the best performance. It is clear that the average makespan of DBA_S, in comparison with that of SPT_S and GA_S, decreases by about 3% and 6% when η = 1.0, and decreases by about 3% and 8% when η = 2.0.
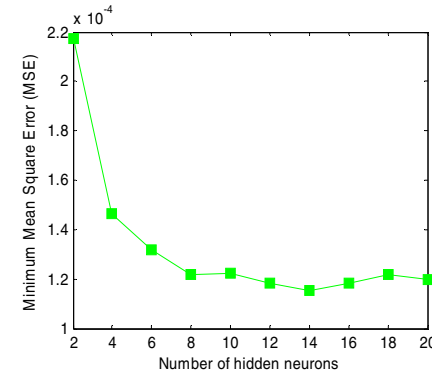

(a) simultaneous job arrivals   (b) non-simultaneous job arrivals

Figure 3. The minimal MSEs with various numbers of hidden neurons (η = 1.0)

(a) simultaneous job arrivals   (b) non-simultaneous job arrivals

Figure 4. The minimal MSEs with various numbers of hidden neurons (η = 2.0)

TABLE III.   COMPARION OF MAKESPANS OF VARIOUS SCHEDULING ALGORITHMS

Problem Size (No. of jobs x no. of stages)	No. of Parallel Machines in each stage	Setup Time Severity (η)					
		η = 1.0			η = 2.0		
		SPT_S	GA_S	DBA_S	SPT_S	GA_S	DBA_S
20×6	2	1.000	0.997	0.950	1.000	1.020	0.939
20×6	3	1.000	1.019	0.960	1.000	1.030	0.961
20×6	4	1.000	1.024	0.979	1.000	1.040	0.965
20×10	2	1.000	0.992	0.954	1.000	1.049	0.969
20×10	3	1.000	1.014	0.967	1.000	1.048	0.961
20×10	4	1.000	1.034	0.968	1.000	1.042	0.972
20×15	2	1.000	1.020	0.972	1.000	1.045	0.972
20×15	3	1.000	1.051	0.975	1.000	1.059	0.977
20×15	4	1.000	1.040	0.984	1.000	1.103	0.997
30×6	2	1.000	0.999	0.949	1.000	1.038	0.957
30×6	3	1.000	1.029	0.967	1.000	1.035	0.968
30×6	4	1.000	1.037	0.979	1.000	1.077	0.982
30×10	2	1.000	1.009	0.947	1.000	1.030	0.967
30×10	3	1.000	1.016	0.963	1.000	1.067	0.977
30×10	4	1.000	1.044	0.965	1.000	1.070	0.966
30×15	2	1.000	1.011	0.951	1.000	1.037	0.960
30×15	3	1.000	1.048	0.975	1.000	1.063	0.974
30×15	4	1.000	1.061	0.980	1.000	1.107	0.996
40×6	2	1.000	1.006	0.973	1.000	1.015	0.944
40×6	3	1.000	1.018	0.954	1.000	1.029	0.974
40×6	4	1.000	1.024	0.963	1.000	1.083	0.979
40×10	2	1.000	0.980	0.942	1.000	1.027	0.960
40×10	3	1.000	1.061	0.979	1.000	1.067	0.972
40×10	4	1.000	1.049	0.978	1.000	1.089	0.981
40×15	2	1.000	1.004	0.954	1.000	1.042	0.965
40×15	3	1.000	1.058	0.982	1.000	1.090	0.968
40×15	4	1.000	1.067	0.981	1.000	1.107	0.985
Average		1.000	1.026	0.966	1.000	1.056	0.970

VI.   CONCLUSION

This paper proposed a decomposition-based approach (DBA) to minimise the makespan of an FFS scheduling problem with stochastic setup times. In this approach, machines are grouped into several machine clusters by a neighbouring K-means clustering algorithm without predefining the number of clusters, and each machine cluster is scheduled by either SPT or GA. The effectiveness of DBA was validated with experiment results. It was found that the DBA gives promising and better results as compared to SPT and GA alone. The better performance of DBA results from the decomposition strategy – to schedule with GA in a low stochastic environment and with SPT in a high stochastic environment.

The proposed DBA provides a promising way to address FFS scheduling under stochastic setup times. Further research effort can be devoted to extending the DBA to solve job shop scheduling problems with stochastic setup times, which is essentially more complex than FFS.

REFERENCES

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

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

[3] A. Allahverdi, C. T. Ng, T. C. E. Cheng, and M. Y. Kovalyov, "A survey of scheduling problems with setup times or costs," European Journal of Operational Research, 187(3), pp. 985-1032, 2008.

[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] L. Tang, W. Liu, and J. Liu, "A neural network model and algorithm for the hybrid flow shop scheduling problem in a dynamic environment," Journal of Intelligent Manufacturing, 16 (3), pp. 361-370, 2005.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] A.Allahverdi, J. N. D. Gupta, and T. Aldowaisan, "A review of scheduling research involving setup considerations," Omega, 27(2), pp. 219-239, 1999.

[12] S. C. Kim and P. M. Bobrowski, "Scheduling jobs with uncertain setup times and sequence dependency," Omega, 25(4), pp. 437-447, 1997.

[13] D. Y. Lin and S. L. Hwang, "Use of neural networks to achieve dynamic task allocation: a flexible manufacturing system example," International Journal of Industrial Ergonomics, 24(3), pp. 281-298, 1999.

[14] Kim, M. and Ramakrishna, R. S. "New indices for cluster validity assessment," Pattern Recognition Letters, 26(15), pp. 2353-2363, 2005.