A Multi-objective Model and Iterated Greedy for Robust Permutation Flow Shop Scheduling in the Presence of Disruptions

Mohanad Al-Behadili
University of Basrah

Djamila Ouelhadj
University of Portsmouth

A Multi-objective Model and Iterated Greedy for Robust Permutation Flow Shop Scheduling in the Presence of Disruptions

Mohanad AL-Behadili[1,2], Djamila Ouelhadj[2], Dylan Jones[2]

[1] Department of Mathematics, College of Science, University of Basra, Basra, Iraq
mohanad.saad@uobasrah.edu.iq

[2] Department of Mathematics, University of Portsmouth,
Lion Gate Building, Portsmouth PO1 3HF, UK
djamila.ouelhadj@port.ac.uk, dylan.jones@port.ac.uk

Abstract

This study presents a multi-objective optimisation model and a predictive-reactive approach that uses an Iterated Greedy algorithm for robust dynamic permutation flow shop scheduling under different types of real-time events including machine breakdown and arrival of new jobs. The multi-objective model considers three important performances, namely; utility, stability and robustness. The results illustrate that the performance of the new model and the solution method outperform both of the bi-objective model [1] and the classical makespan model.

1 Introduction

The flow shop scheduling problem is defined as a production problem where a set of n jobs have to be executed with identical flow patterns on a set of m machines. When the order of jobs processing on all machines is the same, we have the permutation flow shop scheduling problem (PFSP). This kind of problem is NP-hard [2]. In the PFSP, the most common criterion that has been studied in the literature is the minimisation of the total completion time or makespan. PFSP in the presence of disturbances has been intensively studied in the literature [3–5]. A comprehensive survey on this problem is presented in [6]. There are two rescheduling strategies to handle the effect of disruptions, including; (1) make the solution feasible again and (2) improve the efficiency of the schedule so as to reduce the deviation between the initial schedule and the current one. [1, 7, 8] introduce bi-objective models that aim to minimise the utility and stability performances. Rahmani and Heydari [9] introduce a multi-objective optimisation model with three measures namely; utility, stability and robustness for a flow shop problem under uncertain processing times and new jobs arrival. This model has only considered the case of $n \times 2$ ($jobs \times machines$) and the authors apply an exact approach to small size instances. The majority of rescheduling approaches in the literature focused on a single type of independent disruption only, such as new job arrival and machine breakdown [10–15]. However, Katragjini et al. [1] simulate a novel instances for the PFSP considering three different types of events simultaneously. They use a predictive-reactive procedure with different heuristics and an Iterated Greedy algorithm (IG) for the PFSP under three different real-time events, which are; machine breakdown, arrival of new jobs and job ready time variations. According to [1], the IG algorithm shows the best performance comparing to other heuristic algorithms. Li and Mao [16] propose a discrete teaching-learning-based optimisation to solve the flow shop rescheduling problem under five types of uncertainties including machine breakdown, arrival of new jobs, cancellation of jobs, job processing variation and job release variation. The authors use the bi-objective model that was proposed by [1] to minimise the makespan and instability measures. There are few successful techniques that have been proposed in the literature to solve the dynamic PFSP due to the NP-hard nature of the problem. These methods are classified into exact, heuristics, meta-heuristics, and hybrid methods. Ruiz and Stützle [17] proposed an IG algorithm, which has been successfully used for many combinatorial optimisation problems and subsequently for the PFSP. Quan-Ke et al. [18] apply IG algorithm for the hybrid flow shop scheduling problem with the objective of minimising the bi-objective function of both of the weighted earliness and tardiness from the due window. They also compare the IG algorithm for this problem against nine other competing procedures where the IG algorithm shows the best performance. The IG algorithm has powerful advantages of fast convergence and simplicity and it consists of two phases; destruction and construction. The construction procedure is based on the NEH heuristic insertion [19] with the objective of minimising the total completion time. The contribution of this work is as follows: (a) introducing a multi-objective optimisation model for the problem of dynamic PFSP in the presence of different uncertainties. (b) Implement a predictive-reactive technique and the IG method to solve this problem. (c) The presented multi-objective model is compared against both of the bi-objective model [1] and the single makespan model. (d) Finally, the multi-objective model is normlised to ensure fair comparison and a practical weight sensitivity algorithm [20] is applied to evaluate the efficiency of the presented multi-objective model and solution approaches. The remainder of this paper is organised as follows: In Section 2, the multi-objective model is described. Section 3 provides the methodology of the proposed predictive-reactive approach and IG algorithm. In Section 4, the computational results and comparisons are provided with a statistical study. Finally, conclusions are drawn together with future research directions in Section 5.

2 The multi-objective optimisation model for robust dynamic PFSP

In this paper, we present a multi-objective optimisation model based on the models proposed by [21] and [9]. This model considers three important objectives; utility, stability and robustness. The utility is a classical makespan objective function. The instability performance calculates the deviation of completion time between the initial (baseline) and the new schedule. Finally, the robustness objective measures the difference between the total completion time of the baseline and current schedule. The multi-objective optimisation model is given by the following equation:

$$MinMO = \alpha U_n(S^*) + \beta I_n(S^*) + \gamma R_n(S^*) \tag{1}$$

Where $0 \leq \alpha, \beta, \gamma \leq 1$ and S^* is the new solution after the time of disruption t_D . $U_n(S^*) = \sum_{\acute{j}} CR_{m\acute{j}}$ represents the real utility (makespan) in real schedule and $I_n(S^*) = \sum_i \sum_{\acute{j}} |CR_{i\acute{j}} - CP_{i\acute{j}}|$ refers to the instability performance where $CR_{i\acute{j}}$ and $CP_{i\acute{j}}$ are the real and predicted completion time respectively, according to the baseline. Also, $R_n(S^*) = |\sum_{\acute{j}} CR_{m\acute{j}} - \sum_{\acute{j}} CP_{m\acute{j}}|$ is the measure of robustness where $\sum_{\acute{j}} CP_{m\acute{j}}$ is the predictive value of makespan corresponding to the baseline. Moreover, n is the total number of jobs and m is the total number of machines. As well as, $i = \{1, 2, ..., n\}$ refers to the index for machines and $\acute{j} = \{1, 2, ..., \acute{n}\}$ is the index of jobs which have not been processed at the time of disruption on the first machine yet and the newly arrived jobs. The subsequence of the jobs that have already been executed or are in progress on the first machine before the time of disruption is denote by π_{BD}. Also, the permutable partial fixed list of jobs that arrive after the moment of disruption whose sequential index can be adjusted is represented as π_{AD}. It should be noted that the model (1) considers the case of n jobs and m machines. To enable a fair comparison between models, the objectives of the MO model are normalised as follows:

$$NMO = \alpha NU_n(S^*) + \beta NI_n(S^*) + \gamma NR_n(S^*) \tag{2}$$

where the function (NMO) is the normalised version of MO model. Also, $NU_n(S^*)$, $NI_n(S^*)$ and $NR_n(S^*)$ are the normalised measures of makespan, instability and robustness respectively. The procedure of normlisation the new objectives $NU_n(S^*)$ and $NI_n(S^*)$ are explain in details in [1]. However, the $NR_n(S^*)$ measure is calculated as follows:

$$NR_n(S^*) = \frac{R_n(S^*) - Min(R_n)}{Max(R_n) - Min(R_n)} \tag{3}$$

in this equation, the term $R_n(S^*)$ represents the robustness measure after the time of disruption t_D. Also, $Max(R_n)$ is the upper robustness bound and $Min(R_n)$ is the lower robustness bound. These bounds are obtained by solving the MO model three times where $\alpha = 1$ for the first solution while both of β and γ are zeros, also, $\beta = 1$ for the second solution while both of α and γ are zeros, and finally, $\gamma = 1$ for the last solution while both of α and β are zeros. These values are given in 3×3 matrix form as follows:

$$\begin{array}{cc} & \begin{array}{ccc} U_n & I_n & R_n \end{array} \\ \begin{array}{c} U_n^* \\ I_n^* \\ R_n^* \end{array} & \begin{pmatrix} f_{1,1} & f_{1,2} & f_{1,3} \\ f_{2,1} & f_{2,2} & f_{2,3} \\ f_{3,1} & f_{3,2} & f_{3,3} \end{pmatrix} \end{array} \tag{4}$$

The values of $Max(R_n)$ and $Min(R_n)$ are then calculated as follows: $Min(R_n) = min\{f_{1,3}, f_{2,3}, f_{3,3}\}$ and $Max(R_n) = max\{f_{1,3}, f_{2,3}, f_{3,3}\}$.

3 The proposed Solution Method

We introduce a robust rescheduling approach of predictive-reactive procedure for the dynamic PFSP in the presence of machine breakdown and arrival of new jobs. These real-time events could occur separately or together during the scheduling process. A predictive-reactive technique generates an initial robust predictive schedule then it reacts at each disruption point (rescheduling point) and it uses an IG algorithm at the rescheduling point for the subsequence of jobs which have not been processed on any machine yet after the disruption point (π_{AD}) with the objective of minimising the multi-objective function of model MO for π_{AD}. Figure (1) explain the procedure of the predictive-reactive approach.

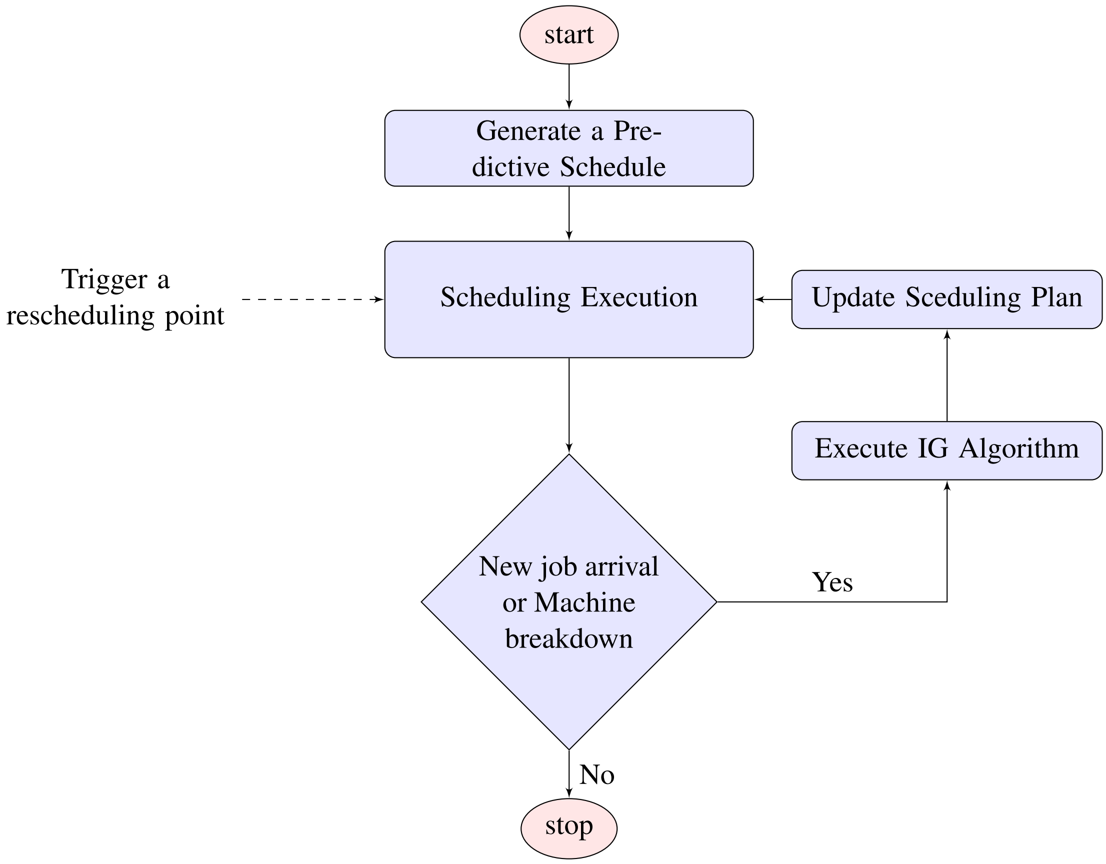

Figure 1: Predictive-Reactive approach

3.1 Iterated Greedy (IG)

The application of IG algorithm for the PFSP has been proposed by [17]. This method has been used successfully for many types of scheduling problems since then. The main feature of the IG algorithm is its simplicity where it has very few parameters. Also, the IG algorithm has shown the best performance for different flow shop scheduling problems with different objectives. To construct an initial solution for the PFSP, the IG algorithm begins with an initial solution as the current solution, this initial solution is generated by the NEH algorithm of Nawaz, Enscore, and Ham [22]. It is based on the idea that jobs with high total processing times on all machines should be scheduled as early as possible. The IG algorithm consists of two phases; destruction and construction. In the destruction phase d jobs are selected randomly and extracted from the current permutation π and inserted into a list of removed jobs π_R. Then, in their construction phase, the NEH insertion procedure is applied to reinsert all jobs from π_R individually into π again. At the construction phase, there is an optional step of applying a local search technique to improve the generated solutions. This local search step is based on the insertion neighborhood technique, which is efficeint and common local search precuedure for the PFSP [17]. For a permutation of jobs π, the insertion neighborhood is determined by taking in account all possible orders of pairs $r, s \in \{1, 2, ..., n\}$ of π, $r \neq s$ where the job at location r is removed and reinserted into location s. Thus, the new list of jobs will be as follows: $\acute{\pi} = (\pi(1), ..., \pi(r-1), \pi(r+1), ..., \pi(s), \pi(r), \pi(s+1), ..., \pi(n))$ if $r < s$, or $\acute{\pi} = (\pi(1), ..., \pi(s-1), \pi(r), \pi(s+1), ..., \pi(r-1), \pi(r+1), ..., \pi(n))$ if $r > s$. The sequence of insertion moves I is determined as $\{(r, s) : j \neq s, 1 \leq r, s \leq n \wedge r \neq s-1, 1 \leq r \leq n, 2 \leq s \leq n\}$, also the insertion neighborhood of sequence π is defined as $V(I, \pi) = \{\pi_v : v \in I\}$. The optional local search phase is detailed in Algorithm 1 where $C_{max}(\pi)$ is the total completion time for the sequence of jobs π.

Algorithm 1 Iterative improvement of neighborhood Local search

procedure ITERATIVE_IMPROVEMENT_INSERTION (π)
    improve := true;
    while (improve = true) **do**
        for i:=1 to n **do**
            remove a job s at random from π (without repetition)
            $\acute{\pi}$:= best permutation obtained by inserting s in any possible positions of π;
            if $C_{max}(\acute{\pi}) \leq C_{max}(\pi)$ **then**
                $\pi = \acute{\pi}$
                improve := true;
            end if
        end for
    end while
    Return π
end procedure

The next step is to decide whether to keep the incumbent solution or replace it with the new one, and to do this, we use an acceptance criterion based on the constant temperature Simulated Annealing-like criterion [19], which is basically calculates a constant temperature as follows:

$$Temperature = T \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} p_{ij}}{m \times n \times 10} \tag{5}$$

where T is a further value to calibrate. The final proposed IG method is given in Algorithm 2.

Algorithm 2 Pseudo code of IG Algorithm

procedure IG ALGORITHM
  Generate initial solution π_0;
  Apply Local search to π_0, and put modified solution into π_s;
  repeat
  $\pi_d = Destruction(\pi_s)$
  $\pi_c = Construction(\pi_d)$
  $\pi_l = LocalSearch(\pi_c)$
  $\pi_f = AcceptanceCriterion(\pi_s, \pi_l)$
  Until termination condition met
end procedure

4 Computational results

To verify the effectiveness of the proposed multi-objective model and approches, we provide the experimental evaluation and comparisons in this section. The proposed multi-objective model and solution method have been implemented in Java using the eclipse platform and Windows7 64bit, 6GB of RAM and a processor Intel Core i5, 2.30 ghz, was used for this purpose. The benchmark instances used are the ones introduced by [1] based on the well-known standard benchmarks of Taillard [23]. Taillard benchmarks are grouped into twelve sets, and each set has ten different instances from the same size, ranging from 20 jobs and 5 machines to 500 jobs and 20 machines. In this benchmark, some instances are very difficult to solve. In this experiment, each instance is run five times, then the average is calculated. To evaluate the efficiency of the proposed models and approach, a practical weight sensitivity algorithm [20] is applied; this algorithm gives thirteen different set of weights. These weights represent the relative importance of each objective in the MO model. To obtain the normalised model (2) we chose three weights, which are; *(0.999,0.001,0.001)*, *(0.001,0.999,0.001)* and *(0.001,0.001,0.999)*. The remaining weights are used to evaluate the multi-objective model (MO). These weights are as follows:
$W_1 =$ *(0.333, 0. 0.333, 0.333)*, $W_2 =$ *(0.666, 0.166, 166)*, $W_3 =$ *(0.499, 0.499, 0.002)*,
$W_4 =$ *(0.416, 0.416, 0.166)*, $W_5 =$ *(0.166, 0.666, 0.166)*, $W_6 =$ *(0.002, 0.499, 0.499)*,
$W_7 =$ *(0.166, 0.416, 0.166)*, $W_8 =$ *(0.166, 0.166, 0.666)*, $W_9 =$ *(0.499, 0.002, 0.499)*,
$W_{10} =$ *(0.416, 0.166, 0.416)*.
Additionally, the following components are used for the bi-objective model of [1];
$\alpha =$ *0.333, 0.666, 0.499, 0.416, 0.166, 0.002, 0.166, 0.166, 0.499, 0.416*.
For each instance, the Relative Percentage Deviation (RPD) over the best solution for each compared model is given as follows:

$$RPD = \frac{M - Best_{Sol}}{Best_{Sol}} \times 100 \tag{6}$$

Where $Best_{Sol}$ is the average lower bound solution of ten Taillard instances that have the same size $n \times m$ and M is the solution obtained from the presented models and methods. Table 1 illustrates the values of average RPD for Taillard problems corresponding to the objectives weights where **MO** denotes the solution values obtained by using the NMO model and **BO** is the solution from a bi-objective model [1].

Table 1: RPD for MO and BO models

	Ta	20 × 5	20 × 10	20 × 20	50 × 5	50 × 10	50 × 20	100 × 5	100 × 10	100 × 20	200 × 0	200 × 20	500 × 20	Total
W_1	**MO**	15.00	15.36	19.07	8.86	15.69	15.09	14.79	13.16	11.80	11.08	10.71	8.38	158.98
	BO	14.22	13.99	19.32	9.20	14.44	15.18	16.02	11.45	12.22	10.71	10.74	8.81	156.30
W_2	**MO**	11.77	16.81	20.37	8.16	16.03	14.45	10.89	11.65	11.70	12.00	11.62	8.83	154.26
	BO	15.95	16.71	17.71	9.10	16.59	14.30	18.20	12.00	12.35	11.62	10.88	9.62	165.02
W_3	**MO**	12.38	15.29	15.91	10.33	15.00	13.48	14.70	12.30	11.95	13.16	10.77	9.47	154.75
	BO	16.00	15.65	19.19	11.54	14.85	14.20	13.72	10.86	12.28	10.77	10.48	8.36	157.89
W_4	**MO**	13.61	8.83	15.87	9.39	15.00	15.02	16.71	11.35	11.70	10.97	11.48	9.47	149.40
	BO	12.72	10.64	16.26	9.82	14.85	14.70	15.35	11.70	11.95	11.48	10.75	8.36	148.58
W_5	**MO**	12.95	14.74	15.60	9.47	15.32	14.12	16.78	10.92	11.99	15.06	10.46	8.91	156.31
	BO	14.13	15.71	16.00	10.57	15.71	13.77	15.48	12.15	12.59	10.46	11.10	8.55	156.23
W_6	**MO**	**12.37**	**10.48**	**15.58**	**8.23**	**14.75**	**14.81**	**13.36**	**12.19**	**11.86**	**11.78**	**10.51**	**8.90**	**144.79**
	BO	13.01	11.98	15.97	8.85	17.47	14.22	19.87	11.34	11.93	10.51	10.66	8.99	154.79
W_7	**MO**	12.46	10.19	16.23	9.37	15.21	14.59	12.13	12.08	11.74	13.07	10.36	8.12	145.53
	BO	13.37	12.03	16.08	8.74	15.21	14.26	15.00	12.34	12.62	10.36	10.92	8.45	149.38
W_8	**MO**	13.71	13.02	16.68	10.82	15.48	13.72	14.92	11.75	12.35	10.93	10.73	8.83	152.92
	BO	15.37	13.30	16.06	8.86	16.58	14.63	14.58	11.98	11.92	10.73	10.91	8.46	153.38
W_9	**MO**	13.05	13.23	16.92	8.76	15.36	13.82	11.76	12.15	12.48	11.07	10.82	8.91	148.32
	BO	14.63	13.84	17.06	9.92	15.94	15.83	13.33	10.97	12.09	10.82	10.85	8.40	153.66
W_{10}	**MO**	12.72	17.37	17.57	7.36	15.34	15.83	16.98	11.66	12.16	11.83	10.71	9.49	159.01
	BO	12.43	13.86	16.39	10.88	15.43	14.56	17.41	11.02	11.94	10.71	10.71	8.99	154.31

In the first row of Table (1), **Ta** is the problem of size of $n \times m$. According to this table, different sets of weights produce different objective functions values, which means the solution is sensitive to different weights. It is clear from this table that the stability and robustness performances are key to improve the solution. As Table 1 shows, the weights W_4, W_6 and W_7 have minimum RPDs in compared with the other weights. Moreover, the weight W_6 produces a solution with lower RPD values, which proves that giving more priority to the stability and robustness objectives produces minimum RPDs.

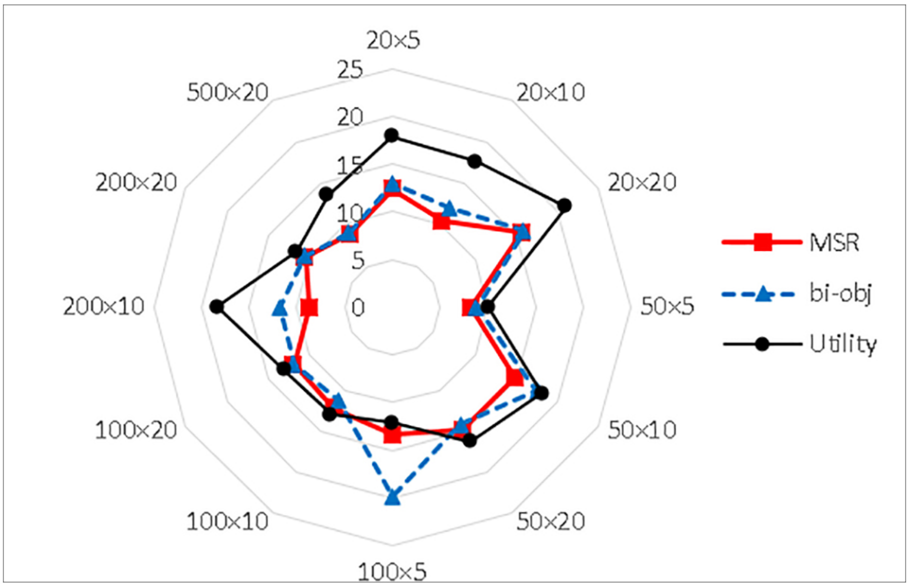

Figure 2: RPD for all models corresponding to W_6

For this reason, we select the weight corresponding to the lowest RPD value, which is W_6. The RPD for models MO, BO and UT corresponding to weight W_6 are given in figure 2 where UT represents the solution of a classical model of minimising total completion time (makespan). From this figure, it is clear that the MO model has minimum RPD values in general compared with the other models.
The single factor mean of an Analysis of Variance (ANOVA) is applied for further investigation for the impact of these optimisation models on the dependent variable RPD. The ANOVA tested statistically the results of all models where MO corresponding to W_6 and BO corresponding to $\alpha = 0.449$. In ANOVA, both hypotheses (null and alternative) are given as follows:

H_0: all means are same.
H_A: at least one mean is different.
Table 2 shows the p-values and F-ratios; it is clear that the p-values is smaller than 0.05, and hence the null hypothesis of no difference is rejected. This means statistically, that these factors have a significant impact on RPD. The p-values only permit to reject the null hypothesis and hence accept the alternative one. However, it does not show which of models have significant different mean values. So, we perform a 95% confidence interval test to determine the models that are significantly different. Figure 3 shows that both of the MO and BO models are significantly different with the UT model.

Table 2: Analysis of Variance.

Groups	**Count**	**Sum**	**Average**	**Variance**
MO	12	141.768	11.814	6.254
BO	12	156.06	13.005	11.145
UT	12	182.842	15.236	11.594

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	72.462	2	36.231	3.748	0.034	3.284
Within Groups	318.939	33	9.664			
Total	391.401	35				

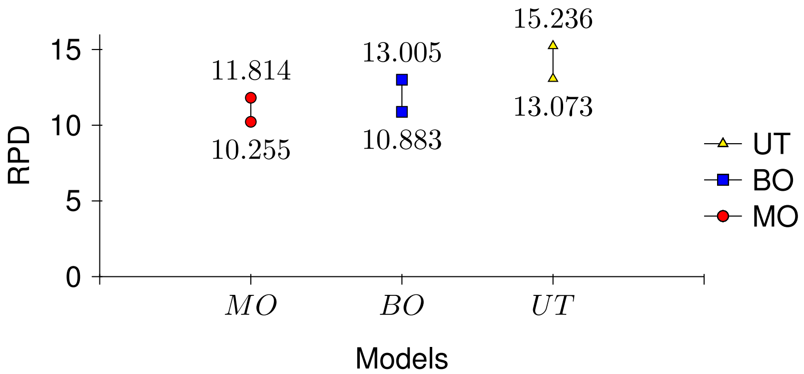

Figure 3: 95% Tukey confidence intervals for all models

5 Conclusion

In this work, a multi-objective optimisation model is developed to generate stable and robust schedules for the PFSP under two different uncertainties, which are; machine breakdown and new job arrivals. A new solution method, applying the effective predictive-reactive procedure and an IG method has been proposed. The obtained results show the high performance of the IG algorithm, which is handling the dynamic PFSP with large size instances and generating robust solutions. Moreover, the comparison of the multi-objective model against the bi-objective model [1] and the single objective model of makespan, indicates that the multi-objective optimisation model outperforms the aforementioned models. The computational results of sensitivity analysis revealed that a higher priority for stability and robustness performances resulted in low RPDs. From the statistical test of ANOVA, it can be seen that the multi-objective optimisation model has a significant difference with classical makespan model. For future works, other approaches and solution methods can be implemented for the purpose of comparison. Another suggestion is to use the multi-objective optimisation model for the PFSP under other different types of uncertainties.

Finally, the presented model and approaches can be adjusted for other scheduling problems, e.g. other flow shop scheduling problems and job shop scheduling problem.

References

[1] K. Katragjini, E. Vallada, and R. Ruiz, "Flow shop rescheduling under different types of disruption," *International Journal of Production Research*, vol. 51, pp. 780–797, 2013.

[2] R. L. Graham, E. L. Lawer, J. Lenstra, and K. A. H. G. R. Kan, "Optimisation and approximation in deterministic sequencing and scheduling: a survey," *Annals of Discrete Mathematics*, vol. 5, pp. 287–326, 1979.

[3] I. Sabuncuoglu and S. Goren, "Hedging production schedules against uncertainty in manufacturing environment with a review of robustness and stability research," *International Journal of Computer Integrated Manufacturing*, vol. 22, no. 2, pp. 37–41, 2009.

[4] H. Suwa and H. Sandoh, *Online Scheduling in Manufacturing A Cumulative Delay Approach*. Springer-Verlag, 2013.

[5] 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.

[6] D. Ouelhadj and S. Petrovic, "A survey of dynamic scheduling in manufacturing systems," *Journal of Scheduling*, vol. 12, no. 4, pp. 417–431, 2009.

[7] P. Cowling and M. Johansson, "Using real time information for effective dynamic scheduling," *European Journal of Operational Research*, vol. 139, no. 2, pp. 230–244, 2002.

[8] P. Cowling, D. Ouelhadj, and S. Petrovic, "A multi-agent architecture for dynamic scheduling of steel hot rolling," vol. 14, pp. 457–470, 2003.

[9] D. Rahmani and M. Heydari, "Robust and stable flow shop scheduling with unexpected arrivals of new jobs and uncertain processing times," *Journal of Manufacturing Systems*, vol. 33, no. 1, pp. 84–92, 2014.

[10] L. Chhurch and R. Uzsoy, "Analysis of periodic and event-driven rescheduling policies in dynamic shops," *International Journal of Computer Integrated Manufacturing*, vol. 5, no. 3, pp. 153–163, 1992.

[11] R. O'Donovan, R. Uzsoy, and K. McKay, "Predictable scheduling of a single machine with breakdowns and sensitive jobs," *International Journal of Production Research*, vol. 37, no. 18, pp. 4217–4233, 1999.

[12] G. Vieira, J. Herrmann, and E. Lin, "Predicting the performance of rescheduling strategies for parallel machine systems," *Journal of Manufacturing Systems*, vol. 19, no. 4, pp. 256–266, 2000.

[13] N. Hall and C. Potts, "Rescheduling for new orders," *Operations Research*, vol. 52, no. 3, pp. 440–453, 2004.

[14] R. Rangsaritratsameea, W. G. F. Jr., and M. B. Kurz, "Dynamic rescheduling that simultaneously considers efficiency and stability," *Computers & Industrial Engineering*, vol. 46, no. 1, pp. 1–15, 2004.

[15] K. M., E. G. Capn-Garca, A. Espua, and L. Puigjaner, "Costs for rescheduling actions: A critical issue for reducing the gap between scheduling theory and practice," *Ind. Eng. Chem. Res.*, vol. 47, no. 22, pp. 8785–8795, 2008.

[16] J.-Q. Li, Q. ke Pan, and K. Mao, "A discrete teaching-learning-based optimisation algorithm for realistic flowshop rescheduling problems," *The Scientific World Journal*, vol. 2014, pp. 1–11, 2014.

[17] R. Ruiz and T. Stützle, "A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem," *European Journal of Operational Research*, vol. 177, p. 20332049, 2007.

[18] Q.-K. Pan, R. Ruiz, and P. Alfaro-Fernndez, "Iterated search methods for earliness and tardiness minimisation in hybrid flowshops with due windows," *Computers and Operations Research*, vol. 80, pp. 50–60, 2017.

[19] I. Osman and C. Potts, "Simulated annealing for permutation flow-shop scheduling," *OMEGA, The International Journal of Management Science*, vol. 17, no. 6, pp. 551–557, 1989.

[20] D. Jones, "A practical weight sensitivity algorithm for goal and multiple objective programming," *European Journal of Operational Research*, vol. 213, no. 1, pp. 238–245, 2011.

[21] P. Cowling, O. Djamila, and S. Petrovic, "Dynamic scheduling of steel casting and milling using multi-agents," *Production Planning & Control*, vol. 15, no. 2, pp. 178–188, 2004.

[22] M. Nawaz, E. E. E. Jr, and I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem," *OMEGA, The International Journal of Management Science*, vol. 11, no. 1, pp. 91–95, 1983.

[23] E. Taillard, "Benchmarks for basic scheduling problems," *European Journal of Operational Research*, vol. 64, no. 2, pp. 278–285, 1993.