



Ensemble of multi-objective metaheuristics for multiprocessor scheduling in heterogeneous distributed systems: a novel success-proportionate learning-based system

Nasser Lotfi¹

© Springer Nature Switzerland AG 2019

Abstract

Multi-objective multiprocessor scheduling problem in heterogeneous distributed systems is a remarkable NP-hard problem in parallel processing environments. This study proposes a novel ensemble system comprising six population-based metaheuristics for solving multi-objective multiprocessor scheduling problem. The novelty of this study is mainly related to proposed cooperation strategy. In proposed strategy, solutions and metaheuristics are selected based on dominance-rank and success-rate values respectively. All metaheuristics work on a common population of solutions and during the running time performance of metaheuristics is evaluated and used as success rates of the metaheuristics. Choosing the next metaheuristic based on success rate values leads to the fact that better metaheuristics are used frequently while the poor ones are employed rarely. The advantage of using different metaheuristics is that each metaheuristic covers inabilities of others to discover more promising regions of solution space. Likewise to keep extracted non-dominated solutions, local archives and a common global archive are used. The performance of proposed system is evaluated using well-known benchmarks reported in state-of-the-art literature. The evaluations are done over Gaussian elimination, fast Fourier transformation and internal rate of return (IRR) task graphs. Evaluation outcomes prove the robustness of the proposed system and exhibits that ensemble system outperforms its competitors. As instance, the proposed system finds 52, 19 and 43 as the values of three objectives namely makespan, average flow time and reliability for graph presented in Fig. 2 which are better than results obtained by MFA and NSGAI. Also the results show that ensemble system works better than all individual metaheuristics in which it reaches 475, 523 and 381 as values of three objectives for IRR graph.

Keywords Multi-objective optimization · Multi heuristic system · Dominance rank · Multiprocessor scheduling problem · Task graph

1 Introduction

Usually the real-world problems have a number of objectives contradicting each other and hence, there has been an upward attention to apply multi-objective optimization methods for solving such problems during the last three decades [1]. Multi-objective optimization is of significant challenge in majority of engineering applications and they have been extensively solved by multi-objective type metaheuristics due to their capability in extracting good

solutions in an acceptable computational time. Eventually, extracting near optimal solutions satisfying the objectives is the aim of the multi-objective optimization task [2]. The set of all non-dominated solutions extracted by the algorithm is known as optimal Pareto-front. A solution is considered as non-dominated solution if it is better than other solutions in at least one objective and it is not worse in terms of other objectives [3, 4]. Figure 1 represents the optimal Pareto-front for a minimization problem consisting of seven non-dominated solutions for two objective

✉ Nasser Lotfi, nasserlotfi@gau.edu.tr | ¹Faculty of Engineering, Girne American University, Girne, N. Cyprus Via Mersin 10, Turkey.



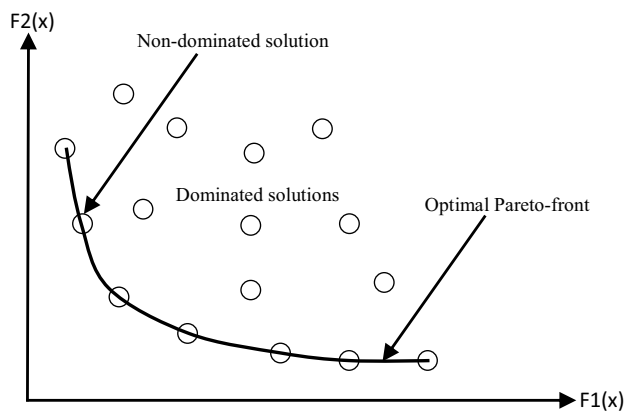


Fig. 1 Optimal Pareto-front and dominated solutions for a minimization problem

functions $F1$ and $F2$. All other nine solutions are called as dominated solutions.

Multi-objective multiprocessor scheduling problem in heterogeneous systems is one of the remarkable and important problems in parallel and distributed environments. Due to being Np-hard, researchers have been extensively solving this challenging problem using evolutionary methods. In multi-objective multiprocessor scheduling problem, parallel program is represented as a directed acyclic graph (DAG) which is called as task graph. The nodes represent the tasks which are the program units with certain running times. Also, the directed edges illustrate the execution order of tasks and communication costs between processors. Aim of problem is to distribute tasks over processors in order to improve objective functions. In single objective type of problem, the goal is to minimize the total completion time of task graph called Makespan. Also in multi-objective type problem, minimizing the makespan and average flow-time together with maximizing the reliability is the aim of problem. Detailed description of the problem and state-of-the-art methods are given in next sections.

Pretty wide state-of-the-art methods can be found in literature for solving multi-objective multiprocessor scheduling problem [5–15]. In related works, authors mostly used Multi-objective Genetic Algorithm (MOGA) [10, 11], Multi-objective Evolutionary Programming (MOEP) [14], Modified Genetic Algorithm (MGA) [7], Bi-objective Genetic Algorithm (BGA) [8], Mean Field Annealing (MFA) [5], Weighted-sum Genetic Algorithm (WGA) [9], Heterogeneous Earliest Finish Time Algorithm (HEFT) [15], Critical Path Genetic Algorithm (CPGA) [15] and some hybrid methods [16]. In most of the related works, the proposed methods are based on simple evolutionary algorithms and weighted-sum methods. Detailed description of state-of-the-art methods is given in Sect. 3.

This study proposes a novel method to ensemble the metaheuristics for solving multi-objective multiprocessor scheduling problem. The suggested system consists of six population-based multi-objective metaheuristics which cooperate together based on a proposed strategy. The metaheuristics applied in the proposed ensemble system are listed as: Non-dominated Sorting Genetic Algorithm (NSGAI) [17], Multi-objective Genetic Algorithm (MOGA) [18], Multi-objective Differential Evolution (MODE) [19], Intelligent Multi-objective Particle Swarm Optimization (IMOPSO) [20], Strength Pareto Evolutionary Algorithm (SPEA2) [21] and Multi-objective Artificial Bee Colony algorithm (MOABC) [22].

The novelty of this study is mainly related to proposed cooperation strategy. In proposed strategy, solutions and metaheuristics are selected based on dominance-rank and success-rate values respectively. All metaheuristics work on a common population of solutions and during the running time performance of metaheuristics is evaluated and used as success rates of the metaheuristics. Choosing the next metaheuristic based on success rate values leads to the fact that better metaheuristics are used frequently while the poor ones are employed rarely.

Success rate values are between 0 and 100 which are initialized by 50 in the beginning. These values are being decreased and increased based on the metaheuristics performance during the execution. After each metaheuristic selection, system extracts the search space by exploration and exploitation techniques. In exploration phase, system chooses a subpopulation of solutions through the general population and runs the selected metaheuristic over chosen subpopulation. Afterwards, an exploitation phase is done over the extracted Pareto-front by local search method. To choose a subpopulation, tournament selection approach is used based on dominance rank values. Dominance rank of a solution s is the number of other solutions dominating s . This way, better solutions in terms of dominance will have more chance to be selected. Thereafter, according to the results of metaheuristic, the success rate is updated and hence, the ensemble system will learn gradually how good the metaheuristics are. Briefly, the proposed system prefers many short runs of different metaheuristics instead of one long run of a single metaheuristic.

Likewise, the advantage of using different metaheuristics is that capabilities of each metaheuristic covers inabilities of other metaheuristics. This way, it might be possible to discover more promising regions of the solution space. In order to keep extracted non-dominated solutions in proposed system, each metaheuristic has its own local archive while a common global archive is used by whole metaheuristics to keep all non-dominated

solutions found so far. Description of suggested ensemble system is expressed in Sect. 4.

The performance of proposed system is evaluated using well-known benchmarks reported in state-of-the-art literature. The evaluations are done over Gaussian Elimination (GE), Fast Fourier Transformation (FFT) and Internal Rate of Return (IRR) task graphs. Evaluation outcomes prove the robustness of the proposed system and exhibits that ensemble system outperforms its competitors. As instance, the proposed system finds 52, 19 and 43 as the values of three objectives namely Makespan, Average Flow Time and Reliability for graph presented in Fig. 2 which are better than results obtained by MFA and NSGAI. Also the results show that ensemble system works better than all individual metaheuristics in which it reaches 475, 523 and 381 as values of three objectives for IRR graph.

The rest of the paper is formed as following: The detailed description of Multi-objective multiprocessor scheduling problem is given in Sect. 2. Section 3 presents the state-of-the-art methods for solving the Multi-objective multiprocessor scheduling problem. The proposed novel ensemble system is expressed in Sect. 4. Section 5 demonstrates the algorithm parameters, experimental and comparison results. Finally, the conclusions and some future research works are given in Sect. 6.

2 Multi-objective multiprocessor scheduling in heterogeneous distributed systems

This section describes the multi-objective multiprocessor scheduling problem in details. In the problem, a parallel program is represented as a directed acyclic graph (DAG) which is called as task graph $G = (V, E)$, where the V and E represent the set of nodes and set of directed edges respectively [23, 24]. The nodes represent the tasks which are the program units with certain running times. Also, the directed edges illustrate the execution order of tasks

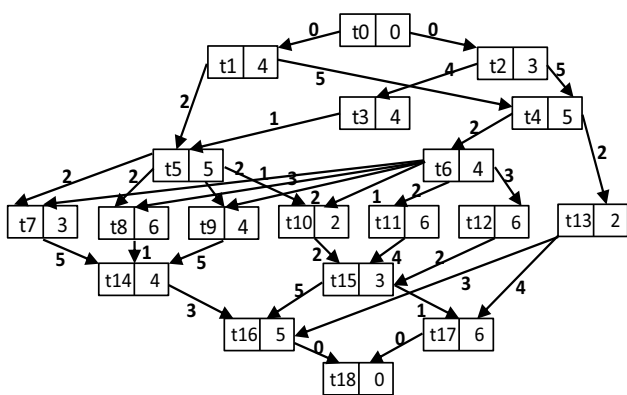


Fig. 2 Sample task graph

and communication costs between processors [24]. Likewise, the entry node and exit node of the task graph are t_0 and t_{18} respectively. It is obvious that if two tasks are performed on same processor, the communication cost becomes zero, because there is nothing to be transmitted. Figure 2 indicates a sample task graph consisting of nineteen tasks [5, 24, 25].

The heterogeneous distributed system includes a set P of heterogeneous processors which are fully connected. A schedule s is a function $S: V \rightarrow P$ which maps tasks to processors. Aim of the problem is to map tasks to processors and find an optimal schedule in terms of objective functions.

Single objective type of the problem considers the completion time of the parallel program (task graph) to be minimized [24]. The completion time of the task graph on multiprocessors is also known as Makespan.

In the multi-objective type of the problem, two more objectives are defined as minimization of Average flow-time and maximization of Reliability [6]. It should be noticed that in the literature, the minimization of reliability index is considered instead of maximization of reliability [6]. The values of these three objectives specify the quality of solutions. Formulation of the multiprocessor scheduling as multi-objective problem is given as follows [6–9, 16, 23, 26, 27]:

$$\text{Min } f = [f_1, f_2, f_3] \tag{1}$$

where f_1, f_2, f_3 are objective functions defined as below:

The first objective f_1 is the completion time (Makespan) of the schedule s which is computed as

$$f_1 = \max_j C_j(s) \tag{2}$$

where $C_j(s)$ is the completion time of processor p_j and $\max_j C_j(s)$ is the completion time of all processors in schedule s .

The completion time of processor p_j is calculated as

$$c_j(s) = \sum_{i \in v(j,s)} (st_{ij} + w_{ij}) \tag{3}$$

where $v(j, s)$ is the set of tasks assigned to processor p_j . Also, st_{ij} and w_{ij} are the start time and execution time of task v_i on processor p_j .

The second objective is the average flow-time which is formulated as

$$f_2 = \text{aft}(s) = \frac{\sum_j C_j(s)}{|P|} \tag{4}$$

where $\text{aft}(s)$ denotes the sum of the completion time of all processors in schedule s divided by the number of processors. Also $|P|$ denotes the number of processors in set P .

The processors in set P can fail but their failure is transient. A processor failure affects only the task being executed on that processor without any effects on the following tasks. The failures are statistically independent events and follow the Poisson's law. λ_j as a constant parameter is the failure rate of processor p_j . The probability of running all tasks successfully on a processor p_j is calculated as:

$$P_{succ}^i(s) = e^{-\lambda_j c_j(s)} \tag{5}$$

And finally, the probability of finishing schedule s successfully is

$$P_{succ} = e^{-\sum_j \lambda_j c_j(s)} \tag{6}$$

In case of having link between processors p_m and p_n in heterogeneous system, the link reliability is computed as

$$R_{mn}(V, S) = e^{-\lambda_{mn} \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} s_{im} s_{jn} c_{ij}} \tag{7}$$

where λ_{mn} is the failure rate of link between processor p_m and p_n . Also s_{im} and s_{jn} are the mapping of task i on processor p_m and task j on processor p_n respectively for schedule s. Likewise, c_{ij} is the communication cost between task i and j.

The reliability index of a schedule s is the third objective which is defined as below

$$f_3 = rel(s) = \sum_j \lambda_j c_j(s) + \sum_{m=1}^{|P|} \sum_{n=1}^{|P|} \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} \lambda_{mn} s_{im} s_{jn} c_{ij} \tag{8}$$

where $|V|$ denotes the number of tasks in set V.

Minimizing f_3 maximizes the success probability of schedule s.

3 State-of-the-art in methods for solving multi-objective multiprocessor scheduling problem

Evolutionary methods including different metaheuristics have been extensively used to solve single- and multi-objective multiprocessor scheduling problem [8]. This section reviews state-of-the-art in methods for solving multi-objective multiprocessor scheduling problem but it should be noticed that there is no much literature in this context.

Chitra et al. [8] compared some existent algorithms by applying them on bi-objective Gaussian elimination graph and some randomly generated graphs. The goal of the optimization was to minimize the makespan (total completion time) and maximize the reliability. They applied standard Genetic Algorithm (GA) [10, 11] and Evolutionary Programming (EP) [14] methods along with weighted sum approach over a Gaussian

Elimination graph with 18 nodes presented in Fig. 15 and some randomly generated task graphs. Meanwhile, they used Multi-objective Genetic Algorithm (MOGA) [18] and Multi-objective Evolutionary Programming (MOEP) [12] to solve the same problems. The obtained results showed that MOEP achieves better distributed Pareto-front compared to MOGA and weighted sum approach.

Eswari and Nicholas [7] proposed a firefly based algorithm as a solution of two-objective multiprocessor scheduling problem. The paper considered makespan and reliability as objective functions. They compared obtained results to modified genetic algorithm (MGA) [9] and bi-objective genetic algorithm (BGA) [13] and the evaluation results showed that firefly based algorithm can minimize makespan and maximize reliability in minimum number of generations.

Lotfi and Acan [5] applied multi-objective mean field annealing method for solving three-objective multiprocessor scheduling problem. The study evaluated the results over some task graphs and compared to NSGAll [17] and MOGA [9] algorithms. The results showed that Mean Field Annealing algorithm outperforms NSGAll and MOGA on some task graphs in terms of extracted Pareto-front quality.

Chitra et al. [6] used some existent metaheuristics for solving three-objective multiprocessor scheduling problem. They applied weighted-sum Genetic Algorithm (GA) [10], weighted-sum Evolutionary Programming (EP) [14], Multi-objective Genetic Algorithm (MOGA) [18] and Multi-objective Evolutionary Programming (MOEP) [12] over Gaussian Elimination Graph. The considered objectives are Makespan, Average Flow-time and Reliability.

Chen et al. [15] proposed HEFT-NSGA algorithm to solve the multiprocessor scheduling problem aiming to minimize the Makespan and maximize the reliability. They evaluated the method over randomly generated graphs and some application graphs. The obtained results were compared to "Heterogeneous Earliest Finish Time Algorithm" (HEFT) [28] and "Critical Path Genetic Algorithm" (CPGA) [29]. Consequently, the comparison results demonstrated that NSGA outperforms those two competitors in terms of both objectives.

Chitra et al. [16] took two conflicting objective functions namely, makespan and reliability into account for solving multi-objective multiprocessor scheduling problem. The authors proposed hybrid algorithms by mixing the multi-objective evolutionary methods with simple neighborhood local search. They also compared SPEA2 [21] and NSGAll [17] algorithms with each other in both pure and hybrid version. The evaluation results indicated that hybrid version of NSGAll outperforms the other methods involved in comparison.

4 The proposed ensemble system for solving multi-objective multiprocessor scheduling problem

This section describes the proposed innovative ensemble system comprising six population-based metaheuristics for solving bi-objective and three-objectives multiprocessor scheduling problem in heterogeneous distributed systems. The ensemble system provides an environment for different metaheuristics to cooperate and collaborate in order to achieve common goals. The ensemble system operates based on exploration and exploitation mechanisms, in which, the exploration is carried out by six metaheuristics and exploitation is done by local search technique. To be defined shortly, the exploration technique explores the search space by big jumping steps and exploitation mechanism exploits the neighborhood area of solutions. The proposed ensemble system is learning-based and success-proportional, working based on a suggested strategy which is accurately described in the following. The novelty of this ensemble method is mostly related to cooperation strategy. Figure 3 illustrates the architecture of proposed ensemble system.

The system consists of six metaheuristics namely, MOGA [18], IMOPSO [20], MODE [19], SPEA2 [21], NSGAI [17] and MOABC [22], which share a common solution pool. The system performs based on the suggested strategy. Figure 4 demonstrates the flowchart of proposed ensemble strategy.

The system starts with some initializations in which it initiates the success rate of all metaheuristics by 50 (to give same chance for metaheuristics to be selected) and public solution pool by random. All the metaheuristics use the same solution representation, hence any converting is needless. A solution (scheduling) is represented as a two dimensional array with 2 rows and n columns, where n is the number of tasks in graph. For instance, a scheduling for the graph represented in Fig. 2 over three processors is given in Fig. 5.

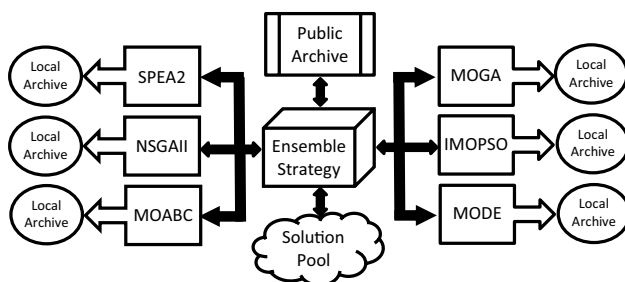


Fig. 3 Proposed ensemble system

Public solution pool is initialized randomly using the algorithm shown in Fig. 6. The processors are selected randomly but the tasks are chosen according to a topological sorting order. The advantage of this algorithm is being able to generate diverse, valid and random solutions in which Generated solutions are different in terms of tasks order and processors. The algorithm keeps all tasks ready to be executed in ReadyTasks list and chooses tasks one by one randomly from this list to generate solutions. One task is ready to execute if all of its parents have been already executed. For this, algorithm assigns a parent counter to each task and decreases it when a parent is executed. This way, a task is added to ReadyTasks list, when the parent counter becomes zero.

In the next step, the system calculates the values of either two or three objectives according to the descriptions and formulas in Sect. 2. As an example, algorithm for computing the Makespan is given in Fig. 7. In the algorithm, $AT[t_i]$ is the time that task t_i would be ready to be run and $FT[t_i]$ is the finish time of task t_i . Also, the $P[p_i]$ keeps the time which processor p_i will become idle. $\#p$ and $\#t$ denote for number of processors and number of tasks respectively. At the end, the longest busy time or maximum $P[p_i]$ among all processors is taken into account as Makespan of schedule.

Later on, the system works in consecutive sessions until the termination criteria are satisfied. Each session consists of two steps based on exploration and exploitation. System explores the problem search space by running metaheuristics to discover more promising regions of the space. Meanwhile, an exploitation process is done over the extracted Pareto-front by local search approach.

In the exploration phase, the dominance rank of all solutions is computed so that the dominance rank of a solution s is the number of other solutions dominating s . This rank value represents the goodness of solution, in which the lower rank means better solution. It is obvious that the best solutions in population have rank value equal to zero and such solutions are called as non-dominated solutions. Then, the ensemble strategy selects a subpopulation from public solution pool using the algorithm represented in Fig. 4. The selection is carried out using tournament selection. Afterwards, system chooses a metaheuristic among six available metaheuristics based on their success rate using the selection algorithm presented in Fig. 4. Then, selected metaheuristic is executed over the selected subpopulation and the quality of final subpopulation is calculated to find the extracted Pareto-front. Crossover and mutation operators are carried out in a way that they generate feasible solutions. In the crossover operation, the two-point crossover is applied only on the processors. This way only the processors are combined and the tasks order remains same. Figure 8 indicates the crossover algorithm.

Fig. 4 Proposed ensemble strategy

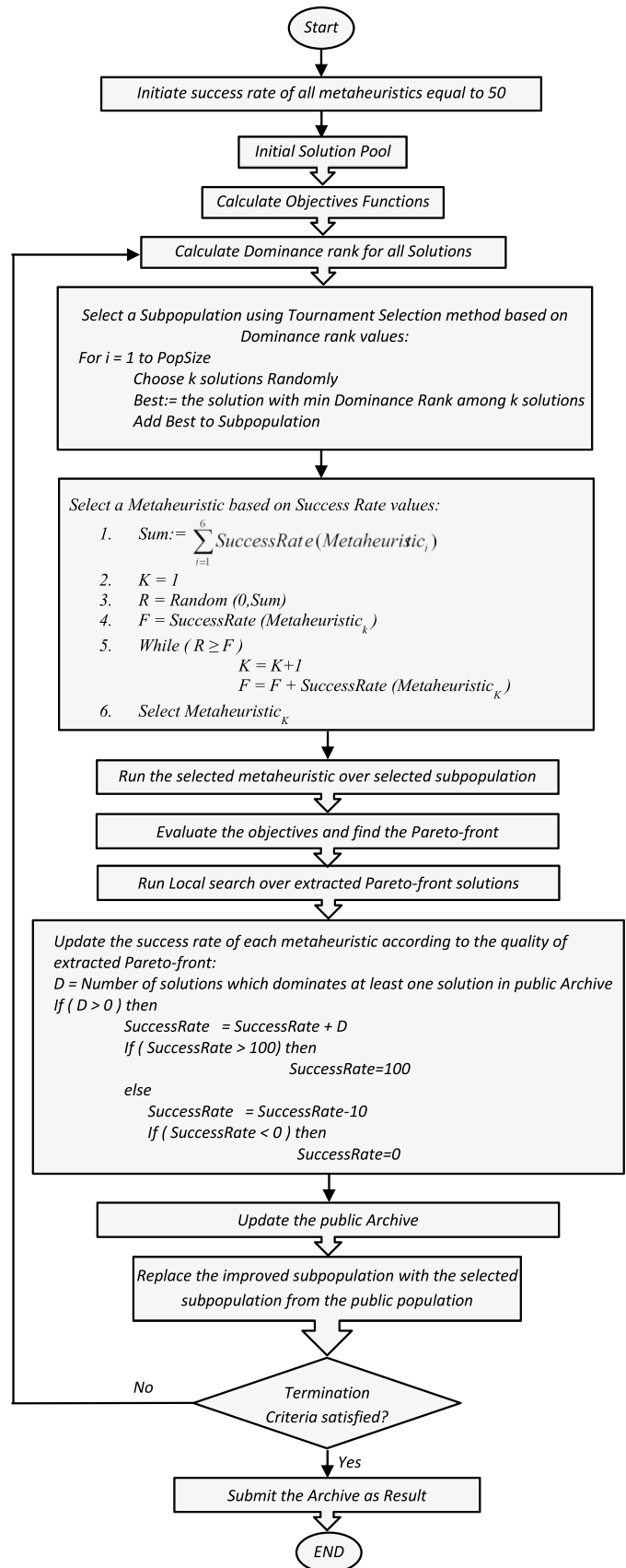


Fig. 5 A sample solution for task graph in Fig. 1

t ₀	t ₂	t ₃	t ₁	t ₄	t ₅	t ₆	t ₁₁	t ₁₃	t ₁₂	t ₉	t ₁₀	t ₈	t ₇	t ₁₄	t ₁₅	t ₁₆	t ₁₇	t ₁₈
p ₀	p ₁	p ₂	p ₀	p ₁	p ₂	p ₁	p ₁	p ₂	p ₁	p ₀	p ₀	p ₀	p ₂	p ₀	p ₁	p ₀	p ₁	p ₁

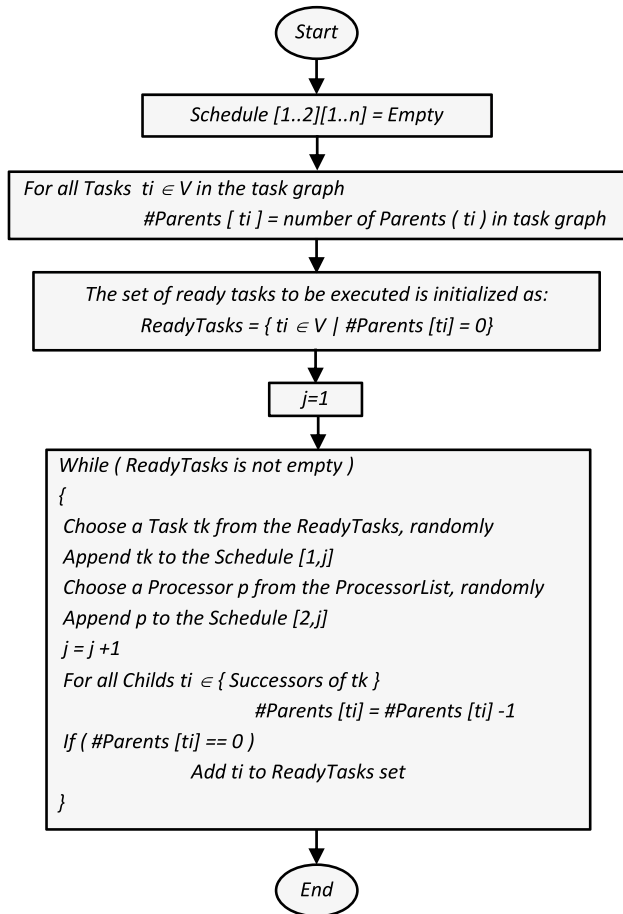


Fig. 6 Schedule initialization algorithm

The proposed method for mutation operator modifies the order of tasks as well. Moreover, it selects a few processors and changes them randomly. The mutation algorithm is presented in Fig. 9. The algorithm specifies a random point on solution between 1 and number of tasks. Then it keeps the tasks order unmodified until that random point, but the order of tasks after the point is modified randomly. The modification is carried out as the process in Fig. 4 in which after the specified point, the rest of solution is regenerated randomly. The advantage of proposed mutation operator is that mutated solution is changed in terms of both tasks and processors and it makes the algorithm jumping through search space and extract more promising solutions.

Later on, the exploitation phase is carried out over the extracted Pareto-front by using local search algorithm. This way, the non-dominated solutions in Pareto-front are adjusted and become more accurate. Local search

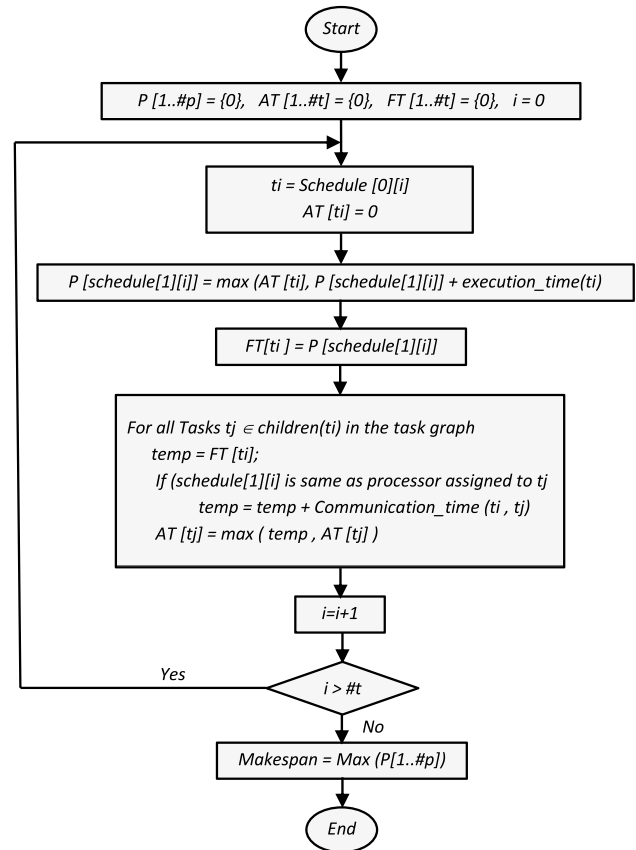


Fig. 7 Makespan calculation algorithm

method adjusts and improves the solutions by applying very small modifications in processors. Due to time limitations, a fast local search is used in this phase. It tries at most 10 solutions nearby to find a better solution and replace it by the old one. The system updates the success rate of the selected metaheuristic according to its success and amount of improvement achieved during the execution using the algorithm shown in Fig. 4. Success rate is increased if the resulted Pareto-front dominates one or more solutions of public archive. Otherwise, the success rate is decreased by 10. Also, success rate value is bounded between 0 and 100. Hence, this value is used as selection probability of metaheuristic in the ensemble strategy. In the rest of session, the public archive is updated with applying the new extracted Pareto-front by the selected metaheuristic. Finally, at the end of session, the final subpopulation is replaced by the selected subpopulation from the solution pool.

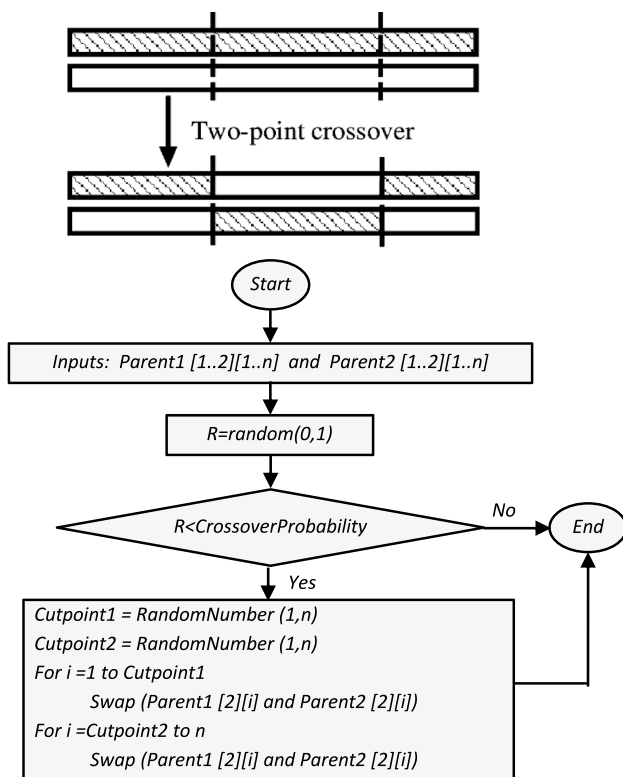


Fig. 8 Two-point crossover

To be explained in more details, the solution pool keeps the public population to be used by all metaheuristics, in which, the metaheuristics select subpopulations from the solution pool, improve it and move back to the solution pool. This way, system provides an environment for metaheuristics to collaborate and cooperate. Ensemble system prefers many short runs of metaheuristics instead of one single long run of a specific metaheuristic, because using only one metaheuristic leads to some problems like early convergence and etc. Also, a single metaheuristic has some inabilities in searching whole search space [30]. Advantage of using multiple metaheuristics in system is that each metaheuristic can cover the inabilities of other metaheuristics in searching the more promising parts of search space. All metaheuristics used in the system are implemented according to the standard algorithms introduced in literature. Proposed ensemble system uses success rate values and learns gradually how good the metaheuristics are. This way, system selects the next running metaheuristic based on their success rate values. Apart from all metaheuristics have same success rate in the beginning, values are changed based on performance of metaheuristics during execution. This way, good metaheuristics and worse metaheuristics have high chance and low chance to be selected respectively.

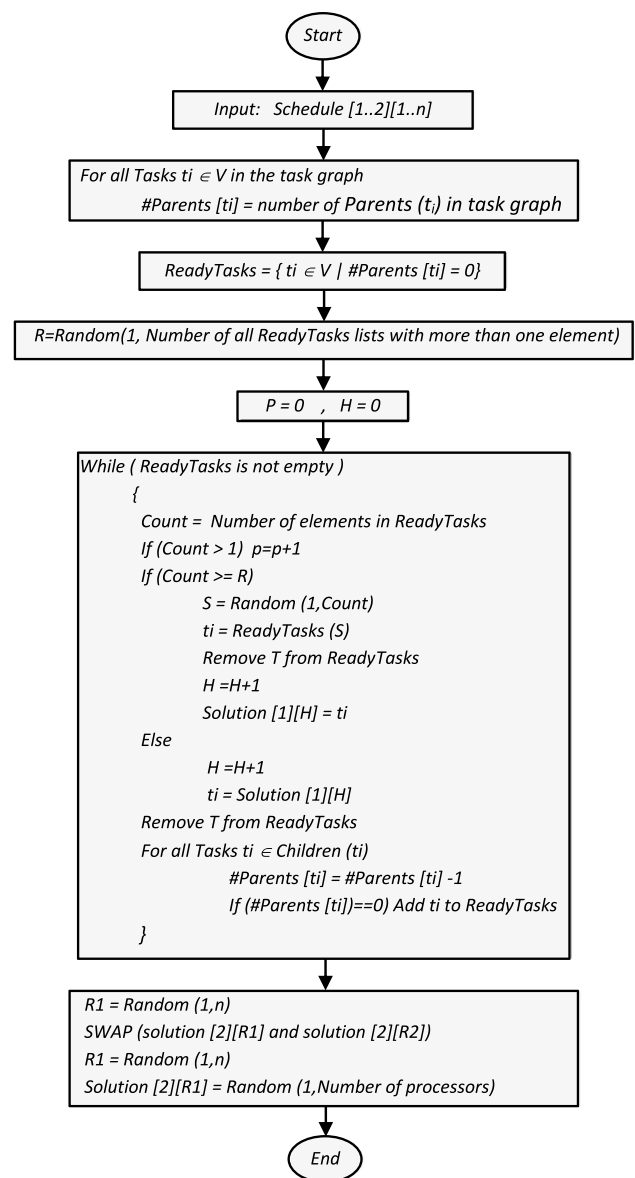


Fig. 9 The proposed mutation algorithm

5 Evaluation and experimental results

This section demonstrates the evaluation results of applying the proposed ensemble system over well-known benchmarks reported in state-of-the-art literature [6–8, 16, 26, 27]. The effectiveness of suggested system is evaluated over practical Bi-objective and Three-objective Multiprocessor Scheduling Problems [6, 27]. Table 1 presents the values of all parameters related to each metaheuristic within the proposed ensemble system. The parameter values in Table 1 have been collected from well-known conventional implementations of algorithms. Introduced ensemble system has been implemented in Matlab® programming language environment. In the evaluation

Table 1 Parameter values of metaheuristics within the suggested ensemble system

Metaheuristic	Parameter values
MOGA	Pop =40, P _C =0.8, P _m =0.2, Gaussian_Sigma_Pm=20
IMOPSO	Pop =40, C1=2.0, C2=2.0, ω _{max} =0.9, ω _{min} =0.4
MODE	Pop =40, Scaling_Factor=0.5, P _C =0.8,
SPEA2	Pop =40, P _C =0.9, P _m =1.0/Num_Vars, Distribution_Index=20
NSGAI	Pop =40, P _C =0.9, P _m =1.0/Num_Vars, Distribution_Index=20,
MOABC	Pop =40, MNC=500, SNOV=80, Limit=-MNC×(#of decision variables)/2

process, the number of generations and number of processors are taken into account with respect to the values reported in related literature [6–8, 16, 26, 27].

Also, the population size is set as 40 for all algorithms. Since there are six algorithms, the total population size becomes 240 which is less than or equal to what the state-of-the-art methods considered. Likewise the number of generations is set according to values reported in literature. Therefore the time complexity of proposed ensemble method is same as state-of-the-art methods reported in literature. However, based on the following experiments, even though the time complexity is same, the quality of extracted Pareto-front is higher than state-of-the-art methods.

In all experiments carried out in this section, the Pareto-front True (PF-Optimal) is not known in literature. For this reason, some of performance metrics could not be calculated. Therefore, the metrics which do not require the researcher to know Pareto-front True are calculated. In order to measure diversity of extracted Pareto-front, the spacing metric can be used as follows [8]. Spacing metric measures the distribution of vectors in PF.

$$S = \sqrt{\frac{1}{|PF| - 1} \sum_{i=1}^{|PF|} (\bar{d} - d_i)^2} \tag{9}$$

where $d_i = \min_j (|f_1^i(x) - f_1^j(x)| + |f_2^i(x) - f_2^j(x)|)$; $i, j = 1, 2, 3, \dots$, number of vectors in PF, \bar{d} is the mean of all d_i 's.

The second metric to be calculated is hypervolume (HV) which is a comprehensive indicator [31]. HV measures both convergence and spread performance of discovered Pareto-front. A larger HV indicates better performance. Hypervolume represents the sum of the areas enclosed within the hyper cubes formed by the points on the Pareto-front and a chosen reference point [31]. Reference points are considered as (0, 0) and (0, 0, 0) for two and three objectives Pareto-fronts.

The first evaluation is carried out over a real world problem which is the Gaussian Elimination Graph (GE) [32]. The algorithm for Gaussian Elimination is given in Fig. 10 where m is the matrix dimension. Also, $T_{k,k}$ and $T_{k,j}$

```

for k = 1 to m-1 do
    Tk,k: { for i = k+1 to m do
              aik = aik / aik }
    for j = k+1 to m do
        Tk,j: { for i = k+1 to m do
                  aij = aij - aik * akj }
    
```

Fig. 10 Gaussian elimination algorithm

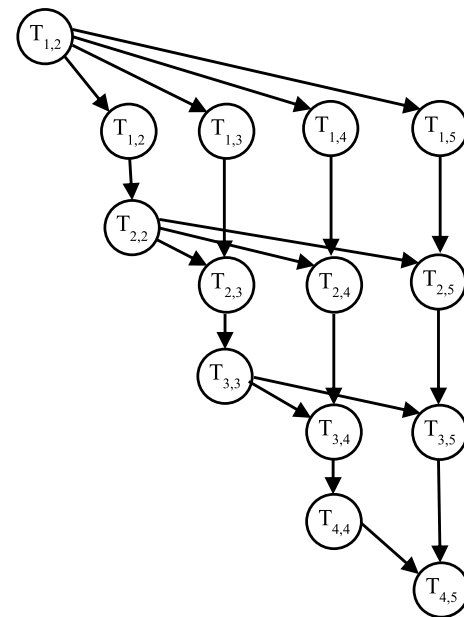


Fig. 11 Gaussian elimination graph for $m=5$

represent a pivot column operation and update operation respectively.

The Gaussian Elimination Graph for $m=5$ is drawn in Fig. 11.

In the evaluation done in [7], a GE graph with matrix size equal to 10 is considered. Therefore, the number of nodes in graph is 54. Moreover, two objectives are taken into account namely, Makespan and Reliability index

Table 2 Makespan and Reliability index values obtained by FA, MGA, BGA and Ensemble system

CCR	FA		MGA		BGA		Ensemble system	
	Makespan	Reliability	Makespan	Reliability	Makespan	Reliability	Makespan	Reliability
1	458	9.45	591	13.17	616	9.48	420	8.29
5	687	7.4	1070	15.93	1103	12.20	657	6.76
10	1144	15.48	1426	23.54	1490	22.47	1069	14.83

[7]. The comparison is done against three competitors namely Firefly based algorithm (FA) [7], Modified Genetic Algorithm (MGA) [9] and Bi-objective Genetic Algorithm (BGA) [13] as reported in [7]. The competitors use the following equation to weight the objective functions [9]:

$$f(x) = w \times \frac{F_1 - MinF_1}{maxF_1 - minF_1} + (1 - w) \times \frac{F_2 - MinF_2}{maxF_2 - minF_2} \tag{10}$$

where w is the coefficient value which specifies the importance percentage of objectives in which, when w is 1, makespan is only considered objective.

Table 2 shows the objective values obtained by the proposed ensemble system and three competitors for various CCR (Communication to computation ratio) values (1, 5 and 10). Equation 2 shows how to calculate CCR.

$$CCR = \frac{\text{Average Communication Cost}}{\text{Average Computation Cost}} \tag{11}$$

The objective values shown in Table 2 for ensemble system are chosen from the set of solutions in extracted Pareto Front. Also, the matrix size for Gaussian Elimination Graph, value of w and number of processors are considered as 10, 0.5 and 4 respectively.

The proposed ensemble system discovers a Pareto Front as problem solution. This way, it would be possible to choose diverse solutions from Pareto front based on requirements as we chose for Table 2. As it is shown in Table 2, the proposed system has obtained (420, 8.29), (657, 6.76) and (1069, 14.83) as values of makespan and reliability for CCR= 1, CCR=5 and CCR= 10 respectively. It can be seen that all obtained results are better than values obtained by other three methods (FA, MGA and BGA). Therefore the proposed ensemble system outperforms its competitors in terms of both objectives.

The Pareto Fronts including 10, 8 and 12 solutions obtained by ensemble system for CCR= 1, 5 and 10 are presented in Figs. 12, 13 and 14 respectively.

The metric values calculated for the obtained Pareto-front are given in Table 3.

The second experiment is done over a Parallel Gaussian elimination graph based on results reported in [8]. The authors in [8] used an 18 tasks Gaussian Elimination graph to compare two different algorithms namely Genetic

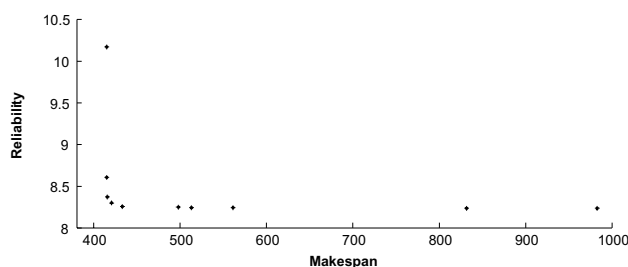


Fig. 12 Pareto front extracted by ensemble system for CCR= 1

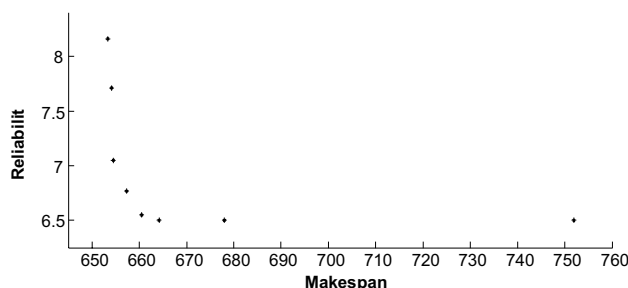


Fig. 13 Pareto front extracted by ensemble system for CCR= 5

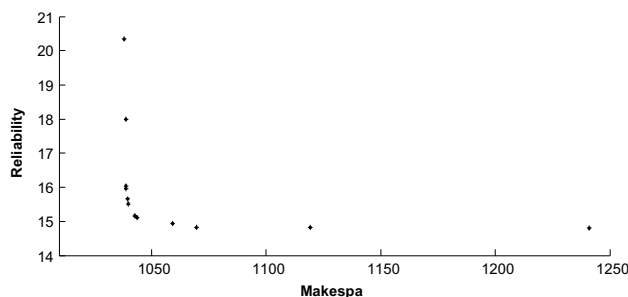


Fig. 14 Pareto front extracted by ensemble system for CCR= 10

Table 3 Spacing and hypervolume values of PF obtained by hybrid method

CCR	Spacing	Hypervolume
1	33.12	0.7203645
5	26.32	0.90128477
10	32.06	0.79129405

Algorithms (GA) and Evolutionary Programming (EP) [13]. The test graph is presented in Fig. 15 [13, 23, 33].

Table 4 shows the objective values (Makespan and Reliability index) obtained by the proposed ensemble system

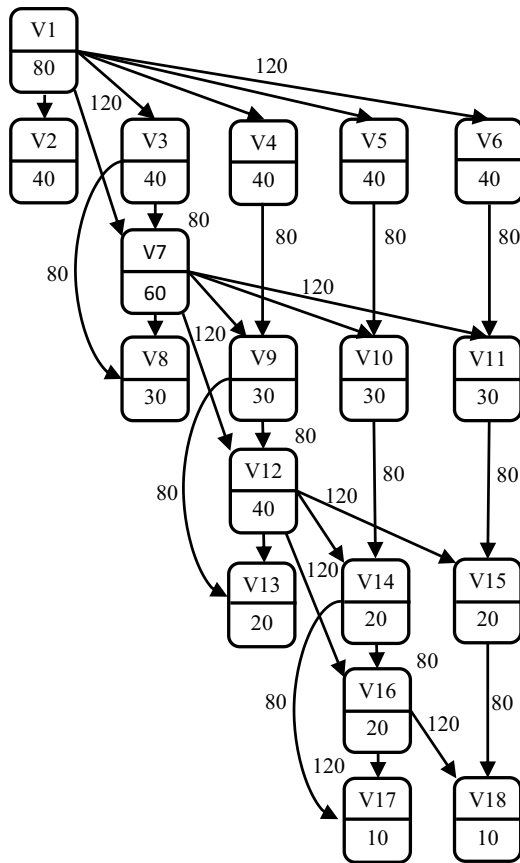


Fig. 15 Gaussian elimination graph

and two competitors in [8] namely GA and EP [13] over 4 processors.

According to Table 4, the proposed system has obtained (410.46, 3.92) as best values of makespan and reliability for CCR=2. It can be seen that all obtained results are better than values obtained by other three methods (GA, EP). Therefore the proposed ensemble system outperforms its competitors in terms of both objectives.

Figure 16 shows the Pareto Front extracted by the Ensemble system including 10 solutions over 5 processors. Values in Table 3 have been selected from Pareto front. Authors in [8] also applied MOGA and MOEP over the same task graph and reported the extracted Pareto-front as shown in [8] over 5 processors. The extracted Pareto-front

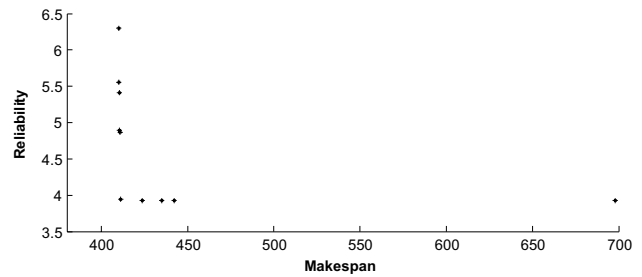


Fig. 16 Pareto front extracted by ensemble system for CCR=2

Table 5 Spacing and hypervolume values of PF obtained by hybrid method

Spacing	Hypervolume
14.29	0.7905164

by the ensemble system is better than those reported by MOGA and MOEP in [8].

The metric values calculated for the obtained Pareto-front obtained by MOGA, MOEP and hybrid method are given in Table 5.

The next evaluation is carried out for comparing proposed ensemble system to those reported in [16]. The test graph is the same graph as Fig. 5 with CCR=2 and 3 processors. Authors in [16] solved this problem using Genetic Algorithms, Evolutionary Programming and Hybrid Genetic Algorithm. The objective values (Makespan and Reliability index) obtained by the proposed ensemble system and three competitors in [16] are presented in Table 6.

As it is shown in Table 6, the proposed system has obtained 471.23 and 3.62 as values of makespan and reliability respectively. It can be seen that all obtained results are better than values obtained by other three methods (GA, EP and HGA). Therefore the proposed ensemble system outperforms its competitors in terms of both objectives.

We continue the evaluation of the proposed ensemble system with results reported in [6]. Authors in [6] solved three objective multiprocessor scheduling problem using Genetic Algorithm (GA) and Evolutionary Programming (EP) over Gaussian Elimination Graph with the matrix size of 10. The considered objectives are Makespan, Average Flow-time and Reliability. Objective values (Makespan, Average Flow-time and Reliability) obtained

Table 4 Makespan and Reliability index values obtained by GA, EP and Ensemble system

CCR=2	Best makespan			Best reliability		
	GA	EP	Ensemble system	GA	EP	Ensemble system
Makespan	467	472	410.46	600	604	423
Reliability	6.01	6.81	5.13	4.16	4.54	3.92

Table 6 Makespan and reliability values obtained by GA, EP, HGA and ensemble system

	Best makespan				Best reliability			
	GA	EP	HGA	Ensemble system	GA	EP	HGA	Ensemble system
Makespan	584	594	562	471.23	961	964	1243	661.43
Reliability index	14.88	15.77	13.37	8.85	6.64	7.19	4.35	3.62

Table 7 Makespan and reliability index values obtained by GA, EP and ensemble system

	Best makespan			Best average flow-time			Best reliability		
	GA	EP	Ensemble system	GA	EP	Ensemble system	GA	EP	Ensemble system
Makespan	416	419	404.10	603	632	483.88	810	818	502.76
Average flow-time	372.75	368.5	303.59	280.75	292	266.06	281.25	284.55	276.34
Reliability index	7.18	7.02	5.18	5.6	5.6	3.81	3.07	3.10	2.83

by the proposed ensemble system and two competitors in [6] namely GA and EP on 4 processors are presented in Table 7.

As it can be seen in Table 7, the proposed system has obtained 404.10, 266.06 and 2.83 as values of makespan, average flow-time and reliability respectively. It is clear that all obtained results are better than values obtained by other three methods (GA, EP). Therefore the proposed ensemble system outperforms its competitors in terms of both objectives.

The Pareto fronts extracted by MOGA and MOEP are reported in [6]. Figure 17 shows the Pareto Front extracted by the Ensemble system including 65

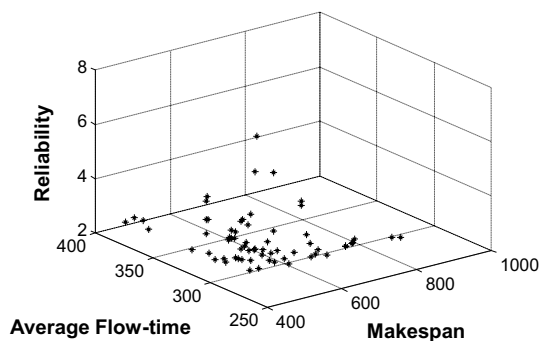


Fig. 17 Pareto front extracted by proposed ensemble system

Table 8 Makespan and reliability index values obtained by NSGAll, MFA and ensemble system

	Best makespan			Best average flow-time			Best reliability		
	NSGAll	MFA	Ensemble system	NSGAll	MFA	Ensemble system	NSGAll	MFA	Ensemble system
Makespan	62	59	52	65	61	59	65	61	59
Average flow-time	25	25	21	24	24	19	24	24	21
Reliability	53	51	53	49	50	46	49	50	43

solutions. The values in Table 5 have been selected from the Pareto front. The spacing and hypervolume values for obtained Pareto-front are 22.64 and 0.8193348 respectively.

Authors in [5] have evaluated their proposed MFA method against NSGAll over the test graph presented in Fig. 2 on 4 processors. They have considered Makespan, Average Flow-time and Reliability as problem objectives.

The objective values (Makespan, Average Flow-time and Reliability) obtained by the proposed ensemble system and two competitors in [5] namely NSGAll and MFA are presented in Table 8.

The proposed system has obtained (52, 19, 43) as values of makespan, average flow-time and reliability respectively. It can be seen that all obtained results are better than values obtained by other three methods (NSGAll and MFA). Therefore the proposed ensemble system outperforms its competitors in terms of both objectives.

Figure 18 demonstrates the Pareto Front extracted by the Ensemble system including 52 solutions. Values in Table 6 have been selected from the Pareto front. Meanwhile, the Pareto fronts reported in [5] consist of only 3 non-dominated solutions. The spacing and hypervolume values for obtained Pareto-front are 48.05 and 0.9544932 respectively.

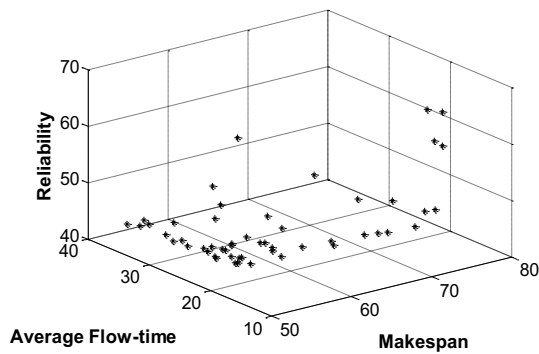


Fig. 18 Pareto front obtained by proposed ensemble system

In the next evaluation step, the proposed ensemble system is evaluated over two well-known benchmarks. The first one is Fast Fourier Transformation (FFT) task graph which is shown in Fig. 19 [34].

The multobjective type of FFT graph includes Makespan, Average flow time and Reliability values. Table 9 demonstrates the comparison between the proposed

ensemble system and all metaheuristics used within the system individually on 4 processors.

It can be observed from Table 9 that proposed ensemble system performs better than its individual metaheuristics. Figure 20 shows the Pareto Front extracted by the Ensemble system including 52 solutions.

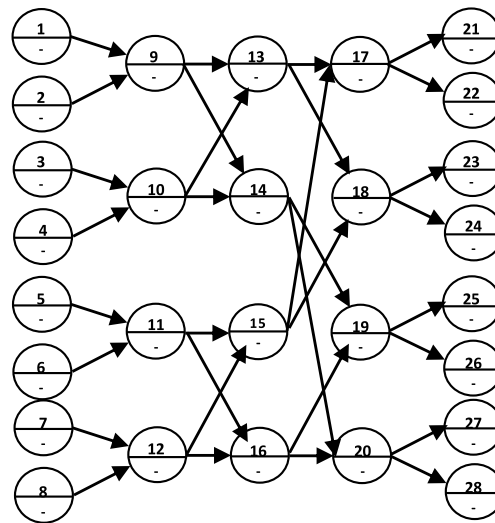
The second benchmark is Internal Rate of Return (IRR) which is presented in Fig. 21 [34].

The multobjective type of IRR graph includes Makespan, Average flow time and Reliability values. Table 10 compares the obtained results of proposed ensemble system and all individual metaheuristics used within the system on 4 processors.

It can be observed from Table 10 that proposed ensemble system performs better than its individual metaheuristics. Figure 22 shows the Pareto Front extracted by the Ensemble system including 67 solutions.

The Final comparison is carried out with results reported in [15]. The suggested method HEFT-NSGA [15] was compared to HEFT [28] and CPGA algorithms [29].

Fig. 19 FFT task graph



Vertex#	FFT-1	FFT-2	FFT-4
1 – 8	1	60	20
9 – 12	20	50	20
13 – 16	30	5	30
17 – 20	20	5	20
21 – 28	1	5	5

Table 9 Objective values obtained by ensemble system and individual metaheuristics for FFT1

	Best makespan			Best average flow-time (AFT)			Best reliability index		
	Makespan	AFT	Reliability	Makespan	AFT	Reliability	Makespan	AFT	Reliability
Ensemble system	126	168	144	148	118	158	148	128	112
MOGA	173	191	188	212	158	182	202	188	158
MOPSO	175	201	212	201	178	201	224	191	177
MODE	148	178	184	168	148	193	181	182	152
SPEA2	152	221	178	178	188	178	178	206	151
AMOSa	168	210	192	191	178	183	191	189	166
NSGAII	142	181	201	148	159	198	166	162	144
MOABC	153	198	235	212	168	224	202	198	189

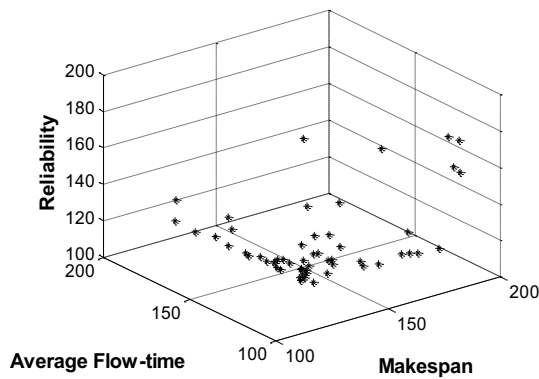


Fig. 20 Pareto-front extracted by ensemble system over FFT1

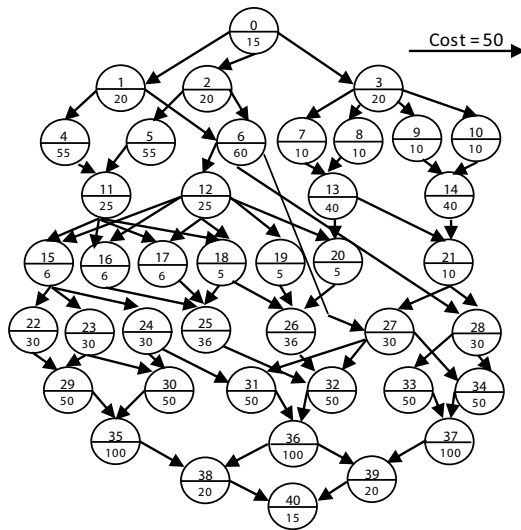


Fig. 21 Internal rate of return graph [34]

Results are reported over the Gaussian Elimination and Fast Fourier Transformation graphs [34]. One more comparison metric was considered in [15], Schedule Length Ratio (SLR), which is calculated as follows:

$$SLR = \frac{Makespan}{\sum_{v_i \in CP_{min}} \min(cost(v_i))} \tag{12}$$

where CP is the critical path of the graph which is based on minimum computation costs. SLR is the summation of the minimum computation costs of tasks on the critical path. An algorithm with lowest SLR is the best algorithm. Figure 23 represents the Makespan, SLR and Reliability values for HEFT, CPGS, HEFT-NSGA algorithms and proposed ensemble system over Gaussian Elimination graph on 4 processors.

Figure 24 represents the Makespan, SLR and Reliability values for HEFT, CPGS, HEFT-NSGA algorithms and proposed ensemble system over FFT graph.

It is observed that the proposed ensemble method reaches better performance compared to the CPGS, HEFT and HEFT-NSGA.

In order to illustrate the robustness of the proposed approach against replacement of a metaheuristics with a new one or adding a new one to the existing system, Table 11 is provided over task graph presented in Fig. 21.

It can be seen that proposed system is robust against replacement and insert actions in which the results are very similar and there is no significant difference.

6 Conclusions and future works

This paper presents a novel method of ensemble learning for solving multi-objective multiprocessor scheduling problem in heterogeneous distributed systems. Multiprocessor scheduling problem is of great importance in distributed systems in which it aims to distribute a parallel program through the processors in order to optimize objective functions. The objective functions to be considered are namely makespan, average flow-time and reliability. The proposed ensemble system comprises six metaheuristics working on a shared population and

Table 10 Objective values obtained by ensemble system and individual metaheuristics for IRR

	Best makespan			Best average flow-time (AFT)			Best reliability index		
	Makespan	AFT	Reliability	Makespan	AFT	Reliability	Makespan	AFT	Reliability
Ensemble system	475	643	414	580	523	465	600	596	381
MOGA	580	688	525	623	645	525	645	688	464
MOPSO	600	688	568	720	645	525	704	691	464
MODE	580	721	525	720	712	668	754	782	521
SPEA2	640	665	725	710	702	735	715	606	545
AMOSa	600	786	715	680	675	668	755	789	464
NSGAII	515	643	490	590	585	490	625	662	420
MOABC	625	743	774	680	675	820	710	798	686

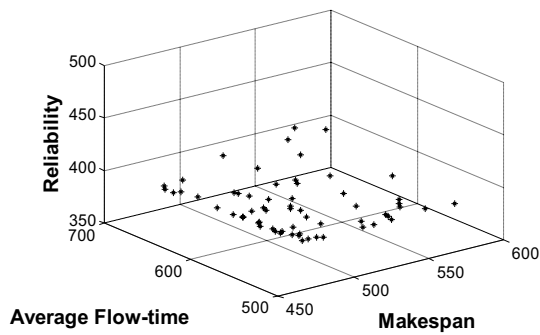


Fig. 22 Pareto-front extracted by ensemble system over IRR graph

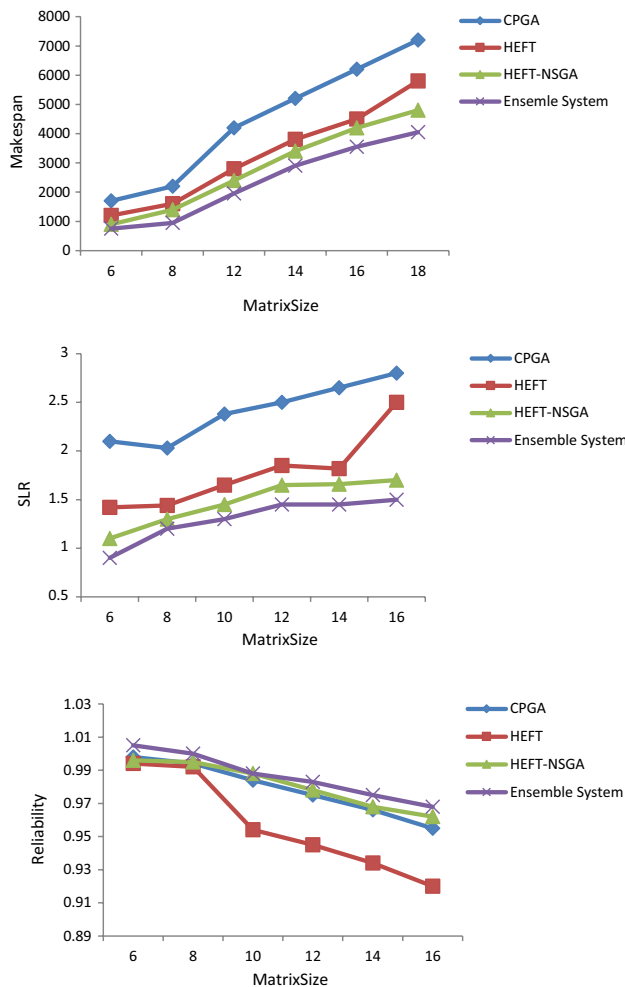


Fig. 23 Makespan, SLR and reliability index of CPGS, HEFT and HEFT-NSGA for Gaussian elimination graph

cooperating based on a proposed strategy. To increase exploration efficiency, subpopulations and metaheuristics are selected based on dominance rank and success-rate values respectively. This way, system can learn gradually how effective the metaheuristics and how good the

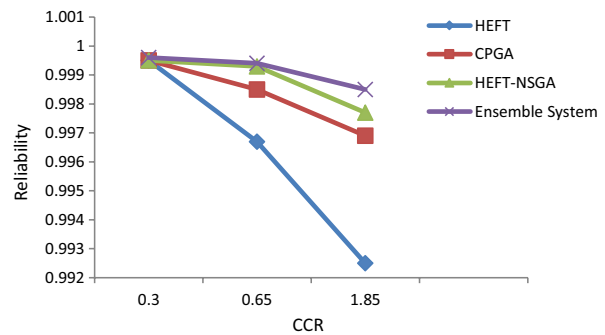
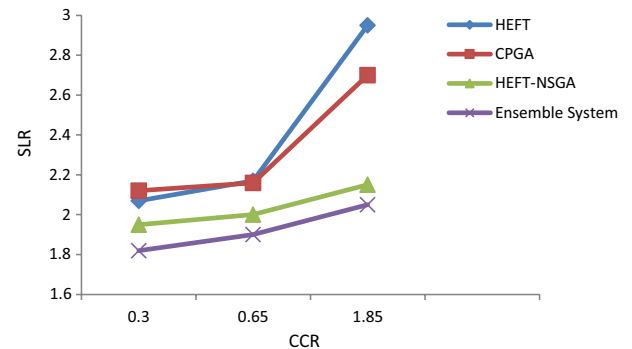
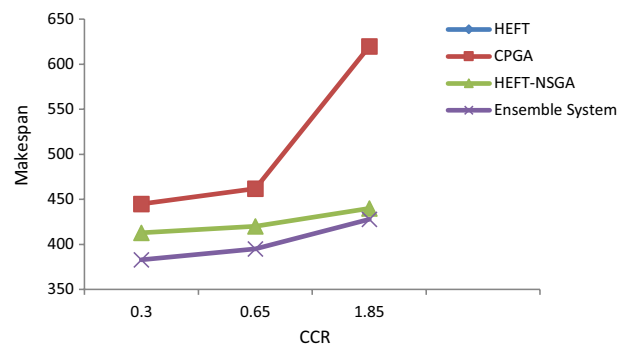


Fig. 24 Makespan, SLR and reliability of CPGS, HEFT and HEFT-NSGA for FFT graph

population members are. Likewise, the collaboration of metaheuristics makes it possible to discover more promising parts of search space. All non-dominated solutions found during execution of the system are kept by local and global archives considered in proposed framework. Briefly, the proposed ensemble system prefers short-term running of different metaheuristics instead of a long-term running of a single metaheuristic. Obtained results demonstrate that proposed ensemble system outperforms all state-of-the-art methods reported in literature. Also, it can be observed that proposed ensemble system performs better than its individual metaheuristics in terms of objective functions.

Further works are planned to use the proposed system with better strategies over different problems e.g. task

Table 11 Comparison of objective functions values by replacement of metaheuristics or adding new ones

	Best makespan			Best average flow-time (AFT)			Best reliability index		
	Makespan	AFT	Reliability	Makespan	AFT	Reliability	Makespan	AFT	Reliability
Ensemble system	475	643	414	580	523	465	600	596	381
Adding MOABC	473	645	473	580	523	469	612	588	379
Adding AMOSA	473	648	490	591	526	456	621	602	381
MOABC instead of MOGA	481	633	408	591	521	478	598	596	381
MOABC instead of NSGAI	475	710	423	580	523	478	600	615	388
AMOSA instead of MOGA	477	694	414	602	532	455	600	588	385
AMOSA instead of NSGAI	471	694	414	578	528	499	608	596	379

scheduling in cloud computing. Likewise, it would be beneficial to test the replacement and extension of ensemble system with different and new metaheuristics. In addition to this, the proposed ensemble system is suitable to be used in distributed and parallel processing environments.

Compliance with ethical standards

Conflict of interest The author states that there is no conflict of interest.

References

- Abraham A, Lakhmi J, Goldberg R (2005) Evolutionary multi-objective optimization. Springer, Berlin
- Bosman PAN (2011) On gradients and hybrid evolutionary algorithms for real-valued multi-objective optimization. *IEEE Trans Evolut Comput* 16(1):51–69
- Galgali VS, Ramachandran M, Vaydia GA (2019) Multi-objective optimal sizing of distributed generation by application of Taguchi desirability function analysis. *SN Appl Sci*. <https://doi.org/10.1007/s42452-019-0738-3>
- Tey JY, Shak KPY (2019) Multi-objective optimization of virtual formula powertrain design for enhanced performance and fuel efficiency. *SN Appl Sci*. <https://doi.org/10.1007/s42452-019-1082-3>
- Lotfi N, Acan A (2013) Solving multiprocessor scheduling problem using multi-objective mean field annealing. In: 14th IEEE international symposium on computational intelligence and informatics (CINTI 2013), pp 113–118
- Chitra P, Revathi S, Venkatesh P, Rajaram R (2010) Evolutionary algorithmic approaches for solving three objectives task scheduling problem on heterogeneous systems. In: IEEE 2nd international advance computing conference (iACC), pp 38–43
- Eswari R, Nicholas S (2012) Solving multi-objective task scheduling for heterogeneous distributed systems using firefly algorithm. In: Fourth international conference on advances in recent technologies in communication and computing (ARTCom2012), Springer, pp 57–60
- Chitra P, Venkatesh P, Rajaram R (2011) Comparison of evolutionary computation algorithms for solving bi-objective task scheduling problem on heterogeneous distributed computing systems. *Indian Acad Sci* 36:167–180
- Sathappan OL, Chitra P, Venkatesh P, Prabhu M (2011) Modified genetic algorithm for multiobjective task scheduling on heterogeneous computing system. *Int J Inf Technol Commun Converg* 1(2):146–158
- Correa RC, Ferreira A, Rebreyend P (1999) Scheduling multiprocessor tasks with genetic algorithms. *IEEE Trans Parallel Distrib Syst* 10:825–837
- Omara FA, Arafa MA (2010) Genetic algorithms for task graph scheduling. *J Parallel Distrib Comput* 70:13–22
- Fontes D, Gaspar-Cunha A (2010) On multi-objective evolutionary algorithms: handbook of multicriteria analysis. Applied optimization. Springer, Berlin. https://doi.org/10.1007/978-3-540-92828-7_10
- Dogan A, Ozguner F (2005) Biobjective scheduling algorithms for execution time-reliability trade-off in heterogeneous computing systems. Technical report 2005-001
- David BF, Lawrence LJ (1996) Using evolutionary programming to schedule tasks on a suite of heterogeneous computers. *Comput Oper Res* 23:527–534
- Chen Y, Li D, Ma P (2012) Implementation of multi-objective evolutionary algorithm for task scheduling in heterogeneous distributed systems. *J Softw*. <https://doi.org/10.4304/jsw.7.6.1367-1374>
- Chitra P, Rajaram R, Venkatesh P (2011) Application and comparison of hybrid evolutionary multiobjective optimization algorithms for solving task scheduling problem on heterogeneous systems. *Appl Soft Comput* 11:2725–2734
- Deb K, Agrawal S, Pratap A, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evolut Comput* 6(2):182–197
- Fonseca CM, Fleming PJ (1993) Genetic algorithm for multiobjective optimization, formulation, discussion and generalization. In: Proceeding of the fifth international conference genetic algorithms, pp 416–423
- Robic T, Filipi B (2005) DEMO: differential evolution for multi-objective optimization. In: Proceedings of the 3rd international conference on evolutionary multi-criterion optimization. Springer, pp 520–533
- Dehuri S, Cho SB (2009) Multi-criterion Pareto based particle swarm optimized polynomial neural network for classification. A review and state-of-the-art. *Comput Sci Rev* 3(1):19–40
- Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the strength Pareto evolutionary algorithm for multiobjective optimization. *Evolutionary methods for design optimization and control with applications to industrial problems*, pp 95–100
- Xiang Y, Zhou Y, Liu H (2015) An elitism based multi-objective artificial bee colony algorithm. *Eur J Oper Res* 245:168–193

23. Kwok YK, Ahmad I (1996) Dynamic critical-path scheduling: an effective technique for allocating task graphs to multiprocessors. *IEEE Trans Parallel Distrib Syst* 7:506–521
24. Parsa S, Lotfi S, Lotfi N (2007) An evolutionary approach to task graph scheduling. In: *ICANNGA 2007*. Springer, pp 110–119
25. Al-Mouhamed A (1990) Lower bound on the number of processors and time for scheduling precedence graphs with communication costs. *IEEE Trans Softw Eng* 16(12):1390–1401
26. Srichandan S, Kumar TA, Bibhudatta S (2018) Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm. *Future Comput Inform J*. <https://doi.org/10.1016/j.fcij.2018.03.004>
27. Dongarra JJ, Jeannot E, Saule E, Shi Z (2007) Bi-objective scheduling algorithms for optimizing makespan and reliability on heterogeneous systems. In: *19th annual ACM symposium on parallel algorithms and architectures (SPAA'07)*
28. Ma P, Lee E, Tsuchiya M (1982) A task allocation model for distributed computing systems. *IEEE Trans Comput* 100:41–47
29. Omara FA, Arafa MM (2009) Genetic algorithms for task scheduling problem. In: Abraham A, Hassanien AE, Siarry P, Engelbrecht A (eds) *Foundations of computational intelligence*, vol 3. Springer, Berlin, pp 479–507
30. Lotfi N, Acan A (2017) A multi-agent dynamic rank-driven multi-deme architecture for real-valued multi-objective optimization. *Artif Intell Rev* 48:1–29
31. Han Y, Li J, Gong D, Sang H (2019) Multi-objective migrating birds optimization algorithm for stochastic lot-streaming flow shop scheduling with blocking. *IEEE Access* 3:5946–5962. <https://doi.org/10.1109/ACCESS.2018.2889373>
32. Wu M, Gajski D (1990) Hypertool: a programming aid for message passing systems. *IEEE Trans Parallel Distrib Syst* 3:330–343
33. Topcuoglu H, Hariri S, Wu W (2002) Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE Trans Parallel Distrib Syst* 13:260–274
34. Lotfi N, Acan A (2015) Learning-based multi-agent system for solving combinatorial optimization problems: a new architecture. In: *10th international conference on hybrid artificial intelligent systems (HAIS)*, pp 319–332

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.