A Filtering Genetic Programming Framework for Stochastic Resource Constrained Multi-Project Scheduling Problem under New Project Insertions

HaoJie Chen [1]*,Guofu Ding*[1]*, Jian Zhang*[1]*, Rong Li*[1]*, Lei Jiang*[1]*, Shengfeng Qin*[2]

HaoJie Chen
e-mail: chenhaojie12138@163.com
Guofu Ding
e-mail: dingguofu@163.com
Jian Zhang(✉Correspondence author)
e-mail:jerrysmail@263.net
Rong Li
e-mail: bogiey@163.com
Lei Jiang
e-mail: jianglei0506@163.com
Shengfeng Qin
e-mail: sheng-feng.qin@northumbria.ac.uk

1. **School of Mechanical Engineering, Southwest Jiaotong University, Chengdu 610031, China**
2. **Department of Design, Northumbria University, Newcastle upon Tyne NE1 8ST, UK**

Abstract: Multi-project management and uncertain environment are very common factors, and they bring greater challenges to scheduling due to the increase of problem complexity and response efficiency requirements. In this paper, a novel hyper-heuristic based filtering genetic programming (HH-FGP) framework is proposed for evolving priority rules (PRs) to deal with a multi-project scheduling problem considering stochastic activity duration and new project insertion together, namely the Stochastic Resource Constrained Multi-Project Scheduling Problem under New Project Insertions (SRCMPSP-NPI), within heuristic computation time. HH-FGP is designed to divide traditional evolution into sampling and filtering evolution for simultaneously filtering two kinds of parameters constituting PRs, namely depth range and attribute, to obtain more effective PRs. Based on this, the existing genetic search and local search are improved to meet the depth constraints, and a multi-objective evaluation mechanism is designed to achieve effective filtering. Under the existing benchmark, HH-FGP is compared and analysed with the existing methods to verify its effectiveness.
Key words: Filtering evolution; Genetic programming; Priority rule; Stochastic resource constrained multi-project scheduling

1 Introduction

As a core problem in the field of project management and scheduling, the Resource Constrained Project Scheduling Problem (RCPSP) has been a hot research topic for many years since it was described (Habibi, Barzinpour & Sadjadi, 2018; Pritsker, Watters & Wolfe, 1969). Considering the limited resource supply, RCPSP optimizes single or multiple objectives by generating the sequencing of activities in the project, and its important objectives include makespan and project delay minimization (Hartmann & Briskorn 2010). In the past decade, developing better optimization methods for RCPSP has attracted extensive attention, especially meta-heuristics-based (Pellerin, Perrier & Berthaut, 2020). However, the applications of RCPSP are still greatly limited due to the following two reasons. Firstly, in practice, up to 90% of project management is carried out in a multi-project environment, in which RCPSP needs to consider the resource competition among projects (Payne, 1995). As a result, RCPSP is extended to the Resource Constrained Multi-Project Scheduling Problem (RCMPSP), in which multiple projects are collectively regarded as a *portfolio* (Van Eynde & Vanhoucke, 2020). Secondly, the project execution is often in a stochastic and uncertain environment, such as stochastic activity duration, new project insertion and varying resource availabilities, so that only 40% of projects are completed within their planned time. This uncertainty furthers the scheduling complexity, and leads RCPSP to the Stochastic Resource Constrained Project Scheduling Problem (SRCPSP) being modelled and optimized (Satic, Jacko & Kirkbride, 2020). In actual production and engineering management, there are a lot of situations on which it is needed to consider these two extensions at the same time. For example, in aircraft assembly production, multiple aircrafts often assembled simultaneously, and the duration of each assembly operation is stochastic due to manual assembly and other factors, so multi-project and stochastic activity duration need to be considered in scheduling.

For project scheduling under uncertainty, except for a few pure proactive scheduling studies (Lamas & Demeulemeester, 2016), there are mainly two problem solving strategies. The first one

is called *proactive-reactive scheduling* with a two-step implementation (Van de Vonder & Demeulemeester, 2007). In the planning phase, the proactive part constructs a baseline schedule with adding some buffers based on extra anticipation to deal with the predictable variability or directly ignoring the future uncertainty. Then, in the response/reactive phase, the baseline is adjusted and repaired when unexpected events occur, so as to make it feasible again. In this stage, the optimization objective is always to make the new baseline as close to the original one as possible. The application of proactive-reactive scheduling is very limited, especially in a highly uncertain environment, because according to statistics, almost 95% of the time is spent on revising baseline schedules, which greatly increases the time cost (Wang et al., 2017). The second is *stochastic scheduling*, in which an activity duration in the project is regarded as a known distribution, and the most commonly used optimization objective is transformed into minimizing the expected makespan. In this strategy, there is no baseline generation, and the solution is not a deterministic schedule, but a so-called scheduling policy (Möhring, Radermacher & Weiss, 1984, 1985).

Based on the scheduling policy with the function of transforming activity sequence into schedule, how to sort activities is very important and critical. The existing researches for solving SRC(M)PSP mainly focus on heuristics represented by PRs and meta-heuristics (see Section 2.2 for details). It is worth noting that when scheduling a project or portfolio in an uncertain environment, robustness and response efficiency are also very important objective indicators in addition to schedule quality. For an activity sequence strategy (a meta-heuristic or PR), its (rule) robustness metrics often depends on the deviation from expected value when it is used to schedule an instance in a stochastic environment (Wang et al., 2017). However, in order to consider the generality of a strategy, multiple different instances should be considered, and the comprehensive evaluation of scheduling multiple stochastic instances under this strategy should be built. Therefore, the applicability of meta-heuristics in SRCPSP is limited because its iterative calculation and a large number of random searches deteriorate the responsiveness and robustness respectively. Conversely, PRs can avoid these problems, but the unavoidable defect of PRs is that its lack of optimization ability and its dependence on problems, and it is difficult to artificially select the optimal PR for different problems or construct a better hybrid PR. Hyper-heuristics are proposed to improve this problem, which use the upper-level search mechanism to select the low-level heuristics or automatically generate new heuristics by using the constituent elements of original heuristics, so as to have better performance in the heuristic calculation time, that is, for an input, the scheduling result is obtained directly by executing the generated or selected heuristics without iteration or search time consumption (Burke et al., 2013). The heuristic calculation time here refers to the time to get the scheduling result by executing a heuristic for an input, that is, there is no time consumption of iteration or search. This technology has great potential and recently becomes a research hotspot in the scheduling field, such as flow shop scheduling (Lin, Wang & Li, 2017), job shop scheduling (Hildebrandt et al., 2010; Hildebrandt & Branke, 2015) and project scheduling (see Section 2.3 for details).

In this paper, a novel HH-FGP is proposed to solve SRCMPSP-NPI with multi-objective optimization, which is an extension of the Stochastic Resource Constrained Multi-Project Scheduling Problem (SRCMPSP) considering both stochastic activity durations and random new project insertions, so as to meet the actual needs of multiple projects scheduling with a stochastic environment. In HH-FGP, in order to implement filtering operations to select two effective

parameters (i.e., attributes and depth range) of gene expression tree automatically that affect the performance of evolved PRs, the genetic evolution is divided into two parts depending on the parameters of generation number. The main function of the first part is sampling, in which the population is divided into multiple sub-populations depending on the tree depth for independent evolution. Based on the optimal PR set generated by each sub-population, the depth range and attribute set constituting PRs are filtered and then input into filtering evolution, that is, the invalid attributes and unreasonable depths disappear in the search of filtering evolution. After benchmark-based testing and analysing, HH-FGP can get better results than the existing methods, and the main contributions of this paper are:

1. A novel HH-FGP framework is proposed to solve SRCMPSP-NPI, which makes the traditional genetic programming (GP) more effective by filtering both attribute set and depth range of gene expression tree, and it provides a new idea for the subsequent hyper-heuristic solution to project scheduling, especially in a stochastic environment.

2. A new filtering evaluation mechanism is proposed, which can evaluate the tree depth and attributes under multi-objective optimization to realize effective filtering.

3. The crossover and local search operators in the existing GP are improved to ensure that the PRs of each subpopulation maintain a fixed depth in the sampling evolution and to avoid the depth of PRs exceeding the filtered range in filtering evolution.

The remainder of this study is structured as follows: Section 2 introduces the relevant research and the motivation of this paper. In Section 3, the description and mathematical model of SRCMPSP-NPI are introduced. Section 4 describes the HH-FGP framework and its optimization in detail. In Section 5, numerical experiments with design and result analysis are reported, and the conclusion of this work and some future research directions are given in Section 6.

2 Related work

The core investigation of this study is about depth length and attribute selections through filtering when applying hyper-heuristics to solve SRCMPSP-NPI, the relevant research is thus introduced from three aspects. The first is about existing (meta-) heuristics methods for stochastic project scheduling (i.e., meta-heuristics and PR based heuristics), the second is about hyper-heuristics methods for project scheduling, and the last is about applying filtering techniques in hyper-heuristics methods for scheduling. On this basis, the motivation of this paper is introduced in Section 2.4.

2.1 Existing (meta-) heuristics for stochastic project scheduling

As a more practical problem, the optimization of SRC(M)PSP is constantly being explored. Due to the high demand on computation time, there are only a few studies on exact algorithms, such as branch and bound (Stork, 2002), exact procedure based on Markov chain (Creemers, 2015) and mixed integer linear programming (Alipouri et al., 2020), and they are only suitable for small and medium scale SRC(M)PSP. More importantly, heuristics or meta-heuristics play a major role in this optimization problem. The optimization research of heuristics or meta-heuristics can be divided into two key parts, that is, how to realize activity sequencing and how to convert activity sequence into schedule, leading the studies to explore new policy classes and improved search

methods. Based on these two parts, the research in SRC(M)PSP can be classified as shown in Fig.1.

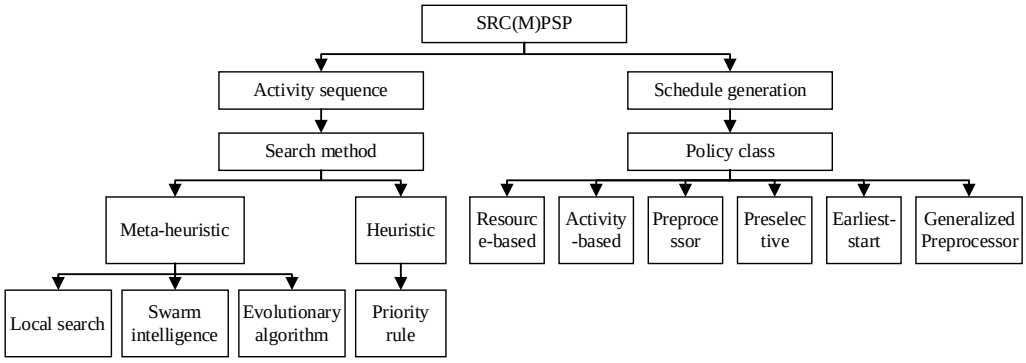

Fig.1 The research classification in SRC(M)PSP

For activity sequencing in SRC(M)PSP, many effective heuristics or meta-heuristics have been proposed and analysed. Similar to the development of other scheduling optimization, the first meta-heuristic studied is greedy and local search. Golenko-Ginzburg & Gonik (1997) proposed three operators with functions of control, calculation and selection, and then combined them into a greedy meta-heuristic algorithm to optimize makespan of SRCPSP. In order to further improve the search ability in solving SRCPSP, Tsai & Gemmill (1998) designed a diversified tabu search algorithm based on multiple tabu lists, randomized short-term memory and multiple starting schedules. Ballestín & Leus (2009) described a GRASP-heuristic strategy combined with descriptive sampling. Then, the improved evolutionary algorithm and swarm intelligence are applied in SRC(M)PSP. By combining permutation-based local search, Fang et al. (2015) improved an estimation of distribution algorithm to solve SRCPSP with a clear dominance under a medium and high variance distribution. Ma et al. (2016) developed a genetic algorithm integrated with a 99-method based uncertain simulation for searching the quasi-optimal schedule of SRCPSP. Satic, Jacko & Kirkbride (2020) compared the optimization ability of five algorithms including genetic algorithm for SRCMPSP, and analysed their performance trend with the uncertainty increasing. Sallam, Chakrabortty & Ryan (2021) proposed a meta-heuristic switching approach based on Q-learning, that is, the reward mechanism is designed to control the alternating search of multi-operator differential evolution and discrete cuckoo search. Due to simplicity, rapidity, stability and intuition (Browning & Yassine, 2010), PR solving SRC(M)PSP has attracted more and more attention in recent years. Chen et al. (2018) summarized 17 PRs for scheduling SRCPSP. The experimental results not only show that the optimal PR is different in a deterministic environment and stochastic environment, but also verify that the optimal PR superior to multiple meta-heuristics with the goal of minimizing the expected makespan. By dividing the expected makespan into scheduling quality and robustness, Wang et al. (2017) and Chen et al. (2019) explored the performance of different PRs under SRCMPSP and SRCMPSP-NPI respectively.

When an activity sequence is generated, it may not be able to be carried out at the same time due to resource constraints. The function of policy class under the schedule generation is to calculate the start and completion time for each activity by adding extra start-finish or start-start constraints, so as to convert resource constraints into the new precedence constraints other than the original ones (Chen et al., 2018). There are six main types in current policy classes. The initial related research involves adding constraints based on the minimum forbidden set, resulting in

Earliest-start Policy Class (ES-policy) (Radermacher, 1981) and *Pre-selective Policy Class* (PS-policy) (Igelmund & Radermacher, 1983). The minimum forbidden set is defined in which there are no precedence constraints among activities, but the sum of all activity resource requirements exceeds the maximum supply for some resources, and any proper subset is resource feasible. However, the calculation of minimum forbidden set construction is very difficult. For example, in PSPLIB benchmark (Kolisch & Sprecher, 1996), the average minimum forbidden set number of J30 is 326, while J120 increases to 243,871, resulting in large calculation consumption of these two policy classes (Stork, 2002). In addition, *Pre-processor Policy Class* (PP-policy) (Ashtiani, Leus & Aryanezhad, 2011) and *Generalized Preprocessor Policy Class* (GP-policy) (Rostami, Creemers & Leus, 2018) are proposed, which automatically find start-finish and start-start constraints by constructing meta-heuristics. Although they have been verified to be effective, similarly, they bring large additional calculations and make the optimization more complex. Finally, two direct policy classes are proposed, including *Resource-based Policy Class* (RB-policy) (Chen et al., 2018) and *Activity-based Policy Class* (AB-policy) (Ballestín, 2007). The RB-policy is similar to the parallel schedule generation scheme in RCPSP (Kolisch, 1996), that is, at any decision-making time, the decision maker starts with all unimplemented activities without violating the resource and precedence constraints. The AB-policy adds a side start-start constraint on the basis of RB-policy. A large number of experiments show that RB-policy is superior to AB-policy in terms of efficiency and quality (Chen et al., 2018).

To sum up, first, the PR based SRC(M)PSP scheduling method is better than the majority meta-heuristics in efficiency and quality, which leads to PR being an important research direction in stochastic problems, especially PR is more practical. Second, compared with other policy classes, the RB-policy with certain advantages is more likely to be selected.

2.2 Hyper-heuristics in project scheduling

Over the past few years, hyper-heuristics algorithms have been applied in project scheduling, but most of them are in a static environment, i.e., RCPSP and its extensions. The key features of hyper-heuristics in comparing to meta-heuristics are…… Because meta-heuristics are often used in the high-level search mechanism of hyper-heuristics, the development of hyper-heuristics for project scheduling is also similar to meta-heuristics. At first, only some high-level optimization frameworks based on local search were explored, such as greedy search (Anagnostopoulos & Koulinas, 2012) and threshold accepting (Koulinas & Anagnostopoulos, 2012). Later, swarm intelligence and evolutionary algorithm are gradually developed as the high-level of hyper-heuristic. For example, Koulinas, Kotsikas & Anagnostopoulos (2014) designed a hyper-heuristic based on particle swarm optimization algorithm to control several low-level heuristics. Asta et al. (2016) integrated Monte-Carlo tree search, novel neighbourhood moves and memetic algorithm into hyper-heuristic for improving the search ability of multi-mode RCPSP. In addition, GP is the most extensive and mature in the research of hyper-heuristic for project scheduling, especially in the last several years. Lin, Zhu & Gao (2020) designed a GP that controls and manages ten heuristic rules to solve RCPSP considering multi-skills, and further proposed a decomposition mechanism to improve population diversity (Zhu et al., 2021). Chand et al. (2018) described a GP to evolve rules more suitable for RCPSP, and the results show that the evolved PRs are superior to the existing state-of-the-art PRs. On that basis, they added dynamic resource

disruptions to expand RCPSP (Chand et al., 2019a) and combined Rollout-Justification procedure to improve GP performance (Chand et al., 2019b). Different from the deterministic environment, hyper-heuristics have only some preliminary explorations in solving SRC(M)PSP. Wang et al. (2015) designed the scheduling process of SRCMPSP as a Markov decision process and used dynamic programming to match the optimal PR for each state. Based on the similarity idea, Alipouri et al. (2019) proposed a self-adaptive differential evolution to realize SRCPSP optimization considering fuzziness. Kühn, Völker & Schmidt (2020) described a two-stage hyper-heuristic, which assigns corresponding weights to different attributes by serially performing training in deterministic and stochastic environments, so as to obtain the combined PRs. Chen et al. (2021) introduced an ensemble genetic programming, in which ensemble learning is combined to evolve a PR decision set to solve SRCPSP, and this is the first time to evolve PRs through GP for solving project scheduling in a stochastic environment.

2.3 Hyper-heuristics with feature/attribute filtering in scheduling

For hyper-heuristics with evolving PRs, the relevant problem attributes are important parameters. The proper selection of effective attributes can not only reduce the solution space, but also improve the scheduling performance of evolved PRs (Branke et al., 2015a). Therefore, applying feature/attribute filtering in selection has become one of the important research directions in the scheduling field with hyper-heuristics and has achieved great success in other scheduling problems represented by job shop scheduling (JSP). Branke et al. (2015b) ignored the attributes that constitute the evolved PRs one by one and proved that some attributes play a significant role in the performance of evolved PRs. For JSP, Mei et al. (2016) designed a GP with feature filtering to select effective attributes by analysing the frequency of attributes in evolved PRs. Further, Mei et al. (2017) found that the attribute frequency could not fully express the attribute importance due to the particularity of tree coding structure and proposed a replacement method to calculate the attribute contribution. Experiments show that this replacement method is more effective. Based on this method, Zhang et al. (2019, 2020) not only extended the GP with feature filtering to the application considering flexible and dynamic JSP (only new job arrive considered), but also integrated the PR generation and feature evaluation into an evolutionary process. Masood et al. (2021) introduced a GP combined with NSGA-III and feature filtering to evolve PRs for multi-objective JSP. In the evaluation of this method, if a PR after replacement dominates the PR before replacement (or dominated by the PR before replacement), a reward of 1 (-1) will be given, while the non-dominated relationship before and after replacement is not handled.

2.4 Motivation

By analysing the above literature, the following conclusions can be obtained, which in turn motivate this research:

(1) PRs are concerned and used in the scheduling of RC(M)PSP because they have the ability of fast response, simplicity and stability. However, when the robustness indicator needs to be considered for SRC(M)PSP, in addition to these advantages, PRs are verified to have better comprehensive performance than meta-heuristics, especially in a high variance stochastic environment. In actual project scheduling, multiple stochastic disturbances (such as stochastic

activity duration and new project insertion) often exist at the same time. Therefore, it is very potential and meaningful to study the hyper-heuristics represented by GP to evolve PRs with better performance for SRCMPSP-NPI.

(2) The existing GP with feature/attribute filtering is oriented to JSP optimization, but lack of research on SRCMPSP(-NPI) to our best knowledge. When facing PR evolution under SRCMPSP(-NPI), in addition to the inconsistency of attribute set composition, the following changes need to be made, scheduling generation policy and objectives (adding the robustness objectives) due to different optimization models, the attribute calculation and normalization and PR performance evaluation caused by stochastic duration (see Section 4 for details). At the same time, for an evolved PR, some attributes either having little effect or even being counterproductive to its scheduling quality may further deteriorate its robustness, because their calculation depends on stochastic duration or new project insertion, resulting in the increase of stochastic variables in the whole priority calculation. Therefore, when optimizing SRCMPSP(-NPI), determining a reasonable attribute set can not only reduce the search space, but also offer great help to improve the PRs robustness, so the research of attribute filtering in SRCMPSP(-NPI) is very meaningful.

(3) Last but not least, the existing GP with filtering is only for attribute set. However, the common gene expression of evolved PRs in GP is in a tree structure, so the depth range of the tree is another important parameter because it determines the possible combination mode of evolved PRs. If the tree depth is too shallow, the scheduling quality of evolved PRs may be poor due to fewer factors considered in the PR structure. On the contrary, the complex PR structure may lead to the robustness deterioration. The effective depth range can not only further reduce the search space, but also improve the performance of evolved PRs. Therefore, it is valuable to study a novel GP framework able to filter attributes and depth range simultaneously.

To sum up, the HH-FGP framework proposed in this paper for solving SRCMPSP-NPI can expand the existing research from both the problem and solution method perspectives, and provide a novel idea for practising project scheduling in a stochastic environment.

3 Description and Mathematical model of SRCMPSP-NPI

In practice, many cases show that in addition to stochastic, new project insertion often needs to be considered in project scheduling. Similarly, taking assembly production as an example, due to the randomness of orders, some assembly tasks are unknown at the initial scheduling decision time (e.g., at the beginning of each month), that is, some assembly tasks need to be inserted after receiving relevant emergency orders. Therefore, Chen et al. (2019) proposed to extend SRCMPSP to SRCMPSP-NPI for considering both dynamic factors of stochastic activity durations and new project insertions on the basis of limited resources. In this problem, in addition to the fact that an activity $a_{i,j}$'s duration $d_{i,j}$ satisfies a known distribution, there are three important constraints described with the symbols shown in Table 1:

- *Precedence constraint*: this constraint is to express the logical relationship between activities in the project, that is, $a_{i,j}$ can start only after all activities in $\boldsymbol{P_{i,j}}$ are completed. It should be noted that there is no precedence constraint between activities from different projects.
- *Resource constraint*: this constraint expression is that when scheduling portfolio, resources are limited, that is, the requirement of activities in $\boldsymbol{A^t}$ for resource k cannot exceed RS_k. The makespan lower bound of p_i is equal to CP_i without considering this constraint.

- *Activity start constraint*: this constraint is to express a new project insertion, that is, the relevant information of p_i is unknown before time pst_i, and all activities in $\boldsymbol{A_i}$ can start only after time pst_i. If p_i is the initial project, then pst_i equals 0.

Table 1 The symbols in SRCMPSP-NPI

Symbol	Significance
$\boldsymbol{PS}$	the project set or portfolio
n	the number of projects in the portfolio
p_i	the ith project in the portfolio, i={1,2,…,n}
CP_i	the critical path length of p_i
AD_i	the expected makespan of p_i, in which the activity duration in p_i is the expected value when calculating it
SAD_i	the average makespan of project p_i derived from Monte Carlo simulations with stochastic activity durations
pst_i	the insertion time of p_i
$\boldsymbol{A^t}$	the execution activity set at time t
$\boldsymbol{A_i}$	the activity set in project p_i
mi	the number of non-dummy activities in project p_i
$a_{i,j}$	The jth activity in the project i, j= {0, 1, …, mi+1}, where j is 0 (mi+1) indicates the start (end) dummy activity
$d_{i,j}$	the duration of $a_{i,j}$
$d_{i,j}^*$	the expected duration of $a_{i,j}$
$st_{i,j}$	the start time of $a_{i,j}$
$\boldsymbol{S_{i,j}}$	the successor set of $a_{i,j}$
$\boldsymbol{TS_{i,j}}$	the total successor set of $a_{i,j}$
$\boldsymbol{P_{i,j}}$	the predecessor set of $a_{i,j}$
$\boldsymbol{K}$	the renewable resource set
RS_k	the maximum supply of the kth resource
$r_{i,j,k}$	the requirement of $a_{i,j}$ for resource k
t	the execution time, t={1,2,…,T}, where T represents a maximum time.

At the same time, since SRCMPSP-NPI is a stochastic multi-project scheduling problem, the scheduling quality and robustness both need to be considered from the perspective of project and portfolio. To sum up, with reference to work in Wang et al. (2017) and Chen et al. (2019) the mathematical expression of relevant constraints and objectives is as follows.

Objective:

$$Q_1 = \frac{1}{n}\sum_{p_i \in \boldsymbol{PS}} \frac{AD_i - CP_i}{CP_i} \tag{1}$$

$$Q_2 = \frac{\max\limits_{p_i \in \boldsymbol{PS}} AD_i - \max\limits_{p_i \in \boldsymbol{PS}} CP_i}{\max\limits_{p_i \in \boldsymbol{PS}} CP_i} \tag{2}$$

$$R_1 = \frac{1}{n}\sum_{p_i \in \boldsymbol{PS}} \left| \frac{SAD_i - AD_i}{AD_i} \right| \tag{3}$$

$$R_2 = \left| \frac{\max_{p_i \in PS} SAD_i - \max_{p_i \in PS} AD_i}{\max_{p_i \in PS} AD_i} \right| \quad (4)$$

s.t:

$$st_{i,j'} - st_{i,j} \geq d_{i,j} \quad \forall a_{i,j'} \in \boldsymbol{S_{i,j}} \quad (5)$$

$$\sum_{p_i \in \boldsymbol{P}, a_{i,j} \in \boldsymbol{A^t}} r_{i,j,k} \leq RS_k \quad \forall k \in \boldsymbol{K}, \forall t \in T \quad (6)$$

$$pst_i \leq st_{i,j} \quad \forall a_{i,j} \in \boldsymbol{A_i} \quad (7)$$

$$r_{i,0,k} = r_{i,(mi+1),k} = 0;\ P(d_{i,0} = 0) = P(d_{i,(mi+1)} = 0) = 1 \quad \forall k \in \boldsymbol{K}, \forall p_i \in \boldsymbol{PS} \quad (8)$$

$$P(d_{i,j} \leq 0) = 0 \quad (9)$$

Eq.(1) and Eq.(2) represent the scheduling quality of the project and portfolio respectively, which express the expected deviation percentage of the scheduling result from the CP_i when ***PS*** is the input. These two objectives can evaluate the scheduling ability of PRs, that is, assuming that the activity duration meets the expected value, we can find which priority rule gets the better scheduling result. However, it does make some errors when the actual activity durations are not equal to the expected value. Therefore, the robustness related objectives R1 and R2 described in Eq.(3) and Eq.(4) respectively are considered to deal with this problem, which represent the deviation from the expectation under multiple simulations. Eq.(3) and Eq.(4) represent robustness objectives, which represent the deviation of scheduling results from AD_i under multiple Monte Carlo simulations. The two objectives are used together to evaluate the deviation between the actual makespan and the expected one by using PRs scheduling when the actual duration of the activities is different from their expected value. Eq.(5) to Eq.(7) express the three important constraints: precedence constraint, resource constraint and start constraint, respectively. Eq.(8) shows the dummy activity constraint, that is, for a dummy activity, its duration is 0 and no resources are required. Eq.(9) indicates that the duration of any activity in the portfolio cannot be negative.

4 The framework and key implementation techniques of HH-FGP

Before describing the HH-FGP framework in detail, the gene expression of PRs in GP needs to be introduced to further understand the significance of attributes and depth range filtering, which often adopts the tree structure shown in Fig.2 (Chand et al. 2018). The PR is used to calculate the corresponding priority value for sorting activities in project scheduling through its own expression, which is represented by a functional function *"f"* such as "+" and "-" and associated useful attributes such as duration and resource requirement, and their combination at multiple levels. It can be seen that an evolved PR is expressed by combining the relevant attributes *"Att"* with the functional function *"f"*. In addition, its top layer has a discriminant *"Jud"* to determine whether the following priority expression is minimized or maximized. It can be seen from Fig.2 that the performance of PR is affected by the combined attributes and the combination mode, that is, the attribute set at each level (depth) and the maximum depth are two key

parameters. Therefore, how to avoid invalid attributes at each level and narrow the depth range of the tree to achieve effective search is very meaningful.

Based on this gene structure and the motivation of this paper, the framework of HH-FGP is shown in Fig.3, in which the highlighted part represents other contributions in addition to this framework. It can be seen that HH-TGP divides the iterative process of traditional GP into two parts, namely sampling and filtering evolution. Firstly, the population is divided into z sub-populations equally depending on the initial depth range, and only PRs with the same maximum depth are retained in each sub-population to fully explore the PR performance at different depths. For example, assuming that the minimum depth of the hyper-parameter is 2, the first sub-population only generates PRs with the maximum depth of 2 in the evolution. When the termination condition is reached in sampling evolution, each sub-population output an optimal (Pareto) PR set for multi-objective optimization. HH-FGP only divides the evolution of traditional GP into two parts, so the termination condition of sampling evolution is set as part of the total termination condition, which is designed to the top 50% in this paper to ensure sufficient sampling and filtering evolution at the same time. Then, HH-FGP collects all PR sets and merges them relying on non-dominated relationships. An attribute and depth evaluation mechanism under multi-objective optimization is designed to realize filtering, and the filtered parameters are used as input to perform the evolution of the second part until the maximum iteration is completed. The implementation of HH-FGP is shown in Section 4.1, and the relevant technical details are described below.

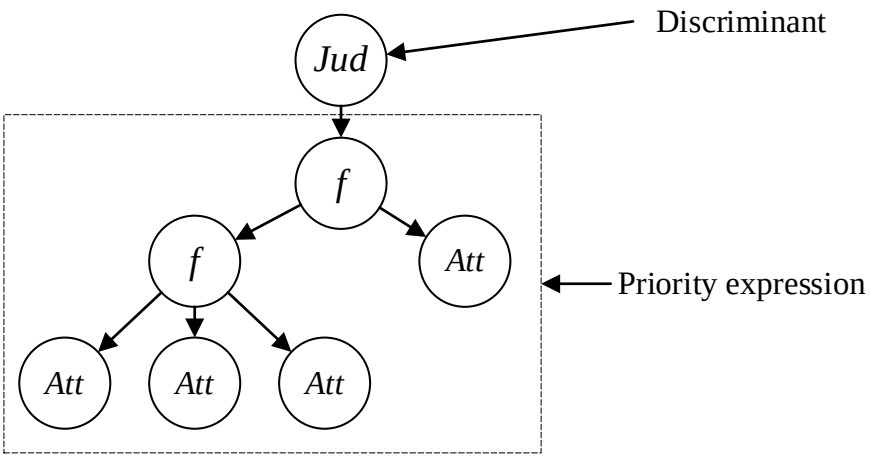

Fig.2 The gene expression of PR

4.1 Initialization and evaluation

The function of initialization is to generate PR individuals based on the gene shown in Fig.2, in which each node represents the discriminant or function or attribute, and the keys are the elements constituting PR and the tree structure generation method. Referring to Chand et al. (2018) and Chen et al. (2021), the discriminant is only *"fall"* and *"rise"*. If the discriminant is "*fall*", the function is minimization, it means the smaller the value calculated by the priority expression, the higher the priority. Otherwise, it is maximization. At the same time, the function set and attribute set in the sampling evolution part are shown in Table 2 and Table 3 respectively. It is worth mentioning that since SRCMPSP-NPI is a multi-project scheduling problem, the attribute

calculation equation in Table 3 has changed compared with the existing research (Chand et al., 2018; Chen et al., 2021), where $\boldsymbol{AE}^t$ represents the eligible activity set at time t, and the "CPL_i" attribute is added.

For the construction method in sampling evolution, the discriminant depends on the probability of 0.5 to randomly generate "*fall*" or "*rise*", while the priority expression adopts the classical ramped half-and-half method (Luke & Panait, 2001), in which the maximum depth of PRs is controlled in 2 to 6 (the depth of discriminant is 0). Compared with the sampling evolution part, in the input of filtering evolution, only the attribute set and the maximum depth range are obtained through filtering, and the others are completely consistent.

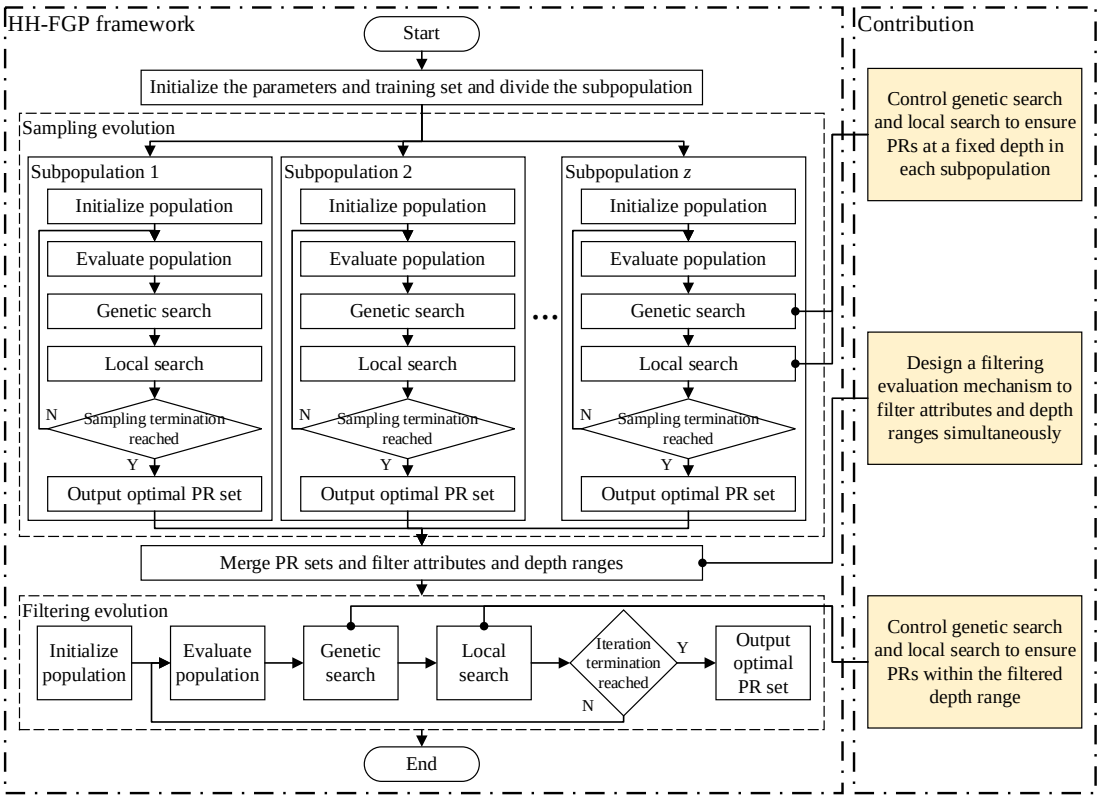

Fig.3 The framework of HH-FGP

Table 2 The function set

Symbol	Function	Formula	Symbol	Function	Formula
+	Add(x,y)	$x+y$	−	Sub(x,y)	$x-y$
×	Mul(x,y)	$x\times y$	Neg	Neg(x)	$-1\times x$
Exp	Exp(x)	e^x	Abs	Abs(a)	$\begin{cases} a & if\ a\geq 0 \\ -1\times a & otherwise \end{cases}$
/	Div(x,y)	$\begin{cases} x/y & if\ y\neq 0 \\ 0 & otherwise \end{cases}$	Max	Max(a,b)	$\begin{cases} a & if\ a\geq b \\ b & otherwise \end{cases}$
Min	Min(x,y)	$\begin{cases} x & if\ x\leq y \\ y & otherwise \end{cases}$	If	If(c,a,b)	$\begin{cases} a & if\ c=0 \\ b & otherwise \end{cases}$

Table 3 The attribute set

Attribute	Normalized calculation formula

Early Finish ($EF_{i,j}$)	$\frac{EF_{i,j}}{\max EF_{i',j'}} \quad a_{i,j}, a_{i',j'} \in \boldsymbol{AE}^t$
Late Start ($LS_{i,j}$)	$\frac{LS_{i,j}}{\max LS_{i',j'}} \quad a_{i,j}, a_{i',j'} \in \boldsymbol{AE}^t$
Late Finish ($LF_{i,j}$)	$\frac{LF_{i,j}}{\max LF_{i',j'}} \quad a_{i,j}, a_{i',j'} \in \boldsymbol{AE}^t$
Total Successor ($TS_{i,j}$)	$\frac{\lvert \boldsymbol{TS}_{i,j} \rvert}{\lvert \boldsymbol{V}_i \rvert - 1} \quad a_{i,j} \in \boldsymbol{AE}^t$
Total Successor Duration ($TSD_{i,j}$)	$\frac{1}{\sum_{a_{i,j'} \in \boldsymbol{V}_i} d_{i,j'}} \sum_{a_{i,j'} \in \boldsymbol{TS}_{i,j}} d_{i,j'} \quad a_{i,j} \in \boldsymbol{AE}^t$
Duration ($DT_{i,j}$)	$\frac{d_{i,j}}{\max d_{i',j'}} \quad a_{i,j}, a_{i',j'} \in \boldsymbol{AE}^t$
Resources Required ($RR_{i,j}$)	$\frac{1}{\lvert \boldsymbol{K} \rvert} \sum_{k=1}^{\lvert \boldsymbol{K} \rvert} \begin{cases} 1 & if\ r_{i,j,k} > 0 \\ 0 & otherwise \end{cases} \quad a_{i,j} \in \boldsymbol{AE}^t$
Average Resource Requirement ($AvgRR_{i,j}$)	$\frac{1}{\lvert \boldsymbol{K} \rvert} \sum_{k=1}^{\lvert \boldsymbol{K} \rvert} \frac{r_{i,j,k}}{RS_k} \quad a_{i,j} \in \boldsymbol{AE}^t$
Maximum Resource Requirement ($MaxRR_{i,j}$)	$\max \frac{r_{i,j,k}}{RS_k} \quad a_{i,j} \in \boldsymbol{AE}^t$
Minimum Resource Requirement ($MinRR_{i,j}$)	$\min \frac{r_{i,j,k}}{RS_k} \quad a_{i,j} \in \boldsymbol{AE}^t$
Critical Path Length (CPL_i)	$\frac{CP_i - \min CP_{i'}}{\max CP_{i'} - \min CP_{i'}} \quad a_{i,j}, a_{i',j'} \in \boldsymbol{AE}^t$

By parsing a PR tree, the sequencing of activities can be realized at different decision times, but two key technologies are still needed to obtain fitness in genetic evolution for evaluating PR. The first is how to transform the sequencing into schedule, that is, the selection of policy class. Based on the analysis of Section 2.2 and Villafáñez et al. (2019), RB-policy combined with critical path method is adopted in this paper. The function of critical path method is to generate a temporary schedule before using RB-policy, so as to dynamically update the attribute values to achieve better results. The second one is how to convert the scheduling results into the fitness function in GP. The R_1 and R_2 in SRCMPSP-NPI are affected by different distributions, and the calculation of R_1 and R_2 needs multiple simulations, resulting in a great increase in training time. Therefore, during the evolution of HH-FGP, the fitness calculation of PRs only depends on the average of Q_1 and Q_2 under multiple instances in the training set. Because there are only two objectives and the calculation of crowding distance is simple, NSGA-II (Deb et al., 2002) is adopted to realize the transformation from objective function value to fitness.

4.2 Genetic search

The iteration in GP is based on genetic evolution, so the genetic operators play an important role in its search process, including selection, crossover and mutation. In HH-FGP, the selection operator chooses tournament selection (Blickle, 2000). The mutation operator is designed to

discriminate mutation, because no other search causes the change of discriminant. The mutation can be described as that when the generated random number is less than the mutation rate p_m, the original discriminant "*fall*" becomes "*rise*", and the mutation rate is set larger than the traditional GP. The crossover operator, as the main search of tree structure, adopts the commonly subtree crossover (Chand et al., 2018) as shown in Fig.4. However, different from the existing GP, in HH-FGP, the sampling evolution stage needs to control that the maximum depth of the parent tree and the offspring tree in each sub-population is the same, and it also needs to control that the maximum depth of the crossed tree should be within the effective range in the filtering evolution. It leads to the design of a node verification procedure based on the traditional random node selection, as shown in Algorithm I, that is, it is executed repeatedly until reasonable nodes are selected. In Algorithm I, when the maximum depth is equal to the minimum depth in the hyper-parameter, this verification is suitable for the sampling evolution.

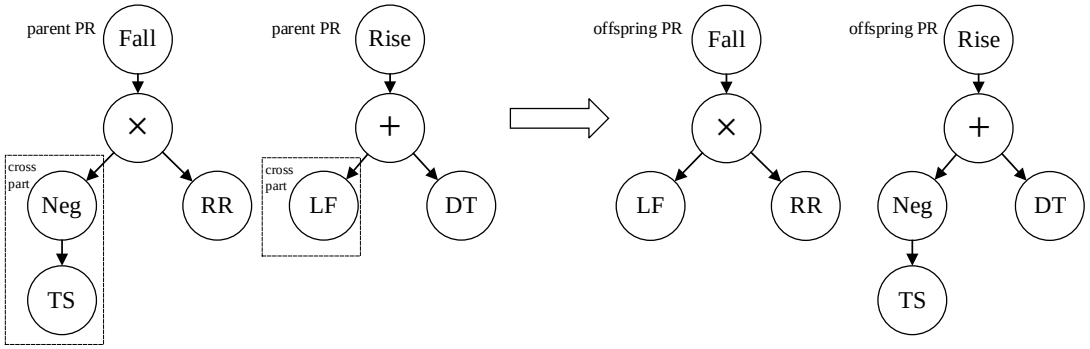

Fig.4 The example of subtree crossover

Algorithm I The verification of node selection in crossover

1: **Input**: the maximum depth max_d, the minimum depth min_d, the two parent PRs pr_1 and pr_2, the nodes selected in two PRs no_1 and no_2.
2: Calculate the depth of no_1 and no_2 as nD_1 and nD_2
3: Calculate the subtrees of no_1 and no_2 as sub_1 and sub_2, and record their maximum depth as aD_1 and aD_2
4: Calculate the remaining subtrees of sub_1 and sub_2 deleted from pr_1 and pr_2, and record their maximum depth as rD_1 and rD_2
5: **if** $nD_1 + aD_2 > max_d \,||\, nD_2 + aD_1 > max_d$
6: **return false**
7: **end if**
8: **if** $rD_1 < min_d$
9: **if** $nD_1 + aD_2 < min_d$
10: **return false**
11: **end if**
12: **end if**
13: **if** $rD_2 < min_d$
14: **if** $nD_2 + aD_1 < min_d$
15: **return false**
16: **end if**
17: **end if**

18: **return true**

4.3 Local search

Local search can further improve the search ability of GP, and three neighbourhood structures suitable for the tree structure are proposed to control the search range shown in Fig.5. Like the crossover operator, the requirements of PR depth range in sampling and filtering evolution should be considered in local search. Therefore, this paper designs the relevant verifications. The first is the verification of whether the deletion local search can be used, as shown in Algorithm II because when some trees with special structure use the local search, PRs that meet the depth range cannot be generated. For example, assuming the PR in Fig.5 faces the case in which the minimum depth of hyper-parameter is 3, the deletion neighbourhood cannot be used. Then, three local searches are described as follows.

- *Subtree replacement local search*: In this local search, a randomly generated subtree replaces a part of the original PR, which may produce a new structure that does not exist in the current population. In order to ensure that the new PR is still within the specified depth range, the depth range of the generated subtree needs to be controlled according to Algorithm III.
- *Node replacement local search*: this local search replaces a node of the original PR by a randomly generated node. In its execution, it only needs to ensure that the child node number of the new node is the same as the original one without other verification. If the replacement node is "If", its first child node is changed from "0" to "1" to realize the calculation in Table 2.
- *Subtree deletion local search*: this local search deletes the part of the original PR on the premise that two conditions are met. Firstly, the maximum depth of the new PR after being deleted needs to be within the specified range. Secondly, when the parent node of the deleted node has only one child node, an attribute node is randomly selected to replace the original node or subtree.

Algorithm II The verification of using deletion local search

1: **Input**: the minimum depth min_d, the original PR *pr*, the attribute set ***Att***.
2: Calculate the maximum depth of *pr* as D_{max}
3: **if** $D_{max} == min_d$
4: Calculate the number of nodes in *pr* as num_n
5: **if** $num_n == min_d+1$
6: **return false**
7: **end if**
8: Get the node set with depth 2 as $\boldsymbol{set_2}$
9: Calculate the child node number of the node with depth 1 as num_{d1}
10: Set $num_{cal}=0$
11: **for** *no*: $\boldsymbol{set_2}$
12: Gets the name of *no* as $name_{no}$
13: **if** ***Att***.contains($name_{no}$)
14: $num_{cal} = num_{cal}+1$
15: **end if**
16: **end for**
17: **if** $num_{cal} == num_{d1}-1$

18: **return false**
19: **end if**
20: **end if**
21: **return true**

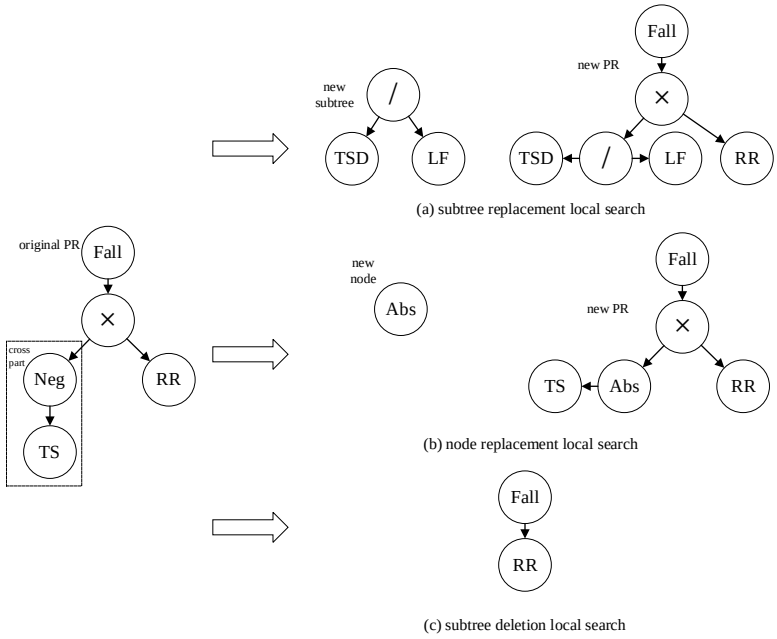

Fig.5 The examples of three local search

Algorithm III The control of new subtree depth range in subtree replacement local search

1: **Input**: the maximum depth max_d, the minimum depth min_d, the original PR *pr*, the selected node *no*.
2: Set max_{dnew}=0, min_{dnew}=0
3: Calculate the depth of *no* as *nD*, the subtree of *no* as *sub*
4: Calculate the maximum depth of *sub* as *aD*, the remaining subtree of *sub* deleted from *pr* as *rD*
5: **if** $rD < min_d$
6: $min_{dnew} = min_d - nD + 1$
7: **else**
8: min_{dnew}=1
9: **end if**
10: $max_{dnew} = max_d - nD$
11: **return** max_{dnew}, min_{dnew}

4.4 Filtering evaluation mechanism

As shown in Fig.3, after sampling evolution, each sub-population presents their sampling results, i.e., the optimal PR set at a certain depth. HH-FGP again uses the non-dominated relationship to eliminate some PRs, because PRs at different depths may dominate each other. When the non-dominated set is obtained, how to use the characteristics of PRs to filter is a key problem, especially in this paper, it is necessary to filter both the depth range and attribute set at the same time. Meanwhile, based on the analysis of Mei et al. (2017) in job shop scheduling, the

contribution of attributes is more important than that of its occurrence frequency. Therefore, a new filtering evaluation mechanism under multi-objective optimization is proposed in this paper. In depth range filtering, the number of PR at each maximum depth from the obtained non-dominated set is counted, and then upward and downward from the depth with the largest number of PRs is extended until it exceeds the judgment threshold *per*. In attribute filtering, the contribution of attribute is judged by replacement. When calculating this attribute, a minimal non-zero constant is used. Then, the contribution weight of this attribute under a PR is calculated according to Eq.(10) in minimization problem, and HH-FGP deletes the attribute in which contribution weight is lower than the average value of all attributes. The detailed description is shown in Algorithm IV.

$$value_{pr} = \begin{cases} |\boldsymbol{V}| & if\ v_{old} < v_{new} \quad \forall v \in \boldsymbol{V} \\ \sum_{v \in \boldsymbol{V}} \frac{v_{new} - v_{old}}{v_{\max} - v_{\min}} & otherwise \\ -|\boldsymbol{V}| & if\ v_{old} > v_{new} \quad \forall v \in \boldsymbol{V} \end{cases} \tag{10}$$

Where v_{new} and v_{old} represent the values before and after replacement under the object v respectively, and $\boldsymbol{V}$ represents the objective set.

Algorithm IV The filtering evaluation mechanism

1: **Input**: the PR non-dominated set $\boldsymbol{Set_{pr}}$, the attribute set $\boldsymbol{Att}$, the judgment threshold *per*, the maximum depth max_d, the minimum depth min_d
// The filtering of depth range
2: Set max_{df}=0, min_{df}=0, [] $\boldsymbol{Set_{depth}}$=**new int**[max_d-min_d]
3: **for** *pr*: $\boldsymbol{Set_{pr}}$
4: Obtain the maximum depth of *pr* as $maxD_{pr}$
5: $\boldsymbol{Set_{depth}}[maxD_{pr}\text{-}min_d] = \boldsymbol{Set_{depth}}[maxD_{pr}\text{-}min_d]+1$
6: **end for**
7: Find the position of the maximum value in $\boldsymbol{Set_{depth}}$ as *pos*
8: Set *num*=*pos*, *sum*=0, max_d=0
9: **if** $\boldsymbol{Set_{depth}}$.length-*pos*-1>*num*
10: *num*= $\boldsymbol{Set_{depth}}$.length-*pos*-1
11: **end if**
12: **for** *i*=0:*num*
13: **if** *i*==0
14: *sum*=*sum*+ $\boldsymbol{Set_{depth}}[pos]$
15: min_{df}=max_d+*pos*, max_{df}=max_d+*pos*
16: **else**
17: **if** *i*<=*pos*
18: *sum*=*sum*+ $\boldsymbol{Set_{depth}}$[*pos*-*i*]
19: min_{df}=max_d+*pos*-*i*
20: **end if**
21: **if** *i*<= $\boldsymbol{Set_{depth}}$.length-*pos*-1
22: *sum*=*sum*+ $\boldsymbol{Set_{depth}}$[*pos*+*i*]
23: max_{df}=max_d+*pos*+*i*
24: **end if**

```
    end if
    if sum> Set_pr.size()*per
        break
    end if
end for
// The filtering of attribute set
Set [] Set_value=new double [Att.size()], avg_value=0
for i=0: Att.size()
    Set value=0, att=Att.get(i)
    for pr: Set_pr
        if the attribute constituting pr contains att
            Calculate value_pr according to Equation 10
            value=value+ value_pr
        end if
    end for
    Set_value[i]= value, avg_value = avg_value +value
end for
Set Att_re, avg_value = avg_value / Att.size()
for i=0: Att.size()
    if Set_value[i]< avg_value
        Att_re.add(Att.get(i))
    end if
end for
Att.removeall(Att_re)
return max_df, min_df, Att
```

5 Numerical experiments

The performance of HH-FGP is fully verified through experimental setup and three groups of experiments, so there are four parts in this section. Firstly, the experimental setup in this study is introduced, including experimental environment, parameter selection of HH-FGP, benchmark composition and evaluation method. Secondly, because this research focuses on the scheduling under heuristic computation time, and the existing meta-heuristics have been proved to be inferior to the optimal PR in stochastic project scheduling (Chen et al., 2018), only the state-of-the-art PRs participate in the comparison with HH-FGP, including 16 traditional PRs and two hybrid PRs. Third, since no other improvements to GP are found in multi-project scheduling to our best knowledge, HH-FGP is compared with GP maintaining the same evolution, i.e., GP adopts improved evolution search operators, such as genetic search and local search, so as to verify the effectiveness of filtering. Finally, HH-FGP is an optimization framework for filtering attributes and depth range at the same time, so the impact of different filtering operations on its performance is further explored, that is, HH-FGP is compared with GP only filtering attributes and GP only filtering depth range respectively.

5.1 Experimental setup

In this research, all experiments are performed on an Intel Core i5-4200 quadcore processor computer with 2.50 GHz clock speed and 8 gigabyte RAM. The implement details of HH-FGP can be divided into two parts. Firstly, based on Fig.3 and Algorithm I to Algorithms IV, the training of HH-FGP is written in Java using MyEclipse 2017 compiler to obtain the evolved PRs, and the example of evolved PR is shown in Fig.6. Secondly, during the test, Math3.jar is used in each instance to generate simulation inputs under different distributions for PRs to perform scheduling and calculate objectives Eq.(1) to Eq.(4). Meanwhile, referring to Chen et al. (2021), the parameters of HH-FGP are shown in Table 4, including the population size pop_{size}, the maximum number of iterations max_{gen}, the crossover rate p_c, the mutation rate p_m and the judgment probability *per*. It can be seen that the parameters of HH-FGP are simple, by which the usability and practicability of HH-FGP are improved, and only the judgment probability is added. The sensitivity of this parameter is not high, because the PRs at each depth in the optimal PR set obtained by sampling evolution is fixed, when *per* floats up and down in a small range, the filtering results will not be affected. We believe that for the depth range filtering, it is sufficient when the effective PRs exceed 80% of the total PRs based on experience.

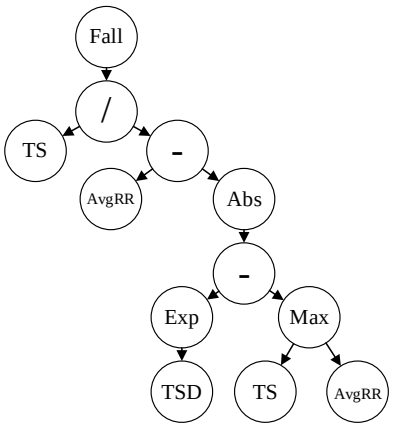

Fig.6 The example of evolved PR

Table 4 The parameters of HH-TGP

Variable	pop_{size}	max_{gen}	p_c	p_m	*per*
Value	300	50	0.9	0.2	0.8

In addition to the implementation and parameters of HH-FGP, another important part of the experimental design is the benchmark verification and evaluation method. Since SRCMPSP-NPI is established by Chen et al. (2019), these two parts are consistent with corresponding parts in that paper. In order to consider new project insertion in different situations, this benchmark first constructs 200 initial portfolios, in which each portfolio includes four projects with 30, 60, 90 and 120 non-dummy activities selected from PSPLIB (Kolisch & Sprecher, 1996) to simulate the situation on which the activity number in different projects is often inconsistent in practice. Then, each initial portfolio is scheduled under five different conditions, in which two new projects with different structures are inserted at different random times under each condition, and the allocation

of four resources needs to be considered. Therefore, the benchmark has 1000 different instances, and details such as the project selection, the new project insertion time and the maximum resource supply need to refer to Chen et al. (2019). Finally, in order to simulate stochastic duration, for each instance, the five most common distributions in SRC(M)PSP (Fang et al., 2015; Rostami et al., 2017; Chen et al., 2018; Chen et al., 2019; Chen et al. 2021) are selected to fully consider the stochastic environment under different variances shown in Table 5, including two low variance distributions (U1 and B1), two medium variance distributions (U2 and B2) and one high variance distribution (E).

Table 5 Five distributions of activity duration

Distribution type	Code	Range	Variance
Uniform distribution	U1	$U(d^*_{i,j}-\sqrt{d^*_{i,j}}, d^*_{i,j}+\sqrt{d^*_{i,j}})$	$d^*_{i,j}/3$
	U2	$U(0, 2d^*_{i,j})$	$(d^*_{i,j})^2/3$
Beta distribution	B1	$B(d^*_{i,j}/2, 2d^*_{i,j}, d^*_{i,j}/2-1/3, d^*_{i,j}-2/3)$	$d^*_{i,j}/3$
	B2	$B(d^*_{i,j}/2, 2d^*_{i,j}, 1/6, 1/3)$	$(d^*_{i,j})^2/3$
Exponential distribution	E	$E(d^*_{i,j})$	$(d^*_{i,j})^2$

In terms of evaluation methods, since SRCMPSP-NPI needs to evaluate the four objectives Q_1, Q_2, R_1 and R_2 at the same time, and their values may vary greatly when facing different objectives or different instances, the performance ranking is introduced (Wang et al., 2017; Chen et al., 2019). It relies on relativity to achieve evaluation. For example, in the minimization problem, for the objective v, the $rule_{num}$ methods participating in the evaluation are sorted according to the value obtained under the portfolio $\boldsymbol{P}$, so as to be transformed into an integer ranking value from 1 to $rule_{num}$. Then calculate the average ranking value under multiple instances and multiple objectives, as shown in Eq.(11). In particular, in order to complete the ranking calculation under R_1 and R_2, by referring to Chen et al. (2019), each distribution performs 10 simulations under each portfolio $\boldsymbol{P}$ to calculate SAD_i in Eq.(3) and Eq.(4). Although it is only executed 10 times in each instance, it can be seen from Eq.(11) that the ranking of each PR depends on the average value of 1000 instances, so it still has statistical accuracy.

$$rk_{rule} = \sum_{v\in \boldsymbol{V}}\Big(\sum_{\boldsymbol{P}\in \boldsymbol{P_{total}}} rk_{rule,v,\boldsymbol{P}} / |\boldsymbol{P_{total}}|\Big) / |\boldsymbol{V}| \quad rule = \{1, 2, ..., rule_{num}\} \tag{11}$$

Where rk_{rule} represents the average ranking value of the $rule$ method, $rk_{rule,v,\boldsymbol{P}}$ represents the objective v value of the $rule$ method under portfolio $\boldsymbol{P}$, and $\boldsymbol{P_{total}}$ represents all portfolios.

5.2 Comparison with existing PRs

With the determination of benchmarks and evaluation method, in order to verify the effectiveness of PRs evolved by HH-FGP, it is necessary to select existing PRs for comparison. Firstly, 20 traditional PRs (Wang et al., 2017; Chen et al., 2019) for SRCMPSP are investigated, but the RB-policy combined with critical path method adopted in this paper for generating

schedule dynamically updates the earliest start time of activities at each decision-making point, so that the functions of some PRs are consistent. Therefore, there are only 16 PRs with different functions by deleting the consistent PRs, as shown in Table 6.

Table 6 The traditional PRs for comparison

PR name	Abbreviation	PR name	Abbreviation
Minimum slack	MINSLK	Shortest operation duration first	SOF
Maximum slack	MAXSLK	Maximum operation duration first	MOF
Shortest activity from shortest project	SASP	Minimum late finish time	MINLFT
Longest activity from longest project	LALP	Maximum schedule pressure	MAXSP
Minimum total work content	MINTWK	Minimum worst case slack	MINWCS
Minimum total work content	MAXTWK	Criticality & resource utilization	WACRU
MAXTWK & earliest late start time	TWKLST	Maximum total successors	MS
First come first serve	FCFS	Maximum critical successors	MCS

Based on the above information, HH-FGP is used to perform 10 evolutions, and an optimal PR set is obtained under each evolution. Taking the first evolution as an example, in which 63 evolved PRs are obtained, the performance ranking is shown in Table 7, where rk_{avg} and rk_{min} respectively represent the average and minimum rk_{rule} of 63 PRs when compared with traditional PRs, and the comparison with the optimal traditional PR (OTPR) under five distributions is shown in Fig.7. It is worth mentioning that since rk_{rule} is the mean value under 1000 instances, the difference of rk_{avg} or rk_{min} between the two PRs in Table 7 is 0.1, indicating that their ranking is 100 different. Thus, the comparison of the performance ranking gap between two PRs in the test set can depend on the gap of rk_{avg} or rk_{min} in Table 7 multiplied by 1000. In addition, the statistical table under 10 evolutions is shown in Table 8, where *Num* is the experiment number, PR_{num} and PR_{per} respectively represent the obtained number of evolved PRs and the percentage of evolved PRs better than all traditional PRs in each evolution. The trend of average PR_{per} under the five distributions is shown in Fig.8.

Table 7 The rk_{min} and rk_{avg} of HH-FGP and traditional PRs

PR	U1		U2		B1		B2		E	
	rk_{avg}	rk_{min}	rk_{avg}	rk_{min}	rk_{avg}	rk_{min}	rk_{avg}	rk_{min}	rk_{avg}	rk_{min}
MINSLK	6.286	6.095	6.775	6.571	6.389	6.190	7.072	6.858	7.492	7.271
MAXSLK	12.759	12.707	12.068	12.032	12.664	12.621	11.808	11.768	11.306	11.230
SASP	8.376	8.330	8.690	8.653	8.369	8.320	9.008	8.976	9.202	9.163
LALP	9.788	9.723	10.134	10.036	9.923	9.852	10.494	10.385	10.838	10.695
MINTWK	8.018	7.873	7.897	7.772	8.075	7.937	7.978	7.851	7.794	7.685
MAXTWK	12.210	12.092	12.111	11.990	12.066	11.954	11.670	11.558	11.485	11.376
TWKLST	12.162	12.045	12.097	11.976	12.068	11.955	11.626	11.516	11.493	11.385
FCFS	7.382	7.059	7.882	7.544	7.543	7.225	8.082	7.727	8.594	8.203
SOF	11.627	11.507	11.195	11.076	11.565	11.448	10.670	10.553	10.137	10.031
MOF	11.964	11.894	11.549	11.486	11.509	11.449	11.480	11.419	11.473	11.377
MINLFT	8.686	8.536	8.466	8.307	8.651	8.505	8.018	7.851	7.966	7.777
MAXSP	7.267	7.103	7.641	7.453	7.374	7.209	8.022	7.820	8.061	7.834
MINWCS	6.294	6.107	6.792	6.591	6.411	6.221	7.103	6.891	7.515	7.298

WACRU	7.983	7.910	7.295	7.193	7.924	7.848	6.903	6.804	6.510	6.371
MS	6.654	6.519	7.000	6.847	6.803	6.665	7.294	7.125	7.255	7.065
MCS	7.035	6.882	7.288	7.126	7.100	6.951	7.596	7.421	7.926	7.729
HH-FGP	**6.233**	**5.509**	**6.205**	**5.650**	**6.293**	**5.609**	**6.370**	**5.824**	**6.337**	**5.880**

As can be seen from Table 7, when dealing with scheduling under different distributions, the OTPR changes, so the OTPR under different distributions refers to different traditional PRs in Fig.7 (U1, U2 and B1 are MINSLK, B2 and E are WACRU). However, no matter from the perspective of rk_{min} or rk_{avg}, HH-FGP is great superior to the OTPR. For example, under U1 distribution, although the percentage of HH-FGP in rk_{min} is only 9.6% higher than that of OTPR, the actual ranking of HH-FGP is 586 higher than that of OTPR in 1000 instances, which means that the PRs evolved by HH-FGP is better than OTPR in most instances. More importantly, it can be analysed from Table 8 and Fig.8 that nearly 70 evolved PRs can be obtained in each evolution, more than half of the PRs obtained by HH-FGP are better than all traditional PRs under low variance distribution (U1 and B1), and this percentage reaches 80% or even 90% in medium or high variance distribution. Undoubtedly, it can be concluded that *compared with traditional PRs, HH-FGP is an effective algorithm for stochastic multi-project scheduling, and is more suitable for highly uncertain environments.*

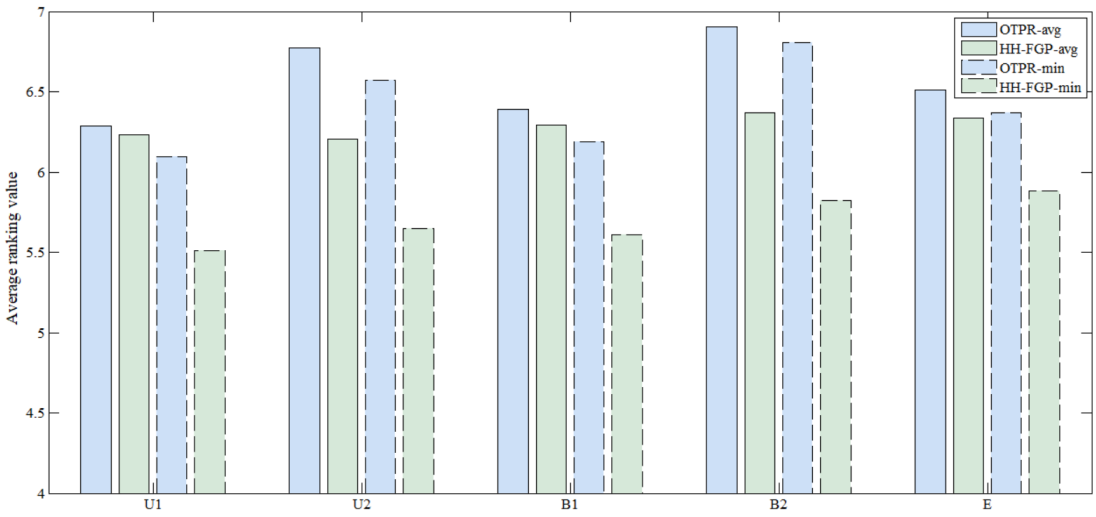

Fig.7 The rk_{avg} and rk_{min} of HH-FGP and the best traditional PR

Table 8 The statistical results of HH-FGP in comparison with traditional PRs

Num	PR_{num}	U1			U2			B1			B2			E		
		rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}
1	63	6.233	5.509	58.73%	6.205	5.650	95.24%	6.293	5.609	60.32%	6.370	5.824	95.24%	6.337	5.880	87.30%
2	59	6.273	5.472	50.85%	6.318	5.632	86.44%	6.354	5.596	49.15%	6.435	5.771	89.83%	6.382	5.792	86.44%
3	71	6.240	5.517	54.93%	6.312	5.681	91.55%	6.326	5.553	57.75%	6.417	5.787	91.55%	6.311	5.720	83.10%
4	60	6.228	5.537	53.33%	6.280	5.679	86.67%	6.296	5.585	53.33%	6.528	5.912	83.33%	6.403	5.771	80.00%
5	86	6.250	5.617	55.81%	6.332	5.730	86.05%	6.321	5.664	55.81%	6.445	5.818	94.19%	6.401	5.680	74.42%
6	73	6.292	5.476	52.05%	6.328	5.639	84.93%	6.366	5.588	52.05%	6.456	5.798	87.67%	6.378	5.765	83.56%
7	69	6.285	5.460	49.28%	6.330	5.640	92.75%	6.384	5.635	49.28%	6.492	5.863	92.75%	6.458	5.891	78.26%
8	67	6.312	5.544	47.76%	6.300	5.684	92.54%	6.383	5.657	49.25%	6.385	5.764	95.52%	6.317	5.710	82.09%
9	68	6.311	5.584	45.59%	6.324	5.668	85.29%	6.378	5.637	47.06%	6.427	5.746	92.56%	6.324	5.726	85.29%

10	69	6.291	5.491	46.38%	6.275	5.603	89.86%	6.369	5.566	47.83%	6.429	5.774	92.75%	6.387	5.829	82.61%
avg	68.5	6.272	5.521	51.47%	6.300	5.661	89.13%	6.347	5.609	52.18%	6.438	5.806	91.54%	6.370	5.776	82.31%

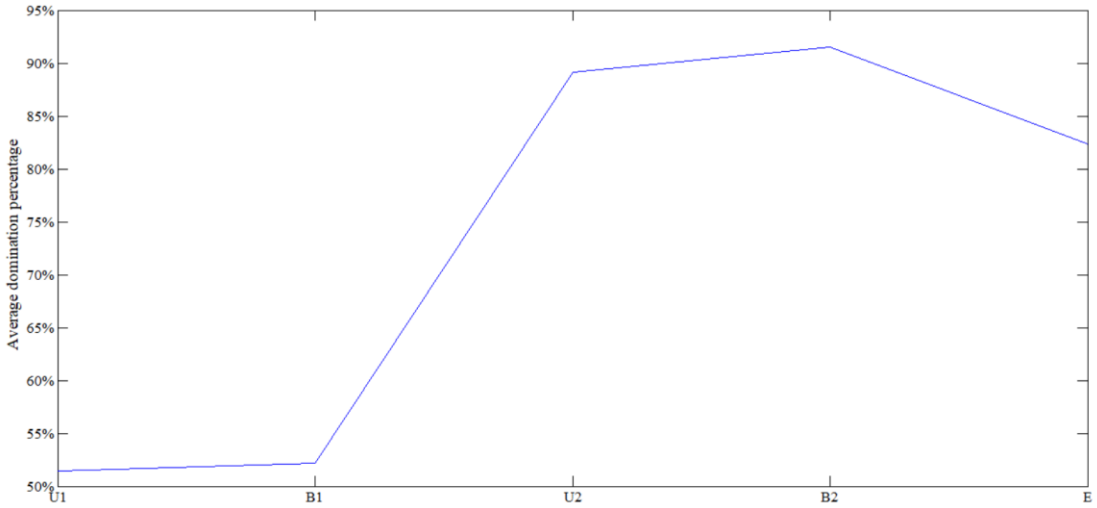

Fig.8 The variation trend of average PR_{per} in HH-FGP compared with traditional PRs under different distributions

In addition to the traditional PRs, Chen et al. (2019) proposed a hybrid method and obtained six effective hybrid PRs, which are state-of-the-art PRs for SRCMPSP. With similar steps, HH-FGP is compared with two hybrid PRs after deleting the same function to further verify its effectiveness, and this is also a comparison between HH-FGP and the hybrid method proposed by Chen et al. (2019). The comparison of HH-FGP and two hybrid PRs under the first evolution is shown in Table 9 and Fig.9, where WCS-SLK(FCFS-SLK) represents the mixture of MINWCS(FCFS) and MINSLK at different decision points, and the statistical results under the 10 evolutions are shown in Table 10 and Fig.10.

Table 9 The rk_{min} and rk_{avg} of HH-FGP and hybrid PRs

PR	U1		U2		B1		B2		E	
	rk_{avg}	rk_{min}	rk_{avg}	rk_{min}	rk_{avg}	rk_{min}	rk_{avg}	rk_{min}	rk_{avg}	rk_{min}
WCS-SLK	2.061	1.870	2.085	1.882	2.063	1.869	2.109	1.900	2.135	1.915
FCFS-SLK	1.995	1.822	2.033	1.841	1.990	1.809	2.034	1.823	2.054	1.813
HH-FGP	**1.890**	**1.742**	**1.841**	**1.732**	**1.894**	**1.760**	**1.822**	**1.709**	**1.780**	**1.698**

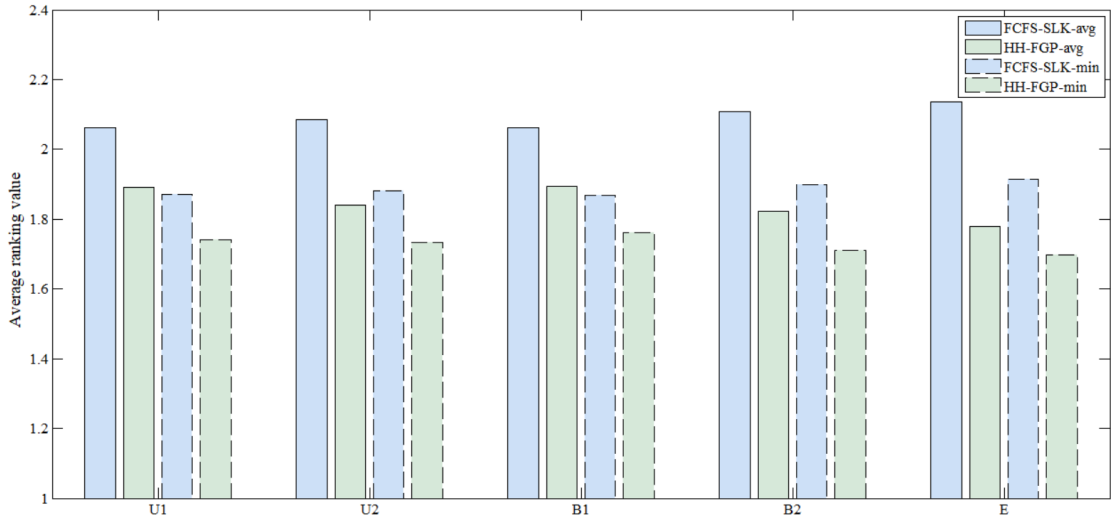

Fig.9 The rk_{avg} and rk_{min} of HH-FGP and the best hybrid PR

Table 10 The statistical results of HH-FGP in comparison with hybrid PRs

Num	U1			U2			B1			B2			E		
	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}
1	1.890	1.742	87.30%	1.841	1.732	95.24%	1.894	1.760	88.89%	1.822	1.709	95.24%	1.780	1.698	95.24%
2	1.897	1.753	77.97%	1.860	1.726	89.83%	1.908	1.765	76.27%	1.836	1.703	89.83%	1.788	1.672	89.83%
3	1.877	1.756	91.55%	1.841	1.736	97.18%	1.885	1.754	85.92%	1.816	1.708	97.18%	1.766	1.691	97.18%
4	1.901	1.757	77.67%	1.865	1.742	90.00%	1.908	1.766	75.00%	1.861	1.742	88.83%	1.800	1.707	90.00%
5	1.882	1.768	89.53%	1.851	1.741	95.35%	1.890	1.767	77.91%	1.829	1.735	95.35%	1.780	1.700	95.35%
6	1.903	1.749	83.56%	1.867	1.728	89.04%	1.911	1.762	80.82%	1.845	1.713	89.04%	1.791	1.684	89.04%
7	1.892	1.737	82.61%	1.854	1.727	92.75%	1.901	1.761	75.36%	1.839	1.722	92.75%	1.792	1.697	92.75%
8	1.893	1.754	82.09%	1.849	1.727	95.52%	1.900	1.764	76.12%	1.820	1.708	95.52%	1.766	1.676	95.52%
9	1.891	1.77	76.74%	1.848	1.736	91.18%	1.896	1.778	76.47%	1.825	1.719	91.18%	1.768	1.686	91.18%
10	1.887	1.75	81.16%	1.839	1.719	92.75%	1.893	1.759	76.81%	1.821	1.707	92.75%	1.776	1.674	92.75%
avg	1.891	1.754	83.02%	1.852	1.731	92.88%	1.899	1.764	78.96%	1.831	1.717	92.77%	1.781	1.689	92.88%

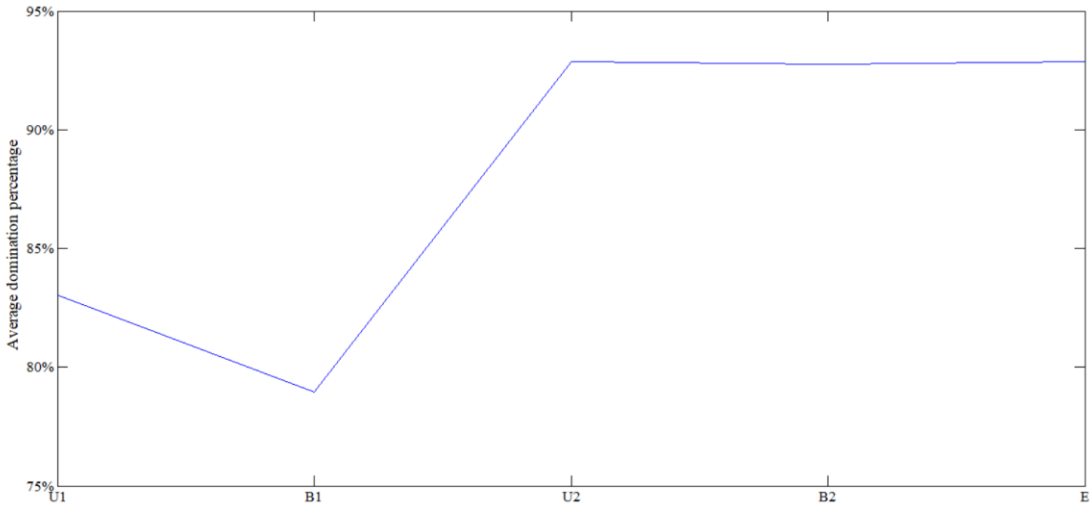

Fig.10 The variation trend of average PR_{per} in HH-FGP compared with hybrid PRs under different distributions

By analysing the information in Table 9 and Fig.9, it can be obtained that HH-FGP is still

superior to the optimal hybrid PR under any distribution. For example, under U1, the rk_{min} of HH-FGP is 128 higher than that of the FCFS-SLK in 1000 instances. However, the difference is that the average PR_{per} in low variance distribution is about 80%, and it has consistently exceeded 90% compared with hybrid PRs under medium and high variance distribution shown in Table 10 and Fig.10, which can further prove that *HH-FGP is still effective compared with hybrid PRs for SRCMPSP-NPI*.

5.3 The validation of attributes and depth range filtering

From the Section 5.2, it can be seen that HH-FGP is a very effective algorithm to solve SRCMPSP-NPI, but the effectiveness of the depth range control and attributes filtering, which are the core contribution in this paper, needs to be further verified. Therefore, GP is used to realize PRs evolution for comparison with HH-FGP. In order to ensure the fairness of comparison, GP evolution applied in this section also has other improved search operators except the filtering of attributes and depth range to ensure the same evolution process as HH-FGP, such as three local searches. At the same time, in order to ensure the same search, the parameters of GP should also be consistent with HH-FGP, so as to ensure that the training amount of the two algorithms is the same. The difference is that all 50 generations of GP are to obtain the optimal PR set, while the sampling and filtering evolution of HH-FGP occupy 25 generations respectively. Based on the comparison with traditional PRs, the statistical results of single GP under 10 evolutions are shown in Table 11, and the comparison with HH-FGP in average rk_{avg}, rk_{min} and PR_{per} is shown in Fig.11 and Fig.12.

Table 11 The statistical results of single GP in comparison with traditional PRs

Num	PR_{num}	U1			U2			B1			B2			E		
		rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}
1	68	6.483	5.520	42.65%	6.553	5.696	77.94%	6.530	5.567	45.49%	6.531	5.750	82.35%	6.581	5.678	64.71%
2	62	6.673	5.687	19.35%	6.804	5.746	54.84%	6.682	5.799	25.81%	6.747	5.810	66.13%	6.867	5.783	41.94%
3	39	6.575	5.632	28.21%	6.695	5.758	74.36%	6.600	5.740	38.64%	6.603	5.619	82.05%	6.774	5.900	61.54%
4	52	6.596	5.708	26.92%	6.652	5.728	78.85%	6.597	5.743	38.46%	6.579	5.599	86.54%	6.630	5.662	63.46%
5	76	6.508	5.692	39.47%	6.579	5.860	81.58%	6.546	5.713	47.37%	6.561	5.816	86.84%	6.531	5.797	72.37%
6	48	6.474	5.727	50.00%	6.627	5.803	75.00%	6.517	5.817	54.17%	6.543	5.721	79.17%	6.715	5.716	50.00%
7	56	6.496	5.586	41.07%	6.555	5.605	82.14%	6.521	5.684	51.79%	6.542	5.652	83.93%	6.549	5.726	69.64%
8	74	6.831	5.913	12.16%	7.019	5.991	51.35%	6.836	5.943	27.03%	6.919	5.994	59.46%	7.080	6.011	21.62%
9	49	6.664	5.826	22.45%	6.855	5.972	51.02%	6.707	5.839	30.61%	6.753	5.780	67.35%	6.895	6.057	38.78%
10	54	6.488	5.639	38.89%	6.571	5.707	77.78%	6.508	5.638	48.15%	6.564	5.729	83.33%	6.681	5.749	48.15%
avg	57.8	6.579	5.693	32.18%	6.691	5.787	70.49%	6.604	5.748	40.75%	6.634	5.747	77.72%	6.730	5.808	53.22%

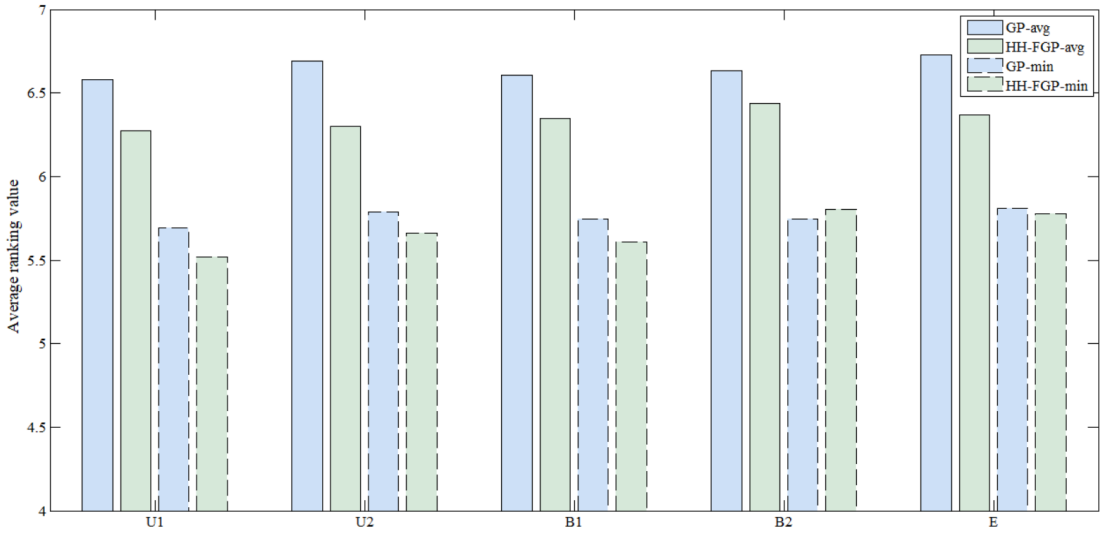

Fig.11 The average rk_{avg} and rk_{min} comparison between HH-FGP and GP

Through the data comparison at different levels, it can be seen that HH-FGP is more effective than GP. By comparing the average PR_{num} in Table 8 and Table 11, it can be seen that GP can get about 58 PRs on average under 10 evolutions, while HH-FGP can get about 68 PRs, which indicates that more evolved PRs can be obtained after filtering for decision makers to choose. More importantly, it can be seen from the following two accounts that the PRs obtained by HH-FGP is more effective. Firstly, it can be seen from Fig.11 that in terms of average rk_{min}, HH-FGP is better than GP except for B2 distribution, so it can be concluded that the better PR can be obtained after filtering in most distributions. Secondly and more significantly, from the perspective of average rk_{avg}, HH-FGP is 0.3 better than GP in different distributions, and even reaches 0.36 in high variance distribution, which means that the average ranking of PRs obtained by HH-FGP is 360 higher than GP in 1000 instances. Similar conclusions can also be drawn from Fig.12. For example, under the medium and low variance distribution, the average PR_{per} of HH-FGP is almost 20% higher than that of GP, which means that although HH-FGP obtains more PRs, the proportion of evolved PRs better than all traditional PRs is increased. At the same time, this proportion gap reaches 30% under the high variance distribution, which indicates that HH-FGP is more suitable for SRCMPSP-NPI with high variance distribution. It is worth mentioning that through HH-FGP evaluation, the important attributes applicable to SRCMPSP-NPI are "*TS*", "*TSD*", "*AvgRR*" and "*MaxRR*", and the appropriate depth range is 4 to 6. Through this part of the experiment, it can be shown that HH-FGP can obtain more effective PRs under the same search (even its evolution of 25 generations is for sampling), so that *the effectiveness of depth range and attribute filtering is verified*.

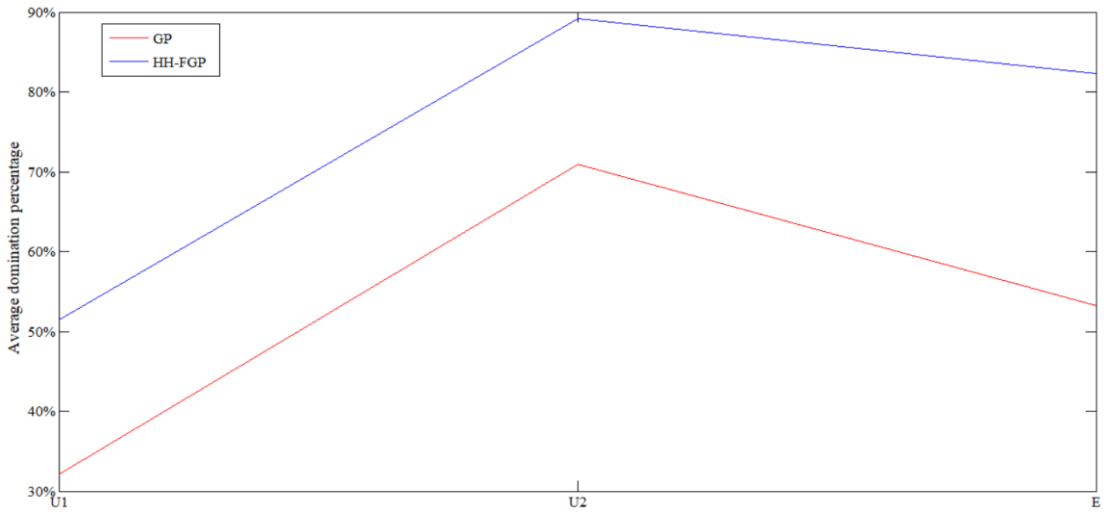

Fig.12 The average PR_{per} comparison between HH-FGP and GP under different variance distributions

5.4 The effect analysis of the depth range and attributes filtering

The effectiveness of attribute and depth range filtering is verified in Section 5.3, but the evolution of HH-FGP filters two parameters at the same time, so the necessity of filtering two parameters needs to be further explored, that is, assuming only depth range or attribute filtering, how does the performance of HH-FGP change. Therefore, in this section, the filtering in HH-FGP is divided into two parts, namely GP only with the depth range filtering (HH-DFGP) and GP only with the attributes filtering (HH-AFGP), to complete PRs evolution. Similarly, except that HH-DFGP and HH-AFGP only perform depth range and attributes filtering when executing Algorithm IV, other evolution processes and parameters are consistent with HH-FGP. With the same experimental process, the statistical results of HH-DFGP and HH-AFGP under 10 evolutions are shown in Table 12 and Table 13, and their comparisons on the average rk_{avg}, rk_{min} and PR_{per} are shown in Fig.13 and Fig.14.

Table 12 The statistical results of HH-DFGP in comparison with traditional PRs

Num	PR_{num}	U1			U2			B1			B2			E		
		rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}
1	41	6.663	5.653	21.95%	6.699	5.831	63.41%	6.704	5.786	21.95%	6.735	5.837	75.61%	6.688	5.852	41.46%
2	37	6.580	5.876	35.14%	6.539	5.871	83.78%	6.601	5.951	37.84%	6.556	5.853	86.49%	6.541	5.821	64.86%
3	37	6.425	5.632	45.95%	6.513	5.738	81.08%	6.471	5.695	51.35%	6.516	5.765	83.78%	6.549	5.699	64.86%
4	53	6.555	5.693	32.08%	6.597	5.810	81.13%	6.606	5.781	39.62%	6.656	5.842	77.36%	6.623	5.699	47.17%
5	48	6.597	5.784	18.75%	6.656	5.834	83.33%	6.639	5.837	35.42%	6.695	5.746	89.58%	6.608	5.764	75.00%
6	49	6.549	5.586	32.65%	6.611	5.696	75.51%	6.616	5.690	36.73%	6.678	5.908	75.51%	6.621	5.849	57.14%
7	39	6.525	5.687	41.03%	6.570	5.728	84.62%	6.595	5.780	48.72%	6.560	5.691	84.62%	6.619	5.780	53.85%
8	39	6.660	5.745	35.90%	6.751	5.862	74.36%	6.699	5.868	41.03%	6.784	5.919	74.36%	6.824	5.995	64.10%
9	47	6.468	5.676	38.30%	6.521	5.750	80.85%	6.508	5.759	42.55%	6.528	5.689	87.23%	6.498	5.666	68.09%
10	50	6.359	5.438	36.00%	6.405	5.523	90.00%	6.431	5.547	42.00%	6.511	5.697	84.00%	6.436	5.616	52.00%
avg	44	6.538	5.677	33.78%	6.586	5.764	79.81%	6.587	5.769	39.72%	6.622	5.795	81.85%	6.601	5.774	58.85%

Table 13 The statistical results of HH-AFGP in comparison with traditional PRs

Num	PR_{num}	U1			U2			B1			B2			E		
		rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}	rk_{avg}	rk_{min}	PR_{per}
1	39	6.500	5.684	33.33%	6.552	5.624	87.18%	6.541	5.790	41.03%	6.555	5.667	89.74%	6.553	5.554	71.79%
2	47	6.620	5.663	40.43%	6.758	5.813	74.47%	6.652	5.720	46.81%	6.720	5.761	78.72%	6.792	5.736	55.32%
3	37	6.393	5.752	43.24%	6.412	5.790	86.49%	6.431	5.881	45.95%	6.481	5.956	89.19%	6.491	5.837	64.86%
4	46	6.472	5.626	23.91%	6.453	5.733	89.13%	6.530	5.767	30.43%	6.540	5.832	93.48%	6.461	5.764	60.87%
5	38	6.491	5.642	50.00%	6.644	5.810	71.05%	6.540	5.684	50.00%	6.597	5.887	73.68%	6.701	5.836	52.63%
6	43	6.628	5.739	30.23%	6.687	5.797	67.44%	6.665	5.783	34.88%	6.703	5.881	72.09%	6.698	5.853	62.79%
7	36	6.387	5.639	36.11%	6.438	5.702	80.56%	6.435	5.758	36.11%	6.490	5.692	80.56%	6.466	5.789	55.56%
8	46	6.465	5.709	41.30%	6.456	5.695	86.96%	6.501	5.788	45.65%	6.507	5.731	84.78%	6.396	5.575	76.09%
9	38	6.619	5.957	28.95%	6.669	5.938	76.32%	6.672	6.022	31.58%	6.710	6.037	71.05%	6.673	5.987	34.21%
10	46	6.525	5.510	32.61%	6.579	5.695	71.74%	6.560	5.567	36.96%	6.635	5.796	78.26%	6.617	5.758	39.13%
avg	41.6	6.510	5.692	36.01%	6.565	5.760	79.13%	6.553	5.776	39.94%	6.594	5.824	81.16%	6.585	5.769	57.33%

Through the analysis and comparison of relevant data, the following results can be obtained. Firstly, based on the PR_{num} in Table 8, Table 12 and Table 13, it can be seen that the number of PRs obtained by both HH-DFGP and HH-AFGP decreases sharply, that is, they only reach about 60% of HH-FGP or even worse than GP. Therefore, it can be concluded that if only a single parameter (attributes or depth range) is filtered, less PRs will be obtained. Secondly, it can be seen from Fig.13 that with respect to average rk_{min}, HH-DFGP and HH-AFGP can achieve the effect similar to HH-FGP under B2 and E distributions, but inferior to HH-FGP in other distributions. From the perspective of average rk_{avg}, HH-FGP is still better than HH-DFGP and HH-AFGP in all distributions, and it is found that HH-DFGP and HH-AFGP are even similar to GP by comparing Fig.11 and Fig.13. Thus, it can be obtained that only filtering either depth range or attributes can only improve the ability to obtain the optimal PR under medium or high variance distribution, but the average performance of the evolved PRs is not improved. Finally, as can be seen from Fig.14, compared with GP, although the average PR_{per} of HH-DFGP and HH-AFGP is better than GP under different variance distributions, the number of effective PRs obtained by them is not much different from GP due to the reduction of PR_{num}. On the contrary, HH-FGP can not only increase the number of evolved PRs, but also greatly improve average PR_{per} as shown in Fig.14. In a word, if only depth range or attributes filtering is adopted, the effect is just that it is easy to select an effective PR in a small range and improve the performance of the evolved optimal PR under medium or high variance distribution. Therefore, the experiments in this section show *the necessity of simultaneous filtering of both depth range and attributes, and verify the effectiveness of HH-FGP again.*

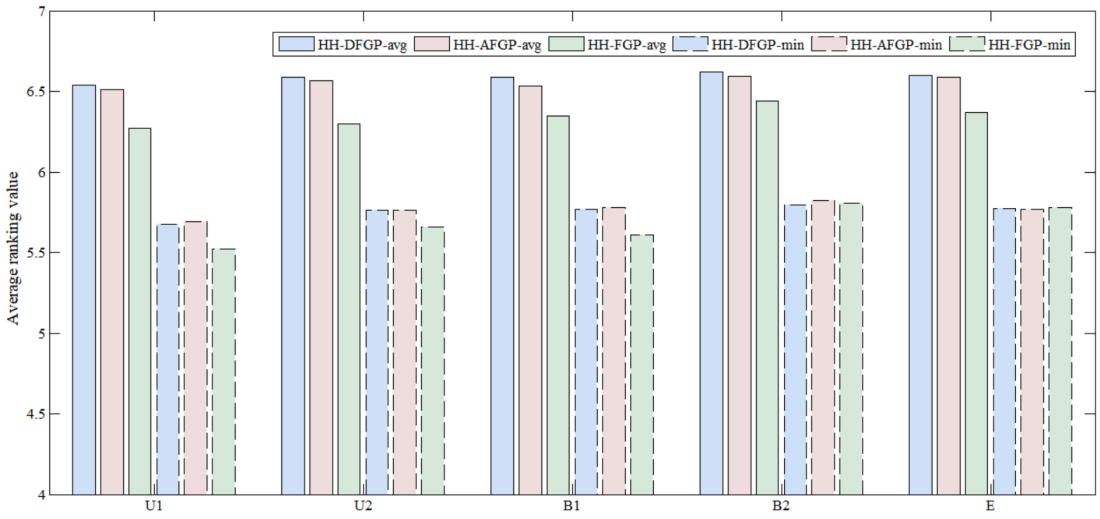

Fig.13 The average rk_{avg} and rk_{min} comparison of HH-FGP, HH-DFGP and HH-AFGP

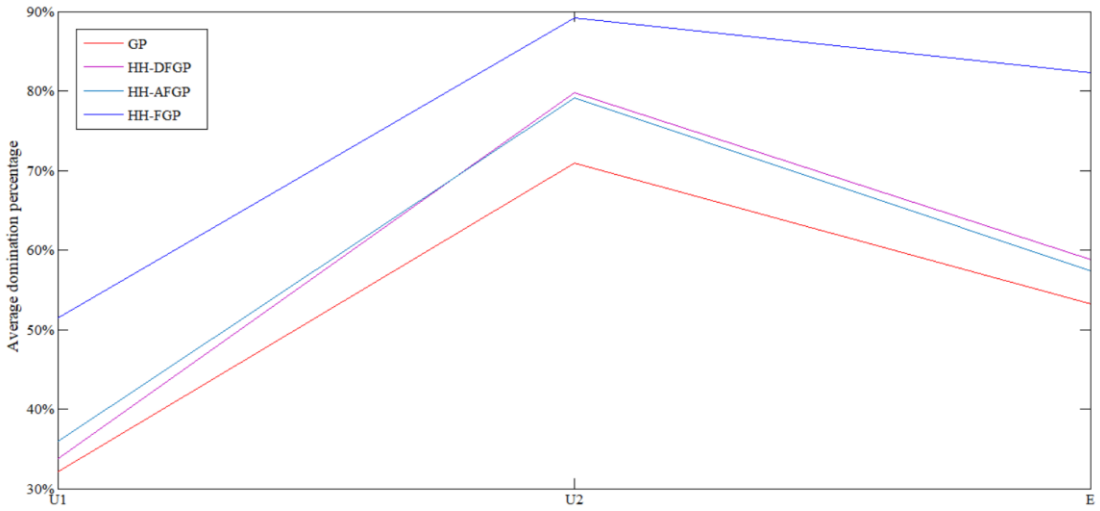

Fig.14 The average PR_{per} comparison of HH-FGP, HH-DFGP and HH-AFGP under different variance distributions

5.5 Discussion on performance indicator

It is well-noticed that efficiency is an important performance indicator of an optimization algorithm beside the quality of solutions, but it is difficult to qualitatively compare the convergence and efficiency due to the following two factors. From the perspective of convergence, since the result of each generation in HH-FGP is a set of PRs with non-dominated performance (Pareto frontier), there is no suitable evaluation or intuitive curve to describe the change of Pareto frontier with generation. For example, the iteration result of a certain generation maybe adds a non-dominated PR, updates a PR in the previous generation PR set, or even deletes a variety of PRs in the previous generation PR set due to the acquisition of a new dominant PR.

On the other hand, it is also very difficult to qualitatively prove the efficiency of HH-FGP through training time, because the gap between GP, HH-FGP, HH-DFGP and HH-AFGP in Section 5.3 and Section 5.4 can be ignored due to the same training parameters and search amount. For example, in our current results, each training process takes nearly 18 hours due to the very large training data, and the time difference between them is no more than 5 minutes. However,

when using evolved PRs for decision-making, the response is very rapid. For example, in our results, the time consumption of a PR scheduling of an instance is less than 50ms.

Thus, how to evaluate efficiency performance of the proposed algorithm is a future investigation question, though it can be seen from Sections 5.3 and 5.4 that HH-FGP has a good quality of solutions.

6 Conclusion

In this paper, a novel HH-FGP framework is proposed to solve SRCMPSP-NPI more effectively by filtering both the depth range and attributes simultaneously. After exploring and implementing the framework, the existing research is extended from the following two aspects. Firstly, it provides a hyper-heuristic method for evolving PR to solve the multi-disturbance multi-objective stochastic scheduling problem through obtaining more effective PRs, especially the stochastic project scheduling, which is helpful for the scheduling optimization in the heuristic computation time. Secondly, it extends the existing hyper-heuristic filtering method, that is, it should filter two parts at the same time, rather than only considering attributes.

As for innovations in implementation, firstly, under ensuring the same search, this unique framework divides the evolution of traditional GP into two parts, namely sampling and filtering evolution, so as to obtain an optimal PR set under the consideration of the depth range and attributes and realize the filtering PR evolution. Secondly, in the sampling and filtering evolution, the existing genetic search and local search are improved to control evolved PRs without violating the depth constraint, and they also provide a search method to meet the need of controlling the depth range. Finally, a multi-objective filtering evaluation mechanism is designed, which realizes the depth range and attribute filtering under sampling by designing judgment threshold and combining contribution weight. Tested with a large number of experiments, HH-FGP verifies its effectiveness by comparing with the existing PRs and GP and impacts of applying the two filtering methods separately.

It is worth mentioning that when HH-FGP is applied to the actual SRCMPSP-NPI represented by assembly production, it only needs to perform training regularly on the basis of collecting duration and new project insertion information (in order to realize the fitting of distribution), and the scheduling PRs can be selected and updated to realize the decision-making in this cycle. Meanwhile, if the schedule generation policy and basic attributes in Table 3 are replaced, HH-FGP can easily realize PR evolution for other scheduling problems in stochastic environment, such as job shop and flow shop. It can be shown that the study of HH-FGP is very meaningful.

In the future work, HH-FGP will be studied from two aspects to realize its expansion. Firstly, SRCMPSP-NPI is a stochastic project scheduling problem considering two disturbances. When facing more disturbance factors such as resource disruptions and more scheduling scenarios such as multi-skill resource constrained project scheduling, whether HH-FGP can ensure the same effectiveness or further improve the framework needs to be explored. Second, HH-FGP is a search framework connected by two parts. For the filtering evolution, the search process in sampling evolution is unknown, but it may produce effective guidance information. Therefore, reinforcement learning can be combined into HH-FGP to realize synchronous dynamic update of

sampling and filtering evolution, resulting in its further improvement.

Acknowledgements

This research is supported by Sichuan Science and Technology Program (Grant number 2020ZDZX0015).

References

Alipouri, Y., Sebt, M. H., Ardeshir, A., & Zarandi, M. H. F. (2020). A mixed-integer linear programming model for solving fuzzy stochastic resource constrained project scheduling problem. *Operational Research*, 20(1), 197-217.

Alipouri, Y., Sebt, M. H., Ardeshir, A., & Chan, W. T. (2019). Solving the FS-RCPSP with hyper-heuristics: A policy-driven approach. *Journal of the Operational Research Society*, 70(3), 403-419.

Anagnostopoulos, K., & Koulinas, G. (2012). Resource-constrained critical path scheduling by a GRASP-based hyperheuristic. *Journal of Computing in Civil Engineering*, 26(2), 204-213.

Ashtiani, B., Leus, R., & Aryanezhad, M. B. (2011). New competitive results for the stochastic resource-constrained project scheduling problem: Exploring the benefits of pre-processing. *Journal of Scheduling*, 14(2), 157-171.

Asta, S., Karapetyan, D., Kheiri, A., Özcan, E., & Parkes, A. J. (2016). Combining Monte-Carlo and hyper-heuristic methods for the multi-mode resource-constrained multi-project scheduling problem. *Information Sciences*, 373, 476-498.

Ballestín, F. (2007). When it is worthwhile to work with the stochastic RCPSP?. *Journal of Scheduling*, 10(3), 153-166.

Ballestin, F., & Leus, R. (2009). Resource‐constrained project scheduling for timely project completion with stochastic activity durations. *Production and Operations Management*, 18(4), 459-474.

Blickle, T. (2000). Tournament selection. *Evolutionary Computation*, 1, 181-186.

Branke, J., Nguyen, S., Pickardt, C. W., & Zhang, M. (2015a). Automated design of production scheduling heuristics: A review. *IEEE Transactions on Evolutionary Computation*, 20(1), 110-124.

Branke, J., Hildebrandt, T., & Scholz-Reiter, B. (2015b). Hyper-heuristic evolution of dispatching rules: A comparison of rule representations. *Evolutionary Computation*, 23(2), 249-277.

Browning, T. R., & Yassine, A. A. (2010). Resource-constrained multi-project scheduling: Priority rule performance revisited. *International Journal of Production Economics*, 126(2), 212-228.

Burke, E. K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., & Qu, R. (2013). Hyper-heuristics: A survey of the state of the art. *Journal of the Operational Research Society*, 64(12), 1695-1724.

Chand, S., Huynh, Q., Singh, H., Ray, T., & Wagner, M. (2018). On the use of genetic programming to evolve priority rules for resource constrained project scheduling problems. *Information Sciences*, 432, 146-163.

Chand, S., Singh, H., & Ray, T. (2019a). Evolving heuristics for the resource constrained project scheduling problem with dynamic resource disruptions. *Swarm and Evolutionary Computation*, 44, 897-912.

Chand, S., Singh, H., & Ray, T. (2019b). Evolving rollout-justification based heuristics for resource constrained project scheduling problems. *Swarm and Evolutionary Computation*, 50, 100556.

Chen, H., Ding, G., Zhang, J., & Qin, S. (2019). Research on priority rules for the stochastic resource constrained multi-project scheduling problem with new project arrival. *Computers & Industrial Engineering*, 137, 106060.

Chen, H., Ding, G., Qin, S., & Zhang, J. (2021). A hyper-heuristic based ensemble genetic programming approach for stochastic resource constrained project scheduling problem. *Expert Systems with Applications*, 167, 114174.

Chen, Z., Demeulemeester, E., Bai, S., & Guo, Y. (2018). Efficient priority rules for the stochastic resource-constrained project scheduling problem. *European Journal of Operational Research*, 270(3), 957-967.

Creemers, S. (2015). Minimizing the expected makespan of a project with stochastic activity durations under resource constraints. *Journal of Scheduling*, 18(3), 263-273.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2), 182-197.

Fang, C., Kolisch, R., Wang, L., & Mu, C. (2015). An estimation of distribution algorithm and new computational results for the stochastic resource-constrained project scheduling problem. *Flexible Services and Manufacturing Journal*, 27(4), 585-605.

Golenko-Ginzburg, D., & Gonik, A. (1997). Stochastic network project scheduling with non-consumable limited resources. *International Journal of Production Economics*, 48(1), 29-37.

Habibi, F., Barzinpour, F., & Sadjadi, S. (2018). Resource-constrained project scheduling problem: review of past and recent developments. *Journal of Project Management*, 3(2), 55-88.

Hartmann, S., & Briskorn, D. (2010). A survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of Operational Research*, 207(1), 1-14.

Hildebrandt, T., & Branke, J. (2015). On using surrogates with genetic programming. *Evolutionary Computation*, 23(3), 343-367.

Hildebrandt, T., Heger, J., & Scholz-Reiter, B. (2010, July). 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).

Igelmund, G., & Radermacher, F. J. (1983). Preselective strategies for the optimization of stochastic project networks under resource constraints. *Networks*, 13(1), 1-28.

Kolisch, R. (1996). Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. *European Journal of Operational Research*, 90(2), 320-333.

Kolisch, R., & Sprecher, A. (1996). PSPLIB—A project scheduling problem library. *European Journal of Operational Research*, 96, 205–216.

Koulinas, G. K., & Anagnostopoulos, K. P. (2012). Construction resource allocation and leveling using a threshold accepting–based hyperheuristic algorithm. *Journal of Construction Engineering and Management*, 138(7), 854-863.

Koulinas, G., Kotsikas, L., & Anagnostopoulos, K. (2014). A particle swarm optimization based hyper-heuristic algorithm for the classic resource constrained project scheduling problem. *Information Sciences*, 277, 680-693.

Kühn, M., Völker, M., & Schmidt, T. (2020). An Algorithm for Efficient Generation of

Customized Priority Rules for Production Control in Project Manufacturing with Stochastic Job Processing Times. *Algorithms*, 13(12), 337.

Lamas, P., & Demeulemeester, E. (2016). A purely proactive scheduling procedure for the resource-constrained project scheduling problem with stochastic activity durations. *Journal of Scheduling*, 19(4), 409-428.

Lin, J., Zhu, L., & Gao, K. (2020). A genetic programming hyper-heuristic approach for the multi-skill resource constrained project scheduling problem. *Expert Systems with Applications*, 140, 112915.

Lin, J., Wang, Z. J., & Li, X. (2017). A backtracking search hyper-heuristic for the distributed assembly flow-shop scheduling problem. *Swarm and Evolutionary Computation*, 36, 124-135.

Luke, S., & Panait, L. (2001, July). A survey and comparison of tree generation algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)* (pp. 81-88). Morgan Kaufmann San Francisco, California, USA.

Mei, Y., Zhang, M., & Nyugen, S. (2016, July). Feature selection in evolving job shop dispatching rules with genetic programming. In *Proceedings of the Genetic and Evolutionary Computation Conference* 2016 (pp. 365-372).

Mei, Y., Nguyen, S., Xue, B., & Zhang, M. (2017). An efficient feature selection algorithm for evolving job shop scheduling rules with genetic programming. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 1(5), 339-353.

Ma, W., Che, Y., Huang, H., & Ke, H. (2016). Resource-constrained project scheduling problem with uncertain durations and renewable resources. *International Journal of Machine Learning and Cybernetics*, 7(4), 613-621.

Masood, A., Chen, G., & Zhang, M. (2021, June). Feature Selection for Evolving Many-Objective Job Shop Scheduling Dispatching Rules with Genetic Programming. In *2021 IEEE Congress on Evolutionary Computation (CEC)* (pp. 644-651).

Möhring, R. H., Radermacher, F. J., & Weiss, G. (1984). Stochastic scheduling problems I—General strategies. *Mathematical Methods of Operations Research*, 28(7), 193-260.

Möhring, R. H., Radermacher, F. J., & Weiss, G. (1985). Stochastic scheduling problems II-set strategies. *Mathematical Methods of Operations Research*, 29(3), 65-104.

Payne, J. H. (1995). Management of multiple simultaneous projects: a state-of-the-art review. *International Journal of Project Management*, 13(3), 163-168.

Pellerin, R., Perrier, N., & Berthaut, F. (2020). A survey of hybrid metaheuristics for the resource-constrained project scheduling problem. *European Journal of Operational Research*, 280(2), 395-416.

Pritsker, A. A. B., Waiters, L. J., & Wolfe, P. M. (1969). Multiproject scheduling with limited resources: A zero-one programming approach. *Management Science*, 16(1), 93-108.

Radermacher, F. J. (1981). Cost-dependent essential systems of ES-strategies for stochastic scheduling problems. *Methods of Operations Research*, 42, 17-31.

Rostami, S., Creemers, S., & Leus, R. (2018). New strategies for stochastic resource-constrained project scheduling. *Journal of Scheduling*, 21(3), 349-365.

Sallam, K. M., Chakrabortty, R. K., & Ryan, M. J. (2021). A reinforcement learning based multi-method approach for stochastic resource constrained project scheduling problems. *Expert Systems with Applications*, 169, 114479.

Satic, U., Jacko, P., & Kirkbride, C. (2020). Performance evaluation of scheduling policies for the

dynamic and stochastic resource-constrained multi-project scheduling problem. *International Journal of Production Research*, 1-13.

Stork, F. (2001) Stochastic resource-constrained project scheduling (Ph.D. thesis), *Technical University of Berlin*.

Tsai, Y. W., & Gemmill, D. D. (1998). Using tabu search to schedule activities of stochastic resource-constrained projects. *European Journal of Operational Research*, 111(1), 129-141.

Van de Vonder, S., Demeulemeester, E., & Herroelen, W. (2007). A classification of predictive-reactive project scheduling procedures. *Journal of Scheduling*, 10(3), 195-207.

Van Eynde, R., & Vanhoucke, M. (2020). Resource-constrained multi-project scheduling: benchmark datasets and decoupled scheduling. *Journal of Scheduling*, 23(3), 301-325.

Villafáñez, F., Poza, D., López-Paredes, A., Pajares, J., & del Olmo, R. (2019). A generic heuristic for multi-project scheduling problems with global and local resource constraints (RCMPSP). *Soft Computing*, 23(10), 3465-3479.

Wang, X., Chen, Q., Mao, N., Chen, X., & Li, Z. (2015). Proactive approach for stochastic RCMPSP based on multi-priority rule combinations. *International Journal of Production Research*, 53(4), 1098-1110.

Wang, Y., He, Z., Kerkhove, L. P., & Vanhoucke, M. (2017). On the performance of priority rules for the stochastic resource constrained multi-project scheduling problem. *Computers & Industrial Engineering*, 114, 223-234.

Zhu, L., Lin, J., Li, Y. Y., & Wang, Z. J. (2021). A decomposition-based multi-objective genetic programming hyper-heuristic approach for the multi-skill resource constrained project scheduling problem. *Knowledge-Based Systems*, 225, 107099.

Zhang, F., Mei, Y., & Zhang, M. (2019, July). A two-stage genetic programming hyper-heuristic approach with feature selection for dynamic flexible job shop scheduling. In *Proceedings of the Genetic and Evolutionary Computation Conference* (pp. 347-355).

Zhang, F., Mei, Y., Nguyen, S., & Zhang, M. (2020). Evolving scheduling heuristics via genetic programming with feature selection in dynamic flexible job-shop scheduling. *IEEE Transactions on Cybernetics*, 51(4), 1797-1811.