A REVIEW OF GENETIC PROGRAMMING FOR DYNAMIC FLEXIBLE JOB SHOP SCHEDULING

by

Gaofeng Shi

A thesis
submitted to the Victoria University of Wellington
in fulfilment of the
requirements for the degree of
Master of Science
in Computer Science.

Victoria University of Wellington
2022

Abstract

Job shop scheduling (JSS) is the challenging combinatorial optimisation problem which has been studied for many years. It is to schedule a number of jobs on a number of machines where the scheduling is the set of processing sequences for each of the machines. The objective of JSS problem is minimize the make-span, total flowtime, or total weighted tardiness of job shop. Another extension situation of JSS is the flexible job shop scheduling (FJSS) that allows a job operation which can be processed by any machine if the candidate machines are idle. In addition, FJSS problems are not only for searching the sequencing of operations, but also need to find the routing assignment of operations to the suitable machines. However, in the real world the situation of Job shop scheduling might not always occur under static conditions such as stochastic job arrival, uncertain processing time, and unexpected machines breakdown. In particular, this thesis focuses on the $dynamic$ flexible job shop scheduling (DFJSS) problem which is more complicated under the uncertain complex environment.

As the dynamic flexible job shop scheduling (DFJSS) problems don't have the prior information for scheduling jobs in advance, dispatching rule becomes the best solution for this problem. The dispatching rule makes the decision which job is assigned to a machine when the machine become idle. It can react to unexpected change quickly, particularly for solving the dynamic and stochastic problem.

Design effective dispatching rule for DFJSS problem is the main stream study which has been investigated by both academics and industry experts widely because the dispatching rule has various properties such as simplicity, interpretability, low computational cost. However, dispatching rules need to be redesigned over time to suit the dynamic situation

of DFJSS in the real world scenario. As heuristics are the effective designation for the specific and class problems, which is to search for solution space of problem optimization solutions. Hyper-Heuristic is searching for heuristics in search space instead of directly searching solutions of the problem. It becomes the best approach for designing DFJSS dispatching rules. In particular, many genetic programming hyper-heuristic (GPHH) approaches have been proposed to generate the effective dispatching rules for DFJSS problems. It is worthwhile to study and review those previous studies to get better understanding to prepare for further research.

The goal is to study and review the concepts of evolutionary computation into genetic programming and the various approaches of GPHH for the dynamic flexible job shop scheduling (DFJSS) problem. Firstly to study the background of the evolutionary computation of GP and the problem of DFJSS. Then, it will learn some recently related research of GPHH for DFJSS. Finally, by the understanding of previous work and studies to find the direction for future research with GP for DFJSS.

Acknowledgments

I would like to thank my supervisors, Dr. Yi Mei, Dr. Fangfang Zhang for providing me with constant support and encouragement during my Master's research. The feedback they provided on my research has been invaluable for my learning experiences as a Master's student and helped me improve various skills such as my learning and analytical skills.

Contents

Chapter 1

Introduction

1.1 Problem Statement

Job shop Scheduling (JSS) problems are the computational complicated optimization well know NP-hard problems [16] in the real world. JSS problems can be explained as there are set of jobs and machines denoted by N and M respectively. Each job consists of a sequence of operations P that each operation has the release time and due date and has to be assigned to one of the eligible machines. The constraints of JSS are that all of the job's operations must be in order that one operation only can be processed on one machine, and one machine can only process one operation at a time and can't be interrupted. The scheduling objective of the job shop is to minimize the make-span, the total flowtime, and the total weighted tardiness [23][24].

The extension version of JSS can be categorised to flexible job shop scheduling (FJSS) and dynamic flexible job shop scheduling (DFJSS). The FJSS problem allows the job operation to be processed by any machine if the candidate machines are idle. It makes the problem more complicated comparing with JSS. The scheduling of JSS only requires a sequencing of operations on suitable machines, while FJSS scheduling not only searches for sequencing of operations, but also has to find the routing of operations

to the idle machines [8]. The good solution for FJSS can be searching for heuristic algorithms such as Truncated exponential schemes, Greedy algorithms, Simulated Annealing and Tabu search [3]. Dispatching rules are a distributed sequencing strategy which can assign the priority of job to the machine and also make routing of operations on the machine. There are numbers of dispatching rules developed for FJSS, for example. the shortest Processing time (SPT), and the earliest due date (EDD) [3].

However, the real world of environment of Job Shop scheduling is not static. The job arrives randomly, the machine breakdown, and uncertain job operation processing time which makes the job shop in dynamic situation. Dynamic flexible job shop scheduling (DFJSS) does not have the prior information for scheduling job in advance. As the dispatching rules reacting to the uncertain change quickly, solving DFJSS problem is to study designing the better dispatching rule [5]. Hyper-heuristic is to search for heuristics in search space instead of searching solutions of problem directly [6]. It generates the heuristics as dispatching rules for DFJSS solution that can feasibly adopt the job shop scheduling dynamic environment changing automatically and quickly [5].

Genetic Programming is an evolutionary computation approach which can evolve the represented individual solution automatically by its evolutionary process. The tree structured individual representations of GP are easily encoded to dispatching rule for DFJSS problem. The priority function of dispatching rules of JSS can be represented by GP tree easily as its mathematical expression. The GP evolution operation crossover and mutation evolve the individuals from selected better parents by fitness objective value. Both exploration and exploitation capacity of GP that gives it large heuristic search space to discover unknown and effective dispatching rules. Employing Genetic programming with hyper-heuristic (GPHH) to discover the solution of DFJSS becomes the popular domain study area which attracted many researchers.

Much research has achieved successful performance of solution by im-

plemented GPHH for DFJSS problem in recently. It is worthy to learning and review their studies to obtain the experience and interesting thought for further research.

1.2 Related studies

Dispatching rules in Dynamic Flexible Job Shop scheduling (DFJSS) divided into two parts sequencing and routing rules which can be generated by Genetic Programming with Hyper-heuristic (GPHH) effective and efficiently from cooperative coevolution perspective. Multi-objective cooperative coevolution GPHH for DFJSS has achieved a very high performance result [26][50]. In addition, the single objective GPHH cooperative coevolution method was developed for DFJSS by Yska et al.[39], and the more advance study have proposed by Zhang et al. [43].

The most expensive part of GP computation is long time fitness evaluation. Surrogate model is manipulated to reduce the computation time of GPHH for DFJSS [17][29][49][46]. On the other hand, GPHH with multitask approach for DFJSS has improved the efficiency of algorithm performance dynamically [40][41][45].

Feature selection and construction are the important technical tasks in machine learning. GPHH with feature selection method to achieve the better accuracy and reduce the cost of computation is another direction of research for DFJSS. There are some interesting studies about feature selection of GPHH for DFJSS which will be introduced [24][23][47][44][13]. Meanwhile, Yska et al. [38] proposed a novel feature construction method with GPHH for job shop scheduling problem.

The goal of this thesis is to obtain the professional knowledge and better understanding from recent research studies about how to solve the dynamic flexible job shop scheduling (DFJSS) problem by genetic programming hyper-heuristic (GPHH) methods.

1.3 Further Direction

The main focus of this review is to study and analyse the related previous work from the background information of GP to hyper-heuristic leading edge algorithms for DFJSS, which will summarise the overall challenge and key issues to guide the interesting further research area in job shop scheduling problems.

1.4 Organisation

The rest of this thesis is organised as follows: Chapter II gives the background information, including the problem description and genetic programming explanation. Then,the related work of genetic programming with hyper-heuristic for dynamic flexible job shop scheduling problems studies is learned and analysed in Chapter III. Issues and challenges are discussed in Chapter IV, Chapter V gives the conclusions and future work.

Chapter 2

Background

This Chapter describes the review of background literature on the thesis topic Genetic Programming for Dynamic Job Shop Scheduling problem related concepts and research. First, section 1 will introduce the basic concepts. Job shop scheduling problem will be explained at section 2. The last section of this chapter will introduce genetic programming.

2.1 Basic Concepts

This section will focus on the basic concepts related to the main research area. It will explain some definitions about Machine Learning, Scheduling, Routing and Sequencing, Hyper-Heuristics, and Evolutionary Computation.

2.1.1 Machine Learning

Machine Learning is part of artificial intelligence. It applies example data or past experience programming computers to find solutions of problems, or optimize a performance criterion [12]. Machine learning can build the model with defined parameters, using training data or past experience to learn and optimize the parameters of the models. The model may be

predictive for predictions, or descriptive to obtain the knowledge from the data, or both [12]. Machine Learning can be divided broadly into three categories which are $SupervisedLearning$, $UnsupervisedLearning$, and $ReinforcementLearning$.

Supervised Learning

The machine learning task or model is learnt from maps of input to output which are from the training data or example of input-output pairs [12]. The task of supervised learning can be explained this way. Given a training set of N instances, the pairs input-output $(x_1, y_1), (x_2, y_2), ...(x_N, y_N)$ of each y_j can be generated by an unknown $y = f(x)$. The task explores a h that approximates the true function f [34]. In the task, if the output y are categorical value, the supervised learning will be classification learning problem. Meanwhile, if the y are numerical value it will be regression problem. [34].

Unsupervised Learning

The machine learning algorithm or model which learns patterns or structures of data from unlabeled training instances [12]. Clustering is the most common unsupervised machine learning task that detects potentially useful clusters of input instances[34]. Clustering method called density estimation which is to find clusters or grouping of input. For example, the company allocates their customers to different natural groupings by the past transactions of customers with the clustering model [12].

Reinforcement Learning

The machine learning agent learns what to do and how to associate situations to actions in order to get maximum rewards and less punishments. The agent discovers which actions yield the most reward when trying it,

but is not told which actions to take [36]. The actions may not only generate reward immediately and also the next situation, through that, all subsequent rewards. A learning agent has to sense the state of its environment to some extent and has to take actions which affect the state, and also should have goal relating to environment state [36]. Reinforcement learning is different with supervised learning, the former learned from its own experience and the latter learned from training set of labeled examples. Reinforcement learning also is different with unsupervised learning which is typically about finding pattern hidden in unlabeled data [36]. In addition, applying Reinforcement Learning approach is another interesting popular topic to solve Dynamic Job Shop Scheduling problems.

2.1.2 Scheduling

Scheduling is a type of process for decision-making which becomes a crucial role in manufacturing and modern industries [32]. It allocates the resources to organize the given time periods of tasks and its goal is to make the optimization of objectives. Nowadays, the resources and tasks can be utilized by many different forms in an organization. For instance, the resources can be workshop machines, airport runways, hospital staff, computing process unites, and so on. The tasks may be a production operation process, airplane take-off and landing, the shift of staff, programming executions, and so on [32]. Scheduling the resources and tasks to achieve the objective goals which can be seen as searching the solution of job shop scheduling problem.

2.1.3 Routing and Sequencing

In Job shop scheduling problem (JSS) the scheduling is to plan a set of processing sequences for each of the machines which has a prior assignment of operations to machines. Flexible Job Shop Scheduling (FJSS) problem

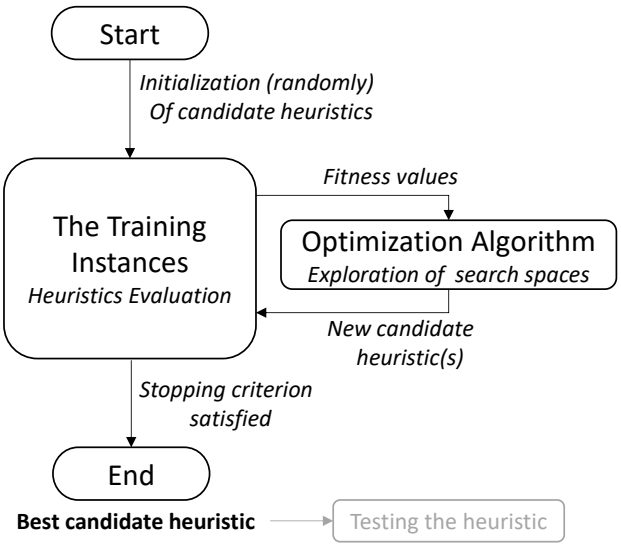

Figure 2.1: The procedure of a hyper-heuristic for heuristics generation [5]

don't have the prior assignment to fix the machines. For each job need to plan a sequence of operations, and for each operation there are a set of equivalent machines can be process the operation with possibly different times. The routing is to assign each operation to a machine, and the sequencing is to assign operations on each machine. Routing and sequencing rule must be solved simultaneously in order to obtain the feasible schedule minimizing a given objective function [3]. The objective function depended on the different environment of manufacturing and industry. For example, it might be the minimization of make-span, flowtime, and tardiness in job shop scheduling.

2.1.4 Hyper-Heuristic

Heuristics definition in search strategy is the method which is applied in order to search a solution space for an optimization of problem solution [30]. Hyper-Heuristic is searching for heuristics in search space instead of

directly searching solutions of problem [6].
Fig. 2.1 shows the basic illustration of the procedure of a hyper-heuristic for the heuristics generation. Encoding or representation of candidate heuristics defined the search space that optimization algorithm explores the search space depended on the fitness value as the quality of candidate heuristic, and recreated new candidate solutions during the iterations until the stopping criterion is achieved[5].
Many real world combinatorial complex problems are implemented successfully by the hyper-heuristics approach such as bin-packing, personnel scheduling, timetabling, production scheduling, and vehicle routing problems[6]. Applied hyper-heuristic search method to find the suitable dispatching rule which can be the best way of solution for dynamic flexible job shop scheduling problem. Because hyper-heuristic can generate heuristics which has the feasibility to adopt the dynamic job shop scheduling environment change automatically and quickly [5].

2.1.5 Evolutionary Computation

Evolutionary computation is a group of algorithms for artificial intelligence deep machine learning optimization which is inspired by biological evolution. Evolutionary computation can be broadly divided to two subcategory which are evolutionary algorithms (EAs) and swarm intelligence (SI).

Evolutionary algorithms

Evolutionary algorithms (EAs) have the common method that started with initialization. The initialization of the population randomly creates the individuals which are the set of candidate solutions. After the creation of the population, the standard of EAs algorithm simulating as natural evolution has three processes to produce the new generation of population for each iteration. Selection based on the value of objective function from

each individual choose the parents to produce the offspring. Breeding is to create the offspring from selected parents by two common operations crossover and mutation, which will create a new generation of population. Fitness function evaluate the objective value of individual which is setup for the measurement of solution. Those three process can be iterated until the best solution is found or a preset of generation limit is reached [11][9]. Meanwhile, there are four main algorithms classified from EAs which are genetic algorithm (GA), genetic algorithm (GP), evolutionary programming (EP), and evolution strategies (ES) [11].

Swarm intelligence

Swarm intelligence, which is another sub area of evolutionary computation, is common with the designed algorithm by using intelligent multi-agent system by inspiration from collective behavior of social insects such as ants, bees, and termites, as well as from other animal behavior such as flocks of birds or schools of fish [2]. They are different from EAs simulating on evolution of generations, SI applies the artificial agents searching through some mathematical space [9]. SI has two main concepts algorithms one is called Particle swarm optimization (PSO) established by Kennedy and Eberhart in 1995, another is Ant Colony Optimization (ACO) introduced by Dorigo et al. in 1996 [11].

2.2 Job shop Scheduling

This section covers the concepts of job shop scheduling (JSS) problem. From the degree of the difficulty, there are three levels of the category into those problems. They will describe separately by following part:.

2.2.1 Job Shop Scheduling

The Job shop scheduling Problem (JSS) is the real world complected optimization problem which belongs to the NP-hard problems [16]. In a job shop scheduling (JSS) problem, there are a number of jobs $J = \{J_1, J_2, ..., J_n\}$ and machines $M = \{M_1, M_2, ..., M_m\}$. For each job J_i it consists of a sequence of operations $O_i = \{O_{i1}, O_{i2}, ..., O_{ij}\}$, as well as an release time $t_0(J_i)$ and a due date $d(J_i)$. Each operation O_{ij} must be allocated to an eligible machine $m(O_{ij}) \in M$.
Job shop scheduling has to follow the constraints:

- Each job predefined the operations in order, and one operation O_{ij} cannot start processing until all its precedent operations are completed.
- Each operation O_{ij} can be processed only by one of its candidate machines $m(O_{ij})$.
- Each machine can process only one operation at same time.
- The scheduling is non-preemptive, i.e., once started, the processing of an operation cannot be stopped or paused until it is finished.

A schedule X in JSS, which will be represented as a set of processing sequences for each of the machines. The processing sequence $\mathbf{P}(M_k)$ consists the processing of all the operations $\{O_{ij}|m(O_{ij}) = M_k\}$, and to be with the starting time $\tau(O_{ij})$. The objectives of JSS commonly includes the minimize of make-span (C_{max}), the total flowtime (ΣC_j), and the total weighted tardiness $(\Sigma w_j T_j)$ [24].

2.2.2 Flexible job shop scheduling

The flexible job shop scheduling (FJSS) problem is an extension of the JSS problem. The flexible job shop scheduling allows a job operation can be

processed by any machine if the candidate machines are in idle. In a flexible job shop scheduling (FJSS), there is a set of jobs $J = \{J_1, J_2, ..., J_n\}$ to be processed on set of machines $M = \{M_1, M_2, ..., M_m\}$. The job consists of a sequence of operations $O_i = \{O_{i1}, O_{i2}, ..., O_{ij}\}$. Each the operation O_{ij} of J_i has to be operated on an available machine $m(O_{ij}) \in M$. All jobs and machines are available at time 0. The following constraints is for FJSS:

- Each job predefined the operations in order, and one operation O_{ij} can only start processing when all its precedent operations completed.
- Each operation O_{ij} can be processed only by one of machines $m(O_{ij})$.
- Each machine can process only one operation at same time.
- It is non-preemptive operation, i.e., once start, the processing of an operation cannot be interrupted until it is completed.

The JSS requires a sequencing of operations on suitable machines, while in the FJSS the allocation of an operation is not fixed in advance and can thus be processed on a set of capable machines. Hence, in FJSS problem, there are not only for searching the sequencing of operations, and also need to deal with the routing assignment of operations to suitable machines[8]. The problems of FJSS is to allocate each operation to an idle machine (routing problem), and to sequence the operations on the machine (sequencing problem)[31]. The objectives of FJSS can be same as JSS which are minimize of make-span (C_{max}), the total flowtime (ΣC_j), or the total weighted tardiness $(\Sigma w_j T_j)$.

2.2.3 Dynamic flexible job shop scheduling

JSS and FJSS can be see as static scheduling problems. Although, The flexible job shop scheduling (FJSS) problems as we know belongs to the category of NP-hard problems[10]. All jobs information is known before the implementation of the system in JSS and FJSS environment. In the

real world the situation of Job shop scheduling might not always be under static conditions such as stochastic job arrivals, uncertain processing times, and unexpected machines breakdowns. This uncertain environment of flexible job shop scheduling is called Dynamic Flexible Job Shop Scheduling (DFJSS) problem. The problem of DFJSS don't have the prior information for scheduling job in advance. A research for DFJSS is to analyse and study the different levels influence of the parameters from machine breakdown [18]. However, the Dynamic flexible job shop scheduling has various properties for the problem consideration. Because dispatching rules can react to unexpected change quickly, particularly for solving dynamic and stochastic problems [5]. Therefore, Dispatching rules study become the main stream for DFJSS. As heuristics are the effective designation for the specific and class problems, hyper-heuristic is the best approach for designing DFJSS dispatching rules.

2.3 Genetic Programming

Genetic Programming (GP) is an Evolutionary Computation (EC) approach inspired by natural selection into the evolutionary process of artificial computing system[20]. Genetic programming started from the initialization of the population. Each of the individuals from the population represents a solution of the problem. The population will be evolved by the GP iteration process: selection, crossover, and mutation. The selection process based on the objective value which was evaluated by the fitness function. The termination of iteration can be completed when the GP achieved the goal of criteria or meet the setup of generation number. The individual of GP is tree structure that contains terminals as leaf nodes of the tree, and functions referred as nodes of the tree.

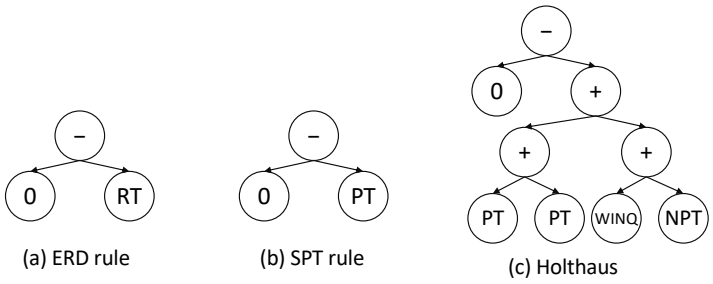

Figure 2.2: Dispatching rule heuristics represented by GP tree. [4]

2.3.1 Representations

GP tree is the representation of GP. The individual of GP population contains two parts composed by terminals and functions which appropriate to the problem domain [20]. The set of terminals applied typically comprises appropriate to the problem domain and numeric constants [20]. For example, in the dispatching rule of JSS, release time of a job (RT), current operation processing time (PT), work content in next queue(WINQ), next processing time (NPT), and constant 0 which are used as the terminals representing the current time decision of job [4][25]. Fig. 2.2 shows benchmark dispatching rule heuristics of JSS represented by tree-based GP. Earliest Release Data first (ERD) means the priority of processing the job order is calculated by $-RT = (0 - RT)$. Shortest Processing Time First (SPT) represented by $-PT = (0 - PT)$, and Holthaus rule use $-(2PT + WINQ + NPT)$[4]. The set of functions can be used by arithmetic operations $(+, -, \times, \div)$, mathematical functions, conditional logical operations, and domain-specific functions [20]. The initialization process randomly generate individuals of population by compositions of available functions and terminals. However, the sufficient set of primitive function should be well defined for any combination of arguments with every terminals their value may encounter, which terminals and functions satisfies the closure properties [20].

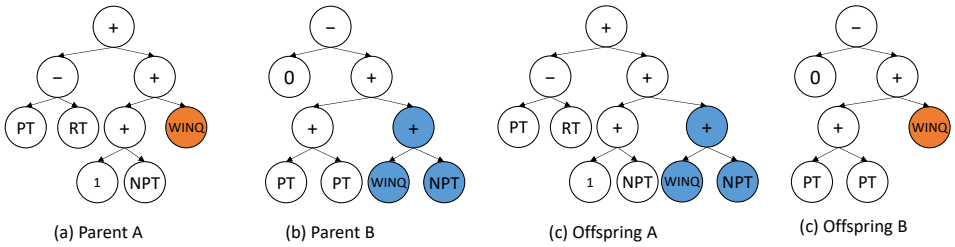

Figure 2.3: GP crossover operation

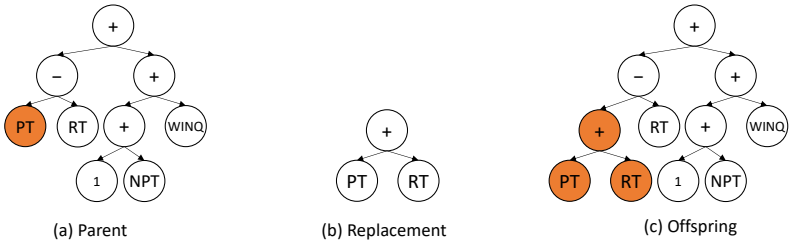

Figure 2.4: GP mutation operation

2.3.2 GP genetic operations

For the GP tree based representation, which evolve the new generation of individual by genetic operations during the GP iterations. Subtree crossover and subtree mutation are the commonly genetic operation in GP to search the new solutions by exploration and exploitation [25]. The subtree crossover operation reproduces new individuals by randomly combining subtrees, which are the parts of subtree from selected two parents. Fig. 2.3 shows the selected parents produce the offspring by GP crossover operation. In addition, the operation subtree mutation creates the new individual by modification of chosen individual, which by selecting a node from the individual and replacing by randomly-generated subtree from the root of that selected node[25]. Fig. 2.4 shows the mutation operation of GP. The crossover operation can be seem as a local search that will explore the best solutions of the generation, and the mutation operation seem as global

search which will take a exploitation the new solutions.

2.3.3 Fitness Function

Fitness function in GP is either to measure or compare in terms of how well each individual solution in the population which performs the particular problem environment [20] [21]. Fitness naturally is measured by the error calculated by the computer program which error closer to zero is the better one [20]. In real practice of many problems, fitness cases may represent different values of the program inputs, different system initial conditions, or different environments [21]. For instance, the design of new scheduling heuristics may contains multiple conflicting fitness objectives such as the three objectives of dispatching rules of DFJSS (makespan, mean tardiness, and flowtime) [25]. Some GP program in multiple fitness cases that its fitness is measured as sum, average, or weighted over the variety representative of different situations [20]. However, Nguyen et al.[25] suggests if there are no prior knowledge available, weighted aggregated fitness objective is not good approach for multi-objective design problem, for example GP design the dispatching rules for DFJSS. A multi-objective genetic programming based hyper-heuristic approach was developed by Nguyen et al.[27]

2.3.4 Genetic Programming Hyper-Heuristic for DFJSS

Genetic Programming is fixed length encoding representation which is very suitable with unknown the length of optimal heuristics in the given problem domain[6]. DFJSS dispatching rule can be represented by GP tree structure. Terminal set of GP tree can be easily represented from DFJSS features. Function set of GP tree will be relevant to dispatching rule of DFJSS. GP tree has the fixed length structure which is suitable to encode different scheduling rules as its flexibility [25] [28]. Genetic programming is particularly suitable for dynamic job shop scheduling problem. Although,

before applying GP for DFJSS, many research show the dispatching rule was hard to cope with multiple objective dynamic job shop situation [1] [33] [35] . Genetic programming based hyper-heuristic (GPHHs) has the advantage to design heuristics as their search algorithms are very flexible and many advanced apparochs have been developed to cope with multiple objectives of DFJSS [25]. The GP tree can easily represented priority function of dispatching rules by it's mathematical expression [19], and GP has large heuristic search space to discover unknown and effective dispatching rules [25]. Moreover, sophisticated dispatching rules can be encoded and evolved by GP individuals which has the flexibility of representations, and the advantages of available EC search mechanisms makes GP can enhance the dispatching rules quality of DFJSS [25].

2.4 Summary

This chapter has introduced the fundamental concepts of machine learning, heuristics and hyper-heuristic and evolutionary computation. Job shop scheduling problem was explained by different level space of solution. The last part described genetic programming which will apply to build the solution with hyper-heuristic for dynamic flexible job shop scheduling problems.

Chapter 3

Related Work of GP for DFJSS

This Chapter introduces the recently related studies of Dynamic Flexible Job Shop Scheduling (DFJSS) problem by various approach with Genetic Programming algorithm. The first, section 1 will learn multi-objective cooperative coevolution method with GPHH for DFJSS, and following section 2 introduce the single objective correlation coevolution approach. Surrogate model with GPHH can make the approach more efficient for DFJSS will be studied at section 3. Section 4 will describe the multitask GPHH for DFJSS which has improved performance significantly in different scenario of DFJSS. The last section 4 will explain the feature selection in GPHH which increased efficiently and interpretability of dispatch rules of DFJSS. Table. 3.1 provides a summary of the related work of GPHH for DFJSS that will review in this chapter.

Table 3.1: The Related Work of GPHH for DFJSS Addressed by Approach

Approach	References
Multi-objective Coevolution	[26], [50]
Cooperation Coevolution	[39], [43]
Surrogate Model	[17], [29], [46], [49]
Multitask Method	[40], [41], [45]
Feature Selection Method	[13], [23], [24], [44], [47].

3.1 Multi-objective Genetic Programming

3.1.1 multi-objective cooperative coevolution (DMOCC)

Nguyen et al.[26] developed a novel genetic programming based hyper-heuristic (GPHH) for DFJSS in 2012, The method was called multi-objective cooperative coevolution (DMOCC) method, which was the first developer applying the GP to solve dispartching rules (DRs) and due-data assignment rules (DDARs) simultaneously. In their study, the GP representations divided to two sub-populations DDARs and DRs by multi-objective GP methods. The individuals of DDARs and DRs use different terminals. DDAR determine flowtimes/due-dates more focusing on detailed operation information which assign a due-date to the job when the new job arrived. DR is a priority function which use different pieces of information from job and machines which applied for the jobs in the queue of the machine when the machine becomes idle. Those scheduling policies are combined the representative individuals by two sub-population.

Nguyen at al.[26] proposed the novel approach DMOCC employed cooperative coevolution method to evolve two decision rules combined to two sub-populations as scheduling policies. They store the non-dominated scheduling policies to an external archive for the selection of GP individuals, which was cooperative genetic operators between the two sub-population based on the crowding distance and the non-dominated rank of individuals.

Their experimental results of DMOCC evolved scheduling policies are much better on the Pareto front multi-objective method as compared to NSGA-II and SPEA2, and also it reduced the complexity of evolving sophisticated scheduling policies by multiple scheduling decisions evolved from different sub-populations. This method can be modified to reduce the computational time by take the advantage of the parallelisation techniques [26].

3.1.2 Hyper-Heuristic Coevolution Multi-Objective GP

Another research paper about multi objective GP for DFJSS is Hyper-Heuristic Coevolution for Multi-objective Dynamic flexible job shop scheduling which was proposed by Zhou and Yang et al.[50]. They employ three different hyper-heuristic coevolution methods with multi objective GP approach to evolve dispatching rule for dynamic flexible job shop scheduling problem. The first method called MO-CCGP which initializes two sub-populations of individuals each two types of individual GP trees represents sequencing rule and routing rule respectively. MO-TTGP is the second method which starting one population its individual contains two decision rules by two GP trees. The last MO-GEP method employs genetic expression programming (GEP) [14] individual consisting two chromosomes for the two decision rules. Those three coevolution method all employed with multi-objective approach NSGAII and SPEA2 [50].

The final performance of those methods shows the best methods is MO-CCGP with NSGAII approach refer to Pareto front value [50]. On the other hand, some studies focus on single objective GP hyper-heuristic coevolution GP for DFJSS [39][43].

3.2 Cooperation Coevolution GP

Cooperation Coevolution with Genetic programming hyper-heuristic for Dynamic flexible job shop scheduling is the first piece of work to co-evolving routing and sequencing rule simultaneously[39]. The algorithm started initialization with two populations representing routing rule and sequencing rule respectively. Then the routing rules and sequencing rules evolved separately by the GP process operator crossover, mutation and reproduction at each generation. The new generated rule evaluated by evaluate function to get fitness value. The best individual is in the final generation of each population the minimum fitness value rules [39].

Zhang et al.[43] proposed correlation coefficient-based recombinative guidance with GPHH for DFJSS is based on cooperation coevolution GPHH [39] to guide GP process genetic operators help GPHH find better scheduling heuristics of DFJSS. This algorithm finds the more useful subtrees of rule tree to guide the crossover bias better subtrees to be swiped and inherited. The difference of this proposed approach comparing with the traditional cooperation coevolution GPHH is there are two extra components: calculation of the importance of subtrees and crossover with recombinative guidance. The calculation of importance of subtrees based on Spearman correlation coefficient between subtrees and the rule tree which is the correlation of decision made by the subtree and decision made by whole tree. The recombinative guidance crossover selecting the important subtree instead of selecting the unimportant subtree will depend on the subtrees correlation value, which higher value will be high probability to be chose [43].

3.3 Surrogate Genetic Programming

The most time-consuming part of Genetic programming as an evolutionary algorithm is the fitness evaluation. The surrogate function was developed in order to reduce the cost of GP fitness evaluation process. The concept of surrogate model aims to minimize the number of expensive fitness evaluation in preference to the computationally cheaper function as far as possible[17]. There are couple of research about the implementation of surrogate model into GP of DFJSS.

3.3.1 GP employs surrogate function

Hildebrandt and Branke [17] first introduced GP surrogate function called phenotypic characterization solution for DFJSS. Their surrogate function base on GP phenotypic characterization to estimate the fitness value of

new individuals which has much lower computational cost than the full simulation evaluation. The surrogate function uses the nearest neighbor regression to predict the fitness of a new individual. All the individuals will calculate their Euclidean distance between phenotypic characterization in the database. The most similar fitness of the individual will return the fitness estimate to new individual. The database contains the full fitness evaluations are the last two generations which applied for the nearest neighbor surrogate model. As applying the surrogate function in GP by measure distance on tree structure representation is much more difficult than GA's genotypic distance metric. Their model computes the phenotypic characterization using a reference rule and a set of decision situation.

The result of this surrogate function GP approach for DFJSS has reduced the fitness evaluations cost up to 80% compare to the standard GP model[17]. Also their work shows the phenotypic characterization surrogate model achieved much more accuracy than the model based on genotypic distance.

3.3.2 simplified surrogate assisted GP

The simplified surrogate assisted GP devoloped by Nguyen et al.[29] aims to improve the quality of the dispatching rules of DFJSS without the huge computational costs. This simplified surrogate GP model has improved some aspects of limitation in previous approach [17]. For instance, if the dimension of decision vectors doesn't contain enough good differentiating evolved rules, it might not cope some complex situation of DFJSS. Also it will not be suitable with some scheduling decisions such as due data assignment, routing, and order release [29].

Nguyen et al.[29] proposed surrogate assisted GP (SGP) based on the simplified simulation model which potentially reduced the computational cost of fitness estimation. The simplified model of simulation which is by reducing the warmup and running time of the simulation or reducing the

number of machines and the number of operation per job. Nguyen et al.[29] found the most suitable simplified model is HalfShop which only half the scale of the original shop.

The process of SGP is the same normal genetic programming mainly that through the population initialisation, fitness evaluation, update best individual of GP run, and termination criteria archived to stop or iteration by apply genetic operation. The difference in SGP, the genetic operations create an intermediate population which is a large size than the original population in order to improve the chance to discover better results by increasing the diversity of population. SGP has three fitness functions:

1. $f(\Delta_i)$ the real fitness of evolved rules Δ in DFJSS .

2. $f'(\Delta_i)$ the estimated fitness obtained with the simplified model S.

3. $f_g(\Delta_i)$ the performance of rule Δ_i in a particular generation (intermediate population with a specific replication π).

As applying the large number of replications to obtain $f(\Delta_i)$ is very expensive, SGP apply one replication per generation to evaluate the quality of evolved rules $f_g(\Delta_i)$. Only the best $f_g(\Delta_i)$ will fully evaluate to obtain $f(\Delta_i)$. The newly generated rules fitness evaluated faster by applying the simplified model S for the intermediate population generated based on the generation fitness $f_g(\Delta_i)$.

The fitness $f'(\Delta_i)$ helps the SGP produce more potential rules by roughly determining rules quality, and fitness $f_g(\Delta_i)$ improve the diversity of SGP by discovering the most potential rules for full evaluations. In addition, the intermediate population is k times larger than the original population which increases the diversity of GP and has more chance to find the better rules.

3.3.3 Surrogate-assisted cooperative coevolution GP

The study cooperative coevolution GP with surrogate assisted model GP for DFJSS proposed by Zhou et al. [49] was to investigate and analyse the appropriate parameters of fitness evaluation of GP model. The study introduced three types coevolution GP representation tree: two sub-populations for dispatching rules routing and sequencing separated by two individual trees called CCGP, one population two parts of dispatching rule into one individual tree named TTGP, and GEP same as TTGP by sub-chromosome types tree. Those three coevolution GP employed with different configuration of surrogate-assisted simulation model to generate the DFJSS dispatching rule, CCGP with surrogate assisted model testing result is the highest performance than the other two [49].

3.3.4 Adaptive surrogate GP

Zhang et al.[46] proposed a surrogate-assisted GP for DFJSS called adaptive surrogate genetic programming ASGP. The main idea of this algorithm is to build a very simple surrogate model at the GP early generation then deliberately increase the surrogate models to enlarge the accuracy purpose. It is different with the stable surrogate model at the whole GP process such as HalfShop model [29]. The surrogate model at simulation jobs number and warmup jobs number are using formula:

$$Number = Number * (generation/Generation_{max})$$

That will increase simulation jobs and warmup number linearly which the surrogates range from low-fidelity to high-fidelity increase following the generation increase. [46].

Collaborative multi-fidelity surrogate GP for DFJSS introduced by Zhang et al.[42] is a novel multi-fidelity based surrogate-assisted genetic programming. It has a different method with paper[46] for surrogate model which is k designed surrogate models wtih different fidelities. The k dif-

ferent surrogates model for k multiple subpopulations will be evaluated respectively, while during the evaluation process each subpopulation will share knowledge by the particular knowledge transfer algorithm to produce next generation. The key point of the approach is to transfer the knowledge between different fidelity levels of surrogate model. The subpopulation with lower fidelity (simpler problem) easily search promising individuals, and learning knowledge from the high fidelity model to improve its quality quickly. Meanwhile, the subpopulation with higher fidelity (complex problem) will speed up the convergence as well. The transfer algorithm applies to find the promising individuals for knowledge transfer which the individuals to be selected with smaller fitness values than the fitness of the knee point individual [48] as promising individuals [42].

This multifidelity-based surrogate model of GP [42] has achieved higher performance that to be able to dramatically reduce the computational time of genetic programming at all the DFJSS scenarios.

3.4 Multitask Genetic Programming

Multitask learning is an inductive transfer approach that improves generalization performance by leveraging the domain-specific information contained in training related tasks sharing representation simultaneously[7].

Genetic programming with multitasking method which improves the efficiency of solution for multiple dynamic flexible job shop scheduling problems with scheduling heuristics [40][41][45].

3.4.1 A preliminary multitask GP for DFJSS

Zhang et al.[41] introduced multitasking genetic programming hyper-heuristic to evolve DFJSS scheduling heuristics, which is the first attempt to explore the tree-based heuristic search space by evolutionary multitask GP algo-

rithm. The different job shop tasks in DFJSS have the same objective but different configurations which can be optimised as different but related simultaneously.

This preliminary multitask GP defines the tasks from the DFJSS with different utilisation levels but with same objective as related tasks to be optimised together[41]. The algorithm separates the entire GP population into several subpopulations which represents different utilisation levels but the same objective. The larger utilisation level contains more complex scheduling tasks. The individuals in the different subpopulations match the different tasks and evolved respectively, however, different subpopulations share their knowledge from each other.

The transfer knowledge from different tasks through the crossover GP operation which defines a transfer ratio tr to control the frequency of transfer knowledge from other subpopulations and simply transfer knowledge at each generation.

- If $rand <= tr$, The first $parent_1$ selected from current subpopulations, The second $parent_2$ selected from other subpopulations.
- If $rand > tr$, two parents selected both from the current subpopulations.

The knowledge transfer will share the knowledge for the complex task subpopulation which speeds up its convergence from simple task. Meanwhile, the simple task of subpopulation will increase the quality of individuals from others[41].

3.4.2 Multitask generative hyperheuristic GP

Zhang et al.[40] proposed multitask GP-based generative hyperheuristic algorithm has improved the quality of the evolved high-level heuristics which considered all the tasks in the multitask scenario.

This novel approach for DFJSS different scenarios defined the multitask problem with same objective while with the different utilization levels

as homogeneous multitask. However, the multitask problem with different objectives but with the same utilization level as heterogeneous multitask. The two types multitask method focus on DFJSS problem objective and utilization levels respectively. The individual sharing knowledge through GP crossover operation which the mechanism is mostly the same with the preliminary multitask GP [41]. This not only obtains useful information for the individual but also maintains the individuals original characteristics.

Both homogeneous and heterogeneous multitask scenarios of DFJSS implemented by the multitask generative hyper-heuristic GP can achieve better quality scheduling heuristics and get better performance. The heterogeneous multitask scenario has more potential chance to be optimized [40].

3.4.3 Surrogate-Assisted Multitasking GP for DFJSS

Surrogate-assisted evolutionary multitasking genetic programming is combined surrogate function [17] and multitask generative hyper-heuristic algorithm [40] to share useful knowledge between different DFJSS scheduling tasks. This novel approach[45] can improve the quality of scheduling heuristics at all the DFJSS scenarios significantly.

The algorithm of Surrogate-Assisted Multitasking GP[45] initialises the population with k subpopulations for the k tasks solution first. At GP evaluation process, the individuals are evaluated with different training instances according to the tasks in different subpopulations. Meanwhile, the phenotypic characterisations of individuals and their corresponding fitness in each subpopulation build the surrogate models. The phenotypic characterisation individual with Euclidena distance apply the KNN method to discover the most similar individual as a decision vector based on a set of decision situations [17]. There are k surrogates for each generation, and the surrogates will be rebuilt at the next generation[45]. At

the evolution stage, an offspring pool is filled by $n * subpopsize$ number of offspring which are generated for each subpopulation. The final offspring new individuals for subpopulation for tasks by removing the duplicated individuals according to their phnotypic characterisation from the offspring pool, and selected by the ranking of the fitness which estimated by the surrogate S_i.

Zhang et al.[45] proposed the surrogate-assisted evolutionary multitasking algorithm with genetic programming hyper-heuristics for dynamic flexible job shop scheduling problems has very good performance with high convergence speed, quality of heuristics in different scenario and utilization level. Because the GP involved large number of new offspring, evaluation by surrogate model, and share individual knowledge by multitasks. There are some interesting directions which can be studied in the future, such as Gaussian processes and neural networks surrogate model.

3.5 Feature Selection

Feature selection process removes the irrelevant redundant and noisy attributes of data, which is an important and necessary task in machine learning and data mining [37][22]. There are broadly three categories of feature selection methods, which are filter, wrapper, and embedded [22]. However, traditional feature selection methods are not suitable applying on GPHH for DFJSS [23] because the tasks of GPHH are completely different to traditional methods, and the instances in GPHH for DFJSS are not independent on each other which is opposite to traditional machine learning and data mining [23].

This section will introduce feature selection approaches by GPHH for DFJSS which recently implemented successfully [24][23][47][44][13]. The following subsections organised as: subsection I explains a domain-knowledge-free feature ranking and selection method of GP for DFJSS [24]. Subsection II introduces an efficient practical feature selection based on niching and

surrogate model algorithm [23]. Subsection III illustrates the feature selection process applied in GPHH for DFJSS with two-stage approach and its extension version [47][44].

3.5.1 Domain-knowledge-free feature ranking and selection

Feature selection of GP mainly focus on genetic programming terminals selection. The important factor for GP to evolve good dispatching rules of JSS effectively is the terminals to be involved [24]. There are large number of job shop features involved in different scenarios for different DFJSS objectives optimization such as global shop-level features (current time), job-related features (processing time, due data, waiting time), and machine-related features (machine ready time, work in the queue, utilisation level). Mei et al.[24] proposes a domain-knowledge-free feature ranking and selection approach with GP for DFJSS which improves the disadvantages of feature frequency ranking method [15] and has created the basic concept for feature selection process in GP for JSS.

The domain-knowledge-free feature ranking and selection of GP algorithm [24] based on feature removal in priority function to find the irrelevant features. The feature removal in priority function Eq. 3.1:

$$g(\overrightarrow{x} \setminus x_i) = g(\overrightarrow{x}|x_i = 1). \tag{3.1}$$

Where $\overrightarrow{x}$ is the vector of feature, and x_i is the removing feature fix with 1 instead.

Then, apply this function on training instances to calculate the contribution on different objective values. The formula is Eq. 3.2:

$$\zeta(x_i; g(\overrightarrow{x})) = \phi\tau(g(\overrightarrow{x} \setminus x_i)) - \phi\tau(g(\overrightarrow{x})). \tag{3.2}$$

Where $\zeta(x_i; g(\overrightarrow{x}))$ is defined as the contribution of the feature x_i to a priority function $g(\overrightarrow{x}$ on a set of JSS instances τ. Note that if contribution

$\zeta(x_i; g(\overrightarrow{x}))$ is positive value implicates that removing x_i feature from the dispatching rule will generate worse schedules for the tested JSS instances. So the more contribution value means the feature is the more relevant feature.

This algorithm of GP feature selection is an offline process, which has three stages [24]. Stage1 obtain 30 best rules by running 30 independent standard GP with entire feature set of JSS. Stage2 to select the relevant features based on contribution value as the new terminal set. The final stage conducted standard GP again by using the new terminal set.

The results of this feature selection GP [24] have shown the better performance compare with standard GP for JSS.

3.5.2 An efficient practical feature selection Algorithm

Mei et al.[23] proposed an efficient practical feature selection of GP algorithm for JSS by employing niching and surrogate model which is the first online process of feature selection of GP. The Niching-GP feature selection algorithm has two steps which the first step is to obtain the best diverse set of individuals by Clearing method of Niching techniques, the second step is to select the best set of terminals by the important feature depending both the individual fitness and contributions on individuals [23].

In the first step, the clearing method and selecting the best diverse set of individuals both apply the niching technique by set two parameters radius σ for controlling the range of each niche, and the capacity k to determine the number of individuals in each niche. The clearing method based on phenotypic characterisation distance measure [17] applying niching algorithm to clear the poor individual in the crowding areas. The final best diverse set of individual is also obtained by applying the niching technique from the cleared population [23].

The feature selection step select the feature subset from the final best diverse individual set base on the feature into individual fitness and con-

tributions of individual. The contributions of individual calculated by the Domain-knowledge-free feature ranking method [24] show Eq.3.3. The individual fitness will be used for weighted majority voting process Eq.3.4 and Eq. 3.5 to select features set. The feature is to be selected when its $\zeta(t, \overrightarrow{r})$ larger than $threshold = 0.001$ and the total weight voting for it is larger than the total weight not voting for it [23].

$$\zeta(t, \overrightarrow{r}) = fit(\overrightarrow{r} - fit(\overrightarrow{r}|t = 1), \tag{3.3}$$

$$w(r) = max\Big\{\frac{g(r) - g_{min}}{g_{max} - g_{min}}, 0\Big\}, \tag{3.4}$$

$$g(r) = \frac{1}{1 + fit(r)}, \tag{3.5}$$

This piratical feature selection algorithm with surrogate model with GP for JSS have achieved significantly better results than using the entire feature sets GP [23].

3.5.3 Feature selection approach with GPHH for DFJSS

A two stage genetic programming hyper-heuristic approach with feature selection for DFJSS proposed by Zhang et al.[47], which can be considered the first feature selection method with GPHH for DFJSS. This approach is different with classical GP separating the whole GP process into two stages. The first stage (set it before generation 50) is for feature selection GP proceeds with a niching evaluator and a surrogate model that produces two sets of informative terminals for evolving routing and sequencing rules respectively by feature selection mechanism triggered when GP at generation 50. The second stage GP main process is the same as the classical GP which the population will be evaluated without niching and surrogate model, while the mutation process will be using the two new routing and sequencing terminals which selected by the first stage.

The feature selection algorithm with GPHH for DFJSS [47] is modified from feature selection with GP for JSS approach [23]. In the Clearing

method of Niching technique [23] at feature selection stage, the clearing process with a radius $\sigma = 0$ and a capacity $k = 1$ after traditional fitness evaluation. The surrogate model is aims to reduce the computation cost for evaluate the fitness objective [47].

The two-stage GP with feature selection and individual adaptation strategies approach [44] is the extension version of previous algorithm [47]. The difference of the two implemented approach is the initialisation of stage 2, which the initialisation adapts the final population of stage 1 as part of the initial population, the other part generated from selected feature terminals randomly. Meanwhile, the mutation of stage 2 will choose selected feature terminals as well.

Both two feature selection with GPHH for DFJSS approach [47][44] shows that they have better interpretability as there are fewer unique features in the rules and smaller size of rules. They weren't compromising the performance, and be able to efficiently evolve more interpretable rules automatically [44].

Fan and Xiong et al. [13] presented a GPHH for DFJSS with extended technical precedence constraints (ETPC) approach which employs a problem specific attribute selection and individual selection with threshold and probabilistic function algorithm. For the DFJSS with ETPC in the GP terminal set, the ETPC related specific attribute will increase its weight to obtain more chance to be processed. The second method in the study is applying the threshold condition to detect inferior individuals. The individuals are applied to the predefined training instance to obtain objective value if exceeded the threshold value that individual will be inferior. Then, whether the inferior individuals terminating or not which have to check the probabilistic value by a probability function Eq. 3.6.

$$p_r = e^{\left(-\frac{C}{x}\right)}, \tag{3.6}$$

where C is a constant factor set on 5, and x is iteration number of GP generations. The probability function adapted by GP iteration which may be suitable on other GPHH technical model for DFJSS.

Chapter 4

Issues and Challenges Discussion

This Chapter briefly discusses some issues and challenges from previous research and find the directions for future study.

The genetic programming with hyper-heuristic evolved scheduling policies for dynamic flexible job shop scheduling problems have been studied from many aspects for many years. That research has achieved significantly better performances by improving effectiveness, reducing the computational time, obtaining more efficiency and better interpretability.

The DFJSS problem has different scheduling scenarios. How to balance the solution objective is the challenge. Although, the multi-objective coevolution GPHH for DFJSS studies [26][50] have introduced solutions, there are some further research directions which can be pointed such as analysis of the multi-objective algorithm parameter setting, and the GP evolved tree structure scheduling.

The effectiveness of GPHH is the important study prospective in DFJSS problems. The guidance operation of GPHH based on cooperation coevolution approach [43] has improved the quality of GP offspring to obtain effective scheduling heuristic of DFJSS problems. However, further research may focus on the rule size investigation to reduce the individual tree size which can have better understanding of the GP automatically designed DFJSS dispatching rule.

The surrogate model employed by GPHH for DFJSS can accelerate the convergence speed of GP iteration process by reducing the fitness evaluation cost. It obviously reduced the GPHH computational cost. Hildebrandt and Branke [17] developed a surrogate model based on nearest neighbor model to evaluate individual with the simple decision situation. Furthermore, by employing the use of simplified simulation model to reduce the fitness evaluation cost is the other popular surrogate model [29] [46] [49]. The KNN measurement method is used for ranking similarity phenotypic behavior of the new individual with the samples individual in the surrogate pool. Future study could attempt applying probabilistic learning algorithm as evaluation individuals surrogate model.

Another view of improving the effectiveness approach is to manipulate the multitask learning into GPHH for DFJSS. It has achieved better quality scheduling heuristic by sharing knowledge between variety utilization levels. As the dynamic flexible job shop scheduling problem has different scenarios by different objectives, it is interesting to investigate employing the multitask into the scenario level.

The GP trees representing DFJSS scheduling policies should have good interpretability which is the important aspect of judgement in machine learning. Applying suitable machine learning features not only reduces the computational cost but also improves the algorithm interpretability. Feature selection approach employed in GPHH for DFJSS has manipulated successfully [23][47][13]. In addition, feature construction method proposed by Yska et al.[38] which might need more investigation to have better understanding the individual heuristics based on GP trees structure.

Although, the GPHH generated dispatching rules can have higher competitive performance than the manually designed rules by the experiences of experts. The GP dispatching rules are still difficult to clearly explain how it works in detail because of its complicated deep structures. Both feature selection and feature construction approaches can potentially reduce the size of DFJSS dispatching rule which is generated by GPHH. It is

an interesting direction for further research which will reduce the computation time and concurrently increase the interpretability of GP for DFJSS.

Chapter 5

Conclusions

This thesis is mainly focused on the literature review of genetic programming hyper-heuristic (GPHH) to automatically generate dispatching rules for dynamic flexible job shop scheduling (DFJSS) problems. The goal of the review is to learn and obtain better understanding of GPHH for DFJSS. Firstly, the background chapter has learnt and explained the concepts of evolutionary computation genetic programming based hyper-heuristic algorithm and dynamic flexible job shop scheduling problems. The related work of genetic programming hyper-heuristic for dynamic flexible job shop scheduling problems studies have been reviewed and analysed at chapter 3. Finally, the issues and challenges of recently research and studies were discussed at the following chapter.

The related work of previous studies can be categorised into five types of approach such as multi-objective, cooperation coevolution, surrogate model, multitask method, and feature selection. DFJSS has different conflicted objectives such as make-span, flowtime and tardiness which described as the different scenarios. Applying GPHH with multi-objective cooperation coevolution algorithm can cope with those DFJSS objectives. It also has more improvement space for further research. Yska and Mei [39]proposed cooperation coevolution GPHH evolving sequencing rule and routing rule simultaneously which becomes the benchmark algorithm

of GPHH for DFJSS in many studies. Based on cooperation coevolution Zhang et al. [43] employed recombinative guidance by subtrees correlation coefficient optimized GPHH operation process which not only shows better performance of generated dispatching heuristics but also has good interpretability as well. Both the surrogate model and multitask method have increased the convergence speed of GPHH to make the algorithm more effective and efficient for DFJSS. In addition, the feature selection approach has achieved the better interpretability of DFJSS dispatching rules, while not compromising the performance.

More investigation and further research can point to some interesting directions in future. The surrogate model can employ different machine learning algorithm to choose the better offspring. GP subtrees analysis based on their correlation coefficient with individual tree which can obtain more understanding of DFJSS dispatching rules. We can study the feature construction approach in GPHH for DFJSS which will reduce the dispatching rule size to increase the quality of the solution and its interpretablility.

Bibliography

[1] BLACKSTONE, J. H., PHILLIPS, D. T., AND HOGG, G. L. A state-of-the-art survey of dispatching rules for manufacturing job shop operations. *International Journal of Production Research 20* (1 1982), 27–45.

[2] BLUM, C., AND LI, X. Swarm intelligence in optimization. In *Swarm Intelligence. Natural Computing Series.* Springer Berlin Heidelberg, 2008, pp. 43–85.

[3] BRANDIMARTE, P. Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations Research 41* (1993), 157–183.

[4] BRANKE, J., HILDEBRANDT, T., AND SCHOLZ-REITER, B. Hyper-heuristic evolution of dispatching rules: A comparison of rule representations. *Evolutionary Computation 23* (6 2015), 249–277.

[5] BRANKE, J., NGUYEN, S., PICKARDT, C. W., AND ZHANG, M. Automated design of production scheduling heuristics: A review; automated design of production scheduling heuristics: A review. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION 20* (2016).

[6] BURKE, E. K., HYDE, M. R., KENDALL, G., OCHOA, G., OZCAN, E., AND WOODWARD, J. R. Exploring hyper-heuristic methodologies with genetic programming. In *Computational intelligence*. Springer, 2009, pp. 177–201.

[7] CARUANA, R. Multitask learning. *Machine Learning 28* (1997), 41–75.

[8] CHAUDHRY, I. A., AND KHAN, A. A. A research survey: Review of flexible job shop scheduling techniques. *International Transactions in Operational Research 23*, 3 (2016).

[9] DARWISH, A., HASSANIEN, A. E., AND DAS, S. A survey of swarm and evolutionary computing approaches for deep learning. *Artificial Intelligence Review 53* (2020), 1767–1812.

[10] DURASEVIĆ, M., AND JAKOBOVIĆ, D. A survey of dispatching rules for the dynamic unrelated machines environment. *Expert Systems with Applications 113* (12 2018), 555–569.

[11] EIBEN, A. E., AND SCHOENAUER, M. Evolutionary computing. *Information Processing Letters 82* (2002), 1–6.

[12] ETHEM, A. A. *Introduction to Machine Learning*, third edition. ed. Cambridge, Massachusetts : MIT Press, 2014.

[13] FAN, H., XIONG, H., AND GOH, M. Genetic programming-based hyper-heuristic approach for solving dynamic job shop scheduling problem with extended technical precedence constraints. *Computers and Operations Research 134* (10 2021).

[14] FERREIRA, C. Gene expression programming: A new adaptivealgorithm for solving problems. *Complex Systems 13* (2001), 87–129.

[15] FRIEDLANDER, A., NESHATIAN, K., AND ZHANG, M. Meta-learning and feature ranking using genetic programming for classification: Variable terminal weighting. *2011 IEEE Congress of Evolutionary Computation (CEC)* (6 2011), 941–948.

[16] GAREY, M. R., JOHNSON, D. S., AND SETHI, R. The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research 1* (5 1976), 117–129.

[17] HILDEBRANDT, T., AND BRANKE, J. On using surrogates with genetic programming. *Evolutionary Computation 23* (9 2015), 343–367.

[18] HOLTHAUS, O. Scheduling in job shops with machine breakdowns: an experimental study. *Computers & Industrial Engineering 36* (1 1999), 137–162.

[19] JAYAMOHAN, M. S., AND RAJENDRAN, C. New dispatching rules for shop scheduling: A step forward. *International Journal of Production Research 38* (2 2000), 563–586.

[20] KOZA, J. R. Genetic programming as a means for programming computers by natural selection. *Statistics and Computing 4* (1994), 87–112.

[21] KOZA, J. R., AND POLI, R. Genetic programming. In *Search methodologies*. Springer, 2005, pp. 127–164.

[22] LIU, H., AND YU, L. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering 17* (4 2005), 491–502.

[23] MEI, Y., NGUYEN, S., XUE, B., AND ZHANG, M. An efficient feature selection algorithm for evolving job shop scheduling rules with genetic programming. *IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE 1* (2017), 339.

[24] MEI, Y., ZHANG, M., AND NYUGEN, S. Feature selection in evolving job shop dispatching rules with genetic programming. *Proceedings of the Genetic and Evolutionary Computation Conference 2016* (7 2016), 365–372.

[25] NGUYEN, S., ZHANG, M., JOHNSTON, M., AND TAN, K. *Genetic Programming for Job Shop Scheduling*. Springer International Publishing, 06 2018, pp. 143–167.

[26] NGUYEN, S., ZHANG, M., JOHNSTON, M., AND TAN, K. C. A coevolution genetic programming method to evolve scheduling policies for dynamic multi-objective job shop scheduling problems. *2012 IEEE Congress on Evolutionary Computation* (6 2012), 1–8.

[27] NGUYEN, S., ZHANG, M., JOHNSTON, M., AND TAN, K. C. Dynamic multi-objective job shop scheduling: A genetic programming approach. In *Automated Scheduling and Planning. Studies in Computational Intelligence*, vol. vol 505. Springer, Berlin, Heidelberg, 2013, pp. 251–282.

[28] NGUYEN, S., ZHANG, M., JOHNSTON, M., AND TAN, K. C. Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative coevolution genetic programming. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION 18* (2014).

[29] NGUYEN, S., ZHANG, M., AND TAN, K. C. Surrogate-assisted genetic programming with simplified models for automated design of dispatching rules. *IEEE Transactions on Cybernetics 47* (9 2017), 2951–2965.

[30] PEARL, J. *Heuristics : intelligent search strategies for computer problem solving*. Reading, Mass. : Addison-Wesley Pub. Co., 1984.

[31] PEZZELLA, F., MORGANTI, G., AND CIASCHETTI, G. A genetic algorithm for the Flexible Job-shop Scheduling Problem. *Computers and Operations Research 35*, 10 (oct 2008), 3202–3212.

[32] PINEDO, M. L. *Scheduling*, fifth edition ed. Springer International Publishing, 2016.

[33] RAJENDRAN, C., AND HOLTHAUS, O. A comparative study of dispatching rules in dynamic flowshops and jobshops. *European Journal of Operational Research 116* (7 1999), 156–170.

[34] RUSSELL, S., AND NORVIG, P. *Artificial Intelligence: a Modern Approach, EBook, Global Edition : A Modern Approach*. Pearson Education, Limited, Harlow, UNITED KINGDOM, 2016.

[35] SELS, V., GHEYSEN, N., AND VANHOUCKE, M. International journal of production research a comparison of priority rules for the job shop scheduling problem under different flow time-and tardiness-related objective functions 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* (2012), 4255–4270.

[36] SUTTON, R. S., AND BARTO, A. G. *Reinforcement Learning, second edition: An Introduction*. Adaptive Computation and Machine Learning series. MIT Press, 2018.

[37] XUE, B., ZHANG, M., BROWNE, W. N., AND YAO, X. A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation 20* (8 2016), 606–626.

[38] YSKA, D., MEI, Y., AND ZHANG, M. Feature construction in genetic programming hyper-heuristic for dynamic flexible job shop scheduling. *Proceedings of the Genetic and Evolutionary Computation Conference Companion* (7 2018), 149–150.

[39] YSKA, D., MEI, Y., AND ZHANG, M. Genetic programming hyper-heuristic with cooperative coevolution for dynamic flexible job shop scheduling. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 10781 LNCS* (2018), 306–321. GPHH cooperative coevolution for DFJSS.

[40] ZHANG, F., MEI, Y., NGUYEN, S., TAN, K. C., AND ZHANG, M. Multitask genetic programming-based generative hyperheuristics: A

case study in dynamic scheduling. *IEEE Transactions on Cybernetics* (2021), 1–14.

[41] ZHANG, F., MEI, Y., NGUYEN, S., AND ZHANG, M. A preliminary approach to evolutionary multitasking for dynamic flexible job shop scheduling via genetic programming. *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion* (7 2020), 107–108.

[42] ZHANG, F., MEI, Y., NGUYEN, S., AND ZHANG, M. Collaborative multifidelity-based surrogate models for genetic programming in dynamic flexible job shop scheduling. *IEEE Transactions on Cybernetics* (2021), 1–15.

[43] ZHANG, F., MEI, Y., NGUYEN, S., AND ZHANG, M. Correlation coefficient-based recombinative guidance for genetic programming hyperheuristics in dynamic flexible job shop scheduling. *IEEE Transactions on Evolutionary Computation 25* (6 2021), 552–566.

[44] ZHANG, F., MEI, Y., NGUYEN, S., AND ZHANG, M. Evolving scheduling heuristics via genetic programming with feature selection in dynamic flexible job-shop scheduling. *IEEE Transactions on Cybernetics 51* (4 2021), 1797–1811.

[45] ZHANG, F., MEI, Y., NGUYEN, S., ZHANG, M., AND TAN, K. C. Surrogate-assisted evolutionary multitask genetic programming for dynamic flexible job shop scheduling. *IEEE Transactions on Evolutionary Computation 25* (8 2021), 651–665.

[46] ZHANG, F., MEI, Y., AND ZHANG, M. Surrogate-assisted genetic programming for dynamic flexible job shop scheduling. *AI 2018: Advances in Artificial Intelligence 11320* (2018), 766–772.

[47] ZHANG, F., MEI, Y., AND ZHANG, M. A two-stage genetic programming hyper-heuristic approach with feature selection for dynamic

flexible job shop scheduling. *Proceedings of the Genetic and Evolutionary Computation Conference* (7 2019), 347–355.

[48] ZHANG, X., TIAN, Y., AND JIN, Y. A knee point-driven evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation 19* (12 2015), 761–776.

[49] ZHOU, Y., JUN YANG, J., AND HUANG, Z. Automatic design of scheduling policies for dynamic flexible job shop scheduling via surrogate-assisted cooperative co-evolution genetic programming. *International Journal of Production Research 58* (5 2020), 2561–2580.

[50] ZHOU, Y., YANG, J.-J., AND ZHENG, L.-Y. Hyper-heuristic coevolution of machine assignment and job sequencing rules for multi-objective dynamic flexible job shop scheduling. *IEEE Access 7* (2019), 68–88.