Machine Learning Approach Based on Gene Expression Programming for Dynamic Production Scheduling Problem

Li Nie, Xiaogang Wang, Yuewei Bai * and Xiaoyan Wu
School of Intelligent Manufacturing and Control Engineering, Shanghai Polytechnic University, Shanghai 201209, China

*Corresponding author e-mail: ywbai@ssp.edu.cn

Abstract. Dynamic Job shop scheduling problem (DJSSP) which involves the dynamic characteristic of job arriving over time is one of the most popular scheduling problems in manufacturing system. In the paper, a machine learning approach is proposed for DJSSP. The proposed machine learning approach employs gene expression programming (GEP) which introduces automatically defined functions (ADF) into the representation of chromosome to construct efficient scheduling rules (SRs) automatically for DJSSP. The proposed GEP-based machine learning approach is evaluated and the results show that the approach is effective.

1. Introduction
Production scheduling is regarded as one of the most important tasks carried out in smart manufacturing systems [1]. It is concerned with the allocation of scarce resources to activities with the objective of optimizing one or more performance measures, such as minimization of makespan, flowtime and so on. It is usually assumed that all jobs to be processed are available at the beginning of the whole planning horizon. However, in many real situations, jobs may arrive over time and their arrival dates cannot be predicted in advance due to transportation, cancellation of orders or machine breakdown etc. Scheduling problem with this feature is usually called dynamic scheduling problem [2].

Scheduling rules (SRs) play a core role in dynamic scheduling problem. However, it is very difficult to construct suitable rules because there are substantial inherent complexity and variability in the problems. Therefore, many researchers had made a considerable effort to develop machine learning approaches based on evolutionary algorithms to create problem specific SRs. For example, Jackobovic & Budin [2], Su et al. [3] and Pickardt et al. [4] proposed the approaches based on genetic programming (GP) for classical JSSP. However, in their work, dynamic feature of scheduling problems was not taken into consideration sufficiently and computation cost was still high. In this paper, a machine learning approach based on gene expression programming (GEP) [5] is proposed to solve dynamic job shop scheduling problem (DJSSP). In order to reinforce the flexibility of chromosomes representation the automatically defined functions (ADF) [6-7] is introduced into GEP.

The remainders of the paper are organized as follows. In Section 2, the definition of DJSSP is given. In Section 3, the machine learning approach based on GEP is elaborated. In Section 4 the computational results are presented. The final conclusions are given in Section 5.
2. DJSSP
There are \( n \) jobs arrive the shop floor over time and they need to be operated on \( m \) machines according with a predetermined sequence of operations which is independent of the other jobs. The constraints must be satisfied: (1) The jobs cannot be processed before their arrival (starting time constraint). (2) Each operation has to be processed on corresponding machine for an uninterrupted time period and no operation may be preempted (non-preemption constraint). (3) Each machine can process only one job and each job can be processed by only one machine at a time (capacity constraint). The scheduling objective is to determine the starting time for each operation in order to optimize a certain criterion with satisfying all the constraints mentioned above. In the paper, the objective of minimize the average flow time is considered.

\[
F_{\text{ave}} = \frac{1}{n} \sum_{i=1}^{n} (c_i - r_i)
\]

Where, \( c_i \) and \( r_i \) denote the completion time and arrive date of job \( i \), respectively. \( n \) denotes the number of jobs. \( F_{\text{ave}} \) denotes mean flow time of the \( n \) jobs.

3. GEP-based Machine Learning Approach

3.1. Flow of GEP
In the approach GEP automatically construct SRs for DJSSP which would yield good results considering given performance criteria. The flow of GEP is shown below. GEP starts with an initial population which consists of a number of chromosomes that are randomly generated. These individuals are evaluated according to a quantitative performance measure. The next population of chromosomes is formed through reproducing and modifying the current excellent individuals using evolutionary search operators such as selection with elitism strategy, replication, mutation, transposition and recombination. The next population of individuals is then evaluated again. This cycle is repeated until the termination condition is satisfied.

3.2. Representation of Chromosome

3.2.1. Encoding Scheme. The function set (FS) and terminal set (TS) which include all the elements that compose all the possible chromosomes are defined below. FS=\{+,-,*,/\}, where, + denotes addition, - denotes subtraction, * denotes multiplication, and / denotes protected division which returns 1 when the denominator is equal to 0. TS={\( r, d, k, o, s, p, e, w, q \)}, where \( r \) denotes arrival time of a job, \( d \) denotes due date of a job, \( k \) denotes remaining processing time of a job, \( o \) denotes number of remaining operations of a job, \( s \) denotes dynamic slack for a job, \( p \) denotes processing time of the current operation of a job, \( e \) denotes queue time of a job, \( w \) denotes waiting time of the machine for a job, \( q \) denotes total work content of jobs in the queue of the next operation of a job. Chromosomes are encoded according to following stipulations: (1) Each chromosome is composed of a conventional part and a control part. Each gene in the conventional part is used to encode a different ADF whereas gene in control part determinates the interaction between the ADFs. (2) Conventional part comprises one or more genes. Control part consists of only one gene. (3) As for the conventional part, the head of a gene may contain symbols from both FS and TS, whereas the tail consists only of symbols come from TS. As for control gene, the head may contain symbols from both FS and \{0, 1, ..., \( N_g \)-1\} (\( N_g \) is the number of gene in conventional part), whereas the tail consists only of symbols come from \{0, 1, ..., \( N_g \)-1\}. (4) The length of head and tail must satisfy the equation \( g=h \times (l-1)+1 \), where \( h \) and \( g \) is the length of head and tail, respectively, and \( l \) is the maximum number of arguments for all operations in FS. To exert these constraints on chromosomes arms to guarantee the legality of each individual in spite of a variety of genetic operators applied on them. A chromosome which is randomly generated with FS and TS defined above is illustrated in Fig. 1.
### 3.2.2. Decoding Scheme.

Decoding is the process transferring the chromosomes to the scheduling solutions. For the example chromosome shown in Fig. 1, Gene 0 in conventional part can be mapped into sub-ET 0 (shown in Fig. 2) following a depth-first fashion. Specifically, first element in gene 0 corresponds to the root of the sub-ET 0. Then, below each function is attached as many branches as there are arguments to that function. A branch of the sub-ET 0 stops growing when the last node in this branch is a terminal. In the same way, other genes of the conventional part can be mapped into sub-ET 1 and sub-ET 2, respectively (shown in Fig. 2). The control gene is also mapped into a control-ET shown in Fig. 2. The symbols 0, 1, 2 in the control-ET denote sub-ETs corresponding to gene 0, gene 1 and gene 2, respectively. The control-ET in fact determinates the interaction between the sub-ETs corresponding to each gene in conventional part and form a multi-subunit ET which is further interpreted into a SR with a mathematical equation (2).

\[
\frac{1}{p - (r + d - q)} \times (p - k / o)
\]  

(2)

**Figure 1.** Encoding scheme of GEP

**Figure 2.** Mapping between gene and ET

### 4. Experiments and Results

In this section, simulation experiments are conducted on several sets of problem instance which are generated in [8]. The proposed approach in the paper (denoted with NGEP) are compared with ten human made SRs, such as FIFO, SPT, EDD, MDD, SL, WINQ, PT + WINQ, PT + WINQ + AT, PT + WINQ + SL, PT + WINQ + AT + SL. And it is also compared with the previous version of GEP-based
approach (denoted with GEP). Table 1 lists the results of the comparison. The results indicate that NGEP is more effective than the human made rules and slightly better than GEP.

In order to further demonstrate the efficiency of NGEP, GP-based approaches [2] are used to be compared with NGEP. Fig 3 shows the results of the best individual in the training set. The graph shows that the results of GP are better than NGEP in earlier generations, but as the generation proceeds the results of NGEP outperform GP. Fig 4 shows the results of validation. It indicates that the results of GP are worse than that of NGEP. GP is over-fitting to the training data. However, NGEP can produce better results both in training and validation sets.

**Table 1.** Comparison between human made rules, GEP and NGEP.

| Rule            | b 1u 60% | b 1u 90% | b 3u 60% | b 3u 90% |
|-----------------|---------|---------|---------|---------|
| FIFO            | 717     | 1080    | 719     | 1096    |
| SPT             | 709     | 1188    | 716     | 1182    |
| EDD             | 713     | 1067    | 708     | 1016    |
| MDD             | 716     | 1019    | 707     | 1018    |
| SL              | 705     | 1048    | 691     | 1005    |
| WINQ            | 663     | 993     | 654     | 1007    |
| PT + WINQ       | 642     | 980     | 648     | 984     |
| PT + WINQ + AT  | 705     | 1038    | 700     | 1054    |
| PT + WINQ + SL  | 682     | 1008    | 653     | 900     |
| PT + WINQ + AT + SL | 706 | 1047 | 700 | 1052 |
| GEP             | 565     | 817     | 569     | 811     |
| NGEP            | 512     | 802     | 554     | 798     |

**Figure 3.** Result of best individual

**Figure 4.** Result of validation
5. Conclusion
This paper proposes a machine learning approach based on GEP for DJSSP. The encoding scheme of chromosome is an indirect encoding approach which increases the representation flexibility of chromosomes. What’s more, the decoding scheme transfer GEP chromosomes into scheduling solutions for scheduling problem instances. To verify the effectiveness of the presented GEP-based approach, computational experiments are performed. GEP-based approach is compared with ten classical human made SRs selected from literature and GP-based approach. The results show that GEP can automatically create efficient SRs which distinctly outperform the selected human made SRs and GP-created SRs. In the future research, we is going to study multi objective optimizing algorithm based on GEP to automatically create SRs which can generate scheduling solutions considering several objectives simultaneously and compromising between them.

Acknowledgments
This work was financially supported by the National Natural Science Foundation of China under Grant No. 51605273 and U1537110 and the National Key R&D Program of China under Grant No. YS2017YFGH000967.

References
[1] P. Zheng, H. Wang, Z. Sang, Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives, Frontiers of Mechanical Engineering 13(2018) 137-150.
[2] D. Jakobovic, L. Budin, Dynamic scheduling with genetic programming, Lect. Notes Comput. Sc. 3905 (2006) 73-84.
[3] N. Su, M. Zhang, M. Johnston, A Computational Study of Representations in Genetic Programming to Evolve Dispatching Rules for the Job Shop Scheduling Problem, IEEE T. Evolut. Compt. 17 (2013) 621-639.
[4] C.W. Pickardt, T. Hildebrandt, J. Branke, Evolutionary generation of dispatching rule sets for complex dynamic scheduling problems, Int. J. Prod. Econ. 145 (2013) 67-77.
[5] C. Ferreira, Gene expression programming: a new adaptive algorithm for solving problems, Complex Syst. 13 (2001) 87–129.
[6] C. Ferreira, Automatically Defined Functions in Gene Expression Programming, In: Nedjah N., Mourelle L.M., Abraham A. (eds) Genetic Systems Programming. Studies in Computational Intelligence, vol 13. Springer, Berlin, Heidelberg, 2006.
[7] Y. Jiang, L. He, Q. Liu, Research of Gene Expression Programming Algorithm Based on Cusp Cloud Model, Journal of Southwest Minzu University (Natural Science Edition), 44 (2018) 291-296.
[8] L. Nie, L. Gao, P.G. Li, L.P. Zhang, Application of gene expression programming on dynamic job shop scheduling problem, In: Proceedings of the 15th International Conference on Computer Supported Cooperative Work in Design CSCWD 2011, pp. 291-295.