Centralized and Decentralized Methods of Efficient Resource Allocation in Cloud Computing

by

Su Seon Yang

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved April 2016 by the
Graduate Supervisory Committee:

Nong Ye, Chair
Teresa Wu
Rong Pan
Sik-Sang Yau

ARIZONA STATE UNIVERSITY

December 2016

ABSTRACT

Resource allocation in cloud computing determines the allocation of computer and network resources of service providers to service requests of cloud users for meeting the cloud users' service requirements. The efficient and effective resource allocation determines the success of cloud computing. However, it is challenging to satisfy objectives of all service providers and all cloud users in an unpredictable environment with dynamic workload, large shared resources and complex policies to manage them.

Many studies propose to use centralized algorithms for achieving optimal solutions for resource allocation. However, the centralized algorithms may encounter the scalability problem to handle a large number of service requests in a realistically satisfactory time. Hence, this dissertation presents two studies. One study develops and tests heuristics of centralized resource allocation to produce near-optimal solutions in a scalable manner. Another study looks into decentralized methods of performing resource allocation.

The first part of this dissertation defines the resource allocation problem as a centralized optimization problem in Mixed Integer Programming (MIP) and obtains the optimal solutions for various resource-service problem scenarios. Based on the analysis of the optimal solutions, various heuristics are designed for efficient resource allocation. Extended experiments are conducted with larger numbers of user requests and service providers for performance evaluation of the resource allocation heuristics. Experimental results of the resource allocation heuristics show the comparable performance of the heuristics to the optimal solutions from solving the optimization problem. Moreover, the

resource allocation heuristics demonstrate better computational efficiency and thus scalability than solving the optimization problem.

The second part of this dissertation looks into elements of service provider-user coordination first in the formulation of the centralized resource allocation problem in MIP and then in the formulation of the optimization problem in a decentralized manner for various problem cases. By examining differences between the centralized, optimal solutions and the decentralized solutions for those problem cases, the analysis of how the decentralized service provider-user coordination breaks down the optimal solutions is performed. Based on the analysis, strategies of decentralized service provider-user coordination are developed.

ACKNOWLEDGMENTS

First of all, I am very thankful to God for His grace and blessings. This journey from start to finish has done with His continuous help and love.

I would like to express my sincere thanks to all of my committee members, my family members and friends, and CIDSE staffs at ASU for their big support, help and great encouragement during my Ph.D. study.

First, I would like to gratefully acknowledge my wonderful committee whose support and inspiration enabled me to achieve the goal of completing this research.

I would like to thank my academic advisor, Dr. Nong Ye for being a great mentor with big support and guidance. I have learned not only knowledge but also strong mind and passion, dedication to professional career, research and life. I really appreciate her for giving me a great motivation continuously by her wholehearted support.

I would also like to thank Dr. Teresa Wu, Dr. Rong Pan and Dr. Sik-Sang Yau for having served on my dissertation committee. I greatly appreciate their time, support and valuable insights. I also want to thank Dr. Muhong Zhang for her constructive comments.

Thanks also go to my fellow colleagues for making my time in the Ph.D. program more enjoyable, especially thank to Zhuoyang Zhou who was very kind and supportive.

My special thanks go to all of my family members in God for their love and prayers at Irvine World Vision Church in California, Bupyeong Methodist Church in South Korea and The Lord's Church of Praise in Arizona. I would like to thank Rev. Chan Hong Kim and Young Mi Kim to support me with God's words and prayers in AZ.

Lastly, I would like to express my deep appreciation to my beloved parents, Han Seob Yang and Soon Ok Hwang and my beloved parents-in-law, Eun Pa Hong and Mi

Young Kim for their unconditional love, support and encouragement. Many thanks to my beloved husband, Shinil Hong, who made this long journey much more pleasant with his love and prayers.

TABLE OF CONTENTS

Page

LIST OF TABLES

Table Page

Table Page

Table Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Many large IT service providers and organizations such as Google, IBM, Amazon, Microsoft, Yahoo, and Sun use service clouds for service provision to their users (Zhang & Zhang, 2009). Service clouds contain a large number of computer and network servers that take dynamic service orders from a large number of users. With the dynamic inflow of service orders, IT organizations are also required to keep pace with a high rate of changes with regard to user demands, market and techniques. They face rapidly changing market conditions, new competitive pressures and threats, and new regulatory flats that demand compliance. All of these situations drive the need for the IT infrastructure of an organization to respond quickly in support of new business models and requirements (Kreger, 2001; Papazoglou, Traverso, Dustdar, & Leymann, 2007).

In those environments, cloud computing has gained great attention as the next generation of the computing paradigm from the generations of mainframe, personal computers, client-server computing and web computing (Rajan & Jairath, 2011). A cloud is defined as a virtual pool of resources which are built on distributed infrastructure. Cloud computing is a type of parallel and distributed system consisting of collections of virtual computing, data and software resources to provide on-demand IT services to users in a pay-as-you-go manner and is accessible as a composable service by the network or typically by the Internet (Armbrust et al., 2010; Rajan & Jairath, 2011; Zhang, Zhang, & Cai, 2007).

With cloud computing, it can reduce IT complexity and costs by eliminating ownership costs such as investment, upgrade and maintenance costs and, instead, need to

pay rental usage of IT resources. Moreover, it can provide a better quality of services to wider range of users by having on-demand, highly elastic, portable, agile service delivery from service providers at anytime from anywhere (Endo et al., 2011; Foster, Zhao, Raicu, & Lu, 2008; Lin & Lu, 2011; Yigitbasi, Iosup, Epema, & Ostermann, 2009; Zhang & Zhang, 2009).

A major advantage of cloud computing is high scalability to have a large capacity of virtual resources by pulling together a large pool of physical resources. The Open Cloud Computing Federation involving multiple Cloud Computing Service Providers (CCSPs) provides a uniform resource interface for the rapid growth of cloud users. In addition to scalable services, on-demand usage and a pay-as-you-go business model, other benefits of cloud computing include reliability of services, increased resource utilization rather than having additional and expensive resources just for peak service periods of a limited user population, and so on (Jamkhedkar, Lamb, & Heileman, 2011).

The success of cloud computing mainly depends on the allocation of resource in an efficient and effective way (Shyamala & Rani, 2015)**.** Resource management in cloud computing is challenging as it has to satisfy objectives of all CCSPs and cloud users in an unpredictable environment with fluctuating workload, large shared resources and complex policies to manage them. General policies for consideration in cloud resource management are as follows. Admission control determines whether a cloud user's service request is admitted for service processing in the cloud. Resource distribution provides Virtual Machines (VMs) onto physical machines and assigns resources to service requests. Energy optimization optimizes the use of energy in cloud data centers. Quality

of Service (QoS) guarantees the quality of a cloud user's service request in terms of response time, operational cost and system throughput**.**

Resource allocation system aims to ensure that the requested services' requirements are facilitated by CCSP's infrastructure. A CCSP offers IT resources as a service to the cloud users either through private or public networks. For efficient resource allocation in cloud computing with related policies, there is a need to obtain accurate information about the global state of the system (Marinescu, 2013). Hence, many studies propose to use centralized algorithms from mathematical programming, game modeling to heuristics, which require a central entity to be present for either solving the resource allocation optimization problem directly or coordinating solutions of the resource allocation problem with information of all service requests' requirements and resource status of all service providers. However, the centralized methods easily suffer from the scalability problem in generating the optimal solutions for increasing problem sizes (Selvi, Valliyammai, & Dhatchayani, 2014).

Decentralized algorithms solely rely on interactions among service providers or interactions between service providers and end users to seek the solution of a resource allocation problem, without any central entity to be present. Using such self-management principles with the decentralized manner in many studies does not guarantee generating optimal or near-optimal solutions in resource allocation. More research for obtaining efficient resource allocation solutions in real time is still on-going by examining new algorithms, which can guarantee solution optimality as well as scalability. This dissertation contributes to identify and establish two efficient resource allocation methods

to generate optimal or near-optimal solutions for resource allocation with satisfactory scalability.

Chapter 2 aims at identifying centralized heuristics to generate optimal or near-optimal solutions within a short time period. To achieve this research goal, it starts with a set of representative problem cases to obtain the centralized, optimal solutions. The problem cases involve various types of a service provider's configuration and various problem sizes with different numbers of service providers and service requests. By investigating the optimal solutions for the problem cases, this research analyzes how the centralized, optimal solutions make decisions of which service request is sent to which service provider for the service provision while satisfying the service request's requirement with accurate resource assignment. As a result, important heuristics are suggested for various conditions, which capture the centralized decision making in optimal solutions, in generating optimal or near-optimal resource allocation solutions. Then, the proposed heuristics are tested in another set of representative problem cases with introduction of more problem complexity by increasing the number of service providers and the number of service requests to evaluate the performance quality of the proposed heuristics in larger problem cases. The experimental results demonstrate how competitive the heuristic algorithms are by comparing with the optimal solutions.

Chapter 3 aims at identifying elements of service provider-user coordination that can lead a scalable, distributed algorithm to the optimal or near-optimal solution. To achieve this research goal, it starts with a simple service provider-user coordination protocol in a scalable, distributed algorithm. By examining differences between the centralized, optimal solutions and the decentralized solutions for various problem cases

involving various types of a service provider's configuration and different numbers of service requests, this research analyzes how the decentralized service provider-user coordination breaks down the centralized, optimal solutions and, as a result, suggests key elements of the decentralized service provider-user coordination strategies.

CHAPTER 2

THE ANALYSIS OF CENTRALIZED, OPTIMAL SOLUTIONS TO DEVELOP HEURISTICS

Efficient resource allocation is one of the most important parts in cloud computing, which heavily relies on allocation of computer and network resources of service providers to requested services of users for satisfying the users' service requirements. This study in Chapter 2 first defines the resource allocation problem as an optimization problem in MIP and obtains the optimal solutions for various resource-service problem scenarios. Based on the analysis of the optimal solutions in those problem scenarios, important heuristics are designed for efficient resource allocation. Then, extended experiments are conducted with larger numbers of user requests and service providers for performance evaluation of the resource allocation heuristics. Experimental results of the resource allocation heuristics show the comparable performance of the heuristics to the optimal solutions from solving the centralized optimization problem. Moreover, the resource allocation heuristics demonstrate better computational efficiency and thus scalability than solving the centralized optimization problem.

2.1 Literature Review

Table 1 shows a summary of comparisons between existing work in resource allocation and this study in several aspects. Appendix A shows the description of various heuristics for efficient resource allocation used in the existing studies.

Table 1

Analysis of Literatures in Comparison with This Study

Literatures	SLA (QoS, Resource Requirements, etc.)	Service, System Models	Insufficient resource capacity	Scalability	Solution optimality (compared by optimal solutions)
This study	Yes	Yes	Yes	Yes	Yes
(Messina, Pappalardo, & Santoro, 2012, 2014; Papagianni et al., 2013)	Yes	No	No	Yes	No
(Son, Jung, & Jun, 2013)	No	No	No	Yes	No
(Kadda, Benhammadi, Sebbak, & Mataoui, 2015; Li, Wang, & Liu, 2014; Srinivasa et al., 2014; Wang & Fang, 2014; Zhou, Dutkiewicz, Liu, Fang, & Liu, 2014)	No	No	No	No	No
(Zuo, Zhang, & Tan, 2014)	Yes	No	Yes	Yes	Yes
(Goudarzi & Pedram, 2011b)	Yes	No	No	Yes	No
(Nesmachnow, Iturriaga, & Dorronsoro, 2015)	Yes	No	Yes	Yes	No
(Hsu, Chen, & Park, 2008; Liu, Zhou, Fu, & Liu, 2014; Suresh & Vijayakarthick, 2011; Varalakshmi, Judgi, & Hafsa, 2013; Wu, Deng, Zhang, Zeng, & Zhou, 2013)	Yes	No	No	No	No
(Sharma, Tantawi, Spreitzer, & Steinder, 2010; Wei & Blake, 2013)	Yes	No	No	No	Yes
(Dhingra & Paul, 2014)	No	Yes	Yes	No	No
(Kumar, Feng, Nimmagadda, & Lu, 2011)	Yes	Yes	Yes	Yes	No

A lot of studies have introduced various algorithms for efficient resource allocation to service requests in cloud computing environments as shown in Table 1 and Appendix A. This study considers satisfaction of QoS and resource requirements stated in Service Level Agreements (SLA) when determining allocation of computer and network resources to service requests. Some studies allow SLA violations in generating their solutions. Son et al. (2013) proposed to evaluate appropriateness of each data center by considering geographical locations of cloud users and CCSPs and resource workload through its utility function. However, it resulted in 0.1%~13% of SLA violations and placement failures for its solutions even when resource capacity over all servers was sufficient to serve all service requests. Dhingra and Paul (2014) focused on total profit maximization than SLA satisfaction of service requests such as QoS requirements or resource constraints in generating resource allocation solutions. When a violation occurred in resource constraints or the QoS requirement of a service request was not satisfied, a penalty value was imposed affecting the system profit adversely. Genetic algorithm was proposed for QoS-aware service composition (Canfora, Penta, Esposito, & Villani, 2005). It had a single-objective fitness function where factors were aggregated using a weighted sum, and it allowed violation of constraints with a static penalty value.

To ensure satisfaction of requirements for all service requests, this study introduces the use of service and system models for precise resource allocation similar to our previous work (Ye, Yang, & Aranda, 2013). Only a few studies (Dhingra & Paul, 2014; Kumar et al., 2011) used such service and system models for resource allocation problems. However, the solutions did not get compared with the optimal solutions. In

(Kumar et al., 2011), Earliest Deadline First-greedy algorithm was used to allocate tasks to available VMs through a lookup table based on computing speeds and costs of VMs. In addition, a simple equation was provided to estimate completion time for each task. However, the experimental results were evaluated by comparing with other scheduling solutions. An optimization technique called Bacterial Foraging used in (Dhingra & Paul, 2014) tried to optimize resource allocation and thus, improved energy efficiency of the data center specifically in power consumption. The power consumption model by CPU utilization was defined and used to determine efficient resource allocation solutions. However, it only provided analysis of three different heuristic solutions.

Proposed heuristics in this study can generate solutions even when overall resource capacity is not sufficient for all service requests. Other studies introduce the use of external VMs with different pricing schemes or VM migration instead of dropping the overloaded requests. As an example, Zuo et al. (2014) proposed the use of self-adaptive learning particle swarm optimization to solve the task allocation problem modeled as an integer programming. When resources were not sufficient to meet the demand, outsourcing tasks to external clouds was proposed rather than generating solutions with some tasks dropped under its own resource capacity. Moreover, the performance of our proposed heuristics is compared with the optimal solutions, whereas most literatures show the quality of their proposed solutions compared with other common heuristics. Only a few studies (Sharma et al., 2010; Wei & Blake, 2013; Zuo et al., 2014) analyzed their solutions by comparing with the optimal solutions based on the assumptions to know estimation of resource and service relationships.

Furthermore, the heuristic solutions in this study are generated with great scalability, while some literatures focus on generating good solutions without addressing scalability issue. A new task scheduling approach based on adjusting maximum entropy method in (Li et al., 2014) considered each scheduler to calculate the best task slicing scheme, resulting in minimum task response time. Srinivasa et al. (2014) introduced a game modeling between clients with the utility factor, which considered time and budget constraints. The request with highest expected utility from the entire set of waiting requests was selected for service scheduling. A modified elite chaotic immune clonal selection algorithm by (Zhou et al., 2014) was developed to increase overall efficiency of system with ranking, evaluation and mutation processes. The algorithms used in those studies generated solutions without addressing computational efficiency and were not able to compare their solutions with the optimal solutions.

From the literatures reviewed above, two methods are found in an approach to obtain a resource allocation solution. First, all service requests are sorted by their priority values and each requested service is assigned to a CCSP one at a time. In (Kadda et al., 2015), jobs and tasks in a job were sorted in an ascending order of their Computation Time (CT) and assigned to the selected clusters and servers with minimum CT respectively. Similarly, in (Wu et al., 2013) task priority was computed by its attributes of user privilege, task length or its waiting time in queue, and a sorted task was assigned to a server with minimum CT. A self-adaptive learning particle swarm optimization used in (Zuo et al., 2014) assigned different priorities to all tasks by four velocity updating strategies and allocated tasks to cloud based on the assigned priority. In (Goudarzi & Pedram, 2011b), an initial solution obtained by a greedy algorithm determined the order

of resource assignment processing for clients. Then, a heuristic of force-directed resource assignment was applied to a mixed integer non-linear programming problem by checking partial profit gained from allocating each portion of a client's request to a server. Heuristics used in (Nesmachnow et al., 2015) assigned different priorities to requests with diverse criteria, and reordering local search was applied to improve solution accuracy.

Second, a group of service requests by various criteria are sent to different servers, and each server processes the given requests with a master-slave structure. For linear programming problem formulated in (Shi & Hong, 2010), one master node sent tasks to sub-nodes for executing the received tasks based on two modes of the system, either budget-bound mode or communication-bound mode. An efficient resource allocation strategy was proposed in (Hsu et al., 2008) by combination of one resource broker (as the master node) and a number of heterogeneous clusters (as slave processors) for distributing tasks onto computing nodes with smallest communication ratio. In (Suresh & Vijayakarthick, 2011), the metascheduler scheduled all service requests to maximize resource utilization by parallel job scheduling strategies, and jobs were executed at the cloud cluster by each local scheduler. A MIP problem was formulated in (Papagianni, et al., 2013) in a way that requests were mapped to two phases by solving flow allocation (as a node mapping) and allocating virtual links to substrate (as a link mapping) as the multicommodity flow allocation problem. Urgaonkar, Kozat, Igarashi, and Neely (2010) introduced a joint utility function of an average application throughput and energy costs of a data center. Jobs were routed to join the shortest queue policy with knowledge of

queue backlog information, and with the assigned jobs, the optimal resource allocation at each active server was solved.

Similar to the first method, in this study, all service requests are randomly ordered first and then each requested service is assigned to a CCSP one at a time through the proposed heuristics. The detail of the heuristics are described in Chapter 2.5.1.

2.2 Research Focus

Heuristics have been introduced in many researches to get optimal or near-optimal solutions in resource allocation problem, rather than solving the centralized optimization problem directly for resource allocation decisions due to its computation complexity. In other words, the solutions of the optimization problem cannot be obtained in realistic time except for small problem cases.

Centralized heuristics (i.e. heuristics in a centralized form are applied to obtain a solution with a much smaller solution space than the centralized algorithm) has not been well addressed in existing work for resource allocation in cloud computing environments. A major challenge is to produce the heuristic solutions as good as or close to the optimal solutions that can be obtained by solving the optimization problem with all information available and all decisions made in one place. Another challenge is to produce the heuristic solutions with fast convergence rate regardless of problem sizes. Moreover, there are few studies to work on task assignment with efficient resource allocation simultaneously. To deal with those challenges the current studies have, it is essential to investigate and discover heuristics in producing optimal or near-optimal solutions for resource allocation with great scalability.

This study in Chapter 2 aims at identifying new heuristics to generate optimal or near-optimal resource allocation solutions within a short time period. To achieve this research goal, it first formulates the resource allocation optimization problem and obtains the optimal solutions for various problem cases. Based on the analysis of optimal solutions for the problem cases under different types of service providers' configuration and problem sizes with various numbers of service providers and service requests, this study proposes heuristics for efficient resource allocation in those problem cases. The proposed heuristics are designed especially by capturing the decision making behavior of solving the optimization problem to generate the optimal solutions. Then, the heuristics are tested in extended problem cases with introduction of more complexity by increasing the number of service providers and the number of service requests to evaluate performance quality of the proposed heuristics.

2.3 The Formulation of the Resource Allocation Optimization Problem

Resource allocation is often addressed as an optimization problem consisting of objectives, decision variables, constraints, and algorithms to solve it. There are mainly three types of optimization objectives in resource allocation: 1) resource performance objectives such as resource utilization, load balancing, and energy saving by switching on and off servers depending on their workload and resource status (Berman, 1999; Kuribayashi, 2011; Livny & Raman, 1999; Rezvani, Akbari, & Javadi, 2015; Wuhib, Stadler, & Spreitzer, 2010; Yin, Wang, Meng, & Qiu, 2012), 2) system performance objectives including system throughput measured by the number of jobs executed by the system (Atiewi, Yussof, & Ezanee, 2015; Berman, 1999; Mehdi, Mamat, Ibrahim, & Subramaniam, 2011; Shi & Hong, 2010; Urgaonkar et al., 2010; Yang, Qin, Li, & Yang,

2013), and 3) application performance objectives of response time (e.g., execution time and makespan), QoS (Ardagna, Casolari, & Panicucci, 2011; Berman, 1999; Gong, Ramaswamy, Gu, & Ma, 2009; Laili et al., 2013; Sindhu & Mukherjee, 2013; Wang & Su, 2015).

The optimization problems in resource allocation are subject to various types of constraints. Application requirements of resource and QoS are stated in SLA between cloud users and CCSPs (Endo et al., 2011; Wuhib et al., 2010) including CPU and memory requirements for host machines, bandwidth, delay, and QoS requirements (e.g., execution time) of services and applications (Chen, Farley, & Ye, 2004; Hu, Cao, & Gu, 2008; Lamparter, Ankolekar, & Studer, 2007; Staikopoulos, Cliffe, Popescu, Padget, & Clarke, 2010; Tran, Tsuji, & Masuda, 2009; Wang, Vitvar, Kerrigan, & Toma, 2006; Zheng, Yang, & Zhao, 2010); capacity limits of resources are given to indicate the maximum capacity of each system resource; service and system models are provided to describe how services produce resource workloads and thus change the state of system resources which in turn affect the performance of services (Yau et al., 2009; Ye et al., 2010). The accuracy and quality of predicted behavior based on such models are fundamental to the effectiveness of precise resource allocation and service scheduling (Berman, 1999; Marinescu, 2013).

Service requests by cloud users require some resource amount to run them with satisfying requirements of the users. Here, service parameter values of service requests are assigned to provide the services to users, and the values of service parameters affect quality of the services. In the optimization problem for this study, decision variables are used to assign requested services of cloud users to CCSPs with specific values of service

parameters for the services. In addition, service and system models for each CCSP are included in constraints to predict resource workloads and service performance for precise resource allocation. This optimization problem addresses resource, system and application objectives of resource allocation and is solved for each epoch of dynamic resource allocation. The resource allocation optimization problem is formulated as a MIP. Note that a service provider may have one or more servers and that a service user may generate one or more clients. Each client may request one service. Hence, in the following formulation, the terms of server and client are used. Table 2 and Table 3 indicate variables and indices, and decision variables and given inputs used in the formulation respectively.

Table 2

Variables and Indices for the Resource Allocation Optimization Formulation

k	A given client, $k = 1,\ldots, K$
i	A given server, $i = 1,\ldots, I$
w_i	Resource variable w of server i, $w_i = 1,\ldots, W_i$
s	A service type, $s = 1,\ldots, S$
d_s	Service parameter d of service s, $d_s = 1,\ldots, D_s$
p_s	QoS variable p of service s, $p_s = 1,\ldots, P_s$
R_{kiw_i}	The amount of resource variable w of server i taken by client k's service request as a positive real value
Q_{kp_si}	The value of QoS variable p_s of client k's service request on server i as a positive real value

Table 3

Decision Variables and Given Inputs for the Resource Allocation Optimization Formulation

X_{ki}	Binary decision variables such that $X_{ki} = 1$ if client *k*'s service request is assigned to server *i* $X_{ki} = 0$ if client *k*'s service request is not assigned to server *i*
A_{kd_si}	Positive integer decision variables, Level of service parameter d_s for client *k*'s service request on server *i*
U_{ks}	Given inputs from client *k*, such that $\sum_s U_{ks} = 1$ for a given *k*, $U_{ks} = 1$ if client *k*'s service request uses service *s* $U_{ks} = 0$ if client *k*'s service request does not service *s*
V_{is}	Given inputs from server *i*, $V_{is} = 1$ if service *s* is provided by server *i* $V_{is} = 0$ if service *s* is not provided by server *i*
$A^l_{kd_si}$	Given inputs as a positive integer value from client *k*, Limit (i.e. the maximum level) of service parameter d_s of client *k*'s service request on server *i.*
$R^l_{iw_i}$	Given inputs as a positive real value from server *i* to indicate the resource capacity, Limit of resource variable *w* of server *i*
$Q^l_{kp_s}$	Given inputs as a positive real value from client *k* to specify QoS requirements, Limit of QoS variable p_s of client *k*'s service request

The formulation of the resource allocation optimization problem is as follows.

$$\text{Minimize} \sum_k \sum_{p_s} \frac{\left|\sum_i Q_{kp_si} - Q^l_{kp_s}\right|}{Q^l_{kp_s} * P_s} \tag{1}$$

subject to

$$\sum_i X_{ki} \leq 1 \qquad \forall\, k \tag{2}$$

$$X_{ki} U_{ks} \leq V_{is} \qquad \forall k, i, s \tag{3}$$

$$A_{kd_si} \leq A^l_{kd_si} \quad \forall i, k, d_s \tag{4}$$

$$R_{kiw_i} = X_{ki} F_{iw_i}\left(A_{k1i}, \ldots, A_{kD_si}\right) \quad \forall i, k, w_i \tag{5}$$

$$Q_{kp_si} = X_{ki} G_{ip_s}\left(R_{ki1}, \ldots, R_{kiW_i}\right) \quad \forall i, k, p_s \tag{6}$$

$$\sum_k R_{kiw_i} \leq R^l_{iw_i} \quad \forall i, w_i \tag{7}$$

$$Q_{kp_si} \leq X_{ki} Q^l_{kp_s} \text{ or } Q_{kp_si} \geq X_{ki} Q^l_{kp_s} \quad \forall i, k, p_s \tag{8}$$

The objective function in Equation (1) is to make the levels of the QoS variables closest to the QoS requirements. The difference between the actual QoS level (Q_{kp_si}) and the required QoS level ($Q^l_{kp_s}$) for each QoS variable (p_s) is first normalized by the required QoS level, then summed and normalized over the total number of QoS variables, finally summed over all clients' service requests. More importantly, this objective function ultimately makes as many clients' services to be served as it can since the penalty of not serving a client's service request (i.e. 1) is always bigger than a difference between any QoS level provided and the required QoS level.

As a server-client coordination constraint, Equation (2) guarantees that client *k*'s service request can be assigned to one server *i* at most. Equation (3), as a service constraint, requires if client *k*'s service request is assigned to server *i*, the service type *s* of client *k*'s service request must be provided by server *i* (i.e. if $X_{ki} = 1$ and $U_{ks} = 1$, then $V_{is} = 1$). As another service constraint, Equation (4) enforces that the level of service parameter d_s of client *k*'s service request on server *i* should not exceed the limit (i.e. the maximum level).

As service-resource-QoS relation constraints, Equation (5) gives relations of service parameters with resource usages in function F_{iw_i} of the assigned level of service parameter d_s of client *k*'s service request on the server only if client *k*'s service request is

assigned to the server i. Equation (6) gives relations of resource usages with QoS performance in function G_{ip_s} of the service's resource usages on the server only if client k's service request is assigned to the server i.

As a resource capacity constraint, Equation (7) enforces that the total resource amount on the resource variable w of server i used by all service requests for all clients should not exceed the maximum resource capacity for this resource variable. As a QoS requirement constraint, Equation (8) ensures that the QoS level of p_s for the client k's service request at server i is equal to or less than the maximum QoS requirement or equal to or greater than the minimum QoS requirement, only if client k's service request is assigned to server i. Appendix B shows one example, which gives realization of the resource allocation problem formulation.

The MIP problem is implemented in ILOG OPL Development Studio IDE Version 6.1. ILOG CPLEX 11.2.0 is used as a solver to the MIP problem. A laptop computer used to run the software is a Samsung Q320 with Intel Core 2 Duo T6500 2.1 GHz processor, 4 GB RAM, and Windows 7. The ILOG OPL and CPLEX are integrated into C# code in Microsoft Visual Studio 2010. The C# code first loads all the necessary input files of the problem including given inputs as well as service-resource relation functions and resource-QoS relation functions for each service type. With the loaded input files, the C# code then calls ILOG OPL and CPLEX to run the MIP optimization and solve the problem to generate an optimal solution. The computation time of obtaining the optimal solution is recorded by the C# code. Note that times required for loading input files and generating output files are also included in the computation time of obtaining the optimal solution.

2.4 Design of Problem Cases for Optimal Solutions

In this section, a set of problem cases with different experimental settings are designed to analyze the optimal solutions and to identify important heuristics in order to generate optimal or near-optimal resource allocation solutions. This study uses two types of services: a communication intensive service and a computation intensive service. The communication intensive service has one service parameter and one QoS variable with QoS levels of Low (L), Medium (M) and High (H). Similarly, the computation intensive service has one service parameter and one QoS variable with QoS levels of L, M and H. If a provided QoS level of a service request is equal to or greater than a required QoS level, then the service request is considered as satisfied in both types of services. In this study, three experiments named Step 1, Step 2 and Step 3 are conducted. Table 4 shows three levels of QoS variables and limit (i.e. the maximum level) of service parameters for the communication intensive service and the computation intensive service used in the experiments.

Table 4

QoS Levels and Limit of Service Parameters for Communication Intensive Service and Computation Intensive Service in Step 1, Step 2 and Step 3 Experiments

Experiment	Communication Intensive Service (s = 1)		Computation Intensive Service (s = 2)	
	QoS Levels ($Q_{kp_1}^l$)	Limit of service parameter ($A_{kd_1i}^l$)	QoS Levels ($Q_{kp_2}^l$)	Limit of service parameter ($A_{kd_2i}^l$)
Step 1	L (5) M (15) H (25)	5	L (6) M (17) H (30)	5
Step 2	L (8) M (15) H (21)	5	L (7) M (14) H (25)	5
Step 3	L (4~5) M (15~17) H (21~23)	4	L (3~4) M (12~14) H (18~19)	4

Step 1 experiment lets each client using the communication intensive service set the maximum level of service parameter to 5 and set the QoS requirement to one of three levels: 5 for L, 15 for M and 25 for H. It also lets each client using the computation intensive service set the maximum level of service parameter to 5 and set the QoS requirement to one of three levels: 6 for L, 17 for M and 30 for H. Similarly, Step 2 experiment lets each client using the communication intensive service set the maximum level of service parameter to 5 and set the QoS requirement to one of three levels: 8 for L, 15 for M and 21 for H. It also lets each client using the computation intensive service set the maximum level of service parameter to 5 and set the QoS requirement to one of three levels: 7 for L, 14 for M and 25 for H.

Different from the experiments of Step 1 and Step 2, Step 3 experiment has a range of QoS requirement for each level and thus randomly assigns a specific value of the QoS variable given the QoS requirement level from a client. Hence, Step 3 experiment lets each client using the communication intensive service set the maximum level of the service parameter to 4 and set the QoS requirement to one of three levels of L, M and H, and a specific QoS value is randomly selected from 4 to 5 for L, from 15 to 17 for M and from 21 to 23 for H. It also lets each client using the computation intensive service set the maximum level of the service parameter to 4 and set the QoS requirement to one of three levels of L, M and H, and a specific QoS value is randomly selected from 3 to 4 for L, from 12 to 14 for M and from 18 to 19 for H.

This study uses two types of servers: communication-centered server and computation-centered server. Both types of servers have two resource variables of CPU resource and bandwidth resource. The communication-centered server has more bandwidth resource than CPU resource, and the computation-centered server has more CPU resource than bandwidth resource. Table 5 shows capacity limits of two resource variables: CPU resource and bandwidth resource for the communication-centered server and the computation-centered server, which are used in Equation (7) for the optimization problem. The communication-centered server has resource levels of Small (S), Medium (M) and Large (L), and the computation-centered server also has resource levels of S, M and L.

Table 5

Capacity Limits of Two Resource Variables ($\boldsymbol{R_{i1}^{l}}$, $\boldsymbol{R_{i2}^{l}}$) for Communication-Centered Server and Computation-Centered Server and Number of Servers used in Step 1, Step 2 and Step 3 Experiments

Experiment	Communication-Centered	Computation-Centered	Number of Servers
Step 1	S (30,100) M (36,200) L (41,350)	S (116,27) M (232,32) L (406,35)	2
Step 2	S (45,100) M (65,135) L (90,245)	S (105,50) M (155,70) L (205,80)	2
Step 3	S (24~26, 81~84) M (30~33, 93~95) L (38~40, 104~107)	S (78~79, 25~26) M (87~89, 29~30) L (95~97, 32~33)	3

Step 1 experiment lets each communication-centered server set its resource capacity limits to one of three levels: 30 as the capacity limit of CPU resource and 100 as the capacity limit of bandwidth resource for S, 36 of CPU resource and 200 of bandwidth resource for M and 41 of CPU resource and 350 of bandwidth resource for L. It also lets each computation-centered server set its resource capacity limits to one of three levels: 116 of CPU resource and 27 of bandwidth resource for S, 232 of CPU resource and 32 of bandwidth resource for M and 406 of CPU resource and 35 of bandwidth resource for L.

Similarly, Step 2 experiment lets each communication-centered server set its resource capacity limits to one of three levels: 45 as the capacity limit of CPU resource and 100 as the capacity limit of bandwidth resource for S, 65 of CPU resource and 135 of bandwidth resource for M and 90 of CPU resource and 245 of bandwidth resource for L. It also lets each computation-centered server set its resource capacity limits to one of

three levels: 105 of CPU resource and 50 of bandwidth resource for S, 155 of CPU resource and 70 of bandwidth resource for M and 205 of CPU resource and 80 of bandwidth resource for L.

Different from the experiments of Step 1 and Step 2, Step 3 experiment has a range of capacity limits for two resource variables: CPU resource and bandwidth resource with resource levels of S, M and L. Given the resource level from a server, it randomly assigns each specific value for CPU resource and bandwidth resource as the capacity limits. Hence, Step 3 experiment lets each communication-centered server set its resource capacity limits to one of three levels of S, M and L, and a specific resource amount is randomly selected from 24 to 26 as the capacity limit of CPU resource and from 81 to 84 as the capacity limit of bandwidth resource for S, from 30 to 33 of CPU resource and from 93 to 95 of bandwidth resource for M and from 38 to 40 of CPU resource and from 104 to 107 of bandwidth resource for L. It also lets each computation-centered server set its resource capacity limits to one of three levels of S, M and L, and a specific resource amount is randomly selected from 78 to 79 as the capacity limit of CPU resource and from 25 to 26 as the capacity limit of bandwidth resource for S, from 87 to 89 of CPU resource and from 29 to 30 of bandwidth resource for M and from 95 to 97 of CPU resource and from 32 to 33 of bandwidth resource for L. Note that Step 1 and Step 2 experiments have two servers in total, and Step 3 experiment has three servers in total.

For the two services used in the experiments, two variables of resource usages are defined as a key role in determining QoS performance of the services: CPU resource (R_{ki1}) and bandwidth resource (R_{ki2}). The communication intensive service requires more bandwidth resource than CPU resource, while the computation intensive service

requires more CPU resource than bandwidth resource. Such resource and QoS impact models of services are needed for Equations (5) and (6) in the optimization formulation, and thus simple but general forms of resource and QoS impact models for the communication intensive service and the computation intensive service are introduced. Table 6 shows resource and QoS impact models of the two services used in the experiments of Step 1, Step 2 and Step 3. Tables 7, 8 and 9 give the experimental setups of Step 1, Step 2 and Step 3 experiments with different numbers of service requests respectively.

Table 6

Resource and QoS Impact Models of Communication Intensive Service and Computation Intensive Service in Step 1, Step 2 and Step 3 Experiments

Experiment	Communication Intensive Service ($s = 1$)	Computation Intensive Service ($s = 2$)
Step 1	$R_{ki1} = 0.1 * A_{kd_1 i}$ $R_{ki2} = 5.0 * A_{kd_1 i}$ $Q_{kp_1 i} = 2R_{ki1} + R_{ki2}$	$R_{ki1} = 5.8 * A_{kd_2 i}$ $R_{ki2} = 0.3 * A_{kd_2 i}$ $Q_{kp_2 i} = R_{ki1} + R_{ki2}$
Step 2	$R_{ki1} = 0.3 * A_{kd_1 i}$ $R_{ki2} = 4.0 * A_{kd_1 i}$ $Q_{kp_1 i} = 2R_{ki1} + R_{ki2}$	$R_{ki1} = 3.8 * A_{kd_2 i}$ $R_{ki2} = 0.5 * A_{kd_2 i}$ $Q_{kp_2 i} = R_{ki1} + 3R_{ki2}$
Step 3	$R_{ki1} = 0.3 * A_{kd_1 i}$ $R_{ki2} = 5.0 * A_{kd_1 i}$ $Q_{kp_1 i} = 3R_{ki1} + R_{ki2}$	$R_{ki1} = 4.7 * A_{kd_2 i}$ $R_{ki2} = 0.1 * A_{kd_2 i}$ $Q_{kp_2 i} = R_{ki1} + 2R_{ki2}$

Table 7

Design of Experimental Setup with Different Numbers of Service Requests in Step 1 Experiment

Service Requests / Server Configuration	All communication intensive				All computation intensive				Both service types			
	All-L (1-1)	All-M (1-2)	All-H (1-3)	Mixed (1-4)	All-L (2-1)	All-M (2-2)	All-H (2-3)	Mixed (2-4)	All-L (3-1)	All-M (3-2)	All-H (3-3)	Mixed (3-4)
1. All communication centered at S-S (1-1)	A.15 B.35 C.50	A.6 B.10 C.15	A.4 B.7 C.10	A.10 B.17 C.25	A.4 B.8 C.13	A.1 B.2 C.3	A.1 B.2 C.3	A.1 B.3 C.5	A.15 B.27 C.38	A.7 B.12 C.18	A.3 B.5 C.8	A.5 B.8 C.13
2. All communication centered at M-M (1-2)	A.30 B. 75 C.100	A.12 B.24 C.33	A.8 B.14 C.20	A.20 B.36 C.50	A.5 B.11 C.15	A.2 B.4 C.5	A.1 B.2 C.3	A.2 B.3 C.5	A.18 B.32 C.45	A.6 B.10 C.15	A.8 B.14 C.20	A.8 B.14 C.20
3. All communication centered at L-L (1-3)	A.60 B.125 C.175	A.20 B.42 C.58	A.10 B.25 C.35	A.23 B.58 C.80	A.5 B.12 C.18	A.2 B.4 C.5	A.1 B.2 C.3	A.3 B.5 C.8	A.11 B.20 C.28	A.19 B.39 C.55	A.11 B.25 C.35	A.17 B.41 C.55
4. All communication centered at S-M (1-4)	A.15 B.52 C.75	A.5 B.17 C.24	A.3 B.11 C.15	A.8 B.27 C.38	A.4 B.10 C.14	A.1 B.3 C.4	A.1 B.2 C.3	A.1 B.3 C.4	A.13 B.31 C.42	A.4 B.11 C.17	A.2 B.10 C.14	A.4 B.11 C.17
5. All communication centered at S-L (1-5)	A.15 B.84 C.113	A.5 B.28 C.37	A.3 B.17 C.23	A.8 B.38 C.53	A.4 B.11 C.15	A.1 B.3 C.4	A.1 B.2 C.3	A.1 B.5 C.5	A.8 B.24 C.33	A.6 B.27 C.37	A.2 B.16 C.22	A.5 B.25 C.34
6. All communication centered at M-L (1-6)	A.30 B.100 C.138	A.12 B.32 C.45	A.8 B.20 C.28	A.16 B.47 C.65	A.5 B.12 C.17	A.2 B.4 C.5	A.1 B.2 C.3	A.2 B.4 C.7	A.8 B.27 C.37	A.5 B.26 C.35	A.7 B.20 C.28	A.7 B.28 C.46
7. All computation centered at S-S (2-1)	A.4 B.9 C.13	A.1 B.2 C.3	A.1 B.2 C.3	A.1 B.3 C.5	A.15 B.35 C.50	A.6 B.11 C.15	A.4 B.7 C.10	A.9 B.18 C.25	A.8 B.20 C.28	A.6 B.12 C.18	A.2 B.4 C.5	A.5 B.7 C.13
8. All computation centered at M-M (2-2)	A.5 B.11 C.15	A.2 B.4 C.5	A.1 B.2 C.3	A.2 B.5 C.8	A.30 B.70 C.100	A.12 B.23 C.33	A.7 B.14 C.20	A.20 B.35 C.75	A.10 B.22 C.36	A.3 B.7 C.10	A.4 B.9 C.13	A.4 B.7 C.14
9. All computation centered at L-L (2-3)	A.5 B.13 C.18	A.2 B.4 C.5	A.1 B.2 C.3	A.3 B.7 C.11	A.60 B.120 C.175	A.20 B.40 C.58	A.12 B.25 C.35	A.25 B.58 C.85	A.15 B.39 C.74	A.6 B.12 C.23	A.5 B.12 C.24	A.8 B.18 C.36
10. All computation centered at S-M (2-4)	A.4 B.10 C.14	A.1 B.3 C.4	A.1 B.2 C.3	A.1 B.4 C.6	A.15 B.55 C.75	A.6 B.18 C.24	A.4 B.11 C.15	A.8 B.27 C.63	A.9 B.22 C.35	A.3 B.10 C.20	A.2 B.7 C.9	A.3 B.7 C.9
11. All computation	A.4	A.1	A.1	A.1	A.15	A.5	A.3	A.8	A.10	A.6	A.2	A.3

centered at S-L (2-5)	B.11 C.15	B.3 C.4	B.2 C.3	B.6 C.8	B.84 C.113	B.28 C.37	B.17 C.23	B.39 C.94	B.30 C.48	B.13 C.24	B.9 C.12	B.12 C.17
12. All computation centered at M-L (2-6)	A.5 B.13 C.17	A.2 B.4 C.5	A.1 B.2 C.3	A.2 B.6 C.9	A.30 B.100 C.138	A.12 B.33 C.45	A.7 B.20 C.28	A.16 B.47 C.115	A.10 B.32 C.49	A.3 B.10 C.14	A.4 B.10 C.19	A.4 B.13 C.21
13. Both server types (communication, computation) at S-S (3-1)	A.4 B.24 C.32	A.1 B.7 C.9	A.1 B.5 C.7	A.4 B.10 C.15	A.4 B.24 C.32	A.1 B.7 C.9	A.1 B.5 C.7	A.1 B.11 C.26	A.8 B.36 C.54	A.2 B.13 C.18	A.2 B.7 C.10	A.2 B.18 C.16
14. Both server types (communication, computation) at M-M (3-2)	A.5 B.45 C.58	A.2 B.15 C.19	A.1 B.9 C.12	A.2 B.22 C.37	A.5 B.45 C.58	A.2 B.15 C.19	A.1 B.9 C.12	A.2 B.22 C.48	A.10 B.27 C.109	A.4 B.24 C.35	A.2 B.14 C.20	A.4 B.37 C.44
15. Both server types (communication, computation) at L-L (3-3)	A.5 B.76 C.97	A.2 B.25 C.32	A.1 B.15 C.19	A.3 B.35 C.80	A.5 B.76 C.97	A.2 B.25 C.32	A.1 B.15 C.19	A.2 B.37 C.54	A.12 B.130 C.175	A.4 B.42 C.59	A.2 B.26 C.35	A.2 B.63 C.73
16. Both server types (communication, computation) at S-M (3-4)	A.5 B.25 C.33	A.2 B.8 C.10	A.1 B.5 C.7	A.2 B.12 C.28	A.4 B.44 C.57	A.1 B.14 C.18	A.1 B.9 C.12	A.2 B.21 C.27	A.11 B.56 C.75	A.3 B.18 C.24	A.2 B.11 C.15	A.3 B.28 C.38
17. Both server types (communication, computation) at S-L (3-5)	A.5 B.26 C.34	A.2 B.8 C.10	A.1 B.5 C.7	A.2 B.13 C.28	A.4 B.74 C.94	A.1 B.24 C.30	A.1 B.15 C.19	A.1 B.35 C.79	A.10 B.84 C.113	A.3 B.27 C.37	A.2 B.16 C.23	A.3 B.40 C.55
18. Both server types (communication, computation) at M-L (3-6)	A.5 B.46 C.59	A.2 B.15 C.19	A.1 B.9 C.12	A.2 B.23 C.49	A.5 B.75 C.95	A.2 B.25 C.32	A.1 B.15 C.19	A.2 B.36 C.80	A.12 B.103 C.138	A.4 B.34 C.45	A.2 B.20 C.28	A.3 B.50 C.68
19. Both server types (communication, computation) at M-S (3-7)	A.4 B.44 C.57	A.1 B.14 C.18	A.1 B.9 C.12	A.4 B.21 C.46	A.5 B.25 C.33	A.2 B.8 C.10	A.1 B.5 C.7	A.2 B.12 C.28	A.10 B.55 C.75	A.3 B.18 C.24	A.2 B.11 C.15	A.3 B.27 C.40
20. Both server types (communication, computation) at L-S (3-8)	A.4 B.74 C.94	A.1 B.24 C.30	A.1 B.15 C.19	A.1 B.33 C.79	A.5 B.26 C.34	A.2 B.8 C.10	A.1 B.5 C.7	A.2 B.13 C.28	A.10 B.84 C.113	A.3 B.26 C.37	A.2 B.17 C.23	A.5 B.38 C.55
21. Both server types (communication, computation) at L-M (3-9)	A.5 B.75 C.95	A.2 B.25 C.32	A.1 B.15 C.19	A.2 B.34 C.80	A.5 B.46 C.59	A.2 B.15 C.19	A.1 B.9 C.12	A.2 B.23 C.49	A.10 B.104 C.138	A.4 B.33 C.45	A.2 B.20 C.28	A.3 B.48 C.65

Table 8

Design of Experimental Setup with Different Numbers of Service Requests in Step 2 Experiment

Service Requests / Server Configuration	All communication intensive				All computation intensive				Both service types			
	All-L (1-1)	All-M (1-2)	All-H (1-3)	Mixed (1-4)	All-L (2-1)	All-M (2-2)	All-H (2-3)	Mixed (2-4)	All-L (3-1)	All-M (3-2)	All-H (3-3)	Mixed (3-4)
1. All communication centered at S-S (1-1)	A.8 B.22 C.30	A.5 B.11 C.15	A.3 B.9 C.13	A.7 B.15 C.23	A.4 B.10 C.13	A.2 B.6 C.8	A.2 B.4 C.5	A.3 B.6 C.8	A.14 B.30 C.40	A.7 B.16 C.20	A.5 B.12 C.15	A.8 B.20 C.28
2. All communication centered at M-M (1-2)	A.15 B.29 C.40	A.7 B.15 C.20	A.4 B.12 C.15	A.9 B.19 C.28	A.6 B.15 C.20	A.4 B.10 C.13	A.2 B.6 C.8	A.5 B.11 C.15	A.20 B.41 C.58	A.9 B.23 C.28	A.6 B.16 C.20	A.11 B.27 C.35
3. All communication centered at L-L (1-3)	A.20 B.53 C.75	A.12 B.27 C.38	A.10 B.22 C.30	A.16 B.38 C.55	A.8 B.21 C.28	A.6 B.14 C.18	A.3 B.8 C.10	A.8 B.16 C.23	A.18 B.39 C.53	A.12 B.27 C.38	A.11 B.19 C.33	A.14 B.31 C.48
4. All communication centered at S-M (1-4)	A.10 B.25 C.35	A.5 B.13 C.18	A.4 B.10 C.14	A.6 B.17 C.25	A.4 B.13 C.17	A.3 B.8 C.10	A.2 B.5 C.7	A.3 B.9 C.12	A.11 B.29 C.39	A.7 B.17 C.24	A.5 B.10 C.18	A.9 B.25 C.33
5. All communication centered at S-L (1-5)	A.10 B.38 C.53	A.5 B.19 C.27	A.4 B.16 C.22	A.7 B.25 C.37	A.3 B.16 C.20	A.3 B.10 C.13	A.2 B.6 C.8	A.3 B.11 C.14	A.13 B.30 C.47	A.7 B.20 C.29	A.5 B.18 C.24	A.9 B.28 C.33
6. All communication centered at M-L (1-6)	A.15 B.42 C.58	A.7 B.21 C.29	A.5 B.17 C.23	A.9 B.28 C.40	A.6 B.18 C.24	A.4 B.12 C.15	A.2 B.7 C.9	A.5 B.13 C.18	A.13 B.34 C.45	A.10 B.25 C.33	A.7 B.18 C.27	A.10 B.28 C.37
7. All computation centered at S-S (2-1)	A.5 B.11 C.15	A.2 B.6 C.8	A.2 B.4 C.5	A.3 B.7 C.8	A.10 B.24 C.33	A.8 B.16 C.23	A.3 B.10 C.13	A.8 B.18 C.25	A.12 B.28 C.38	A.3 B.18 C.10	A.5 B.11 C.15	A.6 B.20 C.18
8. All computation centered at M-M (2-2)	A.6 B.15 C.20	A.3 B.8 C.10	A.2 B.6 C.8	A.4 B.9 C.13	A.18 B.35 C.50	A.10 B.24 C.33	A.6 B.14 C.20	A.12 B.26 C.38	A.12 B.24 C.35	A.7 B.13 C.20	A.5 B.14 C.18	A.10 B.21 C.30
9. All computation centered at L-L (2-3)	A.8 B.18 C.25	A.4 B.9 C.13	A.3 B.7 C.10	A.5 B.12 C.15	A.20 B.47 C.65	A.15 B.31 C.43	A.8 B.19 C.25	A.15 B.34 C.48	A.14 B.31 C.43	A.11 B.24 C.35	A.7 B.20 C.28	A.12 B.22 C.35
10. All computation centered at S-M (2-4)	A.5 B.13 C.18	A.3 B.7 C.9	A.2 B.5 C.7	A.3 B.8 C.10	A.10 B.30 C.42	A.8 B.20 C.28	A.4 B.12 C.17	A.7 B.22 C.30	A.6 B.20 C.28	A.3 B.12 C.15	A.5 B.13 C.17	A.6 B.17 C.24
11. All computation	A.5	A.3	A.2	A.3	A.10	A.8	A.4	A.7	A.6	A.3	A.5	A.7

centered at S-L (2-5)	B.14 C.20	B.7 C.10	B.6 C.8	B.10 C.12	B.35 C.49	B.24 C.33	B.14 C.19	B.27 C.35	B.21 C.32	B.16 C.23	B.10 C.22	B.22 C.27
12. All computation centered at M-L (2-6)	A.7 B.16 C.23	A.3 B.8 C.12	A.3 B.7 C.9	A.4 B.10 C.14	A.18 B.41 C.58	A.10 B.28 C.38	A.7 B.17 C.23	A.12 B.30 C.43	A.12 B.30 C.39	A.6 B.19 C.28	A.4 B.17 C.23	A.10 B.26 C.34
13. Both server types (communication, computation) at S-S (3-1)	A.5 B.16 C.23	A.3 B.8 C.12	A.2 B.7 C.9	A.3 B.9 C.12	A.4 B.17 C.23	A.3 B.12 C.15	A.2 B.7 C.9	A.3 B.13 C.17	A.6 B.30 C.42	A.3 B.18 C.24	A.4 B.12 C.15	A.6 B.20 C.29
14. Both server types (communication, computation) at M-M (3-2)	A.7 B.22 C.30	A.3 B.11 C.15	A.2 B.9 C.12	A.4 B.15 C.18	A.6 B.25 C.35	A.4 B.17 C.23	A.2 B.10 C.14	A.5 B.18 C.27	A.10 B.44 C.60	A.6 B.26 C.35	A.4 B.18 C.23	A.4 B.30 C.38
15. Both server types (communication, computation) at L-L (3-3)	A.8 B.36 C.50	A.4 B.18 C.25	A.3 B.15 C.20	A.6 B.18 C.33	A.10 B.34 C.47	A.5 B.23 C.30	A.3 B.14 C.18	A.6 B.25 C.34	A.10 B.63 C.84	A.6 B.37 C.48	A.4 B.24 C.33	A.7 B.44 C.60
16. Both server types (communication, computation) at S-M (3-4)	A.6 B.19 C.25	A.3 B.9 C.13	A.3 B.8 C.10	A.4 B.14 C.20	A.4 B.23 C.32	A.3 B.16 C.20	A.2 B.10 C.13	A.3 B.17 C.23	A.9 B.38 C.52	A.5 B.22 C.30	A.4 B.15 C.20	A.6 B.26 C.35
17. Both server types (communication, computation) at S-L (3-5)	A.8 B.20 C.28	A.4 B.10 C.14	A.3 B.8 C.12	A.5 B.14 C.19	A.4 B.30 C.39	A.3 B.20 C.25	A.2 B.12 C.15	A.3 B.20 C.28	A.10 B.44 C.60	A.5 B.26 C.35	A.4 B.17 C.23	A.7 B.30 C.42
18. Both server types (communication, computation) at M-L (3-6)	A.8 B.23 C.33	A.4 B.12 C.17	A.3 B.10 C.13	A.5 B.16 C.23	A.6 B.31 C.43	A.4 B.21 C.28	A.2 B.13 C.17	A.6 B.22 C.32	A.13 B.49 C.69	A.7 B.30 C.40	A.4 B.20 C.25	A.5 B.35 C.44
19. Both server types (communication, computation) at M-S (3-7)	A.5 B.20 C.28	A.3 B.10 C.14	A.2 B.8 C.10	A.3 B.11 C.16	A.7 B.20 C.27	A.4 B.13 C.18	A.2 B.8 C.10	A.5 B.14 C.20	A.10 B.36 C.50	A.5 B.21 C.29	A.3 B.14 C.18	A.4 B.26 C.32
20. Both server types (communication, computation) at L-S (3-8)	A.5 B.33 C.45	A.2 B.17 C.23	A.2 B.14 C.18	A.3 B.23 C.29	A.10 B.22 C.30	A.6 B.15 C.20	A.3 B.9 C.12	A.6 B.17 C.23	A.8 B.50 C.65	A.4 B.28 C.37	A.3 B.20 C.25	A.6 B.35 C.48
21. Both server types (communication, computation) at L-M (3-9)	A.7 B.35 C.48	A.3 B.18 C.24	A.2 B.14 C.19	A.4 B.18 C.29	A.10 B.28 C.39	A.6 B.19 C.25	A.3 B.12 C.15	A.6 B.21 C.28	A.9 B.57 C.75	A.4 B.33 C.43	A.3 B.23 C.30	A.6 B.40 C.54

Table 9

Design of Experimental Setup with Different Numbers of Service Requests in Step 3 Experiment

Service Requests / Server Configuration	All communication intensive				All computation intensive				Both service types			
	All-L (1-1)	All-M (1-2)	All-H (1-3)	Mixed (1-4)	All-L (2-1)	All-M (2-2)	All-H (2-3)	Mixed (2-4)	All-L (3-1)	All-M (3-2)	All-H (3-3)	Mixed (3-4)
1. All communication centered at S-S-S (1-1)	A.13 B.45 C.60	A.4 B.15 C.19	A.2 B.11 C.15	A.4 B.22 C.23	A.4 B.14 C.19	A.1 B.3 C.4	A.1 B.3 C.4	A.2 B.7 C.8	A.16 B.55 C.75	A.5 B.18 C.23	A.4 B.14 C.19	A.5 B.25 C.27
2. All communication centered at M-M-M (1-2)	A.13 B.50 C.68	A.5 B.17 C.23	A.3 B.12 C.15	A.4 B.21 C.27	A.6 B.17 C.23	A.2 B.6 C.8	A.1 B.3 C.4	A.2 B.10 C.12	A.20 B.63 C.87	A.5 B.19 C.27	A.5 B.15 C.19	A.7 B.30 C.34
3. All communication centered at L-L-L (1-3)	A.18 B.56 C.75	A.6 B.18 C.23	A.5 B.14 C.19	A.6 B.25 C.27	A.6 B.21 C.30	A.2 B.6 C.8	A.2 B.5 C.8	A.3 B.12 C.12	A.23 B.73 C.98	A.7 B.24 C.30	A.6 B.17 C.23	A.9 B.33 C.42
4. All communication centered at S-M-M (1-4)	A.51 B.47 C.65	A.3 B.16 C.22	A.3 B.12 C.15	A.4 B.22 C.25	A.5 B.16 C.22	A.1 B.5 C.7	A.1 B.3 C.4	A.2 B.9 C.10	A.18 B.61 C.83	A.5 B.19 C.25	A.4 B.15 C.19	A.6 B.27 C.32
5. All communication centered at S-L-L (1-5)	A.14 B.51 C.70	A.5 B.17 C.22	A.4 B.13 C.18	A.5 B.21 C.25	A.4 B.19 C.27	A.1 B.5 C.7	A.1 B.5 C.7	A.2 B.9 C.10	A.19 B.66 C.90	A.6 B.22 C.28	A.4 B.16 C.22	A.5 B.29 C.37
6. All communication centered at M-L-L (1-6)	A.17 B.52 C.73	A.6 B.18 C.23	A.4 B.13 C.18	A.5 B.23 C.27	A.5 B.20 C.28	A.2 B.6 C.8	A.1 B.5 C.7	A.2 B.9 C.12	A.21 B.69 C.94	A.7 B.22 C.29	A.5 B.16 C.22	A.8 B.32 C.39
7. All computation centered at S-S-S (2-1)	A.4 B.13 C.19	A.1 B.3 C.4	A.1 B.3 C.4	A.2 B.6 C.8	A.10 B.45 C.60	A.4 B.15 C.19	A.3 B.11 C.15	A.4 B.20 C.23	A.16 B.57 C.75	A.5 B.17 C.23	A.3 B.14 C.19	A.6 B.24 C.30
8. All computation centered at M-M-M (2-2)	A.5 B.15 C.19	A.1 B.3 C.5	A.1 B.3 C.4	A.2 B.8 C.8	A.15 B.49 C.68	A.5 B.17 C.23	A.4 B.12 C.15	A.5 B.24 C.23	A.19 B.62 C.87	A.6 B.19 C.27	A.4 B.15 C.19	A.8 B.29 C.38
9. All computation centered at L-L-L (2-3)	A.5 B.17 C.23	A.2 B.6 C.8	A.1 B.3 C.4	A.3 B.10 C.12	A.18 B.54 C.75	A.5 B.18 C.23	A.4 B.14 C.19	A.6 B.25 C.27	A.21 B.70 C.94	A.6 B.24 C.30	A.4 B.15 C.19	A.8 B.34 C.38
10. All computation centered at S-M-M (2-4)	A.4 B.15 C.19	A.1 B.3 C.5	A.1 B.3 C.4	A.2 B.8 C.8	A.15 B.48 C.65	A.4 B.16 C.22	A.3 B.12 C.15	A.5 B.24 C.23	A.19 B.62 C.83	A.5 B.19 C.25	A.4 B.15 C.19	A.7 B.31 C.35
11. All computation centered at S-L-L (2-5)	A.4 B.16 C.22	A.1 B.5 C.7	A.1 B.3 C.4	A.2 B.9 C.10	A.15 B.51 C.70	A.4 B.17 C.22	A.4 B.13 C.18	A.6 B.22 C.25	A.17 B.66 C.88	A.6 B.22 C.28	A.5 B.15 C.19	A.7 B.33 C.35
12. All computation centered at M-L-L (2-6)	A.4 B.17 C.22	A.1 B.5 C.7	A.1 B.3 C.4	A.2 B.10 C.10	A.17 B.52 C.73	A.5 B.18 C.23	A.3 B.13 C.18	A.4 B.24 C.25	A.20 B.68 C.92	A.7 B.22 C.29	A.4 B.15 C.19	A.6 B.33 C.38

13. Both server types (communication, computation, computation) at S-S-S (3-1)	A.4 B.24 C.33	A.1 B.7 C.9	A.1 B.6 C.8	A.2 B.12 C.13	A.3 B.35 C.47	A.1 B.11 C.14	A.1 B.9 C.12	A.2 B.16 C.18	A.6 B.57 C.75	A.2 B.18 C.23	A.2 B.14 C.19	A.4 B.28 C.29
14. Both server types (communication, computation, computation) at M-M-M (3-2)	A.5 B.27 C.35	A.1 B.8 C.11	A.1 B.6 C.8	A.2 B.15 C.15	A.5 B.39 C.53	A.2 B.13 C.18	A.1 B.9 C.12	A.2 B.20 C.19	A.8 B.64 C.87	A.2 B.21 C.27	A.2 B.15 C.19	A.4 B.31 C.53
15. Both server types (communication, computation, computation) at L-L-L (3-3)	A.5 B.30 C.40	A.2 B.10 C.13	A.1 B.7 C.9	A.3 B.16 C.17	A.7 B.45 C.60	A.2 B.14 C.18	A.2 B.11 C.15	A.3 B.20 C.22	A.10 B.71 C.95	A.3 B.24 C.30	A.2 B.16 C.20	A.6 B.33 C.40
16. Both server types (communication, computation, computation) at S-M-M (3-4)	A.4 B.24 C.33	A.1 B.7 C.9	A.1 B.6 C.8	A.2 B.13 C.13	A.4 B.38 C.52	A.1 B.13 C.17	A.1 B.9 C.12	A.2 B.16 C.18	A.8 B.60 C.83	A.2 B.19 C.25	A.2 B.15 C.19	A.4 B.28 C.36
17. Both server types (communication, computation, computation) at S-L-L (3-5)	A.5 B.26 C.35	A.2 B.9 C.12	A.1 B.6 C.8	A.3 B.14 C.16	A.4 B.40 C.57	A.1 B.13 C.17	A.1 B.10 C.14	A.2 B.18 C.20	A.9 B.65 C.88	A.3 B.22 C.28	A.2 B.15 C.19	A.4 B.32 C.34
18. Both server types (communication, computation, computation) at M-L-L (3-6)	A.5 B.28 C.38	A.2 B.10 C.13	A.1 B.6 C.8	A.2 B.12 C.17	A.5 B.42 C.58	A.2 B.14 C.18	A.1 B.11 C.14	A.3 B.19 C.22	A.10 B.69 C.92	A.2 B.22 C.29	A.2 B.15 C.19	A.5 B.31 C.37
19. Both server types (communication, computation, computation) at M-S-S (3-7)	A.4 B.26 C.35	A.1 B.8 C.10	A.1 B.6 C.8	A.2 B.12 C.14	A.5 B.36 C.48	A.2 B.12 C.15	A.1 B.9 C.12	A.3 B.15 C.19	A.8 B.59 C.79	A.2 B.19 C.24	A.2 B.15 C.19	A.4 B.28 C.32
20. Both server types (communication, computation, computation) at L-S-S (3-8)	A.4 B.28 C.38	A.1 B.8 C.10	A.1 B.7 C.9	A.2 B.12 C.14	A.7 B.37 C.50	A.2 B.12 C.15	A.2 B.9 C.13	A.3 B.18 C.19	A.8 B.62 C.83	A.3 B.20 C.25	A.2 B.16 C.20	A.5 B.29 C.34
21. Both server types (communication, computation, computation) at L-M-M (3-9)	A.5 B.29 C.38	A.1 B.8 C.12	A.1 B.7 C.9	A.2 B.15 C.14	A.7 B.42 C.55	A.2 B.14 C.18	A.2 B.10 C.13	A.3 B.18 C.19	A.10 B.68 C.90	A.3 B.22 C.28	A.2 B.16 C.20	A.4 B.30 C.39

As shown in Table 7, this study introduces twenty one server configurations which are determined by two factors: server type and resource level of a server. With two server types of the communication-centered server and the computation-centered server, there are three combinations of server types for servers designed:

- All servers are communication-centered servers (Server configurations 1~6 in Table 7).
- All servers are computation-centered servers (Server configurations 7~12 in Table 7).
- Some servers are communication-centered servers, and the others are computation-centered servers (Server configurations 13~21 in Table 7).

Since there are three resource levels (S, M and L) available to indicate a server's resource capacity, six combinations of resource levels for servers are constructed when all servers are either communication-centered servers or computation-centered servers:

- All servers of same server type have one resource level of S (Server configurations 1 and 7 in Table 7).
- All servers of same server type have one resource level of M (Server configurations 2 and 8 in Table 7).
- All servers of same server type have one resource level of L (Server configurations 3 and 9 in Table 7).
- Servers of same server type have mixed resource levels of S and M (Server configurations 4 and 10 in Table 7).

- Servers of same server type have mixed resource levels of S and L (Server configurations 5 and 11 in Table 7).
- Servers of same server type have mixed resource levels of M and L (Server configurations 6 and 12 in Table 7).

With servers of both resource types, there are nine combinations of resource levels for servers:

- Servers of both server types have one resource level of S (Server configuration 13 in Table 7).
- Servers of both server types have one resource level of M (Server configuration 14 in Table 7).
- Servers of both server types have one resource level of L (Server configuration 15 in Table 7).
- Some servers are communication-centered servers with the resource level of S, and the others are computation-centered servers with the resource level of M (Server configuration 16 in Table 7).
- Some servers are communication-centered servers with the resource level of S, and the others are computation-centered servers with the resource level of L (Server configuration 17 in Table 7).
- Some servers are communication-centered servers with the resource level of M, and the others are computation-centered servers with the resource level of L (Server configuration 18 in Table 7).

- Some servers are communication-centered servers with the resource level of M, and the others are computation-centered servers with the resource level of S (Server configuration 19 in Table 7).
- Some servers are communication-centered servers with the resource level of L, and the others are computation-centered servers with the resource level of S (Server configuration 20 in Table 7).
- Some servers are communication-centered servers with the resource level of L, and the others are computation-centered servers with the resource level of M (Server configuration 21 in Table 7).

This study also introduces twelve service request combinations which are determined by two factors: service type and QoS level of a service request. With two service types of the communication intensive service and the computation intensive service, three combinations of service types for service requests are designed:

- All service requests are communication intensive services (First column in Table 7).
- All service requests are computation intensive services (Second column in Table 7).
- Some service requests are communication intensive services, and the others are computation intensive services (Third column in Table 7).

Since there are three QoS levels (L, M and H) available to present a service request's QoS requirement, four combinations of QoS levels for service requests are constructed:

- All service requests have one QoS level of L (All-L in Table 7).

- All service requests have one QoS level of M (All-M in Table 7).
- All service requests have one QoS level of H (All-H in Table 7).
- Service requests have mixed QoS levels of L, M and H (Mixed in Table 7).

From twenty one server configurations and twelve service request combinations, 252 problem cases are designed for an experiment. Tables 7, 8 and 9 have rows labeled as 1-1, 1-2, 1-3, 1-4, 1-5, 1-6, 2-1, 2-2, 2-3, 2-4, 2-5, 2-6, 3-1, 3-2, 3-3, 3-4, 3-5, 3-6, 3-7, 3-8, 3-9, and columns labeled as 1-1, 1-2, 1-3, 1-4, 2-1, 2-2, 2-3, 2-4, 3-1, 3-2, 3-3, 3-4. In each problem case, three sub-cases (referred as Case A, Case B and Case C throughout the dissertation) are also designed with different numbers of service requests in order to cover the following three conditions of resource capacity:

A. Each server has a sufficient resource capacity to satisfy all service requests of all clients,
B. Each server does not have a sufficient resource capacity to satisfy all service requests of all clients, but the total resource capacity of all servers is sufficient to satisfy all service requests of all clients,
C. Neither each server nor all servers together have a sufficient resource capacity to satisfy all service requests of all clients.

For each problem case, different numbers of service requests are set up to maintain three conditions of resource capacity in the sub-cases of Case A, Case B and Case C. The specific number of service requests is determined by looking into resource usages based on *F* functions of service-resource relations and the corresponding QoS values based on *G* functions of resource-QoS relations (as shown in Table 6) to meet the QoS requirements of all the service requests (as shown in Table 4) under the given

resource capacity (as shown in Table 5). For example, in the problem case with server configuration 1 of Step 1 experiment, two servers are communication-centered servers with one resource level of S, which is 30 as the capacity limit of CPU resource and 100 as the capacity limit of bandwidth resource. If all service requests are communication intensive services with one QoS level of L, fifteen service requests are designed for Case A in order that each server has a sufficient resource capacity to meet the QoS requirements of all the fifteen service requests. For Case B, where overall resource capacity is sufficient to serve all service requests with the same capacity limits of 30 for CPU resource and 100 for bandwidth resource on two servers, the number of service requests is increased to thirty five. Similarly for Case C, where overall resource capacity is not sufficient to serve all service requests with the same resource capacity limits, a total of fifty service requests is designed.

Each experiment has a total of 756 problem cases, which consist of 252 problem cases for Case A, 252 problem cases for Case B and 252 problem cases for Case C. The Step 1 and Step 2 experiments have an even number of servers (i.e. two servers), while Step 3 experiment has an odd number of servers (i.e. three servers). Hence, for the problem cases with server configurations 13 through 21 involving two server types, the first server is the communication-centered server and the second server is the computation-centered server in Step 1 and Step 2 experiments, while the first server is the communication-centered server and the other two are the computation-centered servers in Step 3 experiment. Note that, in all the experiments, each server provides both service types of the communication intensive service and the computation intensive service.

2.5 Research Methodology

2.5.1 The analysis of optimal solutions to develop heuristics. The optimal solution to each problem in Step 1, Step 2 and Step 3 experiments in Chapter 2.4 is analyzed to gain insights into the resource allocation decision made in the optimal solution. Based on the analysis and insights gained from the optimal solutions, important heuristics are developed for Case A, B and C of resource-service conditions and described in this section.

For all problems of Case A where each server has the sufficient resource capacity to satisfy all service requests of all clients, the optimal solutions assign most service requests to one server (i.e., the first server with $i = 1$ in the problem formulation). Hence, the following heuristic is identified for Case A as shown in Table 10. The probability value in Table 10 is arbitrarily chosen to obtain heuristic solutions, and Table 14 summarizes two different sets of probability parameters used for extended experiments in Chapter 2.6. Figure 1 shows the coverage of Case A problems by heuristic A-1. To implement and test this heuristic in the experiments described in Chapter 2.6, the server with $i = 1$ is designated as the dominant server.

Table 10

The Heuristics for Case A Problems

Heuristic	Description
A-1	Designate one server as the dominant server, and select the dominant server to serve a service request with the probability of α and other server(s) to serve the service request with the probability of (1-α). The parameter, α, takes a value in (0, 1] and is closer to 1 than 0 (e.g., 0.9).

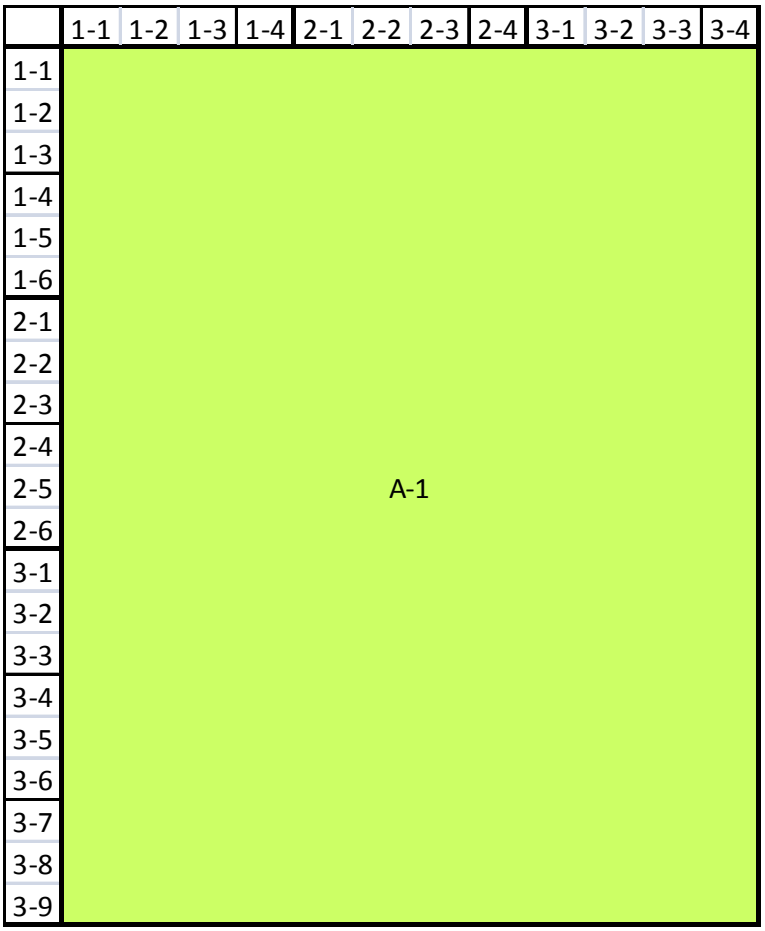


Figure 1. The Coverage of Case A Problems by Heuristic A-1.

The optimal solutions to the problems of Case B, where each server does not have the sufficient resource capacity to satisfy all service requests but the total resource capacity of all servers is sufficient to satisfy all service requests, reveal five different heuristics as shown in Table 11. Probability values in Table 11 are arbitrarily chosen to obtain heuristic solutions, and Table 14 summarizes two different sets of probability parameters used for extended experiments in Chapter 2.6.

Table 11

The Heuristics for Case B Problems

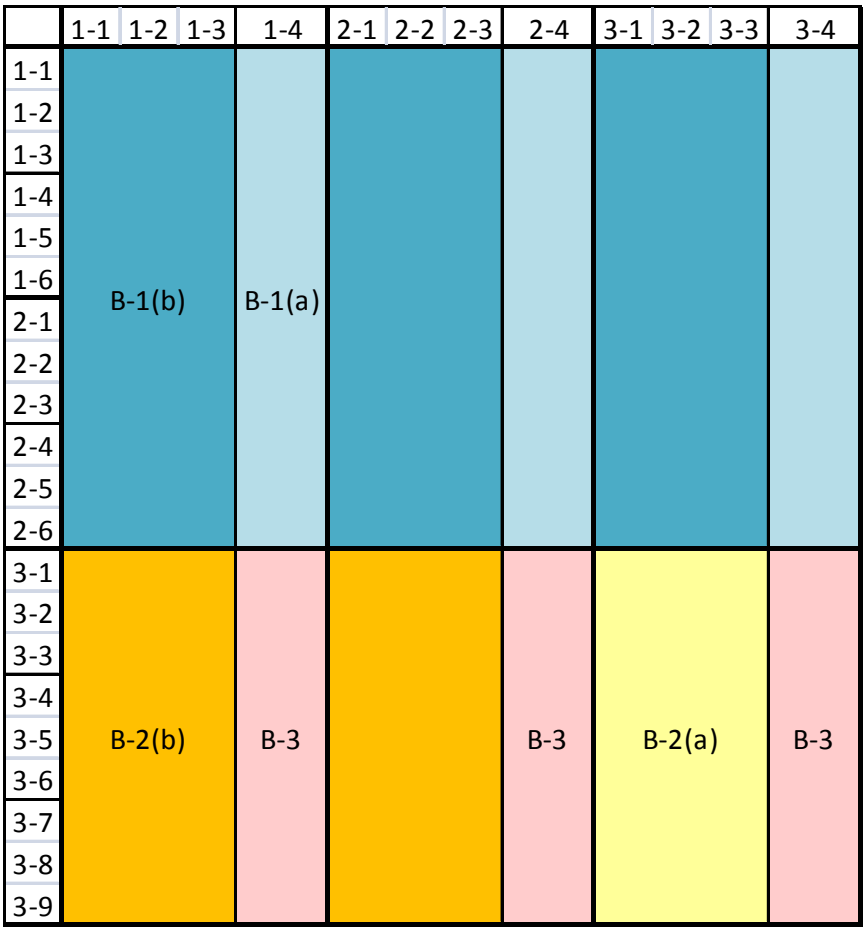
Heuristic	Description
B-1(a)	Select a server randomly to serve a service request.
B-1(b)	Designate a server as the dominant server (i.e., the server with $i = 1$), and select the dominant server to serve a service request with the probability of β and another server to serve the service request with the probability of $(1- \beta)$. The parameter, β, takes a value in (0, 1] and is closer to 1. This heuristic is same as A-1 applying to B cases.
B-2(a)	Select a server of one server type (e.g., communication-centered server) randomly to serve a service request of the same type (e.g., communication intensive service) with the probability of γ and a server of a different server type (e.g., computation-centered server) randomly to serve the service request with the probability of $(1- \gamma)$. The parameter, γ, takes a value in (0, 1] and is closer to 1.
B-2(b)	Designate a server of each server type as the dominant server of the server type, select the dominant server of one server type to serve a service request of the same type with the probability of γ, and the dominant server of a different server type to serve the service request with the probability of $(1- \gamma)$. The parameter, γ, takes a value in (0, 1] and is closer to 1.
B-3	Select a server of one server type (e.g., communication-centered server) randomly to serve a service request of the same type (e.g., communication intensive service) with a given QoS level of L, M or H with the corresponding probability of δ_1, δ_2 or δ_3, respectively, and a server of a different server type (e.g., computation-centered server) to serve the service request with the given QoS level of L, M or H with the probability of $(1 - \delta_1)$, $(1 - \delta_2)$ or $(1 - \delta_3)$, respectively. Each parameter, δ_1, δ_2 or δ_3, takes a value in (0, 1], and $\delta_1 \leq \delta_2 \leq \delta_3$.

Heuristic B-2(a) has a server of one server type (e.g., communication-centered server) serve more service requests of the same service type (e.g., communication intensive service) and less service requests of a different service type (e.g., computation intensive service). Heuristic B-2(a) is employed under only one condition when the total resource requirements from all service requests takes at least 70% of the total resource

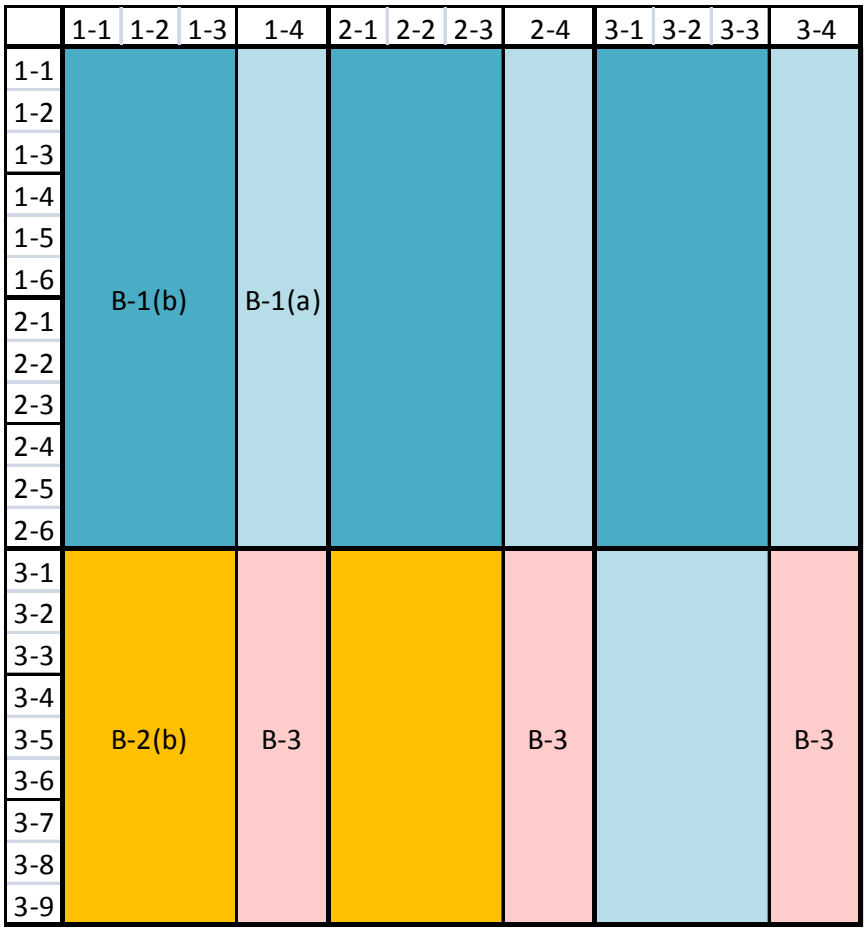
capacity of all servers, that is, when the total resource capacity is tight to serve the service requests. Hence, Figure 2 has Part (a) and Part (b) since Heuristic B-2(a) is employed only under this condition.

Figure 2 shows the coverage of Case B problems by the heuristics as follows.

- If all servers are either communication-centered servers or computation-centered servers, apply Heuristic B-1(a) and Heuristic B-1(b) (see rows 1-1 to 1-6 and 2-1 to 2-6).
- If some servers are communication-centered servers and others are computation-centered servers, and all service requests have the same QoS level of L, M or H, apply Heuristic B-2(a) and Heuristic B-2(b) under the condition of the tight resource capacity but apply Heuristic B-1(a) and Heuristic B-2(b) not under the condition of the tight resource capacity (see rows 3-1 to 3-9 and columns 1-1 to 1-3, 2-1 to 2-3 and 3-1 to 3-3).
- If some servers are communication-centered servers and others are computation-centered servers, and service requests have the mixed QoS levels of L, M and H, apply Heuristic B-3 (see rows 3-1 to 3-9 and columns 1-4, 2-4 and 3-4).



(a) Under the Condition of Heuristic B-2(a)



(b) Not Under the Condition of Heuristic B-2(a)

Figure 2. The Coverage of Case B Problems by the Heuristics.

For all problem scenarios in Case B, server types and service types need to be carefully looked into and compared with each other in finding important heuristics from the optimal solutions. Table 12 shows and examines service decisions in the optimal solutions from Step 1 experiment for two problem cases with server configuration 13 and with seven clients and ten clients respectively.

Table 12

Service Decisions made in the Optimal Solutions for Two Problem Cases in Case B

Problem Case	Client	Service	QoS Requirement ($Q^{l}_{kp_s}$)	Server	Service parameter (A_{kd_si}) and QoS provision (Q_{kp_si})
server conf. 13	$k = 1$	comm. intensive service	M (15)	comm. centered server	$A_{1,1,1}$: 3 $Q_{1,1,1}$: 15.60
	$k = 2$	comm. intensive service	M (15)	comm. centered server	$A_{2,1,1}$: 3 $Q_{2,1,1}$: 15.60
	$k = 3$	comm. intensive service	M (15)	comp. centered server	$A_{3,1,2}$: 3 $Q_{3,1,2}$: 15.60
	$k = 4$	comm. intensive service	M (15)	comm. centered server	$A_{4,1,1}$: 3 $Q_{4,1,1}$: 15.60
	$k = 5$	comm. intensive service	M (15)	comm. centered server	$A_{5,1,1}$: 3 $Q_{5,1,1}$: 15.60
	$k = 6$	comm. intensive service	M (15)	comm. centered server	$A_{6,1,1}$: 3 $Q_{6,1,1}$: 15.60
	$k = 7$	comm. intensive service	M (15)	comm. centered server	$A_{7,1,1}$: 3 $Q_{7,1,1}$: 15.60
server conf. 13	$k = 1$	comm. intensive service	M (15)	comm. centered server	$A_{1,1,1}$: 3 $Q_{1,1,1}$: 15.60
	$k = 2$	comm. intensive service	L (5)	comp. centered server	$A_{2,1,2}$: 1 $Q_{2,1,2}$: 5.20
	$k = 3$	comm. intensive service	M (15)	comp. centered server	$A_{3,1,2}$: 3 $Q_{3,1,2}$: 15.60
	$k = 4$	comm. intensive service	M (15)	comm. centered server	$A_{4,1,1}$: 3 $Q_{4,1,1}$: 15.60
	$k = 5$	comm. intensive service	L (5)	comm. centered server	$A_{5,1,1}$: 1 $Q_{5,1,1}$: 5.20
	$k = 6$	comm. intensive service	H (25)	comm. centered server	$A_{6,1,1}$: 5 $Q_{6,1,1}$: 26.00
	$k = 7$	comm. intensive service	L (5)	comp. centered server	$A_{7,1,2}$: 1 $Q_{7,1,2}$: 5.20
	$k = 8$	comm. intensive service	L (5)	comm. centered server	$A_{8,1,1}$: 1 $Q_{8,1,1}$: 5.20
	$k = 9$	comm. intensive service	L (5)	comm. centered server	$A_{9,1,1}$: 1 $Q_{9,1,1}$: 5.20
	$k = 10$	comm. intensive service	H (25)	comm. centered server	$A_{10,1,1}$: 5 $Q_{10,1,1}$: 26.00

Server configuration 13 has two servers: the first server is the communication-centered server with the resource level of S, and the second server is the computation-centered server with the resource level of S. For the problem case with seven clients in Table 12, all service requests are communication intensive services with one QoS level of M. The optimal solution shows that the communication-centered server as the first server selects to serve six service requests of the communication intensive service (client k = 1, 2, 4 through 7), while the computation-centered server as the second server selects to serve one service request of the communication intensive service (client k = 3). It can be generalized to Heuristic B-2(b) such that a server of one server type (e.g., communication-centered server) serves a service request of the same type (e.g., communication intensive service) with a higher probability than the probability a server of a different server type (e.g., computation-centered server) has.

For another problem case with ten clients in Table 12, all service requests are communication intensive services with mixed QoS levels of L, M and H. The optimal solution shows that the communication-centered server as the first server selects to serve three out of five service requests of the communication intensive service with the QoS level of L (client k = 5, 8 and 9), two out of three service requests of the communication intensive service with the QoS level of M (client k = 1 and 4) and all two service requests of the communication intensive service with the QoS level of H (client k = 6 and 10), while the computation-centered server as the second server selects to serve two out of five service requests of the communication intensive service with the QoS level of L (client k = 2 and 7) and one out of three service requests of the communication intensive service with the QoS level of M (client k = 3). Hence, it can be generalized to Heuristic

B-3 such that a server of one server type (e.g., communication-centered server) serves a service request of the same type (e.g., communication intensive service) with a given QoS level of L, M or H with the corresponding probability of δ_1, δ_2 or δ_3 respectively, and a server of a different server type (e.g., computation-centered server) serves the service request with the given QoS level of L, M or H with the probability of $(1 - \delta_1)$, $(1 - \delta_2)$ or $(1 - \delta_3)$ respectively, taking a value in (0, 1], and $\delta_1 \leq \delta_2 \leq \delta_3$.

For Case C problems, not only each server but all servers together do not have the sufficient resource capacity to satisfy all service requests, Table 13 gives the heuristics for Case C problems based on the analysis of their optimal solutions to these problems. Probability values mentioned in Table 13 are arbitrarily chosen to obtain heuristic solutions, and Table 14 summarizes two different sets of probability parameters used for extended experiments in Chapter 2.6. Figure 3 shows the coverage of Case C problems by these heuristics.

Table 13

The Heuristics for Case C Problems

Heuristic	Description
C-1	Select a server randomly to serve a service request. If the selected server is full, then a service request is randomly assigned to another server. Drop a service request if its QoS requirement cannot be satisfied by the available resource capacity. This heuristic is same as B-1(a) with the addition of dropping a service request due to the insufficient capacity.
C-2	Select a server of one server type (e.g., communication-centered server) randomly to serve a service request of the same type (e.g., communication intensive service) with the probability of γ and a server of a different server type (e.g., computation-centered server) to serve the service request with the probability of (1- γ). Drop a service request if its QoS requirement cannot be satisfied by the available resource capacity. The parameter, γ, takes a value in (0, 1] and is closer to 1. This heuristic is same as B-2(a) with the addition of dropping a service request due to the insufficient capacity.
C-3	Select a server of one server type (e.g., communication-centered server) randomly to serve a service request of the same service type (e.g., communication intensive service) with a given QoS level of L, M or H at the corresponding probability of δ_1, δ_2 or δ_3, respectively, and a server of a different server type (e.g., computation-centered server) to serve the service request with the given QoS level of L, M or H with the probability of $(1-\delta_1)$, $(1-\delta_2)$ or $(1-\delta_3)$, respectively. Drop a service request if its QoS requirement cannot be satisfied by the available resource capacity. Each parameter, δ_1, δ_2 or δ_3, takes a value in (0, 1], and $\delta_1 \leq \delta_2 \leq \delta_3$. This heuristic is same as B-3 with the addition of dropping a service request due to the insufficient capacity.

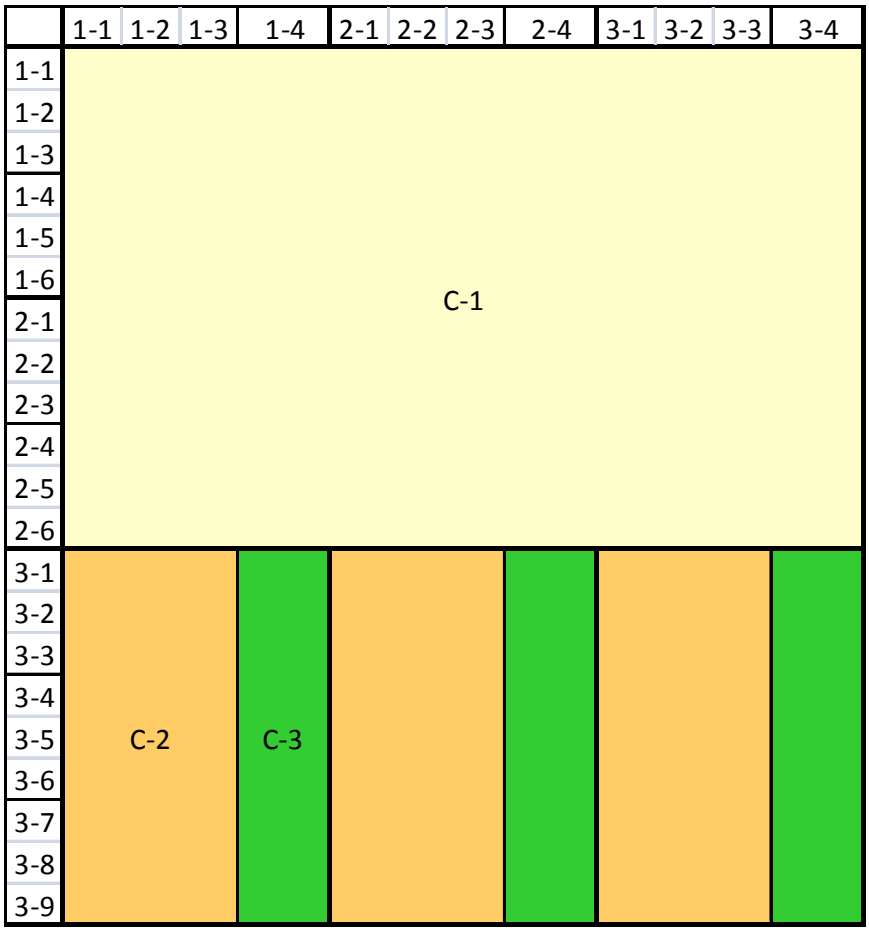


Figure 3. The Coverage of Case C Problems by the Heuristics.

Figure 3 shows the coverage of Case C problems by the heuristics as follows.

- If all servers are either communication-centered servers or computation-centered servers, apply Heuristic C-1 (see rows 1-1 to 1-6 and rows 2-1 to 2-6).
- If some servers are communication-centered servers and the others are computation-centered servers, and all service requests have the same QoS level of L, M or H, apply Heuristic C-2 (see rows 3-1 to 3-9 and columns 1-1 to 1-3, 2-1 to 2-3 and 3-1 to 3-3).
- If some servers are communication-centered servers and the others are computation-centered servers, and all service requests have the mixed QoS levels

of L, M and H, apply Heuristic C-3 (see rows 3-1 to 3-9 and columns 1-4, 2-4 and 3-4).

Table 14

Two Sets of Probability Parameters used for Obtaining Heuristic Solutions

Parameter	α	β	γ	δ_1	δ_2	δ_3
Set 1	0.90	0.80	0.85	0.75	0.80	0.85
Set 2	0.98	0.90	0.80	0.70	0.80	0.90

For all problem scenarios in Case C, server types and service types need to be carefully looked into and compared with each other in finding important heuristics from the optimal solutions. Table 15 shows and examines service decisions in the optimal solutions from Step 1 experiment for two problem cases with server configuration 16 and seven clients and with server configuration 13 and sixteen clients.

Table 15

Service Decisions made in the Optimal Solutions for Two Problem Cases in Case C

Problem Case	Client	Service	QoS Requirement ($Q^{l}_{kp_s}$)	Server	Service parameter (A_{kd_si}) and QoS provision (Q_{kp_si})	Estimated objective value
server conf. 16	k = 1	comm. intensive service	H (25)	comm. centered server	$A_{1,1,1}$: 5 $Q_{1,1,1}$: 26.00	0.04
	k = 2	comm. intensive service	H (25)	comm. centered server	$A_{2,1,1}$: 5 $Q_{2,1,1}$: 26.00	0.04
	k = 3	comm. intensive service	H (25)	comm. centered server	$A_{3,1,1}$: 5 $Q_{3,1,1}$: 26.00	0.04
	k = 4	comm. intensive service	H (25)	comm. centered server	$A_{4,1,1}$: 5 $Q_{4,1,1}$: 26.00	0.04
	k = 5	comm. intensive service	H (25)	comp. centered server	$A_{5,1,2}$: 5 $Q_{5,1,2}$: 26.00	0.04
	k = 6	comm. intensive service	H (25)	none	none	0.04
	k = 7	comm. intensive service	H (25)	none	none	0.04
server conf. 13	k = 1	comm. intensive service	M (15)	comm. centered server	$A_{1,1,1}$: 3 $Q_{2,1,2}$: 15.60	0.04
	k = 2	comm. intensive service	M (15)	comm. centered server	$A_{2,1,1}$: 3 $Q_{2,1,1}$: 15.60	0.04
	k = 3	comm. intensive service	H (25)	none	none	0.04
	k = 4	comm. intensive service	H (25)	comm. centered server	$A_{4,1,1}$: 5 $Q_{4,1,1}$: 26.00	0.04
	k = 5	comp. intensive service	L (6)	comm. centered server	$A_{5,1,1}$: 1 $Q_{5,1,1}$: 6.10	0.02
	k = 6	comm. intensive service	H (25)	comm. centered server	$A_{6,1,1}$: 5 $Q_{6,1,1}$: 26.00	0.04
	k = 7	comp. intensive service	M (17)	none	none	0.07
	k = 8	comm. intensive service	L (5)	comm. centered server	$A_{8,1,1}$: 1 $Q_{8,1,1}$: 5.20	0.04
	k = 9	comp. intensive service	H (30)	comp. centered server	$A_{9,1,2}$: 5 $Q_{9,1,2}$: 30.50	0.02
	k = 10	comp. intensive service	L (6)	comm. centered server	$A_{10,1,1}$: 1 $Q_{10,1,1}$: 6.10	0.02
	k = 11	comp. intensive service	H (30)	comp. centered server	$A_{11,1,2}$: 5 $Q_{11,1,2}$: 30.50	0.02
	k = 12	comp. intensive service	H (30)	comp. centered server	$A_{12,1,2}$: 5 $Q_{12,1,2}$: 30.50	0.02
	k = 13	comp. intensive service	H (30)	comp. centered server	$A_{13,1,2}$: 5 $Q_{13,1,2}$: 30.50	0.02
	k = 14	comm. intensive service	L (5)	comm. centered server	$A_{14,1,1}$: 1 $Q_{14,1,1}$: 5.20	0.04
	k = 15	comm. intensive service	L (5)	comm. centered server	$A_{15,1,1}$: 1 $Q_{15,1,1}$: 5.20	0.04
	k = 16	comm. intensive service	H (25)	none	none	0.04

For the problem case with server configuration 16 and seven clients in Table 15, there are two servers used: the first server is the communication-centered server with the resource level of S, and the second server is the computation-centered server with the resource level of M, and all service requests of seven clients are communication intensive services with one QoS level of H. The optimal solution shows that the communication-centered server as the first server selects to serve four service requests of the communication intensive service (client k =1 through 4), while the computation-centered server as the second server selects to serve one service request of the communication intensive service (client k = 5). It can be generalized to Heuristic C-2 such that a server of one server type (e.g., a communication-centered server) serves a service request of the same type (e.g., a communication intensive service) with higher probability than the probability that a server of a different server type (e.g., computation-centered server) has.

For the problem case with server configuration 13 and sixteen clients in Table 15, there are two servers: the first server is the communication-centered server with the resource level of S and the second server is the computation-centered server with the resource level of S, and sixteen clients' service requests: some service requests are communication intensive services with mixed QoS levels of L, M and H and the others are computation intensive services with mixed QoS levels of L, M and H. The optimal solution shows that the communication-centered server as the first server selects to serve three service requests of the communication intensive service with the QoS level of L (client k =8, 14 and 15), two service requests of the communication intensive service with the QoS level of M (client k = 1 and 2), two service requests of the communication

intensive service with the QoS level of H (client k = 4 and 6) and two service requests of the computation intensive service with the QoS level of L (client k = 5 and 10), while the computation-centered server as the second server selects to serve four service requests of the computation intensive service with the QoS level of H (client k = 9, 11 through 13). It can be generalized to Heuristic C-3 such that a server of one server type (e.g., communication-centered server) serves a service request of the same type (e.g., communication intensive service) with a given QoS level of L, M or H with the corresponding probability of δ_1, δ_2 or δ_3 respectively, and a server of a different server type (e.g., computation-centered server) to serve the service request with the given QoS level of L, M or H with the probability of $(1-\delta_1)$, $(1-\delta_2)$ or $(1-\delta_3)$ respectively, taking a value in (0, 1] and $\delta_1 \leq \delta_2 \leq \delta_3$.

2.5.2 Statistical data supporting the heuristics. Statistics of the experimental results from Step 1, Step 2 and Step 3 show the support of the heuristics defined in Tables 10, 11 and 13. Nineteen variables shown in Table 17 are collected to obtain the statistics that address two questions for each heuristic: 1) what statistics (what variables with what values) support the heuristic, and 2) those types of statistics do not show up under other heuristics. Tables 18, 19 and 20 show the collection of all variables in Step 1, Step 2 and Step 3 experiment respectively. Var2 through Var19 show an average in the first row and the standard deviation in the second row in Tables 18, 19 and 20.

In Tables 18, 19 and 20, compartment numbers correspond to those defined in Figures 1, 2(a), 2(b) and 3. Each compartment may contain more than one sub-compartment depending on different types of servers and service requests. Compartment I has three sub-compartments, I-1, I-2 and I-3, with all comm. intensive services, all

comp. intensive services and both comm. and comp. intensive services, respectively. Compartment II has three sub-compartments, II-1, II-2 and II-3, with all comm. intensive services, all comp. intensive services and both comm. and comp. intensive services respectively. Compartment III has two sub-compartments, III-1 and III-2, with all comm. intensive services and all comp. intensive services respectively. Compartment IV has one sub-compartment. Compartment V has three sub-compartments, V-1, V-2 and V-3, with all comm. intensive services, all comp. intensive services and both comm. and comp. intensive services respectively.

For each sub-compartment, the statistics obtained from Case A, Case B under tight resource capacity, Case B not under tight resource capacity and Case C problems are shown with each related heuristic. Table 16 shows the design of experimental setup for Case B problems not under tight resource capacity used in Step 1, Step 2 and Step 3 experiments. Design of experimental setups for the other problem cases used in Step 1, Step 2 and Step 3 experiments are shown in Tables 7, 8 and 9, respectively.

Table 16

Design of Experimental Setup not under Tight Resource Capacity Condition for Case B in Step 1, Step 2 and Step 3 Experiments

Service Requests / Server Configuration	All communication intensive				All computation intensive				Both service types			
	All-L (1-1)	All-M (1-2)	All-H (1-3)	Mixed (1-4)	All-L (2-1)	All-M (2-2)	All-H (2-3)	Mixed (2-4)	All-L (3-1)	All-M (3-2)	All-H (3-3)	Mixed (3-4)
1. All communication centered at S-S (1-1)	21	7	5	11	6	2	2	2	17	8	4	6
	13	7	6	10	6	4	3	5	17	9	7	14
	18	7	6	8	7	2	2	3	22	8	7	9
2. All communication centered at M-M (1-2)	41	14	9	21	7	3	2	3	20	8	10	10
	17	9	7	14	9	6	4	6	24	12	9	19
	20	8	6	9	8	4	2	4	25	9	7	11
3. All communication centered at L-L (1-3)	72	24	15	34	8	3	2	4	13	24	16	25
	31	16	13	26	12	8	5	9	22	16	14	32
	22	8	7	9	10	3	3	4	28	10	8	13
4. All communication centered at S-M (1-4)	41	14	9	21	7	3	2	3	21	7	9	10
	17	9	7	13	9	6	4	6	16	12	9	17
	20	8	6	9	8	3	2	5	25	9	6	11
5. All communication centered at S-L (1-5)	71	24	15	34	8	3	2	4	12	24	15	24
	31	16	13	22	12	8	5	7	22	16	14	16
	23	8	6	10	10	3	3	4	29	10	7	13
6. All communication centered at M-L (1-6)	72	24	15	34	8	3	2	4	13	24	15	24
	31	16	13	22	12	8	5	7	22	16	14	16
	23	8	6	10	10	3	3	4	29	10	7	14
7. All computation centered at S-S (2-1)	6	2	2	2	21	7	5	11	13	8	3	4
	7	4	3	4	14	10	6	9	16	5	7	8
	7	2	2	3	18	6	5	8	22	7	6	9
8. All computation centered at M-M (2-2)	7	3	2	3	41	14	9	21	14	5	6	6
	9	5	4	6	21	14	9	11	15	9	8	13
	7	2	2	4	20	7	5	8	25	8	6	12
9. All computation centered at L-L (2-3)	8	3	2	4	72	24	15	35	24	8	8	12
	11	6	5	7	27	18	11	16	18	15	12	15
	8	3	2	4	22	8	6	9	27	10	7	12
10. All computation centered at S-M (2-4)	7	3	2	3	41	14	9	21	14	6	6	5
	9	5	4	6	21	14	9	16	15	9	8	13
	7	2	2	4	20	7	5	8	25	8	6	12
11. All computation	8	3	2	4	71	24	15	35	24	8	8	11

centered at S-L (2-5)	11 8	6 3	5 2	7 5	27 22	18 8	11 6	20 9	18 27	15 9	12 7	15 13
12. All computation centered at M-L (2-6)	8 11 8	3 6 3	2 5 2	4 7 3	71 27 22	24 18 8	15 11 6	35 20 9	24 18 27	8 15 9	8 12 7	12 16 12
13. Both server types (communication, computation) at S-S (3-1)	21 13 18	7 7 6	5 6 5	11 10 7	21 14 18	7 10 6	5 6 5	11 11 8	16 16 12	4 9 4	3 7 4	4 11 5
14. Both server types (communication, computation) at M-M (3-2)	41 17 20	14 9 7	9 7 5	21 11 10	41 21 20	14 14 7	9 9 5	21 16 8	14 20 14	6 14 6	4 10 4	14 11 6
15. Both server types (communication, computation) at L-L (3-3)	71 31 22	24 16 8	15 13 6	34 21 10	71 27 22	24 18 8	15 11 6	35 20 9	14 24 17	6 18 6	4 9 5	6 19 7
16. Both server types (communication, computation) at S-M (3-4)	21 13 18	7 7 6	5 6 5	11 9 8	41 21 20	14 14 7	9 9 5	21 16 8	14 15 14	6 9 5	4 7 4	12 10 5
17. Both server types (communication, computation) at S-L (3-5)	21 13 18	7 7 6	5 6 5	11 10 7	71 27 21	24 18 7	15 11 6	35 20 8	14 16 14	5 9 5	4 7 4	14 11 6
18. Both server types (communication, computation) at M-L (3-6)	41 17 20	14 9 7	9 7 5	21 13 9	71 27 22	24 18 7	15 11 6	35 20 8	14 18 15	6 11 6	4 8 4	12 12 6
19. Both server types (communication, computation) at M-S (3-7)	41 17 20	14 9 7	9 7 5	21 13 9	21 14 18	7 10 6	5 6 5	11 11 8	14 15 13	5 9 5	3 7 4	14 9 6
20. Both server types (communication, computation) at L-S (3-8)	71 31 22	24 16 8	15 13 6	33 24 9	21 14 18	7 10 6	5 6 5	11 11 8	14 25 15	6 14 5	3 7 5	5 12 7
21. Both server types (communication, computation) at L-M (3-9)	71 31 22	24 16 8	15 13 6	33 25 9	41 21 20	14 14 7	9 9 5	21 15 8	14 26 16	5 12 6	4 8 4	6 14 8

Table 17

Definition of Variables for the Supporting Statistics

Variable	Description
Var1	Proportion of problem cases where the first server serves equal or higher number of service requests than the other server(s) (= Problem cases where the first server serves at least the same number of service requests as the other server(s) / All problem cases)
Var2	Proportion of comm. intensive services (L) served by the first server (= The number of comm. intensive services (L) served by the first server / Total number of comm. intensive services (L) served by all servers)
Var3	Proportion of comm. intensive services (L) served by the second server (= The number of comm. intensive services (L) served by the second server / Total number of comm. intensive services (L) served by all servers)
Var4	Proportion of comm. intensive services (L) served by the third server if exist (= The number of comm. intensive services (L) served by the third server / Total number of comm. intensive services (L) served by all servers)
Var5	Proportion of comm. intensive services (M) served by the first server (= The number of comm. intensive services (M) served by the first server / Total number of comm. intensive services (M) served by all servers)
Var6	Proportion of comm. intensive services (M) served by the second server (= The number of comm. intensive services (M) served by the second server / Total number of comm. intensive services (M) served by all servers)
Var7	Proportion of comm. intensive services (M) served by the third server if exist (= The number of comm. intensive services (M) served by the third server / Total number of comm. intensive services (M) served by all servers)
Var8	Proportion of comm. intensive services (H) served by the first server (= The number of comm. intensive services (H) served by the first server / Total number of comm. intensive services (H) served by all servers)
Var9	Proportion of comm. intensive services (H) served by the second server (= The number of comm. intensive services (H) served by the second server / Total number of comm. intensive services (H) served by all servers)
Var10	Proportion of comm. intensive services (H) served by the third server if exist (= The number of comm. intensive services (H) served by the third server / Total number of comm. intensive services (H) served by all servers)
Var11	Proportion of comp. intensive services (L) served by the first server (= The number of comp. intensive services (L) served by the first server / Total number of comp. intensive services (L) served by all servers)

Var12	Proportion of comp. intensive services (L) served by the second server (= The number of comp. intensive services (L) served by the second server / Total number of comp. intensive services (L) served by all servers)
Var13	Proportion of comp. intensive services (L) served by the third server if exist (= The number of comp. intensive services (L) served by the third server / Total number of comp. intensive services (L) served by all servers)
Var14	Proportion of comp. intensive services (M) served by the first server (= The number of comp. intensive services (M) served by the first server / Total number of comp. intensive services (M) served by all servers)
Var15	Proportion of comp. intensive services (M) served by the second server (= The number of comp. intensive services (M) served by the second server / Total number of comp. intensive services (M) served by all servers)
Var16	Proportion of comp. intensive services (M) served by the third server if exist (= The number of comp. intensive services (M) served by the third server / Total number of comp. intensive services (M) served by all servers)
Var17	Proportion of comp. intensive services (H) served by the first server (= The number of comp. intensive services (H) served by the first server / Total number of comp. intensive services (H) served by all servers)
Var18	Proportion of comp. intensive services (H) served by the second server (= The number of comp. intensive services (H) served by the 2nd server / Total number of comp. intensive services (H) served by all servers)
Var19	Proportion of comp. intensive services (H) served by the third server if exist (= The number of comp. intensive services (H) served by the third server / Total number of comp. intensive services (H) served by all servers)

Table 18

Statistics for Step 1 Experimental Result

Compartment #	Heuristic	Var1	Var2	Var3	Var4	Var5	Var6	Var7	Var8	Var9	Var10	Var11	Var12	Var13	Var14	Var15	Var16	Var17	Var18	Var19
I-1. Rows 1-1 thru 2-6 Columns 1-1 thru 1-3	A-1	0.58	0.01	0.99	NA	0.59	0.41	NA	1.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.02	0.02	NA	0.30	0.30	NA	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(b)	0.50	0.43	0.57	NA	0.44	0.56	NA	0.48	0.52	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.08	0.08	NA	0.11	0.11	NA	0.10	0.10	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(b)	0.56	0.20	0.80	NA	0.58	0.42	NA	0.57	0.43	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.19	0.19	NA	0.22	0.22	NA	0.20	0.20	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-1	0.61	0.44	0.56	NA	0.42	0.58	NA	0.45	0.55	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.09	0.09	NA	0.10	0.10	NA	0.09	0.09	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
I-2. Rows 1-1 thru 2-6 Columns 2-1 thru 2-3	A-1	0.94	NA	NA	NA	NA	NA	NA	NA	NA	NA	1.00	0.00	NA	0.60	0.40	NA	1.00	0.00	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.00	0.00	NA	0.29	0.29	NA	0.00	0.00	NA
	B-1(b)	0.67	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.49	0.51	NA	0.44	0.56	NA	0.48	0.52	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.11	0.11	NA	0.12	0.12	NA	0.10	0.10	NA
	B-1(b)	0.78	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.74	0.26	NA	0.57	0.43	NA	0.57	0.43	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.22	0.22	NA	0.21	0.21	NA	0.20	0.20	NA
	C-1	0.61	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.44	0.56	NA	0.42	0.58	NA	0.45	0.55	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.09	0.09	NA	0.10	0.10	NA	0.09	0.09	NA
I-3. Rows 1-1 thru 2-6 Columns 3-1 thru 3-3	A-1	0.78	0.33	0.67	NA	0.70	0.30	NA	1.00	0.00	NA	0.47	0.53	NA	0.62	0.38	NA	1.00	0.00	NA
			0.49	0.49	NA	0.30	0.30	NA	0.00	0.00	NA	0.40	0.40	NA	0.42	0.42	NA	0.00	0.00	NA
	B-1(b)	0.53	0.52	0.48	NA	0.43	0.57	NA	0.46	0.54	NA	0.47	0.53	NA	0.56	0.44	NA	0.46	0.54	NA
			0.24	0.24	NA	0.13	0.13	NA	0.16	0.16	NA	0.12	0.12	NA	0.21	0.21	NA	0.14	0.14	NA
	B-1(b)	0.75	0.71	0.29	NA	0.67	0.33	NA	0.63	0.37	NA	0.46	0.54	NA	0.48	0.52	NA	0.68	0.33	NA
			0.37	0.37	NA	0.33	0.33	NA	0.38	0.38	NA	0.28	0.28	NA	0.36	0.36	NA	0.41	0.41	NA
	C-1	0.58	0.52	0.48	NA	0.47	0.53	NA	0.43	0.57	NA	0.49	0.51	NA	0.42	0.58	NA	0.43	0.57	NA
			0.10	0.10	NA	0.14	0.14	NA	0.25	0.25	NA	0.17	0.17	NA	0.10	0.10	NA	0.14	0.14	NA
II-1. Rows 1-1 thru 2-6 Column 1-4	A-1	0.25	0.00	1.00	NA	0.35	0.65	NA	0.38	0.63	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.00	0.00	NA	0.42	0.42	NA	0.48	0.48	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(a)	0.17	0.17	0.83	NA	0.41	0.59	NA	0.59	0.41	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.27	0.27	NA	0.39	0.39	NA	0.42	0.42	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(a)	0.08	0.00	1.00	NA	0.56	0.44	NA	0.70	0.30	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.00	0.00	NA	0.36	0.36	NA	0.34	0.34	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-1	0.00	0.12	0.88	NA	0.38	0.62	NA	0.68	0.33	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.14	0.14	NA	0.32	0.32	NA	0.39	0.39	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
II-2. Rows 1-1 thru 2-6 Column 2-4	A-1	0.75	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.66	0.34	NA	0.51	0.49	NA	0.54	0.46	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.40	0.40	NA	0.41	0.41	NA	0.50	0.50	NA
	B-1(a)	0.42	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.43	0.57	NA	0.32	0.68	NA	0.41	0.59	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.37	0.37	NA	0.24	0.24	NA	0.36	0.36	NA
	B-1(a)	0.67	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.70	0.30	NA	0.76	0.24	NA	0.15	0.85	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.26	0.26	NA	0.24	0.24	NA	0.31	0.31	NA
	C-1	0.50	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.43	0.57	NA	0.35	0.65	NA	0.44	0.56	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.27	0.27	NA	0.34	0.34	NA	0.32	0.32	NA
II-3. Rows 1-1 thru 2-6 Column 3-4	A-1	0.50	0.40	0.60	NA	0.46	0.54	NA	0.25	0.75	NA	0.89	0.11	NA	0.50	0.50	NA	0.17	0.83	NA
			0.52	0.52	NA	0.51	0.51	NA	0.43	0.43	NA	0.33	0.33	NA	NA	NA	NA	0.39	0.39	NA
	B-1(a)	0.50	0.44	0.56	NA	0.38	0.62	NA	0.43	0.57	NA	0.39	0.61	NA	0.58	0.42	NA	0.42	0.58	NA
			0.40	0.40	NA	0.36	0.36	NA	0.28	0.28	NA	0.26	0.26	NA	0.14	0.14	NA	0.36	0.36	NA
	B-1(a)	0.58	0.39	0.61	NA	0.20	0.80	NA	0.69	0.31	NA	0.58	0.42	NA	0.20	0.80	NA	0.58	0.42	NA
			0.47	0.47	NA	0.23	0.23	NA	0.42	0.42	NA	0.49	0.49	NA	0.45	0.45	NA	0.51	0.51	NA
	C-1	0.67	0.53	0.47	NA	0.51	0.49	NA	0.38	0.62	NA	0.46	0.54	NA	0.69	0.31	NA	0.45	0.55	NA
			0.31	0.31	NA	0.27	0.27	NA	0.18	0.18	NA	0.36	0.36	NA	0.37	0.37	NA	0.29	0.29	NA
III-1. Rows 3-1 thru 3-9 Columns 1-1 thru 1-3	A-1	0.81	0.78	0.22	NA	0.50	0.50	NA	1.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.44	0.44	NA	0.43	0.43	NA	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-2(b)	1.00	0.85	0.15	NA	0.87	0.13	NA	0.87	0.13	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.07	0.07	NA	0.08	0.08	NA	0.06	0.06	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-2(b)	1.00	0.83	0.17	NA	0.86	0.14	NA	0.80	0.20	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.09	0.09	NA	0.10	0.10	NA	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-2	1.00	0.85	0.15	NA	0.87	0.13	NA	0.87	0.13	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.07	0.07	NA	0.08	0.08	NA	0.06	0.06	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

III-2. Rows 3-1 thru 3-9 Columns 2-1 thru 2-3	A-1	0.63	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.22	0.78	NA	0.50	0.50	NA	1.00	0.00	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.44	0.44	NA	0.43	0.43	NA	0.00	0.00	NA
	B-2(b)	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.15	0.85	NA	0.13	0.87	NA	0.13	0.87	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.07	0.07	NA	0.08	0.08	NA	0.06	0.06	NA
	B-2(b)	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.17	0.83	NA	0.15	0.85	NA	0.20	0.80	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.09	0.09	NA	0.09	0.09	NA	0.00	0.00	NA
	C-2	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.15	0.85	NA	0.13	0.87	NA	0.13	0.87	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.07	0.07	NA	0.08	0.08	NA	0.06	0.06	NA
IV. Rows 3-1 thru 3-9 Columns 3-1 thru 3-3	A-1	0.70	0.33	0.67	NA	0.67	0.33	NA	1.00	0.00	NA	0.51	0.49	NA	0.67	0.33	NA	1.00	0.00	NA
			0.50	0.50	NA	0.50	0.50	NA	0.00	0.00	NA	0.39	0.39	NA	0.43	0.43	NA	0.00	0.00	NA
	B-2(a)	0.59	0.86	0.14	NA	0.90	0.10	NA	1.00	0.00	NA	0.20	0.80	NA	0.13	0.87	NA	0.00	1.00	NA
			0.09	0.09	NA	0.05	0.05	NA	0.00	0.00	NA	0.16	0.16	NA	0.09	0.09	NA	0.00	0.00	NA
	B-1(a)	0.74	0.32	0.68	NA	0.83	0.17	NA	1.00	0.00	NA	0.65	0.35	NA	0.59	0.41	NA	0.00	1.00	NA
			0.32	0.32	NA	0.26	0.26	NA	0.00	0.00	NA	0.35	0.35	NA	0.12	0.12	NA	0.00	0.00	NA
	C-2	0.67	0.91	0.09	NA	0.91	0.09	NA	1.00	0.00	NA	0.13	0.87	NA	0.09	0.91	NA	0.08	0.92	NA
			0.05	0.05	NA	0.05	0.05	NA	0.00	0.00	NA	0.06	0.06	NA	0.04	0.04	NA	0.08	0.08	NA
V-1. Rows 3-1 thru 3-9 Column 1-4	A-1	1.00	1.00	0.00	NA	1.00	0.00	NA	1.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.00	0.00	NA	0.00	0.00	NA	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-3	0.89	0.64	0.36	NA	0.86	0.14	NA	1.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.23	0.23	NA	0.13	0.13	NA	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-3	0.89	0.60	0.40	NA	0.90	0.10	NA	1.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.22	0.22	NA	0.14	0.14	NA	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-3	1.00	0.75	0.25	NA	1.00	0.00	NA	1.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.19	0.19	NA	0.00	0.00	NA	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
V-2. Rows 3-1 thru 3-9 Column 2-4	A-1	0.89	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.86	0.14	NA	0.50	0.50	NA	0.14	0.86	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.38	0.38	NA	0.71	0.71	NA	0.38	0.38	NA
	B-3	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.29	0.71	NA	0.13	0.87	NA	0.06	0.94	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.22	0.22	NA	0.10	0.10	NA	0.17	0.17	NA
	B-3	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.22	0.78	NA	0.20	0.80	NA	0.00	1.00	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.14	0.14	NA	0.14	0.14	NA	0.00	0.00	NA
	C-3	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.17	0.83	NA	0.08	0.92	NA	0.00	1.00	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.08	0.08	NA	0.17	0.17	NA	0.00	0.00	NA
V-3. Rows 3-1 thru 3-9 Column 3-4	A-1	1.00	0.60	0.40	NA	1.00	0.00	NA	1.00	0.00	NA	0.88	0.13	NA	1.00	0.00	NA	0.71	0.29	NA
			0.55	0.55	NA	0.00	0.00	NA	0.00	0.00	NA	0.25	0.25	NA	0.00	0.00	NA	0.49	0.49	NA
	B-3	0.67	0.77	0.23	NA	0.94	0.06	NA	1.00	0.00	NA	0.31	0.69	NA	0.12	0.88	NA	0.00	1.00	NA
			0.13	0.13	NA	0.12	0.12	NA	0.00	0.00	NA	0.11	0.11	NA	0.16	0.16	NA	0.00	0.00	NA
	B-3	0.33	0.04	0.96	NA	0.64	0.36	NA	0.89	0.11	NA	0.33	0.67	NA	0.19	0.81	NA	0.00	1.00	NA
			0.11	0.11	NA	0.38	0.38	NA	0.33	0.33	NA	0.37	0.37	NA	0.26	0.26	NA	0.00	0.00	NA
	C-3	0.67	0.87	0.13	NA	0.95	0.05	NA	1.00	0.00	NA	0.30	0.70	NA	0.12	0.88	NA	0.04	0.96	NA
			0.16	0.16	NA	0.08	0.08	NA	0.00	0.00	NA	0.31	0.31	NA	0.11	0.11	NA	0.11	0.11	NA

The statistics in Table 18 are summarized below. Heuristics A-1 and B-1(b) state that a server as the dominant server (i.e., the server with $i = 1$) serves a service request with higher probability than the other server(s). The support of Heuristics A-1 and B-1(b) is shown through Var1 as follows.

- Var1 values are greater than 0.50 in 10 out of 12 problem cases, equal to 0.50 in 1 out of 12 cases and less than 0.50 in 1 out of 12 cases for Heuristic A-1. Similarly, Var1 values are greater than 0.50 in 5 out of 6 problem cases and equal to 0.50 in 1 out of 6 cases for Heuristic B-1(b). In other words, Var1 values are greater than 0.50 in 83% of all problem cases under Heuristics A-1 and B-1(b), indicating that

the first server as the dominant server serves at least same or higher number of service requests than the other server(s) in most problem cases.

- Under other heuristics, Var1 values do not show a consistent pattern indicating no dominant server for service provision.
    - For Heuristics B-1(a) and C-1, Var1 values are greater than 0.50 in 3 out of 7 problem cases (43%) and in 4 out of 6 problem cases (67%), respectively, which are smaller than 83% under Heuristics A-1 and B-1(b).
    - For Heuristic B-2(a), Var1 value is greater than 0.50 in one problem case (100%). However, this problem case involves both server types. Since its solutions are significantly affected by both server types and service types, it cannot be compared to the problem cases involving with one server type under Heuristics A-1 and B-1(b). For Heuristic C-2, Var1 values are greater than 0.50 in 2 out of 3 problem cases (67%), which is smaller than 83% under Heuristics A-1 and B-1(b).
    - For Heuristic B-2(b), Var1 values are greater than 0.50 in 2 out of 4 problem cases (50%), which is smaller than 83% under Heuristics A-1 and B-1(b).
    - For Heuristics B-3 and C-3, Var1 values are greater than 0.50 in 3 out of 6 problem cases (50%) and in 2 out of 3 problem cases (67%), respectively, which are smaller than 83% under Heuristics A-1 and B-1(b).

Heuristics B-1(a) and C-1 state that a server randomly serves a service request. The support of Heuristics B-1(a) and C-1 is shown through Var1 to Var19 as follows.

- For Heuristic B-1(a), Var1 values are greater than 0.50 in 3 out of 7 problem cases, equal to 0.50 in 1 out of 7 cases and less than 0.50 in 3 out of 7 cases. In

addition, Var2 through Var19 values show that the first server is selected with higher probability than the second server to serve more number of service requests in 3 out of 7 problem cases, with same probability as the second server in 1 out of 7 cases and with lower probability than the second server in 3 out of 7 cases. Those statistics support the random selection of a server for service provision under Heuristic B-1(a). For Heuristic C-1, Var1 values are greater than 0.50 in 4 out of 6 problem cases, equal to 0.50 in 1 out of 6 cases and less than 0.50 in 1 out of 6 cases. Var2 through Var19 values show that the first server is selected with higher probability than the second server to serve more number of service requests in 5 out of 6 problem cases and with same probability as the second server in 1 out of 6 cases. Since the problem cases under Heuristic C-1 have insufficient resource capacity to satisfy all service requests and have the second server with at least same resource capacity as the first server, the random server selection for service provision would result in selecting the second server with higher probability than the first server to serve more number of service requests due to its larger resource capacity. Therefore, Var1 through Var19 values support the random selection of a server for service provision under Heuristic C-1.

- Under other heuristics, those values do not show a consistent pattern, indicating there exists a dominant server for service provision.
    - For Heuristics A-1 and B-1(b), Var1 values are greater than 0.50 in 10 out of 12 cases (83%) and in 5 out of 6 cases (83%), respectively, which are greater than 43% and 67% under Heuristics B-1(a) and C-1. Similarly, Var2 through Var19 values show that the first server is selected with higher probability than

the second server in 8 out of 12 cases (67%) and in 3 out of 6 cases (50%) under Heuristics A-1 and B-1(b), respectively, which are greater than 43% under Heuristics B-1(a).

- Problem cases in Heuristics B-2(a), B-2(b), B-3, C-2 and C-3 involve both types of servers. Since their solutions are significantly affected by both server types and service types, they cannot be compared to the problem cases involving with one server type under Heuristics B-1(a) and C-1.

Heuristics B-2(a) and C-2 state that a server of one server type randomly serves a service request of the same type with higher probability than the other server(s) of a different type. The support of Heuristics B-2(a) and C-2 is shown through Var2 to Var19 as follows.

- The values show a consistent pattern in all 1 problem case (100%) for Heuristic B-2(a) and in all 3 cases (100%) for Heuristic C-2, as stated in these heuristics. Note that Heuristics B-2(a) and B-2(b) are same with two servers as in Step 1 and Step 2 experiments.
- Under other heuristics, those values do not show a consistent pattern, indicating that a server serves a service request regardless of its type.
  - For Heuristic A-1, the values show the same pattern in 2 out of 6 problem cases (33%), which is much smaller than 100% under Heuristics B-2(a) and C-2.
  - For Heuristic B-1(a), each server is almost equally likely to be selected for service provision regardless of a service type in one problem case.

- For Heuristics B-3 and C-3, the values show the same pattern in 5 out of 6 cases (83%) and 3 out of 3 cases (100%). However, problem cases under those heuristics involve both types of servers and service requests with mixed QoS levels. Since their solutions are significantly affected by serving different QoS levels of service requests as well as matching server types and service types, they cannot be compared to the problem cases involving with service requests of one QoS level under Heuristics B-2(a) and C-2. Similarly, Heuristics B-1(b) and C-1 cannot be compared since they do not have both server types.

Heuristic B-2(b) states that a server of each server type as the dominant one serves a service request of the same type with higher probability than the dominant server of a different type. The support of Heuristic B-2(b) is shown through Var2 to Var19 as follows.

- The values show a consistent pattern in all 4 problem cases (100%), as stated in Heuristic B-2(b). Note that Heuristics B-2(a) and B-2(b) are same with two servers as in Step 1 and Step 2 experiments. Thus, Heuristic C-2 also shows this consistent pattern as Heuristic B-2(b) in Step 1 and Step 2 experiments.
- Under other heuristics, those values do not show a consistent pattern indicating that a server serves a service request regardless of its type.
  - For Heuristic A-1, the values show the same pattern in 2 out of 6 problem cases (33%), which is much smaller than 100% under Heuristic B-2(b).
  - For Heuristic B-1(a), each server is almost equally likely to be selected for service provision regardless of a service type in one problem case.

- For Heuristics B-3 and C-3, the values show the same pattern in 5 out of 6 cases (83%) and 3 out of 3 cases (100%). However, problem cases under those heuristics involve both types of servers and service requests with mixed QoS levels. Since their solutions are significantly affected by serving different QoS levels of service requests as well as matching server types and service types, they cannot be compared to the problem cases involving with service requests of one QoS level under Heuristic B-2(b). Similarly, Heuristics B-1(b) and C-1 cannot be compared since they do not have both server types.

Heuristics B-3 and C-3 state that a server of one server type randomly serves a service request of the same type with a given QoS level of L, M or H at the corresponding probability of δ_1, δ_2 or δ_3, respectively, and a server of a different server type to serve the service request with the given QoS level of L, M or H with the probability of $(1-\delta_1)$, $(1-\delta_2)$ or $(1-\delta_3)$, respectively. Each parameter, δ_1, δ_2 or δ_3, takes a value in (0, 1], and $\delta_1 \leq \delta_2 \leq \delta_3$. The support of Heuristics B-3 and C-3 is shown through Var2 to Var19 as follows.

- The values show a consistent pattern in all 6 cases (100%) for Heuristic B-3 and in all 3 cases (100%) for Heuristic C-3, as stated in these heuristics.
- Under other heuristics, those values do not show a consistent pattern, indicating that a server of one server type serves a service request of the same type with a given QoS level of L, M or H at the corresponding probability of δ_1, δ_2 or δ_3, but each parameter does not follow $\delta_1 \leq \delta_2 \leq \delta_3$.
    - For Heuristic A-1, the values show the same pattern in 2 out of 3 problem cases (67%), which is smaller than 100% under Heuristics B-3 and C-3.

- All the other heuristics cannot be compared since they do not have mixed levels of service requests.

Table 19

Statistics for Step 2 Experimental Result

Compartment #	Heuristic	Var1	Var2	Var3	Var4	Var5	Var6	Var7	Var8	Var9	Var10	Var11	Var12	Var13	Var14	Var15	Var16	Var17	Var18	Var19
I-1. Rows 1-1 thru 2-6 Columns 1-1 thru 1-3	A-1	0.89	0.75	0.25	NA	0.73	0.27	NA	1.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.25	0.25	NA	0.45	0.45	NA	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(b)	0.47	0.41	0.59	NA	0.48	0.52	NA	0.46	0.54	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.10	0.10	NA	0.08	0.08	NA	0.09	0.09	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(b)	0.78	0.60	0.40	NA	0.68	0.32	NA	0.66	0.34	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.29	0.29	NA	0.16	0.16	NA	0.19	0.19	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-1	0.50	0.44	0.56	NA	0.44	0.56	NA	0.44	0.56	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.07	0.07	NA	0.07	0.07	NA	0.08	0.08	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
I-2. Rows 1-1 thru 2-6 Columns 2-1 thru 2-3	A-1	0.81	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.67	0.33	NA	0.57	0.43	NA	1.00	0.00	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.32	0.32	NA	0.26	0.26	NA	0.00	0.00	NA
	B-1(b)	0.47	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.46	0.54	NA	0.45	0.55	NA	0.46	0.54	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.08	0.08	NA	0.07	0.07	NA	0.07	0.07	NA
	B-1(b)	0.86	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.73	0.27	NA	0.63	0.37	NA	0.67	0.33	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.18	0.18	NA	0.24	0.24	NA	0.17	0.17	NA
	C-1	0.50	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.44	0.56	NA	0.44	0.56	NA	0.44	0.56	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.07	0.07	NA	0.07	0.07	NA	0.07	0.07	NA
I-3. Rows 1-1 thru 2-6 Columns 3-1 thru 3-3	A-1	0.86	0.65	0.35	NA	0.73	0.27	NA	1.00	0.00	NA	0.71	0.29	NA	0.78	0.22	NA	1.00	0.00	NA
			0.40	0.40	NA	0.38	0.38	NA	0.00	0.00	NA	0.43	0.43	NA	0.38	0.38	NA	0.00	0.00	NA
	B-1(b)	0.56	0.49	0.51	NA	0.44	0.56	NA	0.45	0.55	NA	0.45	0.55	NA	0.49	0.51	NA	0.54	0.46	NA
			0.18	0.18	NA	0.13	0.13	NA	0.27	0.27	NA	0.08	0.08	NA	0.24	0.24	NA	0.18	0.18	NA
	B-1(b)	0.78	0.65	0.35	NA	0.66	0.34	NA	0.74	0.26	NA	0.55	0.45	NA	0.64	0.36	NA	0.79	0.21	NA
			0.37	0.37	NA	0.33	0.33	NA	0.17	0.17	NA	0.36	0.36	NA	0.33	0.33	NA	0.21	0.21	NA
	C-1	0.53	0.49	0.51	NA	0.47	0.53	NA	0.45	0.55	NA	0.42	0.58	NA	0.39	0.61	NA	0.44	0.56	NA
			0.06	0.06	NA	0.04	0.04	NA	0.07	0.07	NA	0.11	0.11	NA	0.14	0.14	NA	0.07	0.07	NA
II-1. Rows 1-1 thru 2-6 Column 1-4	A-1	0.83	0.72	0.28	NA	0.75	0.25	NA	0.33	0.67	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.39	0.39	NA	0.45	0.45	NA	0.49	0.49	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(a)	0.58	0.50	0.50	NA	0.72	0.28	NA	0.04	0.96	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.29	0.29	NA	0.23	0.23	NA	0.14	0.14	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(a)	0.83	0.65	0.35	NA	0.86	0.14	NA	0.17	0.83	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.30	0.30	NA	0.22	0.22	NA	0.31	0.31	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-1	0.58	0.49	0.51	NA	0.39	0.61	NA	0.54	0.46	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.29	0.29	NA	0.37	0.37	NA	0.24	0.24	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
II-2. Rows 1-1 thru 2-6 Column 2-4	A-1	1.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.98	0.02	NA	0.77	0.23	NA	0.42	0.58	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.05	0.05	NA	0.39	0.39	NA	0.51	0.51	NA
	B-1(a)	0.42	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.52	0.48	NA	0.63	0.37	NA	0.13	0.88	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.32	0.32	NA	0.33	0.33	NA	0.31	0.31	NA
	B-1(a)	0.75	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.66	0.34	NA	0.84	0.16	NA	0.07	0.93	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.30	0.30	NA	0.31	0.31	NA	0.13	0.13	NA
	C-1	0.58	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.66	0.34	NA	0.42	0.58	NA	0.19	0.81	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.31	0.31	NA	0.32	0.32	NA	0.32	0.32	NA
II-3. Rows 1-1 thru 2-6 Column 3-4	A-1	0.83	0.75	0.25	NA	0.75	0.25	NA	0.50	0.50	NA	0.62	0.38	NA	0.75	0.25	NA	0.42	0.58	NA
			0.38	0.38	NA	0.45	0.45	NA	0.52	0.52	NA	0.47	0.47	NA	0.45	0.45	NA	0.51	0.51	NA
	B-1(a)	0.33	0.42	0.58	NA	0.71	0.29	NA	0.11	0.89	NA	0.47	0.53	NA	0.51	0.49	NA	0.39	0.61	NA
			0.29	0.29	NA	0.29	0.29	NA	0.30	0.30	NA	0.19	0.19	NA	0.29	0.29	NA	0.44	0.44	NA
	B-1(a)	0.50	0.32	0.68	NA	0.48	0.52	NA	0.25	0.75	NA	0.54	0.46	NA	0.54	0.46	NA	0.42	0.58	NA
			0.39	0.39	NA	0.51	0.51	NA	0.45	0.45	NA	0.46	0.46	NA	0.45	0.45	NA	0.51	0.51	NA
	C-1	0.33	0.34	0.66	NA	0.40	0.60	NA	0.67	0.33	NA	0.49	0.51	NA	0.46	0.54	NA	0.08	0.92	NA
			0.26	0.26	NA	0.32	0.32	NA	0.39	0.39	NA	0.36	0.36	NA	0.31	0.31	NA	0.20	0.20	NA
III-1. Rows 3-1 thru 3-9 Columns 1-1 thru 1-3	A-1	1.00	0.92	0.08	NA	1.00	0.00	NA	1.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.10	0.10	NA	0.00	0.00	NA	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-2(b)	1.00	0.70	0.30	NA	0.73	0.27	NA	0.72	0.28	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.11	0.11	NA	0.08	0.08	NA	0.10	0.10	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-2(b)	1.00	0.88	0.12	NA	0.81	0.19	NA	0.87	0.13	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.07	0.07	NA	0.10	0.10	NA	0.04	0.04	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-2	1.00	0.69	0.31	NA	0.69	0.31	NA	0.70	0.30	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.09	0.09	NA	0.09	0.09	NA	0.10	0.10	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

III-2. Rows 3-1 thru 3-9 Columns 2-1 thru 2-3	A-1	0.63	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.23	0.77	NA	0.54	0.46	NA	1.00	0.00	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.35	0.35	NA	0.19	0.19	NA	0.00	0.00	NA
	B-2(b)	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.31	0.69	NA	0.28	0.72	NA	0.29	0.71	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.09	0.09	NA	0.10	0.10	NA	0.09	0.09	NA
	B-2(b)	0.26	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.39	0.61	NA	0.34	0.66	NA	0.37	0.63	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.16	0.16	NA	0.20	0.20	NA	0.15	0.15	NA
	C-2	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.29	0.71	NA	0.28	0.72	NA	0.29	0.71	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.09	0.09	NA	0.09	0.09	NA	0.09	0.09	NA
IV. Rows 3-1 thru 3-9 Columns 3-1 thru 3-3	A-1	0.85	0.59	0.41	NA	0.89	0.11	NA	1.00	0.00	NA	0.56	0.44	NA	0.92	0.08	NA	1.00	0.00	NA
			0.46	0.46	NA	0.33	0.33	NA	0.00	0.00	NA	0.44	0.44	NA	0.25	0.25	NA	0.00	0.00	NA
	B-2(a)	0.48	0.73	0.27	NA	0.75	0.25	NA	0.75	0.25	NA	0.28	0.72	NA	0.28	0.72	NA	0.28	0.72	NA
			0.10	0.10	NA	0.09	0.09	NA	0.09	0.09	NA	0.08	0.08	NA	0.10	0.10	NA	0.09	0.09	NA
	B-1(a)	0.48	0.47	0.53	NA	0.50	0.50	NA	1.00	0.00	NA	0.46	0.54	NA	0.52	0.48	NA	0.00	1.00	NA
			0.30	0.30	NA	0.21	0.21	NA	0.00	0.00	NA	0.33	0.33	NA	0.41	0.41	NA	0.00	0.00	NA
	C-2	0.48	0.76	0.24	NA	0.75	0.25	NA	0.78	0.22	NA	0.27	0.73	NA	0.26	0.74	NA	0.27	0.73	NA
			0.07	0.07	NA	0.09	0.09	NA	0.09	0.09	NA	0.09	0.09	NA	0.08	0.08	NA	0.07	0.07	NA
V-1. Rows 3-1 thru 3-9 Column 1-4	A-1	1.00	1.00	0.00	NA	1.00	0.00	NA	1.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.00	0.00	NA	0.00	0.00	NA	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-3	1.00	0.68	0.32	NA	0.69	0.31	NA	0.72	0.28	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.18	0.18	NA	0.30	0.30	NA	0.16	0.16	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-3	1.00	0.87	0.13	NA	0.94	0.06	NA	0.96	0.04	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.06	0.06	NA	0.13	0.13	NA	0.11	0.11	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-3	0.89	0.58	0.42	NA	0.66	0.34	NA	0.80	0.20	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.27	0.27	NA	0.35	0.35	NA	0.19	0.19	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
V-2. Rows 3-1 thru 3-9 Column 2-4	A-1	0.44	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.37	0.63	NA	0.44	0.56	NA	0.00	1.00	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.48	0.48	NA	0.53	0.53	NA	0.00	0.00	NA
	B-3	0.11	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.45	0.55	NA	0.43	0.57	NA	0.00	1.00	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.14	0.14	NA	0.16	0.16	NA	0.00	0.00	NA
	B-3	0.33	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.54	0.46	NA	0.41	0.59	NA	0.00	1.00	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.34	0.34	NA	0.28	0.28	NA	0.00	0.00	NA
	C-3	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.43	0.57	NA	0.24	0.76	NA	0.15	0.85	NA
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.25	0.25	NA	0.25	0.25	NA	0.23	0.23	NA
V-3. Rows 3-1 thru 3-9 Column 3-4	A-1	1.00	1.00	0.00	NA	1.00	0.00	NA	0.78	0.22	NA	0.83	0.17	NA	0.83	0.17	NA	0.78	0.22	NA
			0.00	0.00	NA	0.00	0.00	NA	0.44	0.44	NA	0.41	0.41	NA	0.41	0.41	NA	0.44	0.44	NA
	B-3	0.44	0.56	0.44	NA	0.81	0.19	NA	0.85	0.15	NA	0.41	0.59	NA	0.25	0.75	NA	0.17	0.83	NA
			0.32	0.32	NA	0.23	0.23	NA	0.34	0.34	NA	0.24	0.24	NA	0.19	0.19	NA	0.35	0.35	NA
	B-3	0.78	0.56	0.44	NA	0.72	0.28	NA	0.89	0.11	NA	0.71	0.29	NA	0.59	0.41	NA	0.12	0.88	NA
			0.53	0.53	NA	0.44	0.44	NA	0.22	0.22	NA	0.45	0.45	NA	0.49	0.49	NA	0.26	0.26	NA
	C-3	0.44	0.68	0.32	NA	0.71	0.29	NA	0.83	0.17	NA	0.35	0.65	NA	0.27	0.73	NA	0.25	0.75	NA
			0.20	0.20	NA	0.34	0.34	NA	0.25	0.25	NA	0.21	0.21	NA	0.17	0.17	NA	0.22	0.22	NA

The statistics in Table 19 are summarized below. Heuristics A-1 and B-1(b) state that a server as the dominant server (i.e., the server with $i = 1$) serves a service request with higher probability than the other server(s). The support of Heuristics A-1 and B-1(b) is shown through Var1 as follows.

- Var1 values are greater than 0.50 in 11 out of 12 problem cases and less than 0.50 in 1 out of 12 cases for Heuristic A-1. Similarly, Var1 values are greater than 0.50 in 4 out of 6 problem cases and less than 0.50 in 2 out of 6 cases for Heuristic B-1(b). In other words, Var1 values are greater than 0.50 in 92% and 67% of all problem cases under Heuristics A-1 and B-1(b), indicating that the first server as

the dominant server serves at least same or higher number of service requests than the other server(s) in many problem cases.

- Under other heuristics, Var1 values do not show a consistent pattern indicating no dominant server for service provision.
  - For Heuristics B-1(a) and C-1, Var1 values are greater than 0.50 in 3 out of 7 problem cases (43%) and in 3 out of 6 problem cases (50%), respectively, which are smaller than 92% and 67% under Heuristics A-1 and B-1(b).
  - For Heuristics B-2(a) and C-2, Var1 values are greater than 0.50 in 0 out of 1 problem case (0%) and in 1 out of 3 problem cases (33%), respectively, which are smaller than 92% and 67% under Heuristics A-1 and B-1(b).
  - For Heuristic B-2(b), Var1 values are greater than 0.50 in 2 out of 4 problem cases (50%), which is smaller than 92% and 67% under Heuristics A-1 and B-1(b).
  - For Heuristics B-3 and C-3, Var1 values are greater than 0.50 in 3 out of 6 problem cases (50%) and in 1 out of 3 problem cases (33%), respectively, which are smaller than 92% and 67% under Heuristics A-1 and B-1(b).

Heuristics B-1(a) and C-1 state that a server randomly serves a service request. The support of Heuristics B-1(a) and C-1 is shown through Var1 to Var19 as follows.

- For Heuristic B-1(a), Var1 values are greater than 0.50 in 3 out of 7 problem cases, equal to 0.50 in 1 out of 7 cases and less than 0.50 in 3 out of 7 cases. In addition, Var2 through Var19 values show that the first server is selected with higher probability than the second server to serve more number of service requests in 3 out of 7 problem cases, with same probability as the second server in

1 out of 7 cases and with lower probability than the second server in 3 out of 7 cases. Those statistics support the random selection of a server for service provision under Heuristic B-1(a). For Heuristic C-1, Var1 values are greater than 0.50 in 3 out of 6 problem cases, equal to 0.50 in 2 out of 6 cases and less than 0.50 in 1 out of 6 cases. Var2 through Var19 values show that the first server is selected with higher probability than the second server to serve more number of service requests in all 6 problem cases. Since the problem cases under Heuristic C-1 have insufficient resource capacity to satisfy all service requests and have the second server with at least same resource capacity as the first server, the random server selection for service provision would result in selecting the second server with higher probability than the first server to serve more number of service requests due to its larger resource capacity. Therefore, Var1 through Var19 values support the random selection of a server for service provision under Heuristic C-1.

- Under other heuristics, those values do not show a consistent pattern, indicating there exists a dominant server for service provision.
    - For Heuristics A-1 and B-1(b), Var1 values are greater than 0.50 in 11 out of 12 cases (83%) and in 4 out of 6 cases (67%), respectively, which are greater than 43% and 50% under Heuristics B-1(a) and C-1. Similarly, Var2 through Var19 values show that the first server is selected with higher probability than the second server in 11 out of 12 cases (83%) and in 3 out of 6 cases (50%) under Heuristics A-1 and B-1(b), respectively, which are greater than 43% under Heuristics B-1(a).

- Problem cases in Heuristics B-2(a), B-2(b), B-3, C-2 and C-3 involve both types of servers. Since their solutions are significantly affected by both server types and service types, they cannot be compared to the problem cases involving with one server type under Heuristics B-1(a) and C-1.

Heuristics B-2(a) and C-2 state that a server of one server type randomly serves a service request of the same type with higher probability than the other server(s) of a different type. The support of Heuristics B-2(a) and C-2 is shown through Var2 to Var19 as follows.

- The values show a consistent pattern in all 1 problem case (100%) for Heuristic B-2(a) and in all 3 cases (100%) for Heuristic C-2, as stated in these heuristics. Note that Heuristics B-2(a) and B-2(b) are same with two servers as in Step 1 and Step 2 experiments.
- Under other heuristics, those values do not show a consistent pattern, indicating that a server serves a service request regardless of its type.
  - For Heuristic A-1, the values show the same pattern in 3 out of 6 problem cases (50%), which is smaller than 100% under Heuristics B-2(a) and C-2.
  - For Heuristic B-1(a), each server is almost equally likely to be selected for service provision regardless of a service type in one problem case.
  - For Heuristics B-3 and C-3, the values show the same pattern in 4 out of 6 cases (67%) and 3 out of 3 cases (100%). However, problem cases under those heuristics involve both types of servers and service requests with mixed QoS levels. Since their solutions are significantly affected by serving different QoS levels of service requests as well as matching server types and service

types, they cannot be compared to the problem cases involving with service requests of one QoS level under Heuristics B-2(a) and C-2. Similarly, Heuristics B-1(b) and C-1 cannot be compared since they do not have both server types.

Heuristic B-2(b) states that a server of each server type as the dominant one serves a service request of the same type with higher probability than the dominant server of a different type. The support of Heuristic B-2(b) is shown through Var2 to Var19 as follows.

- The values show a consistent pattern in all 4 problem cases (100%), as stated in Heuristic B-2(b). Note that Heuristics B-2(a) and B-2(b) are same with two servers as in Step 1 and Step 2 experiments. Thus, Heuristic C-2 also shows this consistent pattern as Heuristic B-2(b) in Step 1 and Step 2 experiments.
- Under other heuristics, those values do not show a consistent pattern indicating that a server serves a service request regardless of its type.
    - For Heuristic A-1, the values show the same pattern in 3 out of 6 problem cases (50%), which is smaller than 100% under Heuristic B-2(b).
    - For Heuristic B-1(a), each server is almost equally likely to be selected for service provision regardless of a service type in one problem case.
    - For Heuristics B-3 and C-3, the values show the same pattern in 4 out of 6 cases (67%) and 3 out of 3 cases (100%). However, problem cases under those heuristics involve both types of servers and service requests with mixed QoS levels. Since their solutions are significantly affected by serving different QoS levels of service requests as well as matching server types and service

types, they cannot be compared to the problem cases involving with service requests of one QoS level under Heuristic B-2(b). Similarly, Heuristics B-1(b) and C-1 cannot be compared since they do not have both server types.

Heuristics B-3 and C-3 state that a server of one server type randomly serves a service request of the same type with a given QoS level of L, M or H at the corresponding probability of δ_1, δ_2 or δ_3, respectively, and a server of a different server type to serve the service request with the given QoS level of L, M or H with the probability of $(1-\delta_1)$, $(1-\delta_2)$ or $(1-\delta_3)$, respectively. Each parameter, δ_1, δ_2 or δ_3, takes a value in (0, 1], and $\delta_1 \leq \delta_2 \leq \delta_3$. The support of Heuristics B-3 and C-3 is shown through Var2 to Var19 as follows.

- The values show a consistent pattern in all 6 cases (100%) for Heuristic B-3 and in all 3 cases (100%) for Heuristic C-3, as stated in these heuristics.
- Under other heuristics, those values do not show a consistent pattern, indicating that a server of one server type serves a service request of the same type with a given QoS level of L, M or H at the corresponding probability of δ_1, δ_2 or δ_3, but each parameter does not follow $\delta_1 \leq \delta_2 \leq \delta_3$.
    - For Heuristic A-1, the values show the same pattern in 1 out of 3 problem cases (33%), which is much smaller than 100% under Heuristics B-3 and C-3.
    - All the other heuristics cannot be compared since they do not have mixed levels of service requests.

Table 20

Statistics for Step 3 Experimental Result

Compartment #	Heuristic	Var1	Var2	Var3	Var4	Var5	Var6	Var7	Var8	Var9	Var10	Var11	Var12	Var13	Var14	Var15	Var16	Var17	Var18	Var19
I-1. Rows 1-1 thru 2-6 Columns 1-1 thru 1-3	A-1	0.75	0.45	0.24	0.31	0.74	0.14	0.12	1.00	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.32	0.21	0.19	0.37	0.31	0.28	0.00	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(b)	0.56	0.32	0.33	0.34	0.34	0.37	0.29	0.33	0.35	0.32	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.05	0.04	0.05	0.12	0.03	0.10	0.02	0.02	0.02	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(b)	0.83	0.38	0.27	0.35	0.60	0.28	0.12	0.56	0.16	0.28	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.25	0.15	0.17	0.17	0.17	0.14	0.09	0.20	0.24	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-1	0.58	0.32	0.34	0.34	0.30	0.35	0.35	0.33	0.34	0.34	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.02	0.02	0.01	0.06	0.07	0.04	0.02	0.01	0.01	NA	NA	NA	NA	NA	NA	NA	NA	NA
I-2. Rows 1-1 thru 2-6 Columns 2-1 thru 2-3	A-1	0.69	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.42	0.35	0.22	0.48	0.23	0.30	1.00	0.00	0.00
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.36	0.25	0.25	0.38	0.39	0.37	0.00	0.00	0.00
	B-1(b)	0.64	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.34	0.37	0.30	0.31	0.35	0.34	0.32	0.36	0.32
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.03	0.05	0.04	0.05	0.03	0.03	0.06	0.03	0.05
	B-1(b)	1.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.64	0.31	0.05	0.63	0.22	0.15	0.62	0.13	0.25
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.09	0.08	0.05	0.18	0.15	0.17	0.18	0.13	0.19
	C-1	0.61	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.31	0.35	0.34	0.30	0.35	0.35	0.30	0.35	0.35
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.03	0.02	0.02	0.05	0.03	0.03	0.05	0.03	0.03
I-3. Rows 1-1 thru 2-6 Columns 3-1 thru 3-3	A-1	0.61	0.38	0.35	0.27	0.44	0.17	0.39	1.00	0.00	0.00	0.25	0.31	0.44	0.49	0.17	0.35	1.00	0.00	0.00
			0.13	0.09	0.12	0.47	0.39	0.46	0.00	0.00	0.00	0.18	0.17	0.24	0.45	0.39	0.42	0.00	0.00	0.00
	B-1(b)	0.44	0.32	0.34	0.34	0.32	0.35	0.33	0.33	0.35	0.32	0.31	0.35	0.33	0.31	0.34	0.35	0.34	0.35	0.31
			0.04	0.04	0.04	0.06	0.04	0.05	0.02	0.02	0.02	0.03	0.04	0.03	0.06	0.04	0.04	0.01	0.03	0.03
	B-1(b)	0.83	0.41	0.25	0.34	0.62	0.17	0.21	0.80	0.19	0.01	0.31	0.36	0.33	0.54	0.26	0.20	0.88	0.08	0.04
			0.20	0.14	0.16	0.34	0.24	0.29	0.25	0.24	0.05	0.23	0.17	0.15	0.35	0.30	0.31	0.13	0.10	0.08
	C-1	0.53	0.31	0.35	0.34	0.30	0.35	0.35	0.33	0.34	0.34	0.31	0.35	0.35	0.30	0.35	0.35	0.33	0.35	0.33
			0.03	0.01	0.02	0.05	0.03	0.03	0.02	0.01	0.01	0.03	0.02	0.03	0.05	0.03	0.03	0.02	0.03	0.02
II-1. Rows 1-1 thru 2-6 Column 1-4	A-1	0.83	0.71	0.00	0.29	0.42	0.00	0.58	0.58	0.42	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.40	0.00	0.40	0.49	0.00	0.49	0.51	0.51	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(a)	0.25	0.32	0.25	0.42	0.35	0.41	0.24	0.23	0.35	0.42	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.25	0.17	0.22	0.29	0.33	0.22	0.30	0.36	0.39	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-1(a)	0.58	0.31	0.35	0.34	0.38	0.33	0.29	0.57	0.21	0.22	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.25	0.26	0.40	0.49	0.37	0.39	0.43	0.28	0.30	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-1	0.42	0.32	0.40	0.28	0.51	0.33	0.16	0.25	0.34	0.41	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.10	0.19	0.20	0.36	0.32	0.15	0.13	0.11	0.08	NA	NA	NA	NA	NA	NA	NA	NA	NA
II-2. Rows 1-1 thru 2-6 Column 2-4	A-1	0.67	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.63	0.21	0.17	0.14	0.36	0.50	0.25	0.46	0.29
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.48	0.40	0.33	0.24	0.48	0.50	0.45	0.50	0.45
	B-1(a)	0.42	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.42	0.32	0.26	0.37	0.37	0.25	0.13	0.29	0.58
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.19	0.14	0.15	0.17	0.21	0.20	0.16	0.30	0.34
	B-1(a)	0.75	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.51	0.18	0.31	0.27	0.50	0.23	0.35	0.29	0.36
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.36	0.29	0.33	0.37	0.48	0.34	0.40	0.40	0.43
	C-1	0.33	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.29	0.33	0.38	0.21	0.53	0.26	0.35	0.27	0.39
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.20	0.24	0.28	0.24	0.30	0.21	0.15	0.17	0.17
II-3. Rows 1-1 thru 2-6 Column 3-4	A-1	0.58	0.50	0.13	0.38	0.50	0.06	0.44	0.48	0.22	0.30	0.51	0.28	0.21	0.33	0.29	0.38	0.33	0.44	0.22
			0.52	0.31	0.48	0.50	0.17	0.53	0.44	0.36	0.45	0.43	0.39	0.40	0.47	0.49	0.49	0.50	0.53	0.44
	B-1(a)	0.42	0.26	0.33	0.40	0.48	0.40	0.12	0.10	0.37	0.53	0.41	0.33	0.26	0.52	0.31	0.16	0.08	0.35	0.58
			0.22	0.20	0.27	0.26	0.28	0.19	0.16	0.37	0.42	0.31	0.22	0.21	0.26	0.31	0.21	0.14	0.42	0.40
	B-1(a)	0.75	0.65	0.18	0.17	0.39	0.17	0.44	0.29	0.47	0.24	0.52	0.29	0.20	0.33	0.50	0.17	0.36	0.45	0.18
			0.29	0.20	0.19	0.39	0.28	0.46	0.31	0.43	0.34	0.32	0.33	0.21	0.38	0.42	0.22	0.39	0.47	0.34
	C-1	0.25	0.28	0.48	0.24	0.24	0.38	0.37	0.41	0.24	0.35	0.38	0.28	0.34	0.23	0.35	0.42	0.32	0.42	0.27
			0.22	0.24	0.20	0.23	0.25	0.29	0.27	0.18	0.18	0.20	0.19	0.20	0.19	0.27	0.26	0.20	0.27	0.20
III-1. Rows 3-1 thru 3-9 Columns 1-1 thru 1-3	A-1	1.00	1.00	0.00	0.00	1.00	0.00	0.00	1.00	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-2(b)	1.00	0.59	0.20	0.21	0.71	0.14	0.15	0.68	0.16	0.16	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.03	0.02	0.02	0.09	0.07	0.08	0.02	0.01	0.01	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-2(b)	1.00	0.48	0.25	0.27	0.84	0.06	0.10	0.81	0.13	0.06	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.08	0.07	0.04	0.04	0.09	0.08	0.02	0.10	0.08	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-2	1.00	0.62	0.19	0.19	0.66	0.16	0.18	0.68	0.16	0.16	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.03	0.01	0.02	0.07	0.04	0.04	0.02	0.01	0.01	NA	NA	NA	NA	NA	NA	NA	NA	NA

III-2. Rows 3-1 thru 3-9 Columns 2-1 thru 2-3	A-1	0.59	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.06	0.89	0.05	0.50	0.33	0.17	1.00	0.00	0.00
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.19	0.23	0.07	0.43	0.50	0.25	0.00	0.00	0.00
	B-2(b)	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.16	0.45	0.39	0.13	0.44	0.43	0.14	0.45	0.41
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.03	0.03	0.01	0.04	0.02	0.02	0.05	0.03	0.04
	B-2(b)	0.30	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.32	0.39	0.29	0.25	0.30	0.46	0.25	0.40	0.35
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.07	0.14	0.11	0.08	0.14	0.16	0.10	0.22	0.25
	C-2	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.15	0.43	0.43	0.13	0.44	0.44	0.13	0.43	0.43
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.03	0.01	0.01	0.04	0.02	0.02	0.04	0.02	0.02
IV. Rows 3-1 thru 3-9 Columns 3-1 thru 3-3	A-1	0.81	0.47	0.26	0.26	0.44	0.11	0.44	1.00	0.00	0.00	0.51	0.23	0.26	0.61	0.11	0.28	1.00	0.00	0.00
			0.34	0.23	0.28	0.53	0.33	0.53	0.00	0.00	0.00	0.41	0.32	0.37	0.49	0.33	0.44	0.00	0.00	0.00
	B-2(a)	0.67	0.62	0.19	0.19	0.68	0.16	0.16	0.68	0.16	0.16	0.14	0.40	0.46	0.11	0.45	0.44	0.11	0.46	0.42
			0.06	0.03	0.03	0.09	0.04	0.04	0.02	0.01	0.01	0.03	0.04	0.03	0.04	0.02	0.02	0.00	0.04	0.04
	B-1(a)	0.70	0.31	0.28	0.40	0.70	0.19	0.11	1.00	0.00	0.00	0.36	0.24	0.41	0.61	0.13	0.26	0.00	1.00	0.00
			0.15	0.17	0.22	0.30	0.23	0.24	0.00	0.00	0.00	0.13	0.16	0.22	0.08	0.20	0.21	0.00	0.00	0.00
	C-2	0.70	0.65	0.18	0.18	0.69	0.15	0.15	0.71	0.12	0.17	0.13	0.43	0.44	0.10	0.45	0.45	0.10	0.45	0.45
			0.05	0.02	0.02	0.08	0.04	0.04	0.06	0.07	0.02	0.02	0.01	0.01	0.04	0.02	0.02	0.01	0.01	0.01
V-1. Rows 3-1 thru 3-9 Column 1-4	A-1	1.00	1.00	0.00	0.00	1.00	0.00	0.00	1.00	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-3	1.00	0.50	0.19	0.31	0.63	0.12	0.24	0.76	0.20	0.04	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.17	0.17	0.19	0.25	0.15	0.16	0.29	0.25	0.11	NA	NA	NA	NA	NA	NA	NA	NA	NA
	B-3	0.89	0.39	0.16	0.45	0.56	0.30	0.15	0.65	0.17	0.19	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.30	0.24	0.23	0.31	0.29	0.34	0.23	0.20	0.23	NA	NA	NA	NA	NA	NA	NA	NA	NA
	C-3	1.00	0.44	0.18	0.38	0.62	0.21	0.17	0.68	0.13	0.19	NA	NA	NA	NA	NA	NA	NA	NA	NA
			0.21	0.19	0.09	0.35	0.25	0.33	0.16	0.13	0.17	NA	NA	NA	NA	NA	NA	NA	NA	NA
V-2. Rows 3-1 thru 3-9 Column 2-4	A-1	0.78	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.44	0.22	0.33	1.00	0.00	0.00	0.00	1.00	0.00
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.53	0.44	0.50	0.00	0.00	0.00	0.00	0.00	0.00
	B-3	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.19	0.41	0.40	0.17	0.52	0.31	0.09	0.35	0.56
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.17	0.23	0.22	0.12	0.19	0.19	0.14	0.18	0.20
	B-3	0.11	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.28	0.26	0.46	0.43	0.28	0.30	0.07	0.39	0.54
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.26	0.16	0.24	0.31	0.22	0.23	0.15	0.24	0.26
	C-3	0.00	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.47	0.29	0.24	0.13	0.39	0.48	0.10	0.50	0.40
			NA	NA	NA	NA	NA	NA	NA	NA	NA	0.24	0.25	0.16	0.13	0.32	0.31	0.10	0.23	0.22
V-3. Rows 3-1 thru 3-9 Column 3-4	A-1	0.67	0.78	0.00	0.22	0.33	0.33	0.33	0.83	0.00	0.17	0.39	0.17	0.44	0.00	1.00	0.00	0.67	0.17	0.17
			0.44	0.00	0.44	0.58	0.58	0.58	0.41	0.00	0.41	0.49	0.35	0.53	0.00	0.00	0.00	0.52	0.41	0.41
	B-3	0.33	0.40	0.27	0.33	0.62	0.16	0.22	0.85	0.15	0.00	0.18	0.42	0.40	0.16	0.39	0.46	0.06	0.50	0.44
			0.24	0.17	0.23	0.15	0.15	0.13	0.23	0.23	0.00	0.15	0.16	0.19	0.13	0.28	0.27	0.13	0.28	0.28
	B-3	0.44	0.25	0.06	0.69	0.33	0.33	0.33	0.56	0.22	0.22	0.39	0.17	0.44	0.33	0.50	0.17	0.00	0.67	0.33
			0.46	0.18	0.46	0.52	0.52	0.52	0.53	0.44	0.44	0.42	0.35	0.46	0.52	0.55	0.41	0.00	0.50	0.50
	C-3	0.44	0.47	0.37	0.16	0.65	0.16	0.18	0.85	0.06	0.10	0.20	0.40	0.40	0.17	0.34	0.49	0.10	0.54	0.36
			0.26	0.24	0.17	0.23	0.14	0.15	0.22	0.11	0.14	0.21	0.36	0.36	0.18	0.20	0.25	0.11	0.24	0.27

The statistics in Table 20 are summarized below. Heuristics A-1 and B-1(b) state that a server as the dominant server (i.e., the server with $i = 1$) serves a service request with higher probability than the other servers. The support of Heuristics A-1 and B-1(b) is shown through Var1 as follows.

- Var1 values are greater than 0.50 in all 12 problem cases for Heuristic A-1. Similarly, Var1 values are greater than 0.50 in 5 out of 6 problem cases and less than 0.50 in 1 out of 6 cases for Heuristic B-1(b). In other words, Var1 values are greater than 0.50 in 100% and 83% of all problem cases under Heuristics A-1 and

B-1(b), indicating that the first server as the dominant server serves at least same or higher number of service requests than the other servers in most problem cases.

- Under other heuristics, Var1 values do not show a consistent pattern indicating no dominant server for service provision.
  - For Heuristics B-1(a) and C-1, Var1 values are greater than 0.50 in 4 out of 7 problem cases (57%) and in 3 out of 6 problem cases (50%), respectively, which are smaller than 100% and 83% under Heuristics A-1 and B-1(b).
  - For Heuristic B-2(a), Var1 value is greater than 0.50 in one problem case (100%). However, this problem case involves both server types. Since its solutions are significantly affected by both server types and service types, it cannot be compared to the problem cases involving with one server type under Heuristics A-1 and B-1(b). For Heuristic C-2, Var1 values are greater than 0.50 in 2 out of 3 problem cases (67%), which is smaller than 100% and 83% under Heuristics A-1 and B-1(b).
  - For Heuristic B-2(b), Var1 values are greater than 0.50 in 2 out of 4 problem cases (50%), which is smaller than 100% and 83% under Heuristics A-1 and B-1(b).
  - For Heuristics B-3 and C-3, Var1 values are greater than 0.50 in 2 out of 6 problem cases (33%) and in 1 out of 3 problem cases (33%), respectively, which are smaller than 100% and 83% under Heuristics A-1 and B-1(b).

Heuristics B-1(a) and C-1 state that a server randomly serves a service request. The support of Heuristics B-1(a) and C-1 is shown through Var1 to Var19 as follows.

- For Heuristic B-1(a), Var1 values are greater than 0.50 in 4 out of 7 problem cases and less than 0.50 in 3 out of 7 cases. In addition, Var2 through Var19 values show that each server is selected for service provision with almost same probability in all 7 problem cases. Those statistics support the random selection of a server for service provision under Heuristic B-1(a). For Heuristic C-1, Var1 values are greater than 0.50 in 3 out of 6 problem cases and less than 0.50 in 3 out of 6 cases. Var2 through Var19 values show that the servers with larger resource capacity are selected with higher probability than the other server(s) to serve more number of service requests in 4 out of 6 problem cases, and each server is selected with almost same probability for service provision in 2 out of 6 cases. Since the problem cases under Heuristic C-1 have insufficient resource capacity to satisfy all service requests and have some servers with at least same resource capacity as the other servers, the random server selection for service provision would result in selecting the servers with larger resource capacity with higher probability than the other(s) to serve more number of service requests due to their larger resource capacity. Therefore, Var1 through Var19 values support the random selection of a server for service provision under Heuristic C-1.
- Under other heuristics, those values do not show a consistent pattern, indicating there exists a dominant server for service provision.
    - For Heuristics A-1 and B-1(b), Var1 values are greater than 0.50 in all 12 cases (100%) and in 5 out of 6 cases (83%), respectively, which are greater than 57% and 50% under Heuristics B-1(a) and C-1. Similarly, Var2 through Var19 values show that the first server as the dominant server is selected with

higher probability than the other servers for service provision in 11 out of 12 cases (83%) and 3 out of 6 cases (50%) under Heuristics A-1 and B-1(b), respectively, whereas there is no dominant server for service provision under Heuristics B-1(a) and C-1.

- Problem cases in Heuristics B-2(a), B-2(b), B-3, C-2 and C-3 involve both types of servers. Since their solutions are significantly affected by both server types and service types, they cannot be compared to the problem cases involving with one server type under Heuristics B-1(a) and C-1.

Heuristics B-2(a) and C-2 state that a server of one server type randomly serves a service request of the same type with higher probability than the other server(s) of a different type. The support of Heuristics B-2(a) and C-2 is shown through Var2 to Var19 as follows.

- The values show a consistent pattern in all 1 problem case (100%) for Heuristic B-2(a) and in all 3 cases (100%) for Heuristic C-2, as stated in these heuristics.
- Under other heuristics, those values do not show a consistent pattern, indicating that a server serves a service request regardless of its type or by different patterns.
  - For Heuristic A-1, the values show the same pattern in 2 out of 6 problem cases (33%), which is much smaller than 100% under Heuristics B-2(a) and C-2.
  - For Heuristic B-1(a), each server is equally likely to be selected for service provision regardless of a service type in one problem case.
  - For Heuristic B-2(b), the values show the same pattern in 3 out of 4 problem cases (75%), which is smaller than 100% under Heuristics B-2(a) and C-2.

- For Heuristics B-3 and C-3, the values show the same pattern in 3 out of 6 cases (50%) and 2 out of 3 cases (67%), which are smaller than 100% under Heuristics B-2(a) and C-2.
- Heuristics B-1(b) and C-1 cannot be compared since they do not have both server types.

Heuristic B-2(b) states that a server of each server type as the dominant one serves a service request of the same type with higher probability than the dominant server of a different type. The support of Heuristic B-2(b) is shown through Var2 to Var19 as follows.

- The values show a consistent pattern in 3 out of 4 cases (75%), as stated in Heuristic B-2(b).
- Under other heuristics, those values do not show a consistent pattern, indicating that a server serves a service request regardless of its type or by different patterns.
  - For Heuristic A-1, the values show the same pattern in 2 out of 6 problem cases (33%), which is much smaller than 75% under Heuristic B-2(b).
  - For Heuristic B-1(a), each server is equally likely to be selected for service provision regardless of a service type in one problem case.
  - For Heuristics B-2(a) and C-2, the values show the same pattern in 0 out of 1 problem case (0%) and in 2 out of 3 problem cases (67%), respectively, which are smaller than 75% under Heuristic B-2(b).
  - For Heuristics B-3 and C-3, the values show the same pattern in 1 out of 6 cases (17%) and 1 out of 3 cases (33%), respectively, which are smaller than 75% under Heuristic B-2(b).

- Heuristics B-1(b) and C-1 cannot be compared since they do not have both server types.

Heuristics B-3 and C-3 state that a server of one server type randomly serves a service request of the same type with a given QoS level of L, M or H at the corresponding probability of δ_1, δ_2 or δ_3, respectively, and a server of a different server type to serve the service request with the given QoS level of L, M or H with the probability of $(1-\delta_1)$, $(1-\delta_2)$ or $(1-\delta_3)$, respectively. Each parameter, δ_1, δ_2 or δ_3, takes a value in (0, 1], and $\delta_1 \leq \delta_2 \leq \delta_3$. The support of Heuristics B-3 and C-3 is shown through Var2 to Var19 as follows.

- The values show a consistent pattern in 5 out of 6 cases (83%) for Heuristic B-3 and in all 3 cases (100%) for Heuristic C-3, as stated in these heuristics.
- Under other heuristics, those values do not show a consistent pattern, indicating that a server of one server type serves a service request of the same type with a given QoS level of L, M or H at the corresponding probability of δ_1, δ_2 or δ_3, but each parameter does not follow $\delta_1 \leq \delta_2 \leq \delta_3$.
  - For Heuristic A-1, the values show the same pattern in 1 out of 3 problem cases (33%), which is much smaller than 83% and 100% under Heuristics B-3 and C-3.
  - All the other heuristics cannot be compared since they do not have mixed levels of service requests.

2.5.3 The systematic flow of the heuristic algorithm. In this section, a systematic flow of resource allocation heuristics is described as follows. In this heuristic algorithm, all service requests of all clients are randomly ordered first, and then each

problem case is determined to be one of three cases: Case A, Case B or Case C depending on different conditions of resource capacity with information including QoS requirements for all service requests of all clients and resource availability and workload of all servers.

For a problem case determined as Case A, client *k*'s service request is assigned to a server following Heuristic A-1 until all service requests of all clients are successfully assigned. For a problem case determined as Case B, it first checks one condition for Heuristic B-2(a) if overall resource requirement for all the service requests takes at least 70% of the total available resource capacity over all servers. For each client *k*'s service request, all the related heuristic Bs depending on its conditions are applied in obtaining its resource allocation solutions until all service requests of all clients are successfully assigned. For a problem case determined as Case C, client *k*'s service request is assigned to a server by applying all the related heuristic Cs depending on its conditions. Since all service requests of all clients cannot be served due to limited overall resource capacity, experimental results are averaged for 100 runs. Note that all service requests of all clients are randomly ordered at each iteration for the problem case in Case C.

In order to obtain efficient resource allocation solutions using the proposed heuristics, the objective function and the constraints expressed as Equations (1) through (8) in Chapter 2.3 are considered as follows:

- Equation (1) as the objective function needs to be considered in such ways that all servers try to serve as many service requests as they can and simultaneously they seek for obtaining the best objective value by providing QoS closer to QoS requirement. It could be achieved by applying the resource allocation heuristics.

- Equation (2) of guaranteeing one server assignment for a service request is satisfied since a client's service request is assigned to one server at most.
- Equation (3) of requiring the selected server's service provision for a service request is not considered since all servers provide both service types of the communication intensive service and the computation intensive service.
- Equations (4) through (8) are especially considered in selecting a server for a client's service request.
    - Equation (8) ensures satisfaction of the QoS requirement for the client's service request when assigning a level of service parameter for the service request. Note that Equation (8) is implemented as $Q_{kp_s i} \geq X_{ki} Q^{l}_{kp_s}$ for both service types in this study.
    - Equation (4) makes sure if the assigned level of service parameter for the service request does not exceed the maximum level of the service parameter.
    - The assigned level of service parameter for the service request can determine the requirement of resource amount, and the resource amount can determine the QoS value as shown in Equations (5) and (6) as resource and QoS impact models.
    - Equation (7) checks if the required resource amount is acceptable with resource capacity limits of the selected server.

Note that, as same as a centralized algorithm, this heuristic algorithm has a central authority to collect important information of all service requests' QoS requirements and resource status of all service providers. Using the updated information at each decision epoch, the heuristic algorithm generates resource allocation solutions. However, this

heuristic algorithm is different from the centralized algorithm such that it does not need to search for all the solution space to produce optimal resource allocation decisions. Instead, the proposed heuristics directly guide the solution path of how to allocate resources of which server to which service request, resulting in a much smaller solution space than the solution space of the centralized algorithm.

2.6 Description of Extended Problem Cases for Performance Comparison

In this section, various problem cases are designed to evaluate performance of the proposed heuristics in Case A, Case B and Case C by comparing the heuristic solutions with the optimal solutions. For the performance evaluation of the heuristic solutions, a total of four experiments are conducted, including the Step 1 and Step 3 experiments described in Chapter 2.4 and additional experiments of Step 4 and Step 5 described as follows. Table 21 shows three levels of QoS variables and the maximum level of service parameters for the communication intensive service and the computation intensive service used in Step 4 and Step 5 experiments.

Table 21

QoS Levels and Limit of Service Parameters for Communication Intensive Service and Computation Intensive Service in Step 4 and Step 5 Experiments

Experiment	Communication Intensive Service (s = 1)		Computation Intensive Service (s = 2)	
	QoS Levels ($Q_{kp_1}^l$)	Limit of service parameter ($A_{kd_1 i}^l$)	QoS Levels ($Q_{kp_1}^l$)	Limit of service parameter ($A_{kd_1 i}^l$)
Step 4	L (6~7) M (13~14) H (20~21)	3	L (6~7) M (14~15) H (22~23)	3
Step 5	L (5~6) M (12~13) H (18~19)	3	L (5~6) M (11~12) H (17~18)	3

As shown in Table 21, Step 4 experiment has a range of QoS requirement for each level and thus randomly assigns a specific value of the QoS variable given the QoS requirement level from a client. Hence, Step 4 experiment lets each client using the communication intensive service set the maximum level of the service parameter to 3 and set the QoS requirement to one of three levels of L, M and H, and a specific QoS value is randomly selected from 6 to 7 for L, from 13 to 14 for M and from 20 to 21 for H. It also lets each client using the computation intensive service set the maximum level of the service parameter to 3 and set the QoS requirement to one of three levels of L, M and H, and a specific QoS value is randomly selected from 6 to 7 for L, from 14 to 15 for M and from 22 to 23 for H.

Similarly, Step 5 experiment lets each client using the communication intensive service set the maximum level of the service parameter to 3 and set the QoS requirement to one of three levels of L, M and H, and a specific QoS value is randomly selected from

5 to 6 for L, from 12 to 13 for M and from 18 to 19 for H. It also lets each client using the computation intensive service set the maximum level of the service parameter to 3 and set the QoS requirement to one of three levels of L, M and H, and a specific QoS value is randomly selected from 5 to 6 for L, from 11 to 12 for M and from 17 to 18 for H.

Table 22 shows capacity limits of two resource variables: CPU resource and bandwidth resource for the communication-centered server and the computation-centered server.

Table 22

Capacity Limits of Two Resource Variables ($\boldsymbol{R_{i1}^{l}}$, $\boldsymbol{R_{i2}^{l}}$) for Communication-Centered Server and Computation-Centered Server and Number of Servers used in Step 4 and Step 5 Experiments

Experiment	Communication-Centered	Computation-Centered	Number of Servers
Step 4	S (23~24, 104~105) M (31~32, 166~167) L (39~40, 208~209)	S (115~116, 21~22) M (183~184, 28~29) L (229~230, 35~36)	10
Step 5	S (19~20, 95~96) M (25~26, 139~140) L (31~32, 177~178)	S (91~92, 19~20) M (133~134, 26~27) L (169~170, 32~33)	20

As shown in Table 22, Step 4 experiment has a range of capacity limits for two resource variables: CPU resource and bandwidth resource with resource levels of S, M and L. Given the resource level from a server, it randomly assigns each specific value for CPU resource and bandwidth resource as the capacity limits. Hence, Step 4 experiment lets each communication-centered server set its resource capacity limits to one of three levels of S, M and L, and a specific resource amount is randomly selected from 23 to 24 as the capacity limits of CPU resource and from 104 to 105 as the capacity limits of

bandwidth resource for S, from 31 to 32 of CPU resource and from 166 to 167 of bandwidth resource for M and from 39 to 40 of CPU resource and from 208 to 209 of bandwidth resource for L. It also lets each computation-centered server set its resource capacity limits to one of three levels of S, M and L, and a specific resource amount is randomly selected from 115 to 116 as the capacity limits of CPU resource and from 21 to 22 as the capacity limits of bandwidth resource for S, from 183 to 184 of CPU resource and from 28 to 29 of bandwidth resource for M and from 229 to 230 of CPU resource and from 35 to 36 of bandwidth resource for L.

Similarly, Step 5 experiment lets each communication-centered server set its resource capacity limits to one of three levels of S, M and L, and a specific resource amount is randomly selected from 19 to 20 as the capacity limits of CPU resource and from 95 to 96 as the capacity limits of bandwidth resource for S, from 25 to 26 of CPU resource and from 139 to 140 of bandwidth resource for M and from 31 to 32 of CPU resource and from 177 to 178 of bandwidth resource for L. It also lets each computation-centered server set its resource capacity limits to one of three levels of S, M and L, and a specific resource amount is randomly selected from 91 to 92 as the capacity limits of CPU resource and from 19 to 20 as the capacity limits of bandwidth resource for S, from 133 to 134 of CPU resource and from 26 to 27 of bandwidth resource for M and from 169 to 170 of CPU resource and from 32 to 33 of bandwidth resource for L. Note that Step 4 experiment has ten servers in total, and Step 5 experiment has twenty servers in total.

Table 23 shows the resource and QoS impact models of the communication intensive service and the computation intensive service used in Step 4 and Step 5 experiments. Table 24 and Table 25 describe the design of experimental setup with

different numbers of service requests used in Step 4 and Step 5 experiments. In the first column of Tables 24 and 25, the server configuration is indicated by #X-#Y with #X for the configuration of the first half of servers and #Y for the second half of servers, where # representing the number of servers and X or Y representing the resource capacity level. For example, in Table 24 for Step 4 experiments with ten servers, 5S-5S for the server configuration 1 represents that the resource capacity of the first 5 servers is set to the S level and the resource capacity of the other 5 servers is set to the S level. Note that each experiment has a total of 756 problem cases, which consist of 252 problem cases for Case A, 252 problem cases for Case B and 252 problem cases for Case C, and all servers provide both service types of the communication intensive service and the computation intensive service in each experiment.

Table 23

Resource and QoS Impact Models of Communication Intensive Service and Computation Intensive Service in Step 4 and Step 5 Experiments

Experiment	Communication Intensive Service (s = 1)	Computation Intensive Service (s = 2)
Step 4	R_{ki1} = 0.2*$A_{kd_1 i}$ R_{ki2} = 6.9*$A_{kd_1 i}$ $Q_{kp_1 i} = 2R_{ki1} + R_{ki2}$	R_{ki1} = 7.6*$A_{kd_2 i}$ R_{ki2} = 0.1*$A_{kd_2 i}$ $Q_{kp_2 i} = R_{ki1} + 3R_{ki2}$
Step 5	R_{ki1} = 0.1*$A_{kd_1 i}$ R_{ki2} = 6.3*$A_{kd_1 i}$ $Q_{kp_1 i} = 3R_{ki1} + R_{ki2}$	R_{ki1} = 6.0*$A_{kd_2 i}$ R_{ki2} = 0.1*$A_{kd_2 i}$ $Q_{kp_2 i} = R_{ki1} + 2R_{ki2}$

Table 24

Design of Experimental Setup with Different Numbers of Service Requests in Step 4 Experiment

Service Request / Server Configuration	Communication intensive service				Computation intensive service				Both types of services (Communication, Computation)			
	All-L	All-M	All-H	Mixed	All-L	All-M	All-H	Mixed	All-L	All-M	All-H	Mixed
1. All communication centered at 5S-5S (1-1)	A.13 B.137 C.188	A.5 B.65 C.87	A.3 B.46 C.63	A.6 B.64 C.88	A.2 B.28 C.38	A.1 B.8 C.13	A.1 B.9 C.13	A.2 B.17 C.25	A.14 B.154 C.213	A.7 B.74 C.100	A.4 B.40 C.63	A.2 B.18 C.25
2. All communication centered at 5M-5M (1-2)	A.21 B.220 C.300	A.10 B.110 C.150	A.7 B.75 C.100	A.11 B.110 C.150	A.4 B.37 C.50	A.2 B.19 C.25	A.1 B.9 C.13	A.2 B.19 C.25	A.25 B.247 C.338	A.11 B.120 C.163	A.8 B.84 C.113	A.4 B.35 C.49
3. All communication centered at 5L-5L (1-3)	A.28 B.275 C.375	A.14 B.140 C.188	A.9 B.95 C.125	A.12 B.136 C.188	A.4 B.46 C.63	A.2 B.19 C.25	A.1 B.9 C.13	A.3 B.28 C.38	A.32 B.312 C.425	A.15 B.159 C.213	A.10 B.104 C.138	A.5 B.58 C.75
4. All communication centered at 5S-5M (1-4)	A.14 B.185 C.244	A.5 B.90 C.119	A.4 B.62 C.82	A.6 B.89 C.119	A.3 B.33 C.44	A.1 B.14 C.19	A.1 B.9 C.13	A.2 B.19 C.25	A.4 B.203 C.275	A.7 B.104 C.132	A.5 B.66 C.88	A.2 B.28 C.38
5. All communication centered at 5S-5L (1-5)	A.15 B.215 C.282	A.6 B.105 C.138	A.4 B.73 C.94	A.7 B.106 C.138	A.2 B.38 C.50	A.1 B.14 C.19	A.1 B.9 C.13	A.2 B.24 C.32	A.15 B.240 C.319	A.7 B.119 C.157	A.4 B.76 C.100	A.2 B.40 C.50
6. All communication centered at 5M-5L (1-6)	A.23 B.250 C.338	A.10 B.125 C.169	A.7 B.85 C.113	A.10 B.127 C.169	A.3 B.42 C.57	A.2 B.19 C.25	A.1 B.9 C.13	A.2 B.24 C.32	A.25 B.283 C.382	A.12 B.139 C.188	A.9 B.94 C.125	A.3 B.49 C.63
7. All computation centered at 5S-5S (2-1)	A.2 B.28 C.38	A.1 B.9 C.13	A.1 B.9 C.13	A.2 B.18 C.25	A.13 B.140 C.188	A.6 B.65 C.88	A.4 B.46 C.63	A.6 B.66 C.88	A.15 B.154 C.213	A.6 B.75 C.100	A.4 B.46 C.63	A.3 B.28 C.38
8. All computation centered at 5M-5M (2-2)	A.3 B.38 C.50	A.2 B.19 C.25	A.1 B.9 C.13	A.2 B.19 C.25	A.22 B.220 C.300	A.10 B.110 C.150	A.7 B.75 C.100	A.10 B.111 C.150	A.23 B.248 C.338	A.11 B.120 C.163	A.9 B.82 C.113	A.4 B.38 C.50
9. All computation centered at 5L-5L (2-3)	A.4 B.46 C.63	A.2 B.18 C.25	A.1 B.9 C.13	A.3 B.28 C.38	A.26 B.275 C.375	A.14 B.140 C.188	A.9 B.92 C.125	A.13 B.138 C.188	A.32 B.310 C.425	A.16 B.168 C.213	A.10 B.109 C.138	A.5 B.58 C.75
10. All computation centered at 5S-5M (2-4)	A.2 B.33 C.44	A.1 B.14 C.19	A.1 B.9 C.13	A.2 B.19 C.25	A.14 B.185 C.244	A.6 B.90 C.119	A.5 B.61 C.82	A.6 B.89 C.119	A.16 B.205 C.275	A.6 B.98 C.132	A.3 B.66 C.88	A.2 B.34 C.44
11. All computation centered at 5S-5L (2-5)	A.3 B.38 C.50	A.1 B.14 C.19	A.1 B.9 C.13	A.2 B.24 C.32	A.15 B.215 C.282	A.7 B.105 C.138	A.4 B.71 C.94	A.5 B.104 C.138	A.15 B.240 C.319	A.7 B.124 C.157	A.4 B.77 C.100	A.2 B.44 C.57
12. All computation centered at 5M-5L	A.4 B.42	A.2 B.18	A.1 B.9	A.2 B.24	A.20 B.250	A.11 B.125	A.7 B.85	A.11 B.124	A.23 B.281	A.12 B.138	A.8 B.92	A.4 B.48

(2-6)	C.57	C.25	C.13	C.32	C.338	C.169	C.113	C.169	C.382	C.188	C.125	C.63
13. Both server types (communication, computation) at 5S-5S (3-1)	A.2 B.88 C.113	A.1 B.39 C.50	A.1 B.29 C.38	A.2 B.44 C.57	A.3 B.87 C.113	A.1 B.39 C.50	A.1 B.29 C.38	A.2 B.43 C.57	A.4 B.168 C.213	A.2 B.70 C.100	A.1 B.48 C.63	A.2 B.24 C.88
14. Both server types (communication, computation) at 5M-5M (3-2)	A.3 B.137 C.175	A.2 B.68 C.88	A.1 B.44 C.57	A.2 B.69 C.88	A.4 B.138 C.175	A.2 B.68 C.88	A.1 B.44 C.57	A.2 B.68 C.88	A.5 B.265 C.338	A.2 B.128 C.163	A.2 B.89 C.113	A.3 B.38 C.138
15. Both server types (communication, computation) at 5L-5L (3-3)	A.4 B.171 C.219	A.2 B.84 C.107	A.1 B.54 C.69	A.3 B.87 C.113	A.3 B.171 C.219	A.2 B.83 C.107	A.1 B.54 C.69	A.2 B.88 C.113	A.6 B.336 C.425	A.3 B.168 C.213	A.2 B.108 C.138	A.5 B.58 C.163
16. Both server types (communication, computation) at 5M-5M (3-4)	A.2 B.92 C.119	A.2 B.43 C.57	A.1 B.29 C.38	A.2 B.44 C.57	A.3 B.133 C.169	A.1 B.64 C.82	A.1 B.44 C.57	A.2 B.68 C.88	A.4 B.218 C.275	A.2 B.103 C.132	A.1 B.69 C.88	A.2 B.28 C.107
17. Both server types (communication, computation) at 5S-5L (3-5)	A.4 B.96 C.125	A.2 B.43 C.57	A.1 B.29 C.38	A.3 B.48 C.63	A.3 B.163 C.207	A.1 B.79 C.100	A.1 B.54 C.69	A.2 B.83 C.107	A.5 B.250 C.319	A.3 B.123 C.157	A.1 B.79 C.100	A.2 B.39 C.113
18. Both server types (communication, computation) at 5M-5L (3-6)	A.4 B.142 C.182	A.2 B.68 C.88	A.1 B.44 C.57	A.3 B.73 C.94	A.4 B.166 C.213	A.2 B.83 C.107	A.1 B.54 C.69	A.2 B.84 C.107	A.6 B.299 C.382	A.3 B.148 C.188	A.2 B.99 C.125	A.3 B.47 C.150
19. Both server types (communication, computation) at 5M-5S (3-7)	A.3 B.134 C.169	A.1 B.64 C.82	A.1 B.44 C.57	A.2 B.68 C.88	A.3 B.91 C.119	A.2 B.43 C.57	A.1 B.29 C.38	A.2 B.44 C.57	A.5 B.215 C.275	A.2 B.103 C.132	A.2 B.53 C.69	A.3 B.38 C.119
20. Both server types (communication, computation) at 5L-5S (3-8)	A.2 B.162 C.207	A.1 B.79 C.100	A.1 B.54 C.69	A.2 B.83 C.107	A.4 B.95 C.125	A.2 B.43 C.57	A.1 B.29 C.38	A.3 B.48 C.63	A.6 B.249 C.319	A.3 B.122 C.157	A.2 B.63 C.82	A.2 B.48 C.125
21. Both server types (communication, computation) at 5L-5M (3-9)	A.4 B.166 C.213	A.2 B.83 C.107	A.1 B.54 C.69	A.2 B.83 C.107	A.5 B.140 C.182	A.2 B.68 C.88	A.1 B.44 C.57	A.3 B.72 C.94	A.6 B.298 C.382	A.2 B.147 C.188	A.2 B.98 C.125	A.4 B.45 C.157

Table 25

Design of Experimental Setup with Different Numbers of Service Requests in Step 5 Experiment

Service Request / Server Configuration	Communication intensive service				Computation intensive service				Both types of services (Communication, Computation)			
	All-L	All-M	All-H	Mixed	All-L	All-M	All-H	Mixed	All-L	All-M	All-H	Mixed
1. All communication centered at 10S-10S (1-1)	A.13 B.290 C.375	A.5 B.135 C.175	A.4 B.96 C.125	A.5 B.136 C.175	A.2 B.58 C.75	A.1 B.19 C.25	A.1 B.20 C.25	A.2 B.38 C.50	A.16 B.328 C.425	A.7 B.155 C.200	A.3 B.78 C.125	A.6 B.136 C.176
2. All communication centered at 10M-10M (1-2)	A.20 B.420 C.550	A.10 B.210 C.275	A.5 B.135 C.175	A.9 B.211 C.275	A.3 B.78 C.100	A.2 B.39 C.50	A.1 B.19 C.25	A.2 B.38 C.50	A.23 B.478 C.625	A.11 B.230 C.300	A.7 B.154 C.200	A.6 B.134 C.175
3. All communication centered at 10L-10L (1-3)	A.25 B.540 C.700	A.13 B.270 C.350	A.7 B.175 C.225	A.13 B.268 C.350	A.4 B.96 C.125	A.2 B.39 C.50	A.1 B.20 C.25	A.3 B.58 C.75	A.29 B.613 C.800	A.14 B.308 C.400	A.9 B.194 C.250	A.7 B.154 C.200
4. All communication centered at 10S-10M (1-4)	A.14 B.360 C.463	A.6 B.175 C.225	A.4 B.116 C.150	A.6 B.175 C.225	A.3 B.68 C.88	A.1 B.29 C.38	A.1 B.19 C.25	A.2 B.39 C.50	A.16 B.404 C.525	A.8 B.194 C.250	A.3 B.125 C.163	A.5 B.134 C.175
5. All communication centered at 10S-10L (1-5)	A.13 B.420 C.538	A.5 B.205 C.263	A.5 B.136 C.175	A.6 B.205 C.263	A.3 B.77 C.100	A.1 B.29 C.38	A.1 B.19 C.25	A.2 B.48 C.63	A.16 B.474 C.613	A.8 B.234 C.300	A.4 B.145 C.188	A.5 B.144 C.188
6. All communication centered at 10M-10L (1-6)	A.20 B.480 C.625	A.10 B.240 C.313	A.6 B.155 C.200	A.10 B.241 C.313	A.3 B.87 C.113	A.2 B.39 C.50	A.1 B.20 C.25	A.2 B.49 C.63	A.22 B.547 C.713	A.11 B.269 C.350	A.7 B.174 C.225	A.7 B.145 C.188
7. All computation centered at 10S-10S (2-1)	A.3 B.57 C.75	A.1 B.19 C.25	A.1 B.20 C.25	A.2 B.37 C.50	A.15 B.290 C.375	A.6 B.135 C.175	A.4 B.96 C.125	A.5 B.135 C.175	A.14 B.328 C.425	A.7 B.154 C.200	A.4 B.91 C.125	A.2 B.59 C.72
8. All computation centered at 10M-10M (2-2)	A.4 B.77 C.100	A.2 B.38 C.50	A.1 B.19 C.25	A.2 B.38 C.50	A.20 B.420 C.550	A.10 B.210 C.275	A.6 B.135 C.175	A.9 B.213 C.275	A.22 B.478 C.625	A.11 B.229 C.300	A.7 B.155 C.200	A.5 B.115 C.150
9. All computation centered at 10L-10L (2-3)	A.5 B.97 C.125	A.2 B.39 C.50	A.1 B.19 C.25	A.3 B.57 C.75	A.25 B.540 C.700	A.13 B.270 C.350	A.8 B.175 C.225	A.13 B.269 C.350	A.28 B.608 C.800	A.14 B.305 C.400	A.9 B.191 C.250	A.5 B.114 C.150
10. All computation entered at 10S-10M (2-4)	A.3 B.68 C.88	A.1 B.29 C.38	A.1 B.20 C.25	A.2 B.38 C.50	A.14 B.356 C.463	A.7 B.175 C.225	A.5 B.115 C.150	A.5 B.174 C.225	A.15 B.405 C.525	A.6 B.193 C.250	A.3 B.125 C.163	A.2 B.88 C.113
11. All computation entered at 10S-10L (2-5)	A.2 B.78 C.100	A.1 B.30 C.38	A.1 B.19 C.25	A.2 B.49 C.63	A.15 B.420 C.538	A.6 B.205 C.263	A.4 B.137 C.175	A.6 B.204 C.263	A.16 B.475 C.613	A.6 B.234 C.300	A.4 B.146 C.188	A.2 B.88 C.114
12. All computation	A.3	A.2	A.1	A.2	A.21	A.10	A.7	A.10	A.24	A.11	A.8	A.4

centered at 10M-10L (2-6)	B.87 C.113	B.39 C.50	B.20 C.25	B.49 C.63	B.480 C.625	B.240 C.313	B.155 C.200	B.240 C.313	B.546 C.713	B.269 C.350	B.173 C.225	B.117 C.150
13. Both server types (communication, computation) at 10S-10S (3-1)	A.2 B.178 C.225	A.1 B.79 C.100	A.1 B.60 C.75	A.2 B.88 C.113	A.3 B.179 C.225	A.1 B.79 C.100	A.1 B.59 C.75	A.2 B.88 C.113	A.4 B.336 C.425	A.2 B.158 C.200	A.1 B.98 C.125	A.3 B.99 C.175
14. Both server types (communication, computation) at 10M-10M (3-2)	A.4 B.257 C.325	A.2 B.128 C.163	A.1 B.79 C.100	A.2 B.128 C.163	A.4 B.257 C.325	A.2 B.130 C.163	A.1 B.79 C.100	A.2 B.128 C.163	A.5 B.497 C.625	A.2 B.238 C.300	A.2 B.159 C.200	A.4 B.127 C.275
15. Both server types (communication, computation) at 10L-10L (3-3)	A.4 B.326 C.413	A.2 B.159 C.200	A.1 B.99 C.125	A.2 B.167 C.213	A.5 B.326 C.413	A.2 B.159 C.200	A.1 B.100 C.125	A.3 B.167 C.213	A.6 B.632 C.800	A.4 B.316 C.400	A.2 B.198 C.250	A.5 B.136 C.350
16. Both server types (communication, computation) at 10S-10M (3-4)	A.3 B.186 C.238	A.2 B.90 C.113	A.1 B.59 C.75	A.2 B.88 C.113	A.3 B.248 C.313	A.1 B.120 C.150	A.1 B.80 C.100	A.2 B.128 C.163	A.4 B.417 C.525	A.2 B.199 C.250	A.2 B.129 C.163	A.3 B.128 C.225
17. Both server types (communication, computation) at 10S-10L (3-5)	A.4 B.196 C.250	A.2 B.89 C.113	A.1 B.60 C.75	A.3 B.97 C.125	A.2 B.308 C.388	A.1 B.150 C.188	A.1 B.99 C.125	A.2 B.158 C.200	A.5 B.486 C.613	A.3 B.239 C.300	A.2 B.149 C.188	A.4 B.126 C.263
18. Both server types (communication, computation) at 10M-10L (3-6)	A.5 B.266 C.338	A.2 B.129 C.163	A.1 B.79 C.100	A.2 B.138 C.157	A.3 B.317 C.400	A.2 B.159 C.200	A.1 B.100 C.125	A.2 B.158 C.200	A.6 B.565 C.713	A.3 B.279 C.350	A.2 B.179 C.225	A.5 B.127 C.300
19. Both server types (communication, computation) at 10M-10S (3-7)	A.2 B.248 C.313	A.1 B.119 C.150	A.1 B.80 C.100	A.2 B.128 C.163	A.3 B.187 C.238	A.2 B.89 C.113	A.1 B.60 C.75	A.2 B.89 C.113	A.4 B.417 C.525	A.2 B.199 C.250	A.1 B.129 C.163	A.3 B.98 C.213
20. Both server types (communication, computation) at 10L-10S (3-8)	A.3 B.308 C.388	A.1 B.149 C.188	A.1 B.99 C.125	A.2 B.158 C.200	A.4 B.196 C.250	A.2 B.90 C.113	A.1 B.59 C.75	A.3 B.97 C.125	A.5 B.486 C.613	A.3 B.239 C.300	A.1 B.149 C.188	A.3 B.106 C.263
21. Both server types (communication, computation) at 10L-10M	A.3 B.317 C.400	A.2 B.159 C.200	A.1 B.100 C.125	A.2 B.158 C.200	A.4 B.266 C.338	A.2 B.129 C.163	A.1 B.79 C.100	A.3 B.137 C.175	A.7 B.565 C.713	A.3 B.278 C.350	A.2 B.178 C.225	A.4 B.137 C.313

2.7. Results and Discussions

This section first defines key measures for performance comparison in terms of solution optimality between the optimal solutions and the heuristic solutions for various problem cases designed in Chapter 2.6. Then, it provides the computation times of obtaining the optimal solutions and the heuristic solutions to examine the scalability of those methods.

Regarding the solution optimality, two measures are introduced: the number of dropped service requests and the average ratio of QoS values assigned for all served service requests. Equation (8) as shown below is the objective function of the optimization problem in Chapter 2.3, which consists of two parts as indicated in Equations (9) and (10).

$$\text{Minimize} \sum_k \sum_{p_s} \frac{\left|\sum_i Q_{kp_si} - Q^l_{kp_s}\right|}{Q^l_{kp_s} * P_s} \tag{8}$$

If $\sum_i Q_{kp_si} < Q^l_{kp_s}$, then Equation (8) can be written as Equation (9).

$$\sum_k \sum_{p_s} \left(\frac{1}{P_s} - \frac{\sum_i Q_{kp_si}}{Q^l_{kp_s} * P_s}\right) \tag{9}$$

If $\sum_i Q_{kp_si} \geq Q^l_{kp_s}$, then Equation (8) can be written as Equation (10).

$$\sum_k \sum_{p_s} \left(\frac{\sum_i Q_{kp_si}}{Q^l_{kp_s} * P_s} - \frac{1}{P_s}\right) \tag{10}$$

As mentioned earlier in Chapter 2.5.3, all experiments in this study consider two services providing QoS at least a service request's minimum QoS requirement for all the served service requests. Thus, if a service request is served by a server, then $\sum_i Q_{kp_si} \geq Q^l_{kp_s}$. If a service request is not served by any server, then $\sum_i Q_{kp_si} = 0$. Note that this

study considers one QoS variable (p_s =1) for each service type, Equations (9) and (10) can be transformed as shown below in Equations (11) and (12).

If client *k*'s service request is not served by any server (i.e. $\sum_i Q_{kp_si} = 0$ and $\sum_i Q_{kp_si} < Q^l_{kp_s}$), then Equation (9) becomes Equation (11) indicating the total number of dropped service requests as the first measure.

$$\sum_k 1 \quad \text{for dropped service requests } k \tag{11}$$

If client *k*'s service request is served by a server (i.e. $\sum_i Q_{kp_si} \geq Q^l_{kp_s}$), then Equation (10) becomes Equation (12).

$$\sum_k \left(\frac{\sum_i Q_{k1i}}{Q^l_{k1}} - 1 \right) \tag{12}$$

From Equation (12), the second key measure of *Service AVerage ratio* (*SAV* ratio) is drawn for the comparison in solution optimality as shown below in Equation (13).

$$\sum_k \left(\frac{\sum_i Q_{k1i}}{Q^l_{k1}} \right) / \sum k \tag{13}$$

Equation (12) considers ratio of the difference between the provided QoS and the required QoS to the required QoS, and then each ratio is summed over all served service requests. Equation (13), as the second measure of performance comparison, considers the average ratio of the provided QoS to the required QoS for all served service requests.

To analyze performance of the resource allocation heuristics in solution optimality, two key measures are introduced: the number of dropped service requests in Equation (11) and the *SAV* ratio in Equation (13). In addition, the percentage of having the same service decisions as in the optimal solutions and the percentage of having different service decisions from the optimal solutions are provided for the comparison in solution optimality. To analyze scalability of the resource allocation heuristics, the

computation times of obtaining the optimal solutions and the heuristic solutions are compared in Chapter 2.7.2.

2.7.1 Solution optimality. Table 26 shows the comparisons between the optimal solutions and the heuristic solutions from the experiments of Step 1, Step 3, Step 4 and Step 5 by using the first set of probability parameters indicated in Table 14.

Table 26

Comparisons between the Optimal Solutions and the Heuristic Solutions with Parameter Set 1

Experiment	Case	Same Decision	Different Decision	Difference in Dropped Requests, (heuristic solution - optimal solution) in number (%)	*SAV* ratio	
Step 1	A	44.44% (112/252)	55.56% (140/252)	0	Heuristic Solutions	Avg. 1.04 Range 1.02 ~ 1.08 St.dev 0.02
					Optimal Solutions	Same above
	B	58.33% (147/252)	41.67% (105/252)	0	Heuristic Solutions	Avg. 1.04 Range 1.02 ~ 1.08 St.dev 0.02
					Optimal Solutions	Same above
	C	52.78% (133/252)	47.22% (119/252)	Avg. 2.25 (4.39%) Range 0.00 ~25.58 St.dev 4.61	Heuristic Solutions	Avg. 1.04 Range 1.02 ~ 1.08 St.dev 0.02
					Optimal Solutions	Same above
Step 3	A	37.70% (95/252)	62.30% (157/252)	0	Heuristic Solutions	Avg. 1.20 Range 1.03 ~ 1.63 St.dev 0.14
					Optimal Solutions	Same above
	B	53.57% (135/252)	46.43% (117/252)	0	Heuristic Solutions	Avg. 1.21 Range 1.05 ~ 1.51 St.dev 0.13
					Optimal Solutions	Same above
	C	61.90% (156/252)	38.10% (96/252)	Avg. 0.62 (1.45%) Range -0.92 ~10.87 St.dev 1.62	Heuristic Solutions	Avg. 1.19 Range 1.04 ~ 1.49 St.dev 0.12
					Optimal Solutions	Avg. 1.17 Range 1.03 ~ 1.45 St.dev 0.11

Step 4	A	43.25% (109/252)	56.75% (143/252)	0		Heuristic Solutions	Avg. 1.11 Range 1.03 ~ 1.32 St.dev 0.06
						Optimal Solutions	Same above
	B	17.06% (43/252)	82.94% (209/252)	0		Heuristic Solutions	Avg. 1.11 Range 1.05 ~ 1.24 St.dev 0.05
						Optimal Solutions	Same above
	C	57.94% (146/252)	42.06% (106/252)	Avg. 5.19 (2.82%) Range 0.00 ~79.92 St.dev 12.71		Heuristic Solutions	Avg. 1.11 Range 1.04 ~ 1.24 St.dev 0.05
						Optimal Solutions	Avg. 1.10 Range 1.04 ~ 1.22 St.dev 0.05
Step 5	A	38.89% (98/252)	61.11% (154/252)	0		Heuristic Solutions	Avg. 1.11 Range 1.02 ~ 1.32 St.dev 0.06
						Optimal Solutions	Same above
	B	20.24% (51/252)	79.76% (201/252)	0		Heuristic Solutions	Avg. 1.11 Range 1.05 ~ 1.23 St.dev 0.05
						Optimal Solutions	Same above
	C	58.33% (147/252)	41.67% (105/252)	Avg. 9.62 (2.69%) Range 0.00 ~168.85 St.dev 24.74		Heuristic Solutions	Avg. 1.10 Range 1.05 ~ 1.23 St.dev 0. 05
						Optimal Solutions	Avg. 1.09 Range 1.04 ~ 1.20 St.dev 0.04

As shown in Table 26, the heuristic solutions using the first set of probability parameters are compared with the optimal solutions in the following four aspects: 1) the percentage of the same service decisions as the optimal solutions, 2) the percentage of different service decisions from the optimal solutions, 3) the difference in dropped service requests of the heuristic solutions compared with the optimal solutions and 4) the *SAV* ratio of the heuristic solutions and the optimal solutions.

For Case A in Step 1 experiment, about 44.44% of 252 problem cases show that the heuristic solutions have same service decisions as the optimal solutions, while about 55.56% of the problem cases show different service decisions from the optimal solutions. The heuristic solutions do not drop any service request as same as the optimal solutions due to a sufficient resource capacity at a server to satisfy all service requests of all clients in Case A. Moreover, the heuristic solutions serve all service requests of all clients with the same QoS values as the optimal solutions and thus have the same *SAV* ratio of 1.04 in average of all problem cases with the range of 1.02 to 1.08 and the standard deviation of 0.02. Here, the *SAV* ratio is an average ratio value of the provided QoS to the required QoS for all served service requests as indicated in Equation (13). If a service request selects to be served, then a server provides at least its QoS requirement, and as a result, the *SAV* ratio is at least 1 for any served service request. The *SAV* ratio of 1.04 in average means that, for all the served service requests, the provided QoS in average is 1.04 times to the QoS requirement.

For Case B in Step 1 experiment, about 58.33% of 252 problem cases show that the heuristic solutions have same service decisions as the optimal solutions, while about 41.67% of the problem cases show different service decisions from the optimal solutions.

The heuristic solutions do not drop any service request as same as the optimal solutions due to a sufficient resource capacity over all servers to satisfy all service requests of all clients in Case B. Moreover, the heuristic solutions serve all service requests of all clients with the same QoS values as the optimal solutions and thus have the same *SAV* ratio of 1.04 in average of all problem cases with the range of 1.02 to 1.08 and the standard deviation of 0.02, which implies that the provided QoS in average is 1.04 times to the QoS requirement for all served service requests.

For Case C Step 1 experiment, about 52.78% of 252 problem cases show that the heuristic solutions have same service decisions as the optimal solutions, while about 47.22% of the problem cases show different service decisions from the optimal solutions. Since neither each server nor all server together have a sufficient resource capacity to satisfy all service requests of all clients in Case C, both of the optimal solutions and the heuristic solutions have some service requests to be dropped. However, the heuristic solutions drop about two more service requests in average (i.e. it takes about 4.39% of all service requests) than the optimal solutions, and the difference range is 0.00 to 25.58 with the standard deviation of 4.61. The heuristic solutions have the same *SAV* ratio of 1.04 in average with the range of 1.02 to 1.08 and the standard deviation of 0.02 as the optimal solutions. It implies that the heuristic solutions serve three less service requests than the optimal solutions in average but provide the same QoS values as the optimal solutions for all the served service requests with the QoS provision of 1.04 times to the QoS requirement.

Table 15 from Chapter 2.5.1 shows service decisions of which service requests to be dropped in the optimal solutions for two problem cases in Case C. In one problem case

with server configuration 16 and seven clients, two service requests of the communication intensive service (client k = 6 and 7) are dropped. All the seven clients' service requests have one service type with one QoS level, and thus it implies that random service requests are dropped when overall resource capacity is not sufficient to serve all service requests of all clients.

In another problem case with server configuration 13 and sixteen clients as shown in Table 15, two service requests of the communication intensive service with the QoS level of H (client k = 3 and 16) and one service request of the computation intensive service with the QoS level of M (client k = 8) are dropped for Case C. The two dropped service requests of the communication intensive service (client k = 3 and 16) have the QoS level of H, which requires the largest resource consumption among the nine service requests of the communication intensive service. The resource requirement can be calculated by resource and QoS impact models in Table 6, and due to its linear relationship between service parameters and resource consumption, service requests with higher QoS levels need higher service parameters to meet their QoS requirements and thus larger resource consumption than the service requests with lower QoS levels. The other dropped service request of the computation intensive service (client k = 8) with the QoS level of M has the highest estimated objective value of 0.07 among all the sixteen service requests. With the objective function in Equation (1), the optimization problem in Chapter 2.3 tries to minimize the overall objective value, and as a result, the service request with the highest estimated objective value is dropped. It implies that a service request requiring the largest resource consumption and/or resulting in the worst estimated

objective value is dropped when overall resource capacity is not sufficient to serve all service requests of all clients.

Figure 4 shows the observation of dropping service requests for Case C, where overall resource is not sufficient to serve all service requests of all clients. If all service requests are either communication intensive services or computation intensive services with one QoS level of L, M or H, then a random service request is dropped. For the other cases, a service request requiring the largest resource consumption and/or resulting in the worst estimated objective value is dropped.

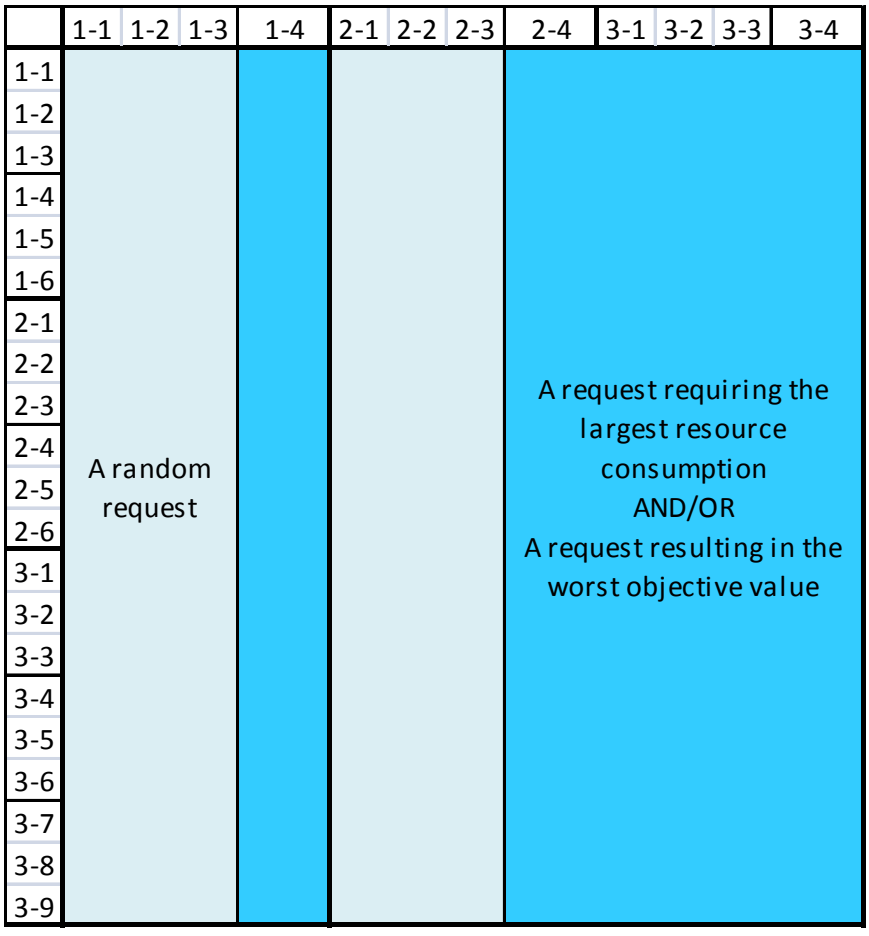


Figure 4. Observation of Dropping Service Requests for Case C.

For Case A and Case B in Step 3 experiment, about 37.70% and 53.57% of 252 problem cases show that the heuristic solutions have same service decisions as the optimal solutions, while about 62.30% and 46.43% of the problem cases show different service decisions from the optimal solutions respectively. The heuristic solutions do not drop any service request as same as the optimal solutions due to a sufficient resource capacity at a server (in Case A) or over all servers (in Case B) to satisfy all service requests of all clients. Moreover, the heuristic solutions serve all service requests of all clients with the same QoS values as the optimal solutions and thus have the same *SAV* ratio of 1.20 in average with the range of 1.03 to 1.63 and the standard deviation of 0.14 in Case A and the same *SAV* ratio of 1.21 in average with the range of 1.05 to 1.51 and the standard deviation of 0.13 in Case B. It implies that the provided QoS in average is 1.20 times and 1.21 times to the QoS requirement for all served service requests in both optimal and heuristic solutions for Case A and Case B, respectively.

For Case C in Step 3 experiment, about 61.90% of 252 problem cases show that the heuristic solutions have same service decisions as the optimal solutions, while about 38.10% of the problem cases show different service decisions from the optimal solutions. Since neither each server nor all server together have a sufficient resource capacity to satisfy all service requests of all clients in Case C, both of the optimal solutions and the heuristic solutions have some service requests to be dropped. However, the heuristic solutions drop about one more service request in average (i.e. it takes about 1.45% of all service requests) than the optimal solutions, and the difference range is -0.92 to 10.87 with the standard deviation of 1.62. Moreover, the heuristic solutions have the *SAV* ratio of 1.19 in average with the range of 1.04 to 1.49 and the standard deviation of 0.12, while

the optimal solutions have the *SAV* ratio of 1.17 in average with the range of 1.03 to 1.45 and the standard deviation of 0.11. It implies that the heuristic solutions serve one less service request than the optimal solutions in average but provide higher QoS values with the QoS provision of 1.19 times to the QoS requirement for all served service requests than the QoS provision of 1.17 times to the QoS requirement for all served service requests in the optimal solutions.

One unusual problem case shows a negative value of -0.92 in the difference of the dropped service requests, implying that the heuristic solution serves about one more service request than the optimal solution. Table 27 shows and examines service decisions made in the optimal solution and the heuristic solution at iteration 100 for the problem case. Different decisions made in the heuristic solution from the optimal solution are marked by "*" in Table 27.

Table 27

Service Decisions made in the Optimal Solution and the Heuristic Solution at Iteration 100

Client	Service	QoS Requirement ($Q^{l}_{kp_s}$)	Estimated Objective Value	Optimal Solution		Heuristic Solution	
				Server	Service parameter (A_{kd_si}) and QoS level(Q_{kp_si})	Server	Service parameter (A_{kd_si}) and QoS level(Q_{kp_si})
$k = 1$	comp. intensive service	L (4)	0.23	comm. centered server	$A_{1,1,1}$: 1 $Q_{1,1,1}$: 4.90	comm. centered server	$A_{1,1,3}$: 1 $Q_{1,1,3}$: 4.90
$k = 2$	comp. intensive service	L (4)	0.23	comm. centered server	$A_{2,1,1}$: 1 $Q_{2,1,1}$: 4.90	comm. centered server	$A_{2,1,1}$: 1 $Q_{2,1,1}$: 4.90
$k = 3$	comp. intensive service	L (3)	0.63	none	none	comm. centered server*	$A_{3,1,2}$: 1* $Q_{3,1,2}$: 4.90*
$k = 4$	comp. intensive service	H (18)	0.09	none	none	comm. centered server*	$A_{4,1,3}$: 4* $Q_{4,1,3}$: 19.60*
$k = 5$	comp. intensive service	L (3)	0.63	none	none	comm. centered server*	$A_{5,1,3}$: 1* $Q_{5,1,3}$: 4.90*
$k = 6$	comp. intensive service	M (14)	0.05	comm. centered server	$A_{6,1,3}$: 3 $Q_{6,1,3}$: 14.70	none*	none*
$k = 7$	comp. intensive service	H (19)	0.03	comm. centered server	$A_{7,1,1}$: 4 $Q_{7,1,1}$: 19.60	none*	none*
$k = 8$	comp. intensive service	H (19)	0.03	comm. centered server	$A_{8,1,2}$: 4 $Q_{8,1,2}$: 19.60	comm. centered server	$A_{8,1,1}$: 4 $Q_{8,1,1}$: 19.60
$k = 9$	comp. intensive service	H (19)	0.03	comm. centered server	$A_{9,1,3}$: 4 $Q_{9,1,3}$: 19.60	comm. centered server	$A_{9,1,2}$: 4 $Q_{9,1,2}$: 19.60
$k = 10$	comp. intensive service	H (18)	0.09	none	none	none	none
$k = 11$	comp. intensive service	L (4)	0.23	comm. centered server	$A_{11,1,2}$: 1 $Q_{11,1,2}$: 4.90	comm. centered server	$A_{11,1,1}$: 1 $Q_{11,1,1}$: 4.90
$k = 12$	comp. intensive service	L (3)	0.63	comm. centered server	$A_{12,1,2}$: 1 $Q_{12,1,2}$: 4.90	comm. centered server	$A_{12,1,2}$: 1 $Q_{12,1,2}$: 4.90

The problem case in Table 27 has that all servers are the communication-centered servers with the resource level of S (i.e. server configuration 2) and all service requests are computation intensive services with mixed QoS levels of L, M and H. The optimal solution drops a total of four service requests: two computation intensive service with the QoS level of H (client k = 4 and 10) and two computation intensive services with the QoS level of L (client k = 3 and 5) due to insufficient resource capacity to satisfy all service requests of all clients. The two dropped computation intensive service (client k = 4 and 10) has the QoS level of H, which requires the largest resource consumption, and the other two dropped computation intensive services (client k = 3 and 5) with the QoS level of L have the highest estimated objective value of 0.63 among twelve service requests of the computation intensive service. In general, the optimal solution selects to serve service requests of smaller resource requirement and, instead, drop service requests of larger resource requirement in order to serve as many clients' service requests as it can. However, in a few cases, the optimal solution selects to serve service requests of larger resource requirement but with lower objective value and, instead, drop service requests of smaller resource requirement but with higher objective value in order to minimize the overall objective value for all service requests of all clients as shown in Equation (1). On the other hand, the heuristic solution drops two service requests of the computation intensive service with the QoS level of H (client k = 7 and 10) requiring the largest resource consumption and one service request of the computation intensive service with the QoS level of M (client k = 6) requiring the second largest resource consumption among twelve service requests of the computation intensive service. Instead, the heuristic

solution selects to serve two more service requests of the computation intensive service (client k = 3 and 5) with the QoS level of L, which require smaller resource consumption. Using the heuristics do not consider an impact of serving a service request on the overall objective value for all service requests of all clients, and thus the heuristic solution selects to serve more number of service requests than the optimal solution but results in the worse objective value for a few cases.

For Case A and Case B in Step 4 experiment, about 43.25% and 17.06% of 252 problem cases show that the heuristic solutions have same service decisions as the optimal solutions, while about 56.75% and 82.94% of the problem cases show different service decisions from the optimal solutions respectively. The heuristic solutions do not drop any service request as same as the optimal solutions due to a sufficient resource capacity at a server (in Case A) or over all servers (in Case B) to satisfy all service requests of all clients. Moreover, the heuristic solutions serve all service requests of all clients with the same QoS values as the optimal solutions and thus have the same SAV ratio of 1.11 in average with the range of 1.03 to 1.32 and the standard deviation of 0.06 in Case A and with the range of 1.05 to 1.24 and the standard deviation of 0.05 in Case B. It implies that the provided QoS in average is 1.11 times to the QoS requirement for all served service requests in both optimal and heuristic solutions for Case A and Case B.

For Case C in Step 4 experiment, about 57.94% of 252 problem cases show that the heuristic solutions have same service decisions as the optimal solutions, while about 42.06% of the problem cases show different service decisions from the optimal solutions. Since neither each server nor all server together have a sufficient resource capacity to satisfy all service requests of all clients in Case C, both of the optimal solutions and the

heuristic solutions have some service requests to be dropped. However, the heuristic solutions drop about five more service requests in average (i.e. it takes about 2.82% of all service requests) than the optimal solutions, and the difference range is 0.00 to 79.92 with the standard deviation of 12.71. Moreover, the heuristic solutions have the *SAV* ratio of 1.11 in average with the range of 1.04 to 1.24 and the standard deviation of 0.05, while the optimal solutions have the *SAV* ratio of 1.10 in average with the range of 1.04 to 1.22 and the standard deviation of 0.05. It implies that the heuristic solutions serve five less service requests than the optimal solutions in average but provide higher QoS values with the QoS provision of 1.11 times to the QoS requirement for all served service requests than the QoS provision of 1.10 times to the QoS requirement for all served service requests in the optimal solutions.

For Case A and Case B in Step 5 experiment, about 38.89% and 20.24% of 252 problem cases show that the heuristic solutions have same service decisions as the optimal solutions, while about 61.11% and 79.76% of the problem cases show different service decisions from the optimal solutions respectively. The heuristic solutions do not drop any service request as same as the optimal solutions due to a sufficient resource capacity at a server (in Case A) or over all servers (in Case B) to satisfy all service requests of all clients. Moreover, the heuristic solutions serve all service requests of all clients with the same QoS values as the optimal solutions and thus have the same *SAV* ratio of 1.11 in average with the range of 1.02 to 1.32 and the standard deviation of 0.06 in Case A and with the range of 1.05 to 1.23 and the standard deviation of 0.05 in Case B. It implies that the provided QoS in average is 1.11 times to the QoS requirement for all served service requests in both optimal and heuristic solutions for Case A and Case B.

For Case C in Step 5 experiment, about 58.33% of 252 problem cases show that the heuristic solutions have same service decisions as the optimal solutions, while about 41.67% of the problem cases show different service decisions from the optimal solutions. Since neither each server nor all server together have a sufficient resource capacity to satisfy all service requests of all clients in Case C, both of the optimal solutions and the heuristic solutions have some service requests to be dropped. However, the heuristic solutions drop about ten more service requests in average (i.e. it takes about 2.69% of all service requests) than the optimal solutions, and the difference range is 0.00 to 168.85 with the standard deviation of 24.74. Moreover, the heuristic solutions have the *SAV* ratio of 1.10 in average with the range of 1.05 to 1.23 and the standard deviation of 0.05, while the optimal solutions have the *SAV* ratio of 1.09 in average with the range of 1.04 to 1.20 and the standard deviation of 0.04. It implies that the heuristic solutions serve ten less service requests than the optimal solutions in average but provide higher QoS values with the QoS provision of 1.10 times to the QoS requirement for all served service requests than the QoS provision of 1.09 times to the QoS requirement for all served service requests in the optimal solutions.

Table 28 shows the comparisons between the optimal solutions and the heuristic solutions in the experiments of Step 1, Step 3, Step 4 and Step 5 by using the second set of probability parameters for obtaining the heuristic solutions.

Table 28

Comparisons between the Optimal Solutions and the Heuristic Solutions with Parameter Set 2

Experiment	Case	Same Decision	Different Decision	Difference in Dropped Requests, (heuristic solution - optimal solution) in number (%)	*SAV* ratio		
Step 1	A	50.00% (126/252)	50.00% (126/252)	0	Heuristic Solutions	Avg. Range St.dev	1.04 1.02 ~ 1.08 0.02
					Optimal Solutions	Same above	
	B	63.10% (159/252)	36.90% (93/252)	0	Heuristic Solutions	Avg. Range St.dev	1.04 1.02 ~ 1.08 0.02
					Optimal Solutions	Same above	
	C	55.16% (139/252)	44.84% (113/252)	Avg. 2.23 (4.35%) Range 0.00 ~26.46 St.dev 4.62	Heuristic Solutions	Avg. Range St.dev	1.04 1.02 ~ 1.08 0.02
					Optimal Solutions	Same above	
Step 3	A	42.46% (107/252)	57.54% (145/252)	0	Heuristic Solutions	Avg. Range St.dev	1.20 1.03 ~ 1.59 0.14
					Optimal Solutions	Same above	
	B	51.98% (131/252)	48.02% (121/252)	0	Heuristic Solutions	Avg. Range St.dev	1.20 1.03 ~ 1.47 0.13
					Optimal Solutions	Same above	
	C	61.51% (155/252)	38.49% (97/252)	Avg. 0.77 (1.73%) Range -0.84 ~14.19 St.dev 2.04	Heuristic Solutions	Avg. Range St.dev	1.20 1.04 ~ 1.51 0.12
					Optimal Solutions	Avg. Range St.dev	1.18 1.03 ~ 1.47 0.11

Step 4	A	57.54% (145/252)	42.46% (107/252)	0		Heuristic Solutions	Avg. 1.11 Range 1.03 ~ 1.32 St.dev 0.06
						Optimal Solutions	Same above
	B	19.84% (50/252)	80.16% (202/252)	0		Heuristic Solutions	Avg. 1.11 Range 1.04 ~ 1.24 St.dev 0.05
						Optimal Solutions	Same above
	C	57.94% (146/252)	42.06% (106/252)	Avg. 6.02 (3.09%) Range 0.00 ~116.38 St.dev 15.66		Heuristic Solutions	Avg. 1.11 Range 1.04 ~ 1.24 St.dev 0.05
						Optimal Solutions	Avg. 1.10 Range 1.04 ~ 1.22 St.dev 0.04
Step 5	A	37.70% (95/252)	62.30% (157/252)	0		Heuristic Solutions	Avg. 1.10 Range 1.02 ~ 1.32 St.dev 0.06
						Optimal Solutions	Same above
	B	22.22% (56/252)	77.78% (196/252)	0		Heuristic Solutions	Avg. 1.11 Range 1.05 ~ 1.23 St.dev 0.05
						Optimal Solutions	Same above
	C	58.33% (147/252)	41.67% (105/252)	Avg. 10.46 (2.82%) Range 0.00 ~183.52 St.dev 27.70		Heuristic Solutions	Avg. 1.10 Range 1.05 ~ 1.22 St.dev 0. 05
						Optimal Solutions	Avg. 1.09 Range 1.04 ~ 1.20 St.dev 0.04

The experimental results with the second set of parameters are similar to the ones with the first set of parameters in terms of solution optimality. Hence, it concludes that the heuristic solutions are as good as or close to the optimal solutions. In Case A, where each server has a sufficient resource capacity and in Case B, where all the servers together have a sufficient resource capacity for all service requests of all clients, the heuristic solutions do not drop any service request as same as the optimal solutions. For Case A and Case B, the heuristic solutions have as same *SAV* ratios as the optimal solutions, implying that using the heuristics serves all the service requests with the same QoS values as in the optimal solutions. For Case C, where all the servers together do not have sufficient resource capacity for all service requests of all clients, the heuristic solutions drop more service requests of about 1% ~ 4% of all service requests than the optimal solutions. However, the heuristic solutions provide at least the same level of QoS to all served service requests as the optimal solutions with the same or higher *SAV* ratios.

2.7.2 Scalability. Table 29 and Figure 5 show computation times (in seconds) of obtaining the optimal and heuristic solutions in the four experiments using the first set of probability parameters. As Figure 5 illustrates, the average computation times of obtaining both the optimal and the heuristic solutions are increased as a problem case becomes more complicated with increasing numbers of service requests and limited resource capacity from Case A, Case B to Case C. The ranges of the computation times for obtaining the optimal and the heuristic solutions also become larger with higher values of the standard deviation from Case A, Case B to Case C. However, the rate of increase in the average computation times with the increasing problem complexity is much larger for the optimal solutions than the heuristic solutions.

Similarly, the average computation times of obtaining the optimal solutions and the heuristic solutions are increased with increasing numbers of servers (from two servers in Step 1 experiment to twenty servers in Step 5 experiment). The ranges of the computation times for obtaining both the optimal and the heuristic solutions also become larger with higher values of the standard deviation from Step 1 experiment to Step 5 experiment. However, the rate of increase in the average computation times with increasing numbers of servers is much larger for the optimal solutions than the heuristic solutions. Hence, using the heuristics demonstrate better computational efficiency and thus scalability than solving the optimization problem.

Note that, as shown in Table 29, the heuristic solutions for Case B have larger ranges of the computation times with higher values of the standard deviation than the ones for Case C in all experiments. The heuristic algorithm for Case B counts all computation times consumed until it generates the final heuristic solution for satisfying all service requests of all clients. On the other hand, for Case C the heuristic algorithm averages the computation times consumed to obtain the heuristic solution for each iteration.

Table 29

Computation Times (in seconds) of Obtaining the Optimal Solutions and the Heuristic Solutions with Parameter Set 1

		Step 1			Step 3			Step 4			Step 5		
		Avg.	Range	St.dev	Avg.	Range	St.dev	Avg.	Range	St.dev	Avg.	Range	St.dev
Optimal Solutions	Case A	0.12	0.02 - 0.60	0.11	0.06	0.02 - 0.30	0.05	0.07	0.03 - 0.39	0.05	0.10	0.03 - 0.48	0.08
	Case B	0.20	0.03 - 0.82	0.13	0.09	0.03 - 0.37	0.06	0.87	0.05 - 4.19	0.85	3.43	0.00 - 17.68	3.37
	Case C	0.41	0.03 - 3.31	0.45	1.50	0.00 - 64.88	6.14	32.83	0.07 - 138.54	50.33	59.20	0.23 - 170.60	65.05
Heuristic Solutions	Case A	0.00	0.00 - 0.00	0.00	0.00	0.00 - 0.00	0.00	0.00	0.00 - 0.00	0.00	0.00	0.00 - 0.00	0.00
	Case B	0.00	0.00 - 0.08	0.01	0.00	0.00 - 0.11	0.01	0.01	0.00 - 0.56	0.05	0.03	0.00 - 1.77	0.16
	Case C	0.00	0.00 - 0.01	0.00	0.00	0.00 - 0.00	0.00	0.00	0.00 - 0.02	0.00	0.01	0.00 - 0.07	0.01

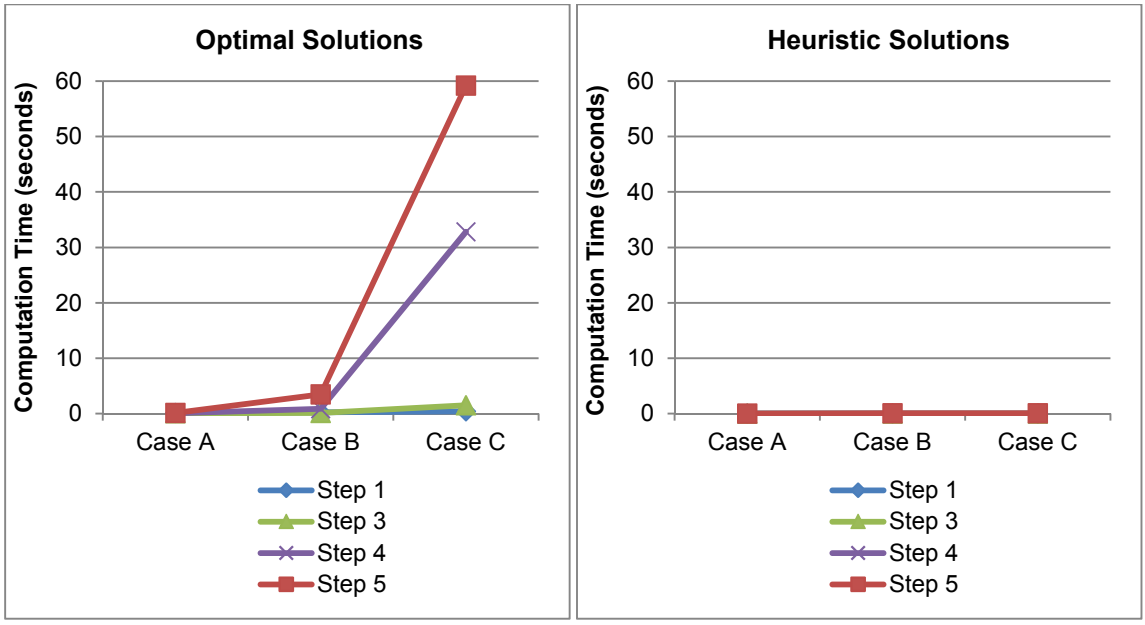


Figure 5. Computation Times (in seconds) of Obtaining the Optimal and Heuristic Solutions.

Table 30 shows computation times (in seconds) of obtaining the optimal and heuristic solutions in the four experiments using the second set of probability parameters. The experimental results using the second set of parameters are similar to the ones with the first set of parameters. Thus it has the same conclusion such that using the heuristics is much more scalable than running the optimization problem.

Table 30

Computation Times (in seconds) of Obtaining the Optimal and Heuristic Solutions with Parameter Set 2

		Step 1			Step 3			Step 4			Step 5		
		Avg.	Range	St.dev	Avg.	Range	St.dev	Avg.	Range	St.dev	Avg.	Range	St.dev
Optimal Solutions	Case A	0.09	0.02 - 0.42	0.08	0.04	0.02 - 0.20	0.03	0.06	0.02 - 0.48	0.04	0.09	0.03 - 1.41	0.12
	Case B	0.16	0.03 - 0.69	0.12	0.08	0.02 - 0.36	0.05	0.74	0.05 - 4.56	0.76	3.10	0.01 - 16.73	3.27
	Case C	0.85	0.03 - 85.33	6.49	2.58	0.03 - 118.72	13.05	27.23	0.06 - 127.22	42.86	56.41	0.19 - 161.76	61.56
Heuristic Solutions	Case A	0.00	0.00 - 0.00	0.00	0.00	0.00 - 0.00	0.00	0.00	0.00 - 0.00	0.00	0.00	0.00 - 0.00	0.00
	Case B	0.00	0.00 - 0.17	0.01	0.00	0.00 - 0.14	0.01	0.00	0.00 - 0.22	0.02	0.04	0.01 - 2.82	0.24
	Case C	0.00	0.00 - 0.00	0.00	0.00	0.00 - 0.00	0.00	0.00	0.00 - 0.02	0.00	0.01	0.00 - 0.06	0.01

2.8 Conclusions

Many heuristics are introduced for efficient resource allocation in cloud computing from the existing work, but more extensive research is required to develop effective heuristics that can achieve optimal or near-optimal solutions with great computational efficiency. This study in Chapter 2 starts with the analysis of the optimal solutions in resource allocation for a set of problem cases and thus proposes heuristics,

which can capture the decision making process from the optimal solutions for the various problem cases. Then the proposed heuristics are further tested for the extended problem cases with larger numbers of user requests and service providers for performance evaluation of the resource allocation heuristics. The heuristic solutions are compared with the optimal solutions in terms of solution optimality and scalability. Here, two key measures of the total number of dropped service requests and the *SAV* ratio for all served service requests are introduced in evaluating the solution optimality.

Experimental results show that the resource allocation decisions from the heuristic solutions are similar to the ones from the optimal solutions. In Case A, where each server has a sufficient resource capacity and in Case B, where all the servers together have a sufficient resource capacity for all service requests of all clients, the heuristic solutions do not drop any service request as same as the optimal solutions. For Case A and Case B, the heuristic solutions have as same *SAV* ratios as the optimal solutions, implying that the heuristic solutions select to serve all the service requests with the same QoS values as in the optimal solutions. For Case C, where all the servers together do not have a sufficient resource capacity for all service requests of all clients, the heuristic solutions drop more service requests of about 1% ~ 4% of all service requests than the optimal solutions. However, the heuristic solutions provide at least the same level of QoS to all served service requests as the optimal solutions with the same or higher *SAV* ratios.

The average computation times of obtaining the optimal solutions and the heuristic solutions are increased as a problem case becomes more complicated by larger numbers of service requests and service providers and having limited resource capacity to serve all service requests of all clients. The ranges of the computation times for obtaining

both the optimal and the heuristic solutions also become larger with higher values of the standard deviation. However, the rate of increase in the computation times with increasing problem complexity is much larger for solving the optimization problem than using the heuristics, especially for Case B and Case C with insufficient resource capacity on each server or all servers together to serve all service requests of all clients and for the problem cases with ten and twenty servers resulting in a larger solution space than the other problem cases. Hence, using the heuristics is much more scalable than solving the resource allocation optimization problem.

CHAPTER 3

THE ANALYSIS OF SERVICE PROVIDER-USER COORDINATION FROM CENTRALIZED ALGORITHM TO DEVELOP A DECENTRALIZED METHOD

A distributed cloud computing environment for IT services requires resource allocation in a decentralized manner through the coordination of service providers and service users. Achieving the optimal solution of resource allocation through the decentralized service provider-user coordination remains as a challenge. This study in Chapter 3 looks into elements of service provider-user coordination first in the formulation of the resource allocation problem in a centralized manner and then in the formulation of the problem in a decentralized manner for various problem cases. By examining differences between the centralized, optimal solutions and the decentralized solutions for those problem cases, the analysis of how the decentralized service provider-user coordination breaks down the optimal solutions is performed. Based on the analysis, strategies of decentralized service provider-user coordination are developed.

3.1 Literature Review

A lot of research work addressing the resource allocation problems in cloud computing generally fall into two types. The first type includes centralized algorithms, which assume to know important information such as requirements of all service requests of all clients and resource status of all service providers (e.g. resource availability, a current workload, etc.). With these information, centralized algorithms solve the optimization problem directly or reach to an optimal solution by coordinating the solutions of the problems. The second type includes decentralized algorithms, which try

to obtain near-optimal solutions by sharing information among service providers or between service providers and users without any global governance.

Many algorithms from linear programming to more complex ones such as Lyapunov optimization are centralized algorithms (Aoun, Doumith, & Gagnaire, 2010; Casati & Shan, 2001; Wang et al., 2006; Zeng et al., 2004). Some studies allow the centralized algorithms to have some levels of relaxations in the optimization problems so that using the algorithms enables to generate solutions in polynomial time. One study proposed to use utility-based service scheduling algorithms (Liu, Quan, & Ren, 2010). However, it resulted in violations of service users' requirements, imposing a penalty value. Ardagna and Pernici (2005, 2007) proposed to formulate the optimization problem in mixed integer linear programming with global constraints, which are obtained by running the local optimization first.

Other studies use the centralized algorithms, which have a central entity to collect all information about requirements of all service requests and resource status of all service providers. With the updated information, an optimal solution is achieved through communication or coordination among different agents. In (Haresh, Kalady, & Govindan, 2011), an agent based resource allocation method was proposed with involvement of three types of agents: Consumer Agent, Resource Brokering Agent and Resource Provider Agent. After obtaining a service request by the Consumer Agent, the Broker Agent assigned a grade to service providers based on the feedback from the consumers. Resource allocation decisions were made by the negotiation between the Broker Agent and the Resource Provider Agent. Similarly, the adaptive resource allocation model with different agents proposed in (Jung & Sim, 2011) discovered a proper data center based on

two evaluations: the geographical distance between a consumer and data centers and the workload of each data center.

Different types of modeling are proposed in many studies for resource allocation problems. An efficient bidding algorithm in combinatorial auction mechanisms used a service user's valuation function for VM instances within the user's budgets (Zaman & Grosu, 2011). Resource scheduling based on Pareto optimality theory was proposed to achieve equilibrium between maximizing service providers' profits and minimizing service users' payments (Li & Li, 2011). Yet, the performance of this algorithm was only proved mathematically. The cooperative resource allocation game in (Hassan, Song, & Huh, 2011) was based on game theory to determine the contribution of service providers to VM resources and used global objective functions to maximize social welfare or total utility of all service providers. Although the centralized algorithms enable to produce optimal solutions in resource allocation, it is difficult to handle real-time service requests due to much larger problem sizes. Moreover, it is not desirable to have a central authority to collect all the information in cloud computing environments where resources are physically distributed.

Hence, numerous studies have proposed to use decentralized algorithms for resource allocation problems. Various decentralized algorithms in several studies include integer linear programming modeling (Rezvani et al., 2015), construction of a hierarchical well-separated tree for matching critical events to available resources (Gao, Guibas, Milosavljevic, & Zhou, 2009), vector packing approaches on heterogeneous distributed platforms (Stillwell, Vivien, & Casanova, 2012), market-based approaches by setting prices for shared resources with market demand (Ercetin & Tassiulas, 2003; Wang

& Li, 2005) and utility-based price proportion methods for profit maximization of every cloud users and the cloud provider through competition among the cloud users (Mao, Shang, Liu, & Chen, 2013).

In some studies using decentralized algorithms, decomposition techniques are applied to the optimization problems in order to achieve near-optimal solutions with improvements in computation times of obtaining the solutions. For example, two studies (Alrifai & Risse, 2009; Alrifai, Skoutas, & Risse, 2010) used a MIP problem to find optimal decomposition of global QoS constraints into local constraints for local service selections in a distributed algorithm. Similarly, optimization decomposition methods were applied to divide global problems into less coupled subproblems (Heo, Jayachandran, Shin, Wang, & Abdelzaher, 2009; Smith, Chong, Maciejewski, & Siegel, 2012; Wendell, Jiang, Rexford, & Freedman, 2009).

Many literatures suggest information sharing among different agents for decentralized algorithms. Some work had local decisions, local data exchange (to share state information) and local interactions between a service entity and its neighbors (Manzalini & Moiso, 2011; Wuhib et al., 2010). However, it assumed that the maximum distance between two agents for direct communication was small compared with the size of the entire system. This assumption was a key factor to guarantee scalability of the proposed algorithms. Shiang and van der Schaar (2009) utilized network nodes to exchange information and further considered delays and cost of exchanging the network information. Other work proposes to use feedbacks from cloud users so as to help resource allocation decisions to service requests. Schlegel, Kowalczyk, and Vo (2008) selected a provider for job execution using feedbacks of previous job allocations. If the

feedback from the previous resource allocation decisions was positive, jobs were allocated to resources used before for similar jobs. If not, jobs were allocated to resources with the highest margin of free expected capacity over all required resource types. Similarly, Varalakshmi et al. (2013) suggested to evaluate trustworthiness when scheduling a job by monitoring resource and reputation. The trust computation was based on the feedbacks collected from users. Heo, Henriksson, Liu, and Abdelzaher (2007) presented a methodology to compose multiple performance management modules in order to reduce possible negative interactions and achieve good aggregate behavior. Rao, Bu, Xu, and Wang (2011) also proposed to evaluate requests submitted by individual VM providers and provide feedbacks. However, the solutions were affected by initial performance considerably.

From the literature reviews using decentralized algorithms, one common method is found for approaching to the efficient resource allocation solutions. Agents from different management levels seek for their own profits in generating resource allocation solutions, trying to achieve global optimality through communication among the agents. The agent-based decentralized algorithm was proposed with coordination among three levels of service management, workflow management and cloud management agents (Wei & Blake, 2013). A similar work was done in (Wei, Blake, & Saleh, 2013) by making resource allocation decisions based on predictive workload with coordination among different management agents. In (Wang & Fang, 2014), a distributed task scheduling model was constructed based on multi-agent coordination and interaction. The fuzzy pattern recognition method in (Wang & Su, 2015) was used to assign tasks to different levels according to their resource requirements, and each node determined the

corresponding task level according to its idle ability. With an arrival of a new task, only the nodes corresponding to the task level joined in the bid, and a successful node for a task was selected through the bidding scheme.

In a similar manner, the decentralized algorithm in Chapter 3 assumes to have two management levels of CCSPs and cloud users. Each service provider makes local resource allocation decisions for the given service requests of cloud users, while each cloud user ensures if the user's workflow of services is satisfied. The detail of which information is shared between service providers and cloud users and the formulation of a service provider's local optimization problem are described in Chapter 3.3.2.

3.2 Research Focus

In a distributed cloud environment involving multiple CCSPs, no one is in control of all physical resources. Hence, there is not a central authority (e.g., agent or broker) that has information of all CCSPs and makes decisions for all of them. Each CCSP may have its own objectives concerning its resource utilization, system performance, and user service satisfaction. CCSPs do not necessarily want to share information about its own objectives along with system resource state (e.g., failure and availability) that changes dynamically.

Without expecting the coordination among CCSPs and among cloud users, it is desirable to develop service provider-user coordination strategies for allocating resources of CCSPs to meet service requests of cloud users in a decentralized manner and achieve the service performance and satisfaction on the side of cloud users and the resource and system performance on the side of CCSPs. The service provider-user coordination does not mean the involvement of CCSPs and cloud users in the coordination. The service

provider-user coordination can be implemented through cloud service provider-user protocols in the cloud computing architecture with software agents for CCSPs and cloud users.

The decentralized resource allocation algorithms through service provider-user coordination have not been well addressed in existing work on cloud computing and traditional distributed computing infrastructures. A major challenge is to produce the decentralized solution as good as or close to the optimal solution that is obtained by solving the centralized problem with all information available and all decisions made in one place. Therefore, it is essential to investigate and identify what elements are necessary in a scalable, distributed algorithm to produce optimal or near-optimal solutions for a set of representative problem cases. This study in Chapter 3 aims at identifying elements of service provider-user coordination that can lead a scalable, distributed algorithm to the optimal or near-optimal solution.

To achieve this research goal, it starts with a simple service provider-user coordination protocol in a scalable, distributed algorithm. By examining differences between the centralized, optimal solutions and the decentralized solutions for various problem cases involving various types of a service provider's configuration and different numbers of service requests, this study analyzes how the decentralized service provider-user coordination breaks down the optimal solutions and, as a result, suggests key elements of the decentralized service provider-user coordination strategies.

3.3 Research Methodology

In this section, the modified formulation of the centralized optimization problem and a decentralized service provider-user coordination strategy are presented to use for

obtaining resource allocation solutions at cloud computing environments. The resource allocation problem in this study handles different workflows of services, both functional (e.g., service type) and non-functional (e.g., QoS) service requirements, and resource and QoS impact models of services. Note that two methods in Chapter 3 consider only one service or sequential processing of a service workflow. One example using the sequential workflow is an encrypted voice data service. First, a voice communication service is used to obtain voice data, and then a data encryption service is used to encrypt the voice data. The encrypted voice data service has such workflow constraints that the throughput of voice communication should not exceed the limit on the input throughput for data encryption.

3.3.1 The modified formulation of the centralized resource allocation problem. The resource allocation optimization problem expressed as a MIP in Chapter 2.3 is modified with an introduction of a workflow of services in this study. The optimization problem is solved for each epoch of dynamic resource allocation. Note that a service provider may have one or more servers and that a service user may generate one or more clients. Each client may request one service or a workflow of services. Hence, in the following formulation of the centralized resource allocation problem, the terms of server and client are used. Table 31 and Table 32 indicate variables and indices, and decision variables and given inputs used in the formulation respectively.

Table 31

Variables and Indices used in the Modified Formulation of the Centralized Resource Allocation Problem

k	A given client, $k = 1,\ldots, K$
j_k	j^{th} service of client k, $j_k = 1,\ldots, J_k$
i	A given server, $i = 1,\ldots, I$
w_i	Resource variable w of server i, $w_i = 1,\ldots, W_i$
s	A service type, $s = 1,\ldots, S$
d_s	Service parameter d of service s, $d_s = 1,\ldots, D_s$
p_s	QoS variable p of service s, $p_s = 1,\ldots, P_s$
$R_{kj_kiw_i}$	The amount of resource variable w of server i taken by client k's j_k^{th} service as a positive real value
$Q_{kj_kp_si}$	The value of QoS variable p_s of client k's j_k^{th} service on server i as a positive real value

Table 32

Decision Variables and Given Inputs used in the Modified Formulation of the Centralized Resource Allocation Problem

X_{kj_ki}	Binary decision variables such that $X_{kj_ki} = 1$ if client k's j_k^{th} service in the workflow is assigned to server i $X_{kj_ki} = 0$ if client k's j_k^{th} service in the workflow is not assigned to server i
$A_{kj_kd_si}$	Positive integer decision variables, Level of service parameter d_s for client k's j_k^{th} service on server i
U_{kj_ks}	Given inputs from client k, such that $\sum_s U_{kj_ks} = 1$ for a given k and a given j_k. $U_{kj_ks} = 1$ if client k's j_k^{th} service in the workflow uses service s $U_{kj_ks} = 0$ if client k's j_k^{th} service in the workflow does not service s
V_{is}	Given inputs from server i, $V_{is} = 1$ if service s is provided by server i $V_{is} = 0$ if service s is not provided by server i
$A^l_{kj_kd_si}$	Given inputs as a positive integer value from client k, Limit (i.e. the maximum level) of service parameter d_s of client k's j_k^{th} service on server i
$R^l_{iw_i}$	Given inputs as a positive real value from server i to indicate the resource capacity, Limit of resource variable w of server i (this limit is set for only some w_i's)
$Q^l_{kj_kp_s}$	Given inputs as a positive real value from client k to specify QoS requirements, Limit of QoS variable p_s of client k's j_k^{th} service

The modified formulation of the centralized resource allocation problem is as follows.

$$\text{Maximize } M \sum_k \frac{\sum_{j_k} \sum_i X_{kj_ki}}{J_k} - \sum_k \sum_{j_k} \sum_{p_s} \frac{\left|\sum_i Q_{kj_kp_si} - Q^l_{kj_kp_s}\right|}{Q^l_{kj_kp_s} * P_s} \tag{14}$$

subject to

$$\sum_i X_{kj_ki} \leq 1 \quad \forall\, k, j_k \tag{15}$$

$$\left(\sum_{j_k}\sum_s U_{kj_ks} - 1\right)\left(\sum_i X_{kj_ki} - \sum_i X_{kj'_ki}\right) = 0 \qquad \forall\, k,\, j_k \neq j'_k \tag{16}$$

$$X_{kj_ki}U_{kj_ks} \leq V_{is} \qquad \forall\, k, j_k, i, s \tag{17}$$

$$A_{kj_kd_si} \leq A^l_{kj_kd_si} \qquad \forall\, i, k, j_k, d_s \tag{18}$$

$$R_{kj_kiw_i} = X_{kj_ki}F_{iw_i}\left(A_{kj_k1i}, \ldots, A_{kj_kD_si}\right) \qquad \forall\, i, k, j_k, w_i \tag{19}$$

$$Q_{kj_kp_si} = X_{kj_ki}G_{ip_s}\left(R_{kj_ki1}, \ldots, R_{kj_kiW_i}\right) \qquad \forall\, i, k, j_k, p_s \tag{20}$$

$$\sum_k\sum_{j_k} R_{kj_kiw_i} \leq R^l_{iw_i} \qquad \forall\, i, w_i \tag{21}$$

$$Q_{kj_kp_si}X_{kj_ki} \leq Q^l_{kj_kp_s} \text{ or } Q_{kj_kp_si} \geq X_{kj_ki}Q^l_{kj_kp_s} \qquad \forall\, i, k, j_k, p_s \tag{22}$$

The first term of the objective function in Equation (14) represents the percentage of satisfied services over all clients' workflows. The second term of the objective function is to make the levels of the QoS variables closest to the QoS requirements. In the second term, the difference between the actual QoS level ($Q_{kj_kp_si}$) and the required QoS level ($Q^l_{kj_kp_s}$) for each QoS variable (p_s) is first normalized by the required QoS level, then summed and normalized over the total number of QoS variables, finally summed over all services and all clients. The normalization in the second term makes all QoS variables to be treated equally. For example, if a motion detection service has two QoS variables, both QoS variables in total are treated equally to only one QoS variable of another service. Here *M* is a positive value to give a tradeoff between two objectives defined by the first and second terms of the objective function.

As server-client coordination constraints, Equation (15) guarantees that client *k*'s j_k^{th} service can be assigned to one server *i* at most, and Equation (16) ensures if client *k* has more than one service in the workflow, client *k*'s different services j_k, j_k' must be

either all served or all not served. As service constraints, Equation (17) requires if client k's j_k^{th} service is assigned to server i, client k's j_k^{th} service's service type s must be provided by server i (i.e. if $X_{kj_ki} = 1$ and $U_{kj_ks} = 1$, then $V_{is} = 1$). Equation (18) enforces that the level of service parameter d_s of client k's j_k^{th} service on server i should not exceed the limit (i.e. the maximum level).

As service-resource-QoS relation constraints, Equation (19) gives relations of service parameters with resource usages in function F_{iw_i} of the assigned level of service parameter d_s of client k's j_k^{th} service on the server only if client k's j_k^{th} service is assigned to the server i. Equation (20) gives the relation of resource usages with QoS performance in function G_{ip_s} of the service's resource usages on the server only if client k's j_k^{th} service is assigned to the server i.

As a resource capacity constraint, Equation (21) enforces that the total resource usages on the resource variable w of server i by all clients' all services should not exceed the maximum resource capacity for this resource variable. As a QoS requirement constraint, Equation (22) ensures that the QoS level of p_s for the client k's j_k^{th} service at server i is equal to or less than the maximum QoS requirement or equal to or greater than the minimum QoS requirement, only if client k's j_k^{th} service is assigned to server i.

Workflow constraints of applications that describe the dependency among services in each application should be satisfied (Berman, 1999; Yau et al., 2009; Ye et al., 2010). For example, Berman (1999) described program models by a weighted data-flow-style program graph or by a set of program characteristics which may or may not include a structural task dependency graph. Lin and Lu (2011) also represented a workflow using a weighted directed acyclic graph. In the graph, vertices represented a set of tasks, and

edges represented a set of data dependencies. The weight of an edge denoted the communication cost, and the weight of a vertex denoted the task computation cost. For workflow constraints considered in this study, specific forms depend on specific services in a workflow and thus are not given here in a general form.

The MIP problem is implemented in ILOG OPL Development Studio IDE Version 6.1. ILOG CPLEX 11.2.0 is used as a solver to the MIP problem. A laptop computer used to run the software is a Samsung Q320 with Intel Core 2 Duo T6500 2.1 GHz processor, 4 GB RAM, and Windows 7. The ILOG OPL and CPLEX are integrated into C# code in Microsoft Visual Studio 2010. The C# code first loads all the necessary input files of the problem including given inputs as well as service-resource relation functions and resource-QoS relation functions for each service type. With the loaded input files, the C# code then calls ILOG OPL and CPLEX to run the MIP optimization and solve the problem to generate an optimal solution. The computation time of obtaining the optimal solution is recorded by the C# code. Note that times required for loading input files and generating output files are also included in the computation time of obtaining the optimal solution.

3.3.2 A decentralized strategy of service provider-user coordination. In this section, a decentralized strategy of service provider-user coordination is constructed. Considering again that a service provider may have one or more servers and a service user may have one or more clients, the terms of server and client are used in the following description of the service provider-user coordination strategy. In this strategy, each client sends the request for each unsatisfied service in the workflow to all servers that provide the service. Each server makes local resource allocation decisions, and

clients coordinate with servers in one or more iterations to obtain a resource allocation solution. The service provider-user coordination strategy consists of the following steps for client *k, k* = 1,…, *K*.

1) If client *k*'s workflow of services is not satisfied, the client sends the request for each service in the workflow to all servers which provide the service. If any service of the client *k*'s workflow is satisfied, the client *k* sends the input of $X_{kj_ki} = 1$ for the satisfied service in the workflow to the server that provides the service so that resources on the server are reserved for the service.
2) Each server *i* solves a local optimization problem and sends the solution, i.e., decisions about X_{kj_ki} and $A_{kj_kd_si}$ along with $Q_{kj_kp_si}$ to client *k*.
3) When client *k* receives the information of X_{kj_ki}, $A_{kj_kd_si}$ and $Q_{kj_kp_si}$ from all servers *i* = 1,…, *I,* the client checks the satisfaction of each service in the workflow and workflow constraints. If more than one server selects to satisfy a service, the client picks the server that gives the QoS levels closest to the QoS requirements. If there are unsatisfied services in the workflow or unsatisfied workflow constraints, the client marks the workflow of services as unsatisfied and goes back to Step 1 for another iteration; otherwise, the client obtains a complete solution satisfying its workflow of services.

Each server *i*, *i* = 1,…, *I,* solves the following local optimization problem.

$$\text{Maximize } M \sum_k \frac{\sum_{j_k} X_{kj_ki}}{\sum_{j_k} \sum_s U_{kj_ks} V_{is}} - \sum_k \sum_{j_k} \sum_{p_s} \frac{\left| Q_{kj_kp_si} - Q^l_{kj_kp_s} \right|}{Q^l_{kj_kp_s} * P_s} \tag{23}$$

subject to

$X_{kj_ki} = 1$ for satisfied services (24)

$X_{kj_ki}U_{kj_ks} \leq V_{is} \qquad \forall\, k, j_k, s$ (25)

$A_{kj_kd_si} \leq A^l_{kj_kd_si} \qquad \forall\, k, j_k, d_s$ (26)

$R_{kj_kiw_i} = X_{kj_ki}F_{iw_i}\left(A_{kj_k1i}, \ldots, A_{kj_kD_si}\right) \qquad \forall\, k, j_k, w_i$ (27)

$Q_{kj_kp_si} = X_{kj_ki}G_{ip_s}\left(R_{kj_ki1}, \ldots, R_{kj_kiW_i}\right) \qquad \forall\, k, j_k, p_s$ (28)

$\sum_k \sum_{j_k} R_{kj_kiw_i} \leq R^l_{iw_i} \qquad \forall\, w_i$ (29)

$Q_{kj_kp_si}X_{kj_ki} \leq Q^l_{kj_kp_s}$ or $Q_{kj_kp_si} \geq X_{kj_ki}Q^l_{kj_kp_s} \qquad \forall\, k, j_k, p_s$ (30)

The objective function in Equation (23) maximizes the percentage of provided services requested and satisfied for clients on server *i* in the first term and makes the QoS levels of each service for each client closest to the required QoS levels in the second term. The two server-client coordination constraints in Equations (15) and (16) of the centralized formation in Chapter 3.3.1 no longer exist in this local optimization problem solved by each server because each individual server cannot take care of the server-client coordination constraints that require information of all servers and all clients. Instead, the constraint in Equation (24) is included to carry the partial solution(s) of satisfying services of some but not all clients from previous iteration(s) to the current iteration because the decentralized strategy may need several iterations to produce a complete solution of resource allocation for all clients. The constraints in Equations (25) through (30) also exist in the centralized formulation.

The C# code is implemented to execute the decentralized strategy of service provider-user coordination. ILOG OPL and CPLEX are used to solve the local optimization problem at each server with the loaded input files. The solution process is

stopped when workflows of all clients are satisfied or the solution from the current iteration is the same as that from the previous iteration. The computation time of obtaining the optimal solution at each server in each iteration is recorded by the C# code. The computation time of obtaining a decentralized solution is computed as the sum of computation times for all solution iterations. The computation time for each iteration is the maximum of computation times used by individual servers to obtain their solutions. It is assumed that each client takes no time to make decisions after receiving solutions from servers. Note that times required for loading input files and generating output files are also included in the computation time of obtaining a solution.

3.4 Description of Experimental Scenarios

Resource and QoS impact models of various services including voice communication, data encryption and motion detection are investigated and established in (Yau et al., 2009; Ye et al., 2010). In a voice communication service, a client requests and receives voice data from a server. The data encryption service encrypts data using an encryption algorithm. In a motion detection service, video data is analyzed to detect motion. Since such resource and QoS impact models of services are needed for Equations (19) and (20) in the centralized method and Equations (27) and (28) in the decentralized algorithm, this study in Chapter 3 uses those three services, and the resource and QoS impact models of the three services are used in Equations (19), (20), (27) and (28).

The voice communication service has two service parameters: sampling rate and buffer size. The sampling rate is the rate of sampling voice data and determines the quality of the sampled voice data. The sampling rate can take one of the five levels: 1 for 44,100 Hz, 2 for 88,200 Hz, 3 for 132,300 Hz, 4 for 176,400 Hz, and 5 for 220,500 Hz.

The buffer size is the size of the buffer holding the sampled voice data at a server before transmission. The buffer size can take one of the five levels: 1 for 16,384 Bytes, 2 for 24,576 Bytes, 3 for 32,768 Bytes, 4 for 40,960 Bytes, and 5 for 49,152 Bytes. This study lets each client using the voice communication service set the maximum level ($A^{l}_{kj_k d_s i}$) of the sampling rate to 5 and the maximum level of the buffer size to 5.

The data encryption service has two service parameters: key length and encryption percentage. The key length is the size of the key used for encryption. The key length can take one of the three levels: 1 for 128 bits, 2 for 196 bits, and 3 for 256 bits. The encryption percentage is the percentage of data for encryption. The encryption percentage can take one of the two levels: 1 for 50%, and 2 for 100%. A larger key length and a larger encryption percentage ensure a better security of protecting data confidentiality. This study lets each client using the data encryption service set the maximum level of the key length to 3 and the maximum level of the buffer size to 2.

The motion detection has one service parameter: video resolution. More motions can be detected from a video frame with a higher resolution, resulting in a higher motion level of detection but more computational resources used for detecting motions. The video resolution can take one of the three levels: 1 for 22*18 pixels, 2 for 44*36 pixels, and 3 for 88*72 pixels. This study lets each client using the motion detection service set the maximum level of the video resolution to 3. Table 33 summarizes levels of service parameters for voice communication, data encryption and motion detection services.

Table 33

Levels of Service Parameters for Voice Communication, Data Encryption and Motion Detection Services

Service Type	Service Parameters	Level 1	Level 2	Level 3	Level 4	Level 5
Voice Communication	Sampling rate (Hz)	44,100	88,200	132,300	176,400	220,500
	Buffer size (Bytes)	16,384	24,576	32,768	40,960	49,152
Data Encryption	Key Length (bits)	128	192	256		
	Encryption Percentage (%)	50	100			
Motion Detection	Video resolution (pixels)	22*18	44*36	88*72		

In this study, there is one QoS variable considered for the voice communication service: throughput in unit of packets/second; one QoS variable for the data encryption service: average delay in unit of milliseconds; and two QoS variables for the motion detection service: average motion level in terms of the percentage of pixels in a video frame that have detected changes, and average delay in unit of milliseconds.

For the three services used in this study, the following variables of resource usages identified in (Yau et al., 2009; Ye et al., 2010) play a key role in determining the QoS performance of the services:

1) Processor time in percentage
2) Committed memory in megabytes (MB),
3) Thread count,
4) IO other operations/sec,
5) IO read operations/sec,

6) IO write operations/sec,

7) File write operations/sec,

8) File control operations/sec,

9) System calls/sec.

The specific *F* functions of relations between service parameters and resource variables are used in Equations (19) and (27), and the specific *G* functions of relations between resource variables and QoS variables are used in Equations (20) and (28) for each of the three services. For example of a voice communication service, Equations (31) through (34) show the specific *F* functions for the resource variables of processor time in percentage, committed memory in MB, thread count and IO other operations/sec respectively. Equation (35) shows the specific *G* function for the QoS variable of the throughput.

$$R_{11i1} = X_{11i}\left[\frac{0.8794\cdot\sum_k\sum_{j_k}U_{kj_k1}+1.03\cdot A_{111i}-0.338\cdot A_{112i}}{\sum_k\sum_{j_k}U_{kj_k1}}\right] \quad (31)$$

$$R_{11i2} = X_{11i}\left[\frac{429.09+15.1907\cdot\sum_k\sum_{j_k}U_{kj_k1}+11.747\cdot A_{111i}+0.4261\cdot A_{112i}}{\sum_k\sum_{j_k}U_{kj_k1}}\right] \quad (32)$$

$$R_{11i3} = X_{11i}\left[\frac{14.79+4.21\cdot\sum_k\sum_{j_k}U_{kj_k1}+0.846\cdot A_{111i}-0.1211\cdot A_{112i}}{\sum_k\sum_{j_k}U_{kj_k1}}\right] \quad (33)$$

$$R_{11i4} = X_{11i}\left[\frac{-19.8+37.4\cdot\sum_k\sum_{j_k}U_{kj_k1}+37.6\cdot A_{111i}-30.9\cdot A_{112i}}{\sum_k\sum_{j_k}U_{kj_k1}}\right] \quad (34)$$

$$Q_{111i} = X_{11i}\left[-\frac{7544.65}{\sum_k\sum_{j_k}U_{kj_k1}} + 174.76 * R_{11i1} + 16.6 * R_{11i2} - 7.86 * R_{11i3} - 0.06 * R_{11i4}\right] \quad (35)$$

Three services are used in two types of service workflow. In type 1 of service workflow, a client requests and receives encrypted voice data by using first a voice

communication service to obtain voice data and then a data encryption service to encrypt the voice data. Hence, type 1 of service workflow involves two services in order of voice communication first and data encryption second. Workflow constraints specify that, for a given client, the throughput of voice communication should not exceed the limit on the input throughput for data encryption. Type 2 of service workflow involves only one service of motion detection.

This study introduces six types of server configurations which are determined by two factors: resource capacity and service provision. There are three types of resource capacity:

- Each server has a sufficient resource capacity to satisfy all service requests of all clients,
- Each server does not have a sufficient resource capacity to satisfy all service requests of all clients, but the total resource capacity of all servers is sufficient to satisfy all service requests of all clients,
- Neither each server nor all servers together have a sufficient resource capacity to satisfy all service requests of all clients.

This study uses two servers with two types of service provision:

- Each server provides all three services of voice communication, data encryption, and motion detection.
- Server 1 provides two services of voice communication and motion detection, and server 2 provides two services of data encryption and motion detection.

The same set of *F* and *G* functions are used on each server.

This study introduces various numbers of clients to give various problem sizes in order to examine how the computation times of obtaining the centralized solution and the decentralized solution change with problem sizes, specifically the number of clients. There are three levels of clients used in this study: four clients, fifty clients and one hundred clients. For four clients, there are two clients that use type 1 of service workflow with the voice communication and data encryption services, while the other two clients use type 2 of service workflow with the motion detection service. For fifty clients, twenty five clients use type 1 of service workflow and the other twenty five clients use type 2 of service workflow. For one hundred clients, fifty clients use type 1 of service workflow, and the other fifty clients use type 2 of service workflow.

Among the nine resource variables, the first two resource variables, processor time in percentage and committed memory in MB have a capacity limit which is used in Equations (21) and (29). Values of capacity limits for each problem case are set up to maintain the type of server configuration for that problem case by looking into resource usages of clients under various values of service parameters based on the F functions of service-resource relations. Table 34 shows the resource capacity limits used in various problem cases.

Table 34

Capacity Limits of Two Resource Variables: Processor Time (%) and Committed Memory (MB)

Server Configuration	Server	Number of Clients		
		4	50	100
1. Type 1 of Resource Capacity Type 1 of Service Provision	Server 1 Server 2	65, 750 65, 750	470 , 1350 470 , 1350	855, 2010 855, 2010
2. Type 1 of Resource Capacity Type 2 of Service Provision	Server 1 Server 2	6, 520 47, 210	375, 1350 40, 100	400, 1650 430, 1650
3. Type 2 of Resource Capacity Type 1 of Service Provision	Server 1 Server 2	31, 400 35, 400	30, 900 405, 500	125, 1210 720, 810
4. Type 2 of Resource Capacity Type 2 of Service Provision	Server 1 Server 2	6, 520 47, 210	375, 1350 40, 100	400, 1650 430, 450
5. Type 3 of Resource Capacity Type 1 of Service Provision	Server 1 Server 2	20, 400 30, 350	61, 825 190, 260	120, 1210 360, 420
6. Type 3 of Resource Capacity Type 2 of Service Provision	Server 1 Server 2	32, 470 10, 100	210, 1090 40, 100	45, 1205 430, 405

For example, in the problem case with type 1 of resource capacity, type 1 of service provision and four clients in Table 34, server 1 has 65% as the capacity limit of processor time and 750 MB of committed memory. When the number of clients increases to fifty, capacity limits are set to 470% of processor time and 1350 MB of committed memory. Considering that the maximum capacity of processor time is 100% in the real-world situation, it may not seem reasonable to set 470% of processor time. This value is set so that the problem case can still maintain type 1 of resource capacity in which each server has a sufficient resource capacity to satisfy service requests of all fifty clients. Hence, although 470% for processor time in Table 34 is unrealistic, mathematically the

value allows to examine the problem of the same nature but with a different size for testing how computation times change with the problem sizes.

QoS requirements ($Q^{l}_{kj_k p_s}$) for each problem case are set up to ensure that an optimal solution for the centralized formulation exists to meet both the resource capacity constraints in Table 34 and QoS requirements of clients by looking into resource usages of clients and QoS performance levels under various values of service parameters based on the F functions of service-resource relations and the G function of resource-QoS relations. Table 35 shows QoS requirements of problem cases with totally four, fifty and one hundred clients.

Table 35

QoS Requirements of Various Problem Cases

Total Number of Clients	Type 1 of Service Workflow			Type 2 of Service Workflow		
	Number of Clients	Voice Communication, Throughput (packets/second)	Data Encryption, Average Delay (milliseconds)	Number of Clients	Motion Detection, Average Motion Level (percentages)	Average Delay (milliseconds)
4	1	850	20	1	0.2	150
	1	950	35	1	0.15	90
50	12	200	10	12	0.01	450
	13	250	15	13	0.03	500
100	25	200	10	25	0.01	450
	25	250	15	25	0.03	500

For the problem cases with totally four clients in Table 35, there are two clients using type 1 of service workflow and the other two clients using type 2 of service workflow. For one of the two clients using type 1 of service workflow, the minimum requirement of the QoS variable for the throughput of the voice communication service is

set to 850 packets/second, and the maximum limit of the QoS variable for the average delay of the data encryption service is set to 20 milliseconds. For another of the two clients using type 1 of service workflow, the minimum limit of the QoS variable for the throughput of the voice communication service is set to 950 packets/second, and the maximum limit of the QoS variable for the average delay of the data encryption service is set to 35 milliseconds. For one of the two clients using type 2 of service workflow, the minimum limit of the QoS variable for the average motion level of the motion detection service is set to 0.2%, and the maximum limit of the QoS variable for the average delay of the motion detection service is set to 150 milliseconds. For another of the two clients using type 2 of service workflow, the minimum limit of the QoS variable for the average motion level of the motion detection service is set to 0.15%, and the maximum limit of the QoS variable for the average delay of the motion detection service is set to 90 milliseconds.

For the problem cases with totally fifty clients in Table 35, twenty five clients use type 1 of service workflow, and the other twenty five clients use type 2 of service workflow. For twenty five clients with type 1 of service workflow, there are twelve clients that have the minimum throughput requirement of 200 packets/second and the maximum delay requirement of 10 milliseconds, and there are the other thirteen clients that have the minimum throughput requirement of 250 packets/second and the maximum delay requirement of 15 milliseconds. For twenty five clients with type 2 of service workflow, there are twelve clients that have the minimum motion level requirement of 0.01% and the maximum delay requirement of 450 milliseconds, and there are the other

thirteen clients that have the minimum motion level requirement of 0.03% and the maximum delay requirement of 500 milliseconds.

For the problem cases with totally one hundred clients in Table 35, fifty clients use type 1 of service workflow, and the other fifty clients use type 2 of service workflow. For fifty clients with type 1 of service workflow, there are twenty five clients that have the minimum throughput requirement of 200 packets/second and the maximum delay requirement of 10 milliseconds, and there are the other twenty five clients that have the minimum throughput requirement of 250 packets/second and the maximum delay requirement of 15 milliseconds. For fifty clients with type 2 of service workflow, there are twenty five clients that have the minimum motion level requirement of 0.01% and the maximum delay requirement of 450 milliseconds, and there are the other twenty five clients that have the minimum motion level requirement of 0.03% and the maximum delay requirement of 500 milliseconds.

The objective functions in Equations (14) and (23) have two terms, the first term to maximize the percentage of clients' services satisfied, and the second term to minimize the difference between the provided QoS levels and the QoS requirements for the satisfaction of the QoS requirements. The M value of 10 is used in the objective functions for all problem cases in this study. The specific value of M is set up in order to give more importance in providing satisfied services for all clients' requests first before considering how close the actual QoS is to the required QoS. The normalized differences between the actual QoS level and the required QoS level in the second term of the objective functions depend on the specific service-resource-QoS relation models of a given service in Equations (19), (20), (27) and (28) since the actual QoS level is determined by those

models. Based on the service-resource-QoS models of the three services used in this study, the ranges of the normalized differences between possible QoS levels and required QoS levels are from 0 to 9.6. Hence, M is set to 10, which is higher than 9.6 so as to give a higher priority to the first term in the objective functions.

3.5 Results and Discussions

This section first gives and compares the centralized solutions and the decentralized solutions for various problem cases to examine the solution optimality of the decentralized solutions and discusses elements of service provider-user coordination based on the comparison results. Then it provides the computation times of obtaining the centralized and decentralized solutions to examine the scalability of the centralized and the decentralized methods.

3.5.1 Solution optimality. The objective functions as shown in Equations (14) and (23) in the resource allocation problem formulations have two terms, the first term to maximize the percentage of clients' services satisfied, and the second term to minimize the difference between the provided QoS levels and the QoS requirements for the satisfaction of the QoS requirements. Table 36 gives the number of clients satisfied in the centralized solutions and the decentralized solutions. Table 37 gives the values of the objective function in the centralized solutions and the decentralized solutions.

Table 36

Number of Clients Satisfied in the Centralized and Decentralized Solutions

Server Configuration	4 Clients		50 Clients		100 Clients	
	Centralized	Decentralized	Centralized	Decentralized	Centralized	Decentralized
1	4	4	50	50	100	100
2	4	4	50	50	100	100
3	4	4	50	50	100	100
4	4	4	50	50	100	100
5	3	3	38	38	75	75
6	3	3	38	38	75	75

For the number of clients satisfied, the decentralized solutions are as good as the optimal solutions from the centralized problem formulation as shown in Table 36. For the values of the objective function that need to be maximized, the decentralized solutions are close to the centralized solutions with slightly different values in some cases as marked by "*" in Table 37, including two out of the six problem cases for four clients, one out of six problem cases for fifty clients and two out of the six problem cases for one hundred clients.

Table 37

Values of the Objective Function in the Centralized and Decentralized Solutions

Server Configuration	4 Clients		50 Clients		100 Clients	
	Centralized	Decentralized	Centralized	Decentralized	Centralized	Decentralized
1	39.10	39.10	447.86	447.86	900.76	900.76
2	38.57	38.57	447.67	447.67	900.22	900.22
3	39.10	38.61*	447.86	447.86	900.62	900.42*
4	38.57	38.57	447.67	447.67	900.22	900.22
5	28.48	27.14*	341.11	340.92*	674.16	674.04*
6	27.15	27.15	340.93	340.93	673.97	673.97

To look into causes for the differences of the decentralized solutions from the centralized solutions, Table 38 shows service decisions made in the centralized and decentralized solutions for two problem cases (server configurations 3 and 5) with four clients. Server configuration 3 has type 2 of resource capacity and type 1 of service provision. That is, there is not a sufficient resource capacity in one server but there is a sufficient resource capacity on all the servers together to satisfy all service requests of all clients, and all three services of voice communication, data encryption and motion detection are provided on each server. Server configuration 5 has type 3 of resource capacity and type 1 of service provision. That is, neither each server nor all the servers have enough resource capacity to satisfy all service requests of all clients, and all three services of voice communication, data encryption and motion detection are provided on each server. Different decisions made in the decentralized solutions from those in the centralized solutions are marked by "*" in Table 38.

Table 38

Service Decisions made in the Centralized and Decentralized Solutions for Two Problem Cases

Problem Case	Client	Service	QoS Requirement	Centralized Solution		Decentralized Solution	
				Server	Service Parameter and QoS Level	Server	Service Parameter and QoS Level
server conf. 3	$k = 1$	voice com.	throughput: 850	server 2	sam. rate: 5 buf. size: 5 throughput: 894.38	server 2	sam. rate: 5 buf. size: 5 throughput: 894.38
		data enc.	delay: 20	server 1	key len.: 3 enc. per.: 1 delay: 16	server 1	key len.: 3 enc. per.: 1 delay: 16
	$k = 2$	voice com.	throughput: 950	server 1	sam. rate: 5 buf. size: 2 throughput: 968.16	server 1	sam. rate: 5 buf. size: 2 throughput: 968.16
		data enc.	delay: 35	server 2	key len.: 3 enc. per.: 2 delay: 33.87	server 2	key len.: 2* enc. per.: 1* delay: 16.72*
	$k = 3$	motion det.	mot. lev.: 0.2% delay: 150	server 1	vid. res.: 3 mot. lev.: 0.34% delay: 134.08	server 2*	vid. res.: 3 mot. lev.: 0.34% delay: 134.08
	$k = 4$	motion det.	mot. lev.: 0.15% delay: 90	server 2	vid. res.: 2 mot. lev.: 0.19% delay: 78.87	server 1*	vid. res.: 2 mot. lev.: 0.19% delay: 78.87
server conf. 5	$k = 1$	voice com.	throughput: 850	server 2	sam. rate: 5 buf. size: 5 throughput: 894.38	server 1*	sam. rate: 5 buf. size: 5 throughput: 894.38
		data enc.	delay: 20	server 2	key len.: 2 enc. per.: 1 delay: 15.75	server 2	key len.: 2 enc. per.: 1 delay: 15.75
	$k = 2$	voice com.	throughput: 950	server 1	sam. rate: 5 buf. size: 2 throughput: 968.16	none*	none*
		data enc.	delay: 35	server 2	key len.: 1 enc. per.: 2 delay: 33.29	none*	none*
	$k = 3$	motion det.	mot. lev.: 0.2% delay: 150	none	none	server 1*	vid. res.: 3* mot. lev.: 0.34%* delay: 134.08*
	$k = 4$	motion det.	mot. lev.: 0.15% delay: 90	server 1	vid. res.: 2 mot. lev.: 0.19% delay: 78.87	server 2*	vid. res.: 2 mot. lev.: 0.19% delay: 78.87

For the problem case with server configuration 3 in Table 38, the decentralized solution differs from the centralized solution in the levels of the service parameters for the data encryption service of client $k = 2$ in that the decentralized solution selects a lower level of the key length at level 2 and the encryption percentage at level 1 instead of the key length at level 3 and the encryption percentage at level 2 in the centralized solution. The decentralized algorithm produces this solution after two iterations. Table 39 shows service decisions made at two iterations in the decentralized solution for the problem case with server configuration 3. The selected services of clients at each iteration are marked by "*" in Table 39.

Table 39

Service Decisions made at Two Iterations in the Decentralized Solution for One Problem Case with Server Configuration 3

Client	Service	QoS Requirement	Iteration 1		Iteration 2	
			Server 1	Server 2	Server 1	Server 2
$k = 1$	voice com.	throughput: 850				sam. rate: 5* buf. size: 5* throughput: 894.38*
	data enc.	delay: 20	key len.: 3 enc. per.: 1 delay: 16	key len.: 3 enc. per.: 1 delay: 16	key len.: 3* enc. per.: 1* delay: 16*	
$k = 2$	voice com.	throughput: 950	sam. rate: 5* buf. size: 2* throughput: 968.16*	sam. rate: 5 buf. size: 2 throughput: 968.16		
	data enc.	delay: 35		key len.: 2* enc. per.: 1* delay: 16.72*		
$k = 3$	motion det.	mot. lev.: 0.2% delay: 150				vid. res.: 3* mot. lev.: 0.34%* delay: 134.08*
$k = 4$	motion det.	mot. lev.: 0.15% delay: 90	vid. res.: 2* mot. lev.: 0.19%* delay: 78.87*	vid. res.: 2 mot. lev.: 0.19% delay: 78.87		

At iteration 1, server 1 selects to serve client 1's data encryption service at the key length of level 3 and the encryption percentage of level 1, and the delay of 16 milliseconds, client 2's voice communication service at the sampling rate of level 5 and the buffer size of level 2, and the throughput of 968.16 packets/second, and client 4's

motion detection service at the video resolution of level 2, and the motion level of 0.19% and the delay of 78.87 milliseconds. Server 2 selects to serve client 1's data encryption service at the key length of level 3, the encryption percentage of level 1, and the delay of 16 milliseconds, client 2's voice communication service at the sampling rate of level 5, the buffer size of level 2, the throughput of 968.16 packets/second and data encryption service at the key length of level 2, the encryption percentage of level 1, and the delay of 16.72 milliseconds, and client 4's motion detection service at the video resolution of level 2, the motion level of 0.19%, and the delay of 78.87 milliseconds.

With these server solutions, each client makes the decisions as follows. Client 1 marks the workflow including the voice communication service and the data encryption service as unsatisfied since the voice communication service is not selected by either server 1 or server 2. Both server 1 and server 2 select to serve the data encryption service only. Client 2 selects server 1 for the voice communication service and server 2 for the data encryption service. Both server 1 and server 2 select and satisfy the voice communication service at the same levels of service parameters and QoS. The client 2 selects server 1 for the voice communication service arbitrarily. Client 3 marks the workflow including the motion detection service as unsatisfied since neither server 1 nor server 2 selects client 3 and its service. Client 4 selects server 1 for the motion detection service. Both server 1 and server 2 select and satisfy the motion detection service at the same levels of service parameters and QoS. The client 4 selects server 1 for the motion detection service arbitrarily. Hence, after iteration 1, client 2 and client 4 have their service workflow satisfied. Client 1 and client 3 request services of their workflows again in iteration 2.

At iteration 2, server 1 selects to serve client 1's data encryption service at the key length of level 3 and the encryption percentage of level 1, and the delay of 16 milliseconds. Server 2 selects to serve client 1's voice communication service at the sampling rate of level 5, the buffer size of level 5, and the throughput of 894.38 packets/second and client 3's motion detection service at the video resolution of level 3, the motion level of 0.34%, and the delay of 134.08 milliseconds. With these server solutions, clients 1 and 3 make the following decisions: client 1 selects server 1 for the data encryption service and server 2 for the voice communication service, and client 3 selects server 2 for the motion detection service. Hence, after iteration 2, all the four clients are satisfied.

From two iterations of the decentralized solution described above as in Table 39, client 2 is satisfied after iteration 1. The satisfaction of the data encryption service of client 2 at a lower level of service parameters and QoS in the decentralized solution than that in the centralized solution is a result of server 2's solution at iteration 1. In server configuration 3, all the services of voice communication, data encryption, and motion detection are provided on each server. That is, server 2 provides all the three services. In order to maximize the percentage of provided services requested and satisfied for clients as stated in the objective function, server 2 selects four services (client 1's data encryption service, client 2's voice communication and data encryption services, and client 4's motion detection service) to satisfy. In contrast, in the centralized solution only three services (client 1's voice communication service, client 2's data encryption service and client 4's motion detection service) are selected by server 2 at higher levels of service parameters and QoS for voice communication and data encryption services than those in

the decentralized solution from iteration 1. Server 2's selection of three services in the centralized solution at higher service levels is ensured by the server-client coordination constraints in Equations (15) and (16) which are not included in the local optimization problem of an individual server in the decentralized method. Without the server-client coordination constraints, each server tries to serve as many services as possible, which can lead to an overlap of the service provision by servers. For example, client 2's voice communication service is selected by both server 2 and server 1. As server 2 tries to maximize the percentage of satisfied services, the service levels of these satisfied services are lowered in order to meet the resource capacity constraints since in server configuration 3 each server does not have a sufficient resource capacity to satisfy all the services of all the clients.

Therefore, the cause for the difference between the centralized solution and the decentralized solution in the problem case with server configuration 3 and four clients is the lack of server-client coordination constraints, more specifically the isolated work of each server to maximize satisfied services at lower service levels without knowing an overlap of the service provision by servers. In contrast, the centralized method uses the server-client coordination constraints to pull all resources of all the servers together optimally to satisfy all services of all clients at higher service levels. To address this cause for the difference of the decentralized solution from the centralized solution, the service provider-user coordination strategy can be revised to distribute requests for services of the workflow to servers in a selective manner without the same service request going to multiple servers to avoid an overlap of service provision among servers, rather than sending requests for services to all servers that provide services.

For the problem case with server configuration 5 in Table 38, both the centralized solution and the decentralized solution serve client 1 and client 4 at the same levels of service parameters and QoS. In server configuration 5, neither each server nor all the servers together have a sufficient resource capacity to satisfy all services of all clients. The centralized solution selects to serve client 2 with two services of voice communication and data encryption and not to serve client 3 with only one service of motion detection. In contrast, the decentralized solution selects to serve client 3 and not to serve client 2. This difference in service decisions between the centralized solution and the decentralized solution yields the better value of the objective function from the centralized solution than that from the decentralized solution as seen in Table 37. The decentralized strategy produces its solution after three iterations. Table 40 shows service decisions made at three iterations in the decentralized solution for the problem case with server configuration 5 with four clients. The selected services of clients at each iteration are marked by "*" in Table 40.

Table 40

Service Decisions made at Three Iterations in the Decentralized Solution for One Problem Case with Server Configuration 5

Client	Service	QoS Requirement	Iteration 1		Iteration 2		Iteration 3	
			Server 1	Server 2	Server 1	Server 2	Server 1	Server 2
$k = 1$	voice com.	throughput: 850	sam. rate: 5* buf. size: 5* throughput: 894.38*					
	data enc.	delay: 20	key len.: 1 enc. per.: 1 delay: 15	key len.: 2* enc. per.: 1* delay: 15.75*				
$k = 2$	voice com.	throughput: 950						
	data enc.	delay: 35	key len.: 1 enc. per.: 1 delay: 16.43	key len.: 1 enc. per.: 1 delay: 16.43		key len.: 1 enc. per.: 1 delay: 16.43		key len.: 1 enc. per.: 1 delay: 16.43
$k = 3$	motion det.	mot. lev.: 0.2% delay: 150			vid. res.: 3* mot. lev.: 0.34%* delay: 134.08*			
$k = 4$	motion det.	mot. lev.: 0.15% delay: 90		vid. res.: 2* mot. lev.: 0.19%* delay: 78.87*				

At iteration 1, server 1 selects to serve client 1's voice communication service at the sampling rate of level 5, the buffer size of level 5, and the throughput of 894.38 and data encryption service at the key length of level 1 and the encryption percentage of level 1, and the delay of 15 milliseconds, and client 2's data encryption service at the key length of level 1 and the encryption percentage of level 1, and the delay of 16.43 milliseconds. Server 2 selects to serve client 1's data encryption service at the key length

of level 2, the encryption percentage of level 1, and the delay of 15.75 milliseconds, client 2's data encryption service at the key length of level 1, the encryption percentage of level 1, and the delay of 16.43 milliseconds, and client 4's motion detection service at the video resolution of level 2, the motion level of 0.19%, and the delay of 78.87 milliseconds.

With these server solutions, each client makes the decisions as follows. Client 1 selects server 1 for the voice communication service and sever 2 for the data encryption service. Both server 1 and server 2 select to serve the data encryption server. However, the QoS level of the data encryption service from server 2 is closest to the QoS requirement. Hence, server 2 is selected for the data encryption service. Client 2 marks the workflow including the voice communication service and the data encryption service unsatisfied since neither server 1 nor server 2 selects to serve the voice communication service. Client 3 marks the workflow including the motion detection service as unsatisfied since neither server 1 nor server 2 selects client 3 and its service. Client 4 selects server 2 for the motion detection service. Hence, after iteration 1, client 1 and client 4 have their service workflow satisfied. Client 2 and client 3 request services of their workflows again in iteration 2.

At iteration 2, server 1 selects to serve client 3's motion detection service at the video resolution of level 3, the motion level of 0.34%, and the delay of 134.08 milliseconds. Server 2 selects to serve client 2's data encryption service at the key length of level 1, the encryption percentage of level 1, and the delay of 16.43 milliseconds. With these server solutions, clients 2 and 3 make the following decisions: client 2 marks the workflow including the voice communication service and the data encryption service

unsatisfied since neither server 1 nor server 2 selects to serve the voice communication service, and client 3 selects server 1 for the motion detection service. Hence, after iteration 2, client 3 has its service workflow satisfied, in addition to clients 1 and 4 satisfied after iteration 1. Client 2 request services of its workflow again in iteration 3.

At iteration 3, server 1 cannot serve any of client 2's two services due to the insufficient resource capacity. Server 2 serves client 2's data encryption service at the key length of level 1, the encryption percentage of level 1, and the delay of 16.43 milliseconds as same as the server's decision from iteration 2. With these server solutions, client 2 still does not have its service workflow satisfied because neither server 1 nor server 2 selects to serve the voice communication service. The decentralized algorithm stops after iteration 3.

By examining the three iterations of the decentralized method, the difference between the service decision of the centralized solution in serving client 2 but dropping client 3 and the service decision of the decentralized solution in serving client 3 but dropping client 2 starts at iteration 1. At iteration 1, server 1 selects to serve three services (client 1's voice communication and data encryption services, and client 2's data encryption service) in the decentralized solution, whereas server 1 selects to serve two services (client 2's voice communication service and client 4's motion detection service) in the centralized solution. Moreover, in the decentralized solution, both server 1 and server 2 select to serve client 1's data encryption service and client 2's data encryption service. Hence, the same cause of lacking server-client coordination and the isolated work of each server to maximize satisfied services without knowing an overlap of the

service provision by servers is observed as in the problem case with server configuration 3.

This cause along with the insufficient resource capacity on each server and all the servers together makes the decentralized strategy produce a different service decision (i.e., server 1 serving client 1's voice communication service and server 2 serving client 4's motion detection service) from the centralized solution (i.e., server 2 serving client 1' voice communication service and server 1 serving client 4's motion detection service) right after iteration 1. The service decision of the decentralized method after iteration 1 leads the solution path of the decentralized method to the eventually suboptimal solution. As discussed previously, the service provider-user coordination strategy needs to be revised to distribute requests for services of workflow to servers in a selective manner to avoid an overlap of same service provision among multiple servers.

Based on the analysis results above, the following directions of developing service provider-user coordination strategies are suggested. In the decentralized resource allocation problem, each client has choices of either submitting the request for each service of the client to more than one server that provides the service, or going for servers one after another. Letting each client submit the request for each service to more than one server may result in the same problem of the overlap of service provision by multiple servers as seen previously, because the local resource allocation problem of each server does not have the server-client coordination constraint in Equation (15). It may also result in having only some but not all services of a client selected by the server—the failure of covering all services of the client, because the local resource allocation problem of each server does not have the service-client coordination constraint in Equation (16). To

overcome these problems caused by removing the server-client coordination constraints in Equations (15) and (16) in the local resource allocation problems of the servers, each client is suggested to take all of its services to one server and send remaining unsatisfied services to another server, that is, going for servers one after another until all services of the client are satisfied.

Considering each server does not want to share its resource state and resource allocation objectives with clients, the order of server selection for clients to send their services of the workflow can be based on different criteria. A random selection of servers can be one criteria, which may lead to the emergence of evenly distributed workloads on all servers over time. Using the past history of server utilization (e.g., frequency of using each server, and satisfaction with each server in terms of how many clients have been satisfied and how well the delivered QoS performance levels are close to the QoS requirements) may result in the emergence of desirable outcomes of match-making between servers and clients through the collective behavior of servers and clients over time without a central match-maker.

3.5.2 Scalability. Table 41 and Figure 6 show computation times of obtaining a centralized solution and a decentralized solution for the problem cases. As Figure 6 illustrates, the computation times of obtaining both the centralized and the decentralized solution increase with the number of clients. However, the rate of increase in the computation times with the number of clients is much larger for the centralized method than the decentralized algorithm, especially for two problem cases with server configuration 3 and server configuration 5, respectively. Due to insufficient resource capacity on each server (in server configuration 3) or all servers together (in server

configuration 5) and the provision of all three services on each server, these two problem cases are much harder to solve with a larger solution space than the problem cases with the other four server configurations. Hence, the decentralized algorithm is much more scalable than the centralized method.

Table 41

Computation Times (in seconds) of Obtaining the Centralized and Decentralized Solutions

Server Configuration	4 Clients		50 Clients		100 Clients	
	Centralized	Decentralized	Centralized	Decentralized	Centralized	Decentralized
1	0.03	0.03	0.41	0.22	1.39	0.81
2	0.05	0.03	0.41	0.24	1.83	1.43
3	0.06	0.07	1.13	0.23	4.39	1.84
4	0.05	0.03	0.41	0.24	1.83	1.43
5	0.06	0.09	7.74	2.13	9.98	2.41
6	0.05	0.06	1.62	0.52	2.29	1.60

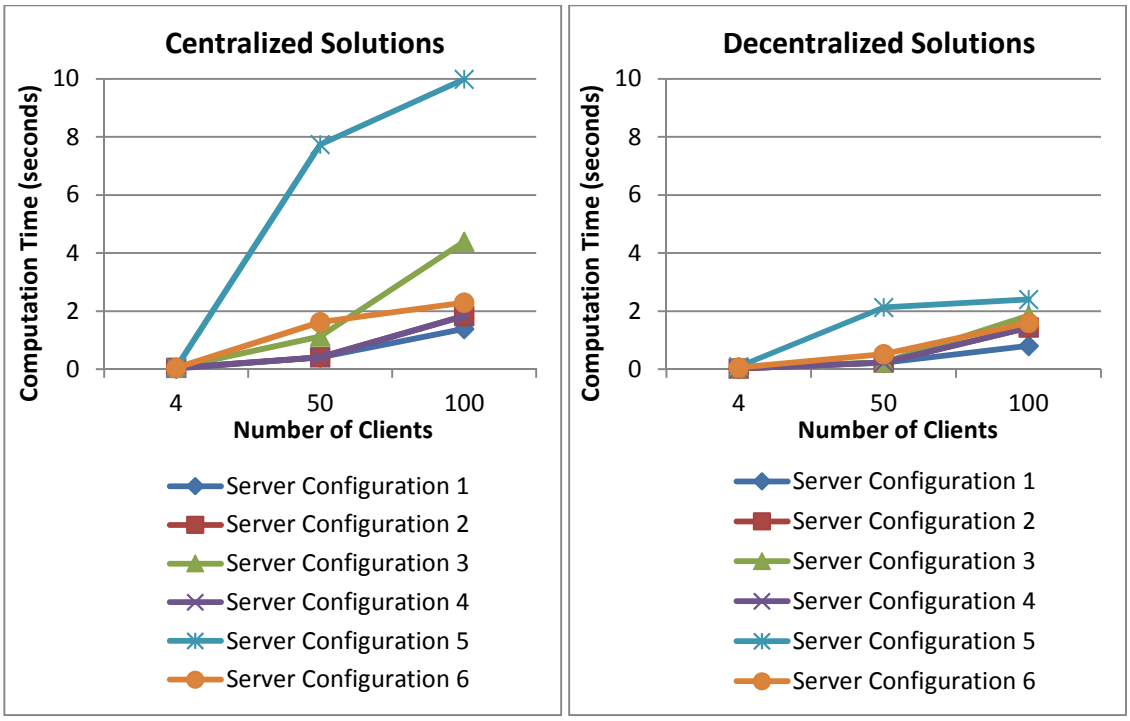


Figure 6. Computation Times (in seconds) of Obtaining the Centralized and Decentralized Solutions.

3.6 Conclusions

Although decentralized methods of resource allocation are highly desirable for scalability and real-world applicability, more research is required to develop service provider-user coordination strategies that can achieve the solution optimality through decentralized resource allocation algorithms. This study in Chapter 3 starts with a server-client coordination strategy for decentralized resource allocation algorithm and compares the decentralized solutions on various problem cases with the optimal solutions from the formulation of the centralized resource allocation problem in terms of solution optimality and scalability.

For one optimization objective of maximizing the number of clients satisfied, the decentralized solutions are as good as the centralized solutions. For another optimization

objective of producing the QoS levels as closest to the QoS requirements, the decentralized solutions are close to the centralized solutions with slightly worse values in two out of the six problem cases for four clients, one out of the six problem cases for fifty clients and two out of the six problem cases for one hundred clients.

The computation times of obtaining both a centralized solution and a decentralized solution increase with the number of clients. However, the rate of increase in the computation times with the number of clients is much larger for the centralized method than the decentralized method, especially for two harder problem cases with a larger solution space due to insufficient resource capacity on each server or all servers together and the provision of all three services on each server. Hence, the decentralized method is much more scalable than the centralized method.

By analyzing some decentralized solutions for small problem cases with four clients in comparison with the centralized solutions, it identifies the lack of server-client coordination as the major cause for differences of the decentralized solutions from the centralized solutions. Specifically, the decentralized method does not have the server-client coordination constraints that are included in the formulation of the centralized resource allocation problem. The lack of these server-client coordination constraints results in the isolated work of each server to maximize satisfied services without knowing an overlap of the service provision by servers. Thus, it is strongly suggested for each client to take its services to one server and send remaining unsatisfied services to another server, that is, to go for servers one after another until all services of the client are satisfied. Furthermore, different criteria of selecting which server to go for first and then next needs to be explored and examined. The criteria can be based on a random selection

or the past history of using the servers (e.g., frequency of using each server, and satisfaction with each server in terms of how many clients have been satisfied and how well the delivered QoS performance levels are close to the QoS requirements).

CHAPTER 4

CONCLUSIONS AND FUTURE WORK

Efficient resource allocation is the most important part in cloud computing, which determines the allocation of computer and network resources of service providers to service requests of cloud users for satisfying the users' service requirements. However, the resource allocation problem have been known to be challenging since it requires to consider all the objectives of service providers and cloud users in an unpredictable environment with dynamic workload, large shared resources and complex policies to manage them. Many research have introduced various centralized algorithms or decentralized algorithms with information sharing through communication protocols. However, achieving both solution optimality and scalability still remains as a major issue among the existing algorithms for efficient resource allocation.

Therefore, this dissertation contributes to propose two efficient resource allocation methods to generate optimal or near-optimal solutions, which can be obtained from solving the centralized optimization problem. The resource allocation methods proposed in this dissertation can be applied for resource allocation decisions with great scalability.

Chapter 2 first introduces a formulation of the resource allocation optimization problem in MIP and then designs a set of representative problem cases to analyze the optimal solutions to identify important heuristics for efficient resource allocation decisions. The proposed heuristics, which capture the centralized decision making behavior in generating the optimal solutions, are capable of making resource allocation decisions as good as or close to the optimal solutions without solving the optimization

problem directly. The resource allocation heuristics are tested in another set of problem cases with the introduction of more complexity by increasing the number of service providers and the total number of service requests, and the heuristic solutions successfully demonstrate the performance quality.

The experimental results show that the resource allocation decisions from the heuristic solutions are close to the ones obtained from the optimal solutions. In Case A, where each server has a sufficient resource capacity and in Case B, where all the servers together have a sufficient resource capacity for all service requests of all clients, the heuristic solutions do not drop any service request as same as the optimal solutions. For Case A and Case B, the heuristic solutions have the same *SAV* ratios as the optimal solutions, implying that the heuristic solutions serve all the service requests with the same quality of QoS values as in the optimal solutions. For Case C, where all the servers together do not have a sufficient resource capacity for all service requests of all clients, the heuristic solutions drop more service requests of about 1% ~ 4% of all service requests in average than the optimal solutions. However, the heuristic solutions provide at least the same levels of QoS to all served service requests as the optimal solutions with the same or higher *SAV* ratios.

The average computation times of obtaining the optimal solutions and the heuristic solutions are increased as problem cases become more complicated by increasing numbers of service requests, increasing numbers of servers and having limited resource capacity to serve all service requests of all clients. The ranges of the computation times for obtaining both the optimal and heuristic solutions also become larger with higher values of the standard deviation. However, the rate of increase in the

computation times with increasing problem complexity is much larger for the optimal solutions than the heuristic solutions, especially for Case B and Case C with insufficient resource capacity on each server or all the servers together and for the problem cases with ten and twenty servers resulting in a larger solution space than the other problem cases. Hence, using the resource allocation heuristics is much more scalable than solving the optimization problem directly.

Chapter 3 provides the modified formulation of the centralized resource allocation problem and the decentralized service provider-user coordination strategy in cloud computing. It specifically looks into elements of service provider-user coordination from the centralized formulation, and the differences between the centralized solutions and the decentralized solutions for various problem cases are analyzed to recognize the key elements of the decentralized service provider-user coordination strategy, which can lead to get optimal or near-optimal solutions.

The experimental results confirm that the decentralized solutions are as good as the centralized solutions in terms of maximizing the number of clients satisfied as one of the objective functions in the optimization problem. For another objective of producing the QoS levels as closest to the QoS requirements, the decentralized solutions are close to the centralized solutions with slightly worse values in two out of the six problem cases for four clients, one out of the six problem cases for fifty clients and two out of the six problem cases for one hundred clients. The computation times of obtaining the centralized solution and the decentralized solution increase with the number of clients. However, the rate of increase in the computation times with the number of clients is much larger for the centralized method than the decentralized method, especially for two

harder problem cases with a larger solution space due to insufficient resource capacity on each server or all the servers together and the provision of all three services on each server. Hence, the decentralized method is much more scalable than the centralized method.

By analyzing some decentralized solutions for small problem cases with four clients in comparison with the centralized solutions, it identifies the lack of server-client coordination as the major cause for differences of the decentralized solutions from the centralized solutions. Specifically, the decentralized method does not have the server-client coordination constraints that are included in the formulation of the centralized resource allocation problem. The lack of these server-client coordination constraints results in the isolated work of each server to maximize satisfied services without knowing an overlap of the service provision by servers. Thus, it is strongly suggested for each client to take its services to one server and send remaining unsatisfied services to another server, that is, to go for servers one after another until all services of the client are satisfied. Furthermore, different criteria of selecting which server to go for first and then next need to be explored and examined. The criteria can be based on a random selection or the past history of using the servers (e.g., frequency of using each server, and satisfaction with each server in terms of how many clients have been satisfied and how well the delivered QoS performance levels are close to QoS requirements).

Two proposed methods in this dissertation show comparable performance to the optimal solutions for resource allocation with respect to solution optimality and scalability. All the experiments, however, are limited to have relatively a small number of servers with simple structures of service provision on servers using up to three different

types of services. Therefore, future work may include expanding problem sizes of the resource allocation optimization by increasing the total number of servers, using various types of workflow such as parallel processing with involvement of several different types of services. Moreover, different service provisions on servers may also be introduced with some degree of overlapping services such that servers provide all the same services, servers provide services partially overlapped and servers provide different services with no overlap. In addition to various changes in the total number of servers, workflow structure and service provision on servers, future work may want to explore other heuristics for further improvements in performance quality of the resource allocation methods.

REFERENCES

Alrifai, M., & Risse, T. (2009). Combining Global Optimization with Local Selection for Efficient QoS-aware Service Composition. *18th International World Wide Web Conference (WWW)*, 881-890.

Alrifai, M., Skoutas, D., & Risse, T. (2010). Selecting Skyline Services for QoS-based Web Service Composition. *19th International World Wide Web Conference*, 11-20.

Aoun, R., Doumith, E. A., & Gagnaire, M. (2010). Resource Provisioning for Enriched Services in Cloud Environment. *2nd IEEE International Conference on Cloud Computing Technology and Science*, 296-303.

Ardagna, D., & Pernici, B. (2005). Global and local QoS Guarantee in Web Service Selection. *IEEE International Conference on Web Services*, 805-806.

Ardagna, D., & Pernici, B. (2007). Adaptive Service Composition in Flexible Processes. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, Vol. 33, No. 6*, 369-384.

Ardagna, D., Casolari, S., & Panicucci, B. (2011). Flexible Distributed Capacity Allocation and Load Redirect Algorithms for Cloud Systems. *IEEE 4th International Conference on Cloud Computing*, 163-170.

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., . . . Zaharia, M. (2010). A View of Cloud Computing. *Communications of The Acm, Vol. 53, No. 4*, 50-58.

Atiewi, S., Yussof, S., & Ezanee, M. (2015). A Comparative Analysis of Task Scheduling Algorithms of Virtual Machines in Cloud Environment. *Journal of Computer Science*, 804-812.

Berman, F. (1999). High-performance schedulers. In *The Grid Blueprint for a New Computing Infrastructure, edited by Ian Foster and Carl Kesselman.* San Francisco, CA: Morgan Kaufman Publishers.

Canfora, G., Penta, M. D., Esposito, R., & Villani, M. L. (2005). An Approach for QoS-aware Service Composition based on Genetic Algorithms. *International Conference on Genetic and Evolutionary Computation (GECCO)*, 1069-1075.

Casati, F., & Shan, M.-C. (2001). Dynamic and adaptive composition of e-services. *Information Systems, 26(3)*, 143–163.

Chen, Y., Farley, T., & Ye, N. (2004). QoS requirements of network applications over the internet. *Information, Knowledge and System Management, Vol. 4, No. 1*, 55-76.

Dhingra, A., & Paul, S. (2014). Green Cloud: Heuristic based BFO Technique to Optimize Resource Allocation. *Indian Journal of Science and Technology, Vol 7(5)*, 685-691.

Endo, P. T., Palhares, A. V., Pereira, N. N., Gonçalves, G. E., Sadok, D., Kelner, J., . . . Mångs, J.-E. (2011). Resource Allocation for Distributed Cloud: Concepts and Research Challenges. *IEEE Network, Vol. 25, No. 4*, 42-46.

Ercetin, O., & Tassiulas, L. (2003). Market-Based Resource Allocation for Content Delivery in the Internet. *IEEE TRANSACTIONS ON COMPUTERS, VOL. 52, NO. 12*, 1573-1585.

Foster, I., Zhao, Y., Raicu, I., & Lu, S. (2008). Cloud Computing and Grid Computing 360-Degree Compared. *Grid Computing Environments Workshop*, 1-10.

Gao, J., Guibas, L., Milosavljevic, N., & Zhou, D. (2009). Distributed Resource Management and Matching in Sensor Networks. *IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 97-108.

Gong, Z., Ramaswamy, P., Gu, X., & Ma, X. (2009). SigLM: Signature-Driven Load Management for Cloud Computing Infrastructures. *IEEE International Workshop on Quality of Service (IWQoS)*, 1-9.

Goudarzi, H., & Pedram, M. (2011a). Maximizing Profit in Cloud Computing System via Resource Allocation. *2011 31st International Conference on Distributed Computing Systems Workshops*, 1-6.

Goudarzi, H., & Pedram, M. (2011b). Multi-dimensional SLA-based Resource Allocation for Multi-tier Cloud Computing Systems. *2011 IEEE 4th International Conference on Cloud Computing*, 324-331.

Haresh, M. V., Kalady, S., & Govindan, V. K. (2011). Agent Based Dynamic Resource Allocation on Federated Clouds. *IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, 111-114.

Hassan, M. M., Song, B., & Huh, E.-N. (2011). Distributed resource allocation games in horizontal dynamic cloud federation platform. *IEEE International Conference on High Performance Computing and Communications*, 822-827.

Heo, J., Henriksson, D., Liu, X., & Abdelzaher, T. (2007). Integrating Adaptive Components: An Emerging Challenge in Performance-Adaptive Systems and a Server Farm Case-Study. *28th IEEE International Real-Time Systems Symposium*, 227-238.

Heo, J., Jayachandran, P., Shin, I., Wang, D., & Abdelzaher, T. (2009). OptiTuner: An Automatic Distributed Performance Optimization Service and a Server Farm

Application. *4th International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks (FeBID)*.

Hsu, C.-H., Chen, T.-L., & Park, J.-H. (2008). On improving resource utilization and system throughput of master slave job scheduling in heterogeneous systems. *The Journal of Supercomputing, Vol. 45*, 129-150.

Hu, L., Cao, J., & Gu, Z. (2008). Modeling semantic web service using semantic templates. *International Conference on Semantics, Knowledge and Grid*, 165-172.

Jamkhedkar, P. A., Lamb, C. C., & Heileman, G. L. (2011). Usage Management in Cloud Computing. *IEEE 4th International Conference on Cloud Computing*, 525-532.

Jung, G., & Sim, K. M. (2011). Agent-based Adaptive Resource Alloc tion on the Cloud Computing Environment. *International Conference on Parallel Processing Workshops*, 345-351.

Kadda, B. B., Benhammadi, F., Sebbak, F., & Mataoui, M. (2015). New Tasks Scheduling Strategy for Resources Allocation in Cloud Computing Environment. *6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO)*, 1-5.

Kreger, H. (2001). *Web Services Conceptual Architecture (WSCA 1.0).* IBM Software Group.

Kumar, K., Feng, J., Nimmagadda, Y., & Lu, Y.-H. (2011). Resource Allocation for Real-Time Tasks using Cloud Computing. *2011 Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN)*, 1-7.

Kuribayashi, S.-i. (2011). Optimal Joint Multiple Resource Allocation Method for Cloud Computing Environments. *International Journal of Research and Reviews in Computer Science (IJRRCS), Vol. 2, No. 1*, 1-8.

Laili, Y., Tao, F., Zhang, L., Cheng, Y., Luo, Y., & Sarker, B. R. (2013). A Ranking Chaos Algorithm for Dual Scheduling of Cloud Service and Computing Resource in Private Cloud. *Computers in Industry, Vol. 64*, 448-463.

Lamparter, S., Ankolekar, A., & Studer, R. (2007). Preference-based selection of highly configurable web services. *International World Wide Web Conference*, 1013-1022.

Li, H., & Li, H. (2011). A Research of Resource Scheduling Strategy For Cloud Computing Based on Pareto Optimality MxN Production Model. *International Conference on Management and Service Science*, 1-5.

Li, K., Wang, Y., & Liu, M. (2014). A Task Allocation Scheme Based on Response Time Optimization in Cloud Computing. *Distributed, Parallel, and Cluster Computing*, 1-19.

Lin, C., & Lu, S. (2011). Scheduling Scientific Workflows Elastically for Cloud Computing. *IEEE 4th International Conference on Cloud Computing*, 746-747.

Liu, S., Quan, G., & Ren, S. (2010). On-line Scheduling of Real-time Services for Cloud Computing. *IEEE 6th World Congress on Services*, 459-464.

Liu, Z., Zhou, H., Fu, S., & Liu, C. (2014). Algorithm Optimization of Resources Scheduling Based on Cloud Computing. *Journal of Multimedia, Vol. 9, No. 7*, 977-984.

Livny, M., & Raman, R. (1999). High Throughput Resource Management. In *The Grid Blueprint for a New Computing Infrastructure, edited by Ian Foster and Carl Kesselman* (p. Chapter 13). San Francisco, CA: Morgan Kaufman Publishers.

Manzalini, A., & Moiso, C. (2011). Self-optimization of resource allocation in decentralised server farms. *15th International Conference on Intelligence in Next Generation Networks*, 219-224.

Mao, Z., Shang, Y., Liu, C., & Chen, J. (2013). Utility-based Price Proportion in Cloud Resource Allocation. *Information Technology Journal 12 (22)*, 6882-6886.

Marinescu, D. C. (2013). *Cloud Computing: Theory and Practice.* 1 edition: Morgan Kaufman.

Mehdi, N. A., Mamat, A., Ibrahim, H., & Subramaniam, S. K. (2011). Impatient Task Mapping in Elastic Cloud using Genetic Algorithm. *Journal of Computer Science 7 (6)*, 877-883.

MessinaFabrizio, PappalardoGiuseppe, & SantoroCorrado. (2012). Decentralised Resource Finding in Cloud/Grid Computing Environments: a Performance Evaluation. “2012 IEEE 21st International WETICE”, 143-148.

MessinaFabrizio, PappalardoGiuseppe, & SantoroCorrado. (2014). Decentralised Resource finding and Allocation in Cloud Federations. “2014 International Conference on Intelligent Networking and Collaborative Systems”, 26-33.

Nesmachnow, S., Iturriaga, S., & Dorronsoro, B. (2015). Efficient Heuristics for Profit Optimization of Virtual Cloud Brokers. *IEEE Computational Intelligence Magazine*, 33-43.

Papagianni, C., Leivadeas, A., Papavassiliou, S., Maglaris, V., Cervello´-Pastor, C., & Monje, A. (2013). On the Optimal Allocation of Virtual Resources in Cloud

Computing Networks. *IEEE TRANSACTIONS ON COMPUTERS, VOL. 62, NO. 6*, 1060-1071.

Papazoglou, M. P., Traverso, P., Dustdar, S., & Leymann, F. (2007). Service-Oriented Computing: State of The Art and Research Challenges. *IEEE Computer, Vol. 40, No. 11*, 38-45.

Rajan, S., & Jairath, A. (2011). Cloud Computing: The Fifth generation of Computing. *International Conference on Communication Systems and Network Technologies*, 665-667.

Rao, J., Bu, X., Xu, C.-Z., & Wang, K. (2011). A Distributed Self-learning Approach for Elastic Provisioning of Virtualized Cloud Resources. *19th Annual IEEE International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems*, 45-54.

Rezvani, M., Akbari, M. K., & Javadi, B. (2015). Resource Allocation in Cloud Computing Environments Based on Integer Linear Programming. *Section B: Computer and Communications Networks and Systems, The Computer Journal, Vol. 58 No. 2*, 300-314.

Schlegel, T., Kowalczyk, R., & Vo, Q. B. (2008). Decentralized Co-Allocation of Interrelated Resources in Dynamic Environments. *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 104-108.

Selvi, S. T., Valliyammai, C., & Dhatchayani, V. N. (2014). Resource Allocation Issues and Challenges in Cloud Computing. *2014 International Conference on Recent Trends in Information Technology*, 1-6.

Sharma, S., Tantawi, A., Spreitzer, M., & Steinder, M. (2010). Decentralized Allocation of CPU Computation Power for Web Applications. *Performance Evaluation 67*, 1187-1202.

Shi, W., & Hong, B. (2010). Resource Allocation with a Budget Constraint for Computing Independent Tasks in the Cloud. *2nd IEEE International Conference on Cloud Computing Technology and Science*, 327-334.

Shiang, H.-P., & van der Schaar, M. (2009). Distributed Resource Management in Multihop Cognitive Radio Networks for Delay-Sensitive Transmission. *IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, VOL. 58, NO. 2*, 941-953.

Shyamala, K., & Rani, T. S. (2015). An Analysis on Efficient Resource Allocation Mechanisms in Cloud Computing. *Indian Journal of Science and Technology, Vol 8(9)*, 814-821.

Sindhu, S., & Mukherjee, S. (2013). A Genetic Algorithm based Scheduler for Cloud Environment. *4th International Conference on Computer and Communication Technology (ICCCT)*, 23-27.

Smith, J., Chong, E. K., Maciejewski, A. A., & Siegel, H. J. (2012). Overlay network resource allocation using a decentralized market-based approach. *Future Genration Computer Systems, Vol. 28, No. 1*, 24-35.

SonSeokho, JungGihun, & JunChanSung. (2013). An SLA-based cloud computing that facilitates resource allocation in the distributed data centers of a cloud provider. "J Supercomput", 606-637.

Srinivasa, K. G., Kumar, K. S., Kaushik, U. S., Srinidhi, S., Shenvi, V., & Mishra, K. (2014). Game Theoretic Resource Allocation in Cloud Computing. *2014 Fifth International Conference on the Applications of Digital Information and WEb Technologies (ICADIWT)*, 36-42.

Staikopoulos, A., Cliffe, O., Popescu, R., Padget, J., & Clarke, S. (2010). Template-based adaptation of semantic web services with model-driven engineering. *IEE Transactions on Services Computing, Vol. 3, No. 2*, 116-128.

Stillwell, M., Vivien, F., & Casanova, H. (2012). Virtual Machine Resource Allocation for Service Hosting on Heterogeneous Distributed Platforms. *26th IEEE International Parallel & Distributed Processing Symposium*, 786-797.

Suresh, A., & Vijayakarthick, P. (2011). Improving Scheduling of Backfill Algorithms using Balanced Spiral Method for Cloud Metascheduler. *IEEE-International Conference on Recent Trends in Information Technology, ICRTIT 2011*, 624-627.

Tran, V. X., Tsuji, H., & Masuda, R. (2009). A new QoS ontology and its QoS-based ranking algorithm for web services. *Simulation Modelling Practice and Theory, Vol. 17, No. 8*, 1378-1398.

Urgaonkar, R., Kozat, U. C., Igarashi, K., & Neely, M. J. (2010). Dynamic Resource Allocation and Power Management in Virtualized Data Centers. *2010 IEEE/IFIP Network Operations and Management Symposium - NOMS*, 479-486.

Varalakshmi, P., Judgi, T., & Hafsa, M. F. (2013). Local Trust Based Resource Allocation in Cloud. *2013 Fift International Conference on Advanced Computing (ICoAC)*, 591-596.

Wang, W., & Li, B. (2005). Market-Based Self-Optimization for Autonomic Service Overlay Networks . *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 23, NO. 12*, 2320-2332.

Wang, X., Vitvar, T., Kerrigan, M., & Toma, I. (2006). A QoS-Aware Selection Model for Semantic Web Services. *International Conference on Service Oriented Computing*, 390-401.

Wang, Z., & Fang, T. (2014). Task Scheduling Model Based on Multi-Agent and Multi-Objective Dynamical Scheduling Algorithm. *Journal of Networks, Vol. 9, No. 6*, 1588-1595.

Wang, Z., & Su, X. (2015). Dynamically hierarchical resource-allocation algorithm in cloud computing environment. *The Journal of Supercomputing*, 2748-2766.

Wei, Y., & Blake, B. M. (2013). Decentralized Resource Coordination across Service Workflows in a Cloud Environment. *2013 Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 15-20.

Wei, Y., Blake, B. M., & Saleh, I. (2013). Adaptive Resource Management for Service Workflows in Cloud Environments. *2013 IEEE 27th International Symposium on Parallel & Distributed Processing Workshops and PhD Forum*, 2147-2156.

Wendell, P., Jiang, J. W., Rexford, J., & Freedman, M. J. (2009). Decentralized Server Selection Through Joint Proximity and Load Optimization. *Princeton University, Computer Science: TR-868-09., Tech. Rep.*

Wu, X., Deng, M., Zhang, R., Zeng, B., & Zhou, S. (2013). A Task Scheduling Algorithm based on QoS-driven in Cloud Computing. *1st Inernational Conference on Information Technology and Quantitative Management, Vol. 17*, 1162-1169.

Wuhib, F., Stadler, R., & Spreitzer, M. (2010). Gossip-based Resource Management for Cloud Environments. *6th International Conference on Network and Service Management*, 1-8.

Yang, Z., Qin, X., Li, W., & Yang, Y. (2013). Optimized Task Scheduling and Resource Allocation in Cloud Computing Using PSO based Fitness Function. *Information Technology Journal 12 (23)*, 7090-7095.

Yau, S. S., Ye, N., Sarjoughian, H. S., Huang, D., Roontiva, A., Baydogan, M., & Muqsith, M. A. (2009). Toward Development of Adaptive Service-Based Software Systems. *IEEE Transactions on Services Computing, Vol. 2, No. 3*, 247-260.

Ye, N., Yang, S. S., & Aranda, B. M. (2013). The Analysis of Service Provider-User Coordination for Resource Allocation in Cloud Computing. *Information KnowledgeSystems Management, Vol. 12, No. 1*, 1-24.

Ye, N., Yau, S., Huang, D., Baydogan, M., Aranda, B. M., Roontiva, A., & Hurley, P. (2010). Models of dynamic relations among service activities, system state and

service quality on computer and network systems. *Information, Knowledge, Systems Management, Vol. 9, No. 2*, 99-116.

Yigitbasi, N., Iosup, A., Epema, D., & Ostermann, S. (2009). C-Meter: A Framework for Performance Analysis of Computing Clouds. *9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, 472-477 .

Yin, B., Wang, Y., Meng, L., & Qiu, X. (2012). A Multi-dimensional Resource Allocation Algorithm in Cloud Computing. *Journal of Information & Computational Science 9: 11*, 3021-3028.

Zaman, S., & Grosu, D. (2011). Efficient Bidding for Virtual Machine Instances in Clouds. *IEEE 4th International Conference on Cloud Computing*, 41-48.

Zeng, L., Benatallah, B., Ngu, A. H., Dumas, M., Kalagnanam, J., & Chang, H. (2004). QoS-Aware Middleware for Web Services Composition. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, Vol. 30, No. 5*, 311-327.

Zhang, L.-J., Zhang, J., & Cai, H. (2007). *Services Computing.* New York, USA: Springer.

Zhang, Z., & Zhang, X. (2009). Realization of Open Cloud Computing Federation Based on Mobile Agent. *IEEE International Conference on the Intelligent Computing and Intelligent Systems*, 642-646.

Zheng, H., Yang, J., & Zhao, W. (2010). QoS Analysis and Service Selection for Composite Services. *IEEE International Conference on Services Computing*, 122 - 129.

Zhou, J., Dutkiewicz, E., Liu, R. P., Fang, G., & Liu, Y. (2014). Modified Elite Chaotic Immune Clonal Selection Algorithm for Sever Resource Allocation in Cloud Computing Systems. *17th International Symposium on Wireless Personal Multimedia Communications (WPMC2014)*, 226-231.

Zuo, X., Zhang, G., & Tan, W. (2014). Self-Adaptive Learning PSO-Based Deadline Constrained Task Scheduling for Hybrid IaaS Cloud. *IEEE Transactions on Automation Science and Engineering, Vol. 11, No. 2*, 564-573.

APPENDIX A

DESCRIPTION OF VARIOUS HEURISTICS IN EXISTING STUDIES

Literatures	Heuristics
(Atiewi, Yussof, & Ezanee, 2015)	Three task scheduling algorithms are used: • Random resource selection - Assign the preferred task in a random manner to the available VM regardless of its workload status of either heavy or light • Round Robin (RR) - Assign the selected tasks over the available VMs in a round-robin order, where each task is equally administered • Green scheduler - Use alternative server and stay away from the congested routes by tracking buffer occupancy on the path
(Dhingra & Paul, 2014)	VMs are migrated from one host to another according to three heuristics: • Maximum Utilization - Migrate a VM from the overloaded hosts with the maximum CPU utilization • Minimum utilization - Migrate a VM from the overloaded hosts with the minimum CPU utilization • Random choice - Migrate a VM, which is selected on the basis of a uniformly distributed discrete random variable
(Goudarzi & Pedram, 2011a)	As the first step of the heuristic, a good initial solution is obtained by processing clients sequentially and assigning to the best cluster on that time. This greedy algorithm is repeated for a number of times and the best initial solution is selected. Then a local search is used to improve the quality of the initial solution.
(Goudarzi & Pedram, 2011b)	A heuristic inspired by the force-directed scheduling is used as a search technique for a solution space. It is based on defined forces between servers and clients, and a force is calculated based on the partial profit gained from allocation each portion of a client's request to a server. A client with highest force difference toward a new server is picked and if the required server is available, the load replacement is done. Forces are updated and new maximum force differential client-to-server assignment is made. It continues until there is no positive force differentials for any client.
(Kadda, Benhammadi, Sebbak, & Mataoui, 2015)	A new task scheduling heuristic is proposed with a matrix of the expected execution time for each job on a cluster. As the first step of this heuristic algorithm, a good initial solution is applied to find two first clusters with minimum Completion Time where jobs are assigned sequentially to the cluster using the Min-min algorithm. Then, to improve the quality of the initial solution, a local assigning in each cluster is applied to allocate the tasks on the different available servers.

(Messina, Pappalardo, & Santoro, 2012, 2014)	A service request can flow on the overlay network, and the heuristic strategies introduced in this study help to guide the traveling of the network. Allocation heuristic specifies how to make a node choice when more nodes are valid candidates, and "First Fit" (i.e. random selection) is applied for the allocation heuristic. If the node is not appropriate, six forward heuristics are executed to find candidate nodes with different criteria as follows: • Best Fit - Select the node with minimal distance from target point • Worst Fit - Select the node with highest amount of available resources and thus the further node with regard to the Euclidean distance • Mass Center - Select the node among wider view of neighbors at 2-hops • Best Fit/Mass Center - Best fit strategy is kept if the request is approaching the admissible region. Otherwise Mass Center strategy is used. • Max Connection - Select the node with highest number of connections/links • First Fit - Select a random neighbor
(Nesmachnow, Iturriaga, & Dorronsoro, 2015)	Seven heuristics are proposed to assign priorities to requests with diverse criteria as follows: • MaxMaxProfit - Assign a request to a VM with the global profit • MaxPMinT - Select a VM to serve a request the soonest and then select the pair (request, VM) with best profit • MinTMaxP - Build a set of (request, VM) pairs with the maximum profit and then select the pair with minimum execution time • MaxQTMaxP - Search for a request with the maximum overall profit and then select the pair with maximum waiting time • MinQTMaxP - Search for a request with the maximum overall profit and then select the pair with minimum waiting time • MinGAPMaxP - Search for a request with maximum overall profit and then select the pair with minimum deadline GAP • MinDMaxP - Search for a request with maximum overall profit and then select the pair with minimum deadline Then, reordering local search is used to improve the solution quality by performing a set of reordering movements on the schedule and executing in a reduced time.
(Yang, Qin, Li, & Yang, 2013)	Particle Swarm Optimization based fitness function scheduling heuristic is to assign each subtask to an appropriate resource (routing problem) and to sequence the subtasks on the resources (sequencing problem).

APPENDIX B

AN EXAMPLE OF THE RESOURCE ALLOCATION PROBLEM FORMULATION

The following example gives realization of the resource allocation problem formulation for a problem case with two servers and four clients. It supposes that the first server is the communication-centered server with resource capacity limits of 36 for CPU resource and 200 for bandwidth resource, and the second server is a computation-centered server with resource capacity limits of 232 for CPU resource and 32 for bandwidth resource. It also supposes that clients 1 and 4 request computation intensive services with minimum QoS requirements of 6 and 30, and clients 2 and 3 request communication intensive services with minimum QoS requirements of 25 and 5. This problem case assumes that each server has a sufficient resource capacity to satisfy all four clients' service requests, and all two services of communication intensive service and computation intensive service are provided by each server.

- Variables and indices

  k: a given client, $k = 1, 2$

  i: a given server, $i = 1, 2$

  w_i: resource variable *w* of server *i,* $w_i = 1, 2$, and $w_i = 1$ for CPU resource, $w_i = 2$ for bandwidth resource

  s: service (type), $s = 1, 2$, and $s = 1$ for a communication intensive service, $s = 2$ for a computation intensive service

  d_s: service parameter *d* of service *s,* $d_s = 1, \ldots, D_s$, and $D_1 = 1$, $d_1 = 1$ for one service parameter of the communication intensive service, $D_2 = 1$, $d_2 = 1$ for one service parameter of the computation intensive service

  p_s: QoS variable *p* of service *s,* $p_s = 1, \ldots, P_s$, and $P_1 = 1$, $p_1 = 1$ for one QoS variable of the communication intensive service, $P_2 = 1$, $p_2 = 1$ for one QoS variable of the computation intensive service

  R_{kiw_i}: amount of resource variable *w* of server *i* taken by client *k*'s service request, R_{kiw_i} is a positive real value

  Q_{kp_si}: value of QoS variable p_s of client *k*'s service request on server *i*, Q_{kp_si} is a positive real value

- Decision variables

X_{ki} = 1 if client *k*'s service request is assigned to server

0 if client *k*'s service request is not assigned to server *i*

A_{kd_si}: level of service parameter d_s for client *k*'s service request on server *i*

A_{11i}= 1, 2, 3, 4, or 5

A_{21i}= 1, 2, 3, 4, or 5

A_{31i}= 1, 2, 3, 4, or 5

A_{41i}= 1, 2, 3, 4, or 5

- Given inputs

U_{ks} = 1 if client k's service request uses service s

0 if client k's service request does not use service s

$U_{11} = 0, U_{12} = 1$ (client 1's service request uses the computation intensive service)

$U_{21} = 1, U_{22} = 0$ (client 2's service request uses the communication intensive service)

$U_{31} = 1, U_{32} = 0$ (client 3's service request uses the communication intensive service)

$U_{41} = 0, U_{42} = 1$ (client 4's service request uses the computation intensive service)

V_{is} = 1 if service s is provided by server i

0 if service s is not provided by server i

$V_{11} = 1$ (server 1 provides the communication intensive service)

$V_{12} = 1$ (server 1 provides the computation intensive service)

$V_{21} = 1$ (server 2 provides the communication intensive service)

$V_{22} = 1$ (server 2 provides the computation intensive service)

$A^l_{kd_si}$: limit (i.e. the maximum level) of service parameter d_s of client *k*'s service request on server *i*

A^l_{11i}= 5

A^l_{21i}= 5

A^l_{31i}= 5

$A_{41i}^{l} = 5$

$R_{iw_i}^{l}$: limit of resource variable *w* of server *i*

$R_{11}^{l} = 36$ (server 1's CPU resource capacity)

$R_{12}^{l} = 200$ (server 1's bandwidth resource capacity)

$R_{21}^{l} = 232$ (server 2's CPU resource capacity)

$R_{22}^{l} = 32$ (server 2's bandwidth resource capacity)

$Q_{kp_s}^{l}$: limit of QoS variable p_s of client *k*'s service request

$Q_{11}^{l} = 6$

$Q_{21}^{l} = 25$

$Q_{31}^{l} = 5$

$Q_{41}^{l} = 30$

- Objective function

$$\text{Minimize} \sum_{k} \sum_{p_s} \frac{\left| \sum_{i} Q_{kp_s i} - Q_{kp_s}^{l} \right|}{Q_{kp_s}^{l} * P_s} \quad (1)$$

$$\frac{|Q_{111}+Q_{112}-6|}{6} + \frac{|Q_{211}+Q_{212}-25|}{25} + \frac{|Q_{311}+Q_{312}-5|}{5} + \frac{|Q_{411}+Q_{412}-30|}{30}$$

- Server-client coordination constraints

$$\sum_{i} X_{ki} \leq 1 \quad \forall k \quad (2)$$

$X_{11} + X_{12} \leq 1$

$X_{21} + X_{22} \leq 1$

$X_{31} + X_{32} \leq 1$

$X_{41} + X_{42} \leq 1$

- Service constraints

$$X_{ki} U_{ks} \leq V_{is} \quad \forall k, i, s \quad (3)$$

$X_{11} U_{12} \leq V_{12}$

$X_{12} U_{12} \leq V_{22}$

$X_{21} U_{21} \leq V_{11}$

$$X_{22}U_{21} \le V_{21}$$
$$X_{31}U_{31} \le V_{11}$$
$$X_{32}U_{31} \le V_{21}$$
$$X_{41}U_{42} \le V_{12}$$
$$X_{42}U_{42} \le V_{22}$$

$$A_{kd_si} \le A^{l}_{kd_si} \quad \forall i, k, d_s \tag{4}$$
$$A_{11i} \le 5$$
$$A_{21i} \le 5$$
$$A_{31i} \le 5$$
$$A_{41i} \le 5$$

- Service-resource-QoS relation constraints

$$R_{kiw_i} = X_{ki}F_{iw_i}(A_{k1i}, \dots, A_{kD_si}) \quad \forall i, k, w_i \tag{5}$$
$$R_{1i1} = X_{1i} * (5.8 * A_{11i}), \qquad R_{1i2} = X_{1i} * (0.3 * A_{11i})$$
$$R_{2i1} = X_{2i} * (0.1 * A_{21i}), \qquad R_{2i2} = X_{2i} * (5.0 * A_{21i})$$
$$R_{3i1} = X_{3i} * (0.1 * A_{31i}), \qquad R_{3i2} = X_{3i} * (5.0 * A_{31i})$$
$$R_{4i1} = X_{4i} * (5.8 * A_{41i}), \qquad R_{4i2} = X_{4i} * (0.3 * A_{41i})$$

$$Q_{kp_si} = X_{ki}G_{ip_s}(R_{ki1}, \dots, R_{kiW_i}) \quad \forall i, k, p_s \tag{6}$$
$$Q_{11i} = X_{1i} * (R_{1i1} + R_{1i2})$$
$$Q_{21i} = X_{2i} * (2R_{2i1} + R_{2i2})$$
$$Q_{31i} = X_{3i} * (2R_{3i1} + R_{3i2})$$
$$Q_{41i} = X_{4i} * (R_{4i1} + R_{4i2})$$

- Resource capacity constraints

$$\sum_k R_{kiw_i} \le R^{l}_{iw_i} \quad \forall i, w_i \tag{7}$$
$$R_{111} + R_{211} + R_{311} + R_{411} \le 36$$
$$R_{112} + R_{212} + R_{312} + R_{412} \le 200$$
$$R_{121} + R_{221} + R_{321} + R_{421} \le 232$$
$$R_{122} + R_{222} + R_{322} + R_{422} \le 32$$

- QoS requirement constraints

$$Q_{kp_si}X_{ki} \leq Q^{l}_{kp_s} \text{ or } Q_{kp_si} \geq X_{ki}Q^{l}_{kp_s} \quad \forall i, k, p_s \tag{8}$$

$$Q_{11i} \geq 6 * X_{1i}$$

$$Q_{21i} \geq 25 * X_{2i}$$

$$Q_{31i} \geq 5 * X_{3i}$$

$$Q_{41i} \geq 30 * X_{4i}$$