Reference Point Adaption Method for Genetic Programming Hyper-heuristic in Many-Objective Job Shop Scheduling

Abstract. Job Shop Scheduling (JSS) is considered to be one of the most significant combinatorial optimization problems in practice. It is widely evidenced in the literature that JSS usually contains many (four or more) potentially conflicting objectives. One of the promising and successful approaches to solve the JSS problem is Genetic Programming Hyper-Heuristic (GP-HH). This approach automatically evolves dispatching rules for solving JSS problems. This paper aims to evolve a set of effective dispatching rules for many-objective JSS with genetic programming and NSGA-III. NSGA-III originally defines uniformly distributed reference points in the objective space. Thus, there will be few reference points with no Pareto optimal associated with them; especially, in the case of discrete and non-uniform Pareto, resulting in many useless reference points during evolution. In other words, these useless reference points adversely affect the performance of NSGA-III and genetic programming. To address the above issue, a new reference point adaptation mechanism has been proposed that is based on the distribution of the candidate solutions. Also, in order to assess the performance of the proposed mechanism, experiments have been carried out on many-objective benchmark JSS instances. Our results clearly show that the proposed strategy is promising in handling useless reference points and significantly improves the performance of NSGA-III and genetic programming.

Keywords: Job shop scheduling. Many-objective optimization. Genetic programming. Reference points.

1 Introduction

Job Shop Scheduling (JSS) [13] is a classical combinatorial optimization problem that has received a lot of attention owing to its wide applicability in the real world. The JSS problem deals with the assignment of tasks or jobs to different resources or machines. The quality of schedule depends on the objective(s) of the JSS problem, e.g., the completion time or makespan.

It is widely noticeable in the literature [8] that JSS is a many-objective (i.e. more than three objectives) optimization problem. In [8] it was shown that makespan, mean flow time, maximum tardiness, mean tardiness and proportion of tardy jobs are potentially conflicting objectives.

JSS problem is widely known to be NP-hard [1] and dealing with many-objective increases its complexity. In practice, it is difficult to obtain an optimal

schedule for large instances within a reasonable amount of time by using exact optimization methods. On the other hand, heuristic methods are fast and appropriate for practical scenarios. Conceptually, dispatching rules can be perceived as priority functions, used to assign priority to each and every waiting job.

Dispatching rules in JSS show promising results[11]; however, designing dispatching rules is a complex and challenging research problem. Furthermore, dispatching rules often have inconsistent behaviors from one scenario and another. Genetic Programming Hyper-Heuristic(GP-HH) is a widely adopted technique in the literature that designs dispatching rules automatically under many conflicting objectives.

Although JSS has been proven to be many-objective optimization problem, only a few studies have focused on this issue. A recent work [8] has proposed a new evolutionary many-objective optimization algorithm for JSS problems called GP-NSGA-III that seamlessly combines GP-HH with NSGA-III [5]. The proposed hybridized algorithm was compared to other approaches, GP-NSGA-II and GP-SPEA2 that combines GP-HH with multi-objective optimization algorithms including NSGA-II [2] and SPEA2 [16]. GP-NSGA-III significantly shows better performance on 4-objective as well as 5-objective JSS problems compared to the other two approaches.

The adoption of uniformly distributed reference points in the GP-NSGA-III remains a challenge to the irregular and non-uniform Pareto front evolved in JSS problem because many of these points are never associated with any dispatching rules on the evolved Pareto front and resulting in many useless-reference points during evaluation. Evidently, useless points poorly affect the performance of algorithms. This issue is also evidenced in the literature [5], [9] while applying NSGA-III to many-objective optimization problems, where Pareto front are not uniform.

Another obvious drawback is demote solution diversity, when algorithm has high number of useless reference points. In other words, GP-NSGA-III combines offspring and parents to form one set of solution of size 2N and has fixed set of N uniformly distributed reference points . After non-dominated sorting; reference points use for the selection of next generation population. In the scenario of irregular Pareto-front, each useful reference point is associated with a cluster of solutions those are existing in the closest proximity of that corresponding reference points. Thus, during Niching[8] selection of that useful points with heap of solutions may not help in exploration and directly reduce the solution diversity for future generation. Due to the above reason, we need to find the better association between reference points and the evolved Pareto-front can help enhance solution diversity as well as reduce the useless reference points .

In order to consider the useless reference point in GP-NSGA-III, we proposed the new reference point adaption mechanism. In this strategy, whole objective space is decomposed into a small sub-simplex and generated reference points according to the density of the solutions in that location. In other words, distribution of reference point follows the distribution of Pareto-front that establish better link between reference points and solutions in GP-NSGA-III.

The goal of this study to reduce the number of useless points by adaptively matching reference points with the evolved Pareto-front so as to promote the solution diversity and performance of GP-NSGA-III.

In the remainder of this paper, we first elaborate the research background including the JSS problem description and related works in section 2. In Section 3, we propose an adaptive NSGA-III algorithm in detail. Section 4 covers the experimental design and parameter setting. Experimental results comparing the performance of the proposed algorithm with another three existing many-objectives JSS algorithms on test instances are presented in Section 5. Finally, Section 6 concludes this paper and highlights possible future research.

2 Background

In this section, the problem description will be presented first and then we will discuss some related works.

2.1 Problem Description of JSS

In a JSS problem, the shop includes a set of N jobs and M machines that need to be scheduled. In particular, each job $j_i,\ 1 \leq i \leq N$ has a sequence of m operations and each operation u_i^k has its own pre-determined route through a sequence of machines $m_i^k, 1 \leq i \leq M$ with the fixed processing time $p_i^k > 0$. Any solution to such a JSS problem has to comply with four important constraints, as described below.

1. Operations are non-preemptive. This means that once an operation starts on a specific machine then the machine cannot be interrupted by any other operations.
2. The due date D_i for each job j_i must be set in advance and fixed during the execution of j_i.
3. Each M machine is persistently available and process one operation from a queue at any given time.
4. Operations must follow precedence constraints, which means succeeding operations cannot be executed until all its previous operations have been completed.

In this study our goal is to evolve a set of non-dominating dispatching rules on the Pareto front. The quality of the schedule that was created by any specific rules will then be evaluated with respect to a number of objectives. In particular, following [8],[12], we focus on minimizing four objectives, i.e. *mean flowtime (MF), maximal weighted tardiness (MaxWT), maximal flowtime (MaxF) and mean weighted tardiness (MWT)* in a many-objective JSS problem.

2.2 Related Work

Over the years, evolutionary multi-objective algorithms *(EMO)* are widely used for handling multiple-objective problems in a job shop [1]. In general,

there are two alternative approaches used for solving multi-objective JSS, i.e. the aggregation method and the Pareto dominance method [7]. Among these approaches, the Pareto dominance concept has been successfully used for developing multi-objective JSS algorithms [12],[8].

Despite of prominent success, little effort[8],[12] has been made in developing effective EC algorithm for many-objective JSS. As far as we know, GP-NSGA-III is one of the first algorithm designed for many-objective JSS and tried to address the issue of scalability and the so-called curse of dimensionality in many-objective JSS. GP-NSGA-III uses uniformly distributed reference points but does not work well with discrete optimization problems, such as the JSS problem [8]. This issue has opened a new research direction and has attracted substantial research attention.

Deb and Jain [6],[4] recently proposed an interesting mechanisms for relocating reference points adaptively in *NSGAIII*. Their strategy is able to adaptively add and delete reference points which depend on the crowding region of current non-dominated front.

In *PSO* based relocation approach;*GP-A-NSGA-III*[9] relocates reference points toward the multiple global best particles, which is the one with the most individuals associated to it.

These two algorithms; *A-NSGA-III and GP-A-NSGAIII* mainly focus on reducing the useless reference points and improve the association between reference points and Pareto-optimal solutions. However,*A-NSGA-III* dynamically change the original size of reference points whereas GP-A-NSGAIII does not change the total number of reference points during evolution. These two algorithms works very well on a number of optimization problems, but their reference point adjustment strategies still have limitation that will address in next paragraph.

One of the limitation existing in ANSGA-III that is also highlighted in [4]; A-NSGAIII does not allowed any extra reference points around the corner points of hyperplane. if many solutions are associated with these boundary points then ultimately many useless points still exist in the algorithm and affect the performance of the algorithm. Also few high quality dispatching rules that is associated with these points may not be selected and ignore in future generation because it can be far from the corresponding corner points. Due to the above reason the algorithm will not improve the solution diversity in the algorithm.

In the case of *GP-A-NSGAIII*, particles (useless reference points) in a swarm only attracted towards the global best particles. Particles is updated its position by using an updated velocity of each generation. Consider a boundary useful points that associated with majority of solutions called global best of that generation. If particles attracted toward the global best and updated its new positions and particle has fast speed,it may find their new position from outside of the simplex. In this case mapping the particles from the non-simplex surface to simplex is so important. Otherwise, this particles can map far from the denser region and can't associate any solutions. In this if scenario, each generation has useless reference points than it ultimately affect on the performance of the algorithm.

So, existing algorithms still have an issue of useless reference points that can affect on the performance of the algorithms, especially in the case of many-objective optimization. In contrast, our proposed approach is a density based approach that generated the reference points according to the distribution of the solutions on a simplex during evolution. Because of the reference points follow the distribution of the solutions that allow improve association and reduce the existence of the useless reference points. Proposed algorithm also minimize an issue of corner points because we generated carefully these reference points and be sure these points must inside in a lower and upper boundary of each sub-simplex. Moreover, This approach does not change the total number of reference points during evolution. It is easy to implement regardless of the number of objectives under consideration. So, this approach is good for many-objective optimization problem such as JSS.

3 Adaptive Reference Points for Many-Objective JSS

The basic framework of the proposed algorithm, GP-NSGA-III with Density Model based Reference Point Adaptation; *GP-NSGA-III-DRA* is described in **Algorithm1**. The basic frame-work of the proposed algorithm is similar to *GP-NSGA-III*. Here, we introduce new adaption mechanism of reference points that can generate in simplex according to the solution distribution.

GP-NSGA-III-DRA starts with initial population P_g and predefined simplex locations $w_i \epsilon W$, which partition the whole objective space in a number of independent sub-simplex developed by *Das and Dennis* systematic approach[3]. For devloping the density model we should know the density of solution of each w_i. To know the density, we calculated the perpendicular distance of each location w_i from each solution $\overline{s_i}$ in the normalized objective space. The solution $\overline{s_i}$ is closest to a location w_i is consider to be a associated with w_i. Then, we need to estimate the reference point on each location, for this any of a probability of selection mechanism can be used but here we particularly used the *roulette wheel selection* that can determine the number of required reference points in each sub-simplex; it is described in **Algorithm4**.After knowing the demand of reference point in each sub-simplex; we generated the reference points on that specific location. Detail of reference points generation method is mentioned in **Algorithm3**. In the following sub sections, each component of proposed algorithm is described in detail.

3.1 Fitness Evaluation

Lines 1 and 6 of **Algorithm1** evaluate each *GP* individual by applying to a set of JSS training instances I_{train} as a dispatching rule. Then, the normalized objective values of the resultant schedules are set to its fitness values. The pseudo code of the fitness evaluation is given in **Algorithm2**.

Algorithm 1: The framework of GPD.

Input : A training set $\boldsymbol{I_{train}}$, rules P, H structured reference points Z_g
Output: A set of non-dominated rules $\boldsymbol{P}^*$

Initialize and evaluate the population $\boldsymbol{P}_0$ of rules by the ramped-half-and-half method;
Generate the $\boldsymbol{W}$ that partition the Objective Space into K locations ;
Set $g \leftarrow 0$;
while $g < g_{\max}$ **do**
    Generate the offspring population $\boldsymbol{Q}_g$ using the crossover, mutation and reproduction of GP;
    foreach $Q \in Q_g$ **do** Evaluate rule Q;
    $\boldsymbol{R}_g \leftarrow \boldsymbol{P}_g \cup \boldsymbol{Q}_g$;
    $(F_1, F_2 \dots) =$ Non-dominated-sort(R_g);
    while $|S_g| \geq N$ **do**
        $S_g = S_g \cup F_i; i = i + 1$;
    end
    if $|S_g| = N$ **then**
    end
    $P_(g + 1) = S_g$;
    if $|S_g| > N$ **then**
        $P_(g + 1) = \cup_{j=1}^{l-1} F_j$;
    end
    $K = |F_l| : K = N - |P_{t+1}|$;
    Normalize Objectives of members in S_g;
    $\overline{\boldsymbol{S_g}} = \boldsymbol{Objective_Normalization(S_g)}$;
    $\boldsymbol{Z_g^* = Generate(Z_g, W)}$;
    foreach icZ_g **do** identify member of $\overline{S_g}$ close to i ;
    $\boldsymbol{E(i) = Associate(S_g, Z_i^*}$;
    $\boldsymbol{P_(g + 1)}$= $\boldsymbol{Niching(K, Z_g^*), \overline{S_g})}$;
    $g \leftarrow g + 1$;
end
return *The non-dominated individuals* $P^* \subseteq P_{g_{\max}}$;

3.2 Reference Point Generation

The procedure is detail mentioned in **Algorithm4**. Reference point generation scheme $Generate(Z_g, W)$ is described in **Algorithm3**. In the algorithm, density of the solution in each sub simplex location $w_i \epsilon W$ is identified by minimum perpendicular distance. This density of the solution is help to find out the number of the required reference points in the specific location by using roulette wheel. For generated the reference points we iteratively compute the average of the candidate solutions in the sub simplex location w_i and then calculated the

Algorithm 2: The fitness evaluation.

Input : A training set $\boldsymbol{I}_{train}$ and an individual (rule) P
Output: The fitness $\boldsymbol{f}(P)$ of the rule P

foreach I *in* $\boldsymbol{I}_{train}$ **do**
    Construct a schedule $\Delta(P, I)$ by applying the rule P to the JSS instance I;
    Calculate the objective values $\boldsymbol{f}(\Delta(P, I))$;
end
$\boldsymbol{f}(P) \leftarrow \frac{1}{|\boldsymbol{I}_{train}|} \sum_{I \in \boldsymbol{I}_{train}} \boldsymbol{f}(\Delta(P, I))$; ;
return $\boldsymbol{f}(P)$;

Algorithm 3: Generate Reference points

Input : $W, \overline{S_g}, E(w)$
Output: $\boldsymbol{Z}_g^*$

foreach $s \epsilon \overline{S}_g$ **do**
    foreach $w \epsilon W_g^*$ **do**
        compute $d^{\perp}(s, w)$;
    end
    if $argmin_{w \epsilon W} d^{\perp}(s, w)$ **then**
        save $s \epsilon \overline{S}_g in E(w)$;
        $Density(w)$=$Density(w)$+1;
    end
end
sampling() ;
return $\boldsymbol{Z}_g = \boldsymbol{Z}_{\boldsymbol{g}}^*$;

perpendicular distance between the center and existing solution. Solutions which is far from the centroid is first selected for generating the reference points.Then selected solutions is excluded for calculated next centroid from existing solutions. It should be mentioned here, if one and only one reference point is required in sub simplex then it should be centroid. Moreover, area of each sub-simplex is very small and randomly distributed solutions are very close to each other. Therefore, generated the reference points from center will cover the proximity of each solutions and help to improve a better association . Later we carefully generated the reference point by using mid point of center and corresponding solution. Here it should be mentioned that we can generate the reference points very close or same position of solutions. However requirement of reference points after roulette wheel in each sub-simplex is lesser than the quantity reside in that sub simplex. There If we generated the reference points by using above mentioned method then many of solutions could not enjoy the selection opportunity because it far from the popular reference points and never be associated.

Algorithm 4: Sampling of X

Input : $Density(w), W, \overline{S_g}$
Output: $\boldsymbol{Z_g^*}$

Set $quantity = 0$;
foreach $w \epsilon W$ **do**
  Set $D(w) = Density(w) \div \sum Density$;
end
while $j < |W|$ **do**
  Set $random \rightarrow (0, \sum D)$;
  foreach $d \epsilon D$ **do**
    $rand = rand - D(d)$;
    if $rand <= 0$ **then**
      $R \rightarrow RUD(d)$;
      $quantity = quantity + 1$;
      $j = j + 1$;
    end
  end
end
foreach $i \epsilon W$ **do**
  if $D(i) \geq 0$ **then**
    foreach $E(i)$ **do**
      for $k = 1 \rightarrow \boldsymbol{quantity}$ **do**
        foreach j *= 1 to* M **do**
          $computeMean(E(i))$
        end
      end
      foreach $s \epsilon \overline{E}(i)$ **do**
        compute $d^{\perp}(s, Mean(E(i)))$;
      end
      if $argmax_{d^{\perp}(s, Mean(E(i)))}$ **then**
        $\boldsymbol{Z_g^*} = Mean(w) + mid((s, Mean(E(i)))$;
      end
    end
  end
end
return $\boldsymbol{Z_g^*}$;

4 Experimental Design

In this section, we explain our design of experiments for the JSS problem and provide the configuration of the GP system, performance measure and the data set in detail.

4.1 Parameter Settings

Dispatching rules in our experiment adopt the tree based GP representation that is also a conman representation in literature[8]. These trees are constructed from the list of functions and terminals, as summarized in Tables 1 and 2. For all compared algorithms, the initial GP population is created by the ramp-half-and-half method. The population size is set to 1024 to ensure that there is an enough diversity in the evolved solutions. The crossover, mutation and reproduction rates are set to 85%, 10% and 5% respectively. The maximal depth is set to 8 for extending the search space of GP in the case of high dimensional JSS problems. In each generation, individuals are selected using tournament selection with the size 7. The maximal number of generations is set to 51 for all 30 runs.

In our experiment, we compared our proposed algorithm with GP-ANSGA-III, GP-NSGA-III and GP-A-NSGA-III(PSO based). Particularly, GP-NSGA-III is a recently developed algorithm specially designed for many-objective JSS problems but without adaptive reference points. A-NSGA-III is a well-known general purpose algorithm with adaptive reference points and GP-A-NSGA-III is also adaptively relocate reference points by using the dynamics of PSO.

Table 1: Functional Sets for GP for JSS.

Function Set	Meaning
$+$	Addition
$-$	Subtraction
$*$	Multiplication
$/$	Protected Division Operator
Max	Maximum
Min	Minimum
$if\text{-}then\text{-}else$	Conditional

Table 2: Terminal set of GP for JSS.

Attribute	Notation
Processing time of the operation	PT
Inverse processing time of the operation	IPT
Processing time of the next operation	NOPT
Ready time of the operation	ORT
Ready time of the next machine	NMRT
Work Remaining	WKR
Number of operation remaining	NOR
Work in the next queue	WINQ
Number of operations in the next queue	NOINQ
Flow due date	FDD
Due Date	DD
Weight	W
Number of operations in the queue	NOIQ
Work in the queue	WIQ
Ready time of the machine	MRT

4.2 Data Set

To verify the effectiveness of our algorithm, we first apply on static JSS that is simpler than dynamic JSS[10]. Therefore, in this study we try to find an optimal solution in a static JSS problem. Our experiment requires static data set for measuring the performance of evolved heuristics. Hence, we selected Taillard (TA) static JSS benchmark instances [14] as a test bed. In a nutshell, it consists of 80 instances (ID from 1 to 80), divided into 8 subsets that cover JSS problems with a varied number of jobs (*15 to 100*) and number of machines (*5 to 20*).

In the experiments, each subset was further divided into training and test sets, each consisting of 40 instances. Since instances were static, release time of all the jobs was safely set to zero. The due date $dd(j_i)$ of each job j_i was calculated by *total workload strategy*. That is,

$$dd(j_i) = \lambda \times \sum_{k=1}^{m} p_i^k, \tag{1}$$

where λ is the due date factor, which is set to 1.3.

4.3 Performance Measures

In order to assess the performance of our proposed algorithm and compared with all algorithms in our experiments, we adopt two common measures, i.e. *Inverted Generational Distance* (IGD) [15] and *Hyper-Volume* (HV) [17]. These indicators describe how well the approximation of the true Pareto front is converged and diversified. Theoretically, a set of non-dominated solutions with better performance should require a smaller IGD value and larger HV value.

5 Results and Discussions

This section compares the obtained Pareto fronts from GP-NSGA-III ,GP-A-NSGA-III(PSO based),GP-NSGA-III-DRA, GP-ANSGA-III on both training and test instances by using IGD [17] and HV [18]. Parallel coordinate plot shows the distribution of population and reference points.

5.1 Overall Result

During the GP search process, a rule is evaluated on the 40 training instances. Moreover, the fitness function for each objective is found as the average normalized objective value [10] each of the 40 training instances of the obtained schedule by applying that rule to each of the 40 training instances. For each algorithm, 30 independent GP runs obtained 30 final dispatching individuals. Then, the rules were tested on the 40 test instances.

Table 3 and 4 show the mean and standard deviation of the average HV and average IGD values of 30 Pareto fronts on the training instances. In addition, the Wilcoxon rank sum test with the significance level of 0.05 was conducted separately on both the HV and IGD of the rules obtained by the four compared

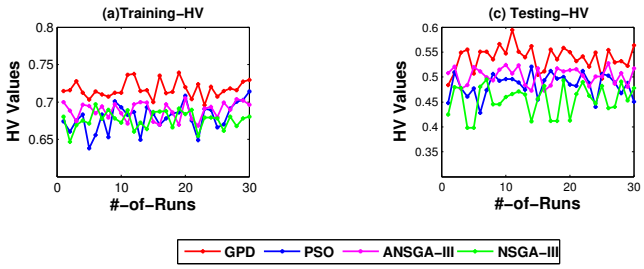
algorithms. That is, if p-value is smaller than 0.05, then the best algorithm is considered significantly better than the other algorithms. The significantly better results was marked in bold.

Table 3 and 4 reveal that GP-NSGA-III-DRA achieved significantly better performance in term of HV as compared to the other algorithms. Also in term of IGD; GP-NSGA-III-DRA clearly outperformed with GP-ANSGAIII, GP-A-NSGA-III(PSO based) and GP-NSGAII. If we take a closer look at the **Figure** :1a and **Figure** :1b, it can be found that GP-NSGA-III-DRA has lower IGD than GP-NSGA-III and higher HV in many independent runs. In regards to the test instances, **Table** 5 and 6 exhibits the same pattern in term of IGD. However, in term of HV proposed algorithm is better but not competitive with GP-A-NSGA-III and significantly better than other two compared algorithms.

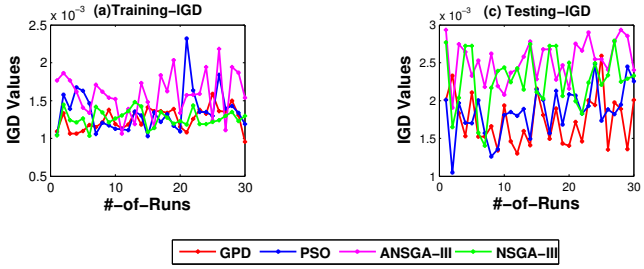
To have a better understanding of these algorithms' performance on the GP search process, we plot (a) the HV of the non-dominated solutions obtained so far and (b) the IGD of the non-dominated solutions obtained so far for each generation during the 30 independent runs of the four compared algorithms, as given in **Figure** : 2. It shows that at early generations of evaluation, when solutions are not matured then our proposed algorithm has same HV and IGD values. Then HV and IGD value of GP-NSGA-III-DRA is gradually improve as compared with GP-A-NSGA-III but significantly improve form GP-PSO and GP-NSGA-III. But when it toward the Pareto front proposed algorithm has significantly better HV and IGD as compared to other algorithms. This graph suggest us that generation by generation optimal solutions are improved in the case of GP-NSGA-III-DRA. This can also see in the parallel coordinate plot that shows the distribution of the reference points and the fitness values of the population in generations on 10, 30 and 50 of GP-A-NSGA-III(PSO based), GP-ANSGA-III and GP-NSGA-III-DRA. GP-NSGA-III works on uniform distributed reference point, therefore we only show the distribution of the fitness value of GP-NSGA-III. **Figure** :4 also shows that GP-NSGA-III-DRA has more diversified solutions as compared to other three algorithms that shows in **Figure** : 3,**Figure** : 5,**Figure** : 6 because the distributions of the reference points and the fitness values of the population become very similar to each others in **Figure** :4 .

Table 3: The mean and standard deviation over the average HV values of the 30 independent runs on Training instances of the compared algorithms in the 4-obj experiment. The significantly better results are shown in bold.

HV				
Statistic	GP-NSGA-III-DRA	GP-A-NSGA-III(PSO)	GP-ANSGA-III	GP-NSGA-III
Mean&(STD)	**0.71699(0.01103)**	0.67953(0.01847)‡	0.68867‡(0.01140)	0.676387(0.01165)‡



(a) Average HV values of the 30 independent runs on Training and Testing instances



(b) Average IGD values of the 30 independent runs on Training and Testing instances

Fig. 1: Average HV and IGD values of Pareto Front

Table 4: The mean and standard deviation over the average IGD values of the 30 independent runs on Training instances of the compared algorithms in the 4-obj experiment. The significantly better results are shown in bold.

IGD				
Statistic	GP-NSGA-III-DRA	GP-A-NSGA-III(PSO)	GP-ANSGA-III	GP-NSGA-III
Mean&(STD)	**0.001257(0.00014)**	0.00135(0.00027)‡	0.00155(0.00028))‡	0.00133(0.00011)‡

Table 5: The mean and standard deviation over the average HV values of the 30 independent runs on Testing instances of the compared algorithms in the 4-obj experiment. The significantly better results are shown in bold.

HV				
Statistic	GP-NSGA-III-DRA	GP-A-NSGA-III(PSO)	GP-ANSGA-III	GP-NSGA-III
Mean&(STD)	**0.538581(0.02380)**	0.484011(0.02299)‡	0.502493(0.01693)†	0.45446(0.02964)‡

6 Conclusion

In this paper, we have addressed a key research issue involved in using uniformly distributed reference points on the objective space for irregular Pareto front such as in many-objective JSS problem. An adaptive reference point strat-

Table 6: The mean and standard deviation over the average IGD values of the 30 independent runs on Testing instances of the compared algorithms in the 4-obj experiment. The significantly better results are shown in bold.

IGD				
Statistic	GP-NSGA-III-DRA	GP-A-NSGA-III(PSO)	GP-ANSGA-III	GP-NSGA-III
Mean&(STD)	**0.001734(0.00032)**	0.00184(0.00031)‡	0.002521(0.00026)‡	0.002277(0.00036)‡

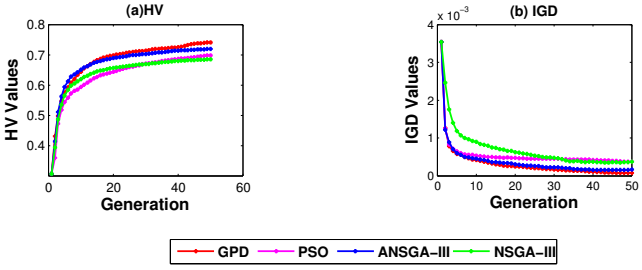
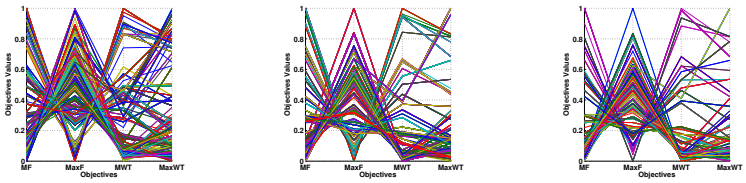


Fig. 2: HV and IGD values of the non-dominated solutions on the training set during the 30 independent GP runs.



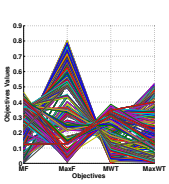
(a) Fitness values of the population in generation 10

(b) Fitness values of the population in generation 30

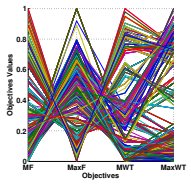
(c) Fitness values of the population in generation 50

Fig. 3: Parallel coordinate plot for the fitness values of the population in generations 10,30 and 50 of NSGA-III.

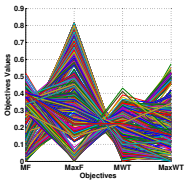
egy is proposed to enhance the performance of GP-NSGA-III for many-objective. We successfully developed this proposed algorithm for generating the reference points according to the density of the solutions that provided more useful reference points. Furthermore, better association between reference points and Pareto Optimal solutions will help to improve the usage of computational resources[8] and promote the solution diversity. To examine the effectiveness and performance of our algorithm, experimental studies have been carried out by using the Taillard static job-shop benchmark set. Experimental study suggest that *GP-NSGA-III-DRA* can perform significantly better than other compared algorithms in both training and testing instances and discover the Pareto-optimal solutions are more diversified than others compared algorithms. This finding lead us to believe that our algorithm *GP-NSGA-III-DRA* is more effective and



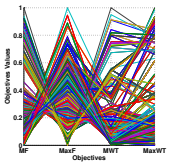
(a) Distribution of adaptive reference points in generation 10



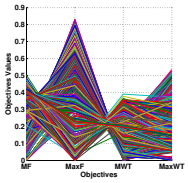
(b) Fitness values of the population in generation 10



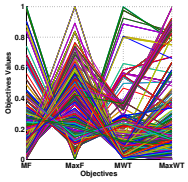
(c) Distribution of adaptive reference points in generation 30



(d) Fitness values of the population in generation 30



(e) Distribution of adaptive reference points in generation 50



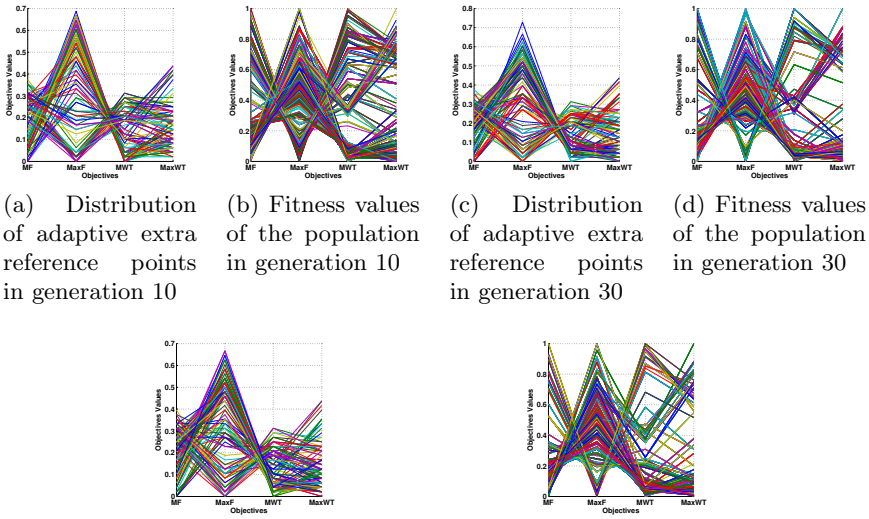
(f) Fitness values of the population in generation 50

Fig. 4: Parallel coordinate plot for the distribution of the reference points and the fitness values of the population in generations 10,30 and 50 of GPD.

efficient for many-objective JSS as compared to the existing algorithms.
Rooms are still available for further improvement. For example, to achieve even a better match in between reference points and sampled solutions, we can utilize a Gaussian Process model to gradually and accurately learn the distribution of the Pareto front.

References

1. Błażewicz, J., Domschke, W., Pesch, E.: The job shop scheduling problem: Conventional and new solution techniques. European journal of operational research 93(1), 1–33 (1996)
2. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6, 182–197 (2002)
3. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. Evolutionary Computation, IEEE Transactions on 18(4), 577–601 (2014)
4. Jain, H., Deb, K.: An Improved Adaptive Approach for Elitist Nondominated Sorting Genetic Algorithm for Many-Objective Optimization. In: Purshouse, R.C.,



(a) Distribution of adaptive extra reference points in generation 10

(b) Fitness values of the population in generation 10

(c) Distribution of adaptive extra reference points in generation 30

(d) Fitness values of the population in generation 30

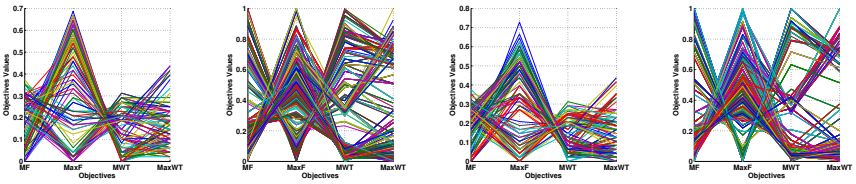
(e) Distribution of adaptive extra reference points in generation 50

(f) Fitness values of the population in generation 50

Fig. 5: Parallel coordinate plot for the distribution of the reference points and the fitness values of the population in generations 10,30 and 50 of GP-A-NSGA-III(PSO).

Fleming, P.J., Fonseca, C.M., Greco, S., Shaw, J. (eds.) EMO. Lecture Notes in Computer Science, vol. 7811, pp. 307–321. Springer (2013)

5. Jain, H., Deb, K.: An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point Based Nondominated Sorting Approach, Part II: Handling Constraints and Extending to an Adaptive Approach. IEEE Trans. Evolutionary Computation 18(4), 602–622 (2014)
6. Jain, H., Deb, K.: An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point Based Nondominated Sorting Approach, Part II: Handling Constraints and Extending to an Adaptive Approach. IEEE Trans. Evolutionary Computation 18(4), 602–622 (2014)
7. Jayamohan, M., Rajendran, C.: New dispatching rules for shop scheduling: a step forward. International Journal of Production Research 38(3), 563–586 (2000)
8. Masood, A., Mei, Y., Chen, G., Zhang, M.: Many-Objective Genetic Programming for Job-Shop Scheduling. IEEE WCCI 2016 conference proceedings, IEEE (2016)
9. Masood, A., Mei, Y., Chen, G., Zhang, M.: A PSO-Based Reference Point Adaption Method for Genetic Programming Hyper-Heuristic in Many-Objective Job Shop Scheduling. In: Wagner, M., Li, X., Hendtlass, T. (eds.) ACALCI. Lecture Notes in Computer Science, vol. 10142, pp. 326–338 (2017)
10. Mei, Y., Zhang, M., Nyugen, S.: Feature Selection in Evolving Job Shop Dispatching Rules with Genetic Programming, in GECCO, ACM, 2016 (2016)



(a) Distribution of adaptive extra reference points in generation 10

(b) Fitness values of the population in generation 10

(c) Distribution of adaptive extra reference points in generation 30

(d) Fitness values of the population in generation 30

(e) Distribution of adaptive extra reference points in generation 50

(f) Fitness values of the population in generation 50

Fig. 6: Parallel coordinate plot for the distribution of the reference points and the fitness values of the population in generations 10,30 and 50 of GP-ANSGA-III.

11. Nguyen, S., Zhang, M., Johnston, M.: A genetic programming based hyper-heuristic approach for combinatorial optimisation. In: Krasnogor, N., Lanzi, P.L. (eds.) GECCO. pp. 1299–1306. ACM (2011)
12. Nguyen, S., Zhang, M., Johnston, M., Tan, K.C.: Dynamic multi-objective job shop scheduling: A genetic programming approach. In: Automated Scheduling and Planning, pp. 251–282. Springer (2013)
13. Pinedo, M.L.: Scheduling: theory, algorithms, and systems. Springer Science & Business Media (2012)
14. Taillard, E.: Benchmarks for basic scheduling problems. european journal of operational research 64(2), 278–285 (1993)
15. Zhang, Q., Zhou, A., Zhao, S., Suganthan, P.N., Liu, W., Tiwari, S.: Multiobjective optimization test instances for the CEC 2009 special session and competition. University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report pp. 1–30 (2008)
16. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the strength pareto evolutionary algorithm. In: EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems. pp. 95–100 (2002)
17. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Da Fonseca, V.G.: Performance assessment of multiobjective optimizers: an analysis and review. Evolutionary Computation, IEEE Transactions on 7(2), 117–132 (2003)