Dynamic Job Shop Scheduling Under Uncertainty Using Genetic Programming

Deepak Karunakaran, Yi Mei, Gang Chen and Mengjie Zhang

School of Engineering
and Computer Science
Victoria University of Wellington
Wellington, New Zealand
{deepak.karunakaran,yi.mei,aaron.chen,mengjie.zhang}@ecs.vuw.ac.nz

Abstract. Job shop scheduling(JSS) is a hard problem with most of the research focused on scenarios with the assumption that the shop parameters such as processing times, due dates are constant. But in the real world uncertainty in such parameters is a major issue. In this work, we investigate a genetic programming based hyper-heuristic approach to evolving dispatching rules suitable for dynamic job shop scheduling under uncertainty. We consider uncertainty in processing times and consider multiple job types pertaining to different levels of uncertainty. In particular, we propose an approach to use exponential moving average of the deviations of the processing times in the dispatching rules. We test the performance of the proposed approach under different uncertain scenarios. Our results verified the effectiveness of the newly developed terminal when the level of uncertainty is not high. In addition, the training configuration selection has an important impact on the generalisation of the evolved rules.

1 Introduction

Most researches on job shop scheduling (JSS) [22] use a deterministic model, in which the shop parameters such as processing times are constant throughout the realization of a schedule. However, in practice, the job shop environment always has uncertainty which makes scheduling a challenging and difficult task [12]. Handling uncertainty during scheduling is of practical importance.

Note that uncertainty is different from dynamic environment. For example, in dynamic JSS (DJSS), the processing time of a job becomes a known constant after its arrival, although unknown in advance. However, with uncertainty in processing time, the actual processing time is uncertain until the operation is completed, and may be different from the (expected) value upon the job arrival. In this paper, we focus on the DJSS with uncertain processing time, since processing time has a huge impact on most scheduling objectives.

Dispatching rules have been shown to be a promising approach for solving DJSS. There have been extensive studies (e.g. [7, 6, 14, 16, 13]) for using dispatching rules for scheduling under uncertainty, and dispatching rules specifically for

dealing with uncertainty such as the "largest variance first" [21] and "smallest variance first" [20] rules were proposed.

Design of dispatching rules is challenging and requires rigorous experimental validation. Recently genetic programming (GP) has been successfully used to automatically evolve dispatching rules [2] due to its flexible representation. However, the existing studies only focus on JSS without uncertainty, in which the properties of each job is exactly known after its arrival. The JSS with uncertainty has been overlooked so far.

It is unknown whether GPHH for JSS without uncertainty can be generalised to JSS with uncertainty. In this paper, our goal is to investigate the generalisation of GPHH under the uncertain processing time, i.e. the processing time of each operation can only be known after it is completed. Specifically, we have the following research objectives:

- Develop a new simulation model for JSS with uncertain processing time for evaluating dispatching rules;
- Study the generalisation of GPHH under different levels of uncertainty;
- Propose new specific terminals for GPHH to improve its ability to tackle uncertainty.

The rest of the paper is organised as follows. In the next section, we present the background to JSS and GPHH. In Section 3, we develop a new simulation model that includes the uncertain processing time. In Section 4, we describe our proposed method which is based on using exponential moving average of processing time deviation in the dispatching rules. In Sections 5 and 6, we present and discuss the experiment design and results. Finally, our conclusions and future work are given in Section 7.

2 Background

2.1 Job Shop Scheduling

JSS problem is to complete n jobs by a job shop with M machines. Each job j has an arrival time r_j, a due date d_j, and a sequence of operations $(o_{j,1} \rightarrow o_{j,2} \rightarrow , \ldots, o_{j,n_j})$, following a predefined route. The ith operation $o_{j,i}$ has a processing time $p_{j,i}$ and the machine $m_{j,i}$ to process it. The problem is to complete all the jobs so that (1) each machine processes no more than one operation at any time and (2) for each job, an operation cannot be started until all its precedent operations in the route have bee completed. Additional assumption included no recirculation of jobs, no preemption, no machine breakdown, and zero transit time between machines.

In dynamic environment, the jobs arrive at the shop in real time. The properties of each job such as the processing times and machines for its operations are unknown until its arrival. When new jobs arrive, the current schedule needs to be adjusted immediately to adapt to the environment change (add the new jobs in the schedule). Furthermore, in JSS with uncertain processing time, the exact processing time of each operation is unknown until it has been completed.

Commonly considered objectives in JSS include the makespan, total flowtime, mean weighted tardiness, number of tardy jobs, etc. In this paper, we consider minimising the total flowtime, which is defined as $F = \sum_j (C_j - r_j)$, where C_j is the completion time of job j.

2.2 Genetic Programming Based Hyper Heuristics (GPHHs)

Hyper-heuristics are a set of approaches which automate the design of heuristics to solve hard problems, particularly combinatorial optimization problems [4]. They search for *heuristic* rather than *solution*.

GP has been shown to be a promising hyper-heuristic [5], and has been successfully applied to evolving dispatching rules in JSS [2]. A variety of important and challenging issues have been investigated, such as rule representations [19, 3], feature selection [10, 18], reusability [17], many-objective JSS [15], machine-related rules [11] and flexible JSS [9]. For example, Tay and Ho [9] evolved scalable and flexible dispatching rules for multi-objective flexible job shop problem. Hildebrandt et al. [8] used GPHH to evolve dispatching rules which generalize well across varying scenarios in the job shop. Hunt et al. [10] introduced "less-myopic" terminals to evolved better dispatching rules. Motivated by the success of GPHH in DJSS, we also use GPHH for evolving dispatching rules for DJSS with uncertain processing time.

3 Simulation Model with Uncertain Processing Time

To carry out experimental studies, a simulation model is required to take uncertain processing time into account. There is no such model so far. Therefore, we develop a new simulation model by extending from the simulation without uncertainty. When a job j arrives, the processing time of each operation $o_{j,i}$ is known as a random variable $P_{j,i}$ rather than a fixed value $p_{j,i}$.

The distribution of the uncertain processing time should refect the real situation. Rai et al. [23] considered the scheduling in printing industry, where the job characteristics are the main sources of variation in processing times [23]. Akker et al. [1] considered processing times with a deterministic component and a random disturbance, which are identically distributed for the operations belonging to the same job. Based on the above existing studies, we assume that (a) the processing time the operations of the same job follow the same distribution and (b) the processing time of different jobs follow different distributions.

In the proposed simulation model, for each operation $o_{j,i}$, the processing time $p_{j,i}$ without uncertainty is assumed to be the expected value, and the actual processing time $p'_{j,i}$ is assumed to be the expected processing time plus a random non-negative delay that is proportional to the expected value, i.e. $p'_{j,i} = (1+\delta_{j,i})p_{j,i}$, $\delta_{j,i} \geq 0$. Here, we assume that the delay factor $\delta_{j,i}$ follows the Gamma distribution, which is widely used to model parameters that are required to be positive or skewed, and have been used to model uncertainty (e.g. [13]). The Gamma distribution has two parameters, *shape* ($\alpha \in \mathbb{R}^+$) and *scale* ($\beta \in \mathbb{R}^+$). To facilitate study, we choose $\alpha = 1$ and $\beta \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.8, 1.2\}$ in

our experiments. The choice of $\alpha = 1$ is motivated by the evidence from literature that the information contained in standard deviation of uncertain processing times is useful for scheduling [21, 13]. For a Gamma distribution, the standard deviation is equal to $\alpha \times \beta^2$ and the mean is equal to $\alpha \times \beta$. If we assign $\alpha = 1$, then the mean and the standard deviation of this Gamma distribution are β and β^2 respectively. Note that when $\alpha = 1$, the distribution essentially becomes exponential, which has been frequently used to model uncertain processing times.

4 Exponential Moving Average (EMA) Terminal for Handling Uncertain Processing Time

When evolving dispatching rules for JSS with uncertain processing time, it is natural to design new terminals for GP to capture the information about the uncertainty, e.g. the distribution of the uncertain processing times. To this end, we develop a new terminal based on the *exponential moving average* of the previous delays, which can be used to estimate the future delay. The new terminal, called EMA, is described as follows.

Based on the assumption that the processing time of all the operations belonging to the same job follow the same distribution, we have $\beta(\delta_{j,i_1}) = \beta(\delta_{j,i_2})$, $\forall\ j$. In other words, all the $\delta'_{j,i}s$ of the same job have the same mean value. Therefore, we can use the exponential moving average of the previous δ values to estimate the mean value of the distribution, and predict the most possible value of the future δ values of future operations of the same job. Specifically, for each job j, the newly proposed EMA terminal, denoted as $\bar{\delta}_{j,i}$, is defined as:

$$\begin{aligned} \bar{\delta}_{j,1} &= 0, \\ \bar{\delta}_{j,i+1} &= \kappa \cdot \delta_{j,i} + (1-\kappa) \cdot \bar{\delta}_{j,i}, \forall\ i > 1 \end{aligned}$$

where $\delta_{j,i} = p'_{j,i}/p_{j,i} - 1$ is the empirical delay factor. From literature, the coefficient $\kappa \leq 0.3$ is considered to be good. Our preliminary studies did no show significant difference between values $\kappa \in \{0.1, 0.2, 0.3\}$. Therefore, we set $\kappa = 0.2$ in our experiments.

5 Experiment Design

In the experimental studies, we aim to analyse the generalisation of the standard GP on the JSS with uncertain processing time, and investigate the efficacy of the newly developed terminal in improving the performance of GP. To this end, we ran the standard GP with the original terminal set (shown in Table 2) and the original terminal set plus the new terminal. The simulation model proposed in Section 3 was used to generate the training and test instances. 30 independent runs were conducted for each algorithm.

5.1 Training and Test Configurations

Training In the experiments, we used two configurations for generating training instances by varying the number of job types.

- In the first case, we assume two job types, i.e. the low-delay and high-delay job types. The low-delay job type follows the distribution with ($\alpha = 1, \beta = 0.1$), and the high-delay one with ($\alpha = 1, \beta = 0.6$). Each job has 50% probability to be either of the two job types.
- In the second case, we consider five types of jobs, whose Gamma distribution parameters are $\alpha = 1,\ \beta \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$. Each job has 20% probability to be any of the five job types.

Testing We designed seven different test sets with varying number of job types. The configurations of the test sets are given in Table 1. Each test set consists of 30 problem instances. The column "#job-type-ratio" shows the ratio of the number of jobs assigned to each type. For example, in test set-II, each job has probability of 2/3 to have $\beta = 0.1$ and probability of 1/3 to have $\beta = 0.6$.

Table 1: Test Configurations

Test-set	scale (β)	#job-type-ratio
I	$\{0.1, 0.6\}$	1 : 1
II	$\{0.1, 0.6\}$	2 : 1
III	$\{0.1, 0.6\}$	3 : 1
IV	$\{0.1, 0.3, 0.6\}$	1 : 1 : 1
V	$\{0.1, 0.3, 0.6, 0.8\}$	1 : 1 : 1 : 1
VI	$\{0.1, 0.2, 0.3, 0.4, 0.5\}$	1 : 1 : 1 : 1 : 1
VII	$\{0.1, 0.3, 0.6, 0.8, 1.2\}$	1 : 1 : 1 : 1 : 1

In both training and test simulations, the other parameters are consistent with that used in [19] (e.g. 2500 jobs and 500 jobs for warm-up). Full details can be found in [19].

5.2 Genetic Programming System

Table 2 lists all the terminals used by the GP in the experimental studies. The function set includes all the arithmetic operators (the protected division operator returns 1 if the denominator is zero), the 2-argument "max" and "min" operators and the 3-argument "If" operator that returns the second argument if the first argument is positive, and the third argument otherwise.

In the GP system, the population size is set to 1024, and the number of generations is set to 50. The maximal depth of GP-trees is set to 8. The crossover, mutation and elitism rates are set to 0.85, 0.1 and 0.05 respectively. This parameter setting is consistent with the one in [19].

6 Results and Discussions

Given two training sets and two compared algorithms, we obtained four sets of results. For the sake of convenience, we denote the four results as follows:

Table 2: Terminal Set for GP.

Terminal Set	Meaning
DD	Due date of job
PR	Processing time of operation
RO	Remaining operations for job
RJ	Ready time of job
RT	Remaining processing time of job
RM	Ready time of machine
ERC	Ephemeral Random constant
EMA	Exponential moving average of the delay factor $\delta_{j,i}$

- $\mathcal{D}_2$: the GP without the EMA terminal, trained on 2 job types;
- $\mathcal{D}_5$: the GP without the EMA terminal, trained on 5 job types;
- $\mathcal{D}_2^{\bar{\delta}}$: the GP with EMA terminal, trained on 2 job types;
- $\mathcal{D}_5^{\bar{\delta}}$: the GP with EMA terminal, trained on 5 job types.

Based on these four sets of results, we conducted several comparisons to analyse (a) the efficacy of the new EMA terminal on the performance of GP to handle the uncertain processing time and (b) the impact of different training scenarios on the generalisation of the GP-evolved rules.

To investigate the efficacy of the newly developed EMA terminal, we compare between $\mathcal{D}_2^{\bar{\delta}}$ and $\mathcal{D}_2$ and between $\mathcal{D}_5^{\bar{\delta}}$ and $\mathcal{D}_5$. To analyse the impact of different training scenarios, we compare between $\mathcal{D}_2^{\bar{\delta}}$ and $\mathcal{D}_5^{\bar{\delta}}$, and between $\mathcal{D}_2$ and $\mathcal{D}_5$. We present the comparative results in Figs. 2–3. Though we conducted our experiments on 30 problem instances in each test set, we presented the comparisons for only 9 of them due to space limitation. We observe similar pattern for other instances. For each comparison, Wilcoxon rank sum test under significance level of 0.05 was carried out for each of the 30 test instances, and the summary is provided below each figure in the form of [Win – Draw – Lose]. In each figure, the x-axis is the instance ID, and the y-axis is the test performance of the dispatching rules evolved by the compared algorithms. For each comparison, if the former algorithm is significantly better, then the corresponding boxplot is filled in red. If the latter is significantly better, then it is filled in yellow.

6.1 Efficacy of the EMA Terminal: $\mathcal{D}_2^{\bar{\delta}}$ vs $\mathcal{D}_2$ and $\mathcal{D}_5^{\bar{\delta}}$ vs $\mathcal{D}_5$

Fig. 2 shows the comparison between $\mathcal{D}_2^{\bar{\delta}}$ and $\mathcal{D}_2$ on all the seven test sets. Our primary observation is that the newly developed EMA terminal is able to evolve rules which perform better across most of the test configurations. For all pairs of box plots in Fig. 2, the right one corresponds to $\mathcal{D}_2^{\bar{\delta}}$. To be specific:

- When two types of jobs are considered, as in test sets I, II & III, the performance of $\mathcal{D}_2^{\bar{\delta}}$ is significantly better in 14, 26 and 22 out of a total 30 problem instances respectively.
- Similar results are obtained for the test sets IV & V. In these two test sets, the types of jobs considered are 3 and 4 respectively(Table. 1). The performance

of the dispatching rules evolved with the EMA terminal is significantly better in 23 and 13 out of 30 problem instances, respectively.
– We observe that when the number of job types increases to 5, the performance is not good for the dispatching rules which were trained using two types of jobs. In the box plots shown in Fig. 2, the performance is in fact significantly 'worse', especially when the jobs have higher uncertainty levels (test set VII).

This can be explained more clearly when we consider the role of a dispatching rule in generating the schedule. As the jobs arrive at the shop, their operations are queued to the appropriate machines. The dispatching rule assigns priority values to the operations, which are used to select the next operation to be processed on the machine. If the number of job types increases, then all the operations in the queue have variations in processing times. Consequently, the problem becomes harder as the processing time deviation is similarly prominent throughout all the operations in a queue, and GP is not able to accurately prioritize the operations using the uncertainty information contained in $\bar{\delta}$.

Moreover, when the ratio is skewed, i.e. a higher percentage of jobs are associated with one probability distribution than the other, the performance using the new terminal is more significant. This is in line with the previous explanation. The rules are able to assign the priorities more accurately, when the deviation is very prominent in some jobs and less prominent in other. To be specific, for the test sets II, III & IV the performance is better than others because, the deviations are high for a smaller percentage of jobs (33%, 25% and 33% respectively) and low for the remaining. In this case, GP is able to assign priorities to these operations accurately.

Fig. 3(a) shows the comparative results between $\mathcal{D}_5^{\bar{\delta}}$ and $\mathcal{D}_5$ on some representative test sets. Our primary observation is that when using 5 types of jobs in the training set, there is no significant difference between the GPs with and without the new terminal on all the test sets. Our explanation to this phenomenon is similar to the one given before. The scheduling problem becomes more difficult when the number of levels of uncertainty assigned to the different jobs increase. Also, this result points to the fact that $\mathcal{D}_2^{\bar{\delta}}$ suffers from over-fitting problem when presented with problems with $five$ levels of uncertainties in jobs as test sets.

6.2 Impact of Training Configurations: $\mathcal{D}_2^{\bar{\delta}}$ vs $\mathcal{D}_5^{\bar{\delta}}$ and $\mathcal{D}_2$ vs $\mathcal{D}_5$

We also cross-compare the test results between the set of evolved rules which are trained on different configurations. The results are shown as box plots in the Figs. 3 (b and c). For these pairs of box plots, the left one always corresponds to rules trained with 2 job types. From the figures, we have the following observations.

– For test set I with two different job types, the results show that the rules trained on 2 job types are significantly better than those trained on 5 job types. This is an expected result.

– Similarly, for test set VI with 5 job types, the rules obtained by $\mathcal{D}_5^{\bar{\delta}}$ performed significantly better than those obtained by $\mathcal{D}_2^{\bar{\delta}}$ on 13 test instances.

The comparisons show the importance of choosing proper training instances depending on the level of uncertainty in processing time.

6.3 Analysis of Dispatching rules

Now we analyze the evolved rules to make more sense of our observations. Firstly, we count the number of each terminal in the set of 30 evolved rules from $\mathcal{D}_2^{\bar{\delta}}$. The bar chart is shown in Fig. 1. It can be seen that the EMA terminal was the third most frequently chosen terminal, which indicates that the terminal is useful. Furthermore, we counted the number of occurrences of pairs of the terminal EMA and of PR, RT. These pairs are highlighted (in bold) in the example of a dispatching rule below (Listing 1.1). These terminals are chosen because the EMA value is expected to combine with terminals related to processing times. We found that among the 30 rules, they occurred 92 times (multiple occurrences within the same rule). Moreover, we calculated the frequency of these combinations for the dispatching rules in $\mathcal{D}_5^{\bar{\delta}}$. We found that the frequency is 61, which indicates that for a configuration with 5 job types, the effect of EMA is less prominent.

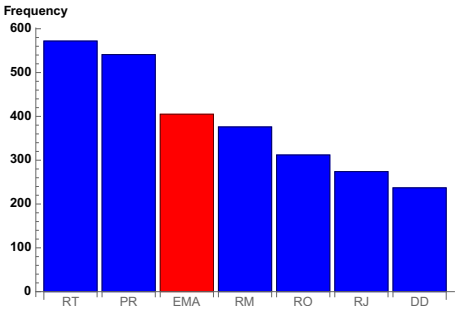


Fig. 1: Frequency of terminals

An example of one of the best evolved dispatching rules is presented below. We can see that the occurrences of the terminal EMA and its combination with the terminals related to processing times is frequent.

Listing 1.1: One of the best evolved rules

```
(* (Min (* RT EMA) (- (/ 0.584 RO) (+ (Max 0.724 RT)
(+ (* (* (If (+ (Max  EMA 0.219) (- RO DD)) (Max
(+ EMA(- RT RO))(If (+ 0.990 (* RT EMA))
0.341 (/ 0.956 (+(/ 0.584 RO) (Max(*RT EMA) (/ 0.584 RO)
))))) 2)(+(+ EMA(- RT RO)) (Min (Min (*EMA PR) (Max RJ
```

```
RM)) (- RT RO)))) (+ (Min RO 0.321) (Min RO 0.321))) (Max
(* RT  EMA) (/ 0.584 (If (+ (Max (*RT EMA)
(/ 0.584 RO)) (- RO DD)) (Max 0.724 RT) (+ 2 )))))))
(Max RO PR))
```

7 Conclusions

In this paper, we focused on evolving dispatching rules using a GP-based hyper heuristic approach for dynamic job shop scheduling problem under uncertain processing times. In particular, we introduced a new terminal to capture the information about uncertainty in processing times to evolve more promising dispatching rules. The terminal computes the exponential moving average of the delay factor, i.e. the proportion of the delay relative to the expected processing time. We considered different uncertainty levels in processing times and considered different ratios of jobs pertaining to these levels.

First, for experimental studies, we developed a new simulation model with uncertain processing times. Then, we conducted several comparisons to investigate the efficacy of the newly developed terminal and the influence of the training set selection on the generalisation of GP. The primary conclusion is that, the inclusion of the new EMA terminal can help GP evolve better dispatching rules with total flow time as the scheduling objective when the number of uncertainty levels is up to $four$. When the number of uncertainty levels becomes $five$, the efficacy of EMA shows a decline. This is because EMA is able to estimate the priority of the operations more accurately when they have similar uncertainty levels.

Furthermore, our observations concur with the fact that different job shop scenarios require specific dispatching rules. In particular, when the following parameters in the job shop are varied: (1) the number of different levels of uncertainty and (2) the ratio of jobs pertaining to the different levels of uncertainty; different sets of dispatching rules are required to maintain schedule quality.

In our future work, we will investigate better estimation techniques to evolve dispatching rules which perform well even with higher number of levels of uncertainty. We will also consider evolving dispatching rules which perform well under more types of uncertainty e.g. variation in due dates.

References

1. Van den Akker, M., Hoogeveen, H.: Minimizing the number of late jobs in a stochastic setting using a chance constraint. Journal of Scheduling 11(1), 59–69 (2008)
2. Branke, J., Nguyen, S., Pickardt, C., Zhang, M.: Automated design of production scheduling heuristics: A review 20(1), 110–124 (2016)
3. Branke, J., Hildebrandt, T., Scholz-Reiter, B.: Hyper-heuristic evolution of dispatching rules: A comparison of rule representations. Evolutionary computation 23(2), 249–277 (2015)
4. Burke, E.K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., Qu, R.: Hyper-heuristics: A survey of the state of the art. Journal of the Operational Research Society 64(12), 1695–1724 (2013)

5. Burke, E.K., Hyde, M.R., Kendall, G., Ochoa, G., Ozcan, E., Woodward, J.R.: Exploring hyper-heuristic methodologies with genetic programming. In: Computational intelligence, pp. 177–201. Springer (2009)
6. Calleja, G., Pastor, R.: A dispatching algorithm for flexible job-shop scheduling with transfer batches: an industrial application. Production Planning & Control 25(2), 93–109 (2014)
7. Gao, K.Z., Suganthan, P.N., Tasgetiren, M.F., Pan, Q.K., Sun, Q.Q.: Effective ensembles of heuristics for scheduling flexible job shop problem with new job insertion. Computers & Industrial Engineering 90, 107–117 (2015)
8. Hildebrandt, T., Heger, J., Scholz-Reiter, B.: Towards improved dispatching rules for complex shop floor scenarios: a genetic programming approach. In: Proceedings of the 12th annual conference on Genetic and evolutionary computation. pp. 257–264. ACM (2010)
9. Ho, N.B., Tay, J.C.: Evolving dispatching rules for solving the flexible job-shop problem. In: 2005 IEEE Congress on Evolutionary Computation. vol. 3, pp. 2848–2855. IEEE (2005)
10. Hunt, R., Johnston, M., Zhang, M.: Evolving less-myopic scheduling rules for dynamic job shop scheduling with genetic programming. In: Proceedings of the 2014 conference on Genetic and evolutionary computation. pp. 927–934. ACM (2014)
11. Jakobović, D., Jelenković, L., Budin, L.: Genetic programming heuristics for multiple machine scheduling. In: Genetic Programming, pp. 321–330. Springer (2007)
12. Kouvelis, P., Yu, G.: Robust discrete optimization and its applications, vol. 14. Springer Science & Business Media (2013)
13. Lawrence, S.R., Sewell, E.C.: Heuristic, optimal, static, and dynamic schedules when processing times are uncertain. Journal of Operations Management 15(1), 71–82 (1997)
14. Liu, K.C.: Dispatching rules for stochastic finite capacity scheduling. Computers & industrial engineering 35(1), 113–116 (1998)
15. Masood, A., Mei, Y., Chen, G., Zhang, M.: Many-Objective Genetic Programming for Job-Shop Scheduling. In: Proceedings of Congress on Evolutionary Computation. IEEE (2016)
16. Matsuura, H., Tsubone, H., Kanezashi, M.: Sequencing, dispatching and switching in a dynamic manufacturing environment. The International Journal of Production Research 31(7), 1671–1688 (1993)
17. Mei, Y., Zhang, M.: A comprehensive analysis on reusability of gp-evolved job shop dispatching rules. In: Proceedings of Congress on Evolutionary Computation. IEEE (2016)
18. Mei, Y., Zhang, M., Nyugen, S.: Feature selection in evolving job shop dispatching rules with genetic programming. In: Genetic and Evolutionary Computation Conference (GECCO) (2016)
19. Nguyen, S.: Automatic design of dispatching rules for job shop scheduling with genetic programming (2013)
20. Pinedo, M.: Stochastic batch scheduling and the "smallest variance first" rule. Probability in the Engineering and Informational Sciences 21(04), 579–595 (2007)
21. Pinedo, M., Weiss, G.: The largest variance first policy in some stochastic scheduling problems. Operations Research 35(6), 884–891 (1987)
22. Pinedo, M.L.: Scheduling: theory, algorithms, and systems. Springer Science & Business Media (2012)
23. Rai, S., Duke, C.B., Lowe, V., Quan-Trotter, C., Scheermesser, T.: Ldp lean document production-or-enhanced productivity improvements for the printing industry. Interfaces 39(1), 69–90 (2009)

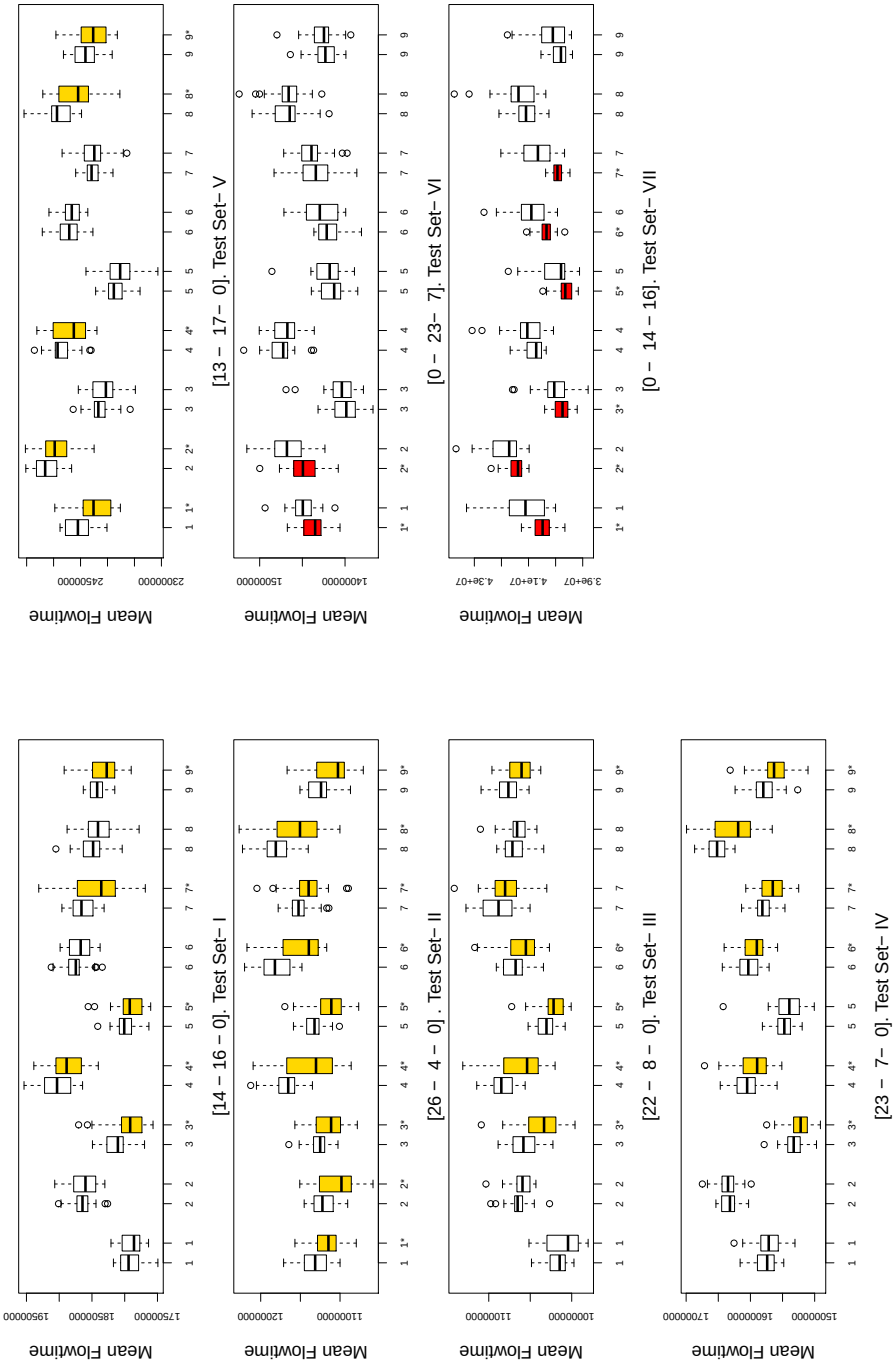
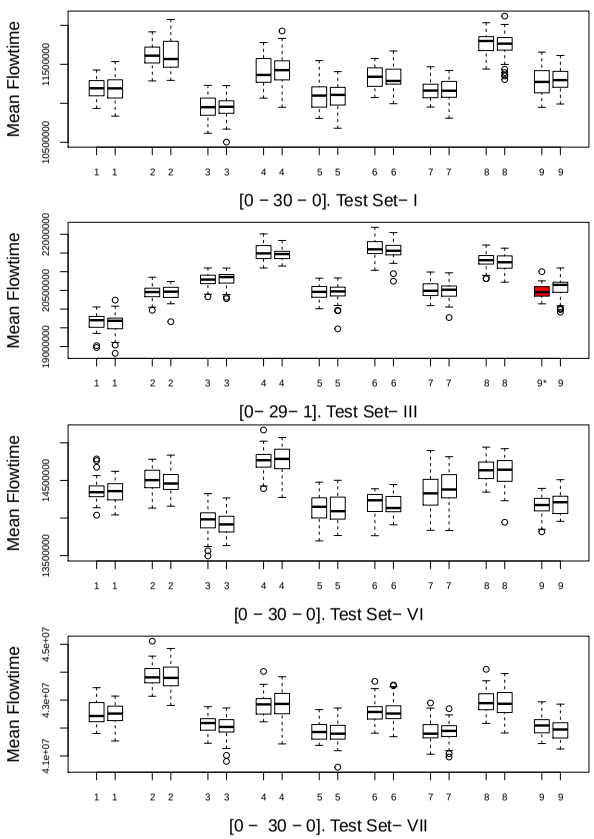
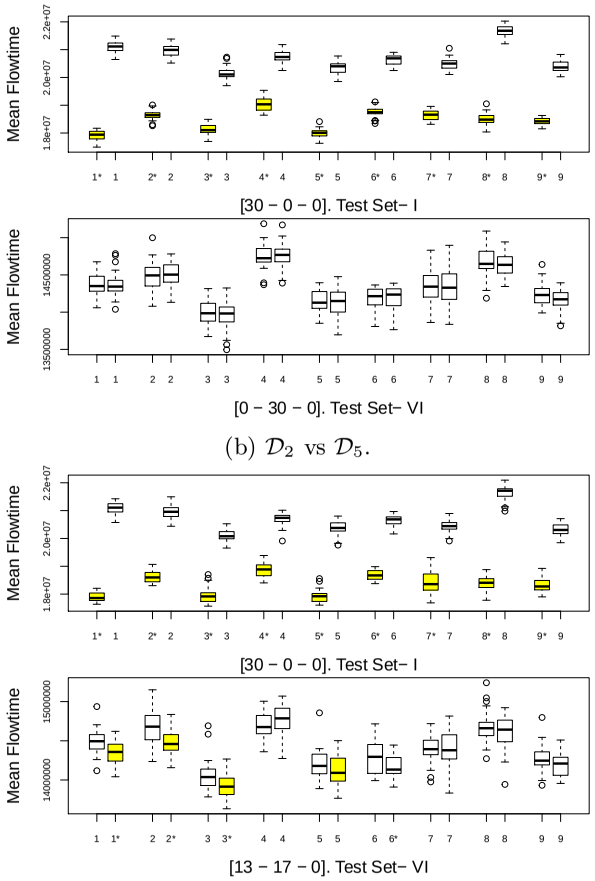


Fig. 2: $\mathcal{D}_2$ vs $\mathcal{D}_2^{\delta}$ on 9 representative instances of the 7 test sets. For each comparison, if the left (right) boxplot is significantly better (Wicoxon rank sum test under significance level of 0.05), then the corresponding boxplot is filled in red (yellow).



(a) $\mathcal{D}_5$ vs $\mathcal{D}_5^{\bar{\delta}}$.



(b) $\mathcal{D}_2$ vs $\mathcal{D}_5$.

(c) $\mathcal{D}_2^{\bar{\delta}}$ vs $\mathcal{D}_5^{\bar{\delta}}$.

Fig. 3: (a) $\mathcal{D}_5$ vs $\mathcal{D}_5^{\bar{\delta}}$, (b) $\mathcal{D}_2$ vs $\mathcal{D}_5$ and (a) $\mathcal{D}_2^{\bar{\delta}}$ vs $\mathcal{D}_5^{\bar{\delta}}$ on 9 representative instances. For each comparison, the significantly better boxplot is filled in color (red or yellow).