MINISTRY OF EDUCATION AND TRAINING
THE UNIVERSITY OF DANANG
***

PHAM NGUYEN MINH NHUT

RESOURCE ALLOCATION FOR VIRTUAL SERVICE BASED ON SHARED HOSTING PLATFORM IN CLOUD COMPUTING

MAJOR: COMPUTER SCIENCE
CODE: 62 48 01 01

DOCTORAL DISSERTATION
(EXECUTIVE SUMMARY)

DA NANG, 2018

MINISTRY OF EDUCATION AND TRAINING
THE UNIVERSITY OF DANANG
***

Advisors

1. Assoc.Prof. Dr. Le Van Son

2. Assoc.Prof. Dr. Doan Van Ban

Reviewer 1: Prof. Dr. Nguyen Thuc Hai

Reviewer 2: Prof. Dr. Nguyen Thanh Thuy

Reviewer 3: Assoc.Prof. Dr. Vo Thanh Tu

The dissertation is defended before The Assessment Committee at The University of Danang.

Time: 08h30

Date: 12/7/2018

The dissertation is available at:

- National Library of Vietnam.

- Learning & Information Resources Center, The University of Danang.

LIST OF PUBPLICATIONS

1. Phạm Nguyễn Minh Nhựt, Đoàn Văn Ban, Lê Văn Sơn. Mô hình nền tảng máy chủ chia sẻ và bài toán Vector Packing trong cung cấp tài nguyên cho dịch vụ ảo hóa, *Tạp chí Tin học và Điều khiển học*, 30(1):63-72, 2014.

2. Phạm Nguyễn Minh Nhựt, Lê Văn Sơn, Đoàn Văn Ban. Thuật toán Max-Min Ant System trong cung cấp tài nguyên cho dịch vụ ảo hóa từ nền tảng máy chủ chia sẻ. *Hội thảo quốc gia lần thứ XVII: Một số vấn đề chọn lọc của Công nghệ thông tin và Truyền thông*, trang 331-336, 2014.

3. Nguyen Minh Nhut Pham, Thu Huong Nguyen, Van Son Le. Resource Allocation for Virtual Service Based on Heterogeneous Shared Hosting Platforms. In *the 8th Asian Conference Intelligent Information and Database Systems*, 9622:51-60. Springer, 2016.

4. Phạm Nguyễn Minh Nhựt, Lê Văn Sơn, Hoàng Bảo Hùng. Thuật toán PSO cải tiến trong cung cấp tài nguyên cho dịch vụ ảo hóa dựa trên nền tảng máy chủ chia sẻ không đồng nhất. *Chuyên san Các công trình nghiên cứu, phát triển và ứng dụng Công nghệ thông tin và Truyền thông*, 2(36):80-95, 2016.

5. Nguyen Minh Nhut Pham, Van Son Le. Applying Ant Colony System Algorithm in Multi-Objective Resource Allocation for Virtual Services. *Journal of Information and Telecommunication*, 1(4):319-333. Taylor & Francis, 2017.

6. Nguyen Minh Nhut Pham, Van Son Le, Ha Huy Cuong Nguyen. Energy-Efficient Resource Allocation for Virtual Service in Cloud Computing Environment. In *the 4th International conference on information system design and intelligent applications*, Advances in Intelligent Systems and Computing, vol 672: 126-136. Springer, 2018.

7. Nguyen Minh Nhut Pham, Van Son Le, Ha Huy Cuong Nguyen. Energy Efficient Resource Allocation for Virtual Service Based on Heterogeneous Shared Hosting Platforms in Cloud Computing. *Cybernetics and Information Technologies*, 17(3): 47-58. BAS, 2017.

INTRODUCTION

1. Statement of the Problem

Cloud Computing Model is the Internet-based development and the use of technology in various fields such as Grid Computing, Cluster Computing, Utility Computing, and Automatic Computing. The main purposes of these systems are to build an efficient computing platform and share integrated computer resources through network access, hardware, and software resources in order to increase performance and fault tolerance, ensure resource availability from separate computers.

While there have certainly been several positive advantages for information technology, the disadvantages Cloud Computing has brought about should not be overlooked. Negative outcomes that need to be addressed are allocating resources in terms of resource constraints, optimizing cost models, and achieving load balancing, etc.

Specially, the rapid needs for Physical Machines (hereinafter called PMs) to allocate resources for virtual services in Data Centers lead to the rapid growth of using PMs which results in increasing the energy consumption and CO_2 emission causing serious problems for the environment.

Consequently, optimization of PM Resource allocation for virtual services on shared hosting platform meeting the objectives of service quality, load balancing, and minimizing the physical resources used as well as the energy consumption play a meaningful role in Cloud Computing. That was the reason why the author decided to carry out the research: "***Resource Allocation for virtual service based on Shared Hosting Platform in Cloud Computing***".

2. Research Subjects and Scope

Resource allocating in Cloud Computing System can be classified into three different types of problem: application allocation problem, virtual machine allocation problem, and physical resource (PM) allocation problem for virtual services in order to create virtual machines for the cloud environment.

The main subjects of this dissertation are the resouce (PM) allocation models for virtual services to build virtual machines qualifying the needs of IaaS service providers. To satisfy the study objectives, Heuristic Algorithm, Particle Swarm Optimization Algorithm, Ant Colony Optimization Algorithm, and Simulated Annealing Algorithm were applied to solve the resource allocation problem.

3. Research Methodology

3.1 Theoretical Methodology

- A comprehensive search through the relevant literature was conducted. Articles from journals, conference proceedings, science reports, and published books about resource allocation in cloud computing systems were

collected, identified, and classified;

- Based on this comprehensive bibliography of the literature on cloud computing, the author analyzed, aggregated, and proposed the research problem. Concurrently, resource allocation algorithms for virtual services on the shared hosting platform were proposed grounded on Greedy Algorithm, Heuristic Algorithm, Particle Swarm Optimization Algorithm, Ant Colony Optimization Algorithm, and Simulated Annealing Algorithm.

3.2 Experimental Methodology

- Creating and selecting experimental data.

- Setting up and evaluating the proposed algorithms on experimental data using CloudSim, a Cloud Computing simulation tool, to experimentally implement, analyze, and evaluate the performances of the proposed algorithms.

4. Significance of the Study

- This dissertation is important for several reasons. First, this research built the resource allocation models for virtual services on shared hosting platform in order to create virtual machines and proposed the complexity algorithm in polynomial time to solve the problem. The proposed algorithms will thus improve the resource allocation performance in cloud computing system.

- This dissertation will be beneficial to the students and researchers who pay much attention to the resource allocation problem for virtual services and explore the heuristic method, approximation method to solve the optimization problems.

5. Research Contributions

(1) Based on the resource allocation model for virtual services, the resource needs model and resource model of Mark Stillwell, in other words, the mathematical model of the resource allocation problem for virtual services on a homogeneous shared hosting platform, with the objective of minimizing the used PMs and proposing the *MDRAVS-MMAS* algorithm which is relied on the *Max-Min Ant System* algorithm to estimate and compare the objective function value and the actual execution time of algorithm with that of the *First Fit* and *Best Fit* algorithm. The experimental results show that the used PMs of proposed *MDRAVS-MMAS* algorithm a less than *FFD* algorithm. The complexity of algorithm is implemented in polynomial time.

(2) This study extended the resource allocation problem on a homogeneous shared hosting platform by applying the resource model and resource needs model of Mark Stillwell, improved energy consumption model of Eugen Feller and built a mathematical model on a heterogeneous shared hosting platform with the objective of minimizing the energy consumption

of the system. The *ECRAVS-PSO* algorithm developed based on the *Particle Swarm Optimization* algorithm and the *ECRAVS-SA* algorithm derived from the *Simulated Annealing* algorithm were proposed to estimate the energy consumption and compare the objective function value and the actual execution time with that of the *FFD* algorithm. The experimental results show that the energy consumption of proposed *MDRAVS-MMAS* and *ECRAVS-SA* algorithms a better performance than FFD algorithm. The complexity of algorithms is implemented in polynomial time.

(3) Based on the system model, resource model, and the resource needs model of the resource allocation problem on a heterogeneous shared hosting platform, this study built the mathematical model to solve the multi-objective resource allocation problem. This problem focused on both balancing the load of PMs and minimizing the energy consumption. Developing the Ant Colony System algorithm, this study proposed the *MORA-ACS* algorithm to estimate and compare the objective function value and the actual execution time of algorithm with that of the *Round Robin* algorithm. The experimental results show that in the CloudSim environment, the MORA-ACS algorithm could balance the load as well as reduce the energy consumption better than the Round Robin algorithm. The complexity of algorithm is implemented in polynomial time.

7. Dissertation Structure

Apart from the Introduction and Conclusions, this dissertation is structured as follows:

- Chapter 1: *Overview of Resource Allocation for Virtual Service Problem.*

- Chapter 2: *Resource Allocation for Virtual Service Based on Homogeneous Shared Hosting Platform.*

- Chapter 3: *Resource Allocation for Virtual Service Based on Heterogeneous Shared Hosting Platform.*

- Chapter 4: *Multi-Objective Resource Allocation for Virtual Service Based on Heterogeneous Shared Hosting Platform.*

The dissertation results were presented in seven scientific studies publised in international and local journals and conferences in which one article was published in a national conference proceeding, two articles were published in national research journals, two articles in international research journals, and two articles in international research conference proceedings.

Chương 1.

OVERVIEW OF RESOURCE ALLOCATION FOR VIRTUAL SERVICE PROBLEM

1.1.Cloud Computing System

Peter Mell and Tim Grance defined the cloud computing system model as: "*a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.*"

- The five key characteristics of the cloud are: broad network access, on-demand self-service, resource pooling, rapid elasticity, and measured service.

- There are three service models, including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Besides, Hardware as a Service (HaaS) is also considered as a service model of cloud computing system.

- Based on the cloud deployment, the cloud deployment models classify into four types: Private Cloud, Public Cloud, Community Cloud and Hybrid Cloud.

1.2.Requests and challenges

- The requirements include: purpose of use, services, Virtualization, QoS Negotiation, user Interface, added value services.

- The challenges include: security, costing model, charging model, resource saving, service level agreement, virtual machine migration.

1.3.Virtual Machine

According to Goldberg, virtual machine is defined as "*an efficient, isolated duplicate of a real computer machine. Current use includes virtual machines which have no direct correspondence to any real hardware*".

1.4.Virtualization Technology

Virtualization Technology can be used to combine or divide computing resources in order to perform one or many operating environments. In Hierarchical Architecture of Virtualization Technology, the virtualization layer partitions the current resources of one or some lower-level physical servers to create several virtual machines. Nowadays, there are two types of virtualization technology: Partitioning Virtualization and Aggregation Virtualization.

1.5.Simulation Tools for Cloud Computing System

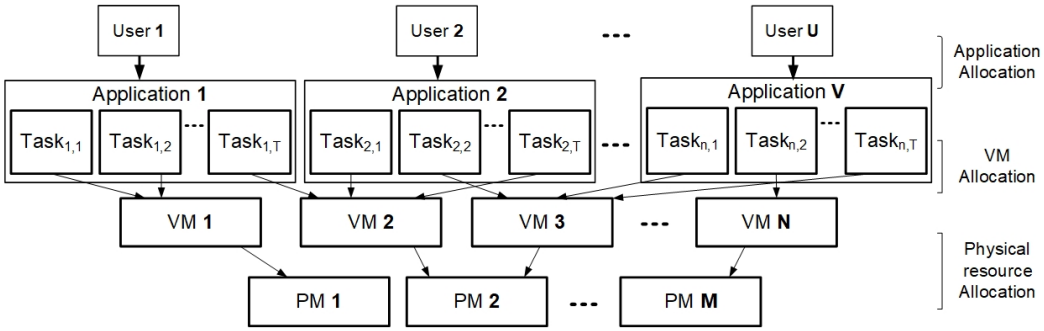
Deploying a real cloud or using a commercial cloud to test can be very expensive. Therefore, simulation tools which have the same features as of a

real cloud are the common solution in the test deployment. Based on the cost criteria, function and experimental requirements of the research, the author decided to choose CloudSim to experimentally deploy and evaluate the algorithms to solve the proposed problems.

1.6.Resource Allocation for Cloud Computing System

1.6.1.Resource Allocation Model

Figure 1.1 presents the resource allocation model in cloud computing. In this model, V is the number of applications, and each application has T tasks. Each application receives demands from users. These applications run on N virtual machines, and the virtual machines are allocated resources from M physical machines.



Hình 1.1: Resource Allocation Model in Cloud Computing System.

As such, resource allocation in cloud computing specifies into three kinds: application allocation, virtual machine allocation, and physical resource allocation.

- Application allocation: When receiving the demands for using services, the system will deploy the applications on virtual machines and meanwhile receive the demands from the next applications.

- Virtual machine allocation: Virtual machine allocation is a process of creating a set of virtual machines for providing hardware and software resources in order to deploy the applications.

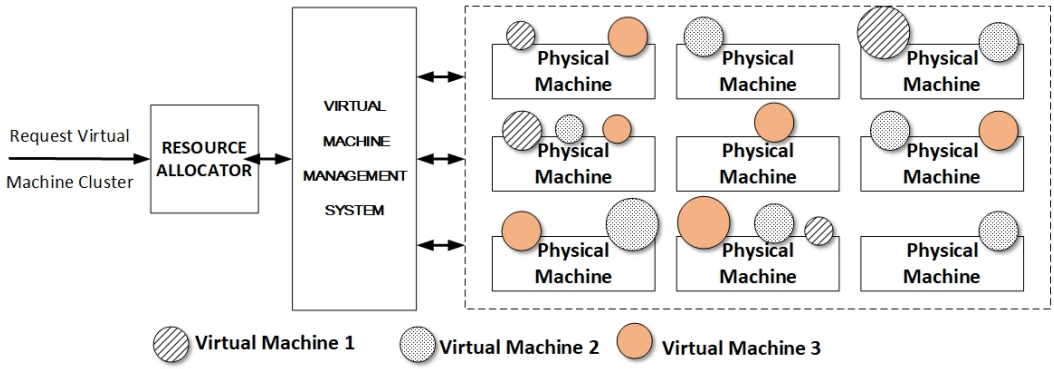
- Physical resource allocation: Physical resource allocation is to provide hardware resources for virtual machines based on resource needs for virtual services.

1.7.Related Works of Resource Allocation for Virtual Services

1.7.1.System Model of Resource Allocation for Virtual Services

Research Objective is a static resource allocation problem, static resource allocation model based on a shared hosting platform of Mark Stillwell by Figure 1.2.

- **A shared hosting platform**: A shared platforms includes a cluster of physical machines is interconnected by a network device is deployed for



Hình 1.2: Resource Allocation Model for Virtual Services based on Shared Hosting Platform.

sharing resources to virtual services. If physical machines has different resource configuration, this system is called a heterogeneous shared platform. In contrast, this system is called a homogeneous shared platform.

- **Virtual Service**: Each virtual service is a resource mapping $f : \mathbb{S} \rightarrow \mathbb{R}$. So, with $s_k \in \mathbb{S}$ and $r_k \in \mathbb{R}$ then $f(s_k) = r_k$ if r_k represents the real resource corresponding to the virtual resource s_k of type k. Inside, $\mathbb{S} = \{s_k | k = 1...D\}$ be the virtual resource set on the virtual machine, $\mathbb{R} = \{r_k | k = 1, ..., D\}$ be the real resource on physical machines and D be the number of resource type.

- **Virtual Machine Manager**: Virtual Machine run on physical machine that are each under the control of a VM monitor. The VM monitor can enforce specific resource consumption rates for different VMs running on the physical machine.

- **Virtual Machine Manager System**: All VM monitors are under the control of a VM management system that can specify resource consumption rates for VMs running on the physical machines. Furthermore, the VM resource management system can enact VM instance migrations among physical machines.

1.7.2. Model of Resource Allocation for Virtual Services with the objective of minimizing the used PMs

1.7.3. Model of Resource Allocation for Virtual Services with the objective of minimizing the energy consumption of the system

1.7.4. Model of Resource Allocation for Virtual Services with the objective of balancing the load of PMs and minimizing the energy consumption

1.8. Research Objectives and Content

1.8.1. Research Objectives

This dissertation proposes the optimal solution for Multi-Dimensional Resource Allocation problem (many types of resources, such as: CPU, RAM,

Disk,...) on shared hosting platform for virtual services in data centers of cloud computing which aims at minimizing the number of used PMs, reducing the energy consumption of the system, and balancing the load of PMs.

1.8.2.Research Content

The research Content are implemented through 03 specific problems as follows:

- **Problem 1**: Multi-Dimensional Resource Allocation for Virtual Services – MDRAVS.

- **Bài toán 2**: Energy Consumption Resource Allocation for Virtual Services - ECRAVS. This is the expansion of Problem 1.

- **Bài toán 3**: Multi-Objective Resource Allocation - MORA. This is the expansion of Problem 2.

To carry out the research, the perspectives of the research are as follows:

- Applying the resource allocation model as presented in Figure 1.2;

- Building the mathematical model as an optimization problem;

- Proposing the Meta-heuristic algorithms to solve the problem;

- Creating and selecting the experimental data relevant to a real cloud.

1.9.Summary of Chapter 1

This chapter presents the overview of cloud computing system. Based on the models and virtualization technology, this dissertation shows the thorough results of the resource allocation problems in cloud computing as well as the other related problems.

Allocating resource in cloud computing system can be classified into three classes of problems: application allocation problem, virtual machine allocation problem, and physical resource (PM) allocation problem for virtual machines in order to create virtual machines for the cloud. However, alomost solutions and models focus on the application allocation and virtual machine allocation problems, a limited study has been done to explore the physical resource allocation problems, yet these problems only examine CPU and memory. Therefore, this dissertation concentrates on the multi-dimensional resource allocation problem for virtual services in cloud computing which aims at minimizing the number of used PMs, reducing the energy consumption of the system, and balancing the load of PMs.

Based on the cost criteria, functions of a simulation tool, and experimental requirements, the CloudSim simulation tool has been chosen to experimentally deploy and evaluate the algorithms to solve the proposed problems.

Chương 2.

RESOURCE ALLOCATION FOR VIRTUAL SERVICE BASED ON HOMOGENEOUS SHARED HOSTING PLATFORM

2.1.Resource and Resource Needs Model

To supply resource needs for virtual services, each physical machine provides several resources, i.e., CPU, RAM, Disk,... The resource needs of each virtual service are categorized into two groups: *rigid needs* and *fluid needs*.

- *A rigid needs* represents a specific fraction of required resource. The service cannot benefit from a larger fraction and cannot operate with a smaller fraction than a rigid need.

- *A fluid needs* specifies the maximum fraction of a resource that the service could use if alone on the server. The service cannot benefit from a larger fraction, but can operate with a smaller fraction than a fluid needs if the cost is reduced.

- The ratio between the allocated resource and the fluid resource needs is known as the yield of the fluid resource need. and we call their value simply the service yield.

2.2.MDRAVS Problem

2.2.1.Statement of MDRAVS Problem

Multi-Dimensional Resource Allocation for Virtual Services Based on Homogeneous Shared Hosting Platform (MDRAVS) is formulated as follows:

$$x_{ij}, y_j, a_{ik} \in \{0,1\}, b_{ij}, r_{ik} \in \mathbb{Q}^+, \quad \forall i \in \mathbb{VS}, j \in \mathbb{PM}, k \in \mathbb{D} \tag{2.1}$$

$$\sum_{j=1}^{M} x_{ij} = 1, \quad \forall i \in \mathbb{VS} \tag{2.2}$$

$$y_j \geq x_{ij}, \quad \forall i \in \mathbb{VS}, j \in \mathbb{PM} \tag{2.3}$$

$$\sum_{i=1}^{N} (((b_{ij} \times (1 - a_{ik}) + a_{ik}) \times r_{ik}) \times x_{ij}) \leq 1, \quad \forall k \in \mathbb{D}, j \in \mathbb{PM} \tag{2.4}$$

$$\sum_{j=1}^{M} y_j \to min \tag{2.5}$$

2.2.2.MDRAVS Problem Complexity

Định nghĩa 2.1. To determine the computational complexity of MDRAVS problem, let's consider MDRAVS-Dec, the decision problem associated with MDRAVS can be stated as: Is it possible to for resource allocation from M physical machines to N virtual services, each of which has a resource needs r_{ik}?.

Mệnh đề 2.1. *The decision problem MDRAVS-Dec is a NP-C.*

2.3.Proposing Solutions for Solving the MDRAVS Problem

2.3.1.Applying *First Fit* Algorithm and *Best Fit* Algorithm

To solve the MDRAVS Problem, *First Fit* and *Best Fit* algorithm were applied and presented as in Algorithm 1. This result was published in the Journal of Computer Science and Cybernetics in 2014 (article 1).

Algorithm 1: *First Fit, Best Fit algorithms.*

Input :

- A set of virtual services $\mathbb{VS} = \{i \mid i = 1, ..., N\}$, each service has resource need r_{ik} corresponding to the type of needs a a_{ik}, a set of resource types $\mathbb{D}$.
- A set of physical machines $\mathbb{PM} = \{j \mid j = 1, ..., M\}$ corresponding service yield b_{ij}.

Output: A set of used physical machines (respectively, x_{ij} =1)).

1 Based on the resource needs of virtual services $\mathbb{VS}$, building N vector with r_{ik} elements.

2 Apply Criteria to sort elements of vector N vector in a descending order $\mathbb{VS}$.

3 Apply the *First Fit* or *Best Fit* algorithm to place the elements of N vector into the physical machines, such that: $\sum_{i=1}^{N} (b_{ij} \times (1 - a_{ik}) + a_{ik}) \times r_{ik} \times x_{ij} \leq 1, \quad \forall k, j$

4 If the resource needs have not completed then go to Step 3.

5 If the resource needs have been completed then the output is a set of used physical machines (this is the outcome of the objective function of the MDRAVS problem).

Before putting elements of $\mathbb{VS}$ into $\mathbb{PM}$, elements $\mathbb{VS}$ are sorted in a descending order based on the following criterias: Lexicographical, Maximum Component, Maximum Sum.

The combinatorial algorithms constructed from greedy algorithms, *First Fit*, *Best Fit* based on the 3 Criteria, therefore we have 06 algorithms including: *BestFitDesSum*, *BestFitDesMax*, *BestFitDesLex*, *FirstFitDesSum*, *FirstFitDesMax* và *FirstFitDesLex*.

If D is considered as a constant, those algorithms have the computational complexity of $\mathcal{O}(N \times M)$

2.3.2.Proposing the *MDRAVS-MMAS* Algorithm

Based on the *Max-Min Ant System* algorithm, the MDRAVS-MMAS algorithm was proposed and presented as in Algorithm 2. This result was published in article 2.

- The MDRAVS problem described in Section 2.2.1 is a static resource allocation problem. Thus, at the time of resource allocation, a set of virtual services $\mathbb{VS}$ with resource needs r_i^*, r_i^{**}, a set of physical machines $\mathbb{PM}$ as well as the resource capacity of the defined physical machine.

- Let $P(numLoop)$ be the probability of finding the solution of the algorithm after $numLoop$ iterations, Stutzle and Dorigo proved that:

$\forall \varepsilon > 0$, if $numLoop$ is big enough then $\lim_{numLoop \to \infty} P(numLoop) = 1$. Therefore, the *MDRAVS-MMAS* algorithm will converge after $numLoop$

iterations.

- If we fix the number of types of resources D, the complexity of the algorithm is $\mathcal{O}(numLoop \times numAnt \times N \times M)$.

Algorithm 2: *MDRAVS-MMAS Algorithm.*

Input :

- A set of virtual services $\mathbb{VS}$, vector r_i^*, r_i^{**}. A set of physical machines $\mathbb{PM}$.
- Parameters: $\alpha, \beta, \rho, p^{best}, \tau^{max}$, $numLoop$ and $numAnt$.

Output: Best solution, $S^{best-toan-cuc}$, is used physical machines.

Set the parameter values: $\alpha, \beta, \rho, p^{best}, \tau^{max}$, $\tau_{ij} = \tau^{max}$;
for $nL := 1 \rightarrow numLoop$ **do**
  for $nA := 1 \rightarrow numAnt$ **do**
    $j := 1$;
    $BINARY^{nA} := [e_{ij} := 0], \forall i \in \{1, ..., N\}, \forall j \in \{1, ..., M\}$;
    while $(\mathbb{VS} \neq \emptyset)$ **do**
      $\mathbb{VS}^{temp} = \left\{ i \in \mathbb{VS} \middle| \sum_{j=1}^{M} x_{ij} = 0 \wedge R_j + r_i^* + r_i^{**} \leq C_j, \forall j \right\}$;
      if $(\mathbb{VS}^{temp} \neq \emptyset)$ **then**
        random choice a virtual machine $i \in \mathbb{VS}^{temp}$ by $p_{ij} := \frac{[\tau_{ij}]^\alpha \times [\eta_{ij}]^\beta}{\sum_{u \in \mathbb{VS}^{temp}} [\tau_{uj}]^\alpha \times [\eta_{uj}]^\beta}, \forall i \in \mathbb{VS}^{temp}$;
        $e_{ij} := 1$; $\mathbb{VS} := \mathbb{VS} - \{i\}$; $R_j := R_j + (r_i^* + r_i^{**})$;
      else
        $j := j + 1$;
  Compare solutions $BINARY^{nA}$ base on $f(S^{best})$ and store a best solution in $S^{best-vong-lap}$;
  if $(nL = 1 \vee Best - Toan - Cuc(S^{best-vong-lap}))$ **then**
    $S^{best-toan-cuc} := S^{best-vong-lap}$; /* This is a new best global. A function $Best - Toan - Cuc(S^{best-vong-lap})$ check $S^{best-vong-lap}$ which is a best solution. */
  Calculate τ^{min}, τ^{max};
  foreach $(i, j) \in (\mathbb{VS} \times \mathbb{PM})$ **do**
    $\tau_{ij} := \rho \times \tau_{ij} + \Delta\tau_{ij}^{best}$;
    if $(\tau_{ij} > \tau^{max})$ **then**
      $\tau_{ij} := \tau^{max}$;
    if $(\tau_{ij} < \tau^{min})$ **then**
      $\tau_{ij} = \tau^{min}$;
return A Best global Solution $S^{best-toan-cuc}$;

2.3.3.Experiments and Evaluations

2.3.3.1.Simulation Method

In order to evaluate the algorithms, 4500 input samples were created by the random method.

2.3.3.2. Experimental Result Evaluations of *First Fit*, *Best Fit* Algorithms

The experimental results of the six versions of the *First Fit* algorithm and *Best Fit* algorithm are shown in Table 2.1 and 2.2.

Bảng 2.1: The execution times of *First Fit*, *Best Fit* algorithms.

Algorithms	**N=32**	**N=64**	**N=128**	**N=256**	**N=512**
FirstFitDesMax	0,00009	0,00107	0,00303	0,01076	0,03193
FirstFitDesLex	0,00010	0,00114	0,00214	0,00827	0,02529
FirstFitDesSum	0,00016	0,00113	0,00268	0,00932	0,02791
BestFitDesMax	0,00121	0,00254	0,00833	0,03741	0,03741
BestFitDesLex	0,00123	0,00232	0,00783	0,03500	0,12253
BestFitDesSum	0,00128	0,00234	0,00800	0,03693	0,13380

Bảng 2.2: The used physical machines of *First Fit*, *Best Fit* algorithms.

Algorithms	**N=32**	**N=64**	**N=128**	**N=256**	**N=512**
FirstFitDesMax	24	47	90	174	344
FirstFitDesLex	24	47	90	174	344
FirstFitDesSum	24	47	90	174	344
BestFitDesMax	24	47	90	174	344
BestFitDesLex	24	47	89	170	327
BestFitDesSum	24	47	90	174	344

2.3.3.3. Experimental Result Evaluations of *MDRAVS-MMAS* Algorithm, *First Fit* Algorithm and *Best Fit* Algorithm

The parameter values of the *MDRAVS-MMAS* algorithm are: $\alpha = 1,\ \beta = 2,\ \rho = 0,9,\ p^{best} = 0{,}05,\ \tau^{max} = 3,\ numLoop = 5,\ numAnt = 5$.

The experimental results of the *MDRAVS-MMAS* algorithm, *First Fit* algorithm and *Best Fit* algorithm are shown in Table 2.3 and Table 2.4.

Bảng 2.3: The used physical machines of *MDRAVS-MMAS* algorithm and another algorithms.

Algorithms	**N=32**	**N=64**	**N=128**	**N=256**	**N=512**
FirstFit	24	47	90	174	344
BestFit	24	47	89,7	172,7	338,3
MDRAVS-MMAS	24	47	88	172	323

2.4. Summary of Chapter 2

The content of this chapter is to build a resource allocation problem as part of the resource allocation set, define the mathematical model, and

Bảng 2.4: The execution times of *MDRAVS-MMAS* algorithm and another algorithms.

Algorithms	**N=32**	**N=64**	**N=128**	**N=256**	**N=512**
FirstFit	0,000117	0,001113	0,002617	0,009450	0,028377
BestFit	0,001240	0,002400	0,006553	0,036447	0,129133
MDRAVS-MMAS	0,001000	0,003000	0,020890	0,040120	0,074100

propose the algorithm to provide multi-dimensional resource (many types of resources, such as: CPU, RAM, Disk...) on Homogeneous Shared Hosting Platform for virtual services with the objective of minimizing the number of used PMs.

- Based on the standard algorithms to solve the Vector Packing problem, six different algorithm versions derived from the *First Fit* and *Best Fit* algorithm were proposed and evaluated. These algorithms were presented in the Journal of Computer Science and Cybernetics in 2014 (article 1).

- Based on the *Max-Min Ant System* Optimization algorithm, the *MDRAVS-MMAS* was proposed. This algorithm was published in article 2.

All the algorithms were evaluated by using the simulation scripts, the experimental data were created by the random probability method, the comparison results of the algorithms were based on two metrics: The number of used PMs (the objective function of the problem) and the algorithm's execution time.

The experimental results revealed that:

- The complexity of the algorithms is solvable in polynomial time;

- When the number (problem range) of virtual services is small, there is a little difference among the algorithm metrics.

- Yet if the number is large, the results among the algorithms are not the same: The *MDRAVS-MMAS* algorithm produced a better objective function value than the *Best Fit* algorithm. In addition, due to the short execution time, the algorithms are suitable when deploying in a real cloud computing system.

Chương 3.

RESOURCE ALLOCATION FOR VIRTUAL SERVICE BASED ON HETEROGENEOUS SHARED HOSTING PLATFORM

3.1.Resource Model and Resource Needs

Different to Chapter 2 (examining on Homogeneous Shared Hosting Platform), research for Chapter 3 was carried out on Heterogeneous Shared Hosting Platform which includes a cluster of PMs having different resource configuration and being interconnected by a high-speed network device to share resources for virtual services. Thus, the resource model and resource needs are extended as follows:

A heterogeneous shared platform including a set of $\mathbb{PM}$ physical machines, $\mathbb{PM} = \{j \mid j = 1, ..., M\}$, which has different resource configuration, and being interconnected by a high-speed network device which is deployed for sharing resources to set of $\mathbb{VS}$ virtual services, $\mathbb{VS} = \{i \mid i = 1, ..., N\}$, is carried out in this paper. Each physical machine provides a set of $\mathbb{D}$ types of resource, $\mathbb{D} = \{k \mid k = 1, ..., D\}$, where N is the number of virtual services and M is the number of physical machines and D be the dimension of resources.

Each type of ability at an objective engine might have one or more distinct elements such as one or more separate real CPU, one or more single substantial memory and aggregate one. Thus, the capabilities of a corporeal appliance are represented by an ordered pair of multi-dimensional resource vectors. The elementary resource vector expresses the capacity of a single aspect while the aggregate resource vector executes sum of resource capacity counting all factors. Accordingly, the resources of physical machine reached as an ordered pair of multi-dimensional resource vectors (C^e, C^a). In which $C^e = \{c^e_{jk} \mid c^e_{jk} \in \mathbb{Q}^+; j = 1, ..., M; k = 1, ..., D\}$ and $C^a = \{c^a_{jk} \mid c^a_{jk} \in \mathbb{Q}^+; j = 1, ..., M; k = 1, ..., D\}$. And, c^e_{jk}, c^a_{jk} are items which represent the elementary resource and aggregate resource of a physical machine j for a resource type k, respectively.

Similarly, the virtual service′s resource needs are performed by involving the elementary and aggregate one. Indeed, the resource needs of each virtual service are categorized into two groups: *rigid needs* and *fluid needs*.

Hence, the rigid needs of a virtual service i for a resource type k are represented by a first ordered vector pair (R^e, R^a) which determines the resource demands in running the virtual service at the acceptable level. In which, $R^e = \{r^e_{ik} \mid r^e_{ik} \in \mathbb{Q}^+; i = 1, ..., N; k = 1, ..., D\}$ and $R^a = \{r^a_{ik} \mid r^a_{ik} \in \mathbb{Q}^+; i = 1, ..., N; k = 1, ..., D\}$. And, r^e_{ik}, r^a_{ik} are items which are to denote the elementary and aggregate rigid needs of a resource type k of a virtual service i, respectively.

The fluid needs of a virtual service i for a resource type k are repre-

sented by a second ordered vector pair (F^e, F^a), which displays the additional resource demand when running the virtual service at the maximum performance level. In which, $F^e = \{f^e_{ik} \mid f^e_{ik} \in \mathbb{Q}^+; i = 1, ..., N; k = 1, ..., D\}$ and $F^a = \{f^a_{ik} \mid f^a_{ik} \in \mathbb{Q}^+; i = 1, ..., N; k = 1, ..., D\}$. And, f^e_{ik}, f^a_{ik} are items which are to denote the elementary and aggregate fluid needs of a resource type k of a virtual service i, respectively.

Therefore, resource needs of resource type k of virtual service i on physical machine j with additional factor are given by an ordered vector pair $(R^e + Q \times F^e, R^a + Q \times F^a)$. In which, $Q = \{q_{ij} \mid q_{ij} \in \mathbb{Q}^+; i = 1, ..., N; j = 1, ..., M\}$ is a vector of additional factor of virtual services when the user requests. And, q_{ij} intends the additional factor of virtual service i on physical machine j

3.2.Energy Consumption Model

Consider the resource allocation system for virtual services as presented in Section 1.7.1, and the resource model and resource needs for virtual services as presented in Section 3.1.

In order to estimate the energy consumption of the physical machines, we choose approximately the power consumption at a physical machine j as a linear function $P_j(u_j)$ by the formula (3.1).

$$P_j\left(u_j\right) = \left(P_j^{max} - P_j^{idle}\right) \times u_j + P_j^{idle}, \quad \forall j \in \mathbb{PM} \tag{3.1}$$

Among them, $P_j^{max} \in \mathbb{Q}^+$ and $P_j^{idle} \in \mathbb{Q}^+$ are the power of physical machine j in the maximum used utilities state and idle state, respectively. $u_j \in [0, 1]$ is the total used utilities of all resources on the physical machines j and it is calculated by a formula (3.2).

$$u_j = \sum_{k=1}^{D} \frac{u_{jk}}{c^a_{jk}} = \sum_{k=1}^{D} \frac{\sum\limits_{i=1}^{N} \left(r^a_{ik} + q_{ij} \times f^a_{ik}\right) \times x_{ij}}{c^a_{jk}}, \qquad \forall j \in \mathbb{PM} \tag{3.2}$$

Therefore, the energy consumption of the M physical machines in the period t is set as a formula (3.3).

$$E\left(t\right) = \Delta t \times \sum_{j=1}^{M} P\left(u_j\right) \tag{3.3}$$

3.3.Statement of ECRAVS Problem

The problem of Energy Consumption Resource Allocation for Virtual Services – ECRAVS is stated as follows:

$$x_{ij} \in \{0, 1\}; q_{ij}, r^e_{ik}, r^a_{ik}, f^e_{ik}, f^a_{ik}, c^e_{ik}, c^a_{ik} \in \mathbb{Q}^+, \qquad \forall i \in \mathbb{VS}, j \in \mathbb{PM}, k \in \mathbb{D} \tag{3.4}$$

$$\sum_{j=1}^{M} x_{ij} = 1, \quad \forall i \in \mathbb{VS} \tag{3.5}$$

$$(r_{ik}^{e} + q_{ij} \times f_{ik}^{e}) \times x_{ij} \leq c_{jk}^{e}, \quad \forall i \in \mathbb{VS}, j \in \mathbb{PM}, k \in \mathbb{D} \tag{3.6}$$

$$\sum_{i=1}^{N} (r_{ik}^{a} + q_{ij} \times f_{ik}^{a}) \times x_{ij} \leq c_{jk}^{a}, \quad \forall j \in \mathbb{PM}, k \in \mathbb{D} \tag{3.7}$$

$$\left(\sum_{j=1}^{M}\left(\sum_{k=1}^{D}\left(\left(P_j^{max} - P_j^{idle}\right) \times \frac{\sum_{i=1}^{N}(r_{ik}^{a} + q_{ij} \times f_{ik}^{a}) \times x_{ij}}{c_{jk}^{a}} + P_j^{idle}\right)\right)\right) \rightarrow \min \tag{3.8}$$

3.4.Proposing Solutions for Solving ECRAVS Problem

3.4.1.Proposing *ECRAVS-PSO* Algorithm

Based on the Particle Swarm Optimization Algorithm, the *ECRAVS-PSO* algorithm was proposed to solve the ECRAVS Problem. The algorithm detail is presented as in Algorithm 3. This result was publised in article 4.

- The ECRAVS problem described in Section 3.3 is a static resource allocation problem. Thus, at the time of resource allocation, a virtual services set $\mathbb{VS}$ with resource needs r_i^*, r_i^{**}, a set of physical machines $\mathbb{PM}$ as well as the resource capacity of the defined physical machine.

- Van den Bergh proved the convergence of the *Particle Swarm Optimization* algorithm as follows: Let x^* be the overall minimum value of the objective function in the solution space S, $x_{numLoop}$ be the solution series built by the *Particle Swarm Optimization* Algorithm after *numLoop* iterations. The probability is $P(\lim_{numLoop \to \infty} x_{numLoop} \neq x^*) > 0$. Hence, the *ECRAVS-PSO* algorithm converges to the overall minimum value after *numLoop* iterations.

- Let N be the number of virtual services, M be the number of PMs, *numPartical* be the number of *particle*, D be the number of types of resources and *numLoop* là số lần lặp. be the number of iterations. Fix the number of types of resources, the complexity of the *ECRAVS-PSO* algorithm is: $\mathcal{O}(numLoop \times numPartical \times N \times M)$.

3.4.2.Proposing *ECRAVS-SA* Algorithm

Based on the *Simulated Annealing* Algorithm, the *ECRAVS-SA* algorithm was proposed to solve the ECRAVS Problem. The algorithm detail is presented as in Algorithm 4. This result was published in article 6 and 7.

- The ECRAVS problem described in Section 3.3 is a static resource allocation problem. Thus, at the time of resource allocation, a virtual services set $\mathbb{VS}$ with resource needs r_i^*, r_i^{**}, a set of physical machines $\mathbb{PM}$ as well as the resource capacity of the defined physical machine.

Algorithm 3: *The ECRAVS-PSO Algorithm.*

Input :

- A set of virtual services $\mathbb{VS} = \{i \mid i = 1, ..., N\}$, a set of resource types $\mathbb{D} = \{k \mid k = 1, ..., D\}$, resource needs: r^e_{ik},r^a_{ik},f^e_{ik},f^a_{ik} and additional factor q_{ij}.
- A set of physical machines $\mathbb{PM} = \{j \mid j = 1, ..., M\}$, resource of physical machines: c^e_{jk}, c^a_{jk}.
- Number of *particle*, *numParticle* and Number of Loops, *numLoop*.

Output: The best list of virtual services is allocated resource, *gBest*.

```
int numP: = numParticle;
new gBest [M] ;
new pBest [numP] [M];
new particleLocal [numP] [M];
boolean new particleVelocity [numP] [M];
while (nL ≤ numLoop) do
    for (nP := 1 → numP) do
        particalLocal [nP] := initSolution (nP, PM, VS) foreach (j ∈ PM) do
            if (particleLocal [nP] [j] .size = 0) then
                particleVelocity [numP] [M] := 0;
            particleVelocity [numP] [M] := 1;
    for (nP := 1 → numP) do
        for (nQ := 1 → parLoca[nP].length) do
            pBest [nP] [nQ] := particalLocal [nP] [nQ];
    gBest := globalBestParticle(particalLocal, PM)
    for (nP := 1 → numP) do
        particleVelocity [nP] := speedUpdate();
        particleLocal [nP] := positionUpdate();
        pBest[nP] := PBestUpdate(pBest[nP]);
        gBest := GBestUpdate(gBest);
    numLoop + +;
return gBest;
```

- Mitra et al. proved that the *Simulated Annealing* converges to the overall optimal solution with the probability of 1. Therefore, the ECRAVS-SA algorithm also converges after finite iterations (i.e. $T = T^{min}$).

- Let N be the number of virtual services, M be the number of PMs, and D be the number of types of resources. Fix the number of types of resources, the complexity of the *ECRAVS-SA* algorithm is $\mathcal{O}(T \times numLoop \times N \times M)$.

3.4.3.Experiments and Evaluations

3.4.3.1.Simulation Method:

In order to evaluate the algorithms, this dissertation compared the proposed algorithms with the *FFD* algorithm in the CloudSim environment.

- **Parameter values of the *ECRAVS-PSO* algorithm**: From experiment, when the number of *particles* is 20 and the number of iterations *numLoop* is 10, the algorithm produces the best objective function value.

- **Parameter values of the *ECRAVS-SA* algorithm**: From exper-

Algorithm 4: *The ECRAVS-SA Algorithm.*

Input :

- A set of virtual services $\mathbb{VS} = \{i \mid i = 1, ..., N\}$, a set of resource types $\mathbb{D} = \{k \mid k = 1, ..., D\}$, resource needs: r^e_{ik},r^a_{ik},f^e_{ik},f^a_{ik} and additional factor q_{ij}.
- A set of physical machines $\mathbb{PM} = \{j \mid j = 1, ..., M\}$, resource of physical machines: c^e_{jk}, c^a_{jk}.
- T^0, T^{min}, $numLoop$ and CR.

Output: list of used PMs, S^{best}.

```
double T := T^0;
S^best := initializeSolution(PM, VS) ;    /* Execute initialization solution by an FFD */
Calulate E^best ;
while (T > T^min) do
    for (nL := 1 → numLoop) do
        S^current := S^best;
        Calulate E^current ;
        S^neighbor := currentNeighborSolution(M, N);
        Calulate E^neighbor ;
        if (E^neighbor < E^current) then
            S^current := S^neighbor;
            Calulate E^current ;
        if (exp((E^neighbor − E^current)/T) > random(0, 1)) then
            S^current := S^neighbor;
            Calulate E^current ;
        if (E^current < E^best) then
            S^best := S^current;
            Calulate E^current ;
    T := T × (1 − CR) ;                    /* CR is a Cooling Rate */
return S^best;
```

iment, when T=1000, T^{min}=0, CR=5 and $numLoop$= 100, the algorithm produces the best objective function value.

3.4.3.2.Experimental Results and Evaluations

The energy consumption in a time period of $\Delta t = 24$ hours. Measure unit of energy consumption is kWh, and the runtime of algorithm is calculated by second(s).

3.5.Summary of Chapter 3

This chapter models one problems as the linear programming problems with the rational variables and integer variables. To estimate the problems, the author proposed two algorithms to solve and compare. The details are as follows:

The problem model was developed by calculating the energy consumption of all resources (All the current studies only calculate the energy consumption of CPU). The author proposed two algorithms: The *ECRAVS-*

Bång 3.1: Experimental Results of *ECRAVS-PSO*, *ECRAVS-SA*, *FFD* algorithm.

Num. of VM	**Algorithms**	**Times(s)**	**Energy(kWh)**	**Gain Energy(%)**
	FFD	0,031	201,284	
100	*ECRAVS-SA*	0,038	193,000	4,292
	ECRAVS-PSO	0,037	190,726	5,536
	FFD	0,078	396,706	
200	*ECRAVS-SA*	0,088	392,490	1,074
	ECRAVS-PSO	0,084	390,292	1,643
	FFD	0,116	597,989	
300	*ECRAVS-SA*	0,125	584,564	2,297
	ECRAVS-PSO	0,121	574,440	4,099
	FFD	0,144	793,411	
400	*ECRAVS-SA*	0,160	772,185	2,749
	ECRAVS-PSO	0,150	770,908	2,919
	FFD	0,200	994,694	
500	*ECRAVS-SA*	0,218	972,439	2,289
	ECRAVS-PSO	0,216	960,156	3,597

SA algorithm derived from the *Simulated Annealing* algorithm and the *ECRAVS-PSO* algorithm based on the the *Particle Swarm Optimization* algorithm. Using real data for the experiment, the results were compared with that of the *FFD* algorithm indicating that the objective function value from the two proposed algorithms is better than from the *FFD* algorithm.

Although the execution time of the proposed algorithms is longer due to the overall optimization method, these algorithms still ensure to execute in polynomial time and is able to perform in a real cloud. These results were published in article 3, 4, 6 and 7.

Chương 4.

MULTI-OBJECTIVE RESOURCE ALLOCATION FOR VIRTUAL SERVICE BASED ON HETEROGENEOUS SHARED HOSTING PLATFORM

4.1.Load Balancing Model

According to Zongqin et al., resource allocation for load balancing purpose are calculated through the *standard deviation* of the remaining resources on physical machines.

$$\sigma = \sqrt{\frac{1}{M} \times \sum_{j=1}^{M} \left(R_j - \overline{R}\right)^2} \tag{4.1}$$

4.2.Statement of MORA Problem

The problem of Multi-Objective Resource Allocation - MORA for virtual services based on Heterogeneous Shared Hosting Platform is presented as follows:

$$x_{ij} \in \{0,1\}; q_{ij}, r^e_{ik}, r^a_{ik}, f^e_{ik}, f^a_{ik}, c^e_{ik}, c^a_{ik} \in \mathbb{Q}^+, \quad \forall i \in \mathbb{VS}, j \in \mathbb{PM}, k \in \mathbb{D} \tag{4.2}$$

$$\sum_{j=1}^{M} x_{ij} = 1, \quad \forall i \in \mathbb{VS} \tag{4.3}$$

$$(r^e_{ik} + q_{ij} \times f^e_{ik}) \times x_{ij} \leq c^e_{jk}, \quad \forall i \in \mathbb{VS}, j \in \mathbb{PM}, k \in \mathbb{D} \tag{4.4}$$

$$\sum_{i=1}^{N} (r^a_{ik} + q_{ij} \times f^a_{ik}) \times x_{ij} \leq c^a_{jk}, \quad \forall j \in \mathbb{PM}, k \in \mathbb{D} \tag{4.5}$$

$$R_j = \sum_{k=1}^{D} \left(c^a_{jk} - \sum_{i=1}^{N} (r^a_{ik} + q_{ij} \times f^a_{ik}) \times x_{ij} \right), \quad \forall j \in \mathbb{PM} \tag{4.6}$$

$$\overline{R} = \frac{1}{M} \times \sum_{j=1}^{M} R_j \tag{4.7}$$

$$\sigma = \sqrt{\frac{1}{M} \times \sum_{j=1}^{M} \left(R_j - \overline{R}\right)^2} \rightarrow \min \tag{4.8}$$

$$\left(\sum_{j=1}^{M}\left(\sum_{k=1}^{D}\left(\left(P_j^{max}-P_j^{idle}\right)\times\frac{\sum_{i=1}^{N}\left(r_{ik}^a+q_{ij}\times f_{ik}^a\right)\times x_{ij}}{c_{jk}^a}+P_j^{idle}\right)\right)\right)\rightarrow\min \tag{4.9}$$

4.3.Proposing Solutions for Solving MORA Problem

4.3.1.Pareto-optimal Solution

If a solution is not dominated by other solutions in the solution space then it is called the Pareto optimization solution. The set of non-dominated feasible solutions in the solution space is called the Pareto optimization set.

With a Pareto optimization set, the set of objective function values corresponding to each solution in the solution space is called the Pareto Front set.

4.3.2.Proposing *MORA-ACS* Algorithm

Based on the Ant Colony System algorithm, the *MORA-ACS* algorithm was proposed to solve the MORA Problem. The algorithm detail is presented as in Algorithm 5. This result was publised in article 5.

- The MORA problem described in Section 4.2 is a static resource allocation problem. Thus, at the time of resource allocation, a virtual services set $\mathbb{VS}$ with resource needs r_i^*, r_i^{**}, a set of physical machines $\mathbb{PM}$ as well as the resource capacity of the defined physical machine.

- Let $P(numLoop)$ be the probability of finding the solution of the *Ant Colony System* algorithm after $numLoop$ iterations, using the heterogeneous Markov model Stutzle and Dorigo proved that: $\forall\varepsilon > 0$, if $numLoop$ is big enough, then $P(numLoop) > 1-\varepsilon$.

Therefore, $\lim\limits_{numLoop\rightarrow\infty} P\left(numLoop\right) = 1$. This proved that the *MORA-ACS* algorithm converges after $numLoop$ iterations.

- Let N be the number of virtual services, M be the number of PMs, D be the number of types of resources, $numLoop$ be the number of iterations, and $numAnt$ be the number of ants.

Fix the number of types of resources, the complexity of the MORA-ACS algorithm is $\mathcal{O}(numLoop \times numAnt \times N \times M)$.

4.3.3.Experiments and Evaluations

4.3.3.1.Simulation Method

To assess the *MORA-ACS* algorithm, we compared this algorithm to the *Round Robin* algorithm by using the CloudSim simulation tool.

The parameters of algorithm ρ^{local}, ρ^{global} , α, β, q^0 are the experimental

Algorithm 5: *The MORA-ACS Algorithm.*

Input :

- A set of virtual services $\mathbb{VS} = \{i \mid i = 1, ..., N\}$, a set of resource types $\mathbb{D} = \{k \mid k = 1, ..., D\}$, resource needs: r^e_{ik},r^a_{ik},f^e_{ik},f^a_{ik} and additional factor q_{ij}.
- A set of physical machines $\mathbb{PM} = \{j \mid j = 1, ..., M\}$, resource of physical machines: c^e_{jk}, c^a_{jk}.

Output: A set of Pareto, $\mathbb{P}$.

```
Set the parameter values: ρ^local, ρ^global, α, β, q^0, numAnt, numLoop ;
Initialize the empty set of Pareto ℙ;
Execute the initialization solution s^0 using FFD and Calulate τ^0_ij;
while (nL ≤ numLoop) do
    for (nA := 1 → numAnt) do
        j := 1;                                  /* Using one physical machine. */
        while (𝕍𝕊 ≠ ∅) do
            Calculate 𝕍𝕊^temp;
            Calculate η_ij ;
            Calculate p_ij ;
            if (𝕍𝕊^temp ≠ ∅) then
                Generate random number q ∈ [0, 1];
                if (q ≤ q^0) then
                    exploitation;
                    𝕍𝕊 := 𝕍𝕊 − {i}; Load_j := Load_j + (r^a_ik + q_ij × f^a_ik);
                else
                    exploration;
                    𝕍𝕊 := 𝕍𝕊 − {i}; Load_j := Load_j + (r^a_ik + q_ij × f^a_ik);
            else
                j := j + 1;                      /* Using one new physical machine j. */
        Local pheromone updating;
    Calculate objective function E(s) by Formular (3.3) and σ(s) by
      Formular (4.1);
    Update non-dominated solution to Pareto Set P according to Algorithm
      in Section 4.3.1;
    foreach s ∈ ℙ do
        Global pheromone updating;
return Tập Pareto ℙ;
```

parameters. The experience shows that $\rho^{local} = \rho^{global} = 0.45$ and $\alpha = \beta = 0.5$, $q^0 = 0.9$ indicate the best results.

Each algorithm uses three measures: standard deviation (SD), energy consumption in a time period of $\Delta t = 24$ hours and the run-time of algorithm.

4.3.3.2.Experimental Results and Evaluations

The experimental results are shown in Table 4.1.

Bảng 4.1: Consumption Energy and Standard Deviation(SD) of *MORA-ACS* and *Round Robin*.

Num. of VM	Algorithms	Times(s)	Energy(kWh)	Gain Energy(%)	SD
100	*Round Robin*	0,031	201,284		53,24
	MORA-ACS	0,037	190,726	5,536	40,75
200	*Round Robin*	0,078	396,706		49,95
	MORA-ACS	0,084	390,292	1,643	43,04
300	*Round Robin*	0,116	597,989		52,35
	MORA-ACS	0,121	574,440	4,099	41,33
400	*Round Robin*	0,144	793,411		54,63
	MORA-ACS	0,150	770,908	2,919	42,31
500	*Round Robin*	0,200	994,694		53,03
	MORA-ACS	0,216	960,156	3,597	40,52

The experimental results, presented in Table 4.1, show that the energy consumption and the standard deviation of the *MORA-ACS* algorithm are better than the *Round Robin* algorithm.

This is explained by the fact that the *Round Robin* algorithm tends to use a large number of physical machines.

Besides, the *MORA-ACS* algorithm uses a more efficient global search solution. Therefore, it uses fewer resources and ensures better load balancing.

However, the run-time of the *MORA-ACS* algorithm is higher than that of the *Round Robin* algorithm due to the loop parameter, *numLoop,* and the number of ants, *numAnt*.

4.4.Summary of Chapter 4

Chapter 4 presents the Multi-Objective Resource Allocation problem for virtual services on Heterogeneous Shared Hosting Platform which aims at minimizing the energy consumption of the system and balancing the load of PMs. The experimental results indicate that the *MORA-ACS* algorithm yields a better performance in comparison with the *Round Robin* algorithm in terms of energy consumption and load balancing. This result was published in article 5.

CONCLUSIONS AND FUTURE RESEARCH

Conclusions

The major contributions of this dissertation are briefly resumed as follows:

(1) Based on the resource allocation model for virtual services, the resource needs model and resource model of Mark Stillwell, in other words, the mathematical model of the resource allocation problem for virtual services on a homogeneous shared hosting platform, with the objective of minimizing the used PMs and proposing the *MDRAVS-MMAS* algorithm which is relied on the *Max-Min Ant System* algorithm to estimate and compare the objective function value and the actual execution time of algorithm with that of the *First Fit* and *Best Fit* algorithm. The experimental results show that ***the used PMs of proposed MDRAVS-MMAS algorithm a less than FFD algorithm. The complexity of algorithm is implemented in polynomial time***.

(2) This study extended the resource allocation problem on a homogeneous shared hosting platform by applying the resource model and resource needs model of Mark Stillwell, improved energy consumption model of Eugen Feller and built a mathematical model on a heterogeneous shared hosting platform with the objective of minimizing the energy consumption of the system. The *ECRAVS-PSO* algorithm developed based on the *Particle Swarm Optimization* algorithm and the *ECRAVS-SA* algorithm derived from the *Simulated Annealing* algorithm were proposed to estimate the energy consumption and compare the objective function value and the actual execution time with that of the *FFD* algorithm. The experimental results show that ***the energy consumption of proposed MDRAVS-MMAS and ECRAVS-SA algorithms a better performance than FFD algorithm. The complexity of algorithms is implemented in polynomial time.***

(3) Based on the system model, resource model, and the resource needs model of the resource allocation problem on a heterogeneous shared hosting platform, this study built the mathematical model to solve the multi-objective resource allocation problem. This problem focused on both balancing the load of PMs and minimizing the energy consumption. Developing the Ant Colony System algorithm, this study proposed the *MORA-ACS* algorithm to estimate and compare the objective function value and the actual execution time of algorithm with that of the *Round Robin* algorithm. The experimental results show that ***in the CloudSim environment, the MORA-ACS algorithm could balance the load as well as reduce the energy consumption better than the Round Robin algorithm. The complexity of algorithm is implemented in polynomial time***.

Recommendation for Future Research

Future work in this area should pay attention to the dynamic resource allocation problem in cloud computing system. Exploring the algorithms having the smaller execution time is an interested issue and suggested to have a further study.