Surrogate-Assisted Genetic Programming for Dynamic Flexible Job Shop Scheduling

Fangfang Zhang, Yi Mei, and Mengjie Zhang

School of Engineering and Computer Science, Victoria University of Wellington,
PO BOX 600, Wellington 6140, New Zealand
{fangfang.zhang,yi.mei,mengjie.zhang}@ecs.vuw.ac.nz

Abstract. Genetic programming (GP) has been widely used for automatically evolving priority rules for solving job shop scheduling problems. However, one of the main drawbacks of GP is the intensive computation time. This paper aims at investigating appropriate surrogates for GP to reduce its computation time without sacrificing its performance in solving dynamic flexible job shop scheduling (DFJSS) problems. Firstly, adaptive surrogate strategy with dynamic fidelities of simulation models are proposed. Secondly, we come up with generation-range-based surrogate strategy in which homogeneous (heterogeneous) surrogates are used in same (different) ranges of generations. The results show that these two surrogate strategies with GP are efficient. The computation time are reduced by 22.9% to 27.2% and 32.6% to 36.0%, respectively. The test performance shows that the proposed approaches can obtain rules with at least the similar quality to the rules obtained by the GP approach without using surrogates. Moreover, GP with adaptive surrogates achieves significantly better performance in one out of six scenarios. This paper confirms the potential of using surrogates to solve DFJSS problems.

Keywords: Surrogate · Dynamic flexible job shop scheduling · Genetic programming.

1 Introduction

Flexible job shop scheduling (FJSS) is an extension to classical job shop scheduling (JSS). However, in FJSS, one operation can be processed on more than one machine rather than a specified machine. In order to tackle the FJSS problem, two decisions, which are a machine-specific decision and a job-specific decision, have to be made. The machine-specific decision is to allocate a ready operation to an appropriate machine while the job-specific decision aims to select one operation as the next to be processed. FJSS is NP-hard [3].

In practice, the environment is usually dynamic and jobs arrive in the job shop over time without prior information. Dynamic job shop scheduling (DJSS) was proposed for considering this situation. Dispatching rules, as priority functions, have been widely adopted for solving DJSS problems [2, 4], due to the ability to react in real time. A comprehensive comparison of dispatching rules can be found in [9]. Dynamic flexible job shop scheduling (DFJSS) considers

both the characteristics of FJSS and DJSS. Naturally, two kinds of dispatching rules are needed in DFJSS, which are routing rule (machine-specific) and sequencing rule (job-specific), respectively. In this case, the quality of DFJSS schedule depends highly on how well the routing rule and the sequencing rule work together. However, dispatching rules are normally manually designed. That is, the design of dispatching rules is domain-dependent and time-consuming.

Genetic programming (GP) has been successfully applied to automatically evolve dispatching rules for JSS [5, 7]. However, a challenge of using GP is the intensive computation time. Surrogate-assisted evolutionary computation with efficient computation models, known as surrogates, provides a promising means of handing complex applications [1, 8]. The challenge is how to design appropriate surrogates with cheaper computation time that can represent the original models well.

1.1 Goals

To address the challenge above, this paper has the following research objectives.

- Propose adaptive surrogates for GP (ASGP) approach to operate linearly diverse surrogates with different fidelities in the search process.
- Design generation-range-based surrogates for GP (GSGP) that uses homogeneous (heterogeneous) surrogates in the same (different) predefined ranges of generations.
- Verify the effectiveness and efficiency of the proposed algorithms.
- Compare the learning processes of the proposed two algorithms with standard GP without surrogates.

2 The Proposed Surrogate Strategies

2.1 Adaptive Surrogates

In this section, adaptive surrogates are proposed for GP and the corresponding algorithm is named as ASGP. The basic idea is to deliberately enlarge accuracy of the surrogate models by building up a very simple surrogates at the early stage. As the evolutionary optimization proceeds, the accuracy of the surrogates increases gradually and smoothly expecting that the performance of approximated surrogate models is consistent with the original model.

Let N_{job} and N_{warmup} represent the number of jobs and warmup jobs, respectively. At the ith generation, the number of jobs and warmup jobs are denoted as $N_{job,i}$ and $N_{warmup,i}$. The expressions of $N_{job,i}$ and $N_{warmup,i}$ are shown as Eq. (1) and Eq. (2), respectively. In this way, the number of jobs and warmup jobs will increase linearly.

$$N_{job,i} = \begin{cases} N_{job} * \frac{1}{maxGen-1} & gen = 0 \\ N_{job} * \frac{Gen}{maxGen-1} & 1 \leq Gen < maxGen \end{cases} \quad (1)$$

$$N_{warmup,i} = \begin{cases} N_{warmup} * \frac{1}{maxGen-1} & gen = 0 \\ N_{warmup} * \frac{Gen}{maxGen-1} & 1 \leq Gen < maxGen \end{cases} \quad (2)$$

2.2 Generation-range-based Surrogates

For the ASGP, at each generation, different surrogate models are applied. In this section, generation-range-based surrogates is proposed for GP (GSGP) to explore whether a fixed interval change can be more efficient. In this paper, the number of jobs and warmup jobs of the original simulation model are set to 5000 and 1000, respectively. We set every ten generations into a range. The setting details of different surrogates used in different generations are shown in Table 1.

Table 1. The setting of generation-range-based surrogates.

Generation ranges	$N_{job,i}$	$N_{warmup,i}$
$[0, 10)$	500	100
$[10, 20)$	1000	200
$[20, 30)$	1500	300
$[30, 40)$	2500	500
$[40, 50]$	5000	1000

3 Experiment Design

In our experiment, the terminal and function sets in [6] are adopted. It is worth mentioned that "/" is the protected division that returns the largest double positive number if divided by 0. For dynamic simulation, commonly used configuration is adopted [10]. This paper presents the results obtained by the proposed two approaches and CCGP approach [10], which is the state-of-the-art algorithm for DFJSS, using three objectives, namely: (1) max-flowtime, (2) mean-flowtime, and (3) mean-weighted-flowtime. The smaller the result, the better.

4 Results and Analyses

The $(-, +)$ marks show whether our proposed approaches converge significantly better or poorer than CCGP approach in Wilcoxon rank sum test ($p \leq 0.05$), respectively. For the convenience of description, $< obj, uti >$ indicates the simulation scenarios, where obj and uti are the objective and the utilization level.

4.1 Test Performance of Evolved Rules

Table 2 shows that ASGP and GSGP algorithms are no significantly worse than CCGP in general. The mean value obtained by ASGP are about equal with the value obtained by CCGP in all scenarios. It is noted that ASGP significantly outperforms CCGP in scenario $< tmean, 0.85 >$. This clearly shows the potential of using surrogates to improve the performance of GP. It also indicates that the surrogates (approximation models) may not be always harm.

For GSGP, the mean value obtained are smaller than CCGP in four (scenario 1, 2, 3, 5) out of six scenarios. In addition, the variances obtained by GSGP are smaller than CCGP in five (scenario 1, 2, 3, 5, 6) out of six scenarios.

4.2 Training Time

Table 3 shows the computation time (reductions produced by surrogates compared with CCGP) of the three algorithms. Overall, ASGP and GSGP need

Table 2. The mean and deviation error of normalized objective value of the compared algorithms over 30 independent runs for six scenarios.

Index	Scenario	ASGP	GSGP	CCGP
1	$< tmax, 0.85 >$	0.640(0.034)	0.638(0.029)	0.642(0.035)
2	$< tmax, 0.95 >$	0.571(0.023)	0.565(0.018)	0.568(0.030)
3	$< tmean, 0.85 >$	0.772(0.012)(-)	0.768(0.008)	0.772(0.015)
4	$< tmean, 0.95 >$	0.734(0.023)	0.738(0.022)	0.731(0.015)
5	$< twt, 0.85 >$	0.778(0.030)	0.772(0.010)	0.774(0.018)
6	$< twt, 0.95 >$	0.773(0.023)	0.774(0.024)	0.774(0.037)

less training computation times compared with CCGP. The average reductions produced by ASGP and GSGP are 25.7% and 34.4%, respectively.

The experimental results have confirmed that ASGP can reduce the computation time by at least 22.9% in six scenarios. In both scenario 2 and scenario 3, the computation time are reduced the most (27.2%). For GSGP, it is obvious that it can reduce more computation time (from 32.6% to 36.0%) than ASGP (from 22.9% to 27.2%). It is not surprising because the average fidelity of ASGP is higher than GSGP. In addition, the computation time is reduced the most (36.0%) in scenario 1 while the least (32.6%) in scenario 5.

Table 3. The average training time (reduction) of the compared algorithms over 30 independent runs for six scenarios.

Index	Scenario	Training Time (seconds)		
		ASGP	GSGP	CCGP
1	$< tmax, 0.85 >$	3399.8 (26.8%)	2969.9 (36.0%)	4642.8
2	$< tmax, 0.95 >$	3743.6 (27.2%)	3326.2 (35.3%)	5144.9
3	$< tmean, 0.85 >$	3302.5 (27.2%)	2935.0 (35.3%)	4538.5
4	$< tmean, 0.95 >$	3635.3 (25.0%)	3220.2 (33.6%)	4849.9
5	$< twt, 0.85 >$	3436.2 (22.9%)	3004.4 (32.6%)	4458.4
6	$< twt, 0.95 >$	3725.7 (24.8%)	3282.7 (33.8%)	4957.0

4.3 Insight the Learning Process

The lines in Fig. 1 are the average normalized objective value from 30 independent runs. Although all GP methods start with the same population, the starting points are different because they use different surrogates. To be specific, CCGP get the value from surrogates with higher fidelities while ASGP and GSGP get the value from surrogates with lower fidelities.

It is noted that both ASGP and GSGP have higher fluctuations in all scenarios than CCGP, especially at the early stage of evolutionary process. For ASGP, the fidelities of surrogate models change smoothly to handle the learning process gradually. It is expected to meet the need of training. It is interesting that Fig. 1 shows that ASGP and GSGP have basically the same trends in six scenarios. This indicates that the predefined ranges and settings of simulations in GSGP are representative for the learning process. In addition, after generation 40, ASGP and GSGP can achieve almost the same learning ability as CCGP, although they use surrogates with lower fidelities at previous generations.

Fig. 2 shows that CCGP can improve much faster at the beginning of the evolution in six scenarios. This benefits from the precise search with full simulations

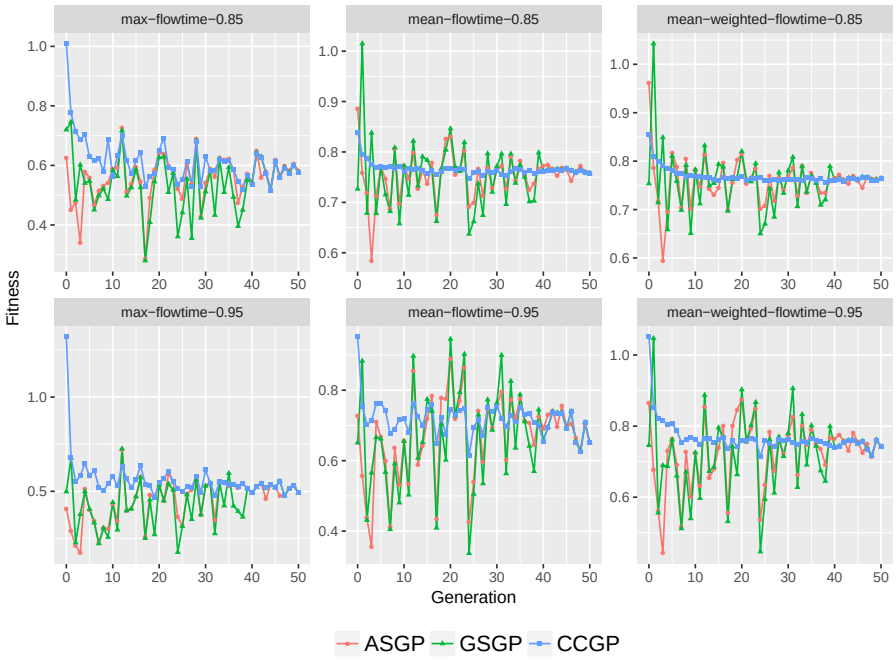

Fig. 1. The convergence curves of the fitness obtained by ASGP, GSGP and CCGP in training process.

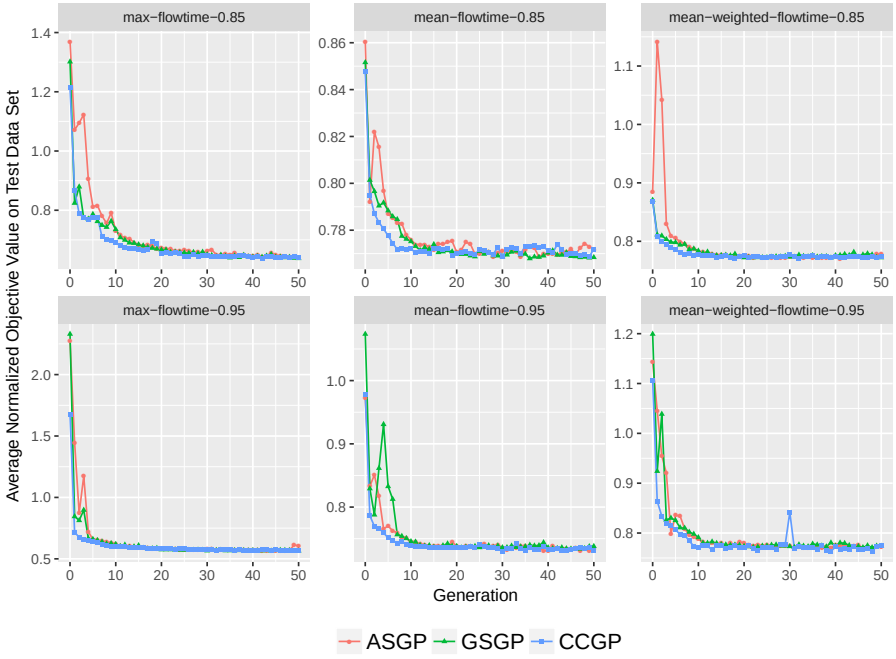

Fig. 2. The convergence curves of the normalized objective value obtained by ASGP, GSGP and CCGP in test process.

at the expense of computation time. However, after generation 10 approximately, the test performance between these three algorithms does not differ obviously.

Overall, taking the computation time and test performance into consideration, the proposed algorithms are more promising than CCGP.

5 Conclusions and Future Work

In order to tackle the intensive computation time of GP approach, this paper proposed two different kinds of strategies of surrogates for GP to automatically design dispatching rules for DFJSS. It is a preliminary attempt to apply surrogates into DFJSS. The results show that the two proposed surrogate strategies managed to reduce computation time without deteriorating the quality of the evolved rules. It also indicates that the proposed strategies have the potential to help GP to achieve more promising dispatching rules.

It is important to further investigate different strategies for surrogates to accelerate the effectiveness and efficiency of the GP approach. Function approximation and evolutionary approximation will be considered in the future.

References

1. Bhattacharya, M.: Reduced computation for evolutionary optimization in noisy environment. In: Proceedings of the 10th Annual Conference Companion on Genetic and Evolutionary Computation. pp. 2117–2122. ACM (2008)
2. Blackstone, J.H., Phillips, D.T., Hogg, G.L.: A state-of-the-art survey of dispatching rules for manufacturing job shop operations. The International Journal of Production Research **20**(1), 27–45 (1982)
3. Brucker, P., Schlie, R.: Job-shop scheduling with multi-purpose machines. Computing **45**(4), 369–375 (1990)
4. Haupt, R.: A survey of priority rule-based scheduling. Operations-Research-Spektrum **11**(1), 3–16 (1989)
5. 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)
6. Mei, Y., Nguyen, S., Zhang, M.: Evolving time-invariant dispatching rules in job shop scheduling with genetic programming. In: European Conference on Genetic Programming. pp. 147–163. Springer (2017)
7. Nguyen, S., Zhang, M., Johnston, M., Tan, K.C.: A computational study of representations in genetic programming to evolve dispatching rules for the job shop scheduling problem. IEEE Transactions on Evolutionary Computation **17**(5), 621–639 (2013)
8. Ratle, A.: Accelerating the convergence of evolutionary algorithms by fitness landscape approximation. In: International Conference on Parallel Problem Solving from Nature. pp. 87–96. Springer (1998)
9. Sels, V., Gheysen, N., Vanhoucke, M.: A comparison of priority rules for the job shop scheduling problem under different flow time- and tardiness-related objective functions. International Journal of Production Research **50**(15), 4255–4270 (2012)
10. Yska, D., Mei, Y., Zhang, M.: Genetic programming hyper-heuristic with cooperative coevolution for dynamic flexible job shop scheduling. In: European Conference on Genetic Programming. pp. 306–321. Springer (2018)