A three-tier programming model for service composition and optimal selection in cloud manufacturing

Abstract

The process of service composition and optimal selection in cloud manufacturing (CMfg-SCOS) involves three types of users: service demanders, resource providers, and cloud platform operators. The interests of all users are a research focus of CMfg-SCOS, as their participation in the CMfg system directly affects the efficiency and long-term development of CMfg. However, the current research on CMfg-SCOS rarely considers the interests of all three types of users simultaneously, and the interest of resource providers is not clearly defined, which lags behind the reality of CMfg. Therefore, this study first proposes a three-tier programming model of CMfg-SCOS that considers the interests of service demanders, cloud platform operators, and resource providers. At the lower level of the model, service demanders are the decision makers, aiming to minimize time and cost and maximize service quality. At the middle level of the model, cloud platform operators are the decision makers, aiming to maximize resource use and flexibility in the face of uncertain environments. At the upper level, resource providers are the decision makers, aiming to maximize enterprise surplus. Then, this study develops an improved fast nondominated sorting genetic algorithm with advancement and inheritance (namely, a-i-NSGA-II) to solve the three-tier model efficiently. Numerical experiments conducted in this study found that in comparison to the art of state algorithms, including original nondominated sorting genetic algorithm II (NSGA-II), multiobjective particle swarm optimization (MOPSO), and multiobjective spotted hyena optimizer (MOSHO), the proposed a-i-NSGA-II has better diversity and comprehensive performance at the middle level of the model and better solution quality at the upper level. Furthermore, a case study of the actual production of an automobile fuel tank assembly enterprise verifies the effectiveness of the proposed model and algorithm.

Keywords: cloud manufacturing; service composition and optimal selection; three-tier programming model; a-i-NSGA-II

Nomenclature

B	Enterprise sales output
C	Enterprise production input
C_1	Direct inputs
C_{11}	Raw materials, fuel and power costs for services
C_{12}	Basic depreciation costs and maintenance costs of machinery and equipment
C_{13}	Wages of production workers and other additional charges
C_2	Indirect inputs
C_{21}	Costs caused by the occupation of fixed or liquid funds of resource providers
C_{22}	Social consumption costs caused by natural resource or material shortages
C_{ma}	Service cost for each resource (service cost)
$C_{\max}$	The highest cost required by service demanders
C_{to}	Total service cost
C_{tr}	Transportation cost between adjacent manufacturing resources (logistics cost)
$CS_i^j(k)$	The $k-th$ candidate service of subtask ST_i^j
CSS_i	Candidate service set of task T_i
CSS_i^j	Candidate services of subtask ST_i^j
D	User demand
E	Enterprise economic benefit
F	Flexibility
F_{co}	Number of cooperative enterprises
$F_{co\min}$	The minimum number of cooperative enterprises required by cloud platform operators
F_E	Evaluation of service
$F_{E\min}$	The minimum service evaluation required by cloud platform operators
F_{fu}	Service function diversity of each manufacturing resource
$F_{fu\min}$	The minimum functional diversity required by cloud platform operators
F_{sa}	Number of manufacturing resources with the same function
$F_{sa\min}$	The minimum number of resources required by cloud platform operators
F_R	Respond ability for changes of manufacturing resources
F_{re}	Reliability of each manufacturing resource
$F_{re\min}$	The minimum reliability required by cloud platform operators
F_T	Respond ability for changes of manufacturing tasks
F_{ty}	Resource type owned by resource providers
$F_{ty\min}$	The minimum number of types of manufacturing resources required by cloud platform operators
I	Number of tasks
J	Number of subtasks
$K_{i,j}$	Number of CSS_i^j 's sub-services
L_d	Demand load of service demanders
L_{p_j}	Remaining service load of resource providers
MRU	Resource utilization

$P_{i\max}$	The highest service price that can be sustained by service demanders
Q	Total service quality
$Q_{\min}$	The minimum service quality required by service demanders
Q_{se}	Quality qualification rate of the manufacturing resource or service
ST	Subtask
ST_i^j	The $j-th$ subtask of the $i-th$ task
T	Task
T_i	The $i-th$ task of demand
T_{ma}	Time for processing resources (running time)
$T_{\max}$	The longest delivery time required by service demanders
T_{tr}	Transportation time between adjacent manufacturing resources (logistics time)
T_{to}	Total service time
T_{wa}	Waiting time for these occupied services when the task has arrived (response time)

1 Introduction

The modern manufacturing industry is undergoing a paradigm shift towards global manufacturing networks and supply chains (Adamson et al., 2017). Cloud manufacturing (CMfg), as a new networked manufacturing mode, promotes the agile, service-oriented, green, and intelligent development of the manufacturing industry (Lim et al., 2020), supports and facilitates this change. The goal of CMfg is to provide resource sharing and an on-demand manufacturing mode to improve operational efficiency and the use of manufacturing resources. The realization of this ambitious vision includes a series of necessary underlying technologies, such as resource virtualization (Liu et al., 2014), encapsulation (Zhang, Zhang, Liu, & Hu, 2017), interoperability (Mourad et al., 2020), task scheduling (Dong et al., 2020), service composition and optimal selection (SCOS) (Wang et al., 2020; Zhou & Yao, 2017), resource allocation (Lin & Chong, 2017; Yang et al., 2021), and security strategy (Esposito et al., 2016). Among them, SCOS is the key to completing the manufacturing task and meeting user requirements (Tao et al., 2011). Furthermore, the quality of its optimization model and solution mechanism will directly affect the quality of manufacturing service and whether the service process can be performed safely and smoothly (Huang et al., 2018). Therefore, research on SCOS is of great significance to the implementation and development of CMfg.

The core users in the process of CMfg-SCOS include three types: service demanders, cloud platform operators, and resource providers, each category of stakeholder is an autonomous decision-making and interest-independent entity (Wang et al., 2021). The benefit of different users is essential factor that scholars must consider in the study of CMfg-SCOS (Zhang et al., 2021). Table 1 shows the typical work in CMfg that considers the interests of different users. Service demanders are the initiators of manufacturing tasks, and their interests should be considered first in CMfg-SCOS. There are extensive researches on the perspective of service demanders. For example, Cao et al. (2016) established a service selection and scheduling model based on fuzzy decision theory. The proposed model could meet the TQCS (time, quality, cost, and service) requirements of service demanders. Zheng et al. (2017) proposed a hybrid energy-aware resource allocation method to help service demanders obtain energy-efficient and satisfactory manufacturing services. Liu et al. (2017) presented a CMfg multi-task scheduling model that incorporates task workload modelling and considers

completion time, task cost and task reliability of service demanders. He et al. (2017) considered the expectations of different service demanders, proposed an integer nonlinear programming model based on prospect theory, and verified that this model could provide service demanders with high-quality and low-cost services. Chen et al. (2016) think that it is necessary to regard the quality of service (QoS) performance and the minimum variance of expected QoS as two objective functions. It provides flexibility for service demanders to choose alternative schemes when no optimal service combination is available to meet the expected QoS. Cloud platform operators are the coordinators to ensure the smooth operation of the entire CMfg process, and the interests of cloud platform operators should also be considered in CMfg-SCOS. Some scholars have considered the needs of service demanders and cloud platform operators in their research on CMfg-SCOS. For example, Wu et al. (2020) propose a multiobjective model (MOIBMFP). It maximizes the manufacturing sustainability requirements of cloud operators and the QoS requirements of multiservice demanders simultaneously. Results show that the proposed model and algorithm obtain a better solution to meet the needs of cloud platform operators and service demanders in the CMfg environment. Su et al. (2015) proposed an evaluation index system for CMfg-SCOS, including the service quality index and flexibility index. They established a bilevel programming model of manufacturing resource allocation. The experimental results showed that the model and algorithm could satisfy the interests of the manufacturing task demander and the CMfg service platform operator. Resource providers are the owners of CMfg resources, and their interests should also be considered in CMfg-SCOS because it helps service providers decide whether to join the CMfg system again. To maintain the decision autonomy of resource providers and achieve the optimal allocation of CMfg services, a decentralized decision mechanism named analytical target cascading was introduced (Zhang, Chen, et al., 2020; Zhang et al., 2018; Zhang, Zhang, Qu, et al., 2017). Some scholars have considered the needs of service demanders and resource providers to research CMfg-SCOS. For example, to solve the core manufacturing SCOS problem in the manufacturers-to-users (M2U) CMfg model, Que et al. (2018) proposed a new adaptive information entropy immune genetic algorithm (IEIGA), achieving high convergence speed, good stability, high quality, and high scalability. Thekinen and Panchal (2017) considered the different preferences of service demanders and resource providers and designed four types of two-way matching mechanisms to find the best match. Cheng et al. (2018) proposed a

bilevel planning model for manufacturing resource optimization in consideration of the interests of customers and enterprises, which realized the optimal allocation of collaborative manufacturing resources in the 3D printing cloud service platform. Zhang et al. (2021) proposed a new utility-aware CMfg multitask scheduling model, which considers the utilities of customers and manufacturers. An extended nondominated sorting genetic algorithm-II and game theory were used to obtain the optimal solution and then recommended for the CMfg system.

Table 1. Typical work on resource optimal allocation in CMfg

Users considered	References	Models or Methods
Service demanders	(Cao et al., 2016)	A service selection and scheduling model based on fuzzy decision theory.
	(Zheng et al., 2017)	A hybrid energy-aware resource allocation method.
	(He et al., 2017)	An integer non-linear programming model based on prospect theory.
	(Chen et al., 2016)	The method of taking the minimum variance of QoS performance and expected QoS as two objective functions.
	(Liu et al., 2017)	A CMfg multi-task scheduling model that incorporates task workload modelling and a number of other essential ingredients regarding services.
Resource providers	(Zhang, Chen, et al., 2020)	The distributed optimization model based on analytical target cascading method.
	(Zhang, Zhang, Qu, et al., 2017)	An analytical target cascading model and OR element.
	(Zhang et al., 2018)	An augmented Lagrangian coordination (ALC) model.
Service demanders & Cloud platform operators	(Wu et al., 2020)	A multi-objective integer bi-level more followers (MOIBMFP) model.
	(Su et al., 2015)	An SCOS evaluation index system with service quality and flexibility indexes and a bi-level programming model of manufacturing resource allocation.
Service demanders & Resource providers	(Que et al., 2018)	A new manufacturers-to-users (M2U) model for CMfg.
	(Cheng et al., 2018)	A bi-level planning model of manufacturing resources optimal allocation for 3D printing cloud service platform.
	(Zhang et al., 2021)	A new utility-aware CMfg multi-task scheduling model.
	(Thekinen & Panchal, 2017)	Four types of two-way matching mechanisms.

Existing research has achieved fruitful results, but limitations remain. 1) Current SCOS methods are inappropriate because they only optimize the goals of one or two types of participants. To the best of the authors' knowledge, no research addresses the CMfg-SCOS problem from the overall perspective of service demanders, cloud platform operators, and resource providers, which cannot fully balance the benefit

distribution of all users and lags behind the actual conditions and realities of CMfg (Liu et al., 2019). 2) The evaluation indicators of cloud platform operators are not sufficiently comprehensive. At present, research on the SCOS process from the perspective of platform operators is rare (Su et al., 2015). Existing research only considers unilateral evaluation indicators, such as resource utilization or resilience ability, in the face of uncertainty but does not comprehensively consider the two aspects of evaluation indicators. 3) At present, the description of the benefits of resource providers is limited to vague qualitative theories (such as satisfaction). While, there is no clear objective function to express the benefits of resource providers, and some studies even ignore the interests of resource providers (Bouzary & Frank Chen, 2018). This situation may cause a decline in the interest and satisfaction of resource providers in the CMfg system; in the worst case, they may even drop out of the system.

Therefore, to address the above limitations, this study proposes an SCOS model based on three-tier model programming that considers the interests of all users in CMfg. Each level has different decision makers and objectives. A new algorithm is designed to solve the model and numerical experiments are designed to verify it. The contributions of this study are as follows:

1) A three-tier programming model of CMfg-SCOS is proposed to satisfy the interests of service demanders, resource providers, and cloud platform operators at the same time.

2) An improved, fast and elitist multiobjective genetic algorithm with advancement and inheritance (a-i-NSGA-II) is proposed to solve the three-tier model efficiently. The advance and inheritance mechanism enhance the correlation between three levels and can get a better and more reliable result when solving the model.

3) A case study of an automobile fuel tank assembly enterprise is conducted from the actual production. The positive case results show that this study provides an important method for balancing the interests of all users in solving the problem of CMfg-SCOS.

The remainder of this paper is organized as follows: First, Section 2 describes the SCOS problem in CMfg. Second, Section 3 proposes a three-tier programming SCOS model that simultaneously considers the interests of all users in CMfg. This is followed by the elaborations and numerical analysis of the improved algorithm (a-i-NSGA-II) to address the three-tier programming model in Section 4. A case study of the proposed model and algorithm is further assessed in Section 5. Finally, Section 6 concludes the

paper and discusses future research directions.

2 SCOS problem in CMfg

The CMfg system follows a tripartite operation mode, which is mainly composed of three types of users: service demanders, resource providers, and cloud platform operators (Li et al., 2010). Resource providers provide various manufacturing resources to the CMfg service platform. Service demanders submit different manufacturing demands to the cloud platform. Cloud platform operators allocate reasonable services for service demanders to complete the manufacturing tasks. Figure 1 shows the process of CMfg service composition. The procedure related to CMfg-SCOS includes the following steps (Zhang, Zhang et al., 2020):

1) Demand description: After receiving user demands, cloud platform operators describe the demand as a series of tasks encapsulated in a process. Describe the demands D as a series of tasks T, $D=\{T_1,T_2,\ldots,T_i,\ldots,T_I\}$, where T_i represents the $i-th$ task, and I is the number of tasks;
2) Task decomposition: Each task T_i can be broken down into a series of subtasks ST, where $T_i=\{ST_i^1,ST_i^2,\ldots,ST_i^j,\ldots,ST_i^J\}$, ST_i^j represents the $j-th$ subtask of the $i-th$ task, and J is the number of subtasks;
3) Service discovery: Manufacturing services that meet task requirements are aggregated to form a set of candidate services. Task T_i corresponds to a set of candidate services CSS_i, $CSS_i=\{CSS_i^1,CSS_i^2,...,CSS_i^j,...,CSS_i^J\}$. Subtask ST_i^j decomposed by task T_i also corresponds to a sub-candidate service set CSS_i^j, CSS_i^j contains a series of sub-candidate services CS, $CSS_i^j=\{CS_i^j(1),CS_i^j(2),...,CS_i^j(K),...,CS_i^j(K_{i,j})\}$, CSS_i^j represents the $j-th$ candidate service corresponding to the $i-th$ task, $CS_i^j(K)$ represents the service corresponding to the $j-th$ subtask, and $K_{i,j}$ is the number of CSS_i^j 's sub-services;
4) Service combination: Task T_i selects service CSS_i^j that meets the requirements of the task from the set of candidate services, and subtask ST_i^j selects service $CS_i^j(k)$ that meets the requirements of the subtask from the selected subcategory

service set CSS_i^j. Finally, the service chain (combined path) that meets the overall requirements is selected, and the selected service chain can be expressed as $CSS_i = \left\{CS_i^1(3), CS_i^2(2), CS_i^3(4), ..., CS_i^j(k), ..., CS_i^J\left(k^{'}\right)\right\}$;

5) Combination optimization: Assuming that subtask ST_i^j has m_j candidate services, in theory, there may be $\prod_{j=1}^{J} m_j$ service chains that meet the requirements to complete task T_i. It is an NP-hard (non-deterministic polynomial, NP) problem to choose the best one among all possible service chains while considering the interests of all users.

CMfg-SCOS is a process of selecting the best service chain (Yu et al., 2018). In this process, the selected services must be continuously adjusted according to the requirements of all users, considering different indicators and constraints. The ultimate goal is to ensure that the selected service chain satisfies all task requirements while maximizing the benefits of service demanders, resource providers, and cloud platform operators. Specific evaluation indicators and constraints are given in Section 3.

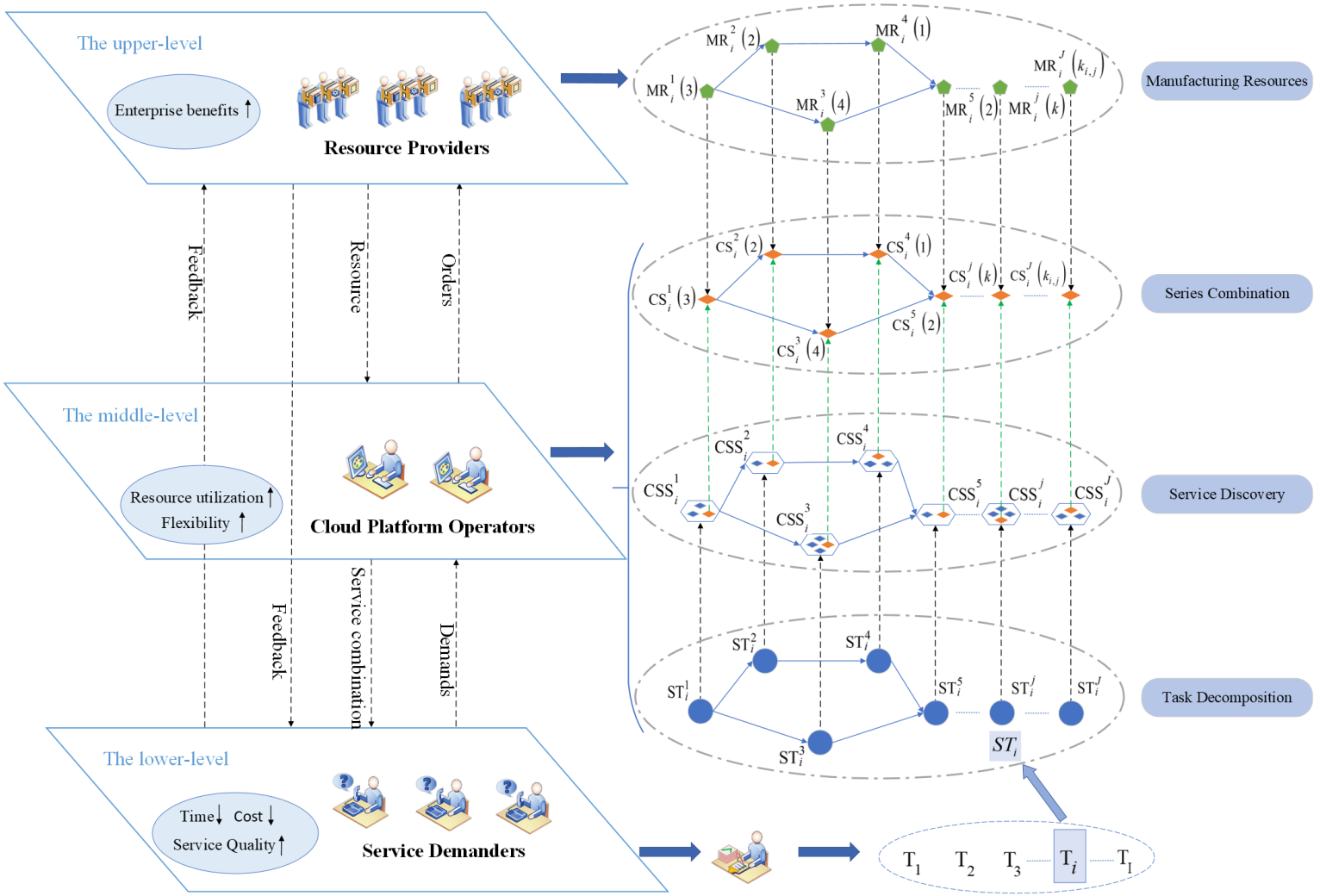


Figure 1. Process of CMfg service composition

3 Three-tier model of CMfg-SCOS based on the interests of all users

In CMfg, each type of user has its own goals and preferences, and the interests of different individuals sometimes conflict with each other. One challenge is to ensure the continuous participation of all users (Liu et al., 2019; Wang et al., 2019). In this case, an important issue is how to balance their interests and achieve their goals. Therefore, this section first proposes a comprehensive evaluation system that includes evaluation indicators of each type of user and then proposes a three-tier programming model that considers the interests of all participants. Table 2 shows comprehensive evaluation indicators of the three-tier CMfg-SCOS model.

3.1 Comprehensive evaluation system of the three-tier CMfg-SCOS model

3.1.1 Evaluation indicators for service demanders

Throughout the CMfg-SCOS process, service demanders, as initiators of the task and consumers of resource services, pay more attention to the completion of the task. That is, it puts forward requirements for the QoS indicators of the CMfg service combination. Therefore, service demanders put forward requirements for the QoS indicators of the CMfg-SCOS. Regarding the interests of service demanders, the QoS indicators of CMfg-SCOS mainly contain three aspects: service time T_{to}, service cost C_{to}, and service quality Q.

(1) Service time indicator T_{to}

The service time of CMfg-SCOS mainly includes three aspects: the time for processing resources (running time T_{ma}), the transportation time between adjacent manufacturing resources (logistics time T_{tr}) and the waiting time for these occupied services when the task has arrived (response time T_{wa}).

(2) Service cost indicator C_{to}

The service cost of CMfg-SCOS mainly includes two aspects: the service cost for each resource (service cost C_{ma}) and the cost of transportation between adjacent manufacturing resources (logistics cost C_{tr}). C_{ma} presents certain gradient changes with different task batches.

(3) Service quality indicator Q

The service quality of CMfg-SCOS is expressed by the service quality Q_{se} of each

manufacturing resource, that is, the quality qualification rate of the manufacturing resource or service when undertaking related tasks.

3.1.2 Evaluation indicators for cloud platform operators

Cloud platform operators run and manage the entire CMfg system. First, it is necessary to ensure the smooth completion of the task and avoid various risks caused by uncertain factors during the operation process. In other words, it puts forward requirements for the flexibility of CMfg-SCOS. In addition, it is essential to ensure the maximum benefit of the cloud platform, that is, to ensure the maximum utilization of resources.

(1) Flexibility indicator F

The flexibility indicator of CMfg-SCOS mainly has three aspects:

1) Ability to respond to changes in manufacturing tasks F_T

F_T represents the ability of the resource provider to complete the manufacturing task even when that task changes, including the service function diversity of each manufacturing resource F_{fu}, the resource type owned by resource providers F_{ty}, and the number of cooperative enterprises F_{co}.

2) Ability to respond to changes in manufacturing resources F_R

F_R represents the ability of the resource provider to complete the manufacturing task when the manufacturing resource changes (such as the offline manufacturing resources for some unpredictable reason), including the reliability of each manufacturing resource F_{re}, the number of manufacturing resources with the same function as the resource provider F_{sa}, and the number of cooperative enterprises F_{co}.

3) Evaluation of service F_E

The service evaluation of CMfg-SCOS is represented by the historical service evaluation of each manufacturing resource. The historical evaluation of manufacturing resources refers to the satisfaction evaluation (0~1) given by service demanders in the process of providing services, which represents the service attitude, service ability and service level of manufacturing services, as well as the cooperation situation and the attitude of handling problems in the process of completing manufacturing tasks. The higher the F_E is, the stronger the ability of manufacturing services to cope with changes in manufacturing tasks and manufacturing resources.

(2) Resource utilization indicator MRU

The utilization rate of manufacturing resources refers to the degree of load occupancy of resources in the process of service composition. During the SCOS process, the residual working capacity of resource providers (residual load) is considered, and resources whose residual load is near the demand load are selected as candidate resources to improve the use of the platform resources and ensure the working efficiency of CMfg.

3.1.3 Evaluation indicators for resource providers

Ensuring the interests of service providers promotes their willingness to join the CMfg system and prompts them to continuously improve their service quality and competitiveness to obtain more manufacturing tasks and profits (Meng & Xu, 2018). Enterprise surplus is an important factor considered by resource providers (Liu et al., 2019), and the economic benefit of the enterprise is used in this study to measure the enterprise surplus of resource providers.

(1) Enterprise economic benefit indicator E

In economics, economic benefit is used to reflect the comparative relationship between input and output (Li, 2015). In this study, enterprise economic benefit is used to measure the amount that the resource provider's output exceeds the CMfg service's input, in other words, the remaining profits of resource providers after compensating for resources and inputs during CMfg. Enterprise economic benefit is the difference between production input C and sales output B.

The production input of the resource provider is divided into direct inputs C_1 and indirect inputs C_2. Direct inputs include raw materials, fuel and power costs for CMfg services C_{11}, basic depreciation costs and maintenance costs of machinery and equipment C_{12} and the wages of production workers and other additional charges (such as resource management fees of the cloud platform) C_{13}; indirect inputs include costs caused by the occupation of fixed or liquid funds of resource providers in the process of CMfg C_{21} and social consumption costs caused by natural resource or material shortages C_{22}. The sales output of the resource provider is mainly the sales price of customized products.

Table 2. Comprehensive evaluation indicators of the three-tier CMfg-SCOS model

	First-level Indicators	Second-level Indicators	Third-level Indicators
Service demanders	Service time T_{to}	Running time T_{ma}	—
		Logistics time T_{tr}	—
		Response time T_{wa}	—
	Service cost C_{to}	Service time C_{ma}	—
		Logistics cost C_{tr}	—
	Service quality Q	Service quality Q_{se}	—
Cloud platform operators	Flexibility F	Respond ability for changes of manufacturing tasks F_T	Service function diversity of each manufacturing resource F_{fu}
			Resource type owned by resource providers F_{ty}
			Number of cooperative enterprises F_{co}
		Respond ability for changes of manufacturing resources F_R	Reliability of each manufacturing resource F_{re}
			Number of manufacturing resources with the same function F_{sa}
			Number of cooperative enterprises F_{co}
		Evaluation of Service F_E	—
	Resource utilization MRU	—	—
Resource providers	Enterprise economic benefit E	Production input C	Direct inputs C_1 (C_{11} C_{12} C_{13})
			Indirect inputs C_2 (C_{21} C_{22})
		Sales output B	—

3.2 Three-tier programming model of CMfg-SCOS

The CMfg-SCOS process based on all users' interests needs to maximize the satisfaction of service demanders, cloud platform operators, and resource providers at the same time. The three-tier programming model (Li et al., 2019) is a hierarchical model with three independent decision makers, and each decision maker optimizes its goals. The decision-making process is influenced by other decision makers, and vice versa, the decision-making results also affect other decision-making processes. The influence relationship among the evaluation indicators of users is shown in Figure 2.

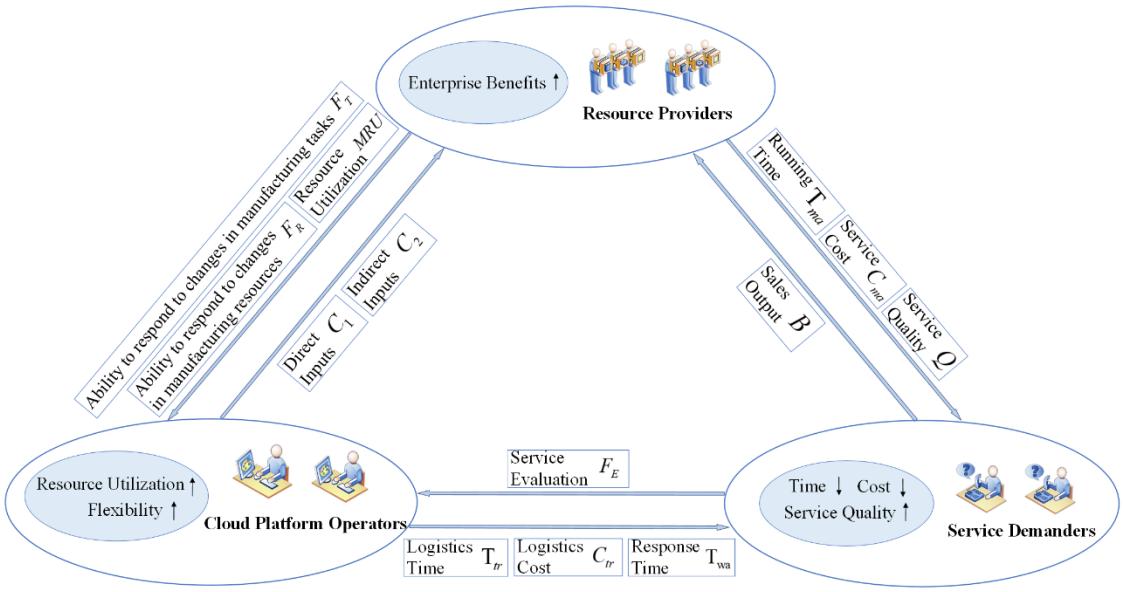


Figure 2. Influence relationship among the evaluation indicators of users

1) For service demanders, the resource attributes of resource providers affect running time T_{ma}, service cost C_{ma}, and service quality Q of service demanders. There is an equal relationship between service cost C_{ma} and sales output B of resource providers. Service demander hopes that service cost C_{ma} is as low as possible, while resource providers hope that sales output B is as high as possible, which need a balance in the service composition process. In addition, logistics time T_{tr}, logistics cost C_{tr}, and response time T_{wa} of resource providers are affected by the way cloud platform operators select resources. Service demanders tend to choose resources with

lower costs and time, but resources with higher costs and time also need to be selected because cloud platform operators consider the flexibility and resource utilization of the cloud platform.

2) For cloud platform operators, decisions of resource providers affect the flexibility of the cloud platform. For example, according to the analysis in section 3.1.2, resource diversity F_{fu}, resource types F_{ty}, and the number of cooperative enterprises F_{co} of resource providers closely related to the ability to respond to changes in manufacturing tasks F_T. Resource reliability F_{re}, the number of manufacturing resources with the same function F_{sa}, and the number of cooperative enterprises F_{co} closely related to the ability to respond to changes in manufacturing resources F_R. In addition, service evaluation F_E of service demanders directly affects the flexibility considered by cloud platform operators.

3) For resource providers, decisions of resource demanders affect sales output B, because resource providers and resource demanders negotiate the price of services in the process of CMfg transactions. In addition, decisions of cloud platform operators influence direct inputs C_1 and indirect inputs C_2 of resource providers. For example, resource management fees of cloud platform operators related to direct inputs C_1 of resource providers.

Therefore, the evaluation indicators of the three types of users influence each other and the combination of these indicators can effectively represent the interests of all users. In this study, service demanders are placed as lower-level decision makers because the ultimate goals of CMfg are to complete tasks and meet customer needs. Cloud platform operators are placed as middle-level decision makers because they are the managers of the CMfg system, and they ensure the smooth running of the CMfg process. Resource providers are placed as upper-level decision makers because the only appeal for resource providers is to ensure that profits can be obtained.

3.2.1 Basic assumptions

In the actual production and operation process, the CMfg mode faces different customer needs and decentralized resources, which is difficult to describe with a unified mathematical model. Therefore, this study makes the following assumptions based on the SCOS targets:

1) The cloud platform has split complex tasks into multiple serial subtasks in accord with the production process standards before SCOS;
2) Each subtask is independent of other subtasks;
3) Each subtask corresponds to a set of candidate services in advance;
4) Only cross-regional resource allocation is considered, and internal resource allocation is neglected;
5) Logistics costs are paid by service demanders, and resource providers only provide resources and production locally.

3.2.2 The determination of objective functions and constraints

CMfg-SCOS has four structures: sequence, parallel, selective, and circular (Bouzary & Chen, 2020). Sequential structure is the most basic and important structure of service composition because the other three structures can be simplified and equated to sequential structure in CMfg-SCOS (Tao et al., 2008). Therefore, this study takes sequential structure as the object of CMfg-SCOS.

(1) The lower-level optimization problem

As the lower-level decision makers of the three-tier programming model, service demanders use QoS indicators as the target from their interests. The objective functions are as follows:

1) Minimize total time

$$\min T_{to} = \min\left(T_{ma} + T_{tr} + T_{wa}\right) = \min\left[\sum_{i=1}^{n} T_{ma}(i) + \sum_{i=1}^{n} T_{tr}(i, i+1) + \sum_{i=1}^{n} T_{wa}(i)\right]$$

Where n is the total number of subtasks.

2) Minimize total cost

$$\min C_{to} = \min\left(C_{ma} + C_{tr}\right) = \min\left[\sum_{i=1}^{n} C_{ma}(i) + \sum_{i=1}^{n} C_{tr}(i, i+1)\right]$$

3) Optimum total quality

$$\max Q = \min\left(1 - Q_{se}\right) = \min\left[1 - \frac{\sum_{i=1}^{n} Q_{se}(i)}{n}\right]$$

The constraints of the lower-level programming model are as follows:

1) Time constraint

The total time T_{to} to complete the task in CMfg-SCOS cannot be greater than the longest delivery time $T_{\max}$ required by service demanders.

$$T_{to} \le T_{\max}$$

2) Cost constraint

The total cost C_{to} to complete the task in CMfg-SCOS cannot be greater than the highest cost $C_{\max}$ required by service demanders.

$$C_{to} \le C_{\max}$$

3) Quality constraint

The service quality of any manufacturing resource $Q_{se}(i)$ in CMfg-SCOS cannot be less than the minimum service quality $Q_{\min}$ required by service demanders.

$$Q_{se}(i) \ge Q_{\min}, i = 1, 2, ..., n$$

(2) The middle-level optimization problem

As the middle-level decision makers of the three-tier programming model, cloud platform operators take maximum flexibility and resource use as their goals. The objective functions are as follows:

1) Maximum flexibility

Ability to respond to changes in manufacturing tasks:

$$\max F_T = \max\left(\omega_{fu} F_{fu} + \omega_{ty} F_{ty} + \omega_{Tc} F_{co}\right) = \max\left[\omega_{fu} \frac{\sum_{i=1}^{n} F_{fu}(i)}{n} + \omega_{ty} \frac{\sum_{i=1}^{n} F_{ty}(i)}{n} + \omega_{Tc} \frac{\sum_{i=1}^{n} F_{co}(i)}{n}\right]$$

where ω_{fu}, ω_{ty}, and ω_{Tc} are the weight of the service function diversity of each manufacturing resource F_{fu}, the resource type owned by resource providers F_{ty} and

the number of cooperative enterprises F_{co}, respectively. $\omega_{fu}+\omega_{ty}+\omega_{Tc}=1$.

Ability to respond to changes in manufacturing resources:

$$\max F_R=\max\left(\omega_{re}F_{re}+\omega_{sa}F_{sa}+\omega_{Rc}F_{co}\right)=\max\left[\omega_{re}\frac{\sum_{i=1}^{n}F_{re}(i)}{n}+\omega_{sa}\frac{\sum_{i=1}^{n}F_{sa}(i)}{n}+\omega_{Rc}\frac{\sum_{i=1}^{n}F_{co}(i)}{n}\right]$$

where ω_{re}, ω_{sa}, and ω_{Rc} are the weight of the reliability of each manufacturing resource F_{re}, the number of manufacturing resources with the same function that the resource provider has F_{sa} and the number of cooperative enterprises F_{co}, respectively. $\omega_{re}+\omega_{sa}+\omega_{Rc}=1$.

Service evaluation:

$$\max F_E=\max\frac{\sum_{i=1}^{n}F_E(i)}{n}$$

The measurement standards of the above three indicators are different, so normalization is performed. The normalization calculation formula of the positive attribute indicator is as follows:

$$Norm\left(Q_k\right)=\begin{cases}\dfrac{q_k-\min q_k}{\max q_k-\min q_k}, & \min q_k\neq\max q_k\\ 1, & \min q_k=\max q_k\end{cases}$$

where $\min q_k$ and $\max q_k$ represent the minimum and maximum values of the kth combination in all possible combination paths, respectively.

Maximum flexibility:

$$\max F=\max\left(F_T,F_R,F_E\right)=\max\left(\omega_{F_T}\left(norm(F_T)\right)+\omega_{F_R}\left(norm(F_R)\right)+\omega_{F_E}\left(norm(F_E)\right)\right)$$

where ω_{F_T}, ω_{F_R}, and ω_{F_E} are the weight of the ability to respond to changes in manufacturing tasks F_T, the ability to respond to changes in manufacturing resources F_R and service evaluation F_E, respectively. $\omega_{F_T}+\omega_{F_R}+\omega_{F_E}=1$.

2) Maximum resource use

$$max\left(MRU\right)=max\left(\frac{L_d}{\sum_{j=1}^{n}L_{p_j}}\right)$$

where L_d is the demand load of service demanders, L_{pj} is the service load of resource providers, and $\sum_{j=1}^{n}L_{p_j}$ is the sum of the remaining service load of n subtasks.

The constraints of the middle-level programming model are as follows:

1) Constraint on the functional diversity of manufacturing resources

The functional diversity of any manufacturing resource $F_{fu}\left(i\right)$ in CMfg-SCOS should not be less than the minimum functional diversity $F_{fu\min}$ required by cloud platform operators.

$$F_{fu}\left(i\right)\geq F_{fu\min},i=1,2,...,n$$

2) Constraint on manufacturing resource types

The number of types of any manufacturing resources $F_{ty}\left(i\right)$ in CMfg-SCOS should not be less than the minimum number of types of manufacturing resources $F_{ty\min}$ required by the cloud platform operators.

$$F_{ty}\left(i\right)\geq F_{ty\min},i=1,2,...,n$$

3) Constraint on manufacturing resource reliability

The reliability of any manufacturing resource service $F_{re}\left(i\right)$ in CMfg-SCOS should not be less than the minimum reliability $F_{re\min}$ required by cloud platform operators.

$$F_{re}\left(i\right)\geq F_{re\min},i=1,2,...,n$$

4) Constraint on the number of manufacturing resources with the same function

The number of any manufacturing resources with the same function $F_{sa}\left(i\right)$ in CMfg-SCOS should not be less than the minimum number of resources $F_{sa\min}$ required by cloud platform operators.

$$F_{sa}\left(i\right)\geq F_{sa\min},i=1,2,...,n$$

5) Constraint on the number of cooperative enterprises

The number of cooperative enterprises owned by any manufacturing resource provider $F_{co}(i)$ in CMfg-SCOS should not be less than the minimum number of cooperative enterprises $F_{co\min}$ required by cloud platform operators.

$$F_{co}(i) \geq F_{co\min}, i = 1, 2, ..., n$$

6) Constraint on service evaluation

The service evaluation of any manufacturing resource $F_E(i)$ in CMfg-SCOS should not be lower than the minimum service evaluation $F_{E\min}$ required by cloud platform operators.

$$F_E(i) \geq F_{E\min}, i = 1, 2, ..., n$$

7) Constraint on resource use

The sum of the remaining service load of resource providers $\sum_{j=1}^{n} L_{p_j}$ (the load capacity of the SCOS) should be greater than the demand load of service demanders L_d.

$$L_d \leq \sum_{j=1}^{n} L_{p_j}, i = 1, 2, ..., n$$

(3) The upper-level optimization problem

As the upper-level decision makers of the three-tier programming model, resource providers take maximum economic benefits as their goal. The objective function is as follows:

$$\max E_{to} = \max(B - C) = \max\left[\sum_{i=1}^{n} P_i - \left(\sum_{i=1}^{n} C_1(i) + \sum_{i=1}^{n} C_2(i)\right)\right]$$

where $\sum_{i=1}^{n} C_1 = \sum_{i=1}^{n}\left(C_{11}(i) + C_{12}(i) + C_{13}(i)\right)$, $\sum_{i=1}^{n} C_2(i) = \sum_{i=1}^{n}\left(C_{21}(i) + C_{22}(i)\right)$

The constraint of the upper-level programming model is as follows:

The sales price of any manufacturing resource P_i in CMfg-SCOS should not be lower than the input cost $P_{i\min}$ of resource providers or higher than the highest service price $P_{i\max}$ that can be sustained by service demanders.

$$C_{1i} + C_{2i} \le P_i \le P_{i\max}, i = 1, 2, ..., n$$

(4) The three-tier programming model

The three-tier model of this study takes service demanders as the lower-level decision makers, cloud platform operators as the middle-level decision makers and resource providers as the upper-level decision makers. Figure 3 is a schematic diagram of the three-tier programming CMfg-SCOS model.

(U)
$$\max E_{to} = \max\left(B - C\right)$$

$$s.t.\left\{C_{1i} + C_{2i} \le P_i \le P_{i\max}, i = 1, 2, ..., n.\right.$$

(M)
$$\begin{cases} \max F = \max\left(F_T, F_R, F_E\right) \\ max\left(MRU\right) \end{cases}$$

$$s.t.\begin{cases} F_{fu}\left(i\right) \ge F_{fu\min}, i = 1, 2, ..., n; \\ F_{ty}\left(i\right) \ge F_{ty\min}, i = 1, 2, ..., n; \\ F_{re}\left(i\right) \ge F_{re\min}, i = 1, 2, ..., n; \\ F_{sa}\left(i\right) \ge F_{sa\min}, i = 1, 2, ..., n; \\ F_{co}\left(i\right) \ge F_{co\min}, i = 1, 2, ..., n; \\ F_E\left(i\right) \ge F_{E\min}, i = 1, 2, ..., n; \\ L_d \le \sum_{j=1}^{n} L_{p_j}, i = 1, 2, ..., n. \end{cases}$$

(L)
$$\begin{cases} \min T_{to} = \min\left(T_{ma} + T_{tr} + T_{wa}\right) \\ \min C_{to} = \min\left(C_{ma} + C_{tr}\right) \\ \max Q = \min\left(1 - Q_{se}\right) \end{cases}$$

$$s.t.\begin{cases} T_{to} \le T_{\max}; \\ C_{to} \le C_{\max}; \\ Q_{se}\left(i\right) \ge Q_{\min}, i = 1, 2, ..., n. \end{cases}$$

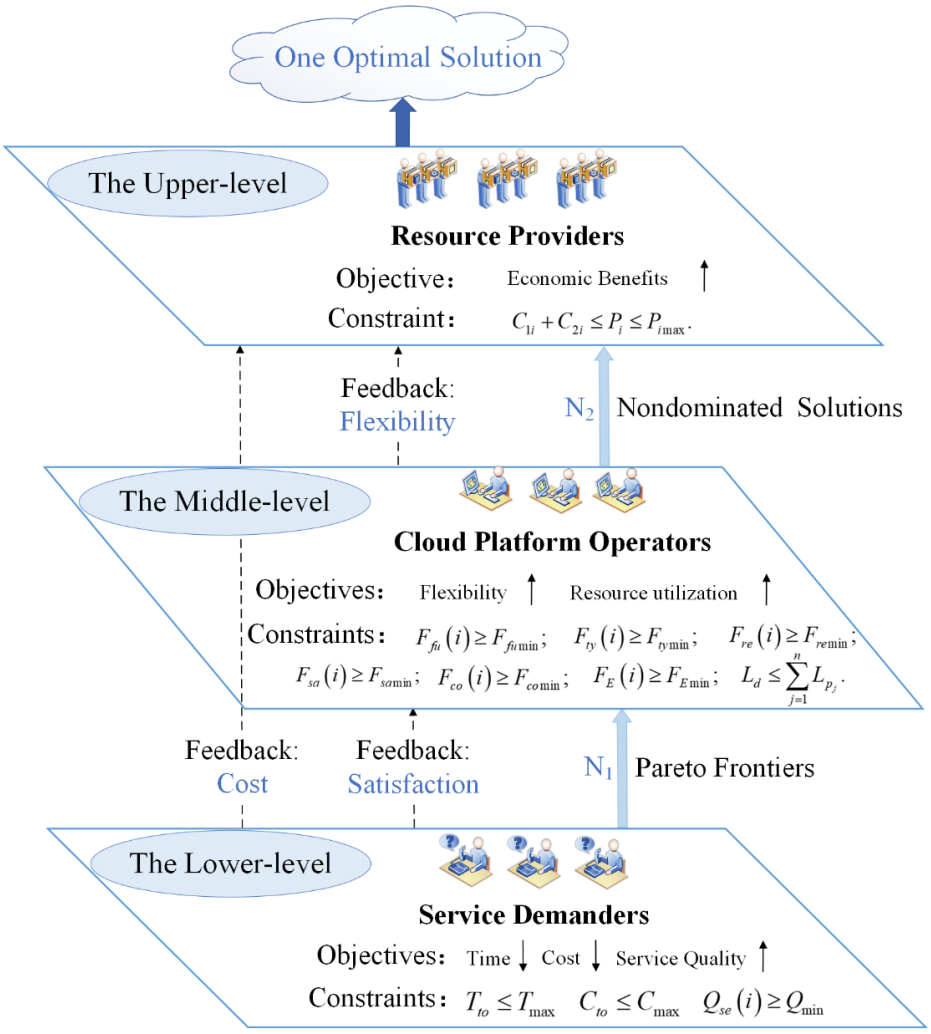


Figure 3. Schematic diagram of the three-tier programming CMfg-SCOS model

4 Solution algorithm

4.1 Basic Principles of NSGA-II

The number of objectives in this study is more than 4, which belongs to the high-dimensional multiobjective solution problem. It is difficult to find the approximate Pareto optimal solutions directly by the multiobjective evolutionary algorithm. Therefore, the hierarchical multiobjective algorithm is used to solve the problem. Three-tier programming is an NP-hard problem for multiobjective optimization, and the process of solving this problem is very complicated. Therefore, many intelligent optimization algorithms have been proposed to solve this type of problem, such as NSGA-II (Deb et al., 2002), MOEA/D (Zhang & Li, 2007), and HypE (Bader & Zitzler, 2011). Among them, NSGA-II (nondominated sorting genetic algorithm II) is an improved algorithm proposed by Deb et al. (2002) on the basis of NSGA. NSGA-II has

the following advantages (Jiao et al., 2009):

(1) The fast, nondominated sorting method based on classification is adopted in NSGA-II, which reduces the computational complexity of the algorithm from $O\left(mN^3\right)$ to $O\left(mN^2\right)$, where m represents the number of objective functions, and N represents the number of individuals in the population.

(2) The crowding degree is used to calibrate the fitness values of different individuals at the same level after fast, nondominated sorting in NSGA-II, so that the individuals in the current Pareto front can be extended to the entire Pareto front and distributed as evenly as possible.

(3) An elite reservation mechanism is introduced to generate the next generation population using the offspring of selected individuals to participate in reproduction to compete with their parents, which is conducive to retaining excellent individuals and improving the overall evolutionary level of the population.

4.2 Improved NSGA-II with advancement and inheritance

NSGA-II is by far one of the best evolutionary multiobjective optimization algorithms. The CMfg-SCOS model proposed in this study belongs to a three-level multiobjective optimization problem. The original NSGA-II cannot solve this model because obtaining the optimal solution of this model is complicated. Therefore, this study proposes an improved NSGA-II with advancement and inheritance (a-i-NSGA-II) to solve the three-tier programming CMfg-SCOS model. The algorithm includes the following improvements: The encoding method and the population screening mechanism are improved to avoid local optimization; the target values of the next level are considered in advance when solving the crowding degree so that more correlation is between levels; the input solutions of the lower level inherit the output solutions of the upper level to make the solution of the three-layer model more reliable. The steps of a-i-NSGA-II are as follows:

Step 1: Limit the search space of the a-i-NSGA-II to the constraints of the upper, middle and lower levels of the three-tier programming CMfg-SCOS model. Form individual genes corresponding to the CMfg-SCOS, in which the number of subtasks is taken as the gene length and the number of subtasks in the candidate set is taken as the gene value.

Step 2: Randomly generate an initial population $P_g\left(g=0\right)$ with several N,

calculate the lower-level target values and fast, nondominant sorting of population P_0. Then, the crowding degree of each individual is calculated. To make the optimal solution more relevant to the target of the next level, the target values of the middle level are considered in advance during the calculation. After that, sort the population according to rank and crowding degree. The crowding degree of the ith individual is defined as: the sum of the difference between all the objective function values of the $(i-1)th$ individual and the $(i+1)th$ individual, and the calculation formula is as follows:

$$i_d = \sum_{j=1}^{m} \left(\left| f_j^{i+1} - f_j^{i-1} \right| \right)$$

Where i_d is the crowding degree of the ith individual, m is the number of objective functions. Since target values of the middle layer is considered in advance, m is the sum of the number of target values of the lower level and the middle level (for the model of this study, $m=6$)

Step 3: Through binary tournaments, individuals are selected from population P_g, and crossover and mutation operations are performed to generate offspring populations C_g and M_g numbered at N_c and N_m, respectively.

Step 4: Combine populations P_g, C_g, and M_g to obtain a combined population T_g numbered at $N+N_c+N_m$. Then, eliminate the individuals in T_g that do not meet the constraints of the upper, middle and lower levels of the three-tier programming CMfg-SCOS model (the specific constraints of the model in this study are in section 3.2.2). The remaining individuals form a new population R_g. If the number of individuals in R_g is less than N, then let $P_g = R_g$ and repeat steps 3 to 4 until the number of individuals in R_g is greater than or equal to N.

Step 5: Calculate the lower-level target values and fast, nondominant sorting of population R_g. Then, calculate the crowding degree of each individual with the consideration of the middle-level targets in advance. According to the elite retention strategy, select the best individuals to form a new population P_{g+1}.

Step 6: Letting $g = g+1$, repeat steps 3 to 5 until the termination condition of the maximum genetic algebra is reached, and the obtained Pareto frontiers are the lower-level solutions Q_1 of the three-tier programming CMfg-SCOS model.

Step 7: Take the Pareto solution set Q_1 obtained in Step 6 as the feasible solutions of the middle level of the three-tier programming CMfg-SCOS model. After calculating the middle-level target values and fast, nondominant sorting of solution set Q_1, the

solutions of "Rank=1" are the middle-level solutions Q_2 of the three-tier programming CMfg-SCOS model.

Step 8: Take the solution set Q_2 obtained in Step 7 as the feasible solutions of the upper level of the three-tier programming CMfg-SCOS model. Calculate the upper-level target values of each solution and sort them according to the degree of superiority and inferiority. Then, the solution ranked as 1 is the final solution of the three-level programming model.

Algorithm 1 Pseudo-code of a-i-NSGA-II

Inputs: The population size N and maximum number of iterations T
Outputs: The best solution and its fitness value

procedure a-i-NSGA-II
Initialize the random population $P_g(g=0)$
Calculate the first-level fitness of each individual in P_g
Fast, non-dominant sorting of population P_g
Calculate the crowding degree of each individual in P_g
Sort the population according to rank and crowding degree
    while (x < Max number of iterations) **do**
        Select population from P_g through binary tournaments
        Generate offspring population C_g by crossover
        Generate offspring population M_g by mutation
        Combine populations P_g, C_g, and M_g to obtain R_g
        Eliminate the individuals in T_g that do not meet the constraints
        The remaining individuals form a new population R_g
        Calculate the first-level fitness of each individual in R_g
        Fast, non-dominant sorting of population R_g
        Calculate the crowding degree of each individual in R_g
        Sort the population R_g according to rank and crowding degree
        Update the population according to the elite retention strategy
        $g = g+1$
    end while
Obtain the Pareto frontiers Q_1
Calculate the second-level fitness of each individual in Q_1
Fast, non-dominant sorting of population Q_1
Obtain the solutions of "Rank=1" Q_2
Calculate the third-level fitness of each individual in Q_2
Return the best solution and its fitness value
end procedure

The process of solving the three-tier programming CMfg-SCOS model using a-i-NSGA-II is shown in Figure 4.

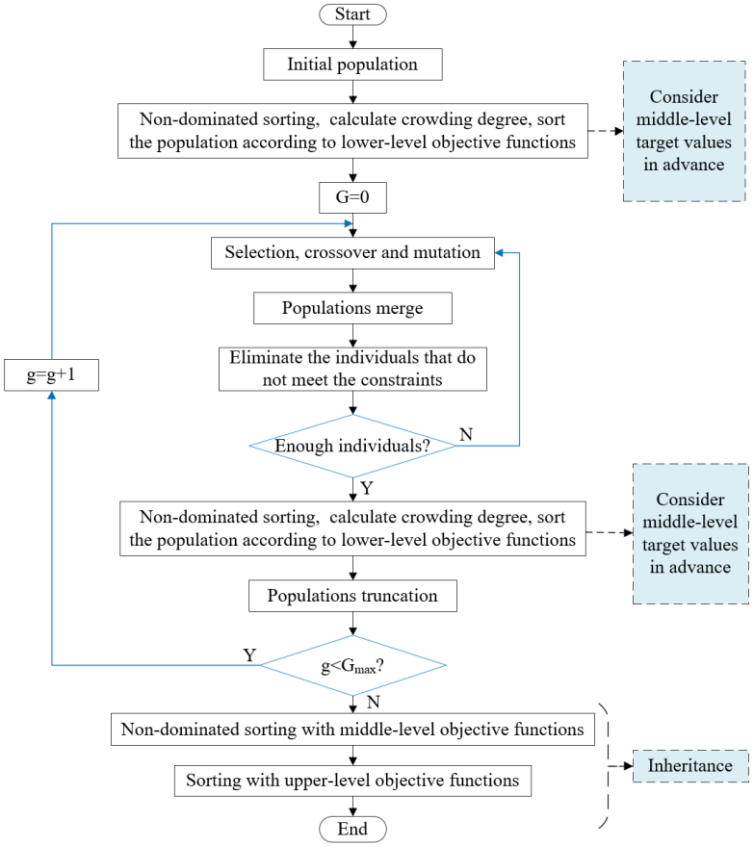


Figure 4. Process of solving the three-tier model using a-i-NSGA-II

4.3 Numerical experiments

To verify the effectiveness of the proposed algorithm, a-i-NSGA-II is compared with the original NSGA-II algorithm, multiobjective particle swarm optimization (Coello Coello & Lechuga, 2002) (MOPSO), and the multiobjective spotted hyena optimizer (Dhiman & Kumar, 2017) (MOSHO). Without loss of generality, five numerical experiments are designed in this study, and the numbers of subtasks and candidate sets of subtasks in the 5 experiments are 10_5, 10_10, 20_5, 20_10, and 30_10. Each experiment is run 50 times independently using the various algorithms. The experimental environment is MATLAB R2016b for the 64-bit Windows 10 1909 operation system on a 1.6 GHz Intel Core i5-8250U CPU with a 16-GB RAM.

The lower-level and middle-level models are multiobjective solution problems. Therefore, in this study, the representative multiobjective algorithm evaluation metrics "generational distance (GD)", "maximum spread (MS)", and "inverted generational distance (IGD)" are selected to measure the convergence, diversity and comprehensive

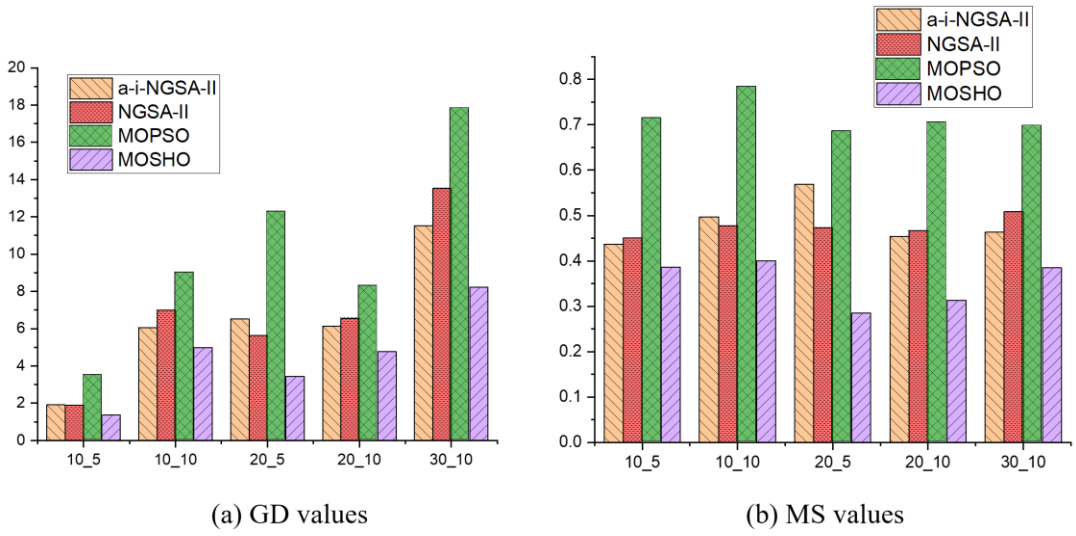
performance of the algorithms, respectively (Han & Zhen-Yu, 2019). The upper level of the model is a single-objective solution problem. Therefore, the quality of the target values of the upper level is used to measure the performance of the algorithms.

(1) Algorithm performance at the lower-level

Generational distance (Schutze et al., 2012) refers to the convergence of the approximated solution set to the true Pareto front. The smaller the GD value is, the better the convergence of the approximated solution set, indicating that the solution set is closer to the entire Pareto front. To obtain the true Pareto front, this study first calculates 2000 approximate solutions using 4 algorithms in each experiment, and then these 2000 approximate solutions are sorted in a nondominated order. The solutions of "Rank=1" are assumed to be the true Pareto front of each experiment.

Maximum spread (Zitzler et al., 2000) refers to the diversity of the approximated solution set. The higher the MS value is, the larger the area covered by the approximate solution set on the true Pareto front, indicating better extension performance of the approximate solution set. Figure 5 shows the average GD and MS values obtained by each algorithm over 50 independent runs at the lower level.

Inverted generational distance (Bosman & Thierens, 2003) refers to the mean of the minimum distance between the individuals on the true Pareto front and the approximate solution set obtained by the algorithm. The smaller the IGD value is, the closer the approximate solution set is to the entire Pareto front, indicating the better comprehensive performance of the algorithm. Figure 6 presents standard statistical box plots (median, dispersion and outliers) of the IGD values over 50 independent runs at the lower level.



(a) GD values (b) MS values

Figure 5. GD and MS values of the lower level

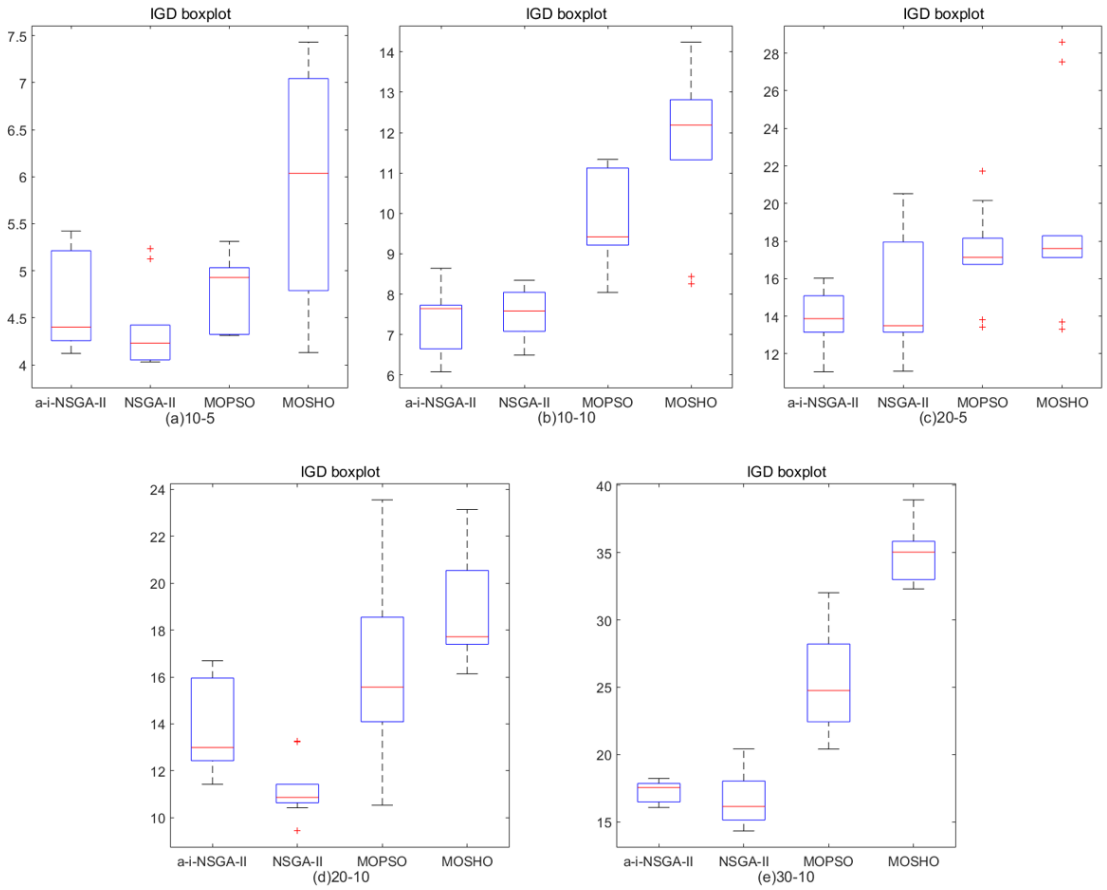


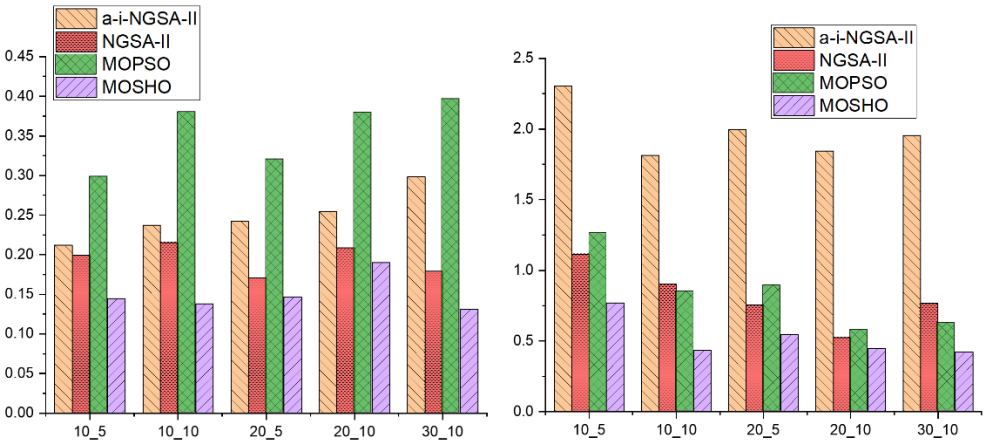
Figure 6. IGD values of the lower level

As shown in Figure 5, the GD values of MOSHO are always the smallest and the MS values of MOPSO are always the largest in the five experiments, indicating that MOSHO has better convergence and MOPSO has better diversity when solving the lower level. The GD and MS values of a-i-NSGA-II and NSGA-II are in the middle position, and their values are approximately equal. As shown in Figure 6, the average and standard deviation of the IGD values obtained from a-i-NSGA-II and NSGA-II are smaller than those obtained from other algorithms, indicating that although the convergence and diversity of a-i-NSGA-II and NSGA-II are not the strongest, the overall performance is better than that of MOPSO and MOSHO. This is because, for the coding method of the CMfg-SCOS in this study, the exploration mechanism of MOSHO makes the lower-level solutions more convergent, and the particle update mechanism of MOPSO makes the lower-level solutions more diversified. The combination of simulating binary crossover, polynomial mutation and elite strategy of a-i-NSGA-II and NSGA-II makes the comprehensive performance in the lower-level solutions stronger. In addition, the performances of a-i-NSGA-II and NSGA-II at the lower level are very similar, indicating that although a-i-NSGA-II adds the advance

mechanism when solving the lower-level solutions, it does not affect the overall performance.

(2) Algorithm performance at the middle level

To obtain the solutions of the middle level of the model, the inheritance mechanism in a-i-NSGA-II is also adopted in NSGA-II, MOPSO and MOSHO in this study. Figure 7 shows the average GD and MS values obtained by each algorithm over 50 independent runs at the middle level. Figure 8 presents standard statistical box plots of the IGD values over 50 independent runs at the middle level.



(a) GD values (b) MS values

Figure 7. GD and MS values of the middle level

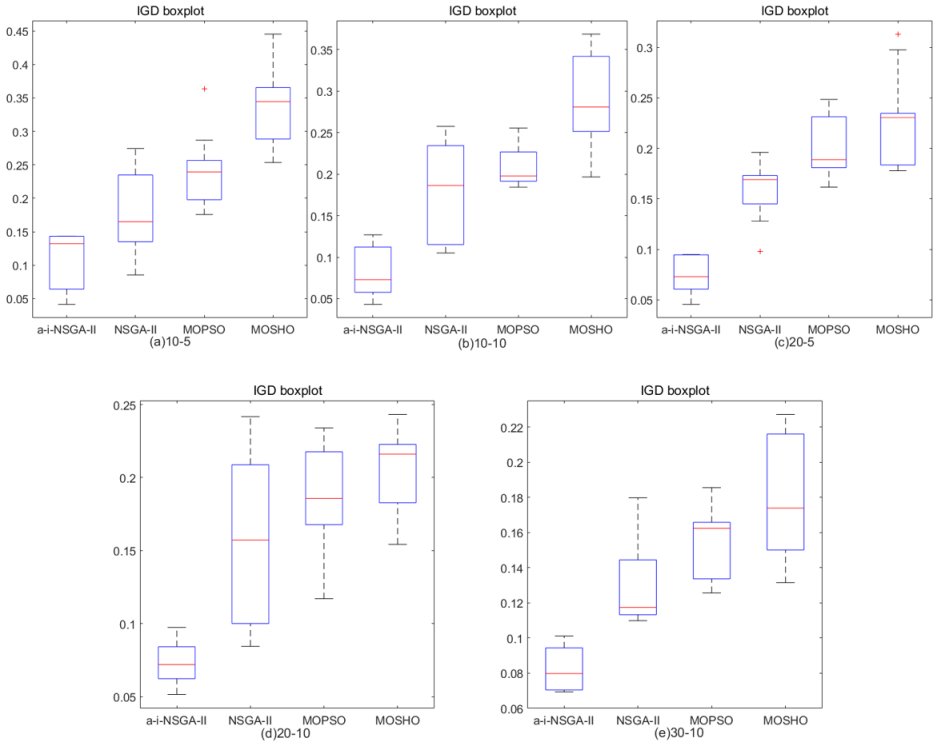


Figure 8. IGD values of the middle level

As shown in Figure 7, the GD values of the MOSHO remain the smallest, which indicates that the search for the prey (exploration) mechanism of the MOSHO improves its convergence. The MS values of a-i-NSGA-II are substantially larger than those of the other three algorithms, indicating that a-i-NSGA-II has very good diversity when solving the middle-level solutions. These results are due to the addition of the proposed advance mechanism when obtaining the solution of the middle level (the targets of the middle level are considered in advance). Correspondingly, the convergence of a-i-NSGA-II when solving the middle-level solutions is slightly weaker than that of NSGA-II. As shown in Figure 8, the average and standard deviation of the IGD values obtained from a-i-NSGA-II are smaller than those obtained from the other three algorithms, indicating that the comprehensive performance of a-i-NSGA-II is optimal when obtaining the middle-level solutions.

In addition, Statistical Product and Service Solutions (SPSS) nonparametric tests are performed on the IGD values obtained by a-i-NSGA-II and NSGA-II. The hypothesis test (Wilcoxon signed-rank test (Woolson, 2007)) results indicate that the IGD values of a-i-NSGA-II and NSGA-II in the lower level come from the same distribution, while the IGD values of the middle level come from different distributions. Therefore, it is proven that a-i-NSGA-II with the proposed advance mechanism enhances the comprehensive performance (especially the diversity) of the middle-level solution while ensuring the stability of the lower-level solution.

(3) Algorithm performance at the upper level

Figure 9 shows the average target values (enterprise surplus) obtained by each algorithm over 50 independent runs at the upper level. The larger the target value is, the more profits the resource providers obtain and the better the solution quality obtained by the algorithm. The target values obtained by a-i-NSGA-II are slightly larger than those obtained by the other three algorithms, which is more obvious at a large scale. These results are due to the higher diversity of middle-level solutions that a-i-NSGA-II has (demonstrated in algorithm performance at the middle level). Because the output solutions of the middle level are the input of the upper level, the input of the upper level is more extensive (Figure 10). Therefore, it is proven that compared with the other three algorithms, a-i-NSGA-II improves the quality of the obtained upper-level solutions.

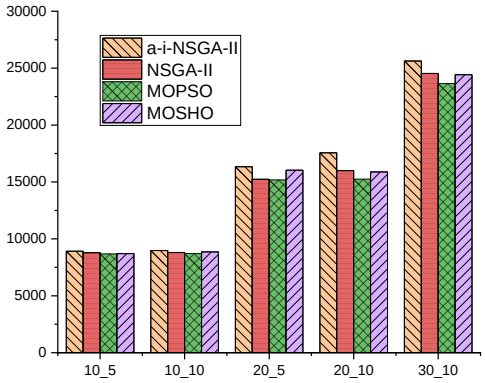


Figure 9. Target values of the upper level

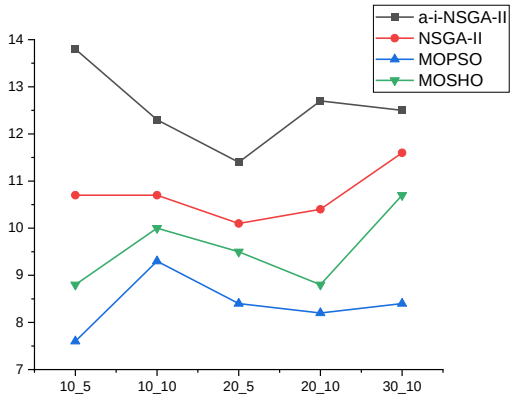


Figure 10. Input number of the upper level

In summary, when obtaining the lower-level solutions, although the convergence and diversity of a-i-NSGA-II are not optimal, its overall performance is good. When obtaining the middle-level solutions, a-i-NSGA-II enhances the diversity and comprehensive performance while ensuring the stability of the lower-level solutions. When obtaining the upper-level solution, because of the enhancement of the diversity of the middle-level solutions, the input of the upper-level solution is correspondingly wider, and the quality of the solution is higher.

In addition, compared with NSGA-II, a-i-NSGA-II has several advantages: 1) The original NSGA-II will only solve the two-level model, while a-i-NSGA-II is able to solve the three-tier model and satisfy the interests of three types of users because of the addition of "inheritance mechanism". 2) Due to the "advance mechanism" added to a-i-NSGA-II, the diversity of a-i-NSGA-II in the middle-level solution is very good, thus enhancing its comprehensive performance. 3) Compared with NSGA-II without "advance mechanism", a-i-NSGA-II makes the input of the upper-level solution wider and the quality of the final solution better.

5 Case study

This section verifies the applicability of the proposed three-tier programming CMfg-SCOS model and a-i-NSGA-II algorithm. An automobile fuel tank assembly company receives a processing order for 100 pieces of customized automobile fuel tank products. The quotation is 60,000 Chinese yuan (CNY), and the delivery time is 2 months. The order needs to be completed through the CMfg service platform outsourcing cooperation. The automobile fuel tank assembly company decomposes the processing order and publishes the decomposed atomic tasks that require outsourcing cooperation to the CMfg service platform. The resource library in the CMfg service

platform already has relevant fuel tank processing resources that can provide outsourcing services.

In this case, the automobile fuel tank assembly company is the service demander, the CMfg outsourcing service platform is the cloud platform operator, and resource providers is the manufacturers providing candidate service. The decomposition task structure of the customized automobile fuel tank in the CMfg environment is shown in Figure 11. The manufacturing of this product includes four tasks: T_1 oil line assembly, T_2 tank shell assembly, T_3 tank welding assembly, and T_4 tank main assembly. Each task is divided into several subtasks. T_1 oil line assembly is decomposed into four subtasks: manufacturing and assembly of crown lock, valve, quick connecter and oil line; T_2 tank shell assembly is decomposed into four subtasks: manufacturing and assembly of tank shell, E-ring, baffle and ICV; T_3 tank welding assembly is decomposed into five subtasks: manufacturing and assembly of welding stud, single clip, nipple, plastic bracket and dust cap for ICV; T_4 tank main assembly is decomposed into seven subtasks: manufacturing and assembly of pump, O-ring, lock ring, left strap, right strap, patch and paper label. Therefore, there are 20 subtasks in the manufacturing process of the customized automobile fuel tank, and each manufacturing subtask ST_i is assigned 5 candidate services $RS_i^j(m), m=1,\ldots,5$. Each candidate service corresponds to 1 manufacturer, so there are 100 manufacturers in this case. Logistics factors are reflected in the evaluation indicators (logistics time and logistics cost) of service demanders.

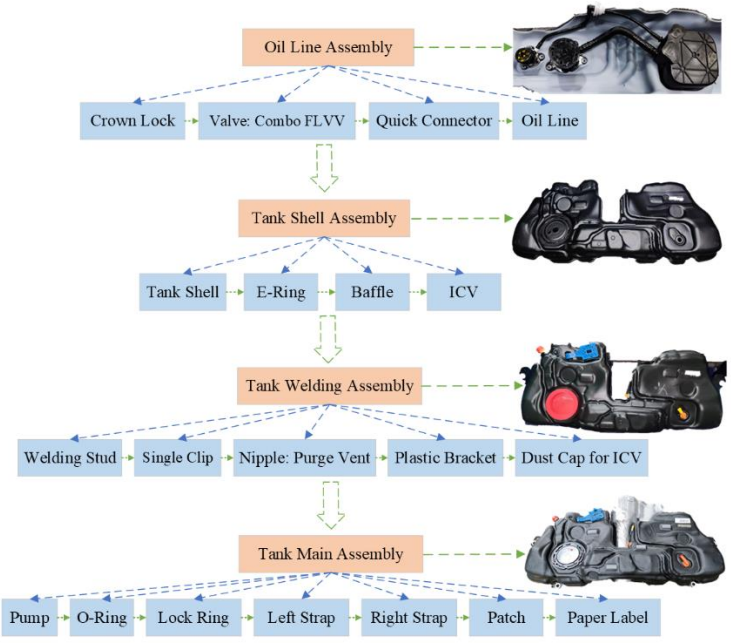


Figure 11. Decomposition task structure of the customized automobile fuel tank

A case of this scale proves the effectiveness of the algorithm in the numerical analysis in Section 4.3. The relevant parameters of the candidate service for each subtask are shown in Table 3 (the data on logistics time T_{tr} and logistics costs C_{tr} take up too much space for inclusion in this table).

Table 3. Relevant parameters of the candidate service for each subtask

Candidate Service		$T_{ma}(h)$	$T_{wa}(h)$	$C_{ma}(CNY)$	Q_{se}	F_{fu}	F_{ty}	F_{co}	F_{re}	F_{sa}	F_E	L_p	$B(CNY)$	$C_{11}(CNY)$	$C_{12}(CNY)$	$C_{13}(CNY)$	$C_{21}(CNY)$	$C_{22}(CNY)$
RS_1^1	(1)	10	3	500	0.93	1	4	4	0.91	3	0.91	21	1000	500	82	71	32	20
	(2)	15	5	300	0.97	2	6	7	0.89	7	0.92	14	600	300	51	45	21	15
	(3)	12	4	400	0.95	1	8	5	0.94	6	0.90	10	800	400	65	61	29	18
	(4)	7	2	600	0.95	3	6	8	0.89	8	0.94	15	1200	600	95	80	35	25
	(5)	13	4	450	0.96	2	7	6	0.88	5	0.88	13	900	450	75	68	27	18
RS_1^2	(1)	100	10	4700	0.98	5	6	7	0.95	2	0.93	15	9400	4700	760	700	300	180
	(2)	120	12	4000	0.93	4	3	5	0.90	3	0.86	8	8000	4000	670	590	250	190
	(3)	108	10	4500	0.97	2	8	7	0.94	2	0.91	5	9000	4500	750	680	270	185
	(4)	98	10	5000	0.97	1	4	8	0.94	6	0.89	10	10000	5000	815	700	310	200
	(5)	110	10	4600	0.98	3	5	6	0.91	4	0.91	12	9200	4600	760	690	280	195
RS_1^3	(1)	12	4	400	0.94	1	4	2	0.86	2	0.96	22	800	400	63	58	25	15
	(2)	8	3	600	0.95	2	6	3	0.94	2	0.95	16	1200	600	97	83	37	26
	(3)	14	3	300	0.91	3	5	7	0.93	5	0.98	18	600	300	50	44	20	14
	(4)	11	3	500	0.91	3	6	8	0.93	6	0.90	15	1000	500	82	70	30	21
	(5)	11	3	450	0.97	2	5	5	0.94	3	0.94	17	900	450	75	66	23	17
RS_1^4	(1)	22	5	1000	0.95	2	3	3	0.91	3	0.96	20	2000	1000	162	140	58	44
	(2)	20	5	1200	0.97	4	8	5	0.92	7	0.91	9	2400	1200	190	170	73	52
	(3)	29	6	800	0.98	2	4	4	0.96	6	0.95	9	1600	800	130	120	50	30
	(4)	24	5	900	0.92	3	8	6	0.92	4	0.91	15	1800	900	155	99	66	50
	(5)	23	5	980	0.93	3	5	5	0.93	4	0.93	16	1960	980	160	138	56	40
RS_2^1	(1)	50	7	2000	0.95	1	6	8	0.93	7	0.95	12	4000	2000	330	285	120	88
	(2)	48	7	2100	0.94	5	4	4	0.90	6	0.94	14	4200	2100	340	295	130	95
	(3)	53	8	1900	0.91	2	2	7	0.87	3	0.91	17	3800	1900	320	265	110	68
	(4)	45	6	2200	0.93	1	4	2	0.95	2	0.91	10	4400	2200	360	305	140	115
	(5)	50	7	2000	0.92	3	5	4	0.91	4	0.91	15	4000	2000	330	285	120	88
RS_2^2	(1)	26	4	1200	0.92	4	12	8	0.96	5	0.98	14	2400	1200	190	173	70	53
	(2)	28	4	1100	0.97	3	5	6	0.88	6	0.89	16	2200	1100	165	150	68	54
	(3)	33	5	1000	0.93	1	3	5	0.87	3	0.92	9	2000	1000	162	141	55	43
	(4)	24	4	1400	0.95	2	7	2	0.97	2	0.90	20	2800	1400	210	190	93	62
	(5)	25	4	1300	0.96	3	4	4	0.89	4	0.89	17	2600	1300	195	184	77	59
RS_2^3	(1)	12	3	1000	0.92	4	5	6	0.88	6	0.93	14	2000	1000	162	140	58	44
	(2)	14	3	900	0.95	5	7	1	0.90	4	0.93	16	1800	900	155	95	65	51
	(3)	10	2	1100	0.93	1	4	2	0.92	3	0.89	20	2200	1100	165	150	67	55
	(4)	7	2	1200	0.96	3	7	7	0.95	8	0.91	15	2400	1200	190	174	71	54
	(5)	10	2	1100	0.95	2	6	4	0.91	5	0.91	13	2200	1100	165	151	66	55
RS_2^4	(1)	53	7	2000	0.96	4	10	7	0.92	7	0.90	5	4000	2000	340	285	130	90
	(2)	48	7	2200	0.96	1	12	6	0.89	3	0.99	8	4400	2200	360	306	141	116
	(3)	60	8	1800	0.94	2	6	5	0.93	5	0.94	9	3600	1800	310	190	130	103
	(4)	43	7	2500	0.91	3	9	4	0.98	6	0.90	7	5000	2500	408	350	115	100

	(5)	50	7	2100	0.96	3	7	5	0.89	4	0.89	6	4200	2100	355	290	145	98
RS_3^1	(1)	17	3	400	0.92	3	6	2	0.94	4	0.98	20	800	400	65	62	29	18
	(2)	12	3	600	0.98	1	10	8	0.88	6	0.95	13	1200	600	95	81	36	25
	(3)	10	3	700	0.92	2	9	6	0.89	5	0.92	18	1400	700	100	90	40	28
	(4)	14	3	500	0.97	2	10	4	0.91	8	0.90	17	1000	500	82	70	30	21
	(5)	14	3	500	0.93	1	8	3	0.88	5	0.88	15	1000	500	81	71	31	21
RS_3^2	(1)	12	3	400	0.95	3	4	2	0.90	2	0.92	16	800	400	65	62	29	18
	(2)	10	2	500	0.98	2	8	3	0.85	4	0.86	10	1000	500	83	70	31	22
	(3)	17	3	300	0.91	1	3	2	0.85	1	0.94	9	600	300	50	43	20	15
	(4)	7	2	700	0.94	3	5	4	0.91	3	0.96	13	1400	700	100	89	39	28
	(5)	8	2	600	0.97	2	6	4	0.87	5	0.86	15	1200	600	96	82	35	26
RS_3^3	(1)	21	5	800	0.91	5	8	8	0.87	4	0.95	16	1600	800	130	131	60	37
	(2)	17	4	1000	0.98	3	11	6	0.88	8	0.95	18	2000	1000	162	140	58	44
	(3)	19	5	900	0.91	3	5	5	0.92	3	0.90	13	1800	900	150	130	50	45
	(4)	24	5	700	0.97	1	10	4	0.96	3	0.96	14	1400	700	99	89	41	27
	(5)	25	5	600	0.95	2	7	5	0.89	5	0.88	15	1200	600	98	84	36	26
RS_3^4	(1)	12	3	400	0.97	2	4	8	0.95	6	0.95	9	800	400	65	62	29	18
	(2)	12	3	400	0.97	4	12	2	0.96	8	0.98	14	800	400	63	58	26	15
	(3)	11	3	500	0.91	2	7	5	0.86	5	0.90	15	1000	500	82	69	29	20
	(4)	10	2	600	0.94	3	11	4	0.93	3	0.97	10	1200	600	95	80	35	24
	(5)	9	2	700	0.97	3	8	3	0.95	4	0.93	13	1400	700	101	89	38	28
RS_3^5	(1)	12	3	400	0.93	1	5	2	0.90	2	0.98	11	800	400	65	59	28	18
	(2)	14	3	300	0.97	2	4	5	0.91	1	0.93	18	600	300	50	44	20	15
	(3)	10	3	500	0.94	5	11	8	0.95	6	0.96	15	1000	500	82	69	28	20
	(4)	8	2	600	0.98	3	4	5	0.94	4	0.90	15	1200	600	96	82	35	24
	(5)	9	3	550	0.94	4	3	4	0.92	3	0.91	12	1100	550	93	78	31	20
RS_4^1	(1)	600	24	20000	0.92	1	6	3	0.97	5	0.92	5	40000	20000	3450	2800	1350	950
	(2)	480	20	22000	0.93	3	3	6	0.91	4	0.90	18	44000	22000	3600	3060	1410	1060
	(3)	450	20	25000	0.92	2	5	7	0.91	7	0.95	12	50000	25000	4080	3500	1650	1000
	(4)	520	22	19000	0.92	5	5	4	0.95	4	0.94	8	38000	19000	3300	2700	1200	900
	(5)	530	22	21000	0.96	4	4	5	0.93	3	0.96	14	42000	21000	3550	2900	1450	980
RS_4^2	(1)	26	4	1200	0.97	2	8	2	0.90	4	0.92	11	2400	1200	190	175	72	53
	(2)	24	4	1400	0.95	1	5	3	0.96	7	0.96	17	2800	1400	210	193	90	63
	(3)	28	5	1100	0.95	4	6	5	0.94	6	0.97	16	2200	1100	165	150	68	54
	(4)	30	5	1000	0.92	3	5	8	0.86	6	0.92	9	2000	1000	162	140	58	44
	(5)	28	5	1100	0.93	2	7	4	0.94	5	0.95	14	2200	1100	165	150	68	54
RS_4^3	(1)	45	6	1900	0.97	5	5	5	0.89	3	0.92	15	3800	1900	330	270	120	90
	(2)	47	7	2000	0.95	3	9	3	0.96	7	0.96	18	4000	2000	345	280	135	95
	(3)	50	7	1800	0.93	1	5	2	0.89	2	0.93	9	3600	1800	310	190	130	103
	(4)	43	7	2400	0.95	4	9	6	0.88	3	0.90	15	4800	2400	400	330	100	90
	(5)	45	7	2100	0.94	3	8	4	0.96	4	0.96	13	4200	2100	355	290	145	98
	(1)	58	8	2400	0.94	2	11	2	0.89	3	0.88	14	4800	2400	400	320	105	91

RS_4^4	(2)	60	8	2200	0.91	4	6	3	0.94	6	0.97	18	4400	2200	360	305	140	115
	(3)	56	8	2500	0.92	1	10	6	0.93	2	0.95	20	5000	2500	380	315	130	110
	(4)	52	8	3000	0.91	3	8	6	0.93	7	0.96	8	6000	3000	500	430	200	150
	(5)	58	8	2300	0.95	1	7	4	0.95	4	0.95	15	4600	2300	370	312	144	117
RS_4^5	(1)	48	7	1800	0.92	2	8	3	0.88	5	0.89	8	3600	1800	310	200	132	100
	(2)	52	8	2000	0.92	5	4	5	0.98	8	0.90	9	4000	2000	345	280	135	95
	(3)	41	7	2200	0.94	1	5	7	0.99	6	0.94	9	4400	2200	360	305	145	115
	(4)	43	7	1700	0.91	2	3	5	0.94	4	0.94	10	3400	1700	300	190	120	100
	(5)	50	8	1900	0.95	3	6	4	0.98	5	0.98	11	3800	1900	320	220	134	105
RS_4^6	(1)	36	6	1400	0.98	4	11	2	0.99	6	0.87	14	2800	1400	210	183	90	63
	(2)	33	6	1200	0.98	1	5	5	0.90	5	0.98	16	2400	1200	190	174	71	54
	(3)	29	5	1500	0.94	3	3	1	0.95	4	0.96	16	3000	1500	230	199	99	82
	(4)	31	5	1300	0.94	2	5	7	0.89	5	0.90	15	2600	1300	200	180	83	52
	(5)	36	6	1400	0.93	2	4	4	0.90	3	0.90	13	2800	1400	215	181	91	65
RS_4^7	(1)	14	4	300	0.93	1	3	2	0.95	2	0.91	8	600	300	47	43	18	13
	(2)	10	3	500	0.98	3	4	1	0.93	4	0.90	20	1000	500	82	71	32	20
	(3)	12	4	400	0.93	2	6	7	0.86	3	0.97	9	800	400	65	61	29	18
	(4)	8	3	600	0.91	1	4	8	0.87	4	0.98	7	1200	600	97	83	37	26
	(5)	10	3	500	0.94	4	5	4	0.93	5	0.93	10	1000	500	81	72	33	21

Notes: The relevant parameters are determined according to the processing order for 100 pieces.

(1) Validation of the proposed model

To verify the effectiveness of the proposed three-tier programming model, this study compares it with single-level programming model and bilevel programming model. For single-level programming model, it is difficult to solve it because there are six objectives (high-dimensional multiobjective problem). Therefore, the multiobjective problem is converted into single-objective problem by setting 6 weights. For bilevel programming model, the solution method of the lower level is the same as that of the three-tier programming model. In the upper level, there are 3 objectives, the multiobjective problem is converted into single-objective problem by setting 3 weights. In order to avoid the interference of different weight values, this study considers cases of different weights. As shown in Table 4, single-level programming model has 6 objectives and 7 ways to value weights, the upper layer of bilevel programming model has 3 objectives and 4 ways to value weights. w_1, w_2, w_3, w_4, w_5 and w_6 are the weights of time, cost, service quality, flexibility, resource utilization rate and enterprise surplus respectively.

Table 4 shows the results of the most frequent solutions for each case after 50 independent runs. For three-tier programming model, the total time T_{to} (1194 hours) and total cost C_{to} (44,208 yuan) are the smallest among the 12 cases, and the total service quality Q (19.08) is similar to that of single-level programming model and bilevel programming model. Therefore, from the indicators of time, cost and service quality, it can be seen that the solution obtained by the three-tier programming model can better meet the interests of service demanders. The flexibility F of the three-tier programming model (0.6896) is the largest among the 12 cases. The resource utilization rate MRU (0.7868) is not much different from that of bilevel programming model and is slightly lower than that of single-level programming model. Therefore, from the indicators of flexibility and resource utilization, it can be seen that the solution obtained by the three-tier programming model can better meet the interests of cloud platform operators. The enterprise surplus of single-level programming model is the lowest among three models (25,000-27,000 yuan), the enterprise surplus of bilevel programming model is between 30,000 yuan and 31,000 yuan, while the enterprise surplus of the three-tier programming model is the highest (31,273 yuan). Therefore, from the enterprise surplus indicator, it can be seen that the solution obtained by the three-tier programming model can better meet the interests of resource providers. Overall, through the analysis of above results, it can be verified that compared with

single-level programming model and bilevel programming model, the proposed three-tier programming model can satisfy the interests of service demanders, resource providers, and cloud platform operators at the same time.

Table 4. Results of case analysis for three models

Models	Weights						Candidate Set Chains	Service Demanders			Cloud Platform Operators		Resource Providers
	w_1	w_2	w_3	w_4	w_5	w_6		$T_{to}(h)$	$C_{to}(CNY)$	Q	F	MRU	$E(CNY)$
Single-level programming model	0.167	0.167	0.167	0.167	0.166	0.166	23231341114221424215	1475	44650	18.99	0.6018	0.875	25760
	0.25	0.15	0.15	0.15	0.15	0.15	23231321235251144255	1386	52188	18.95	0.5494	0.8545	26178
	0.15	0.25	0.15	0.15	0.15	0.15	23234341244215321523	1401	44559	19.1	0.5936	0.8536	26545
	0.15	0.15	0.25	0.15	0.15	0.15	23232341244231215111	1459	52654	19.08	0.5894	0.7823	25937
	0.15	0.15	0.15	0.25	0.15	0.15	23231311244212153215	1443	46854	19.02	0.5926	0.8292	27505
	0.15	0.15	0.15	0.15	0.25	0.15	23232341243151421215	1447	45107	18.97	0.5729	0.8606	26168
	0.15	0.15	0.15	0.15	0.15	0.25	23232311244212143523	1350	46857	19.04	0.5899	0.8367	27603
Bilevel programming model	—			0.3334	0.3333	0.3333	41222211222243244552	1211	45122	19.12	0.6410	0.7778	30783
	—			0.5	0.25	0.25	41122242324144244313	1271	46300	19.07	0.6181	0.8076	27039
	—			0.25	0.5	0.25	41224142323434454334	1252	46617	18.79	0.652	0.7824	27057
	—			0.25	0.25	0.5	41521142422243214312	1201	45077	19.19	0.6884	0.7664	30624
Three-tier programming model	—						41222442352143244542	1194	44208	19.08	0.6896	0.7868	31273

(2) Validation of the proposed algorithm

The three-tier programming CMfg-SCOS model proposed in this study is applied to the above case, and the four algorithms are used to solve the problem 50 times independently. Table 5 shows the results of 50 independent runs. In the lower level of the model, the mean IGD values obtained by a-i-NSGA-II and NSGA-II are lower, indicating their better comprehensive performance in the lower level; in the middle level of the model, the mean MS value obtained by a-i-NSGA-II is significantly greater than those of the other three algorithms, indicating that a-i-NSGA-II has a superior diversity in solving middle-level solutions, and the lower mean IGD value shows its better comprehensive performance; in the upper level of the model, a-i-NSGA-II has a higher proportion of the most frequent solution and higher mean target values of 50 independent solutions, which indicates that the solution obtained by a-i-NSGA-II is not only more stable but also has better quality. The above results are consistent with the numerical analysis results in Section 4.3, which demonstrates the correctness of the case solutions.

Table 5. Results of case analysis for four algorithms

	Lower level			Middle level			Upper level	
	GD	MS	IGD	GD	MS	IGD	The average of the target values	Proportion of the most frequent solution
a-i-NSGA-II	6.6983	0.6298	14.1465	0.2654	1.9745	0.0842	30413.7	72%
NSGA-II	6.2133	0.6234	13.9842	0.1834	0.7254	0.1265	30543.8	64%
MOPSO	12.5234	0.7135	17.2543	0.3267	0.8867	0.1876	30547.9	56%
MOSHO	5.4245	0.5745	18.4652	0.1764	0.6264	0.2231	29734.4	46%

On the basis of analysing the case solutions for 50 independent runs, the upper-level solution with the highest occurrence proportion is selected as the final solution of the case, and the solutions obtained by the four algorithms are shown in Table 6. Although all four algorithms meet the requirements that the quotation is 60,000 CNY and the delivery time is 2 months, the results of a-i-NSHA-II are better. For example, although the total service quality of service demanders obtained by a-i-NSGA-II is not optimal, the total time (1,194 hours) and total cost (44,208 yuan) are relatively small, which better meets the actual needs of service demanders. The flexibility of the cloud platform operators obtained by a-i-NSGA-II (0.6896) is similar to those of the other algorithms, but the resource utilization rate (0.7868) is much higher than those of the other three algorithms, which also meets the interests of the cloud platform operators. The enterprise surplus of the resource providers obtained by a-i-NSGA-II is 31,273

CNY, which is the largest among the four algorithms and satisfies the interests of the resource providers. In addition, the cost of the 100-piece order is 55626 CNY before the automobile fuel tank assembly company uses the CMfg service platform outsourcing cooperation, which is reduced by 20.52% to 44,208 CNY after this company uses the platform, and the manufacturing time is reduced by 17.08% over two months, proving the effectiveness of the proposed model and algorithm in this study.

Table 6. Final solutions of four algorithms

Algorithms	Candidate Set Chains	Lower level (Service Demanders)			Middle level (Cloud Platform Operators)		Upper level (Resource Providers)
		T_{to} (h)	C_{to} (CNY)	Q	F	MRU	E (CNY)
a-i-NSGA-II	41222442352143244542	1194	44208	19.08	0.6896	0.7868	31273
NSGA-II	41224442352243143522	1206	46213	19.14	0.5889	0.7167	30962
MOPSO	41321342452133254542	1208	51528	19.03	0.6960	0.7749	30304
MOSHO	41222242222233312322	1199	48888	19.18	0.6959	0.7394	29964

Therefore, through the analysis of the case solutions, it can be proven that the proposed three-tier programming CMfg-SCOS model can satisfy the interests of service demanders, resource providers, and cloud platform operators at the same time. In addition, compared with the original NSGA-II, MOPSO and MOSHO, the diversity, stability and quality of the solution obtained using a-i-NSGA-II are better.

6 Conclusion

This study proposed a three-tier programming model of CMfg-SCOS which considers the interests of service demanders, cloud platform operators, and resource providers in CMfg. An improved, fast and elitist multiobjective genetic algorithm with advancement and inheritance (a-i-NSGA-II) is proposed to solve this model. Then, numerical experiments and SPSS nonparametric tests prove that compared with the original NSGA-II, MOPSO and MOSHO, the a-i-NSGA-II algorithm with advance and inheritance mechanism not only enhances the diversity and comprehensive performance of middle-level solutions, but also improves the quality of the upper-level solutions. Finally, a case study of an automobile fuel tank assembly enterprise is conducted from the actual production. The results show that compared with single-level programming model and bilevel programming model, the proposed three-tier programming model has positive significance for simultaneously reducing time and cost for service demanders, improving the flexibility for cloud platform operators, and increasing the enterprise surplus for resource providers. The case analysis also verifies

the effectiveness of the proposed algorithm in this study, which not only has better performance but also improves the efficiency and reduces the cost of the automobile fuel tank assembly enterprise. The proposed model and improved algorithm in this study provide theoretical and technical support for solving the service composition problem of cloud platform, and have positive significance for helping CMfg users to make more reasonable decisions.

In this study, some limitations exist. Firstly, this study mainly considers the service composition problem in the case of the serial multi-service composition structure of manufacturing subtasks. In the future, the manufacturing service composition problem of parallel and mixed manufacturing subtasks in the actual manufacturing process will be studied in depth. Secondly, the evaluation indicators used in this study do not consider green factors. Some green and sustainable evaluation indicators deserve to be explored in subsequent research.

References

Adamson, G., Wang, L., Holm, M. & Moore, P. (2017). Cloud manufacturing - a critical review of recent development and future trends. International journal of computer integrated manufacturing, 30(4-5), 347-380.

Bader, J. & Zitzler, E. (2011). HypE: An algorithm for fast hypervolume-based many-objective optimization. Evolutionary computation, 19(1), 45-76.

Bosman, P.A. & Thierens, D. (2003). The balance between proximity and diversity in multiobjective evolutionary algorithms. IEEE Transactions on Evolutionary Computation, 7(2), 174-188.

Bouzary, H. & Chen, F.F. (2020). A classification-based approach for integrated service matching and composition in cloud manufacturing. Robotics and computer-integrated manufacturing, 66, 101989.

Bouzary, H. & Frank Chen, F. (2018). Service optimal selection and composition in cloud manufacturing: a comprehensive survey. International journal of advanced manufacturing technology, 97(1), 795-808.

Cao, Y., Wang, S., Kang, L. & Gao, Y. (2016). A TQCS-based service selection and scheduling strategy in cloud manufacturing. The International Journal of Advanced Manufacturing Technology, 82(1-4), 235-251.

Chen, F., Dou, R., Li, M. & Wu, H. (2016). A flexible QoS-aware Web service composition method by multi-objective optimization in cloud manufacturing. Computers & industrial engineering, 99, 423-431.

Cheng, F., Yu, S., Chu, J. & Fan, J. (2018). Research on Manufacturing Resources Optimal Allocation Strategy of 3D Printing Cloud Service Platform. In: (pp. 1-6): Chinese Automation and Computing Society in the UK - CACSUK.

Coello Coello, C.A. & Lechuga, M.S. (2002). MOPSO: a proposal for multiple objective particle swarm optimization. In: (Vol. 2, pp. 1051-1056 vol.1052): IEEE.

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

Dhiman, G. & Kumar, V. (2017). Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. Advances in engineering software (1992), 114, 48-70.

Dong, T., Xue, F., Xiao, C. & Li, J. (2020). Task scheduling based on deep reinforcement learning in a cloud manufacturing environment. Concurrency and Computation-Practice & Experience, 32(11).

Esposito, C., Castiglione, A., Martini, B. & Choo, K.-K.R. (2016). Cloud Manufacturing: Security, Privacy and Forensic Concerns. Ieee Cloud Computing, 3(4), 16-22.

Han, H.U. & Zhen-Yu, L.I. (2019). A Survey of Performance Indicators for Multi-objective Evolutionary Algorithms. Software Guide.

He, W., Luan, S., Jia, G. & Zong, H. (2017). Resource allocation based on prospect theory in cloud manufacturing environment. In: 2017 International

Conference on Computer Systems, Electronics and Control (ICCSEC) (pp. 1329-1332): IEEE.

Huang, S.Q., Gu, X.J., Zhou, H.M. & Chen, Y.R. (2018). Two-dimensional optimization mechanism and method for on-demand supply of manufacturing cloud service. Computers & industrial engineering, 117, 47-59.

Jiao, L., Yang, D. & Ma, W. (2009). Research on evolutionary multi-objective optimization algorithms. Journal of Software, 20(2), 271-289.

Li, B., Zhang, L., Wang, S., Tao, F. & Chai, X. (2010). Cloud manufacturing:a new service-oriented networked manufacturing model. Computer Integrated Manufacturing Systems, 16(1), 1-7+16.

Li, X. (2015). Empirical Research on Enterprise Economic Benefits Evaluation Based on Super-efficiency DEA. Management & Engineering(19), 55.

Li, Y., Chu, F., Chu, C. & Zhu, Z. (2019). An efficient three-level heuristic for the large-scaled multi-product production routing problem with outsourcing. European journal of operational research, 272(3), 914-927.

Lim, M.K., Xiong, W. & Lei, Z. (2020). Theory, supporting technology and application analysis of cloud manufacturing: a systematic and comprehensive literature review. Industrial management + data systems, 120(8), 1585-1614.

Lin, Y.-K. & Chong, C.S. (2017). Fast GA-based project scheduling for computing resources allocation in a cloud manufacturing system. Journal of intelligent manufacturing, 28(5), 1189-1201.

Liu, N., Li, X. & Shen, W. (2014). Multi-granularity resource virtualization and sharing strategies in cloud manufacturing. Journal of network and computer applications, 46, 72-82.

Liu, Y., Wang, L., Wang, X.V., Xu, X. & Zhang, L. (2019). Scheduling in cloud manufacturing: state-of-the-art and research challenges. International journal of production research, 57(15-16), 4854-4879.

Liu, Y., Xu, X., Zhang, L., Wang, L. & Zhong, R.Y. (2017). Workload-based multi-task scheduling in cloud manufacturing. Robotics and computer-integrated manufacturing, 45, 3-20.

Meng, Q.N. & Xu, X. (2018). Price forecasting using an ACO-based support vector regression ensemble in cloud manufacturing. Computers & industrial engineering, 125, 171-177.

Mourad, M.H., Nassehi, A., Schaefer, D. & Newman, S.T. (2020). Assessment of interoperability in cloud manufacturing. Robotics and computer-integrated manufacturing, 61, 101832.

Que, Y., Zhong, W., Chen, H., Chen, X. & Ji, X. (2018). Improved adaptive immune genetic algorithm for optimal QoS-aware service composition selection in cloud manufacturing. The International Journal of Advanced Manufacturing Technology, 96(9), 4455-4465.

Schutze, O., Esquivel, X., Lara, A. & Coello, C.A.C. (2012). Using the averaged Hausdorff distance as a performance measure in evolutionary multiobjective optimization. IEEE Transactions on Evolutionary Computation, 16(4), 504-522.

Su, K., Xu, W. & Li, J. (2015). Manufacturing resource allocation method based on bi-

level programming in cloud manufacturing. Computer Integrated Manufacturing Systems, 21(7), 1941-1952.

Tao, F., Zhang, L., Guo, H., Luo, Y.-L. & Ren, L. (2011). Typical characteristics of cloud manufacturing and several key issues of cloud service composition. Computer Integrated Manufacturing Systems, 17(3), 477-486.

Tao, F., Zhao, D., Hu, Y. & Zhou, Z. (2008). Resource Service Composition and Its Optimal-Selection Based on Particle Swarm Optimization in Manufacturing Grid System. IEEE transactions on industrial informatics, 4(4), 315-327.

Thekinen, J. & Panchal, J.H. (2017). Resource allocation in cloud-based design and manufacturing: A mechanism design approach. Journal of manufacturing systems, 43, 327-338.

Wang, F., Laili, Y. & Zhang, L. (2020). A many-objective memetic algorithm for correlation-aware service composition in cloud manufacturing. International journal of production research, 1-19.

Wang, T., Zhang, P., Liu, J. & Gao, L. (2021). Multi-user-oriented manufacturing service scheduling with an improved NSGA-II approach in the cloud manufacturing system. International journal of production research, 1-18.

Wang, T.R., Li, C., Yuan, Y.H., Liu, J. & Adeleke, I.B. (2019). An evolutionary game approach for manufacturing service allocation management in cloud manufacturing. Computers & industrial engineering, 133, 231-240.

Woolson, R. (2007). Wilcoxon signed-rank test. Wiley encyclopedia of clinical trials, 1-3.

Wu, Y., Jia, G. & Cheng, Y. (2020). Cloud manufacturing service composition and optimal selection with sustainability considerations: a multi-objective integer bi-level multi-follower programming approach. International journal of production research, 58(19), 6024-6042.

Yang, B., Wang, S.L., Cheng, Q.Q. & Jin, T.G. (2021). Scheduling of field service resources in cloud manufacturing based on multi-population competitive-cooperative GWO. Computers & industrial engineering, 154.

Yu, C.Y., Mou, S.D., Ji, Y.J., Xu, X. & Gu, X.J. (2018). A delayed product differentiation model for cloud manufacturing. Computers & industrial engineering, 117, 60-70.

Zhang, G., Chen, C.-H., Zheng, P. & Zhong, R.Y. (2020). An integrated framework for active discovery and optimal allocation of smart manufacturing services. Journal of cleaner production, 273.

Zhang, G., Zhang, Y., Xu, X. & Zhong, R.Y. (2018). An augmented Lagrangian coordination method for optimal allocation of cloud manufacturing services. Journal of manufacturing systems, 48, 122-133.

Zhang, Q. & Li, H. (2007). MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation, 11(6), 712-731.

Zhang, W., Xiao, J., Zhang, S., Lin, J. & Feng, R. (2021). A utility-aware multi-task scheduling method in cloud manufacturing using extended NSGA-II embedded with game theory. International journal of computer integrated manufacturing,

1-20.
Zhang, Y., Zhang, G., Liu, Y. & Hu, D. (2017). Research on services encapsulation and virtualization access model of machine for cloud manufacturing. Journal of intelligent manufacturing, 28(5), 1109-1123.
Zhang, Y., Zhang, G., Qu, T., Liu, Y. & Zhong, R.Y. (2017). Analytical target cascading for optimal configuration of cloud manufacturing services. Journal of cleaner production, 151, 330-343.
Zhang, Z.J., Zhang, Y.M., Lu, J.W., Gao, F. & Xiao, G. (2020). A novel complex manufacturing business process decomposition approach in cloud manufacturing. Computers & industrial engineering, 144.
Zheng, H., Feng, Y. & Tan, J. (2017). A Hybrid Energy-aware Resource Allocation Approach in Cloud Manufacturing Environment. IEEE Access, 1-1.
Zhou, J. & Yao, X. (2017). A hybrid approach combining modified artificial bee colony and cuckoo search algorithms for multi-objective cloud manufacturing service composition. International journal of production research, 55(16), 4765-4784.
Zitzler, E., Deb, K. & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary computation, 8(2), 173-195.