Manuscript Click here to download Manuscript Revised Manuscript.docx

Click here to view linked References

This is the Pre-Published Version.

This version of the article has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use(https://www.springernature.com/gp/open-research/policies/accepted-manuscript-terms), but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: http://dx.doi.org/10.1007/s10845-018-1417-8.

Hypernetwork-based manufacturing service scheduling for distributed and collaborative manufacturing operations towards smart manufacturing

Ying Cheng [a, b], Luning Bi [a], Fei Tao [a], Ping Ji [b]

[a] *School of Automation Science and Electrical Engineering, Beihang University (BUAA), Beijing, 100191, P. R. China*

[b] *Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong*

Abstract: In the future Smart Manufacturing (SM), both of sensor-based environment in shop floors and cloud-based environment among more and more enterprises are deployed gradually. Various distributed and separated manufacturing facilities are as collaborative cloud services, integrated and aggregated with their real-time information. It provides opportunities for the *distributed and collaborative manufacturing operations* across lots of distributed but networked enterprises on demand with enough flexibility. To this end, the scheduling problem and its result of those collaborative services for distributed manufacturing operations play an important role in improving manufacturing utilization and efficiency. In this paper, we put forward the hypernetwork-based models introducing the thought of graph coloring and an artificial bee colony algorithm based method for this scheduling problem. Three groups of experiments are carried out respectively to discuss therein different situations of distributed and collaborative manufacturing operations, i.e., in a private cloud, in a public cloud, and in a hybrid cloud. Some future studies with further consideration of collaboration equilibrium, dynamic control and data-based intelligence, are finally pointed out in the conclusion.

Keywords: Manufacturing service scheduling, distributed collaboration, smart manufacturing (SM), complex networks, graph coloring

1 Introduction

Along with the gradual applications of some new-generated information technologies (ITs) in manufacturing, pursuing Smart Manufacturing (SM) based on current digital situation becomes one of the worldwide common trends (Tao and Qi 2017a). Towards this trend and requirements of SM, the most typical sensor-based and cloud-based system architecture is explored widely (Tao et al. 2014a; Zhong et al. 2013). In this context, as the sensor-based environment is deployed gradually in shop floors, and the cloud-based environment increasingly covers more and more distributed but networked enterprises, manufacturing facilities (machines in one or different companies), although distributed in different places geographically, are connected together via internet. As well, the collection, utilization and sharing of real-time production data are achieved.

Therefore, it paves the way for a revolution of production, i.e., *distributed and collaborative manufacturing operations* across different enterprises, mainly with the following two improvements. On one hand, the sensor-based environment facilitates connection, interconnection as well as real-time information exchange between "M2M" in shop floors. Thus, it is able to collect massive real-time production data, as well as to support real-time information acquisition and feedback within an enterprise. On the other hand, the cloud-based environment helps treat any manufacturing facility or resource provided by each participated networked enterprise as a cloud service. By sharing real-time information, the distributed manufacturing facilities in the logical form of aggregated cloud services could be easily utilized or even highly shared by much more social users at anywhere and anytime (Tao et al. 2017b).

It means that, treating any available manufacturing facility or its functional capability as a manufacturing service and registered into the cloud service platform, each of the available manufacturing services would be selected or outsourced on demand. Moreover, for all of the participated networked enterprises in the cloud-based environment, their roles (e.g., as a service provider, or as a service consumer, or even as the both) are free (Tao et al. 2014a). Not only their services of available manufacturing facilities, but also their required tasks of various manufacturing operations, are allowed to be submitted to the platform. Therefore, it breaks the traditional habits of manufacturing operations only within an enterprise, and supports the distributed collaborations among lots of networked enterprises with enough flexibility but without the limitations of both scale and scope.

However, for the efficient distributed and collaborative manufacturing operations across different enterprises, the problem of manufacturing service scheduling, is a big challenge. Especially, taking different types of participators in the cloud into consideration, there exists three situations, i.e., a private cloud, a public cloud, and a hybrid cloud. Specific to each kind of situations, the objectives are analyzed and defined as follows: (a) *In a private cloud*, all of the participated enterprises are of the same group enterprise (GE), thus it mainly aims to achieve higher utilization efficiency of their own services. (b) *In a public cloud*, all of the participated small and medium-size enterprises (SMEs) collaborate with the purpose of higher average rate of services invocation. (c) *In a hybrid cloud*, multiple GEs and SMEs co-exist, so that it aims to remain the higher utilization efficiency of services in GEs in priority, and then to obtain the higher average rate of services invocation in the remainder SMEs.

With regard to above three situations, the problem of manufacturing service scheduling for such complicated distributed and collaborative manufacturing operations among different enterprises, is explored in this paper. Section 2 provides a brief investigation on the related work. The hypernetwork and graph coloring based models are proposed in Section 3 to describe the manufacturing service scheduling problem. By the artificial bee colony algorithm based method which is illustrated in Section 4, three groups of experiments and a discussion are carried out in Section 5. Section 6 concludes the full paper.

2 Related work

As indicated, the scale and scope of distributed manufacturing operations undergo change because of more applications of cloud technology and pervasive access of sensors into manufacturing systems. It makes a big difference on the scheduling for distributed manufacturing operations, which is divided into the following two stages.

2.1 For distributed manufacturing operations within an enterprise

The usual *distributed manufacturing operations* (Shen et al. 2006) are mainly limited for discrete manufacturing processes within one enterprise. This scheduling problem serves as a general advanced planning and scheduling (APS) problem, which mainly owns the following two kinds of definitions. One is, it serves as a computer program, using the mathematical optimization or simulation methods to determine the discrete production planning processes, while the scope of the solving methods is limited. The other one is, it serves as a finite scheduling, using the advanced mathematical algorithms to perform optimization (Hvolby & Setger-Jensen 2010).

In most of the existing studies, the problems are described as the integrated or combined problems. Many researchers decompose the problems into sub-problems, including master production schedule (MPS), material requirements planning (MRP), and scheduling for jobs dispatched to machines. The decomposed sub-problems are usually taken as a linear or mixed integer programming model (Steger-Jensen et al. 2011; Moo et al. 2004), constraint programming model (Peng et al. 2014), discrete simulation models, or allocate resources or tasks hierarchically based on flow and path (Tarantilils et al. 2008). However, they are strictly dominated by the mathematical models and solutions by various kinds of heuristic algorithms (Ozturk & Ornek 2016; Zhong et al. 2015). For solving these sub-problems, different methods, such as constraint-based approaches, simulation (Chen et al. 2013), heuristics (Kung & Chern 2009) or meta-heuristics (Goren et al. 2010), are usually carried out.

However, the mathematical models are just built for finite scheduling with the actual shop floor conditions (Rafiei & Rabbani 2012), some of experimentation with sample data sets were provided. Focus on the scheduled objects, these studies mainly discuss and obtain the final plans and schemes for finite scale and static environment, e.g., usually few separate machines and their distributed manufacturing operations were studied, but ignore the interrelations among manufacturing resources or facilities.

2.2 For distributed and collaborative manufacturing operations across different enterprises

The distributed manufacturing operations are revealing much more collaborations among different enterprises after introducing cloud services (Mishra et al. 2016). The service-oriented scheduling problems for such *distributed and collaborative manufacturing operations* across lots of enterprises, attract more and more researchers' attention.

On one hand, lots of scheduling-related issues on manufacturing services are carried out but without consistency, e.g., service selection and composition (Tao et al. 2013), service supply-demand matching (SDM) (Cheng et al. 2017), service scheduling (Tao et al. 2014b), service allocation, and so on. Most of them focus on the multi-criteria quality of service (QoS)-based evaluation, but few considers the inter-time-series duration when a facility is as a service for the required manufacturing operation.

On the other hand, many studies focus on the collaboration or coordination (He et al. 2016) of the upper level of enterprises in the cloud rather than the underlying level of production. In which, the topological models of complex networks or graph (Wang et al. 2004) provide a promising approach to build scalable models so as to describe complex relationships among lots of heterogeneous objects. For example, hypernetwork model based optimization of integrated e-supply chain (Dotoli et al. 2006), hypernetwork-based manufacturing service SDM (Cheng et al. 2017), demand and capacity sharing in supply network collaboration (Moghaddam & Nof 2014), distributed production networks (Lo Nigro et al. 2003; Lv & Lin 2017), and so on. Furthermore, there are studies on robustness and vulnerability analysis of collaborative production networked organizations (Camarinha-Matos 2009), clustering and modularity analysis of enterprise relationship network (Ding et al. 2016). Towards the upcoming environment of SM, both enterprise collaborations and the specific collaborative production activities are determined by the underlying scheduling solution of distributed manufacturing operations.

2.3 A brief summary

Towards the future SM environment, manufacturing operations are more characterized with process complexity, cross-enterprise collaboration, uncertainty, and so on. More and more researches consider using and sharing the collected real-time production data and data mining techniques. However, there still are some main limitations of the existing work summarized as follows:

- Some of the theories claimed to have supporting evidence from experimentation with sample data sets. Despite advances in ITs and computer modeling techniques, there still exists incomplete, ambiguous, inconsistent and untimely data making the planning and scheduling unrealistic.
- The mathematical models are mainly for finite scheduling. Few separate machines and their distributed manufacturing operations were studied, but the interrelations among different manufacturing facilities and their operation processes were ignored.
- The explored collaboration patterns or mechanisms are limited within one enterprise itself, and not suit for distributed collaboration among enterprises towards the much more social environment of SM.

Therefore, it is still hard to describe and solve this manufacturing service scheduling problem for such distributed and collaborative manufacturing operations across different enterprises. With consideration of large scale of real-time data and large scope of collaborative enterprises, in order to discuss and provide enterprises with some theoretical reference to carry out their distributed and collaborative activities efficiently and agilely, the contributions of this paper are listed as follows:

- *For addressing this cross-enterprise and inter-time-series manufacturing service scheduling problem, the hypernetwork and graph coloring based models are proposed considering some of real-time data.*

- *For analyzing the derived service-based distributed collaborations among enterprises, different situations including a private cloud, a public cloud and a hybrid cloud, are discussed respectively based on an artificial bee colony algorithm based method.*

3 Hypernetwork and graph coloring based models of manufacturing service scheduling

Based on the sensor-based and cloud-based environment towards SM, it involves multiple services belonging to multiple enterprises in the platform. A set of tasks submitted by all enterprises need to be appropriately assigned to various services. Assumed that, tasks of manufacturing operations are primitive or decomposed into primitive sub-tasks, each task need to invoke a service to accomplish the corresponding manufacturing operation during a certain duration of its required time interval. In a word, this cross-enterprise and inter-time-series manufacturing service scheduling problem is to answer the questions that, which manufacturing service is selected for each task and what time duration is it in service? It aims to dispatch an appropriate service for the sustainable manufacturing operation during the required time interval of each task. When a task arrivals, the specific selected service would be occupied, and then released after the manufacturing operation is finished.

Based on the previous proposed modeling method of manufacturing service SDM hypernetwork which is donated as $Matching\_Net = \{\boldsymbol{V}(S,T), \boldsymbol{E}(E^S, E^T, E^{S\_T})\}$ (Cheng et al. 2017), the models are supplemented with the collected real-time data specific to the manufacturing service scheduling problem to be explored in this paper. The detailed supplemented models are illustrated with formulas (1) - (11) in Table 1. In order to simplify the decision variables, the concept of time slices is introduced to depict manufacturing operations' duration. So that a task conflict graph for each time slice could be extracted from $Matching\_Net$. In which, if services are assigned with different colors, the manufacturing service scheduling problem could be also addressed as a coloring problem of each task conflict graph. In order to further describe the problem, some related notations of parameters are listed as follows.

S: set of manufacturing service notes.

s_k: manufacturing service node.

T: set of manufacturing task notes.

t_i: manufacturing task node.

EN: set of manufacturing enterprise notes.

en_p: manufacturing enterprise note.

E^S: set of edges among service nodes.

E^T: set of edges among task nodes.

E^{EN}: set of edges among enterprise nodes.

E^{S_T}: set of matchable hyper-edges between services and tasks.

NoS: amount of manufacturing services in the system.

NoT: amount of manufacturing tasks in the system.

NoE: amount of the participated enterprises in the system.

$s_k\_provider$: the enterprise that service s_k belongs to.

s_k_type: type of service s_k.

s_k_state: state of service s_k based on the real-time data.

$t_i\_consumer$: the enterprise that task t_i belongs to.

$t_i\_starttime$: start-time of the required time interval of task t_i.

$t_i\_endtime$: end-time of the required time interval of task t_i.

t_i_state: state of task t_i based on the real-time data.

en_p_type: type of enterprise en_p.

en_p_S: service set provided by enterprise en_p.

en_p_T: task set submitted by enterprise en_p.

e^S_{kl}: edge between service nodes s_k and s_l.

e^T_{ij}: edge between task nodes t_i and t_j.

e^{EN}_{pq}: edge between enterprise nodes en_p and en_q.

$e^{S\_T}_{ki}$: hyper-edge between service node s_k and task node t_i.

G_t: task conflict graph in the t_{th} time slice.

$Conflict_t$: set of tasks which are with time conflict.

E^C_t: set of conflict relationships between any two tasks.

$d^C_t(t_i)$: degree of task node t_i in G_t.

$N^C_t(t_i)$: set of neighbor tasks of t_i in G_t.

$P_t(t_i)$: matrix of task notes coloring, i.e., the identification whether t_i need invoke some service in the t_{th} time slice.

$M^{S\_T}(t_i)$: set of available services could be selected for t_i in $Matching\_Net$.

$d^{S\_T}(t_i)$: hyper-edges oriented degree of t_i in $Matching\_Net$, i.e., the number of available services could be selected for t_i.

$M^{S\_T}(s_k)$: set of tasks could be colored by s_k in $Matching\_Net$.

$d^{S\_T}(s_k)$: hyper-edges oriented degree of s_k in $Matching\_Net$, i.e., the number of tasks could be colored by s_k.

Table 1 The supplemented scalable models of manufacturing service SDM hypernetwork

Models	
$Matching\_Net = \{\boldsymbol{V}(S,T,EN),\boldsymbol{E}(E^S,E^T,E^{EN},E^{S\_T})\}$	(1)
$S=\{s_1,s_2,\dots,s_k,\dots,s_{NoS}\}$	(2)
$s_k=\{s_{k}\_provider\in EN,s_{k}\_type,s_{k}\_state,\dots\}$	(3)
$T=\{t_1,t_2,\dots,t_i,\dots,t_{NoT}\}$	(4)
$t_i = \{t_{i}\_consumer \in EN,t_{i}\_starttime,t_{i}\_endtime,t_{i}\_state,\dots\}$	(5)
$EN=\{en_1,en_2,\dots,en_p,\dots,en_{NoE}\}$	(6)
$en_p=\{en_{p}\_type,en_{p}\_S\subset S,en_{p}\_T\subset T,\dots\}$	(7)
$E^S=\{e_{kl}^S \mid k,l=1,2,\dots,NoS\}$	(8)
$E^T=\{e_{ij}^T \mid i,j=1,2,\dots,NoT\}$	(9)
$E^{EN}=\{e_{pq}^{EN} \mid p,q=1,2,\dots,NoE\}$	(10)
$E^{S\_T}=\{e_{ki}^{S\_T} \mid k=1,2,\dots,NoS;i=1,2,\dots,NoT\}$	(11)

3.1 Time slice division

In order to identify conflicts of the required time intervals among different tasks, the concept of time slices is introduced. So that, the overall time period of all of manufacturing operations, which covers from the earliest start-time to the latest end-time of all of tasks, is divided into some equal time slices. Assumed that the minimal break is Δt if a service would be dispatched in order by any two tasks, the unitage of time slice could be set as $k\Delta t$ (k is a natural number). As a result, the overall time period is as $[\min(\{t_{i}\_starttime\}),\max(\{t_{i}\_endtime\})]$. The number of the equally divided time slices, named N, could be calculated according to formula (12).

$$N=\left\lceil\frac{\max(\{t_{i}\_endtime\})-\min(\{t_{i}\_starttime\})}{k\Delta t}\right\rceil \tag{12}$$

In formula (12), the symbol '⌈ ⌉' stands for the top integral function. As t is the counting parameter, the t_{th} time slice is described as the time interval $[\min(\{t_{i}\_starttime\})+(t-1)\,k\Delta t,\min(\{t_{i}\_starttime\})+t\,k\Delta t]$.

3.2 Task conflict graph description

The task conflict graph in the t_{th} time slice G_t, is determined by conflict identification between each two tasks. Based on the models in Table 1, if the t_{th} time slice is occupied by the required time interval of task t_i, the node of task t_i is included in $Conflict_t$, i.e., $t_i\in Conflict_t$, and vice versa. The conflict identification is judged by formula (13).

$$\begin{aligned}&t_{i}\_endtime\le\min(\{t_{i}\_starttime\})+(t-1)\,k\Delta t\\&\text{or }\min(\{t_{i}\_starttime\})+t\,k\Delta t\le t_{i}\_starttime\end{aligned} \tag{13}$$

Obviously, if two tasks both exist in $Conflict_t$, the edge of conflict relationship between these two tasks exists, i.e., if $t_i\in Conflict_t$ and $t_j\in Conflict_t$, then $e_{t\ ij}^{C}=1$. Therefore, G_t is a complete graph. After the judgement of conflict identification, the model of G_t is composited as formula (14).

$$G_t=(Conflict_t,E_t^C) \tag{14}$$

3.3 Features extraction of graph coloring

Treated the available services as different colors, both for each task to be assigned or colored and for each service or color to be selected and dispatched, their corresponding features are extracted and listed as formulas (15) - (21) in Table 2.

Table 2 The graph coloring related features extraction for task t_i and service s_k

Features	Rules or description	
For t_i	If $t_i\in Conflict_t$, then $d_t^C(t_i)=\lvert Conflict_t\rvert-1$, else $d_t^C(t_i)=0$	(15)
	$N_t^C(t_i)=\{t_* \mid e_{t\ i*}^{C}=1\}$ i.e., $N_t^C(t_i)=Conflict_t-\{t_i\}$	(16)
	If $t_i\in Conflict_t$, then $P_t(t_i)=1$, else $P_t(t_i)=0$	(17)
	$M^{S\_T}(t_i)=\{s_k \mid e_{ki}^{S\_T}=1,k=1,2,\dots,NoS\}$	(18)
	$d^{S\_T}(t_i)=\lvert M^{S\_T}(t_i)\rvert$	(19)
For s_k	$M^{S\_T}(s_k)=\{t_i \mid e_{ki}^{S\_T}=1,i=1,2,\dots,NoT\}$	(20)
	$d^{S\_T}(s_k)=\lvert M^{S\_T}(s_k)\rvert$	(21)

3.4 Optimization models of manufacturing service scheduling

3.4.1 Decision variables

Define the decision variable of the t_{th} time slice is $a^t{}_{ki}$, so the decision result of the t_{th} time slice is as $A^t = \{a^t{}_{ki}\}_{NoS\times NoT}$, then the scheduling result of this problem covering the overall time period is collectedly denoted as $A = \{A^1, A^2, \dots, A^t, \dots, A^N\}$. When $a^t{}_{ki} = 1$, it means that the service s_k is dispatched to the task t_i in the t_{th} time slice. Otherwise, $a^t{}_{ki} = 0$.

3.4.2 Objectives

(1) In a private cloud

In a private cloud, all of the participated enterprises are of the same group enterprise (GE). The constituent enterprises are devoting to remaining the utilization efficiency of services of the whole GE. In other words, the objective is to use less services as possible for the definite requirement of tasks.

As the models of $Matching\_Net$ illustrated in Table 1, the set of the constituent enterprises of the GE is $EN = \{en_1, en_2, \dots, en_p, \dots, en_{NoE}\}$, and the set of services of the whole GE is $S = \{s_1, s_2, \dots, s_k, \dots, s_{NoS}\}$. Once the service s_k is selected, no matter in any time slice, it defines $a_k = 1$. Conversely, it assigns $a_k = 0$ to indicate that the service s_k is never invoked all the way. Therefore, it results in $a_k = 1$ if $\sum_{t=1}^{N}\sum_{i=1}^{NoT} a^t{}_{ki} > 0$, else it makes $a_k = 0$ if $\sum_{t=1}^{N}\sum_{i=1}^{NoT} a^t{}_{ki} = 0$. The objective in a private cloud is formulated as formula (22).

$$min(\sum_{k=1}^{NoS} a_k) \tag{22}$$

(2) In a public cloud

Regarding to a public cloud, it is composed of a number of SMEs with their collaboration. As the set of services provided by the enterprise en_p is $en_p\_S = \left\{s_{k_p} \middle| k_p \in 1,2,\dots,NoS\right\}$, it is known that $\bigcup_{p=1}^{NoE} en_p\_S = S$. Similarly, so long as the specific service s_{k_p} of en_p is selected (i.e., $\sum_{t=1}^{N}\sum_{i=1}^{NoT} a^t{}_{k_p i} > 0$), it defines $a_{k_p} = 1$. On the contrary, it defines $a_{k_p} = 0$ when $\sum_{t=1}^{N}\sum_{i=1}^{NoT} a^t{}_{k_p i} = 0$. Therefore, the rate of services invocation of en_p is modeled as formula (23). Considering all of the existing SMEs in a public cloud, the objective of the higher average rate of services invocation is described as formula (24).

$$\textit{The rate of services invocation of } en_p = \frac{\sum_{k_p}^{|en_p\_S|} a_{k_p}}{|en_p\_S|} \tag{23}$$

$$max(\frac{\sum_{p=1}^{NoE}\left(\left(\sum_{k_p}^{|en_p\_S|} a_{k_p}\right)/|en_p\_S|\right)}{NoE}) \tag{24}$$

(3) In a hybrid cloud

In a hybrid cloud, it is assumed that, besides the tasks of GEs are assigned by their own services, the constituent enterprises of GEs could also provide some redundant services for the tasks submitted by SMEs. But the services of SMEs could not be selected by the constituent enterprises of GEs. It defines that the quantity of GE is NoG, and the quantity of constituent enterprises of the m_{th} GE is g_m. The set of constituent enterprises of the m_{th} GE is as $G\_en_m = \left\{en_{p_m} \middle| p_m \in 1,2,\dots,NoE\right\}$. In addition, the set of services provided by the constituent enterprise en_{p_m} is represented as $en_{p_m}\_S = \left\{s_{k_{p_m}} \middle| k_{p_m} \in 1,2,\dots,NoS\right\}$. Naturally, $|G\_en_m| = g_m$, it is inferred that the remainder set of SMEs is formulated as $SM\_en = EN - \bigcup_{m=1}^{NoG} G\_en_m$, and the quantity of SMEs is calculated as $|SM\_en| = NoE - \sum_{m=1}^{NoG} g_m$.

For each constituent enterprise of G_en_m in the hybrid cloud (i.e., $en_p \in G\_en_m$), the provided service and required task of en_p are marked as s_{k_p} and t_{i_p} (i.e., $s_{k_p} \in en_p\_S$ and $t_{i_p} \in en_p\_T$). Then, the quantity of the dispatched services is counted as $\sum_{k_p}^{|en_p\_S|} a_{k_p}$. Therefore, considering all of the constituent enterprises in G_en_m, the set of tasks is integrated as $\bigcup_{\substack{p \\ en_p \in G\_en_m}}^{|G\_en_m|} en_p\_T$. The primary internal objective of each GE is to maximize the utilization efficiency of its own services at first, shown as formula (25).

$$min\left(\sum_{\substack{p \\ en_p \in G\_en_m}}^{|G\_en_m|} \sum_{\substack{k_p \\ s_{k_p} \in en_p\_S}}^{|en_p\_S|} a_{k_p}\right) \tag{25}$$

On the other hand, if the enterprise en_p is one of SMEs in the hybrid cloud (i.e., $en_p \in SM\_en$), the set of tasks submitted by all of SMEs is integrated as $\bigcup_{\substack{p \\ en_p \in SM\_en}}^{|SM\_en|} en_p\_T$. Based on formula (23), it also aims to make all of SMEs achieve their maximal average rate of services invocation, the corresponding objective is as formula (26).

$$max\left(\sum_{\substack{p \\ en_p\in SM\_en}}^{|SM\_en|} \frac{\sum_{k_p}^{|en_p\_S|} a_{k_p}}{|en_p\_S|}\right)\Big/|SM\_en| \qquad (26)$$

3.4.3 Constrains

The following restrictions need to be considered in this scheduling problem: (a) It must be functionally matchable between each task and its corresponding dispatched service. (b) Each task is assigned to only one service. Once its operation starts, it could not be broken off. (c) At any time, each service could be dispatched for a task at most, that is, it is non-reusable. (d) For each dispatched service, there exists a gap between the end-time of previous operation to the start-time of the next operation.

Considering each service as a color, and treating the scheduling problem as a coloring problem of the task conflict graph in each time slice, the constrains corresponding to above four restrictions are as follows:

Constrain 1: Not all services are available for each task. It needs to select a matchable service for a task from the set of hyper-edges E^{S_T}, shown as formula (27).

$$A^t = \{a^t{}_{ki}\}_{NoS\times NoT} \subseteq E^{S\_T} \qquad (27)$$

Constrain 2: Each task could be colored by only one service, and the period to be colored of each task must be in accordance with corresponding required time interval of manufacturing operation, shown as formula (28).

$$\sum_{k=1}^{NoS} a^t{}_{ki} = P_t(t_i) \qquad (28)$$

Constrain 3: Each service/color could be assigned for at most one task in each time slice, shown as formula (29).

$$\sum_{i=1}^{NoT} a^t{}_{ki} = 1 \qquad (29)$$

Constrain 4: It is necessary to make a break (marked as Δ_k) for each service to execute operations of different tasks in order, described as formula (30).

$$\begin{array}{l}\text{If } a^t{}_{ki_1} = 1 \ \& \ a^r{}_{ki_2} = 1, \text{where } t < r \ \& \ i_1 \neq i_2, \\ \quad \text{then } t_{i_2}\_starttime - t_{i_1}\_endtime \geq \Delta_k\end{array} \qquad (30)$$

4 An artificial bee colony algorithm based method

The ABC algorithm is proposed by Karaboga to solve the algebra problems (Liu et al. 2015). One of its advantages is that it does not require convergence analysis of the models. Meantime, it is beneficial for global optimization. The optimal result can be obtained just by comparing the fitness of solutions. In essence, the manufacturing service scheduling problem is a combinatorial explosion problem. There are so many variables in the problem models. Some algorithms are usually easy to run into a local optimum. So the ABC algorithm is selected for this scheduling problem and carried out based on the proposed hypernetwork models and the derived coloring models of each task conflict graph.

4.1 Algorithm structure

There are four basic elements, i.e., food source, employed bees, onlookers and scouts, included in the ABC algorithm. By the cooperation among employed bees, onlookers and scouts, the optimal solution would be found quickly (Liu et al. 2015). Corresponding to the problem discussed in this paper, the main definitions are as follows.

- ***Food sources:*** Each food source represents a service scheduling solution, and the amount of nectar represents the fitness of solution (i.e., the aforementioned three objectives), such as formulas (22), (24), as well as the combination of (25) and (26).
- ***Employed bees:*** Each employed bee corresponds to a food source (i.e. service scheduling solution), and stores the position information (i.e. the selected service/color of each task).
- ***Onlookers:*** They can observe the employed bees to get the information about food sources. If the amount of nectar is huge, onlookers can be transformed into employed bees.
- ***Scouts:*** Scouts are responsible to randomly search for new food sources.

In order to improve the convergence speed of the ABC algorithm, the strategy of tabu search is introduced. If the trial times of a certain position exceeds the limit, the position, may be a local optimal point, is added into the tabu list. So that it will be avoided in the next search. The structure of the ABC algorithm with the strategy of tabu search, as illustrated in Fig. 1, is described as follows.

Step 1: Initialization. Adopt the encoding method to generate some solutions of manufacturing service scheduling. Set the length of the tabu list and the upper limit of the trail times.

Step 2: Employed bees mode. Change the selected services of certain tasks in the neighborhood of the solution to generate the new solution.

Step 3: Update. Calculate the fitness of the new solution using the proposed self-validated decoding method. If the fitness is better, replace the old solution with the new one; otherwise, the trail times plus one.

Step 4: Onlookers mode. Use roulette to select the solution according to the fitness. The better the fitness is, the more possible the solution is to be selected.

Step 5: Update.

Step 6: If the trail times exceeds the limit, go to the next step; otherwise go to *Step 8*.

Step 7: Scouts mode. Randomly change the selected service of some task. Calculate the Euclidean distance between the positions in the tabu list and the new solution. If the distance is below the predefined value, repeat the above operation. Then set the trail times as 0.

Step 8: If the number of iterations exceeds the max value, output the optimal result; otherwise, go to *Step 2*.

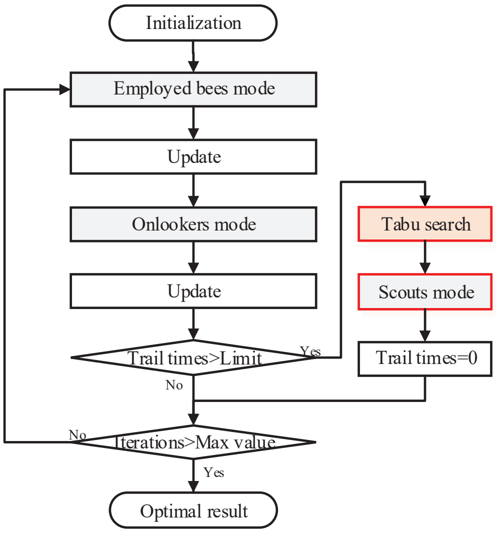


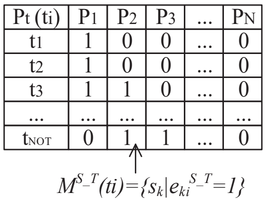
Fig. 1 Structure of the ABC algorithm with tabu search strategy

4.2 Encoding and decoding of graph coloring based solutions

4.2.1 Real encoding

As $a^t{}_{ki}$ is a three-dimensional decision variable, it uses a two-dimensional matrix of task notes coloring to simplify and represent the solution, which is denoted as $P_t(t_i)$ as shown in Fig. 2. The row represents from the 1st to Nth time slices, and the column represents from the 1st to NoTth task nodes. When a cell of $P_t(t_i)$ is assigned as 1, it means that the task node t_i needs to invoke a service in the tth time slice. Therefore, the decision problem is to select an appropriate color/service for each cell which is assigned as 1, and the appropriate service for each cell should be selected from $M^{S\_T}(t_i)$.

There are two steps for real encoding, including *matrix generation of task notes coloring* and *code generation*. (a) The matrix of task notes coloring is directly generated according to the task conflict graph of each time slice. (b) As to the code generation, firstly, read all cells that are assigned with 1 in $P_t(t_i)$, and record the total number, that is also the number of decision variables. It is worth noting that, the cells in any line that are continuously assigned with 1 should be regarded as one common decision variable, so as to avoid generating a lot of infeasible solutions by assigning different values to those consecutive cells. Secondly, specific to each cell that is assigned with '1' in $P_t(t_i)$, the value of each corresponding decision variable is generated randomly between 0 and 1. So that, a string of code representing a specific solution of manufacturing service scheduling can be obtained.



$P_t(t_i)$	P_1	P_2	P_3	...	P_N
t_1	1	0	0	...	0
t_2	1	0	0	...	0
t_3	1	1	0	...	0
...	...	...	...	...	...
t_{NOT}	0	1	1	...	0

$M^{S\_T}(ti)=\{s_k|e_{ki}{}^{S\_T}=1\}$

Fig. 2 The matrix of task notes coloring for decision variables simplification

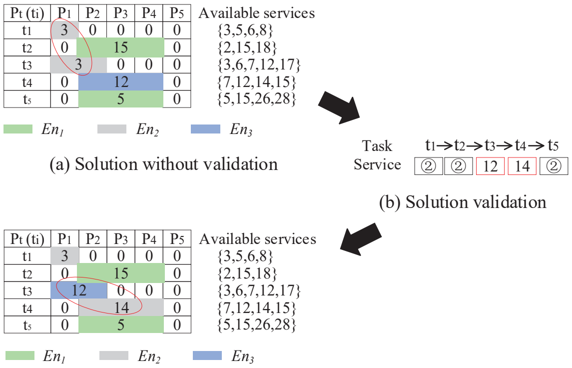
4.2.2 Self-validated decoding

For settling complex constraints, the traditional method is to examine the feasibility in the encoding stage. It costs a lot of time and hinders the algorithm running. Thus, a self-validated decoding method is proposed to solve this problem, which is divided into the following two steps.

The first step of solution generation by decoding, is to multiply the value of each variable with the total number of available services for each task (i.e., the total number of matchable hyper-edges of each task in $M^{S\_T}(t_i)$). So the index of selected service can be obtained by rounding the multiplied result. For example, if the value of the variable is 0.64 and the total number of available services is 7, the selected service is the 4th service (i.e., 0.64*7=4.48). By the first step, a solution of manufacturing service scheduling is obtained. However, this obtained solution cannot be executed immediately because there may exist some conflict with the constraint as formula (29) shows.

The second step of solution validation, is to check the selected services of tasks one by one. If the selected service has been occupied by the previously assigned task at a certain period of time, then randomly reselect another service from $M^{S\_T}(t_i)$. It should be noted that, there may exist an extreme case that all of available services specific to a task are occupied by the previous tasks. In this case, the solution should be regenerated. However, considering diversity of available services and discrepancy of their different time periods to be invoked, it is a small probability event.

An example is illustrated in Fig. 3. In which, Fig. 3(a) shows the obtained solution after the first step of decoding without validation. For validating the obtained solution, it is found that the selected service of t_3 (i.e., s_3) has been already assigned to t_1 when it turns to check t_3. So that its selected service has to be changed to the others in $M^{S\_T}(t_3)$, e.g., s_{12}, and then check t_4 in succession and change to select s_{14} for it, as shown in Fig. 3(b). Importantly, when checking each task, its following tasks are not considered. As a result, the final solution shown in Fig. 3(c) is brought out by solution validation.



Pt (ti)	P1	P2	P3	P4	P5	Available services
t1	3	0	0	0	0	{3,5,6,8}
t2	0	15			0	{2,15,18}
t3	3		0	0	0	{3,6,7,12,17}
t4	0	12			0	{7,12,14,15}
t5	0	5			0	{5,15,26,28}

(a) Solution without validation

(b) Solution validation

Pt (ti)	P1	P2	P3	P4	P5	Available services
t1	3	0	0	0	0	{3,5,6,8}
t2	0	15			0	{2,15,18}
t3	12		0	0	0	{3,6,7,12,17}
t4	0	14			0	{7,12,14,15}
t5	0	5			0	{5,15,26,28}

(c) Final solution after validation

Fig. 3 An example for self-validated decoding

4.2.3 Validation of the code

Validation of the code, is the step of anti-coding. The original solution is validated after decoding. But there are some differences between the original solution without validation and the final one after validation. Strictly speaking, the final solution is not corresponding to the code because of the random behaviors in the self-validated decoding operations. It has great influence on the evolutionary direction of the algorithm. Thus, by generating the correct code based on anti-coding operation according to the final solution is necessary.

5 Case study

In this section, a case study is provided to validate the proposed hypernetwork & graph coloring based models and the ABC algorithm based method. Three groups of experiments are carried out to analyze the collaboration resulted by manufacturing service scheduling.

As indicated, this case is modeled as an initial hypernetwork, in which totally 50 service nodes and 30 task nodes distributed to 10 enterprises are included. The overall time period in the problem is divided into 30 time slices. The specific correspondence among each enterprise, its available services and submitted tasks are listed in Table 3. Therefore, the hypernetwork model of this case could be generated, in which the sparse matrix of hyper-edges revealing the corresponding available services for each task is given in detail in Fig. 4. Moreover, the required time intervals of those 30 tasks, namely the Gantt chart of the task nodes to be colored, are shown in Fig. 5.

Table 3 Setting on enterprises, services and tasks

Enterprises ($NoE = 10$)	**Services** ($NoS = 50$)	**Tasks** ($NoT = 30$)
en_1	$s_1 \sim s_3$	$t_1 \sim t_2$
en_2	$s_4 \sim s_8$	$t_3 \sim t_4$
en_3	$s_9 \sim s_{10}$	$t_5 \sim t_6$
en_4	$s_{11} \sim s_{14}$	$t_7 \sim t_8$
en_5	$s_{15} \sim s_{20}$	$t_9 \sim t_{10}$
en_6	$s_{21} \sim s_{24}$	$t_{11} \sim t_{14}$
en_7	$s_{25} \sim s_{32}$	$t_{15} \sim t_{18}$
en_8	$s_{33} \sim s_{38}$	$t_{19} \sim t_{22}$
en_9	$s_{39} \sim s_{46}$	$t_{23} \sim t_{26}$
en_{10}	$s_{47} \sim s_{50}$	$t_{27} \sim t_{30}$

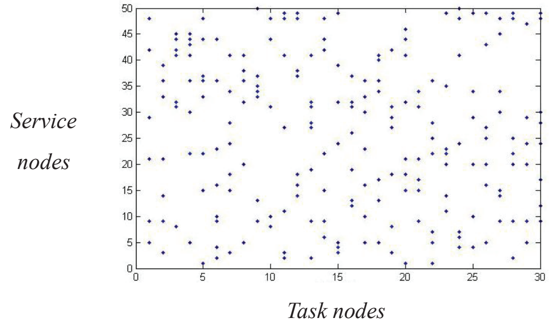


Fig. 4 The sparse matrix of hyper-edges of the case

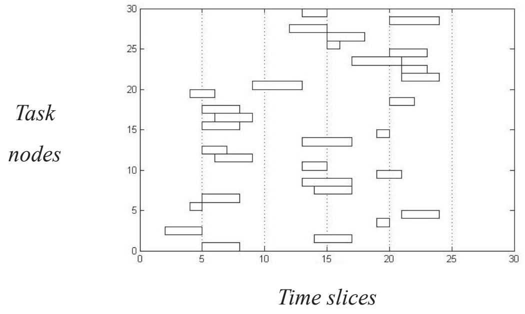
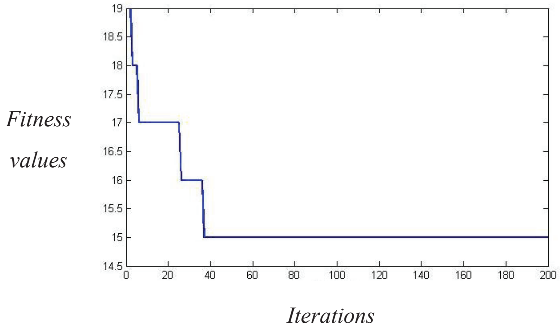


Fig. 5 Gantt chart of the task nodes to be colored with their required time intervals

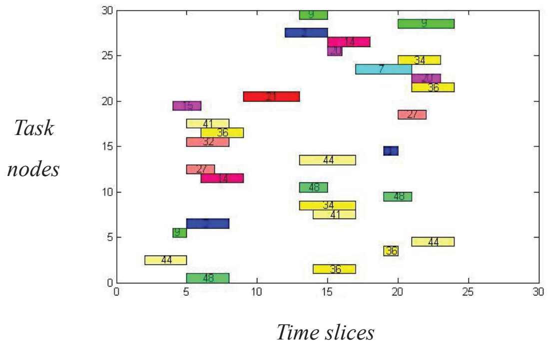
The experiment environment information are listed as follows: Intel® Core^{TM} i5-CPU 3230M @ 2.60GHz, dual-core, 12GB memory, and Windows 10 of 64 bits.

5.1 Manufacturing service scheduling in a private cloud

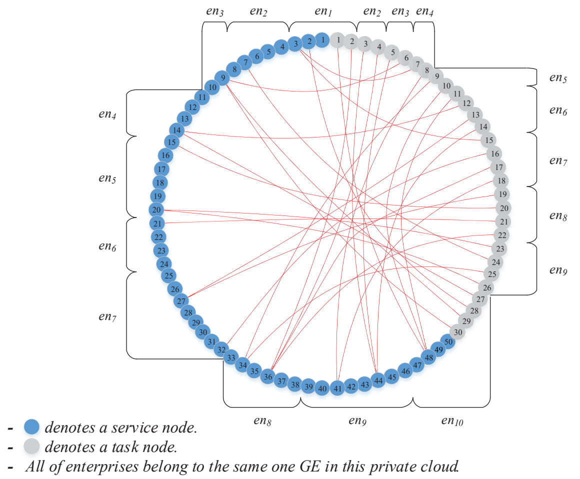
For the problem in a private cloud, the objective for achieving the maximal utilization efficiency of services also means invoking the least amount of services. Based on the ABC algorithm based method, the obtained solution is carried out in Fig. 6. In which, Fig. 6(a) is the running result and Fig. 6(b) shows the result of task nodes coloring. Furthermore, based on the result of task nodes coloring, the corresponding situation of collaboration among those 10 enterprises is illustrated in Fig. 6(c).



(a) Running result



(b) Gantt chart of the task nodes coloring result

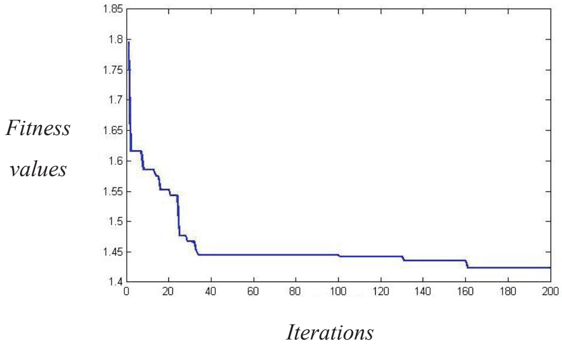


(c) The resulted enterprises collaboration in hypernetwork

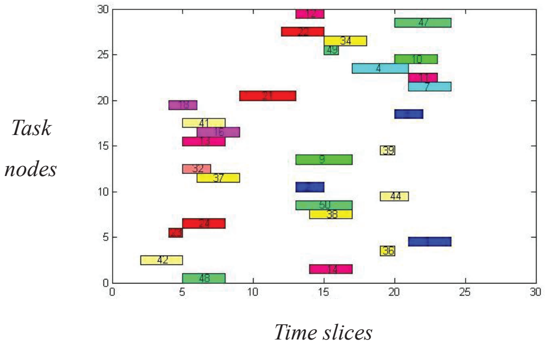
Fig. 6 The scheduling solution in a private cloud

Known from the running result, at least 15 services are selected for those 30 tasks and meeting their different required time intervals. For example, s_{44} is assigned respectively for t_3, t_{14} and t_5 in the periods of [3, 5], [14, 17] and [21, 24], and s_{48} conducts manufacturing operations in the periods of [5, 8], [14, 15] and [19, 21] for t_1, t_{11} and t_{10} in order.

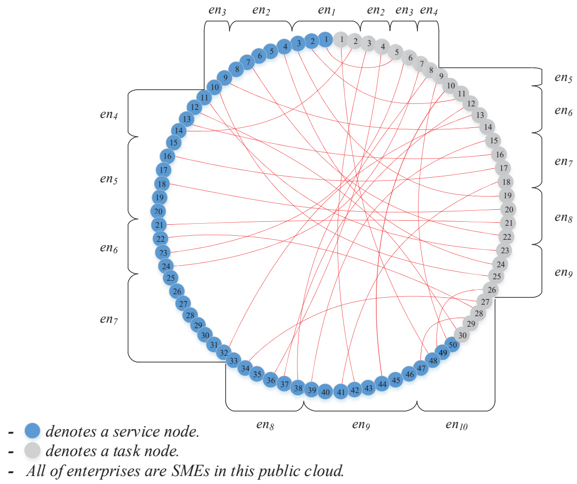
5.2 Manufacturing service scheduling in a public cloud



(a) Running result



(b) Gantt chart of the task nodes coloring result



(c) The resulted enterprises collaboration in hypernetwork

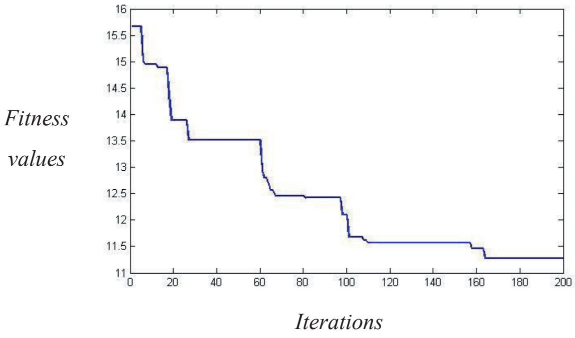
Fig. 7 The scheduling solution in a public cloud

As to the problem in a public cloud, all of enterprises are SMEs rather than belong to the same one GE in the private cloud. In this context, it aims to make the average rate of services invocation as higher as possible.

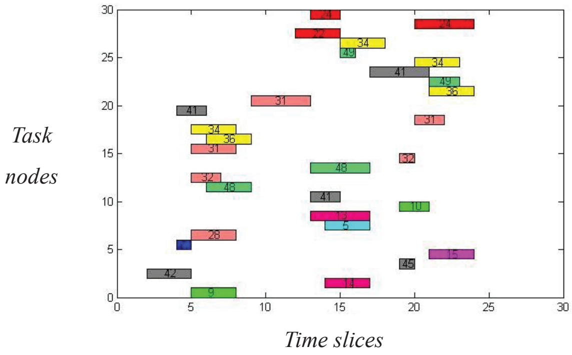
Take the reciprocal of the average rate of services invocation formulated in formula (24) as the fitness of the algorithm. As the running result shown in Fig. 7(a), the convergent fitness value is 1.4235, so the maximal average rate of services invocation of all enterprises is 0.7025. In addition, the result of task nodes coloring in this context is given in Fig. 7(b), the derived enterprises collaboration situation according to the task nodes coloring result is illustrated in Fig. 7(c).

5.3 Manufacturing service scheduling in a hybrid cloud

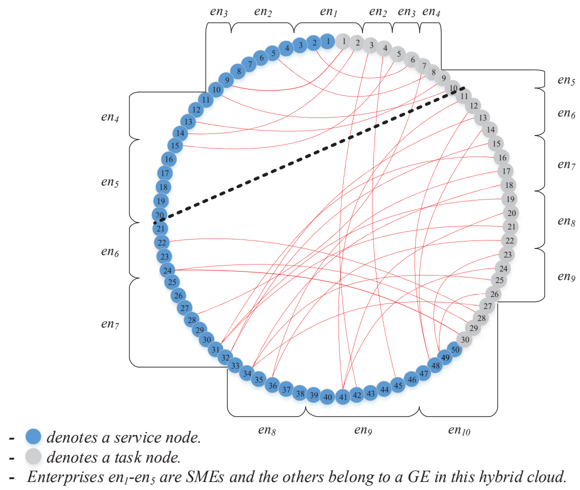
In the context of hybrid cloud, the constituent enterprises of a GE can also act as general SMEs to provide their remainder services for those tasks submitted by other SMEs. Therefore, the manufacturing service scheduling problem in this situation is to achieve a higher average rate of services invocation among all of SMEs meanwhile guarantee the utilization efficiency of services in GEs.



(a) Running result



(b) Gantt chart of the task nodes coloring result



(c) The resulted enterprises collaboration in hypernetwork

Fig. 8 The scheduling solution in a hybrid cloud

In this group of experiment, the role setting of different enterprises are supplemented, i.e., $en_1 \sim en_5$ are defined as SMEs and the reminder of enterprises are defined as those five constituent enterprises of the same one GE. Combining formulas (25) and (26) as the objective, both of running result of the algorithm and the corresponding task nodes coloring result are obtained as illustrated in Fig. 8 (a) and (b). Accordingly, based on the coloring result of each task node, the entire collaboration among all of enterprises is also given in Fig. 8(c).

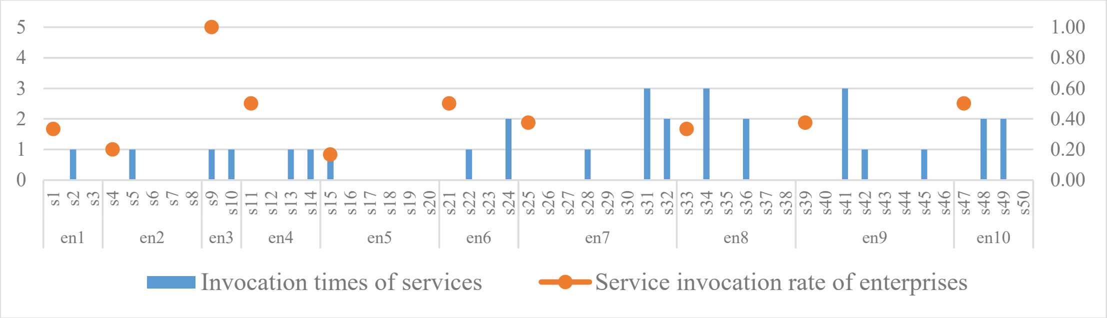
Different from the above two groups of experiments, the obtained manufacturing service scheduling solution shows that all manufacturing tasks in this GE are assigned by their own services, and the dispatched services for all manufacturing tasks submitted by the distributed SMEs are provided jointly by both of SMEs and some constituent enterprises of this GE.

5.4 Discussion

Similar to the concepts of production efficiency and productivity, the concepts of services utilization efficiency and services invocation are defined based on hypernetwork and task conflict graph coloring in this paper. Based on these three groups of experiments, the proposed hypernetwork and graph coloring based models and the ABC algorithm method for the manufacturing service scheduling problem are validated. Based on the scheduling solutions illustrated in Fig. 6, 7 and 8, the resulted service invocation statistics of each enterprise in different situations are presented in detain in Fig. 9. Moreover, Table 4 lists the total statistics and comparison on services invocation in the above three groups of experiments.

(a) In a private cloud

(b) In a public cloud



(c) In a hybrid cloud

Fig. 9 The scheduling solution based service invocation statistics of each enterprise in different situations

Table 4 Total statistics on services invocation of all of enterprises in different situations

Three kinds of situations	**In a private cloud**	**In a public cloud**	**In a hybrid cloud**		
	Within the GE	Among all of SMEs	Within the GE	Among the remainder SMEs	In total or in average
Number of the invoked services	15	30	12	7	19
The total rate of services invocation	15/50=0.3	30/50=0.6	12/30=0.4	7/20=0.35	19/50=0.38
The average rate of services invocation	0.33	0.7025	0.4167	0.44	0.4283

In the first group of experiment for the private cloud, at least 15 services are dispatched for those 30 tasks, which makes the maximal average rate of services among all of the constituent enterprises arrive at 0.33. In the second group of experiment for the public cloud, 30 services are dispatched evenly for those 30 tasks so that the average rate of services invocation among all of SMEs is maximized as 0.7025. In addition, in the third group of experiment for the hybrid cloud, the result can be divided into two parts, as shown in Fig. 9(c). Focus on the part of SMEs (i.e., $en_1 \sim en_5$), in order to improve their average rate of services invocation, it could not be found any two tasks which are assigned by the same service. Regarding to the part of GE, there are 8 services being repeatedly dispatched for the higher utilization efficiency. Moreover, the services, like s_{42}, s_{45} and s_{28}, which are dispatched respectively by the SMEs' tasks t_3, t_4 and t_7, are provided by the constituent enterprises of the GE. It reveals a little bit of collaborations between multiple SMEs and a GE, not only the collaborations among some SMEs or among the constituent enterprises within a GE.

However, as indicated before, in a hybrid cloud, it is an extremely complex situation when SMEs and GEs co-exist. In this context, the former two conditions of a private cloud and a public cloud need to be combined together and become to an equilibrium. There still is no unified collaboration patterns or even specific scheduling mechanisms for the collaboration equilibrium. In this paper, we just define and analyze one kind of scheduling rule. Actually, both of collaboration patterns and specific scheduling mechanisms or rules, and especially the equilibrium approaches between lots of SMEs and GEs in a hybrid cloud situation, are necessary to be further explored for achieving better conditions of services invocation, sharing and utilization efficiency.

6 Conclusion

Driven by the permeation of some new-generated ITs into manufacturing, the upcoming SM is making a big difference on industry. Especially, because the sensor-based and cloud-based industrial environment are being deployed, it will integrate more and more social resources. When more manufacturing facilities whoever they belong to, and their functional capabilities, are encapsulated and utilized as cloud services, as well as their real-time production information are more sufficient and transparent, such distributed and collaborative manufacturing operations across lots of distributed but networked enterprises would be promoted and popularized.

For this case, how to settle the manufacturing service scheduling problem towards multiple enterprises and the distributed and collaborative manufacturing operations among them, is explored in this paper. The hypernetwork based models and graph coloring based models are introduced specific to the cross-enterprise and inter-time-series characteristics of this scheduling problem. In addition, as to the resulted service-based distributed collaborations among different enterprises, three kinds of situations in the cloud are explored.

According to the preliminary discussion in this paper, some further studies on this issue are pointed out as follows:

- *Collaboration equilibrium*: The situation of a hybrid cloud in which massive SMEs and GEs co-exist, is more realistic. Different collaboration patterns and scheduling mechanisms could determine various results. The approaches to make the complicated collaborations among lots of enterprises arrive at an equilibrium are necessary to be investigated.

- *Dynamic control*: With regard to various kinds of uncertain and dynamic events during the distributed manufacturing operations by lots of collaborative services, if necessary, either local adjustment or global re-scheduling through sharing and feedback of real-time information, is also important.
- *Data-based intelligence*: Benefiting from the sensor-based environment, enough real-time data is collected from each manufacturing facility and its every operation and then integrated. How to extract and utilize much more valuable information, so as to improve the scheduling solutions and promote the distributed collaborations, is one of the cores to make progress towards SM.

Acknowledgements

This work is partly supported by National Natural Science Foundation of China (Grants 51475032 and 51705014), and Hong Kong Scholar Program (Project Nos. XJ2016004 and G-YZ0K in The Hong Kong Polytechnic University).

References

Camarinha-Matos, L. (2009). Collaborative networked organizations: status and trends in manufacturing. *Annual Reviews in Control*, *33*(2): 199-208.

Chen, J., Huang, P., Wu, J., Lai, K., Chen, C., & Peng, T. (2013). Advanced planning and scheduling for TFT-LCD color filter fab with multiple lines. *International Journal of Advanced Manufacturing Technology*, *67*(1-4): 101-110.

Cheng, Y., Tao, F., Zhao, D., & Zhang, L. (2017). Modeling of manufacturing service supply-demand matching hypernetwork in service-oriented manufacturing systems. *Robotics and Computer-Integrated Manufacturing*, *45*(SI): 59-72.

Ding, K., Jiang, P., Leng, J., & Cao, W. (2016). Modeling and analyzing of an enterprise relationship network in the context of social manufacturing. *Journal of Engineering Manufacture (Proc ImechE Part B: J Eng Manufact)*, *230*(4): 752-769.

Dotoli, M., Fanti, M., Meloni, C., & Zhou, M. (2006). Design and optimization of integrated e-supply chain for agile and environmentally conscious manufacturing. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *36*(1): 62-75.

Goren, H., Tunali, S., & Jans, R. (2010). A review of applications of genetic algorithms in lot sizing. *Journal of Intelligent Manufacturing*, *21*(4): 575-590.

He, J., Chen, X., Chen, X.D., & Liu, Q. (2016). Distributed production planning based on ATC and MOILP considering different coordination patterns. *Journal of Intelligent Manufacturing*, *27*(5): 1067-1084.

Hvolby, H., & Setger-Jensen, K. (2010). Technical and industrial issues of Advanced Planning and Scheduling (APS) systems. *Computers in Industry*, *61*(9): 845-851.

Kung, L., & Chern, C. (2009). Heuristic factory planning algorithm for advanced planning and scheduling. *Computers & Operations Research*, *36*(9): 2513-2530.

Liu, J., Zhu, H., Ma, Q., Zhang, L., & Xu, H. (2015). An artificial bee colony algorithm with guide of global & local optimal and asynchronous scaling factors for numerical optimization. *Applied Soft Computing*, *37*(C): 608-618.

Lo Nigro, G., La Diega, S., Perrone, G., & Renna, P. (2003). Coordination policies to support decision making in distributed production planning. *Robotics and Computer-Integrated Manufacturing*, *19*(6): 521-531.

Lv, Y., & Lin, D. (2017). Design an intelligent real-time operation planning system in distributed manufacturing network. *Industrial Management & Data Systems*, *117*(4): 742-753.

Mishra, N., Singh, A., Kumari, S., Govindan, K., & Ali, S. (2016). Cloud-based multi-agent architecture for effective planning and scheduling of distributed manufacturing. *International Journal of Production Research*, 54(23): 7115-7128.

Moghaddam, M., & Nof, S. (2014). Combined demand and capacity sharing with best matching decisions in enterprise collaboration. *International Journal of Production Economics*, *148*: 93-109.

Moo, C., Kim, J., & Gen, M. (2004). Advanced planning and scheduling based on precedence and resource constraints for e-plant chains. *International Journal of Production Research*, *42*(15): 2941-2954.

Ozturk, C., & Ornek, M. (2016). Optimisation and constraint based heuristic methods for advanced planning and scheduling systems. *International Journal of Industrial Engineering-Theory Applications and Practice*, *23*(1): 26-48.

Peng, Y., Lu, D., & Chen, Y. (2014). A Constraint Programming Method for Advanced Planning and Scheduling System with Multilevel Structured Products. *Discrete Dynamics in Nature and Society*, doi:10.1155/2014/917685.

Rafiei, H., & Rabbani, M. (2012). Capacity coordination in hybrid make-to-stock/make-to-order production environments. *International Journal of Production Research*, *50*(3): 773-789.

Shen, W., Wang, L., & Hao, Q. (2006). Agent-based distributed manufacturing process planning and scheduling: a state-of-the-art survey. *IEEE Transactions on Systems, Man, and Cybernetics-Part T: Applications and Reviews*, *36*(4): 563-577.

Steger-Jensen, K., Hvolby, H., Nielsen, P., & Nielsen, I. (2011). Advanced planning and scheduling technology. *Production Planning & Control*, *22*(8): 800-808.

Tao, F., & Qi, Q. (2017a). New IT Driven Service-Oriented Smart Manufacturing Framework and Characteristics. *IEEE Transactions on Systems, Man and Cybernetics: Systems*, doi: 10.1109/TSMC.2017.2723764.

Tao, F., Cheng, Y., Zhang, L., & Nee, A. (2017b). Advanced manufacturing systems: socialization characteristics and trends. *Journal of Intelligent Manufacturing*, *28*(5): 1079-1094.

Tao, F., Cheng, Y., Xu, L., Zhang, L., & Li. B. (2014a). CCIoT-CMfg: Cloud Computing and Internet of Things based Cloud Manufacturing Service System. *IEEE Transactions on Industrial Informatics*, *10*(2): 1435-1442.

Tao, F., Feng, Y., Zhang, L., & Liao, T. (2014b). CLPS-GA: A case library and Pareto solution-based hybrid genetic algorithm for energy-aware cloud service scheduling. *Applied Soft Computing*, *19*: 264-279.

Tao, F., Laili, Y., Xu, L., & Zhang, L. (2013). FC-PACO-RM: A parallel method for service composition optimal-selection in cloud manufacturing system. *IEEE Transactions on Industrial Informatics*, *9*(4): 2023-2033.

Tarantilils, C., Kiranoudis, C., & Theodorakopoulos, N. (2008). A web-based ERP system for business services and supply chain management: application to real-world process scheduling. *European Journal of Operational Research*, *187*(3): 1310-1326.

Wang, K., Veeramani, D., & Chen, j. (2004). Distributed production planning using a graph-based negotiation protocol. *International Journal of Production Research*, *42*(15): 3077-3099.

Zhong, R., Huang, G., Lan, S., Dai, Q., Zhang, T., & Xu, C. (2015). A two-level advanced production planning and scheduling model for RFID-enabled ubiquitous manufacturing. *Advance Engineering Informatics*, *29*(4): 799-812.

Zhong, R., Li, Z., Pang, L., Pan, Y., Qu, T., & Huang. G. (2013). RFID-enabled real-time advanced planning and scheduling shell for production decision making. *International Journal of Computer Integrated Manufacturing*, *26*(7): 649-662.