Robotics and Computer--Integrated Manufacturing 45 (2017) 3–20

Contents lists available at ScienceDirect

Robotics and Computer–Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Workload-based multi-task scheduling in cloud manufacturing

Yongkui Liu[a,b], Xun Xu[a,*], Lin Zhang[b], Long Wang[c], Ray Y. Zhong[a]

[a] Department of Mechanical Engineering, The University of Auckland, Auckland 1142, New Zealand
[b] School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China
[c] Center for Systems and Control, College of Engineering, Peking University, Beijing 100871, China

ARTICLE INFO

Keywords:
Cloud manufacturing
Multi-task scheduling
Task workload

ABSTRACT

Cloud manufacturing is an emerging service-oriented business model that integrates distributed manufacturing resources, transforms them into manufacturing services, and manages the services centrally. Cloud manufacturing allows multiple users to request services at the same time by submitting their requirement tasks to a cloud manufacturing platform. The centralized management and operation of manufacturing services enable cloud manufacturing to deal with multiple manufacturing tasks in parallel. An important issue with cloud manufacturing is therefore how to optimally schedule multiple manufacturing tasks to achieve better performance of a cloud manufacturing system. Task workload provides an important basis for task scheduling in cloud manufacturing. Based on this idea, we present a cloud manufacturing multi-task scheduling model that incorporates task workload modelling and a number of other essential ingredients regarding services such as service efficiency coefficient and service quantity. Then we investigate the effects of different workload-based task scheduling methods on system performance such as total completion time and service utilization. Scenarios with or without time constraints are separately investigated in detail. Results from simulation experiments indicate that scheduling larger workload tasks with a higher priority can shorten the makespan and increase service utilization without decreasing task fulfilment quality when there is no time constraint. When time constraint is involved, the above strategy enables more tasks to be successfully fulfilled within the time constraint, and task fulfilment quality also does not deteriorate.

1. Introduction

Cloud manufacturing is a new service-oriented business model aiming for sharing and collaboration of large-scale manufacturing resources [1,2]. It realizes its objective through establishment of a common cloud manufacturing platform, which aggregates distributed manufacturing resources encompassed in the entire product life cycle, transforms them into manufacturing services, and manages them centrally [3,4]. Through centralized management and operation of services, cloud manufacturing is able to deal with multiple requirement tasks at the same time. A critical issue with cloud manufacturing is therefore how to schedule multiple tasks to achieve optimal system performance. Different from the scenario in cloud computing, task scheduling in cloud manufacturing is usually accompanied by logistics. The involvement of logistics makes the multi-task scheduling in cloud manufacturing more complicated.

Multi-task scheduling in cloud manufacturing refers to process of allocating services over time to perform a set of tasks while satisfying constraints in terms of time, cost, QoS, and service availability. Task scheduling is an intrinsic part of a cloud manufacturing system, and has a major impact on system performance. Effective task scheduling methods are capable of significantly enhancing system performance. For multi-task scheduling, scheduling objective should be achieving the overall optimization of all tasks. Multi-task scheduling requires the consideration of coupling relationships (e.g. different tasks may require the same type of services) among multiple tasks. Traditional methods for single-task scheduling may not achieve the optimal system performance under multi-task scenarios as they do not deal with all task as a whole [5–8]. Multi-task scheduling in cloud manufacturing has been considered in literature [9–11]. However, these works dealt with either only homogeneous tasks or using a different model and method. Multi-task scheduling in cloud computing has also been studied [12,13]. However, due to the fundamental differences between cloud manufacturing and cloud computing [4,14], the proposed approaches cannot be applied directly to cloud manufacturing. It is therefore necessary to explore new, effective methods for multi-task scheduling in cloud manufacturing.

In this paper, we address the issue of multi-task scheduling in cloud

* Corresponding author.
E-mail address: x.xu@auckland.ac.nz (X. Xu).

http://dx.doi.org/10.1016/j.rcim.2016.09.008
Received 4 October 2015; Received in revised form 18 September 2016; Accepted 26 September 2016
Available online 13 October 2016
0736-5845/ © 2016 Elsevier Ltd. All rights reserved.

Nomenclature

$a_{k,u}$	Unit service amount for $s_{k,u}$
$A_{i,s}$	Quantity of $S_{i,s}$
At_k	Arriving time of T_k
AC	Average cost of all tasks
AR	Average reliability of all tasks
AT	Average completion time of all tasks
AU	Average utility of all tasks
c_l	Logistics cost for unit weight and unit distance
$c_{i,s}$	Unit cost of $S_{i,s}$
C_k	Cost of T_k
$Cap_{i,s}$	E_i's capacity for $S_{i,s}$
$Cons_{T_k}^{T}$	Completion time constraint of T_k
CT_k	Completion time of T_k
$d_{ii'}$	Geographical distance between E_i and $E_{i'}$
E_i	Enterprise i
I	Number of enterprises in a cloud manufacturing system
J	Number of service types in a cloud manufacturing system
K	Number of tasks in a cloud manufacturing system
l_i	Number of service types of E_i
LC_k	Logistics cost of T_k
$LC_k^{u,u+1}$	Logistics cost between $s_{k,u}$ and $s_{k,u+1}$
LT_k	Logistics time of T_k
$LT_k^{u,u+1}$	Logistics time between $s_{k,u}$ and $s_{k,u+1}$
n_k	Number of subtasks of T_k
p_δ	Probability for logistics between subtasks
Q_k	Total QoS utility of T_k
Q_k^C	Cost utility of T_k
Q_k^T	Time utility of T_k
Rel_{E_i}	Reliability of E_i's services
Rel_k	Total reliability of services for T_k
$Rel_{k,u}$	Reliability of service for $s_{k,u}$
Q_k^{Rel}	Reliability utility of T_k
$r_{k,u}$	Required service type of $s_{k,u}$
$R_{i,s}$	Type of $S_{i,s}$
$s_{k,u}$	uth subtask of T_k
$S_{i,s}$	E_i's sth type of service
SC_k	Service cost of T_k
$SC_{k,u}$	Service cost of $s_{k,u}$
ST_k	Service time of T_k
$ST_{k,u}$	Ideal service time of $s_{k,u}$
$ST'_{k,u}$	Real service time of $s_{k,u}$
t_l	Logistics time for unit distance
t_k	Required service time of T_k
$t_{k,u}$	Required service time of $s_{k,u}$
T_k	Task k
w_C	Cost preference weight
w_{Rel}	Reliability preference weight
w_{SU}	Resource utilization weight
w_T	Time preference weight
w_{TCT}	Total completion time weight
wl_k	Workload of T_k
$wl_{k,u}$	Workload of $s_{k,u}$
$W_k^{u,u+1}$	Logistics weight between $s_{k,u}$ and $s_{k,u+1}$
WT_k	Waiting time of T_k
$WT_{k,u}$	Waiting time of $s_{k,u}$
α_0=1.0	Benchmark efficiency coefficient
α_i	Efficiency coefficient of E_i's services
$\delta_k^{u,u+1}$	No logistics exists between $s_{k,u}$ and $s_{k,u+1}$ for $\delta_k^{u,u+1}$=0 and vice sersa

manufacturing based on task workload [13]. The innovations of this work are as follows. First of all, we proposed a new multi-task scheduling model for cloud manufacturing based on service composition idea and method. Some critical issues pertaining to scheduling in cloud manufacturing such as logistics are taken into account. Secondly, the proposed model incorporates novel methods for modelling task workload and service (including service quantity and efficiency) [15,16], which enables us to dynamically calculate task (or subtask) fulfilment time and service utilization [17]. More importantly, based on different workload-based task scheduling methods, we find that scheduling larger-workload tasks with a higher priority can lead to better system performance such as a shorter makespan and higher service utilization. Monte Carlo methods are employed to reveal the regularity behind the scheduling methods [9–11].

The rest of this paper is structured as follows. In Section 2, a systematic literature review and corresponding analysis are conducted. Section 3 gives an example that motivates the establishment of the current multi-task scheduling model. Section 4 elaborates on the multi-task scheduling model in detail. In Section 5, a concrete multi-task scheduling example is given. Section 6 presents the results of simulation experiments and associated analysis. And finally, Section 7 concludes this paper followed by discussions on future research.

2. Literature review

First of all, it is necessary to clarify a number of fundamental concepts such as manufacturing tasks, resources, and services. Wang et al. [18] discussed the manufacturing task semantic modelling and description in a manufacturing system. In their view, manufacturing tasks can be divided into nine categories, including design tasks, manufacturing and processing tasks, logistics and inventory tasks, etc. A manufacturing task information model consisting of static information, subtask set, relation constraint, and service/capability demand was proposed. Wang et al. [19] described customers' requirement tasks at four different levels, namely, products, parts, processing technology, and machining procedure (or process). Accordingly, manufacturing resources can be categorized into four different levels, i.e. enterprise level, workshop level, cell level, and device level [20]. The classifications above reflect the multi-level and multi-granularity characteristics of requirement tasks and resources. Liu et al. [20] proposed a multi-granularity manufacturing resource model for the multi-granularity matching between manufacturing tasks and manufacturing capabilities, as well as the approach to encapsulating manufacturing capabilities into manufacturing cloud services by extending OWL-S.

To date, a couple of works have addressed the multi-task scheduling issue in cloud manufacturing. Cheng et al. [10] dealt with multiple task-oriented virtual resource integration and optimal scheduling from the perspective of cloud manufacturing enterprises. They focused on the issue of scheduling tasks as many as possible onto a fixed amount of resources to obtain a higher profit for an enterprise under the constraint of delivery deadlines. Jian et al. [9] dealt with the scheduling of a batch of workshop production tasks with the same characteristic and production process. The research issue is that, given the production time and production cost of each task in a production process, how to schedule tasks to minimize the total cost and time. Different from the aforementioned two works where only the same type of tasks was tackled, Li et al. [11] addressed the scheduling of multiple heterogeneous tasks at the subtask level. Also, the transportation of components or products between subtasks is taken into account. To achieve the optimization objectives, all of the three works above considered resource occupancy and time division sharing. Lartigau et al. [17] discussed scheduling methodology for production services in cloud manufacturing, and proposed a framework for scheduling methodology

at the cloud platform level. In their approach, a couple of important concepts in production service, such as batch of a task, quantity of resources that were usually ignored or not fully considered in many previous works were taken into account. Liu et al. [21,22] addressed the problem of multi-task service composition in cloud manufacturing with the objective of maximizing the overall QoS of all tasks. As they focused on service composition rather than service scheduling, they did not consider the change of service state (e.g. service occupancy). This does not accord with practical situations where service states (e.g. service availability) are constantly changing over time, which can lead to the changes in QoS of services (especially in terms of time [7]). Another problem is that they only took into account service functionality, and left out service quantity. As a matter of fact, services like manufacturing resources also have the property of quantity. For instance, the service quantity offered by 10 machine tools is larger than that of 5 or less machine tools. The time for an enterprise to complete a manufacturing task is closely related to the quantity of services [17]. Moreover, the absence of service quantity makes it impossible to calculate service utilization.

Many works dealt with service (or resource) scheduling from different perspectives. Cao et al. [5] discussed a service selection and scheduling strategy in cloud manufacturing for a single task. Different from most of previous works on service composition, service occupancy during the service selection process was explicitly taken into account. This represents an important progress in cloud manufacturing service composition since service selection takes place during the process of service scheduling in which service occupancy frequently occurs. Wei et al. [23] proposed a scheduling model for cloud design resources for a sequence of atomic service requests. Laili et al. [24] addressed the scheduling of multiple collaborative design tasks with precedence constraint in cloud manufacturing. The multiple tasks in the above two works are actually atomic tasks requiring only one type of resource or service. Very recently, Lin et al. [25] dealt with the project scheduling for computing resources allocation in a cloud manufacturing system. A project in the research resembles a composite task requiring multiple types of resources with complex precedence relationships among tasks, and a task is actually similar to a subtask in the current model. In their case, one type of computing resource can be used for handling multiple tasks at the same time, which is usually not the case for production resources where exclusive use is usually assumed. Moreover, the QoS of computing resources was not fully considered. Importantly, the scheduling is not from the perspective of service composition. Cheng et al. [26] discussed resource service scheduling in cloud manufacturing from the perspective of resource service and capability transaction and scheduling management. The determination of scheduling objective is an important issue for cloud manufacturing service scheduling as three types of stakeholders, (i.e. operator, demanders and providers) are involved in cloud manufacturing. Different scheduling objectives lead to different ways of distributing interest. In this aspect, Tao et al. [27] discussed utility modelling, equilibrium and coordination of resource service transactions in a service-oriented manufacturing system.

Many authors have studied the problem of cloud manufacturing service composition, which bears some relevance to the current research. Tao et al. [6] proposed a parallel method for service composition optimal-selection in a cloud manufacturing system. Lartigau et al. [7] dealt with the issue of cloud manufacturing service composition taking geo-perspective transportation and execution time into account. Jin et al. [8] considered correlations among cloud services in cloud manufacturing.

Task scheduling in cloud computing has been intensively studied. Kumar et al. [12] discussed the scheduling of independent tasks in cloud computing using an improved genetic algorithm, which incorporates Min-Min and Max-Min algorithms. Wu et al. [13] proposed a QoS-driven task scheduling algorithm in cloud computing. In their model, they first computed the priority of tasks according to task attributes such as user privilege, task urgency, latency time and task workload, and then scheduled them according to their priority. The objective is to reduce the completion time and latency time, as well as achieve a better load balancing. In the above two works, the authors considered the scheduling of atomic computing tasks of the same type, which is quite different from heterogeneous composite manufacturing tasks considered in this paper. In addition, due to they concentrated on cloud computing task scheduling, logistics was not considered there.

3. A motivating example

Consider a cloud manufacturing platform involving 10 enterprises from Guangdong Zhaoqing Automotive Parts Industry Association, which are Huaiji Dengyun Auto-parts (Holding) Co., Ltd. (Dengyun), Zhaoqing Honda Foundry Col, Ltd. (Honda), Guangdong Hongtu Technology (Holdings) Co., Ltd. (Hongtu), Guangdong Sihui ShiLi Connecting-Rod Co., Ltd. (Shili), Guangdong Zhaoqing Power Foundry (Holding) Co., Ltd. (Power), Guangdong Hong Teo Accurate Technology Co., Ltd. (Hong Teo), Delta Aluminium Industry Co., Ltd. (Delta), Zhaoqing Huafeng Electron Lvbo Company Ltd. (Huafeng), Zhaoqing Sunspring Industrial Co., Ltd. (Sunspring), and Zhaoqing Fenghua Advanced Co., Ltd. (Fenghua). Each enterprise has a number of different types of manufacturing resources such as milling, turning, drilling, punching, welding, grinding, planing, and boring machine tools (Table 1. Note that the data shown in Table 1 for each case company is representative in order to keep the confidentiality of their key businesses). Table 2 presents their geographical distances between these case enterprises.

Assume that at a time the cloud manufacturing platform receives eight tasks of producing typical automotive engine parts such as valve, EGR passage, clutch housing, and oil pan, and each task consists of six subtasks (Table 3). Each subtask needs a certain type of resources

Table 1
Case companies and the offered resources.

Company/ Location	Resources	Quantity	Reliability (pass-rate) (%)	Unit Cost ($)	Efficiency (part/day)
Dengyun/ Huaiji	Turning	32	92	15	5
	Drilling	53	86	24	2
	Welding	48	95	19	3
	Planing	39	94	16	3
Honda/ Zhaoqing	Boring	43	82	20	5
	Grindering	37	95	22	1
	Turning	55	92	19	3
Hongtu/ Gaoyao	Punching	38	89	17	6
	Milling	46	95	23	4
	Welding	54	94	21	1
Shili/Shihui	Planing	32	92	16	5
	Turning	46	83	22	6
	Drilling	58	94	25	4
Power/ Duanzhou	Boring	56	95	24	4
	Planing	36	98	23	5
	Welding	40	92	19	5
Hong Teo/ Dinghu	Milling	52	84	25	6
	Drilling	41	96	19	3
	Grinding	31	93	21	5
Delta/Longpu	Boring	52	83	25	5
	Grinding	45	94	23	6
	Punching	37	92	18	4
Huafeng/ Lantang	Welding	43	94	15	4
	Planing	36	92	15	5
	Turning	50	86	18	6
Sunspring/ Beishui	Drilling	35	94	16	3
	Milling	42	87	24	6
	Boring	50	93	20	6
Fenghua/ Xialong	Milling	45	85	21	4
	Drilling	50	94	23	5
	Boring	45	95	17	6
	Punching	56	81	19	6

Table 2
Geographical distances between the case companies (km).

	Dengyun	Honda	Hongtu	Shili	Power	Hong Teo	Delta	Huafeng	Sunspring	Fenghua
Dengyun	0	355.4	245.5	272.3	20.2	126.2	55.2	10.8	19.3	170.3
Honda	355.4	0	100.2	21.5	153.9	200.2	24.5	15.4	148.9	298.1
Hongtu	245.5	100.2	0	292.6	60.2	28.7	73.2	22.2	21.1	18.4
Shili	272.3	21.5	292.6	0	113.5	175.8	178.1	301.6	156.9	279.3
Power	20.2	153.9	60.2	113.5	0	41.6	16.5	121.0	18.6	16.0
Hong Teo	126.2	200.2	28.7	175.8	41.6	0	19.6	63.1	92.4	186.2
Delta	55.2	24.5	73.2	178.1	16.5	19.6	0	16.6	48.9	32.5
Huafeng	10.8	15.4	22.2	301.6	121.0	63.1	16.6	0	102.7	64.3
Sunspring	19.3	148.9	21.1	156.9	18.6	92.4	48.9	102.7	0	136.1
Fenghua	170.3	298.1	18.4	279.3	16.0	186.2	32.5	64.3	136.1	0

(Table 4). After task composition, the cloud manufacturing platform will schedule these tasks onto the services (as in the context of cloud manufacturing) offered by the 10 enterprises mentioned above. Each task has a specific workload, which can be expressed as the product of resource efficiency and required completion time for a unit resource. For instance, Milling-1/2/10 in Table 1 indicates that the operation needs 1 unit milling equipment resource with the efficiency of 2 parts per day to work for 10 days. There are constraints on task execution in terms of time, cost, and reliability (reliability is measured by pass-rate) (the last column of Table 4). For instance, 15/5K/0.96 means that the task is required to be finished within 15 days with less than 5000USD and the pass-rate should not be below 96%. Each task has a fixed subtask execution flow (i.e. subtask structure). Fig. 1 shows the subtask execution flow of all tasks shown in Table 3. It should be noted that some subtasks of these tasks are identical and thus require the same resources.

In this example, an important issue is how to schedule these tasks with different workloads onto the services to better satisfy users' requirements and achieve better system performance such as a shorter makespan and higher service utilization. This is indeed a multi-task scheduling problem. Solving this problem requires the establishment of a suitable model.

4. A multi-task scheduling model

4.1. Enterprises and services

Assume that there are I registered enterprises in the current cloud manufacturing system, which are denoted by $Ent(I)=\{E_1, \ldots, E_i, \ldots, E_I\}$ (Fig. 2). Enterprise E_i $(1 \le i \le I)$ offers l_i $(1 \le l_i \le J)$ different types of manufacturing services (such as design services, production services, and processing services), which are selected randomly from total J types of services in the entire cloud manufacturing system. The sth $(1 \le s \le l_i)$ type of service is denoted by $S_{i,s}$. The following attributes of $S_{i,s}$, including type $R_{i,s}$, quantity $A_{i,s}$, unit cost $c_{i,s}$, efficiency coefficient α_i, and reliability Rel_{E_i}, are taken into account. The introduction of efficiency coefficient α_i is motivated by the fact that different enterprises may have different efficiencies in fulfilling a task (or subtask) with the same type of resource, which means that they may need different amounts of time for fulfilling the same task (or subtask) even using the same type and the same amount of manufacturing resources [15,16]. For example, the efficiencies of the lathing resources provided by Dengyun, Honda, Shili, and Huafeng are 5, 3, 6, and 6 parts per day, respectively. The average efficiency is thus 5 parts per day. Hence, the efficiency coefficients are 1.0 (5/5), 0.6 (3/5), 1.2 (6/5), and 1.2 (6/5), respectively. The efficiency of E_i's services depends on many factors such as enterprise management level, resource quality. For the sake of simplicity but without loss of generality, we assume that all services of an enterprise have the same efficiency. The quantity $A_{i,s}$ of $S_{i,s}$ along with its efficiency coefficient α_i characterizes the capacity of enterprise E_i with respect to $S_{i,s}$, which is defined as $Cap_{i,s}=A_{i,s}\times\alpha_i$. The introduction of the concept of enterprise capacity facilitates the calculation of the time required for $S_{i,s}$ to complete a subtask.

4.2. Requirement tasks

Requirement tasks come from the decomposition of users' orders. Assume that at a time there are K tasks to be processed in the current cloud manufacturing system, which are represented by $Task(K)=\{T_1, \ldots, T_k, \ldots, T_K\}$ [21,22] (Fig. 2). A task may require one type of service or multiple different types of services. Here, we deal with the latter type of tasks so that T_k can be decomposed into n_k subtasks, with the uth $(1 \le u \le n_k)$ subtask being represented by $s_{k,u}$. Each subtask requires a different type of service, which is selected randomly from the total J types of services in the entire cloud manufacturing system. T_k has a certain subtask structure, which is usually a combination of the four basic structures, including sequential, parallel, selective, and circular [28]. For simplicity and without affecting the credibility of our results, the sequential subtask structure is assumed, i.e. T_k's subtasks have a linear structure so that they are executed sequentially [7,8].

For $s_{k,u}$, the following attributes, including the required service type $r_{k,u}$, the required service time $t_{k,u}$ when using a unit service $a_{k,u}$, and the benchmark efficiency coefficient α_0, are taken into account. The introduction of the variables above is motivated by the fact that each subtask has a certain workload $wl_{k,u}$, which will take a period of time for its completion using a certain amount of service (of a certain type and with a certain efficiency coefficient) [18]. Based on the concepts

Table 3
Subtask information of each task.

Task	Subtask1	Subtask2	Subtask3	Subtask4	Subtask5	Subtask6
30207537	Valve	Clutch housing	Crankcase	Oil pan	Connecting rod	Gear housing
30207538	EGR passage	Crankcase	Valve	Oil pan	Gear housing	Connecting rod
30207540	Crankcase	EGR passage	Valve	Clutch housing	Connecting rod	Gear housing
30207541	Gear housing	EGR passage	Valve	Crankcase	Valve	Oil pan
30207543	Valve	Crankcase	Connecting rod	Oil pan	Gear housing	EGR passage
30207568	Gear housing	Clutch housing	EGR passage	Valve	Crankcase	Oil pan
30207573	Oil pan	Gear housing	Connecting rod	Valve	EGR passage	Clutch housing
30201025	Crankcase	Oil pan	Connecting rod	Valve	EGR passage	Clutch housing

Table 4
Required resource information of tasks.

Task/Batch	Subtask1 Res-T/W	Subtask2 Res-T/W	Subtask3 Res-T/W	Subtask4 Res-T/W	Subtask5 Res-T/W	Subtask6 Res-T/W	Constraints
30207537/150	Lathing-1/5/20	Punching-1/2/15	Planing–1/5/15	Drilling-1/4/20	Boring-1/5/20	Welding-1/2/5	80/15K/95%
30207538/30	Milling-1/2/10	Grindering-1/4/10	Lathing–1/5/20	Drilling-1/4/20	Welding-1/2/5	Planing-1/5/15	15/5K/96%
30207540/50	Grindering-1/4/10	Milling-1/2/10	Lathing–1/5/20	Planing-1/5/15	Punching-1/2/15	Welding-1/2/5	30/20K/90%
30207541/60	Welding-1/2/5	Milling-1/2/10	Lathing–1/5/20	Punching-1/2/15	Lathing-1/5/20	Drilling-1/4/20	40/50K/95%
30207543/150	Grindering-1/4/10	Grindering-1/4/10	Punching–1/2/15	Drilling-1/4/20	Welding-1/2/5	Milling-1/2/10	80/60K/96%
30207568/100	Welding-1/2/5	Punching-1/2/15	Milling–1/2/10	Lathing-1/5/20	Planing-1/5/15	Boring-1/5/20	50/40K/92%
30207573/300	Drilling-1/4/20	Welding-1/2/5	Planing-1/5/15	Lathing-1/5/20	Milling-1/2/10	Lathing-1/5/20	120/50K/95%
30201025/50	Grindering-1/4/10	Drilling-1/4/20	Boring-1/5/20	Lathing-1/5/20	Boring-1/5/20	Boring-1/5/20	30/40K/93%

introduced above, the workload of $s_{k,u}$ can be expressed as $wl_{k,u}=a_{k,u}\times\alpha_0\times t_{k,u}$. Thus, the total workload wl_k of T_k can be expressed as $\sum_{u=1}^{n_k} wl_{k,u}$. Task T_k arrives at the system at At_k.

4.3. Task scheduling

4.3.1. Service searching and matching

After a task is submitted to a cloud manufacturing platform, it first should be decomposed into a number of subtasks so that the platform can search for available matching services for each subtask. In this paper, we assume that task T_k has already been decomposed, and are not concerned with how to decompose a task (as this is largely beyond the scope of this paper) (Fig. 2).

After T_k has been decomposed, the cloud manufacturing platform searches for the matching services for each subtask among all services in the service pool. This results in a service set for each subtask (Fig. 3). Note that the services searched include not only the currently available ones but also the ones that are currently occupied by other tasks. Note also that the service searching and matching is subject to the constraint that each subtask $s_{k,u}$ can be undertaken by only one enterprise at each period p (the concept of p will be introduced in Section 4.3.2) and that once an enterprise is selected, $s_{k,u}$ will be undertaken by this enterprise until it is completed. In order to describe this constraint, a Boolean variable $y(i, s, k, u, p)$ is introduced. If $S_{i,s}$ is selected to fulfil $s_{k,u}$ at period p, then $y(i, s, k, u, p)=1$, otherwise $y(i, s, k, u, p)=0$. The constraint that only one enterprise is allowed for undertaking $s_{k,u}$ at period$p$ can be described by $\sum_{i=1}^{I}\sum_{s=1}^{l_i} y(i, s, k, u, p)\le 1$ for any fixed k, u and p ($1 \le k \le K$, $1 \le u \le n_k$, and $1 \le p \le \infty$).

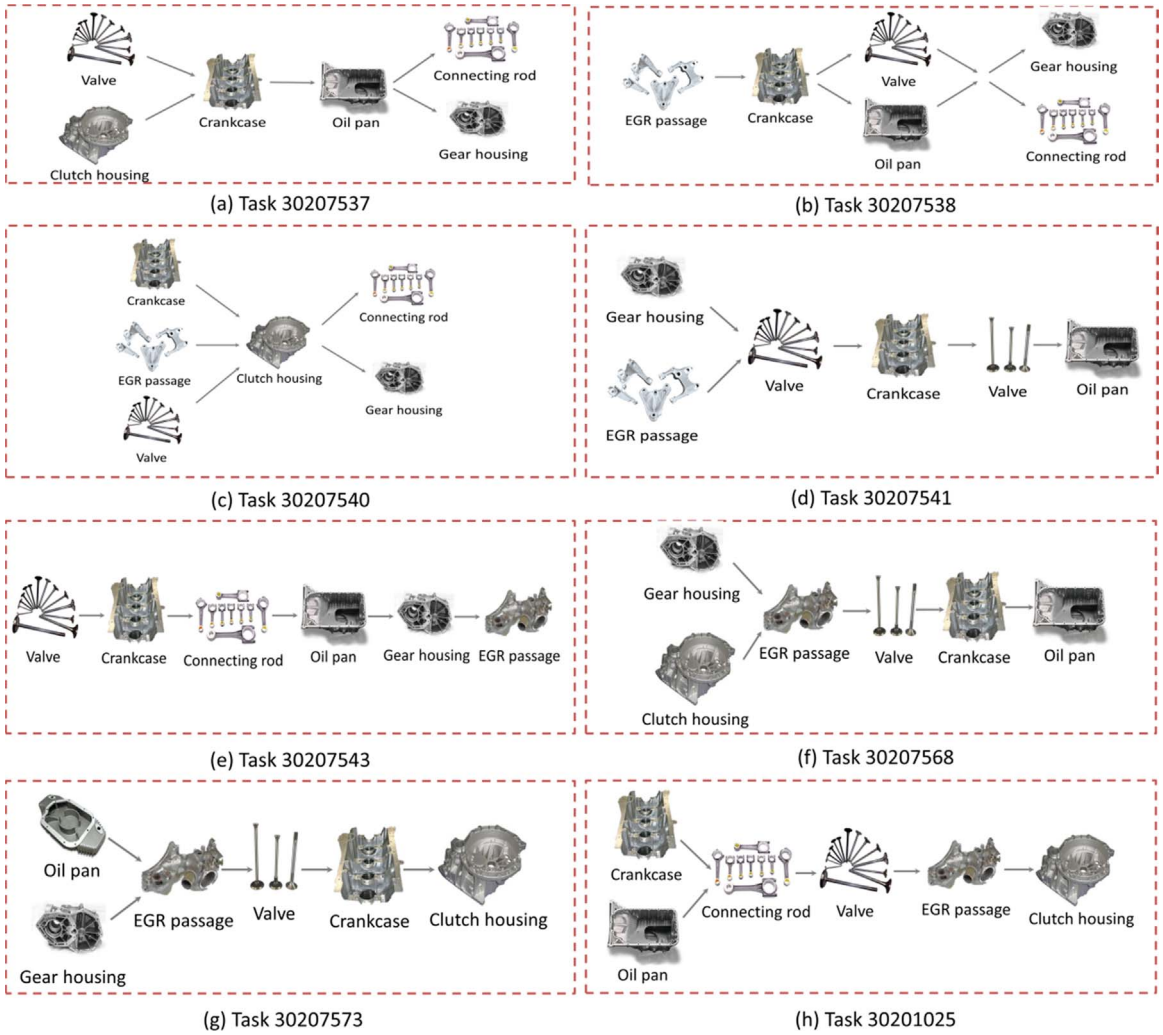

Fig. 1. Subtask execution flow of tasks shown in Table 3.

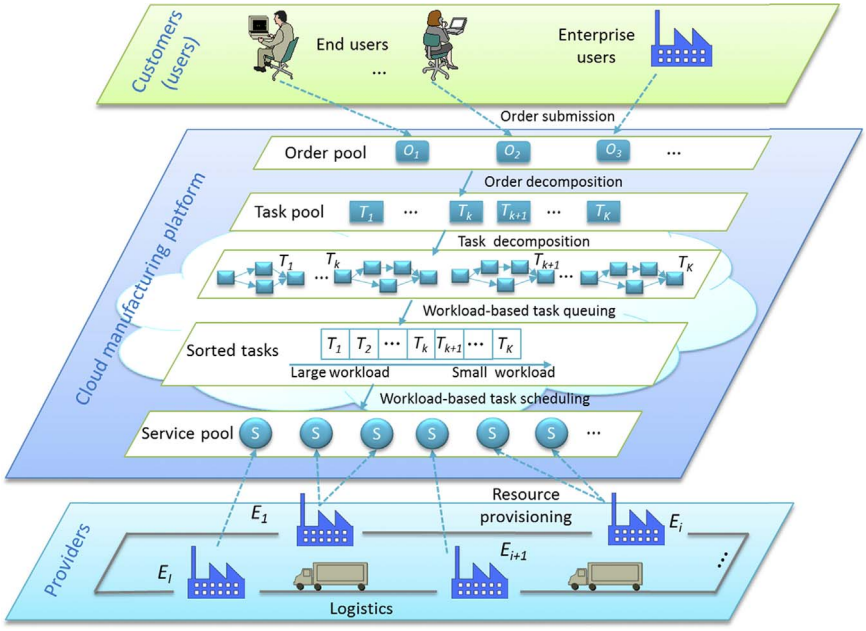

Fig. 2. Schematic diagram of workload-based multi-task scheduling in cloud manufacturing.

4.3.2. Completion time

The completion time CT_k of task T_k is the time required for completing it. The total completion time (i.e. TCT) of all tasks is the time required for the completion of all tasks. Time is measured in period p, which is the minimum, inseparable time unit. The completion time CT_k of T_k consists of three parts: service time (including all types of times needed to fulfil a task such as setup time, execution time, maintenance time) ST_k, logistics time LT_k, and waiting time WT_k (which is caused by service occupancy).

The service time for the uth subtask $s_{k,u}$ of T_k to be undertaken by enterprise E_i with $S_{i,s}$ is $ST_{k,u}=wl_{k,u}/Cap_{i,s}$. The service time is usually decimal. In order to be measured in period p, it needs to be rounded, i.e. $INT(ST_{k,u})$. Here, the decimal is rounded to the smallest integer greater than or equal to the decimal. It should be noted that this treatment statistically does not influence the credibility of the results (or qualitatively change the results) as it applies to all situations.

Another important type of time is logistics time as task execution in cloud manufacturing is usually accompanied by the physical flow of raw materials, products or components, etc. [7]. The logistics time $LT_k^{u,u+1}$ between two successive subtasks $s_{k,u}$ and $s_{k,u+1}$ depends on three factors: the geographical distance $d_{ii'}$ between enterprises E_i and $E_{i'}$ that undertake the subtasks, logistics time for unit distance t_l, and logistics probability p_δ (which characterizes whether logistics is needed (in fact, in cloud manufacturing logistics is not always needed. For example, there is no need for logistics for design services)). The Boolean $\delta_k^{u,u+1}$ is introduced to characterize whether logistics is needed between subtask $s_{k,u}$ and $s_{k,u+1}$. $\delta_k^{u,u+1}=0$ means that there is no need for logistics and vice versa. Thus, the logistics time between subtask $s_{k,u}$ (undertaken by E_i) and $s_{k,u+1}$ (undertaken by $E_{i'}$) is $LT_k^{u,u+1}=\delta_k^{u,u+1}\times t_l\times d_{i,i'}$. If two successive subtasks are undertaken by the same enterprise, then the logistics time is zero.

In our model, we assume that service $S_{i,s}$ can be occupied by only one subtask at a time. This can be described by $\sum_{k=1}^{K}\sum_{u=1}^{n_k} y(i, s, k, u, p)\le 1$, which means that $S_{i,s}$ can be occupied by only one subtask at any period p. Waiting time is caused by service occupancy. Two types of tasks need to be considered when it comes to whether the execution processes of subtasks can be interrupted. The first type of tasks are those whose subtasks' execution processes can be interrupted (i.e. subtasks' execution may span discontinuous periods), and the other type of tasks are those whose subtasks must be performed within a continuous period of time until their completion. This paper concentrates on the first type of tasks (in fact, we have

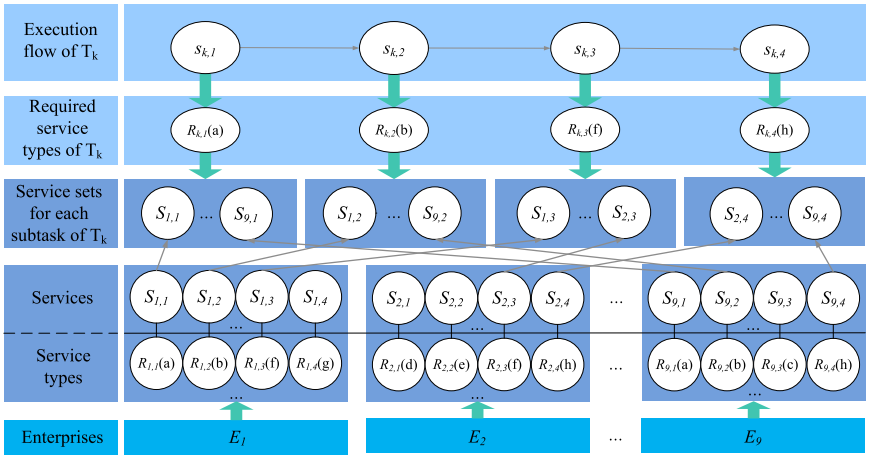

Fig. 3. An example of service searching and matching between task T_k with four subtasks (i.e. $s_{k,1}$, $s_{k,2}$, $s_{k,3}$, and $s_{k,4}$) and services offered by 9 enterprises (i.e. E_1 to E_9). The letters in parentheses denote the type of the corresponding required services.

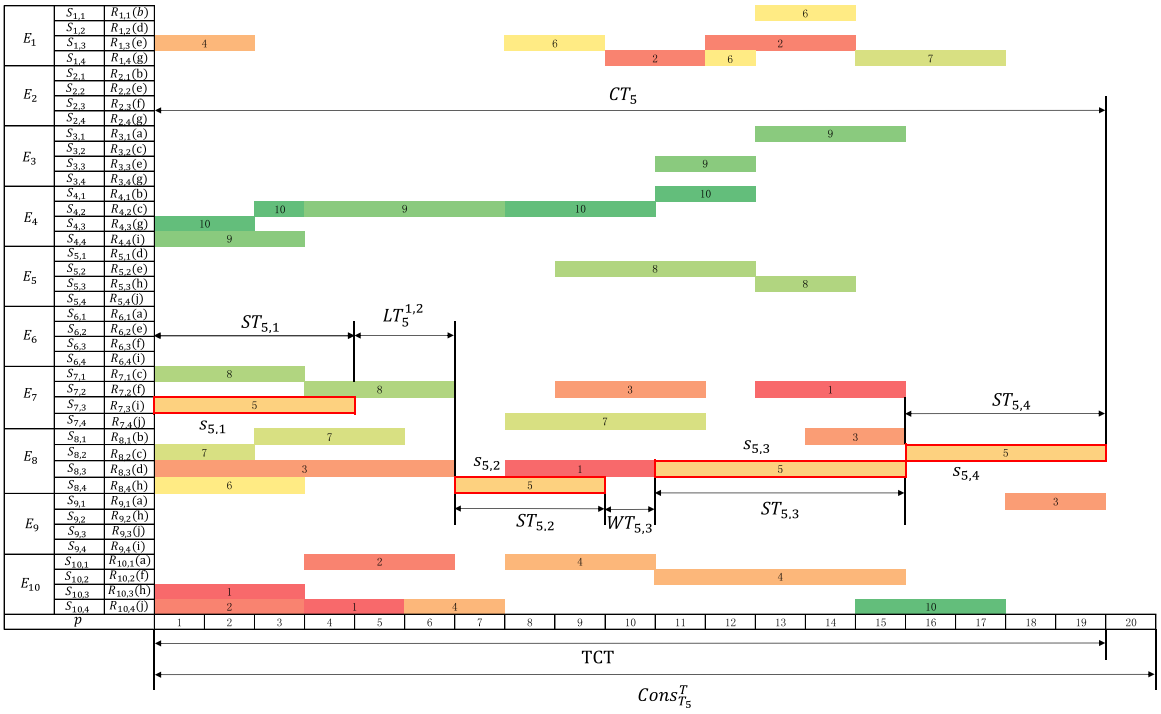

Fig. 4. Diagram of scheduling 10 tasks onto 10 types of services offered by 10 enterprises.

checked that the obtained results do not qualitatively change for the second type of tasks).

In order to compute the waiting time of subtask $s_{k,u}$, concepts including ideal service time $ST_{k,u}$ and real service time $ST'_{k,u}$ are introduced (Fig. 4). The former is the time required for a service to perform $s_{k,u}$ without being occupied, while the latter is the time required for the case where service occupancy exists. $ST'_{k,u}$ is the number of periods from the ideal staring time of $s_{k,u}$ to the point in time at which $s_{k,u}$ is completed. The ideal starting time of $s_{k,u}$ is the next period of the logistics end time of $s_{k,u-1}$ (the ideal starting time for $s_{k,1}$ is p=1). Thus, the waiting time of $s_{k,u}$ is $WT_{k,u}$=$ST'_{k,u}$ − $ST_{k,u}$.

Based on the three types of times, and according to the precedence relationships between the subtasks of T_k, one can calculate the total completion time of T_k. Taking for example the sequential structure scenario, the completion time of T_k is CT_k=ST_k+LT_k+WT_k, where ST_K=$\sum_{u=1}^{n_k} ST_{k,u}$ is the total service time, LT_K=$\sum_{u=1}^{n_k-1} LT_k^{u,u+1}$ is the total logistics time, and WT_K=$\sum_{u=1}^{n_k} WT_{k,u}$ is the total waiting time.

4.3.3. Task cost

T_k's cost C_k includes service cost and logistics cost. The service cost for E_i to fulfil $s_{k,u}$ with $S_{i,s}$ is $SC_{k,u}$=$A_{i,s}$×$c_{i,s}$. The logistics cost from E_i (undertaking $s_{k,u}$) to $E_{i'}$ (undertaking $s_{k,u+1}$) is $LC_k^{u,u+1}$=$\delta_k^{u,u+1}$×c_l ×$W_k^{u,u+1}$×$d_{ii'}$, where $\delta_k^{u,u+1}$ is the Boolean variable characterizing whether logistics exists, c_l represents the logistics cost for unit weight and unit distance, $W_k^{u,u+1}$ stands for weight of products, parts or raw materials needed to be transported between $s_{k,u}$ and $s_{k,u+1}$, and $d_{ii'}$ denotes the geographical distance between E_i and $E_{i'}$. As mentioned above, if two consecutive subtasks are executed within the same enterprise, the logistics cost will be zero. Thus, for T_k with a sequence subtask structure, the total cost T_k is C_k=SC_k+LC_k, where SC_k=$\sum_{u=1}^{n_k} SC_{k,u}$ is the total service cost and LC_k=$\sum_{u=1}^{n_k-1} LC_k^{u,u+1}$ is the total logistics cost.

4.3.4. Task reliability

Reliability can also be calculated according to the subtask structure of T_k. Taking for example the sequential subtask structure of T_k, the total reliability is Rel_k=$\prod_{u=1}^{n_k} Rel_{k,u}$, where $Rel_{k,u}$ is the reliability of service for $s_{k,u}$ (when the service for $s_{k,u}$ is provided by E_i, $Rel_{k,u}$ is equal to Rel_{E_i}). Note that hereby only the reliability of manufacturing services is considered, although logistics services are also an essential part of cloud manufacturing services.

4.3.5. Scheduling methods

The scheduling methods are as follows. First, tasks are queued in a descending (or ascending) order of workload, and then processed sequentially. Together with the random scheduling method, there are three types of scheduling methods in total:

- *Random scheduling* (R). In this method, tasks are scheduled in the order of their numberings irrespective of their workloads. This method acts as a benchmark for comparing the results obtained with different methods.
- *Workload-based scheduling*. In this method, tasks are processed in a descending (W1, i.e. tasks with a larger workload are handled with a high priority) or an ascending order of workload (W2, i.e. tasks with a smaller workload are handled with a high priority).

Scheduling scenarios with or without a time constraint are taken into account:

- *Without time constraint*. The detailed steps are as follows: (1) a task is scheduled for execution, (2) a cloud manufacturing platform searches for all matching services (including the occupied ones) for each subtask to obtain a service set, (3) all the possible service composition solutions are calculated, (4) the overall QoS utilities of all the possible composition solutions are calculated, (5) the composition solution with the highest overall QoS utility is selected, and (6) the corresponding services and their occupying periods are recorded. This steps above cycle until all tasks have been executed.
- *With time constraint*. When time constraint is considered, some change needs to be made to step (5) for the scenario without time constraint. In this case, the optimal service composition solutions should be selected among the ones that satisfy the time constraint. If no solution could meet the time constraint of a task, then the task is regarded as being unsuccessfully executed. An unsuccessfully executed task does not occupy any services. That is why failure rate (FR,

see Section 4.3.6 for its definition) needs to be introduced for the scenario with time constraint.

4.3.6. System performance metrics

The following metrics are used to evaluate the system performance with the scheduling methods in Section 4.3.5:

- *Total completion time (TCT,* i.e. makespan*)*. The total completion time is the time from the arrival of the first task until the completion of all tasks.
- *Service utilization* (*SU*). Service utilization is defined as the ratio of the number of the total service occupying periods to that of the total periods within the total completion time.
- *Failure rate* (*FR*). This index is specially introduced for the case with time constraint. The failure rate is the ratio of the number of the tasks that are unsuccessfully executed to that of all tasks.
- *Average completion time* (*AT*). Average completion time is the ratio of the total completion time of all tasks to the number of tasks.
- *Average cost* (*AC*). Average cost is the ratio of the total cost of all tasks to the number of tasks.
- *Average reliability* (*AR*). Average reliability is the ratio of the total reliability of all tasks to the number of tasks.

4.3.7. Scheduling objectives

There are mainly two different task scheduling objectives. According to the task scheduling method, when task T_k is scheduled for execution, we can consider only the QoS utility of T_k, or consider not only the QoS utility of T_k but also the effects of scheduling T_k on system performance such TCT and SU. In the former case, the objective is to achieve the optimal execution of T_k, thus users' requirements can be best satisfied. In the latter case, the objective is to achieve the overall optimization of the entire system (i.e. not only satisfy users' requirements, but also shorten TCT and increase SU). The former is a customer-centric scheduling method while the latter is a comprehen-

Table 5
Enterprise and service information for Fig. 4 (the numbers in parentheses indicate the values (or codes) of the corresponding parameters).

E_i	l_i	$S_{i,s}$	$R_{i,s}$	$A_{i,s}$	$c_{i,s}$	Rel_{E_i}	α_i
E_1	l_1(4)	$S_{1,1}$	$R_{1,1}$(b)	$A_{1,1}$(42.0176)	$c_{1,1}$(24.0857)	Rel_{E_1}(0.9288)	α_1(1.4902)
		$S_{1,2}$	$R_{1,2}$(d)	$A_{1,2}$(51.2052)	$c_{1,2}$(20.2199)		
		$S_{1,3}$	$R_{1,3}$(e)	$A_{1,3}$(48.1298)	$c_{1,3}$(19.1124)		
		$S_{1.4}$	$R_{1.4}$(g)	$A_{1.4}$(43.8166)	$c_{1.4}$(19.7426)		
E_2	l_2(4)	$S_{2,1}$	$R_{2,1}$(b)	$A_{2,1}$(52.595)	$c_{2,1}$(23.3456)	Rel_{E_2}(0.8259)	α_2(1.1304)
		$S_{2,2}$	$R_{2,2}$(d)	$A_{2,2}$(38.088)	$c_{2,2}$(18.8069)		
		$S_{2,3}$	$R_{2,3}$(f)	$A_{2,3}$(33.101)	$c_{2,3}$(20.3639)		
		$S_{2,4}$	$R_{2,4}$(g)	$A_{2,4}$(37.6495)	$c_{2,4}$(16.6092)		
E_3	l_3(4)	$S_{3,1}$	$R_{3,1}$(a)	$A_{3,1}$(58.0425)	$c_{3,1}$(15.9015)	Rel_{E_3}(0.8285)	α_3(1.4206)
		$S_{3,2}$	$R_{3,2}$(c)	$A_{3,2}$(51.359)	$c_{3,2}$(16.6105)		
		$S_{3,3}$	$R_{3,3}$(e)	$A_{3,3}$(53.1022)	$c_{3,3}$(15.7126)		
		$S_{3,4}$	$R_{3,4}$(g)	$A_{3,4}$(42.252)	$c_{3,4}$(18.759)		
E_4	l_4(4)	$S_{4,1}$	$R_{4,1}$(b)	$A_{4,1}$(48.2891)	$c_{4,1}$(21.4467)	Rel_{E_4}(0.9252)	α_4(1.2592)
		$S_{4,2}$	$R_{4,2}$(c)	$A_{4,2}$(44.8402)	$c_{4,2}$(24.2456)		
		$S_{4,3}$	$R_{4,3}$(g)	$A_{4,3}$(40.8383)	$c_{4,3}$(19.2351)		
		$S_{4,4}$	$R_{4,4}$(i)	$A_{4,4}$(51.9825)	$c_{4,4}$(21.194)		
E_5	l_5(4)	$S_{5,1}$	$R_{5,1}$(d)	$A_{5,1}$(47.8368)	$c_{5,1}$(18.2804)	Rel_{E_5}(0.8626)	α_5(1.0917)
		$S_{5,2}$	$R_{5,2}$(e)	$A_{5,2}$(57.3019)	$c_{5,2}$(21.896)		
		$S_{5,3}$	$R_{5,3}$(h)	$A_{5,3}$(38.1622)	$c_{5,3}$(20.132)		
		$S_{5,4}$	$R_{5,4}$(j)	$A_{5,4}$(51.8094)	$c_{5,4}$(22.0586)		
E_6	l_6(4)	$S_{6,1}$	$R_{6,1}$(a)	$A_{6,1}$(55.2144)	$c_{6,1}$(19.7273)	Rel_{E_6}(0.8441)	α_6(0.62149)
		$S_{6,2}$	$R_{6,2}$(e)	$A_{6,2}$(37.8674)	$c_{6,2}$(21.3106)		
		$S_{6,3}$	$R_{6,3}$(f)	$A_{6,3}$(48.5922)	$c_{6,3}$(22.9836)		
		$S_{6,4}$	$R_{6,4}$(i)	$A_{6,4}$(43.4925)	$c_{6,4}$(23.1256)		
E_7	l_7(4)	$S_{7,1}$	$R_{7,1}$(c)	$A_{7,1}$(43.1748)	$c_{7,1}$(18.9332)	Rel_{E_7}(0.9906)	α_7(1.2347)
		$S_{7,2}$	$R_{7,2}$(f)	$A_{7,2}$(59.1421)	$c_{7,2}$(22.6678)		
		$S_{7,3}$	$R_{7,3}$(i)	$A_{7,3}$(46.6942)	$c_{7,3}$(23.4146)		
		$S_{7,4}$	$R_{7,4}$(j)	$A_{7,4}$(51.2033)	$c_{7,4}$(24.0475)		
E_8	l_8(4)	$S_{8,1}$	$R_{8,1}$(b)	$A_{8,1}$(50.6174)	$c_{8,1}$(22.3519)	Rel_{E_8}(0.9634)	α_8(1.4327)
		$S_{8,2}$	$R_{8,2}$(c)	$A_{8,2}$(45.2348)	$c_{8,2}$(15.3595)		
		$S_{8,3}$	$R_{8,3}$(d)	$A_{8,3}$(32.7512)	$c_{8,3}$(21.9829)		
		$S_{8,4}$	$R_{8,4}$(h)	$A_{8,4}$(51.251)	$c_{8,4}$(21.9466)		
E_9	l_9(4)	$S_{9,1}$	$R_{9,1}$(a)	$A_{9,1}$(52.118)	$c_{9,1}$(17.2376)	Rel_{E_9}(0.8539)	α_9(1.2879)
		$S_{9,2}$	$R_{9,2}$(h)	$A_{9,2}$(46.8764)	$c_{9,2}$(16.7695)		
		$S_{9,3}$	$R_{9,3}$(j)	$A_{9,3}$(36.1708)	$c_{9,3}$(24.1015)		
		$S_{9,4}$	$R_{9,4}$(i)	$A_{9,4}$(55.6319)	$c_{9,4}$(22.8924)		
E_{10}	l_{10}(4)	$S_{10,1}$	$R_{10,1}$(a)	$A_{10,1}$(56.0604)	$c_{10,1}$(23.4524)	$Rel_{E_{10}}$(0.9175)	α_{10}(1.1889)
		$S_{10,2}$	$R_{10,2}$(f)	$A_{10,2}$(40.8118)	$c_{10,2}$(18.0982)		
		$S_{10,3}$	$R_{10,3}$(h)	$A_{10,3}$(46.046)	$c_{10,3}$(18.8102)		
		$S_{10,4}$	$R_{10,4}$(j)	$A_{10,4}$(54.0672)	$c_{10,4}$(16.8409)		

Table 6
Task information for Fig. 4 (the numbers in parentheses indicate the values (or codes) of the corresponding parameters).

T_k	n_k	$s_{k,u}$	$r_{k,u}$	$a_{k,u}$	$t_{k,u}$	$W_k^{u,u+1}$
T_1	n_1(4)	$s_{1,1}$	$r_{1,1}$(h)	$a_{1,1}$(1)	$t_{1,1}$(136)	$W_1^{1,2}$(612.903)$W_1^{2,3}$ (829.633)$W_1^{3,4}$(202.222)
		$s_{1,2}$	$r_{1,2}$(j)	$a_{1,2}$(1)	$t_{1,2}$(112)	
		$s_{1,3}$	$r_{1,3}$(d)	$a_{1,3}$(1)	$t_{1,3}$(136)	
		$s_{1,4}$	$r_{1,4}$(f)	$a_{1,4}$(1)	$t_{1,4}$(207)	
T_2	n_2(4)	$s_{2,1}$	$r_{2,1}$(j)	$a_{2,1}$(1)	$t_{2,1}$(136)	$W_2^{1,2}$(271.926)$W_2^{2,3}$ (732.78)$W_2^{3,4}$(843.11)
		$s_{2,2}$	$r_{2,2}$(a)	$a_{2,2}$(1)	$t_{2,2}$(167)	
		$s_{2,3}$	$r_{2,3}$(g)	$a_{2,3}$(1)	$t_{2,3}$(83)	
		$s_{2,4}$	$r_{2,4}$(e)	$a_{2,4}$(1)	$t_{2,4}$(159)	
T_3	n_3(4)	$s_{3,1}$	$r_{3,1}$(d)	$a_{3,1}$(1)	$t_{3,1}$(243)	$W_3^{1,2}$(325.297)$W_3^{2,3}$ (363.726)$W_3^{3,4}$(219.507)
		$s_{3,2}$	$r_{3,2}$(f)	$a_{3,2}$(1)	$t_{3,2}$(212)	
		$s_{3,3}$	$r_{3,3}$(b)	$a_{3,3}$(1)	$t_{3,3}$(126)	
		$s_{3,4}$	$r_{3,4}$(a)	$a_{3,4}$(1)	$t_{3,4}$(134)	
T_4	n_4(4)	$s_{4,1}$	$r_{4,1}$(e)	$a_{4,1}$(1)	$t_{4,1}$(113)	$W_4^{1,2}$(460.482)$W_4^{2,3}$ (748.088)$W_4^{3,4}$(747.087)
		$s_{4,2}$	$r_{4,2}$(j)	$a_{4,2}$(1)	$t_{4,2}$(85)	
		$s_{4,3}$	$r_{4,3}$(a)	$a_{4,3}$(1)	$t_{4,3}$(171)	
		$s_{4,4}$	$r_{4,4}$(f)	$a_{4,4}$(1)	$t_{4,4}$(236)	
T_5	n_5(4)	$s_{5,1}$	$r_{5,1}$(i)	$a_{5,1}$(1)	$t_{5,1}$(210)	$W_5^{1,2}$(484.652)$W_5^{2,3}$ (308.524)$W_5^{3,4}$(606.36)
		$s_{5,2}$	$r_{5,2}$(h)	$a_{5,2}$(1)	$t_{5,2}$(218)	
		$s_{5,3}$	$r_{5,3}$(d)	$a_{5,3}$(1)	$t_{5,3}$(193)	
		$s_{5,4}$	$r_{5,4}$(c)	$a_{5,4}$(1)	$t_{5,4}$(212)	
T_6	n_6(4)	$s_{6,1}$	$r_{6,1}$(h)	$a_{6,1}$(1)	$t_{6,1}$(211)	$W_6^{1,2}$(566.588)$W_6^{2,3}$ (438.557)$W_6^{3,4}$(623.743)
		$s_{6,2}$	$r_{6,2}$(e)	$a_{6,2}$(1)	$t_{6,2}$(130)	
		$s_{6,3}$	$r_{6,3}$(g)	$a_{6,3}$(1)	$t_{6,3}$(58)	
		$s_{6,4}$	$r_{6,4}$(b)	$a_{6,4}$(1)	$t_{6,4}$(110)	
T_7	n_7(4)	$s_{7,1}$	$r_{7,1}$(c)	$a_{7,1}$(1)	$t_{7,1}$(86)	$W_7^{1,2}$(687.906)$W_7^{2,3}$ (567.052)$W_7^{3,4}$(922.141)
		$s_{7,2}$	$r_{7,2}$(b)	$a_{7,2}$(1)	$t_{7,2}$(215)	
		$s_{7,3}$	$r_{7,3}$(j)	$a_{7,3}$(1)	$t_{7,3}$(224)	
		$s_{7,4}$	$r_{7,4}$(g)	$a_{7,4}$(1)	$t_{7,4}$(182)	
T_8	n_8(4)	$s_{8,1}$	$r_{8,1}$(c)	$a_{8,1}$(1)	$t_{8,1}$(120)	$W_8^{1,2}$(457.112)$W_8^{2,3}$ (318.192)$W_8^{3,4}$(936.595)
		$s_{8,2}$	$r_{8,2}$(f)	$a_{8,2}$(1)	$t_{8,2}$(199)	
		$s_{8,3}$	$r_{8,3}$(e)	$a_{8,3}$(1)	$t_{8,3}$(199)	
		$s_{8,4}$	$r_{8,4}$(h)	$a_{8,4}$(1)	$t_{8,4}$(83)	
T_9	n_9(4)	$s_{9,1}$	$r_{9,1}$(i)	$a_{9,1}$(1)	$t_{9,1}$(175)	$W_9^{1,2}$(877.731)$W_9^{2,3}$ (861.812)$W_9^{3,4}$(697.501)
		$s_{9,2}$	$r_{9,2}$(c)	$a_{9,2}$(1)	$t_{9,2}$(196)	
		$s_{9,3}$	$r_{9,3}$(e)	$a_{9,3}$(1)	$t_{9,3}$(131)	
		$s_{9,4}$	$r_{9,4}$(a)	$a_{9,4}$(1)	$t_{9,4}$(216)	
T_{10}	n_{10}(4)	$s_{10,1}$	$r_{10,1}$(g)	$a_{10,1}$(1)	$t_{10,1}$(80)	$W_{10}^{1,2}$(594.983)$W_{10}^{2,3}$ (383.868)$W_{10}^{3,4}$(383.868)
		$s_{10,2}$	$r_{10,2}$(c)	$a_{10,2}$(1)	$t_{10,2}$(213)	
		$s_{10,3}$	$r_{10,3}$(b)	$a_{10,3}$(1)	$t_{10,3}$(94)	
		$s_{10,4}$	$r_{10,4}$(j)	$a_{10,4}$(1)	$t_{10,4}$(165)	

sive scheduling method taking into account of the interest of all parties involved.

Table 8
Default parameters for simulation experiments (except for special statements).

Variable	Value	Unit	Type
I	10		Integer
J	10		Integer
K	10		Integer
l_i	4		Integer
$A_{i,s}$	[30,60]		Decimal
At_k	1		Integer
$c_{i,s}$	[15,25]	Money Unit (MU, e.g. yuan)	Decimal
α_i	[0.5,1.5]		Decimal
Rel_{E_i}	[0.8,1.0]		Decimal
$d_{ii'}$	[125,500]	km	Decimal
n_k	4		Integer
$a_{k,u}$	1		Decimal
$W_k^{u,u+1}$	[200,1000]	kg	Decimal
c_l	0.005	MU/(kg km)	Decimal
t_l	0.008	p/km	Decimal
$t_{k,u}$	[50,250]	p	Integer
p_δ	1.0		Decimal
w_T	0.4		Decimal
w_C	0.3		Decimal
w_{Rel}	0.3		Decimal
$Cons^T$	22	p	Integer

According to the analysis above, task T_k's utility consists of QoS utility and non-QoS utility. QoS utility of T_k depends on the services that are selected for fulfilling it. As T_k requires more than one service for its completion, searching for multiple services for T_k is a service composition problem. As three criteria (including time, cost, reliability) of services are considered, the overall QoS utility of a service composition solution for task T_k consists of three parts: time utility Q_k^T, cost utility Q_k^C, reliability utility Q_k^{Rel}. The non-QoS utility of T_k refers to the effect of scheduling it on system performance such as TCT and SU, and hence the non-QoS utility includes TCT utility Q_k^{TCT} and SU utility Q_k^{SU}.

The simple additive weighting (SAW) technique is employed to compute the QoS utility of T_k [8]. There are two steps for applying SAW: scaling and aggregating. First of all, QoS parameter values are scaled into real values between 0 and 1. Second, the scaled QoS values are multiplied with a weight and then summed. The utility of index x (x stands for *T, C, Rel, TCT, and SU*) Q_k^x can thus be calculated as follows:

$$Q_k^x = \begin{cases} w_x \dfrac{q_{x,max} - q_x}{q_{x,max} - q_{x,min}}, & q_{x,max} - q_{x,min} \neq 0 \\ w_x, & q_{x,max} - q_{x,min} = 0 \end{cases} \quad (1)$$

where $q_{x,max}$ and $q_{x,min}$ denote the maximum and minimum values of the indexes of all possible service composition solutions, respectively, and w_x is the weight of the corresponding indexes. Thus, the overall QoS utility of a composition solution for T_k is $Q_k = \sum_x Q_k^x$ with the constraint $\sum_x w_x = 1$.

Table 7
Geographical distance $d_{ii'}$ between enterprises for Fig. 4.

	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}
E_1	0	378.22	342.055	498.592	490.444	126.362	333.884	453.639	415.448	360.595
E_2	378.22	0	339.64	227.851	468.059	202.639	436.598	324.236	497.459	343.257
E_3	342.055	339.64	0	275.437	184.671	433.439	413.835	261.887	129.532	139.809
E_4	498.592	227.851	275.437	0	410.745	330.885	394.528	475.6	371.25	215.537
E_5	490.444	468.059	184.671	410.745	0	183	184.9	187.864	425.989	214.084
E_6	126.362	202.639	433.439	330.885	183	0	299.482	135.414	340.075	434.664
E_7	333.884	436.598	413.835	394.528	184.9	299.482	0	174.669	247.41	496.189
E_8	453.639	324.236	261.887	475.6	187.864	135.414	174.669	0	125.595	217.586
E_9	415.448	497.459	129.532	371.25	425.989	340.075	247.41	125.595	0	240.989
E_{10}	360.595	343.257	139.809	215.537	214.084	434.664	496.189	217.586	240.989	0

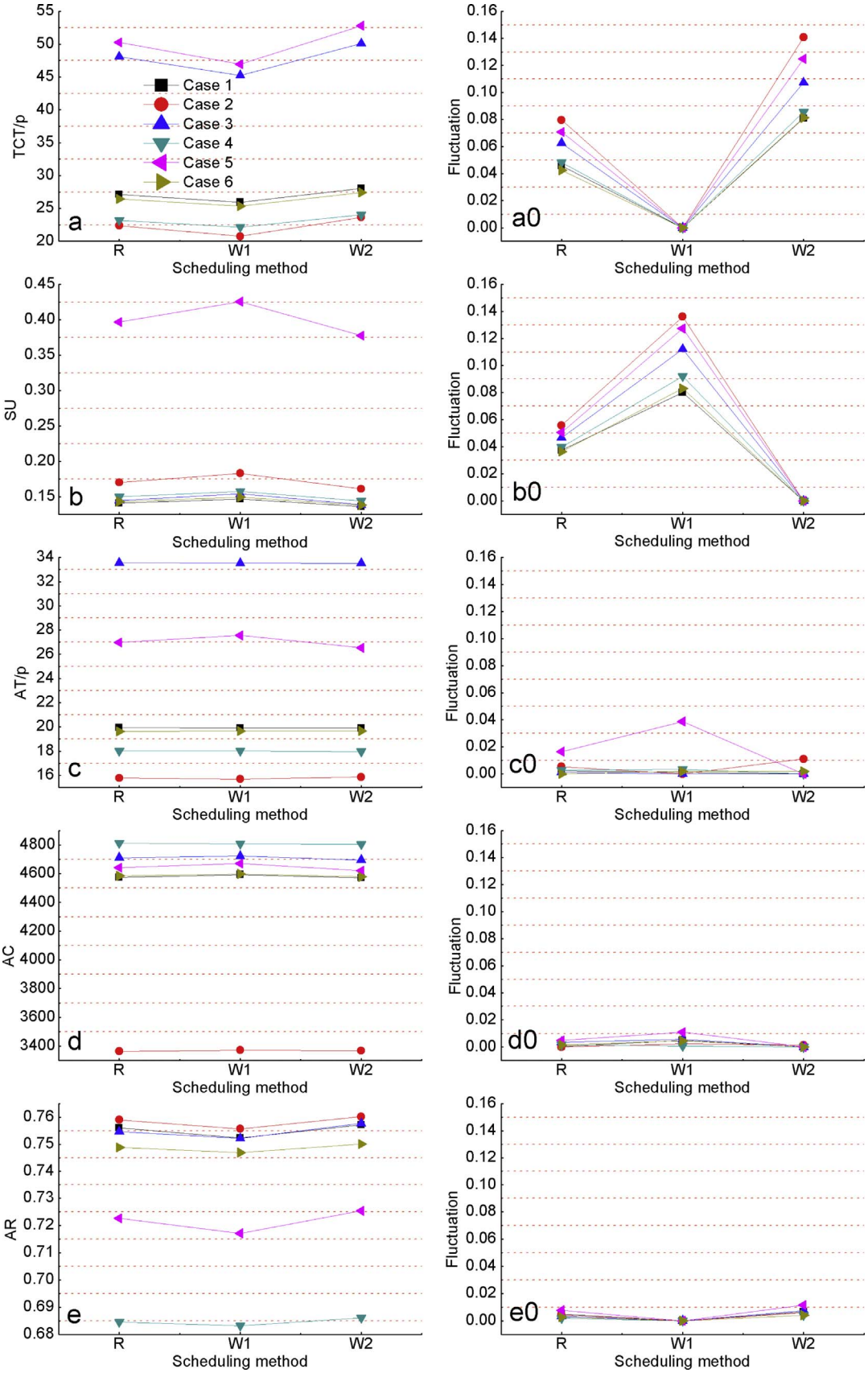

Fig. 5. Effects of different scheduling methods on system performance (without time constraint), and fluctuations of the results.

5. A concrete example for the multi-task scheduling model

Fig. 4 presents a multi-task scheduling example. In this example, there are 10 enterprises, i.e. E_1 to E_{10}. Each enterprise provides 4 different types of services, and each type of service is provided by 4 enterprises (the vertical axis). The letters (from a to j) in the parentheses for $R_{i,s}$ indicate the type of the services. There are 10 tasks (numbering from 1 to 10) in the scheduling scenario shown in Fig. 4, and each task has four subtasks with a sequence structure. In Fig. 4. tasks are scheduled in an ascending order of the numberings. For example, T_5 has four subtasks $s_{5,1}$, $s_{5,2}$, $s_{5,3}$, and $s_{5,4}$. The detailed enterprise-, service-, and task-related parameters for Fig. 4 are shown in Tables 5–7, respectively (as logistics is an important characteristic of cloud manufacturing, and in order to highlight the role of logistics in task scheduling, the distances between enterprises are extended on the basis of the enterprise distances shown in Table 2). Other relevant parameters for Fig. 4 are p_δ=1.0, c_l=0.005, t_l=0.008, w_T=0.4, w_C=0.3, and w_{Rel}=0.3.

In Fig. 4, the various types of times are illustrated taking T_5 as an example. The completion time of T_5 is 19 periods, including 16 periods of service time ST_5, 2 periods of logistics time $LT_5^{1,2}$, and 1 periods of waiting time $WT_{5,3}$. The total completion time TCT of all tasks is 19 periods. The waiting time for performing $s_{5,3}$ can be identified as follows. Subtasks $s_{5,3}$ and $s_{1,3}$ are both performed by E_8 with $S_{8,3}$. Due to $s_{1,3}$ is completed at p=10, which is one period later than the completion of $s_{5,2}$. As a result, the execution of $s_{5,3}$ has to wait for one period and can only start at p=11.

6. Simulation experiments

6.1. Simulation setup

The default simulation parameters are shown in Table 8. The interval parameters such as $A_{i,s}$∈[30,60] follow the uniform distribution. In Table 8, only the atomic variables are shown and the composite variables such as $Cap_{i,s}$ and $wl_{k,u}$ can be derived from these atomic variables according to their definitions. For example, $Cap_{i,s}$=$A_{i,s}$×α_i, $A_{i,s}$∈[30,60], and α_i∈[0.5,1.5], then $Cap_{i,s}$∈[15,90].

6.2. Simulation results

In the following, simulation results without and with time constraint are presented for the different scheduling methods. The simulations are based on Monte Carlo methods. The Monte Carlo simulation programs are written in C/C++ language using Microsoft Visual Studio 2010. Results shown in Figs. 5 and 8 are obtained by averaging over 5000 simulation realizations, and for each simulation realization, all variable values regarding enterprises, services, and tasks (except for resource codes) are randomly generated.

6.2.1. Without time constraint

The following six cases are taken into account to examine the robustness of the results.

1. Case 1 (C1) is a general case, which acts as a benchmark for comparing the results obtained in all cases.
2. Logistics is excluded in Case 2 (C2) to check whether logistics can have an effect on the scheduling results.
3. In Case 3 (C3), the service time of subtasks has a wider distribution range, which leads to a greater difference between task service times.
4. In Case 4 (C4), the time preference of tasks is enhanced to reflect that the strong time preference of customers.
5. In Case 5 (C5), more tasks are considered to check the robustness of the results against the task number variation.
6. In all cases above, the scheduling process is completely customer-centric in the sense that service scheduling is aimed at meeting customers' requirements without incorporating the interests of the platform and/or providers. Differently, Case 6 (C6) takes also TCT and SU as the optimization objectives. The former is an important index of system performance while the latter, to some extent, reflects the interests of service providers. C6 is therefore a comprehensive scheduling strategy in the sense that it considers the interest of all stakeholders.

Fig. 5 shows the simulation experiment results with no time constraint (a, b, c, d, and e), and the fluctuations of the results (a0, b0, c0, d0, and e0). There is a one-to-one correspondence between the results for a, b, c, d, e and that for a0, b0, c0, d0, e0. The fluctuation is calculated for each case, and in each case, the fluctuation of the scheduling result x for a certain scheduling method is calculated as follows: (x-miniValue)/miniValue, where miniValue denotes the minimum value of the scheduling result corresponding to a scheduling method in each case. In this way, we can observe the fluctuations of the results relative to the minimum values.

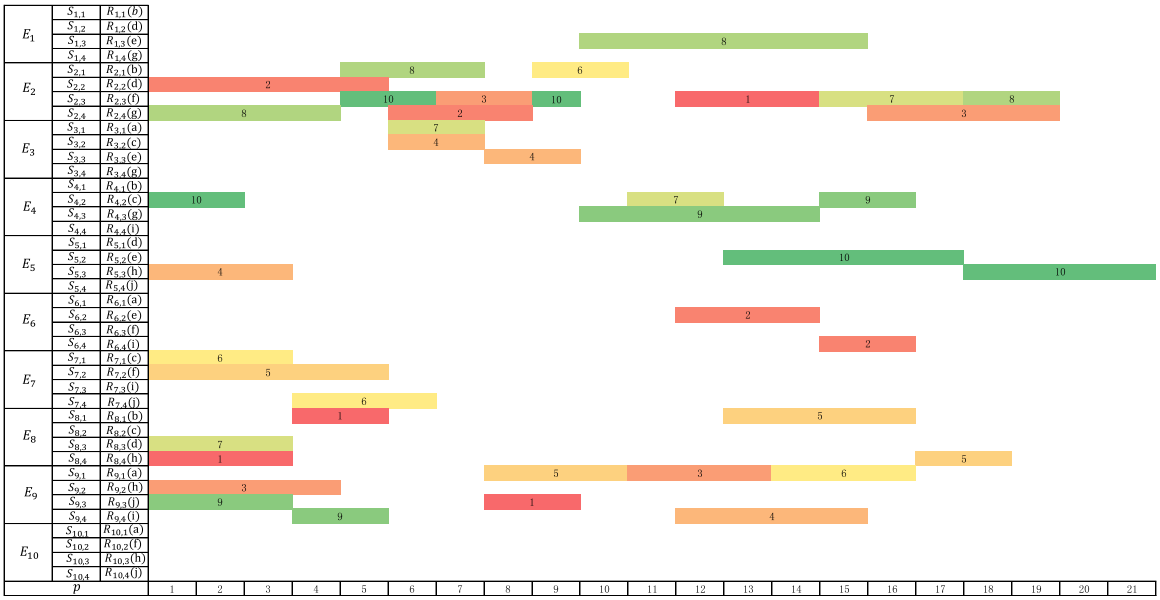

Fig. 6. Diagram of the random scheduling method without time constraint.

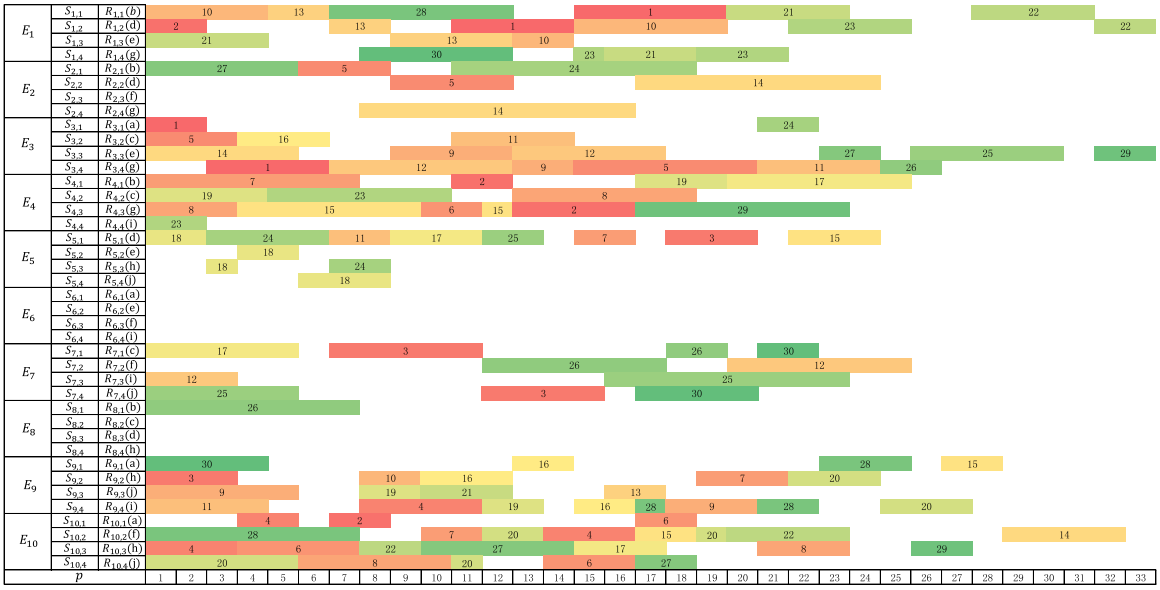

Fig. 7. Diagram of the random scheduling method for K=30 without time constraint.

Fig. 5a shows that overall W1 leads to the shortest TCT among all scheduling methods in all the cases, which is followed by R, and W2 gives rise to the longest TCT. The results indicate that scheduling heavier workload tasks with a higher priority can effectively shorten TCT. Although TCT fluctuates with the change of scheduling methods, the magnitudes of the fluctuations are different for different cases (Fig. 5a0). This indicates that the effects of the scheduling methods on TCT are also influenced by other factors. In addition, different cases lead to quite different TCTs. Specifically, C3 and C5 lead to the longest TCT, and C2 and C4 lead to the shortest TCT. C3 has a larger TCT because each task needs a longer service time. For C5, it is because more tasks usually need a longer time to be completed (see Figs. 6 and 7 for comparison). When no logistics is involved (C2), TCT is shortened because there is no logistics time. In Figs. 6 and 7, there is an overall trend that adjacent subtasks are performed by the same enterprise. This is because fulfilling adjacent subtasks within the same enterprise does not involve logistics time and cost. This indicates the great impact of logistics on task scheduling. In fact, only when an enterprise cannot provide all the required services or the provided services are not good enough for fulfilling a task will some of the subtasks be performed by other enterprises [29]. When customers have a strong time preference, the service composition solutions with a shorter completion time are more preferable. As a consequence, TCT is decreased for C4. It should also be noted that C6 leads to a slight decline in TCT in comparison with C1. This is because TCT has a weight in the total optimization objective in C6, which makes the solutions with a shorter TCT have a higher probability to be selected.

Fig. 5b illustrates that W1 gives rise to a higher SU than R and W2. It is apparent that SU is closely related to TCT, i.e. a smaller TCT leads to a higher SU. For the effects of different cases on SU, C5 leads to a higher SU because more tasks need more services (Fig. 7). When there is no logistics (C2) or customers have a strong time preference (C4), higher SUs can be obtained, which is mainly attribute to the shorter TCT (Fig. 5a). Similarly, C6 has the similar SU as C1. It should be noted that C3 has a larger TCT, but it has the similar SU as C1 and C6, which is because with the increase of TCT services are used for a longer time, which increases the SU accordingly.

We have also presented the results for AT, AC and AR, respectively, which reflect the degree of customers' satisfaction, as shown in Fig. 5c–e, respectively. Fig. 5c shows that ATs in all cases almost do not change for the different scheduling methods (with the exception of C5) (Fig. 5c0). In C5, W1 leads to a slightly higher AT in comparison with other scheduling methods. Regarding the effects of the different cases, one can find that generally the cases leading to a shorter TCT also have a shorter AT, with the exception of C3 and C5 where the order is reversed, i.e. C3 has a shorter TCT but it has a longer AT. This is because of the longer service time of C3, while the longer TCT for C5 in Fig. 5a is just because more tasks are scheduled in the system. However, due to service occupancy, the AT is still prolonged in comparison with that in C1 (Fig. 7). Fig. 5c shows that AT can be effectively decreased when there is no logistics time. It is also shown that C4 leads to a lower AT. This is because in this case the service composition solution with a shorter service time is selected for all tasks, which effectively decreases AT. Corresponding to Fig. 5a, C6 has a slightly shorter AT than C1.

Fig. 5d indicates that, except for C5, AC almost does not change with the variation of scheduling methods for all other cases (Fig. 5c0). The reason for the higher AC for C3 (compared with that for C1) is that the longer service time increases the service cost. The higher AC for C5 (compared with that for C1) is attribute to the fact that more tasks need more services so that the services with a higher cost may also be selected in the service composition solutions (the reason is the same for the lower AR of C5 in Fig. 5e). C2 has the lowest AC because there is no logistics cost. The result also shows that a strong time preference leads to a higher AC (e.g. C4). This is because when customers pay more attention to the time aspect, the services with a higher cost may be selected (the lowest AR for C4 in Fig. 5e has the same rationale).

Fig. 5e and e0 shows the less obvious fluctuation of AR versus the different scheduling methods. As far as the effects of different cases are concerned, C4 leads to the lowest AR and C5 brings about a lower AR (the reasons have been analysed above). The reason for C6 to lead to a slight lower AR than C1 is the decrease of the reliability weight. When there is no logistics (C2), AR is slightly increased compared with that of C1. The reason is complicated, but qualitative analysis can be conducted as follows. When there is no logistics, the system can select services completely based on QoS of services themselves without considering the influence of logistics, which leads to the change of AT and AC (i.e. the decreased AT and decreased AC). According to Eq. (1), this will give rise to a change in the calculation of QoS utility for service composition solutions and of course the change of AR of the selected services.

According to Fig. 5, we come to the conclusion that that when there is no time constraint and the number of tasks is not very large (so that services are abundant relative to the number of tasks), W1 can lead to

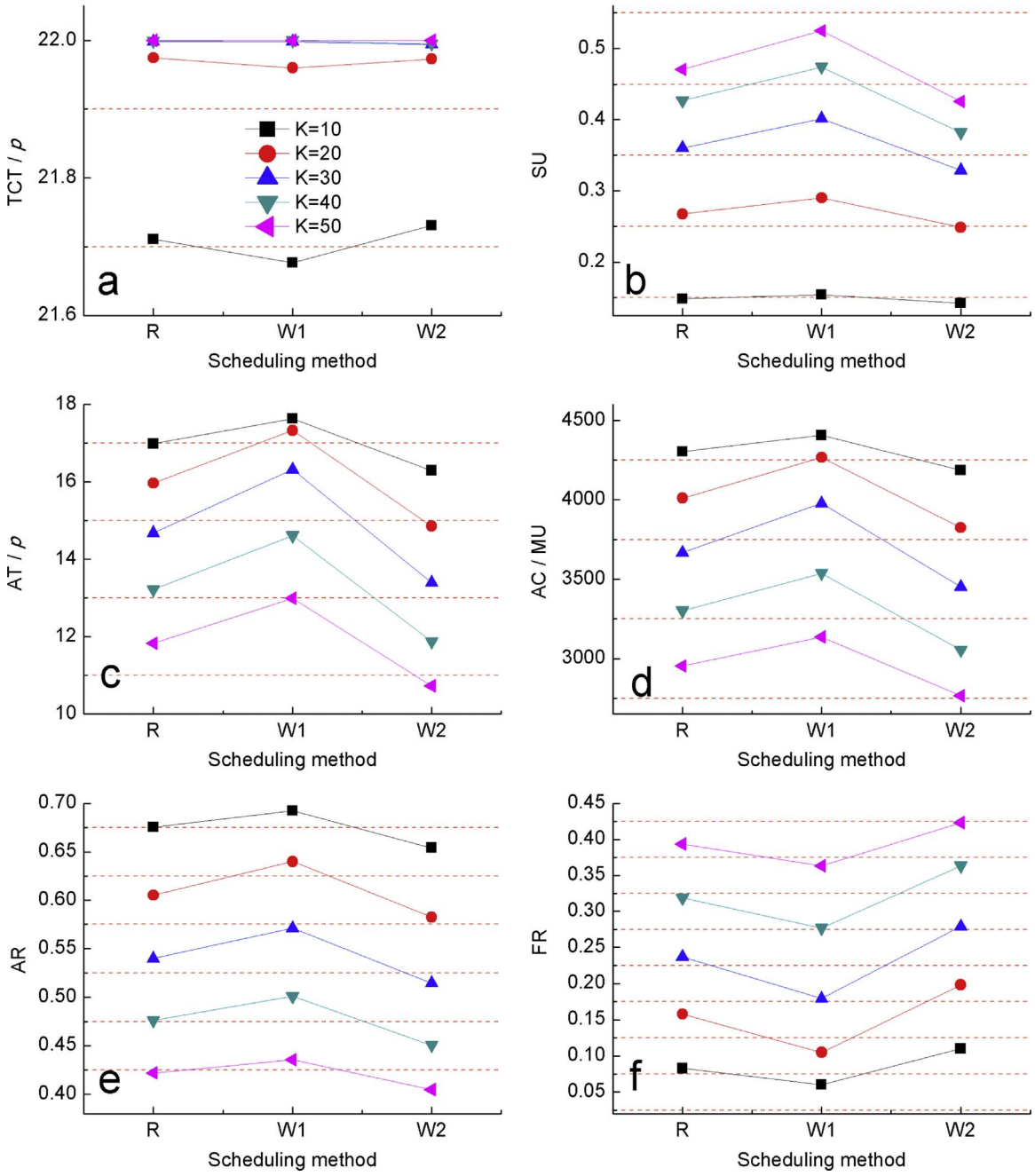

Fig. 8. Effects of different scheduling methods with time constraint $Cons^{T}_{T_k}$=22. Note that the results presented above are only for the successfully executed tasks.

the best results (i.e. a shortest TCT and a highest SU) among all the scheduling methods without apparently deteriorating the task fulfilment quality (because the AT, AC, and AR for W1 are almost the same as that for R and W2 (it can be observed from Fig. 5 that the results for AT, AC, and AR do not drastically fluctuate with the change of the scheduling methods). The results for W2 are the worst, with the result for R being in between. Only the number of tasks is large (so that the quantity of services is relatively not so abundant), W1 can also lead to a shortest TCT, but at the cost of the slightly increased AT, slightly increased AC as well as the slightly decreased AR (Fig. 5c0–e0).

6.2.2. With time constraint

In the real-world situations, customers' requirements are usually accompanied by some constraints on the delivery date, cost, etc. Here, we consider only time constraint. In order to explore the effect of time constraint on system performance for different scheduling methods, a unifying time constraint $Cons^{T}_{T_k}$=22 is introduced for all tasks. It should be noted that when time constraint is introduced, there is a possibility that no service composition solution can meet the time constraint of T_k. In this case, the execution of T_k is regarded as a failure.

Fig. 8 shows the effects of different scheduling methods on the system performance with time constraint $Cons^{T}_{T_k}$=22. Fig. 8a shows that TCT approaches 22 as K increases. It is normal because more tasks need a longer time to be completed due to service occupancy (Figs. 9 and 10). Hence, as the task number increases, TCT increases, and when $K \geq 30$, TCT is equal to 22. For K=10 and 20, the advantage of W1 in shortening TCT can also be observed.

Fig. 8b shows that W1 gives rise to the highest SU among all scheduling methods (especially for larger values of K). As analysed in Fig. 5, when there is no time constraint, shortening TCT can effectively increase SU. However, with the increase of K, TCT is almost equal to the time constraint, and there is therefore almost no room for short-

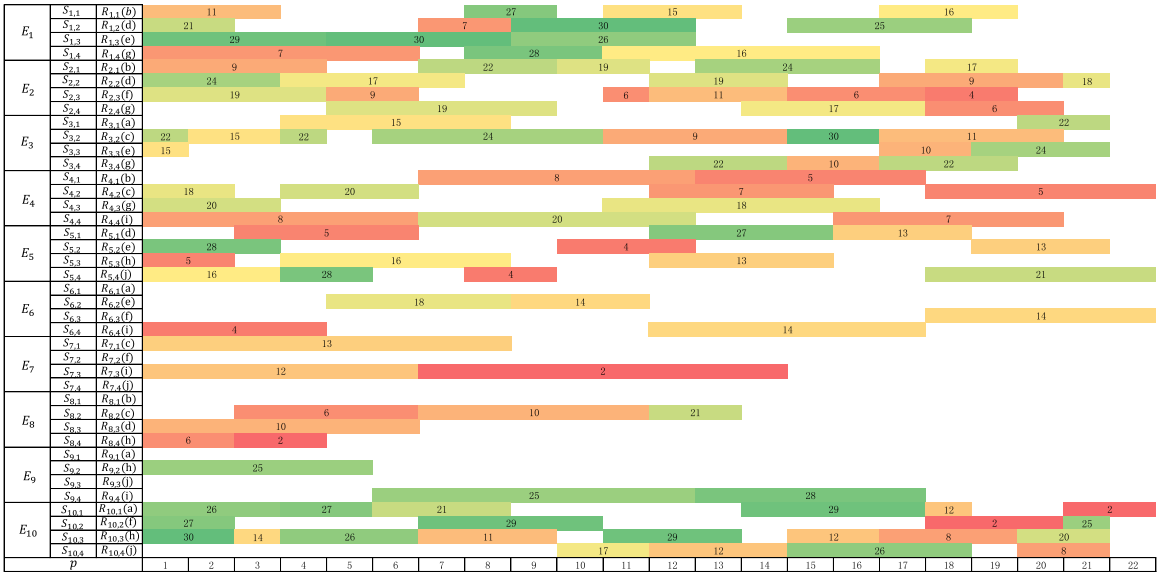

Fig. 9. Diagram of task scheduling for W1 with time constraint $Cons^{T}_{T_k}$=22. For this diagram, K=30, SU=0.4295, FR=0.1, respectively.

ening TCT. Hence, the increased SUs for W1, A1 and T1 should be attribute to the increased number of tasks in the system. As shown in Fig. 8f, W1 leads to a lower FR compared with W2, meaning that more tasks can be executed successfully under the same time constraint. In order to highlight this point intuitively, task scheduling diagrams for W1 and W2 with the time constraint are depicted in Figs. 9 and 10, where it can be visually observed that SU for W1 is higher than that of W2. For the effects of different numbers of tasks, SU increases as K increases. This is because more tasks require more services, which can be visually observed from Figs. 9 and 10.

Fig. 8c and d shows that AT and AC for W1 are higher than that for other scheduling methods, while Fig. 8e indicates the increased ARs for W1. Why does W1 lead to a higher AT, a higher AC, and also a higher AR? Explaining this phenomenon is helpful for understanding whether the higher SUs and the lower FRs of W1 are at the cost of a higher AT and a higher AC. It is undoubtedly related to the task scheduling order. Fig. 11 shows the successfully and unsuccessfully executed tasks for W1 and W2, respectively, with time constraint (note that the data on enterprises, services and tasks for W1 and W2 in Fig. 11 are completely the same in the simulations). In Fig. 11, tasks with a larger workload

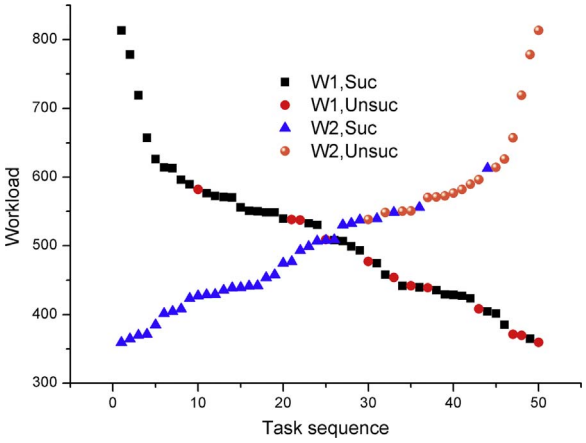

Fig. 11. Successfully (Suc) and unsuccessfully (Unsuc) executed tasks for W1 and W2, respectively, with time constraint. The number of tasks is K=50.

are scheduled first for W1 while for W2 tasks are scheduled in the

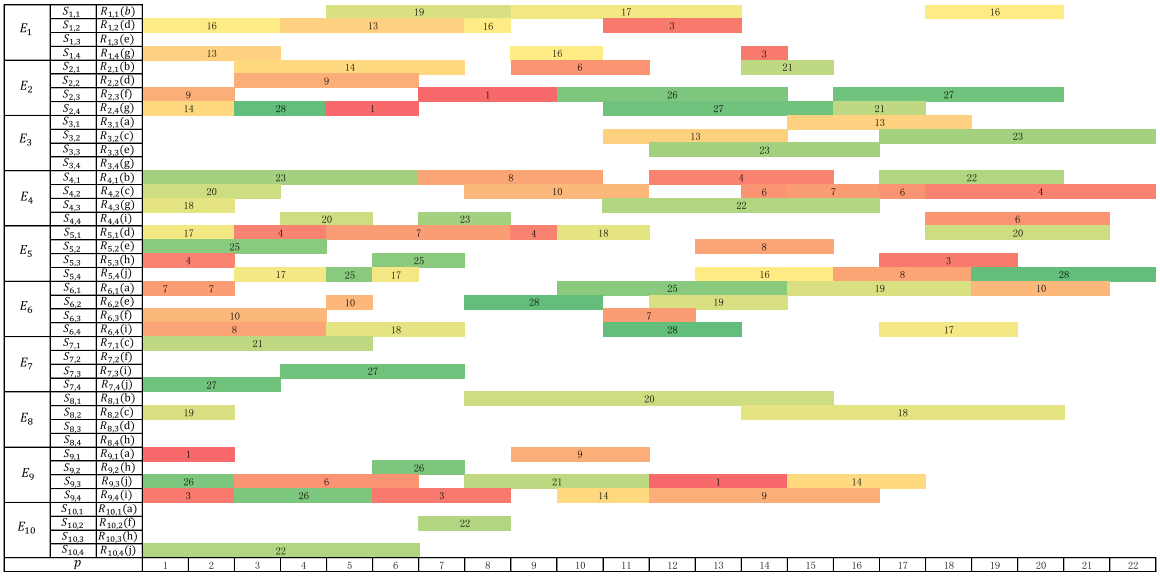

Fig. 10. Diagram of task scheduling for W2 with time constraint $Cons^{T}_{T_k}$=22. For this diagram, K=30, SU=0.3386, FR=0.2667, respectively.

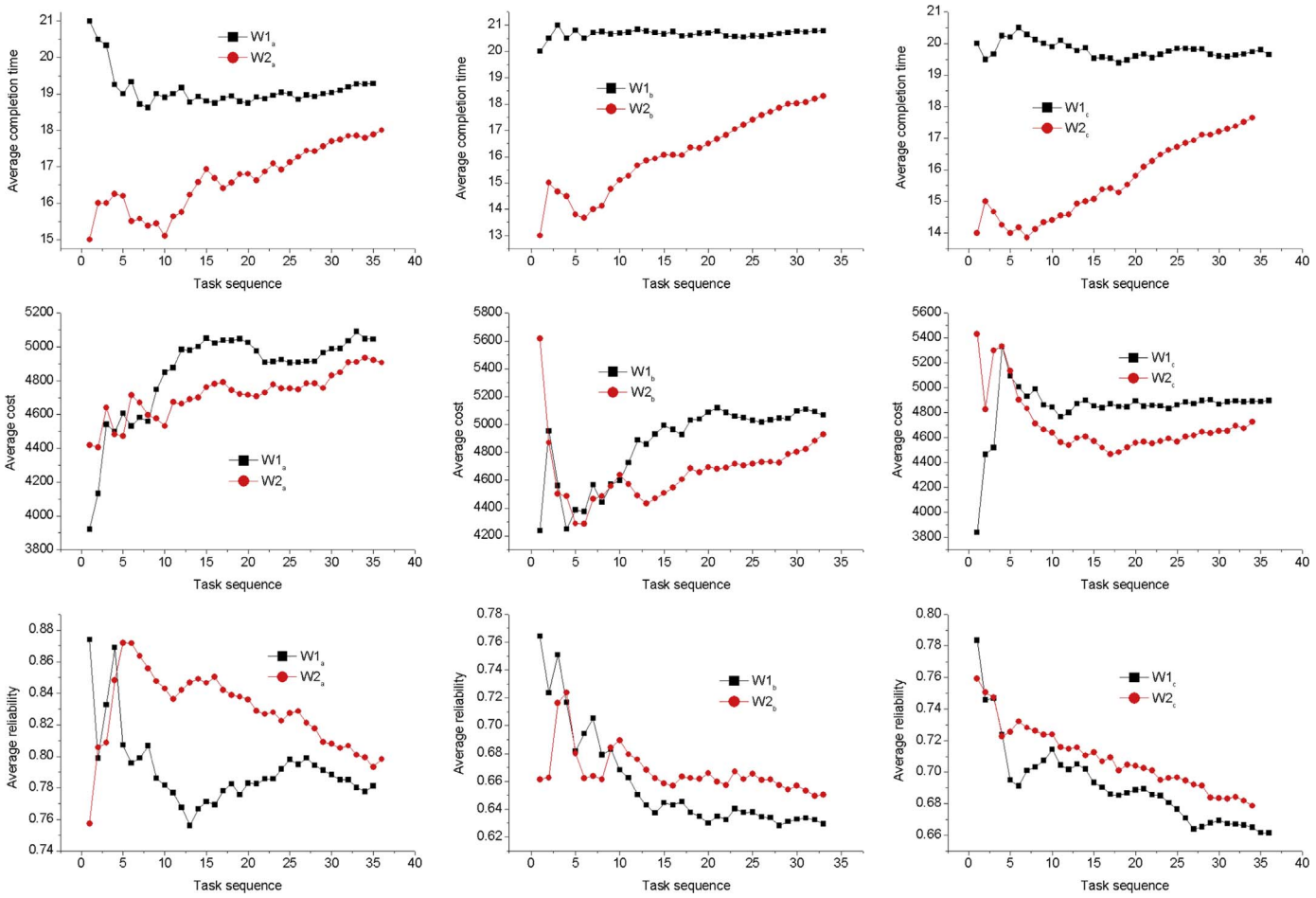

Fig. 12. Evolution of average completion time, average cost, and average reliability of the successfully executed tasks in the task queues of W1 and W2. Time constraint is considered, and the number of tasks is K=50.

reverse order. In addition, most of the tasks that have been successfully executed for W1 are the large tasks while the opposite holds for W2, which makes the successfully executed tasks for W1 (the average workload is 530.956) have a larger average workload than W2 (the average workload is 460.941).

We have also examined the evolutions of AT, AC and AR during task scheduling processes (Fig. 12). Three groups (i.e. a, b, c) of data are presented for comparison, and for each group, the data on enterprises, services and tasks are completely the same for W1 and W2 in the simulations. As shown in Fig. 12, tasks for W1 always take a longer time, and in most cases they also have a higher AC during the scheduling processes. This is consistent with the results shown in Fig. 11 because tasks with a heavier workload usually take a longer time and need a larger quantity of services, and thus a higher cost. One may notice that AR for W1 is almost always lower than that for W2 during the evolutionary processes, which seems to contradict the result shown in Fig. 8e (where W1 has a higher AR). We have further examined this phenomena and found that for a concrete simulation experiment with completely the same data, AR for W1 is higher than that for W2 with a high probability (not always), but statistically, W1 has a higher AR than W2 (in this case the data on enterprises, services and tasks are usually different for W1 and W2). The evolutionary processes of AT, AC and AR demonstrate different degrees of stochasticity. The curves for AT have some monotonous behaviour as more

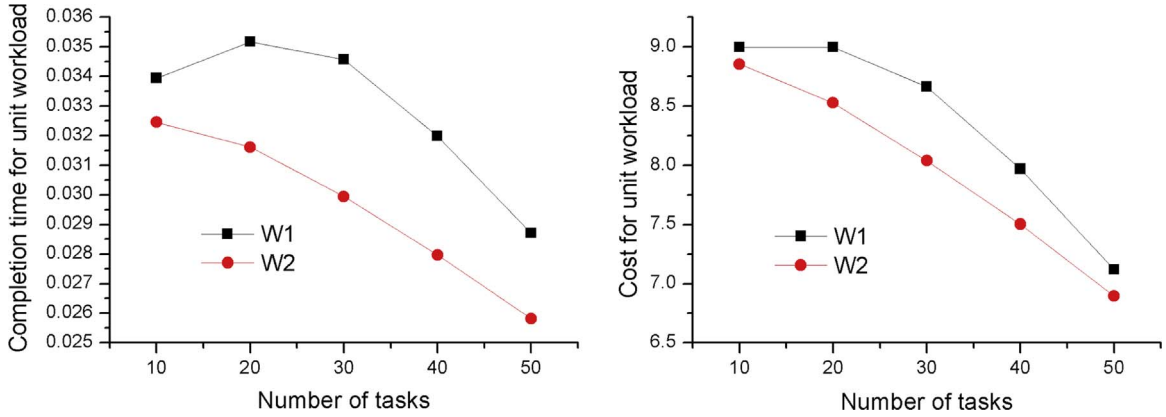

Fig. 13. Completion time and cost per workload unit for W1 and W2 with time constraint.

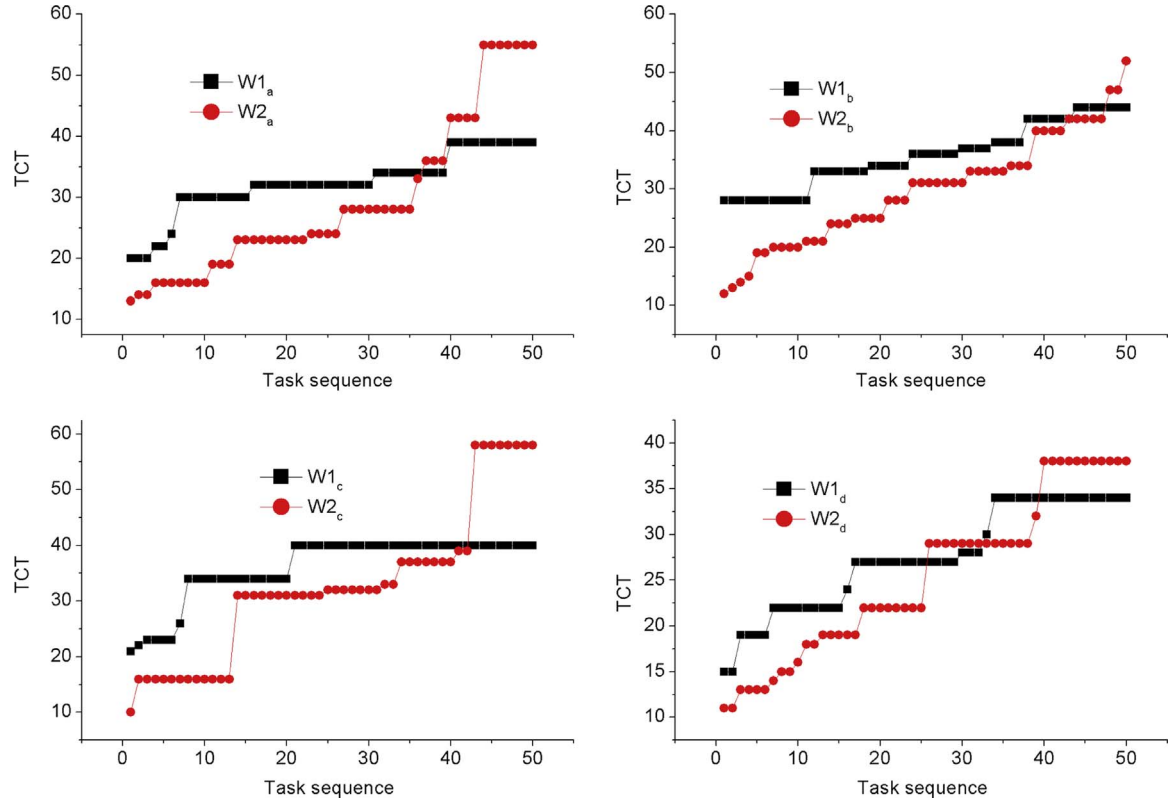

Fig. 14. Evolution of TCT with the increase of the number of tasks in the task queues of W1 and W2. No time constraint is considered here, and the number of tasks is K=50.

Table 9
Parameter values for Case 1 to Case 6.

Case	K	w_T	w_C	w_{Rel}	w_{TCT}	w_{SU}	p_δ	$t_{k,u}$
1	10	0.4	0.3	0.3	0	0	1.0	[50,250]
2	10	0.4	0.3	0.3	0	0	0	[50,250]
3	10	0.4	0.3	0.3	0	0	1.0	[50,550]
4	10	0.8	0.1	0.1	0	0	1.0	[50,250]
5	50	0.4	0.3	0.3	0	0	1.0	[50,250]
6	10	0.3	0.2	0.2	0.2	0.1	1.0	[50,250]

tasks are scheduled while that for AC and AR exhibit a relatively strong stochastic behaviour. This is because under time constraint, the selection of service composition solutions focuses on the time aspect, leaving the cost and reliability indexes being less considered. This means that the selected services need to have a shorter service time but are not necessarily have a lower cost and/or a high reliability, and vice versa for the unselected services.

In order to accurately evaluate the effects of different scheduling methods, the completion time and cost per workload are presented, as shown in Fig. 13. In the calculation, for each successfully executed task, the completion time and cost of a service composition solution are normalized by its workload. It can be observed that W2 has a slightly lower normalized completion time and cost, but the difference is quite small. Therefore, it can be concluded that in the case where the time constraint is considered, W1 makes more tasks successfully executed (most of them are large tasks) while not greatly deteriorating the quality for fulfilling the tasks. Moreover, SU is also increased.

It has been demonstrated that when there is no time constraint, W1 leads to a shorter TCT and a higher SU without dramatically decreasing task fulfilment quality. When time constraint is considered, W1 enables more larger-workload tasks to be successfully executed within the time constraint and also increases the SU. It has also been shown that the above phenomena are closely related to the scheduling order. That is, scheduling larger-workload tasks with a higher priority can effectively decrease TCT in comparison with the opposite scheduling order. Then what is the reason for this? This is also the mechanism responsible for the increased number of successfully executed tasks with time constraint. In order to explain the phenomenon, the evolutionary process of TCT is presented (Fig. 14). Four groups (a, b, c, d) of data are presented. In order for a fair and accurate comparison, the data on enterprises, services, and tasks for W1 and W2 is completely the same. As shown in Fig. 14, W2 always starts with a smaller TCT, there is always an intersection at which TCT of W2 exceeds that of W1. The phenomenon indicates that scheduling higher-workload tasks with a higher priority can lead to the final success in achieving a shorter TCT.

The phenomenon can be explained as follows. It no doubt that the phenomenon is closely related to the different workload-based scheduling orders. First of all, we should be clear that in the case where services are limited (as in the current scenario), scheduling tasks with a higher priority means that the tasks have the chance to use “better” services (“better” services refer to those which can fulfil tasks with a shorter time, a lower cost, and a high reliability, especially in the time aspect), while the tasks scheduled later can only use the services that are not quite good. In this sense, the scheduling method W1 means using the “better” services to handle larger workload tasks and leaving smaller workload tasks to be processed with the relatively “worse” services, and W2 adopts the opposite strategy. Hence, scheduling the smaller workload tasks at a later time or scheduling the larger workload tasks at a later time can both lead to a longer service time compared with the earlier scheduling scenarios. But what is critical here is that due to the larger workload tasks usually take a longer time for their completion, the time will be prolonged more if the larger workload tasks are scheduled later than the case where they are scheduled earlier. Hence, handling larger workload tasks earlier is a better strategy that will lead to shorter makespan. When the time constraint is taken into account, as the scheduling method W1 will result in a shorter makespan, this means that more tasks can be successfully executed within the time constraint (Fig. 11).

So far, we have presented, compared and analysed the results with

and without time constraint for scheduling methods of W1 and W2 under various circumstances (Table 9). The results indicate that, when there is no time constraint, scheduling larger workload tasks with a higher priority (W1) can lead to a shorter makespan and a higher service utilization than the reverse strategy (W2). Moreover, in the case where services are abundant (relative to tasks), the quality of task fulfilment for W1 is almost the same as that for W2. When services are not so abundant, there is some decline in the task fulfilment quality (though not much). In the case where the time constraint exists, more tasks (especially the larger workload tasks) can be successfully executed (see the lower task execution failure rate shown in Fig. 8), which increases service utilization (comparing Fig. 9 with Fig. 10). We have also checked that the increased AT and AC are attributed to the larger workloads of the executed tasks. This means that the task fulfilment quality actually does not deteriorate. In addition, all the above results are independent of the different cases shown in Table 9.

7. Conclusion and discussions

In this paper, we proposed a workload-based multi-task scheduling model for cloud manufacturing. Based on this model, we explored the effects of two different workload-based task scheduling orders on system performance. In this model, we proposed new task workload and service modelling methods, which incorporate new ingredients such as service quantity, service efficiency, task workload, and enterprise capacity. Logistics as an important factor that can greatly affect task scheduling has also been taken into account. All these enable us to dynamically calculate the time for a service to fulfil a subtask as well as service utilization. Two major scenarios with and without time constraint were considered. Our results indicate that scheduling larger workload tasks with a higher priority is a better strategy (than the opposite one) in achieving better system performance such as a shorter makespan, a higher service utilization, and a higher rate of successful execution of tasks without decreasing task fulfilment quality.

The current research can provide a general guidance as to how to schedule multiple tasks with different workloads in cloud manufacturing to achieve optimized system performance under different circumstances. In the cloud manufacturing environment, tasks with different workloads will arrive continuously. How to schedule these tasks to optimize cloud manufacturing system performance is a practical issue in cloud manufacturing. Traditional service composition methods for a single task are not suitable for the multi-task scenario as they focused on a single task and did not consider the coupling relationships and the resulting mutual influence among the multiple tasks. The current work is based on the traditional service composition method, considers coupling relationships of multiple tasks, and moreover, sorts the multiple tasks according to their workload in a descending (or ascending) order. This simple extension enables us to reveal the general regularity regarding multi-task scheduling. The current method is easy to be implemented and applied. In order to apply the current method to cloud manufacturing, what needs to do it just identify tasks' workloads, and then sort the tasks according to their workloads.

Task scheduling is an important and practical issue in cloud manufacturing, and the related research is currently in its infancy. In this work, we built a cloud manufacturing multi-task scheduling model and investigated the role of workload in task scheduling. The current work can provide some insight into task scheduling in cloud manufacturing. In the future work, we will consider the continuous arrival of tasks at different times, and explore other methods for better modelling tasks, enterprises, and services for different research purposes. Using traditional intelligent algorithms to optimize task scheduling in cloud manufacturing is also an important research direction, and in this regard, the current model provides a good support.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant Nos. 61374199 and 51475032, Natural Science Foundation of Beijing, China under Grant No. 4142031, China Postdoctoral Science Foundation under Grant Nos. 2012M520139 and 2013T60052, the Fundamental Research Funds for the Central Universities under Grant No. JB140410, and the International Postdoctoral Exchange Fellowship Program under Grant No. 20140029. Special acknowledgement is given to Guangdong Zhaoqing Automotive Parts Industry Association.

References

[1] B.-H. Li, L. Zhang, S.-L. Wang, F. Tao, J.-W. Cao, X.-D. Jiang, X. Song, X.-D. Chai, Cloud manufacturing: a new service-oriented networked manufacturing model, Comput. Integr. Manuf. Syst. 16 (1) (2010) 1–7.
[2] L. Zhang, Y. Luo, F. Tao, B. Li, L. Ren, X. Zhang, H. Guo, Y. Cheng, A. Hu, Y. Liu, Cloud manufacturing: a new manufacturing paradigm, Enterp. Inf. Syst. 8 (2) (2014) 167–187.
[3] G. Adamson, L. Wang, M. Holm, P. Moore, Cloud manufacturing – a critical review of recent development and future trends, Int. J. Comput. Integr. Manuf. (2015). http://dx.doi.org/10.1080/0951192X.2015.1031704.
[4] X. Xu, From cloud computing to cloud manufacturing, Robot. Comput.-Integr. Manuf. 28 (1) (2012) 75–86.
[5] Y. Cao, S. Wang, L. Kang, Y. Cao, A TQCS-based service selection and scheduling strategy in cloud manufacturing, Int. J. Adv. Manuf. Technol. 82 (1–4) (2016) 235–251.
[6] F. Tao, Y. LaiLi, L. Xu, L. Zhang, FC-PACO-RM: a parallel method for service composition optimal-selection in cloud manufacturing system, IEEE Trans. Ind. Inform. 9 (4) (2013) 2023–2033.
[7] J. Lartigau, X. Xu, L. Nie, D. Zhan, Cloud manufacturing service composition based on QoS with geo-perspective transportation using an improved Artificial Bee Colony optimisation algorithm, Int. J. Prod. Res. 53 (14) (2015) 4380–4404.
[8] H. Jin, X. Yao, Y. Chen, Correlation-aware QoS modelling and manufacturing cloud service composition, J. Intell. Manuf. (2015). http://dx.doi.org/10.1007/s10845-015-1080-2 (in press).
[9] C.F. Jian, Y. Wang, Batch task scheduling-oriented optimization modelling and simulation in cloud manufacturing, Int. J. Simul. Model. 13 (1) (2014) 93–101.
[10] Z. Cheng, D. Zhan, X. Zhao, H. Wan, Multitask oriented virtual resource integration and optimal scheduling in cloud manufacturing, J. Appl. Math. 2014 (2014).
[11] W.LiC.ZhuL.YangE.NgaiY.Ma, Subtask Scheduling for Distributed Robots in Cloud Manufacturing, IEEE SYST. J. 2015 http://dx.doi.org/10.1109/JSYST.2015.2438054.
[12] P.KumarA.Verma, Scheduling using improved genetic algorithm in cloud computing for independent tasks, in: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ACM, 2012, 137–142.
[13] X. Wu, M. Deng, R. Zhang, B. Zeng, S. Zhou, A task scheduling algorithm based on QoS-driven in Cloud Computing, Procedia Comput. Sci. 17 (2013) 1162–1169.
[14] B.-H. Li, L. Zhang, L. Ren, X.-D. Chai, F. Tao, Y.-L. Luo, Y.-Z. Wang, C. Yin, G. Huang, X.-P. Zhao, Further discussion on cloud manufacturing, Comput. Integr. Manuf. Syst. 17 (3) (2011) 450–457.
[15] J.R. Duflou, J.W. Sutherland, D. Dornfeld, C. Herrmann, J. Jeswiet, S. Kara, M. Hauschild, K. Kellens, Towards energy and resource efficient manufacturing: a processes and systems approach, CIRP Ann.-Manuf. Technol. 61 (2) (2012) 587–609.
[16] I.G. Vidayev, N. Martyushev, A.S. Ivashutenko, A.M. Bogdan, The resource efficiency assessment technique for the foundry production, Adv. Mater. Res. 880 (2014) 141–145.
[17] J.LartigauL.NieX.XuT.Mou, Scheduling methodology for production services in Cloud Manufacturing, in: IEEE International Joint Conference on Service Sciences (IJCSS), 2012, 34–39.
[18] T. Wang, S. Guo, C.G. Lee, Manufacturing task semantic modelling and description in cloud manufacturing system, Int. J. Adv. Manuf. Technol. 71 (9–12) (2014) 2017–2031.
[19] S.-L. Wang, W.-Y. Song, L. Kang, Q. Li, L. Guo, G.-S. Chen, Manufacturing resource allocation based on cloud manufacturing, Comput. Integr. Manuf. Syst. 18 (7) (2012) 1396–1405.
[20] N. Liu, X. Li, A resource virtualization mechanism for cloud manufacturing systems, in: Enterprise Interoperability, Springer: Berlin, Heidelberg, 2012, 46–59.
[21] W.-N. Liu, B. Liu, D.-H. Sun, Multi-task oriented service composition in cloud manufacturing, Comput. Integr. Manuf. Syst. 19 (1) (2013) 199–209.
[22] W. Liu, B. Liu, D. Sun, Y. Li, G. Ma, Study on multi-task oriented services composition and optimisation with the 'Multi-Composition for Each Task' pattern in cloud manufacturing systems, Int. J. Comput. Integr. Manuf. 26 (8) (2013) 786–805.
[23] Y.WeiL.Tian, Research on cloud design resources scheduling based on genetic algorithm, in: IEEE International Conference on Systems and Informatics (ICSAI), 2012, 2651–2656.
[24] Y.LailiL.ZhangF.Tao, Energy adaptive immune genetic algorithm for collaborative design task scheduling in cloud manufacturing system, in: IEEE International

Conference on Industrial Engineering and Engineering Management (IEEM), 2011, 1912–1916.
[25] Y. Lin, C.S. Chong, Fast GA-based project scheduling for computing resources allocation in a cloud manufacturing system, J. Intell. Manuf. (2015). http://dx.doi.org/10.1007/s10845-015-1074-0.
[26] Y. Cheng, F. Tao, Y. Liu, D. Zhao, L. Zhang, L. Xu, Energy-aware resource service scheduling based on utility evaluation in cloud manufacturing system, Proc. Inst. Mech. Eng. Part B: J. Eng. Manuf. (2013) (0954405413492966).
[27] F. Tao, Y. Cheng, L. Zhang, D. Zhao, Utility modelling, equilibrium, and coordination of resource service transaction in service-oriented manufacturing system, Proc. Inst. Mech. Eng. Part B: J. Eng. Manuf. 226 (6) (2012) 1099–1117.
[28] F. Tao, D. Zhao, Y. Hu, Z. Zhou, Resource service composition and its optimal-selection based on particle swarm optimization in manufacturing grid system, IEEE Trans. Ind. Inform. 4 (4) (2008) 315–327.
[29] Y. Liu, L. Zhang, F. Tao, L. Wang, Resource service sharing in cloud manufacturing based on the Gale–Shapley algorithm: advantages and challenge, Int. J. Comput. Integr. Manuf. (2015). http://dx.doi.org/10.1080/0951192X.2015.1067916.