IJE TRANSACTIONS A: Basics Vol. 36 No. 07, (July 2023) 1310-1321

International Journal of Engineering

Journal Homepage: www.ije.ir

Rescheduling Unreliable Service Providers in a Dynamic Multi-objective Cloud Manufacturing

M. M. Fazeli*, Y. Farjami, A. Jalaly Bidgoly

Department of Computer and IT, University of Qom, Qom, Iran

PAPER INFO

Paper history:
Received 17 March 2023
Received in revised form 02 May 2023
Accepted 04 May 2023

Keywords:
Cloud Manufacturing
Dynamic Rescheduling Problem
Multi-objective Optimization
Rescheduling Unreliable Service

ABSTRACT

Cloud manufacturing (CMfg) is a new advanced manucatring model developed with the help of enterprise information technologies under the support of cloud computing, Internet of Things and service-based technologies. CMfg compose multiple manufacturing resources to provide efficient and valuable services. CMfg has a highly dynamic environment. In this environment, many disruptions or events may occur that lead the system to unplanned situations. In CMfg, a series of service providers are scheduled for production. During the production operation, some of them may be damaged, stopped, and out of service. Therefore, rescheduling is necessary for the continuation of the production process according to the concluded contracts and initial schedule. When any disruptions or other events occurred, the rescheduling techniques used to updating the inital schedule. In this paper, the dynamic rescheduling problem in CMfg is analyzed. Then the multi-objective rescheduling in CMfg is modeled and defined as a multi-objective optimization problem. Defining this problem as a multi-objective optimization problem provides the possibility of applying, checking and comparing different algorithms. For solving this problem, previous optimization methods have improved and a multi-objective and elitist algorithm based on the Jaya algorithm, called advanced multi-objective elitist Jaya algorithm (AMEJ) is proposed. Several experiments have been conducted to verify the performance of the proposed algorithm. Computational results showed that the proposed algorithm performs better compared to other multi-objective optimization algorithms.

doi*: 10.5829/ije.2023.36.07a.12*

1. INTRODUCTION

Cloud manufacturing is service-oriented and has a sharedable set of diverse and distributed production resources. CMfg aims to increase the efficiency of the production process, reduce manufacturing costs and optimize the use of resources by creating temporary, reconfigurable production lines to respond to customers' requests [1]. As a result, CMfg can satisfy users' specific manufacturing tasks by sharing on-demand networked manufacturing services [2].

CMfg compose multiple manufacturing resources to provide efficient and valuable services. With the help of CMfg resources, capabilities and manufacturing services can be shared [3]. In the CMfg environment, finding a better manufacturing resources composition is needed to achieve high-efficiency manufacturing processes [4].

CMfg has a highly dynamic environment. In this environment, many disruptions or events may occur that lead the system to unplanned situations. So the initial schedule needs to be reviewed. When any disruptions or other events occurred, the rescheduling techniques used to updating the inital schedule [5]. One of these events is service provider failures. Failure or inaccessibility of the service provider puts the initial schedule in an invalid state and it is not possible to continue the production process so reviewing the initial schedule is vital and unavoidable.

Rescheduling in CMfg should be considered as a multi-objective problem because 1) Several objectives must be simultaneously optimized 2) determining constraints in a single-objective optimization problem requires determining customer preferences, which is not always possible and correct 3) multi-objective approaches provide multiple options as result, each of which can be selected according to the system conditions and the opinion of the experts.

*Coresponding Author Institutional Email: m.fazeli@stu.qom.ac.ir (M. M. Fazeli)

This paper analyses the dynamic rescheduling problem in CMfg. Then, the multi-objective rescheduling in CMfg is modeled and a multi-objective optimization problem is defiend based on the propsed model. To solve the rescheduling problem in CMfg, a multi-objective and elitist algorithm based on the Jaya algorithm, called advanced multi-objective elitist Jaya algorithm (AMEJ) is proposed.

There are various algorithms for solving multi-objective optimization problems, most of which require parameters tuning for proper execution. But AMEJ is a parameter-less algorithm that does not require any parameters for the algorithm to works correctly.

The proposed algorithm is based on Jaya algorithm. Jaya is a parameter-less simple algorithm that multiple experiments showed that it has better performance than other multi-objective optimization algorithms [6].

AMEJ adds new operators to the base Jaya algorithm, which improves its performance and results. The computational results show that the proposed algorithm performs better than the base Jaya algorithm and other compared optimization algorithms.

The innovations of this article can be summarized as follows:

1. Mathematical modelling of rescheduling problem in CMfg.
2. Defining the multi-objective rescheduling problem in CMfg as a multi-objective optimization problem that provides the possibility of applying, checking and comparing different algorithms on it.
3. Proposing a competitive and appropriate algorithm to solve this problem.

The rest of the paper is as follows. In section 2, the previous works are studied. Section 3 provides some preliminaries, mathematical model of CMfg, and objective functions for SBCOS and rescheduling problem. Section 4 proposes the AMEJ algorithm. The experiments results and analysis are presented in section 5. Finally, section 6 concludes this research and outlines the directions for future researches.

2. RELATED WORKS

In recent years, solving the SBCOS problem and scheduling in CMfg have been considered by a lot of researchers. In our previous work [7], we proposed an ensemble optimization approach that can be used as a flexible framework. This approach combins multiple optimization methods to improve service composition performance. Zhang and Ren [8] propose a multi-agent simulation model for CMfg. Their model can evaluate the manufacturing process and rescheduling strategies which can help to obtain more accurate value of parameters. Zhang et al. [9] considered disruption like random service breakdowns and proposed a framework for rescheduling the initial schedule of multiple distributed production services. Liu et al. [10] proposed a new efficient way to scheduling a multi-task problem in CMfg based on workload criteria. This work provides general guidance to schedule multiple tasks with different workloads in CMfg under different circumstances. Yang et al. [11] have investigated on scheduling multi-population competitive field resources in CMfg. Their results showed that the FSRS-CMfg model can improve the quality of service critriation of scheduling the field manufacturing resource and proposed a higher precision of convergence without extensively increasing in the manufacturing time. Liu et al. [12] presented a model to solve the personal recommendation issue in CMfg and they proposed a swarm-based optimization algorithm named glow-worm for solving the multi-objective optimization problem. Zhou and Yao [13] proposed an artificial optimization algorithm to solve the service composition problem in CMfg based on cooperation of bees in a bee colony which uses multiple algorithms for selection of an optimal combination of services and considered QoS and randomly arriving of tasks as criterion.

Rescheduling and real-time scheduling is an interesting topic in CMfg and manufacturing. Zhang et al. [6] proposed MMSC model for composition of multi-task manufacturing services that considers multiple tasks in an uncertain environment to solve uncertainty problems such as urgent tasks and delivery delays; also a heuristic algorithm was proposed for the finding an optimal manufacturing service composition. Zhou et al. [14] proposed a model for simulation of dynamic service scheduling in CMfg. This simulation can be done from demanders, tasks, resources, or path perspective or combination of them. Zhou et al. [15] presented an event-based dynamic task scheduling approach to solve the scheduling problem in the CMfg that tasks randomly arrive into the the system. In previous researches various models and algorithms have been proposed, there are some researches that focoused on scheduling models for multi-objective or single-objective problems [16], or association analysis approach [17] and Colony based Optimization algorithms like Ant Colony. But there are less studies on CMfg dynamic scheduling [18].
Champati and Liang [19] proposed a huristic algorithm. Their algorithm can calculate the manufacturing time when a task canceled or rescheduled. They also used simulation to compare the performance of the proposed algorithm with other algorithms. Liu et al. [20] proposed a model for dynamic CMfg scheduling problem that considers dynamic task arrivals. In this model, the failure types and causes of exception conditions faced by cloud services are considered for updating programs and rescheduling production.

Most of the optimization algorithms use multiple parameters and need parameters tuning [21]. But AMEJ is a parameter-less algorithm that does not require any parameters for the algorithm to works correctly. The Jaya algorithm and teaching-learning-based optimization (TLBO) algorithm are two popular parameter-less algorithms.

Rao et al. [22] introduced the TLBO algorithm. This algorithm has two phases; the teacher phase and the learner phase. The population of this algorithm is a group of learners. This population is updated and improved in the two phases mentioned [22]. After the success of previous algorithm, Rao [23] proposed another parameter-less algorithm named Jaya. This one-phase algorithm is simple and fast. Also Rao [23] showed that Jaya has better performance compared to other optimization algorithms.

A review of the existing research literature showed that there are a few studies that have been conducted in relation to dynamic scheduling of CMfg [18]. Various evaluation indicators currently used in CMfg scheduling; most of them evaluate service composition based on quality of service. Yang et al. [24] used six second-level indexes (importance, supply and demand, cost, remaining time, reputation and predetermined cost) as indices for the service composition evaluation. Based on the evaluation of cloud service composition reputation (CSCR), Xie et al. [25] took two types of stability and collaboration ability as the first-level evaluation indices of the service composition and three types (execution time, cost and reliability) as the second-level indices. Li et al. [26] proposed six indices (reliability, reputation, combination collaboration, combination complexity, execution time and execution cost) to evaluate service composition.

Chen et al. [27] studied real-time scheduling problem in the cloud. Resource allocation is the key feature that they considered for scheduling. They also proposed an artificial neural network model for prediction of the status of the task completion. This process can better allocate resources. In other study Zheng and Zheng [28] proposed a simulation-based approach that combines the simulation with some network models. The aims of this approach is to analysis the robustness of cloud manufacturing systems.

Other important articles in the subject area of this research are: Arkat et al. [29] on their article entitled “Reactive Scheduling Addressing Unexpected Disturbance in Cellular Manufacturing Systems” focussed on scheduling in CMfg. Puspitasari et al. [30] on their work entitled “Generator Scheduling Optimization Involving Emission to Determine Emission Reduction Costs” also paid attention on cost anlysis of CMfg. Najafi and Nikaeen [31] addressed on their work for a constraint programming approach in order to solve multi-skill resource-constrained project on scheduling problem with calendars. Torkashvand et al. [32] discussed about the distributed production assembly scheduling using hybrid flowshop in assembly lines. Maghzi et al. [33] investigated on optimization of operating room scheduling using a fuzzy uncertainty approach and metaheuristic algorithms. Halty et al. [34] discussed on Scheduling in cloud manufacturing systems. Rashidifar et al. [35] also presented a mathematical model for cloud-based scheduling using heavy traffic limit theorem in queuing process. Abtahi et al. [36] studied on stochastic model for the prioritized outpatient scheduling in a radiology center for the purose of effective and efficient services given to patients. Yazdi et al. [37] conducted an investigtaion by using a mathematical model for scheduling elective surgeries in order to minimize the waiting list in the emergency surgeries.

3. PRELIMINARIES & BACKGROUND MODELS

3. 1. Machine Breakdown and Rescheduling

In the manufacturing process, any machine may break down and be out of reach for a certain period of time. In this article, we assumed that one of the service provider’s machines breaks down for T unit of times and after this amount of time it can be used again. As a result, we have two unscheduled events, one for when the machine breaks down and becomes unavailable, and the second for when it becomes available again.

Rescheduling is used to update the initial schedule when events occurred that may lead the system to a unstable state [38]. So after any unscheduled events such as machine breakdowns, the initial schedule must be reviewed and the rescheduling process must be performed.

Figures 1 and 2 show the schedules and the impact of a breakdown for three machines and four requests, each with three subtasks. For simplicity, in this example, we assumed that each service provider has only one machine. Figure 1(a) shows the Gantt chart of the initial schedule and the subtasks affected by the second machine failure. Figure 1(b) shows the modified schedule after the second machine becomes unavailable. Figure 2(a) shows the tasks that are affected by the re-availability of the second machine and Figure 2(b) shows the final schedule after the completion of the second rescheduling process.

As shown in Figure 2(a), after the second machine is available again, ongoing requests are completed and the rescheduling process is done only for requests that have not yet started.

3. 2. Non-dominated Sorting

The non-dominated sorting approach is a process for classifying the population in several ranks or Pareto fronts (PF). This approach is based on the Pareto dominance concept which is described for this research as follow [38]:

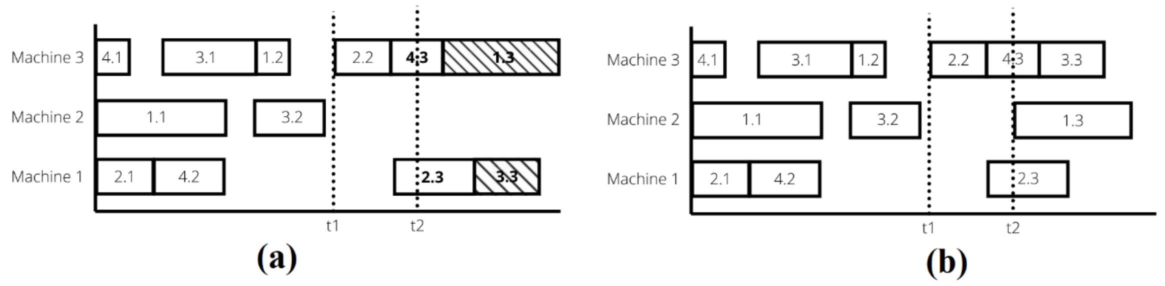

Figure 1. Schedule of 3 machines and 4 requests; (a) initial schedule; (b) reviewed schedule after the second machine breaks down

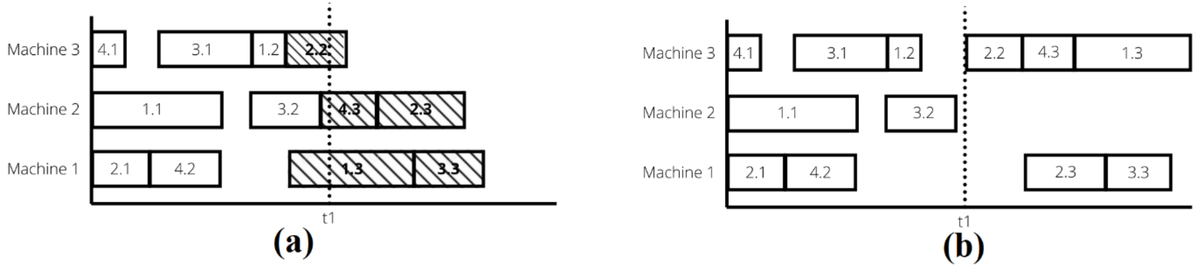

Figure 2. Schedule of 3 machines and 4 requests; (a) the schedule after the second machine breaks down and the subtasks that were affected by the re-availability of the second machine; (b) reviewed schedule after the second machine becomes available

$Solution_j \prec Solution_i \leftrightarrow$
$\forall f \in \{TCT, TC, SI, STT\}\ f(Solution_j) \leq f(Solution_i)$ &
$\exists f' \in \{TCT, TC, SI, STT\}\ f(Solution_j) < f(Solution_i)$

Where $Solution_i$ and $Solution_j$ are two solutions in result space and $Solution_j \prec Solution_i$ means $Solution_i$ dominates $Solution_j$. In the non-dominated sorting approach, for a set of populations, all non-dominated solutions that are not dominated by any other solutions are found. These solutions are rank one (first Pareto front). Then these solutions are deleted from the population and the same process is repeated until all solutions are ranked in their respective front.

3. 3. Crowding Distance The purpose of computing crowding distance (denoted by CD_i for crowding distance of $Solution_i$) is to determine the density of solutions around each solution in the result space. The following algorithm can be used to compute the crowding distance of each solution in the front F [21]:

Crowding Distance Computation Algorithm

```
1. $l = |F|$
2. for each $i$ in $F$ set $CD_i = 0$
3. for each $f$ in $\{TCT, TC, SI, STT\}$
    3.1 Sort $F$ in worst order of $f$
    3.2 $CD_1 = CD_l = \infty$
    3.3 $f_{min}$ = minimum value of $f$ in $F$
    3.4 $f_{max}$ = maximum value of $f$ in $F$
    3.5 for $j = 2$ to $j = l - 1$ set $CD_j = CD_j + \frac{f(j+1) - f(j-1)}{f_{max} - f_{min}}$
```

Based on the above algorithm, the crowded-comparison operator ($\prec_n$) is defined as follow:

$$Rank_j < Rank_i \mid (Rank_j = Rank_i \ \& \ CD_j > CD_i) \rightarrow Solution_j \prec_n Solution_i$$

Where $Rank_i$ and $Rank_j$ are the fronts to which $Solution_i$ and $Solution_j$ belong respectively.

3. 4. Constraint-Dominance Concept a $Solution_i$ is said to constrained-dominate a $Solution_j$ if $Solution_i$ is feasible and $Solution_j$ is not feasible or both solutions are infeasible and $Solution_i$ has a smaller overall constraint violation or both solutions are feasible and $Solution_i$ dominates $Solution_j$. This concept gives feasible solutions a higher rank than infeasible solutions. Between the feasible solutions, the superior (non-dominated) solution, and between the infeasible solutions, the solution with the lowest value of overall constraint violation will have a higher rank [21].

3. 5. Jaya Algorithm The Jaya algorithm used by multiple researchers in thier applications. It's a parameter-less algorithm and in this algorithm, after generating a random initial population, in each iteration, each of the solutions is updated using Equation (1):

$$Solution_{k+1,i,j} = Solution_{k,i,j} + rand_1\left(BS_{k,j} - \left|Solution_{k,i,j}\right|\right) - rand_2\left(WS_{k,j} - \left|Solution_{k,i,j}\right|\right) \quad (1)$$

where $Solution_{k,i,j}$ is jth objective function of ith solution in kth iteration and $BS_{k,j}$ and $WS_{k,j}$ are the jth objective function value of the best and worst solution in kth iteration respectively. $rand_1$ and $rand_2$ are two random numbers in between 0 and 1 [23].

3. 6. Analyzing Dynamic Rescheduling Problem in CMfg

The following flowchart shows the rescheduling problem in a dynamic CMfg environment. The following abbreviation is used to simplify the flowchart:

S: Broken service
Z: Alternative service
R(S): The time required to repair S
T(S): The time required to complete the subtask by S
W(Z): The time remaining until the start of the next subtask of Z
T(Z): The time required to complete the sub-task by Z
T'(Z): The time remaining until the completion of the current subtask of Z

After receiving, the client request is virtualized and decomposed and then saved in the subtasks database. Also, all the services, including manufacturing services and logistics services, are virtualized and saved in the services database.

Then the scheduler generates the initial schedule with the help of information stored in both databases. The schedule is given to the services and the actual manufacturing process start.

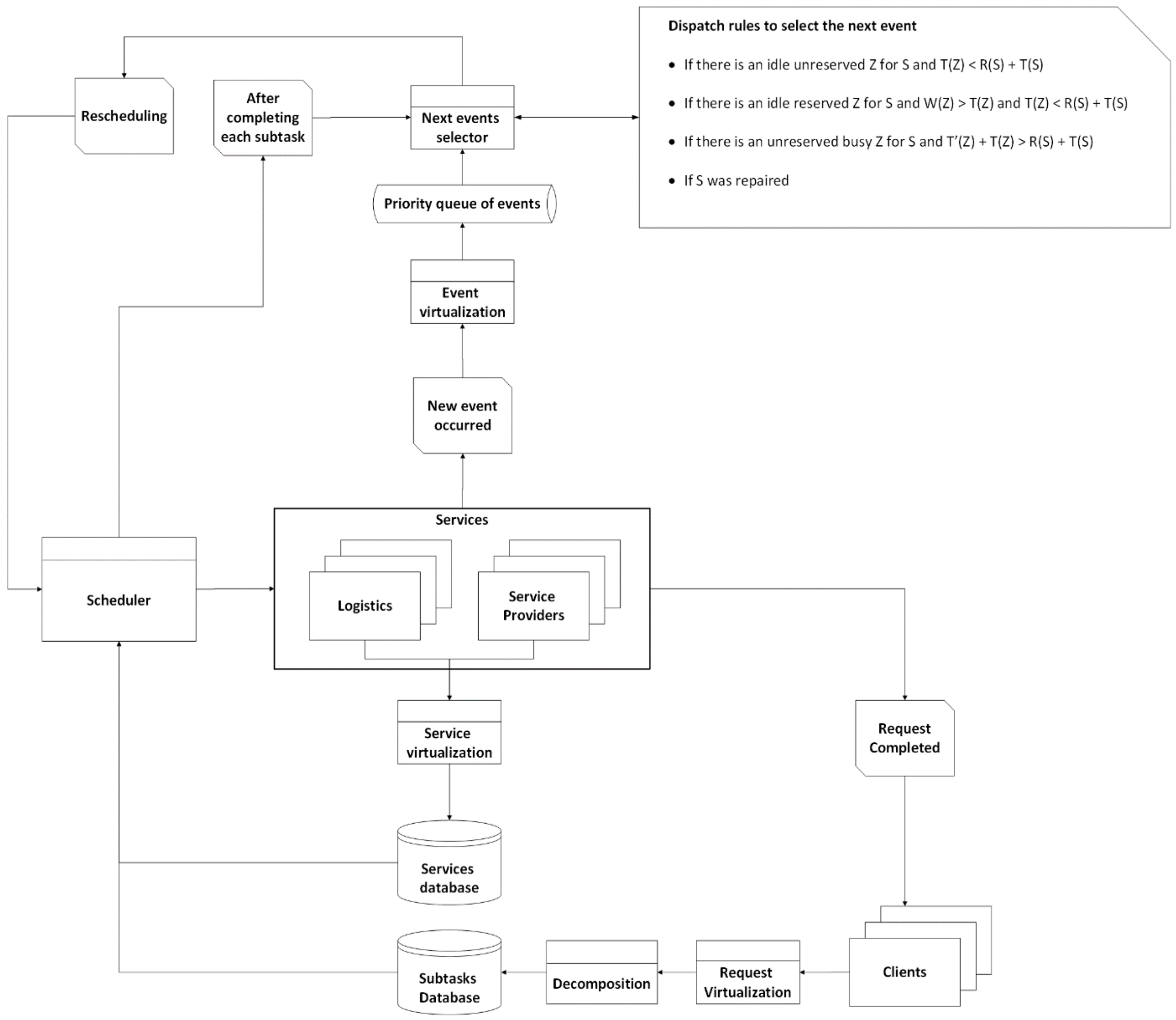
Dispatch rules to select the next event
If there is an idle unreserved Z for S and T(Z) < R(S) + T(S)
If there is an idle reserved Z for S and W(Z) > T(Z) and T(Z) < R(S) + T(S)
If there is an unreserved busy Z for S and T'(Z) + T(Z) > R(S) + T(S)
If S was repaired
Rescheduling
After completing each subtask
Next events selector
Priority queue of events
Event virtualization
New event occurred
Services
Logistics
Service Providers
Scheduler
Service virtualization
Services database
Request Completed
Clients
Request Virtualization
Decomposition
Subtasks Database

Figure 3. Dynamic rescheduling problem in CMfg

If any of the services break down during the manufacturing process, the event will be virtualized and stored in a priority queue. The next event is selected from the queue according to the following dispatching rules. These rules are in order of priority so the first rule is superior to the second one and so on.

1. If there is an idle unreserved Z for S and T(Z) < R(S) + T(S)
2. If there is an idle reserved Z for S and W(Z) > T(Z) and T(Z) < R(S) + T(S)
3. If there is an unreserved busy Z for S and T'(Z) + T(Z) > R(S) + T(S)
4. If S was repaired

Then the selected event information and the candidate services are given to the scheduler to revise the schedule and generate a new schedule.

3. 7. CMfg Model In this article, we assume that there are several customers in the CMfg system whose requests have reached the system simultaneously. The ith request submitted by the customers denoted by Req_i where $i \in \{1,2,3,\dots,nReq\}$ and $nReq$ is the total number of requests.

Multiple subtasks (denoted by ST) should performe to complete a request. So for ith request, we have $Req_i = \{ST_{i,1}, ST_{i,2}, ST_{i,3}, \dots, ST_{i,j}\}$ where $ST_{i,j}$ is jth subtask in the subtasks sequence of Req_i. Total number of subtasks in a Req_i is shown with nST_i so $j \in \{1,2,3,\dots,nST_i\}$.

In the CMfg system, three main processes are done. First, all requests of customers are processed and decomposed, and subtasks sequences for each request are created. Then service provider (SP) send their manufacturing information to the CMfg system.

In the next process, based on subtasks data of requests and providers' data, the CMfg system tries to find the most appropriate service for each subtask of each request. In the CMfg for each subtask of a request, multiple candidate services exist that can do that subtask (candidate services for subtask j of ith request denoted by $CS_{i,j}$). If we show the total number of services in $CS_{i,j}$ with $nCS_{i,j}$, there are $\prod_i^n \prod_{j=1}^{m_i} nCS_{i,j}$ of paths to do all requests.

After fininshing the previous process, the service sequence is created based on the subtasks and services information. When the CMfg system processes are done, Based on the information like sequence of services and assigned tasks, each service provider will know its job.

3. 8. Objective Functions The rescheduling problem in the CMfg is a multi-objective optimization problem. the priori and a posteriori approaches are the main two ways that used for solving a multi-objective optimization problem: Priori approach converts a multi-objective optimization problem into a single objective optimization problem by using some weights for the objective functions. In this approach, user preferences are considered as weights of objective functions. But the posteriori approach can find multiple solutions for a multi-objective optimization problem [21].

In this research, the posteriori approach has been used to solve the multi-objective rescheduling problem in the CMfg. The objective functions used in this problem can be divided into two categories: global objective functions such as total completion time (TCT) and total cost (TC), and rescheduling-specific objective functions such as solution instability (SI) and subtask tardiness (STT).

a. Total Completion Time (TCT): this objective function can be calculated by the following equation [6]:

$$TCT = TMT + TLT + TWT \tag{2}$$

In this equation TMT is the total time needed to complete the request, TLT is the total time that consumed by the logistic processes and TWT is the total amount of time for just waiting for another proccess to be completed or receving the material for starting a procces. These values can be calculated by Equations (2)-(4):

$$TMT = \sum_{i=1}^{nReq} \sum_{j=1}^{nST_i} MT_{i,j} \tag{3}$$

$$TLT = \sum_{i=1}^{nReq} \sum_{j=1}^{nST_i} LT(SP_{i,j}, SP_{i,j+1}) \tag{4}$$

$$TWT = \sum_{i=1}^{nReq} \sum_{j=1}^{nST_i} WT(SP_{i,j}) \tag{5}$$

where $MT_{i,j}$ is the manufacturing time to complete $ST_{i,j}$, and $LT(SP_{i,j}, SP_{i,j+1})$ is the logistic time between service providers selected to perform $ST_{i,j}$ and $ST_{i,j+1}$, and $WT(SP_{i,j})$ is waiting time to complete $ST_{i,j}$ on its corresponding service provider.

b. Total Cost (TC): this objective function can be calculated by the following equation:

$$TC = TMC + TLC \tag{6}$$

In this equation TMC and TLC are the total cost that the manufacturing and the logistic processes needed respectively. These values can be calculated by Equations (7)-(8):

$$TMC = \sum_{i=1}^{nReq} \sum_{j=1}^{nST_i} MC(SP_{i,j}) \tag{7}$$

$$TLC = \sum_{i=1}^{nReq} \sum_{j=1}^{nST_i} LC(SP_{i,j}, SP_{i,j+1}) \tag{8}$$

where $MC(SP_{i,j})$ is the manufacturing cost of $SP_{i,j}$, and $LC(SP_i, SP_{i+1})$ is the logistic cost to perform logistic service between $SP_{i,j}$ and $SP_{i,j+1}$.

c. Solution Instability (SI): Instability is defined as the number of subtasks assigned to another service provider for performing after rescheduling [39]. This objective function is between zero and one, and the smaller it is, the greater stability of the solution. SI can be calculated by Equation (9):

$$SI = \frac{\sum_{i=1}^{nReq}\sum_{j=1}^{nST_i} CH_{i,j}}{\sum_{i=1}^{nReq} nST_i}$$
$$CH_{i,j} = \begin{cases} 0, & SP_{i,j}\ remains\ for\ ST_{i,j} \\ 1, & SP_{i,j}\ is\ changed\ for\ ST_{i,j} \end{cases} \tag{9}$$

Where $SP_{i,j}$ is the selected service provider to perform the jth subtask of ith request.

d. Subtask Tardiness (STT): This objective function is the difference between the initial schedule start time of subtasks and their real start time. STT can be calculated by Equation (10):

$$STT = \sum_{i=1}^{nReq}\sum_{j=1}^{nST_i} FinalST_{i,j} - InitST_{i,j} \tag{10}$$

Where $InitST_{i,j}$ is the initial schedule start time of $ST_{i,j}$ and $FinalST_{i,j}$ is the final schedule start time of $ST_{i,j}$.
The overall multi-objective rescheduling problem is summarized as follows:

$$min\ \{TCT, TC, SI, STT\}$$
$$subject\ to: \begin{cases} TCT < TCT_0 \\ TC < TC_0 \\ SI < SI_0 \\ STT < STT_0 \end{cases}$$

where TCT_0, TC_0, SI_0 and STT_0 are the maximum time, cost, solution instability, and subtask tardiness based on customer preferences, respectively.

4. THE PROPOSED ALFORITHM FOR RESCHEDULING

To solve the rescheduling problem in CMfg, this paper proposed a multi-objective and elitist algorithm based on Jaya, called advanced multi-objective elitist Jaya algorithm (AMEJ). AMEJ is a parameter-less algorithm so there is no need for tuning any parameters for proper execution.

There are two lists in AMEJ: $BestList$ to maintain the best solutions and $WorstList$ to keep the worse solutions in each iteration.

In each iteration, the best solution (denoted by $best$) and the worst solution (denoted by $worst$) are found using the non-dominated sorting approach, constraint-dominance concept and crowding distance computation. Then all solutions in $BestList$ that is dominated by $best$ are removed from this list and the $best$ solution is added to $BestList$. Similarly, all solutions in $WorstList$ that dominates $worst$ are removed from this list and the $worst$ solution is added to $WorstList$. This process removes the dominated solutions (that is worse than the current $bset$ solution) from $BestList$ and the dominant solutions (that is better than current $worst$ solution) from $WorstList$.

Then both lists are sorted by the non-dominated sorting approach, constraint-dominance concept, and crowding distance computation and the best solution from $BestList$ and the worst solution from $WorstList$ are found. These solutions are used to update the population-based on Equation 10 according to the Jaya algorithm. If these solutions have been used for more than two iterations to update the population, they will be removed from the relevant lists and other solutions will be found and used to update the population. This mechanism will lead to better dispersibility of solutions and will allow the appropriate solutions the opportunity to be reused again.

The updated population is then concatenated with the initial population and the whole population is sorted by non-dominated sorting approach, constraint-dominance concept, and crowding distance computation. Then, using a random binary variable, it is determined to separate the number of initial population or twice the initial population from the best individuals of the whole sorted population. This population is used in the next iteration. This approach allows more individuals to be used to search the problem space in some iterations randomly. The flowchart for the AMEJ algorithm is shown in Figure 4.

5. EXPERIMENTS AND DISCUSSIONS

5. 1. Experimental Setup

In this section, AMEJ was compared with three multie-objective optimazation algorithms, including multi-objective Jaya algorithm (MOJaya), non-dominated sorting teaching-learning based optimization algorithm (NSTLBO), and the fast and elitist multi-objective genetic algorithm (NSGA2), based on various comparison criteria.

All the experiments are programed by Python 3.8 programing language on a 64-bit windows 10 OS with an Intel(R) Core i7 2.80 GHz processor and 16-GB RAM system. Source papers for the implementation of the comparison algorithms with explanations of their parameters are listed in Table 1.

The initial population number and the number of iterations of each algorithm were considered 300 and 20 times, respectively. In the rescheduling problem for cloud manufacturing, the constraints imposed on each of the objective functions can be considered as the min and max allowed values. In the following experiments, we considered values 75 and 100 as the min and max allowed values for all four objective functions.

These constraints can be determined by the expert depending on the type of problem. The initial population was also randomly generated in the allowed space of the problem and according to the imposed constraints.

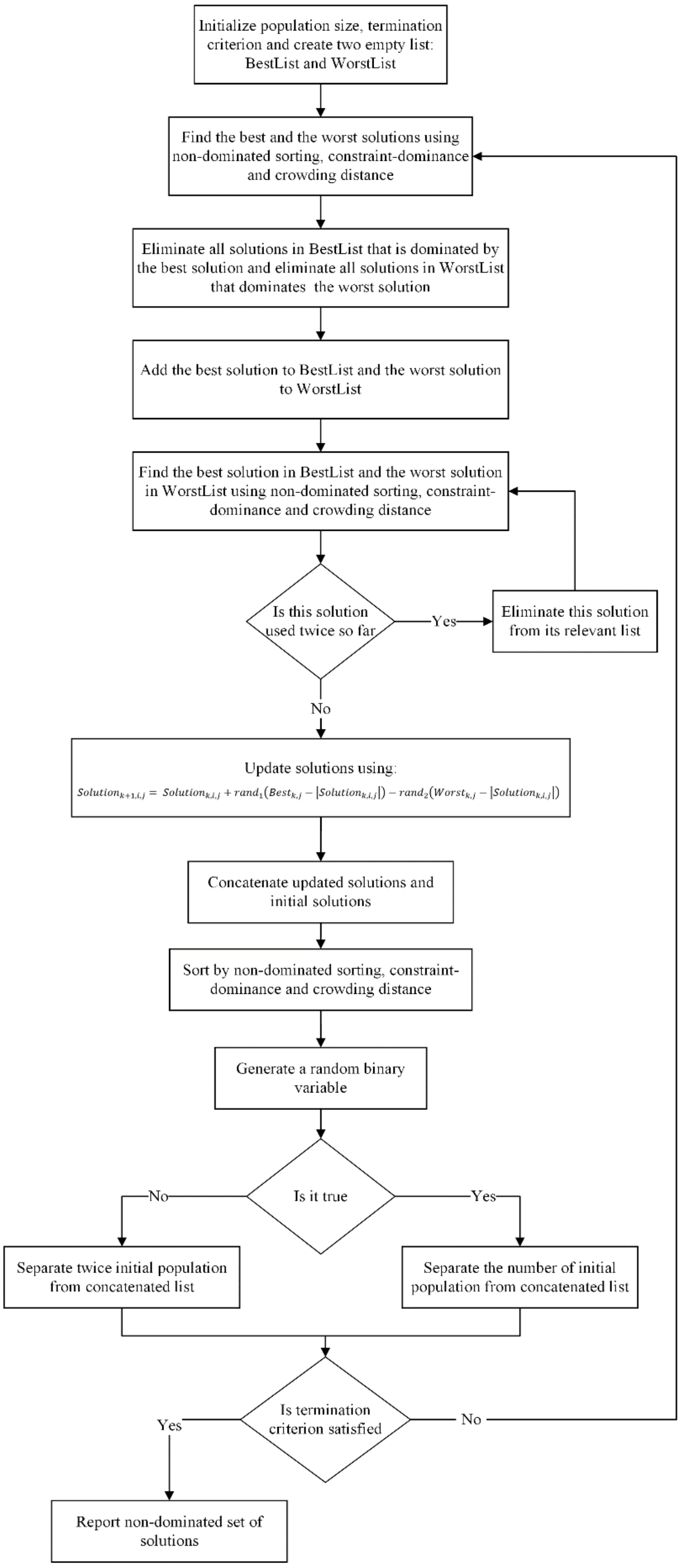
Initialize population size, termination criterion and create two empty list: BestList and WorstList
Find the best and the worst solutions using non-dominated sorting, constraint-dominance and crowding distance
Eliminate all solutions in BestList that is dominated by the best solution and eliminate all solutions in WorstList that dominates the worst solution
Add the best solution to BestList and the worst solution to WorstList
Find the best solution in BestList and the worst solution in WorstList using non-dominated sorting, constraint-dominance and crowding distance
Is this solution used twice so far
Yes
Eliminate this solution from its relevant list
No
Update solutions using:
$Solution_{k+1,i,j} = Solution_{k,i,j} + rand_1(Best_{k,j} - |Solution_{k,i,j}|) - rand_2(Worst_{k,j} - |Solution_{k,i,j}|)$
Concatenate updated solutions and initial solutions
Sort by non-dominated sorting, constraint-dominance and crowding distance
Generate a random binary variable
Is it true
No
Yes
Separate twice initial population from concatenated list
Separate the number of initial population from concatenated list
Is termination criterion satisfied
Yes
No
Report non-dominated set of solutions

Figure 4. Flowchart of the AMEJ algorithm

TABLE 1. Source articles and parameters of comparison algorithms

Algorithm Name	Source article	Parameters
MOJaya	Rao et al. [40]	Parameter-less
NSTLBO	Rao et al. [23]	Parameter-less
NSGA2	Deb et al. [41]	crossover probability = 0.9 mutation probability = 1/n n is the number of decision variables

5. 2. Experimental Results

The first experiment is to compare algorithms based on the number of non-dominated solutions found in 300 individuals of population. Figure 5 shows the result of this comparison.

As shown in Figure 5, the number of non-dominated solutions found by AMEJ is more than other algorithms. This comparison criteria is simple but it can show the better performance of an algorithm in finding non-dominated solutions in the problem space compared to others.

The spacing measure is another comparison criterion used to compare multi-objective algorithms. This criterion shows how the solutions uniformly distributed in a Pareto front. The spacing measure can be calculated by the following equation:

$$Spacing = \sqrt{\frac{1}{|n-1|}\sum_{i=1}^{n}(\bar{d}-d_i)^2} \qquad (11)$$

where n is the non-dominated solutions count and d_i calculated by Equation (12):

$$d_i = \min_{i,i\neq j}\sum_{m=1}^{k}\left|f_m^i - f_m^j\right|, \quad i,j = 1,2,\dots,n \qquad (12)$$

where k is the count of objective functions and f_m is the value of the objective function for mth objective. $\bar{d}$ is the normalized value of d_i and calculated by Equation (13):

$$\bar{d} = \sum_{i=1}^{n} d_i/|n| \qquad (13)$$

The lower the value obtained for spacing measure by an

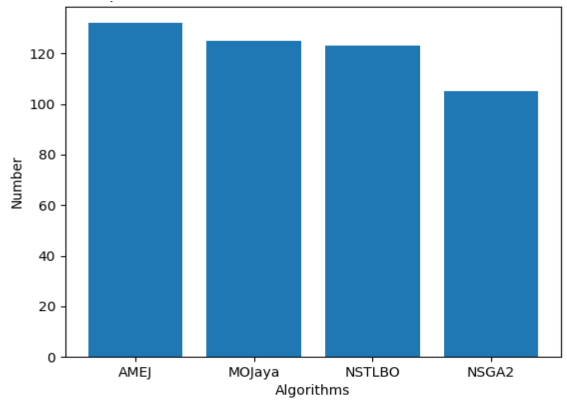

Figure 5. Number of non-dominated solutions found by each algorithm

algorithm, the better its performance based on this criterion. Figure 6 shows the comparison results of algorithms based on spacing measure:

Individual displacement in the problem space can indicate the performance of the algorithm in improving all solutions. Figures 7 shows the position of the individuals in the problem space at the beginning and after the completion of the AMEJ algorithm for each pair

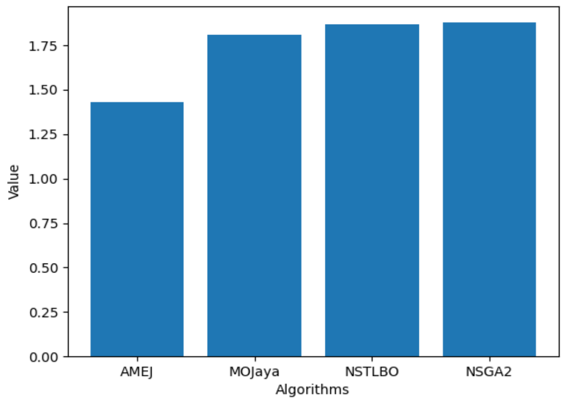

Figure 6. Comparison of algorithms based on the spacing measure

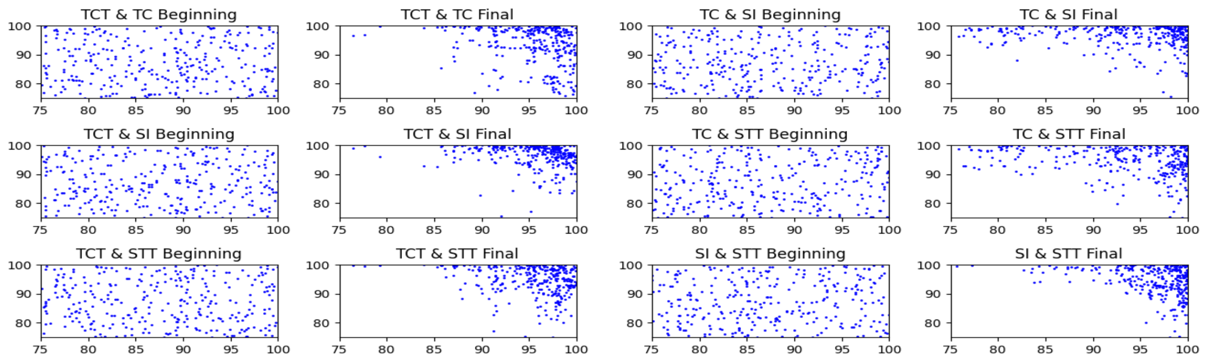

Figure 7. Distribution of individuals in the problem space at the beginning and end of AMEJ run

TABLE 2. Comparison of mean and standard division of non-dominated solutions of AMEJ and MOJaya

	TCT		TC	
	AMEJ	MOJaya	AMEJ	MOJaya
Mean	96.060076	95.186218	97.770860	97.545763
Standard Division	3.667294	4.980794	4.329043	4.597407
	SI		**STT**	
	AMEJ	MOJaya	AMEJ	MOJaya
Mean	96.773871	96.098727	95.059678	94.650871
Standard Division	3.954239	3.628339	4.570308	5.871844

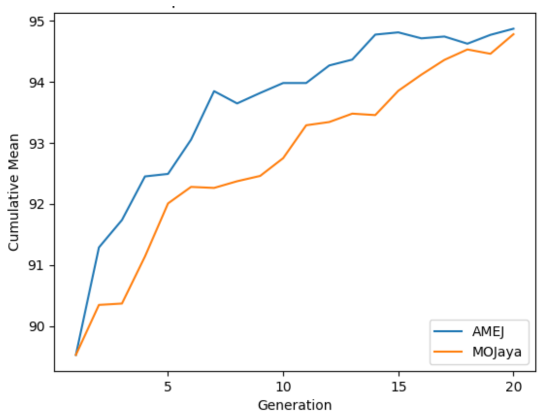

Figure 8. comparison of AMEJ and MOJaya based on cumulative mean in 20 iterations

of objective functions. This figure shows the random distribution of individuals in the problem space at the beginning and the move towards collective improvement at the end.

Since MOJaya is the base algorithm of AMEJ, the following two comparisons were performed between these algorithms. The results show that AMEJ performance is better than MOJaya.

Table 2 shows the mean and standard deviation of non-dominated solutions of each objective functions.

Cumulative mean is another criterion used to compare AMEJ and MOJaya algorithms. To calculate this criterion in each iteration, the values of the objective functions are first normalized. Then for each solution, the sum of all objective function values with an equal weight of one is calculated. Then the mean of all calculated values is reported. For this experiment, all the solutions in the population were used.

Figure 8 shows the result of the comparison of AMEJ and MOJaya based on the cumulative mean.

6. CONCLUSION AND FUTURES WORK

CMfg has a highly dynamic environment. In this environment many disruptions or events may occur that lead the system to unplanned situations. When any disruptions or other events occurred, the rescheduling techniques used to updating the inital schedule. First, this paper analyzes the dynamic rescheduling problem in CMfg. Then the rescheduling problem in CMfg is modeled mathematically. To solve the multi-objective rescheduling problem in CMfg, this paper proposed the AMEJ algorithm which is an advanced multi-objective and elitist algorithm based on Jaya.

To show the performance of the AMEJ algorithm, several experiments are performed. The results indicate that AMEJ has better performance than other multi-objective optimization algorithms such as MOJaya, NSTLBO, and NSGA2.

Considering the multi-objective nature of the rescheduling problem in CMfg, the procedure of modeling the rescheduling problem into a multi-objective optimization problem that was presented in this article can be considered as a suitable approach in solving rescheduling problems in future researches.

The proposed algorithm is also a generic algorithm that can be studied in other areas besides CMfg. Because the proposed algorithm can also be considered in single-objective optimization problems, we will implement a priori approach-based algorithm inspired by AMEJ in our future work and we will conduct more comparisons based on standard benchmark functions.

7. REFERENCES

1. Wu, D., Greer, M.J., Rosen, D.W. and Schaefer, D., "Cloud manufacturing: Drivers, current status, and future trends", in International Manufacturing Science and Engineering Conference, American Society of Mechanical Engineers. Vol. 55461, (2013), V002T002A003.
2. Tao, F., Zhao, D., Yefa, H. and Zhou, Z., "Correlation-aware resource service composition and optimal-selection in manufacturing grid", ***European Journal of Operational Research***, Vol. 201, No. 1, (2010), 129-143. https://doi.org/10.1016/j.ejor.2009.02.025
3. Li, B.-H., Zhang, L., Ren, L., Chai, X.-D., Tao, F., Wang, Y.-Z., Yin, C., Huang, P., Zhao, X.-P. and Zhou, Z.-D., "Typical characteristics, technologies and applications of cloud manufacturing", ***Computer Integrated Manufacturing System***, Vol. 18, No. 07, (2012).

4. Tao, F., Zuo, Y., Da Xu, L. and Zhang, L., "Iot-based intelligent perception and access of manufacturing resource toward cloud manufacturing", ***IEEE Transactions on Industrial Informatics***, Vol. 10, No. 2, (2014), 1547-1557. doi: 10.1109/TII.2014.2306397.

5. Vieira, G.E., Herrmann, J.W. and Lin, E., "Rescheduling manufacturing systems: A framework of strategies, policies, and methods", ***Journal of Scheduling***, Vol. 6, (2003), 39-62. https://doi.org/10.1023/A:1022235519958

6. Zhang, S., Xu, Y. and Zhang, W., "Multitask-oriented manufacturing service composition in an uncertain environment using a hyper-heuristic algorithm", ***Journal of Manufacturing Systems***, Vol. 60, (2021), 138-151. https://doi.org/10.1016/j.jmsy.2021.05.012

7. Fazeli, M.M., Farjami, Y. and Nickray, M., "An ensemble optimisation approach to service composition in cloud manufacturing", ***International Journal of Computer Integrated Manufacturing***, Vol. 32, No. 1, (2019), 83-91. https://doi.org/10.1080/0951192X.2018.1550679

8. Zhang, X. and Ren, D., "Modeling and simulation of task rescheduling strategy with resource substitution in cloud manufacturing", ***Mathematical Biosciences and Engineering***, Vol. 20, No. 2, (2023), 3120-3145. doi: 10.3934/mbe.2023147.

9. Zhang, X., Han, Y., Królczyk, G., Rydel, M., Stanislawski, R. and Li, Z., "Rescheduling of distributed manufacturing system with machine breakdowns", ***Electronics***, Vol. 11, No. 2, (2022), 249. https://doi.org/10.3390/electronics11020249

10. Liu, Y., Xu, X., Zhang, L., Wang, L. and Zhong, R.Y., "Workload-based multi-task scheduling in cloud manufacturing", ***Robotics and Computer-integrated Manufacturing***, Vol. 45, No., (2017), 3-20. https://doi.org/10.1016/j.rcim.2016.09.008

11. Yang, B., Wang, S., Cheng, Q. and Jin, T., "Scheduling of field service resources in cloud manufacturing based on multi-population competitive-cooperative gwo", ***Computers & Industrial Engineering***, Vol. 154, (2021), 107104. https://doi.org/10.1016/j.cie.2021.107104

12. Liu, Z., Guo, S., Wang, L., Du, B. and Pang, S., "A multi-objective service composition recommendation method for individualized customer: Hybrid mpa-gso-dnn model", ***Computers & Industrial Engineering***, Vol. 128, (2019), 122-134. https://doi.org/10.1016/j.cie.2018.12.042

13. Zhou, J. and Yao, X., "A hybrid artificial bee colony algorithm for optimal selection of qos-based cloud manufacturing service composition", ***The International Journal of Advanced Manufacturing Technology***, Vol. 88, (2017), 3371-3387. https://doi.org/10.1007/s00170-016-9034-1

14. Zhou, L., Zhang, L. and Ren, L., "Simulation model of dynamic service scheduling in cloud manufacturing", in IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society, IEEE. (2018), 4199-4204.

15. Zhou, L., Zhang, L., Sarker, B.R., Laili, Y. and Ren, L., "An event-triggered dynamic scheduling method for randomly arriving tasks in cloud manufacturing", ***International Journal of Computer Integrated Manufacturing***, Vol. 31, No. 3, (2018), 318-333. https://doi.org/10.1080/0951192X.2017.1413252

16. Serrano-Ruiz, J.C., Mula, J. and Poler, R., "Smart manufacturing scheduling: A literature review", ***Journal of Manufacturing Systems***, Vol. 61, (2021), 265-287. https://doi.org/10.1016/j.jmsy.2021.09.011

17. Yuan, M., Zhou, Z., Cai, X., Sun, C. and Gu, W., "Service composition model and method in cloud manufacturing", ***Robotics and Computer-integrated Manufacturing***, Vol. 61, (2020), 101840. https://doi.org/10.1016/j.rcim.2019.101840

18. Wang, T., Zhang, P., Liu, J. and Zhang, M., "Many-objective cloud manufacturing service selection and scheduling with an evolutionary algorithm based on adaptive environment selection strategy", ***Applied Soft Computing***, Vol. 112, (2021), 107737. https://doi.org/10.1016/j.asoc.2021.107737

19. Champati, J.P. and Liang, B., "Delay and cost optimization in computational offloading systems with unknown task processing times", ***IEEE Transactions on Cloud Computing***, Vol. 9, No. 4, (2019), 1422-1438. doi: 10.1109/TCC.2019.2924634.

20. Liu, Y., Liang, H., Xiao, Y., Zhang, H., Zhang, J., Zhang, L. and Wang, L., "Logistics-involved service composition in a dynamic cloud manufacturing environment: A ddpg-based approach", ***Robotics and Computer-integrated Manufacturing***, Vol. 76, (2022), 102323. https://doi.org/10.1016/j.rcim.2022.102323

21. Rao, R.V., "Jaya: An advanced optimization algorithm and its engineering applications", (2019).

22. Rao, R.V., Savsani, V.J. and Vakharia, D., "Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems", ***Computer-aided Design***, Vol. 43, No. 3, (2011), 303-315. https://doi.org/10.1016/j.cad.2010.12.015

23. Rao, R., "Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems", ***International Journal of Industrial Engineering Computations***, Vol. 7, No. 1, (2016), 19-34. doi: 10.5267/j.ijiec.2015.8.004.

24. Yang, C., Peng, T., Lan, S., Shen, W. and Wang, L., "Towards iot-enabled dynamic service optimal selection in multiple manufacturing clouds", ***Journal of Manufacturing Systems***, Vol. 56, No., (2020), 213-226. doi. https://doi.org/10.1016/j.jmsy.2020.06.004 23

25. Xie, N., Tan, W., Zheng, X., Zhao, L., Huang, L. and Sun, Y., "An efficient two-phase approach for reliable collaboration-aware service composition in cloud manufacturing", ***Journal of Industrial Information Integration***, Vol. 23, No., (2021), 100211. doi. https://doi.org/10.1016/j.jii.2021.10021124

26. Li, Y., Yao, X. and Liu, M., "Cloud manufacturing service composition optimization based on reliability and credibility analysis", ***Computer Integrated Manufacturing Systems***, Vol. 27, No. 6, (2021), 1-33. doi. https://doi.org/10.1155/2019/7194258

27. Chen, S., Fang,S.,Tang, R., , "An ann-based approach for real-time scheduling in cloud manufacturing", ***Applied Sciences***, Vol. 10 No. 7, (2020). doi: 10.3390/app10072491.

28. Zheng, X. and Zhang, X., "Robustness of cloud manufacturing system based on complex network and multi-agent simulation", ***Entropy***, Vol. 25, No. 1, (2022), 45. doi: 10.3390/e25010045.

29. Arkat, J., Rahimi, V. and Farughi, H., "Reactive scheduling addressing unexpected disturbance in cellular manufacturing systems", ***International Journal of Engineering, Transactions A: Basics,***, Vol. 34, No. 1, (2021), 162-170. doi: 10.5829/ije.2021.34.01a.18.

30. Puspitasari, K.M.D., Raharjo, J., Sastrosubroto, A.S. and Rahmat, B., "Generator scheduling optimization involving emission to determine emission reduction costs", ***International Journal of Engineering, Transactions B: Applications***, Vol. 35, No. 8, (2022), 1468-1478. doi: 10.5829/ije.2022.35.08b.02.

31. Nikaeen, R. and Najafi, A., "A constraint programming approach to solve multi-skill resource-constrained project scheduling problem with calendars", ***International Journal of Engineering, Transactions B: Applications***, Vol. 35, No. 8, (2022), 1579-1587. doi: 10.5829/ije.2022.35.08b.14.

32. Torkashvand, M., Ahmadizar, F. and Farughi, H., "Distributed production assembly scheduling with hybrid flowshop in assembly stage", ***International Journal of Engineering, Transactions B: Applications***, Vol. 35, No. 5, (2022), 1037-1055. doi: 10.5829/ije.2022.35.05b.19.

33. Maghzi, P., Mohammadi, M., Pasandideh, S. and Naderi, B., "Operating room scheduling optimization based on a fuzzy uncertainty approach and metaheuristic algorithms",

International Journal of Engineering, Transactions B: Applications, Vol. 35, No. 2, (2022), 258-275. doi: 10.5829/ije.2022.35.02b.01.

34. Halty, A., Sánchez, R., Vázquez, V., Viana, V., Pineyro, P. and Rossit, D.A., "Scheduling in cloud manufacturing systems: Recent systematic literature review", (2020). doi: 10.3934/mbe.2020377.
35. Rashidifar, R., Chen, F.F., Bouzary, H. and Shahin, M., A mathematical model for cloud-based scheduling using heavy traffic limit theorem in queuing process, in Flexible automation and intelligent manufacturing: The human-data-technology nexus: Proceedings of faim 2022, june 19–23, 2022, detroit, michigan, USA. 2022, Springer.197-206.
36. Abtahi, Z., Sahraeian, R. and Rahmani, D., "A stochastic model for prioritized outpatient scheduling in a radiology center", ***International Journal of Engineering Transactions A: Basics***, Vol. 33, No. 4, (2020). doi: 10.5829/ije.2020.33.04a.11.
37. Yazdi, M., Zandieh, M. and Haleh, H., "A mathematical model for scheduling elective surgeries for minimizing the waiting times in emergency surgeries", ***International Journal of Engineering, Transctions C: Aspects***, Vol. 33, No. 3, (2020), 448-458. doi: 10.5829/ije.2020.33.03c.09.
38. Simon, D., "Evolutionary optimization algorithms, John Wiley & Sons, (2013).
39. Gao, K., Yang, F., Zhou, M., Pan, Q. and Suganthan, P.N., "Flexible job-shop rescheduling for new job insertion by using discrete jaya algorithm", ***IEEE Transactions on Cybernetics***, Vol. 49, No. 5, (2018), 1944-1955. doi: 10.1109/TCYB.2018.2817240.
40. Rao, R.V., Rai, D.P. and Balic, J., "A multi-objective algorithm for optimization of modern machining processes", ***Engineering Applications of Artificial Intelligence***, Vol. 61, (2017), 103-125. https://doi.org/10.1016/j.engappai.2017.03.001
41. Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T., "A fast and elitist multiobjective genetic algorithm: Nsga-ii", ***IEEE Transactions on Evolutionary Computation***, Vol. 6, No. 2, (2002), 182-197. doi: 10.1109/4235.996017.

Persian Abstract

چکیده

CMfg یک مدل ساخت پیشرفته جدید است که با کمک فناوری‌های اطلاعات سازمانی تحت پشتیبانی رایانش ابری، اینترنت اشیا و فناوری‌های مبتنی بر خدمات توسعه یافته است. CMfg منابع تولیدی متعددی را برای ارائه خدمات کارآمد و ارزشمند تشکیل می دهد. CMfg محیطی بسیار پویا دارد. در این محیط، اختلالات یا اتفاقات زیادی ممکن است رخ دهد که سیستم را به موقعیت های برنامه ریزی نشده سوق دهد. در CMfg، یک سری از ارائه دهندگان خدمات برای تولید برنامه ریزی شده اند. در حین عملیات تولید ممکن است برخی از آنها آسیب ببینند، متوقف شوند و از سرویس خارج شوند. لذا برای ادامه فرآیند تولید طبق قراردادهای منعقده و زمانبندی اولیه، زمانبندی مجدد ضروری است. هنگامی که هر گونه اختلال یا رویداد دیگری رخ می دهد، از تکنیک های زمان بندی مجدد برای به روز رسانی برنامه اولیه استفاده می شود. در این مقاله، مسئله زمان‌بندی مجدد دینامیک در CMfg تحلیل می‌شود. سپس زمان بندی مجدد چند هدفه در CMfg مدل شده و به عنوان یک مسئله بهینه سازی چند هدفه تعریف می شود. تعریف این مسئله به عنوان یک مسئله بهینه سازی چندهدفه امکان اعمال، بررسی و مقایسه الگوریتم های مختلف را فراهم می کند. برای حل این مشکل، روش‌های بهینه‌سازی قبلی بهبود یافته و یک الگوریتم چندهدفه و نخبه‌گرا بر اساس الگوریتم جایا به نام الگوریتم پیشرفته چندهدفه نخبه‌گرا جایا (AMEJ)پیشنهاد شد. چندین آزمایش برای تأیید عملکرد الگوریتم پیشنهادی انجام شده است. نتایج محاسباتی نشان داد که الگوریتم پیشنهادی در مقایسه با سایر الگوریتم‌های بهینه‌سازی چندهدفه بهتر عمل می‌کند.