Capacity Management Approaches for Compute Clouds

Mina Sedaghat
Abstract

Cloud computing provides the illusion of a seamless, infinite resource pool with flexible on-demand accessibility. However, behind this illusion there are thousands of servers and peta-bytes of storage, running tens of thousands of applications accessed by millions of users. The management of such systems is non-trivial because they face elastic demand, have heterogeneous resources, must fulfill diverse management objectives, and are vast in scale.

Autonomic computing techniques can be used to tackle the complex problem of resource management in cloud data centers by introducing self-managing elements known as autonomic managers. Each autonomic manager should be capable of managing itself while simultaneously contributing to the fulfillment of high level system-wide objectives. A wide range of approaches and mechanisms can be used to define and design these autonomic managers as well as to organize them and coordinate their actions in order to achieve specific goals.

This thesis investigates autonomic approaches for cloud resource management that aim to optimize the cloud infrastructure layer with respect to various high level objectives. The resource management problem is formulated as a problem of optimization with respect to one or more management objectives such as cost, profitability, or data center utilization, as well as performance concerns such as response time, quality of service, and rejection rates. The aim of the reported investigations is to address the problems of cost-efficient elastic resource provisioning, unified management of cloud resources, and scalability in cloud resource management. This is achieved by introducing three new concepts in capacity management: the Repacking, Holistic, and Peer to Peer approaches.
Preface

This thesis contains an introduction to cloud computing, and a brief discussion on capacity management in cloud data centers, and the below listed papers.


Financial support for this work is provided in part by the European Commission’s Seventh Framework Programme ([FP7/2001-2013]) under grant agreement number 257115, OPTIMIS, the Swedish Research Council (VR) under contract number C0590801 for the project Cloud Control and the Swedish Government’s strategic research project eSSENCE.
Acknowledgments

“Erik, you are simply the Best!”, saying more would just devalue my deep gratitude and appreciation to you as my supervisor, my mentor and a great human being.

I am also thankful to my co-supervisor Francisco Hernández, for his helpful suggestions and perceptive comments during my work. His insights and his support are invaluable.

I would like to thank Gonzalo Rodrigo, my office-mate and my good friend, for his patience and fun attitude dealing with my weather complaints and for the great time we share in the office. I am also greatful for being part of an explosive research group, a group that each and everyone is unique, brilliant, sharp and most importantly fun to work with. Wubin Li, Petter Svärd, Daniel Espling, Johan Tordsson, Luis Tomás, Cristian Klein, P-O Östberg, Ewnetu Bayuh, Ahmed Ali-Eldin, Peter Gardfjäll, Lars Larsson, Amardeep Mehta, Selome Tesfatsion, Tomas Forsman, Olumuyiwa Ibidunmoye and Jakub Krzywda. I am lucky to have you all as my friends and sharing my time with you every single day.

I would like to thank my amazing Dad and my affectionate Mom and my lovely brother Mahbod. It is hard to support a stubborn girl, but you have done it perfectly. Love you all.

Last but not least, I would like to thank Ehsan, my closest friend and my dearest husband for being extremely selfless and compassionate. You gave up on your success just to support mine. You may not realize how much it means to me, but thanks for all your love and support and I wish I can give you back a small part of it.

Umeå, December 2013

Mina Sedaghat
# Contents

1 Introduction                                        1
  1.1 Cloud Characteristics                          2
  1.2 Cloud Service Models                           2
  1.3 Deployment Models                              4
  1.4 Cloud stakeholders                             5
  1.5 Cloud computing challenges                     5

2 Capacity management in clouds                     9
  2.1 Resource allocation                             9
  2.2 Elasticity                                      10
  2.3 Admission control                               10
  2.4 Reliability and fault management                11
  2.5 Monitoring                                      11

3 Contributions                                      13
  3.1 Paper I                                         14
  3.2 Paper II                                        14
  3.3 Paper III                                       15

4 Future work                                        17
Chapter 1

Introduction

Cloud computing has attracted a lot of attention in recent years, but its common understanding has evolved over time. Its core concept can be traced back to 1961, when John McCarthy suggested that computers could function as a public utility in the same way as telephones or electricity. More recently, in 2001 IBM took the first steps towards the realization of a cloud infrastructure while working on a project called Océano [1], an e-business computing utility infrastructure that can allocate resources to customers dynamically as their requirements change.

The contemporary concept of clouds is strongly rooted in the industry when in 2006 Amazon’s chief executive, Jeff Bezos, announced the two cloud computing products of the company, EC2 and S3. Amazon developed these products because it had become apparent that maintaining a reliable, scalable infrastructure in a traditional multi-datacenter model for an extended period of time would be very costly and would require a significant investment of intellectual capital. They therefore decided to offer infrastructural services to other businesses, promising to reduce their clients’ capital and operational costs while also utilizing Amazon’s own immense resources in a more efficient and profitable way.

Pre-existing paradigms such as grid computing and utility computing created an infrastructural pathway that enabled the development of cloud computing. Each of these paradigms contributed one or more of the concepts needed to establish this pathway. Grids evolved to overcome the need for very high computational capacities and were primarily used by scientific organizations. Meanwhile, utility computing introduced the concept of packaging computing resources as utilities that clients would pay for according to their usage.

However, both paradigms lacked some important features that limited their applicability and hindered their adoption, such as flexibility, on-demand availability, or the ability for short term acquisition of seemingly infinite computing resources [2].

Cloud computing extends the concepts that underpin these two paradigms
by offering a large integrated computing capacity to the public in the form of a utility that can be accessed in an on-demand and flexible manner.

The most commonly used definition of the cloud is provided by National Institute of Standards and Technology (NIST) [3]. It states that cloud computing is “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.”

1.1 Cloud Characteristics

Cloud computing has a number of attributes. However, according to the NIST [3], its five defining characteristics are:

- **On-demand service provisioning**: A consumer is able to provision computing resources as needed. This process should be done automatically and without requiring human interaction with the provider.

- **Network-centric**: The provided services are accessible over a network via different devices such as PCs, mobile devices, and laptops, using a standard mechanism (e.g. HTTP).

- **Resource abstraction and pooling**: The provider creates a resource pool that can be assigned to a variety of applications or to multiple consumers, with different levels of control. The exact location of the resource is completely invisible to the consumer. However, at a high level of abstraction, he/she may be able to impose some constraints on the resource’s location (in terms of e.g. the data center, state, or country in which it is situated).

- **Rapid elasticity**: Computing resources are acquired and released quickly, based on user demand and with minimal management effort. This gives the consumer the illusion that there are infinite available resources that can be requested in any quantity at any time.

- **Measured service**: Resource usage is monitored, measured and reported on a per service basis. This information can be used for billing purposes or to manage and optimize the offered services.

1.2 Cloud Service Models

Cloud computing has established the notion of a ”Service” as a basic unit of abstraction [4]. Within this paradigm, any type of IT resource, from CPU and storage capacity to software licenses and information, can be treated as a service and offered to a customer. However, each service type has specific requirements
and characteristics. Depending on the nature of the services offered to the customers, cloud systems can be considered to operate according to one of the three common service models [3]:

- **Infrastructure as a Service (IaaS):** In this model, the provider offers a pool of hardware resources such as CPU, memory, storage, I/O, network bandwidth or IP addresses. These resources are often packaged into virtual machines (VM). Each virtual machine functions as a virtual private server and can be used by the users to manage their operating system or software in a virtualized compute environment.

  The target consumers of IaaS are software developers and system administrators who require resources to launch their applications but do not wish to maintain their own IT infrastructure and instead rent existing infrastructure from a provider via the Internet.

  IaaS providers rely heavily on virtualization to enable the flexible multiplexing of different applications onto physical servers in order to efficiently utilize their resources. Virtualization helps the provider to hide the complexities of the underlying system from the users, instead presenting them with an abstract emulated computing platform.

  Amazon (via the EC2 system [5]), GoGrid [6], and Rackspace [7] are some of the more well-known vendors offering infrastructure as a service.

- **Platform as a Service (PaaS):** PaaS systems provide an operating environment that includes programming languages, libraries, services and tools that enable developers to build software applications over the Internet and deploy them on the cloud infrastructure. PaaS services are made available as cloud services and accessed by users via web browsers. In this service model, consumers have no control over the underlying infrastructure but they do control their deployed applications and may be able to configure their application-hosting environment.

  Some of the most well known PaaS systems are Windows Azure[8], Google AppEngine [9], Amazon’s Elastic Beanstalk [10], and VMWare Cloud Foundry [11].

- **Software as a Service (SaaS):** SaaS clouds allow users to subscribe to applications on-demand rather than buying them. The applications are hosted on a cloud infrastructure and can be accessed via different client devices through a web browser or an application interface. The consumer does not have any control over either the application or the underlying resources.

  Facebook, Flickr, Office software, and Google services such as Google Docs and Gmail are examples of SaaS.
1.3 Deployment Models

Cloud services can be deployed in several ways. Each deployment model defines how cloud resources, in terms of the infrastructure, application or data, are structured, as well as where the resources are located and who is able to manage them. Every model [3] has its own benefits and drawbacks:

- **Public clouds**: The public cloud provides resources for the general public use and is owned and operated by a cloud service provider. It shares its resources between multiple multi-tenant customers and typically charges these customers on a pay-per-use basis. This model provides transparency, flexibility, resilience and consistent availability to its users. It also helps to reduce the provider’s capital expenditure and operational costs. Public clouds are particularly suitable for businesses with limited security concerns and start-ups who cannot reliably predict their future usage patterns or the applications they will require during testing and development. Amazon’s EC2 [5] is one of the more notable public cloud systems.

- **Private clouds**: Private cloud systems provide many of the same benefits as public clouds but are only accessible by their member organizations and may use different billing models than private clouds. They enable the exploitation of cloud efficiencies while retaining customized control over resources and the ability to enforce security policies regarding data belonging to the member organizations.

- **Community clouds**: A community cloud is a joint infrastructure owned by multiple organizations that have a common set of requirements or share specific needs or concerns. As in private clouds, the joint infrastructure is used exclusively by the member organizations. This enables them to combine and share their computing resources, data, and capabilities while reducing their costs.

- **Hybrid clouds**: Hybrid clouds are formed by combining two or more clouds that use different deployment models. Each cloud is distinct and independent, providing a unique set of services. However, the clouds are linked in order to meet specific requirements. For example, a hybrid cloud might consist of a public cloud and a private cloud, allowing the private cloud to burst to the public cloud in order to meet spikes in demand while still being capable of meeting stringent security requirements. In such cases, the availability of the public cloud would enable the system as a whole to provide acceptable performance to all users without exceeding budgetary constraints.
1.4 Cloud stakeholders

The main actors in a cloud environment are:

- **End-users**: End users are the ultimate consumers of a cloud service.

- **Service Providers (SPs)**: Service providers typically provide application-level services to end users. They are the main consumers served by infrastructure providers. SPs manage applications and, at the lowest level, their own VMs in order to achieve SP-level objectives. Their objectives typically include decreasing their resource usage costs while maintaining an acceptable Quality of Service (QoS) for their applications (for instance, ensuring that the applications have acceptable response times).

Service providers have no control over low level hardware resources. However, they can use low level metrics to optimize their applications and to provide application-level performance data to end users or to perform automatic (or manual) scaling of their resources. They may rent or manage their resources directly from an infrastructure provider or via a third party such as a cloud broker.

- **Infrastructure Providers (IPs)**: Infrastructure providers provide virtual hardware resources to the service providers. They manage hardware resources such as virtualized instances, CPU, storage, I/O and network bandwidth. Their main concerns typically involve ensuring efficient resource utilization, energy efficiency, the fair allocation of resources, and maximizing profits while maintaining resource availability to customers.

- **Cloud brokers**: Cloud brokers are third party entities that negotiate, monitor, and manage services that are deployed to a cloud on behalf of a cloud consumer, i.e. an SP.

1.5 Cloud computing challenges

Cloud computing offers agile platforms for service provision that provide high availability and flexibility while requiring less maintenance and having lower costs than the available alternatives. Providers must address several challenges to offer these capabilities:

- **Resource management complexities**: The general problem of resource management in cloud data centers is often computationally complex or even NP-complete [12]. This means that even if enough resources are available to meet the current demand, the problem of correctly matching a set of demands with a set of resources may be too complex to solve in a useful time [12].

In general, to solve this problem it is necessary to optimize the allocation of resources in an on-demand and cost-effective manner while maintaining
acceptable application performance and/or maximizing IP profitability. This entails making decisions on matters such as whether to accept or reject a given application, how to change the allocated capacity of a running application in response to a change in demand, and which resources to allocate or de-allocate. These decisions all relate to complex but fundamental problems that generally lack straightforward solutions.

In addition to the complexity of the problem, a cloud data center consists of thousands of servers and peta-bytes of storage, running tens of thousands of applications that are accessed by millions of users. When working on such a vast scale, it is very challenging to manage the different layers of the abstracted services (which range from applications to VMs) and the physical machines themselves.

In practice, the management of such systems is only possible through the development of complex autonomic management solutions. These often consist of a collection of autonomic managers, which must be capable of managing themselves while making managerial decisions in accordance with high level objectives specified by a human manager [13]. The need to coordinate the actions of different autonomic managers such that they work together towards common goals further increases the complexity of the problem.

In summary, the management of cloud resources is challenging due to the scale and complexity of the problem together with the need to achieve high level objectives.

- **QoS and SLA negotiations:** Most current IaaS providers offer simple SLAs based on resource availability but lack advanced SLA mechanisms based on service-oriented QoS metrics such as performance or response times [14]. The ability to guarantee a service-oriented QoS is highly dependent on understanding the characteristics of the application in question including its architecture and workloads as well as the interdependencies of its components. At the same time, to provide a service to a wide range of customers in a multi-tenant system, one must abstract away all of these application-level complexities insofar as possible. This trade-off between the need for abstraction and the need for customized service delivery hinders the provision of complex SLAs and presents a major challenge for both cloud providers and consumers.

- **Security, data privacy and trust:** Cloud computing presents an added level of risk relative to non-cloud solutions because essential services are often outsourced to a third party. This makes it harder to maintain data security and privacy, support data and service availability, and demonstrate compliance with the relevant regulations and security policies [15]. Cloud environments give attackers opportunities to breach security and trust due to their potential for incomplete data deletion [16] as well as their tendency to locate data in different jurisdictions [17], to allocate and
deallocate VMs [18] in an unrestricted fashion, to perform uncontrolled migration [19], their reliance on public VM image repositories [20], the sharing of resources by different virtual machines [21], and their often lacking authorization protocols.

- **Interoperability**: Cloud interoperability refers to the ability of cloud services to work together with other cloud services and different providers, and with applications or platforms that are not cloud-dependent [22]. Specific business incentives and the diversity of applications and configurations used by each cloud provider can lead to vendor lock-in and hamper communication between different cloud providers. However, complex business applications may need to use resources from different cloud providers and data centers seamlessly. It therefore seems that unified standard cloud APIs and cloud interfaces that support interoperability will be required.

- **Energy management**: It has been estimated that the cost of powering and cooling servers accounts for 53% of the total operational expenditures of data centers [23]. Therefore, a key challenge in data center management is to minimize power consumption. Considerable effort has been devoted to the development of energy-conserving resource management techniques. Progress has been made by using more sophisticated hardware and by increasing the degree of server consolidation via virtualization in order to enable the hibernation of idle servers, among other things. However, there is considerable scope for further reductions in power consumption and improvements in energy efficiency.

The remainder of this thesis focuses on the first of these challenges, i.e. "resource management" in cloud data centers. The following sections discuss the complexities mentioned above in greater detail and present solutions to the underlying problems.
Chapter 2

Capacity management in clouds

As mentioned in Section 1.5, the management of resources in cloud data centers has become very complicated due to the scale of the managed elements and the nature of the problem. In practice, the general problem of cloud resource management has multiple components and can be regarded as a group of sub-problems, each of which is handled by its own set of management processes. All of these processes are essential for the adequate provisioning of resources in a cloud environment and the maintenance of a suitable balance between capacity utilization, cost, and service quality. This chapter deals with the problem of capacity management in cloud data centers. Note that in the remainder of the thesis, the terms resource management and capacity management are used interchangeably.

2.1 Resource allocation

A key problem in the management of cloud data center resources is to identify resources that can meet the requirements of each service and allocate those resources to the services in a way that is compatible with the cloud provider’s objectives. This is achieved by an allocation process whose quality is often measured using a utility function that represents the cloud provider’s objectives [24]. The optimal resource allocation scheme is that which maximizes this utility function. The utility function may be defined to emphasize profit, resource utilization, energy efficiency, or the fairness of the allocation scheme.

Allocation processes should be dynamic because applications’ workloads vary dynamically over time and their lifetimes are generally unknown when resources are initially allocated. Due to its dynamic nature, the allocation process may benefit from re-consolidation and over-consolidation of resources. Re-consolidation is performed to maintain the optimality of the allocation
scheme by re-distributing currently allocated resources, while over-consolidation is performed to increase resource utilization in cases where the allocated resources are not being used to their full potential.

The input data required to solve resource allocation problems include information on the available cloud resources in terms of VMs, memory and CPU capacity; the requirements of the applications, which are usually defined in terms of either on-demand capacity (expressed as workloads) or static capacity (predefined VMs that are requested once); and the provider’s requirements, which are referred to as objectives.

2.2 Elasticity

The primary feature that distinguishes clouds from other distributed computing environments is their capacity for on-demand service provisioning, which is known as elasticity. Cloud users should be able to provision and de-provision their resources as their demand changes, such that the available resources match the current demand as closely as possible at each point in time [25].

Elasticity algorithms rely on monitoring data to estimate demand based on the current load and can be divided in two types: those that change the number of VM instances of a certain type and thereby provide horizontal elasticity, and those that change the size of the VMs and thereby provide vertical elasticity.

Numerous authors have studied elasticity and a wide range of approaches have been used to modify provisioning in response to changes in applications’ demands, including static threshold setting [26, 27], control theory [28, 29], queuing theory [30, 31], and time series analysis [32, 33].

2.3 Admission control

Admission control is the process of deciding which services to accept in order to increase data center utilization and profitability while avoiding SLA penalties and adverse effects on the performance of other running services. It involves evaluating the impact that accepting the new service will have on short- and long-term system behavior, i.e. balancing the increase in utilization achieved by accepting the new service against the risk of SLA violations [34].

Admission decisions would be simple if a fixed amount of resources could be guaranteed to meet the service’s demand. However, the decision is made complex by uncertainties relating to the burstiness of the applications in question and the limited availability of information on future workloads, the exact impact of potential resource shortages on other applications, and the potential side effects of co-locating particular VMs [35].

While the main purpose of admission control processes is to determine whether to accept or reject services, they can also affect data center utilization by over-booking resources. Over-booking is based on the assumption that not all of the resources requested by services will be completely utilized during those
services’ lifetimes. In such scenarios, cloud providers can benefit by reserving more resources than the total available capacity of the cloud to maximize utilization. However, the elastic behaviors of the running applications and the potential fluctuations in their resource demands must be considered when making decisions on over-booking, along with the associated risks.

2.4 Reliability and fault management

Due to their great complexity, even very carefully engineered data centers experience a large number of failures, especially when they are distributed over several locations [36]. A system’s reliability may be affected by failures of physical machines, hardware, VMs and applications, failures due to power outages, failures of monitoring systems, and possibly even failures during live VM migrations. It is therefore essential for cloud providers to consider fault tolerance, reliability and availability in order to ensure correct and continuous system operation even when such failures occur.

Infrastructure providers should transparently provide reliability to the user and effectively supply the desired reliability by allocating virtual back-up nodes that stand ready to take over the running of the affected services in the event of node failure [37]. Reliability is achieved if the running applications continue to perform properly regardless of any underlying breakdowns.

2.5 Monitoring

According to the NIST [3], resource usage in clouds should be monitored, measured and reported to the cloud consumer. In addition, cloud providers require monitoring data for the maintenance and optimization of their own internal functionalities, such as capacity and resource planning, SLA management, billing, trouble-shooting, performance management, and security management [38].

Cloud monitoring can be performed on different levels, each of which requires its own metrics and provides specific information. For example, hardware level monitoring may provide information on factors such as CPU and memory utilization while application level monitoring may provide information on an application’s response times or throughput.

The on-demand nature of clouds requires that different managerial processes receive different kinds of monitoring data at different rates. For instance, an elasticity algorithm may require monitoring data on a per minute or even sub-minute basis, to detect load spikes associated with specific services and respond appropriately. Conversely, admission control processes may request monitoring data much less frequently. The monitoring of cloud resources is therefore complicated by the diverse frequencies at which data must be collected, stored, and processed, and the variation in the granularity and volume of the monitored data.
Chapter 3

Contributions

The complexity, heterogeneity, and scale of the physical and IT infrastructures involved in maintaining cloud environments and the need to consider several different management objectives simultaneously makes autonomic management an absolute necessity for cloud resource management.

The process of designing an autonomic system starts with the definition of high level objectives in terms of utility functions. A combination of modeling, optimization, and (sometimes) learning techniques are then applied to this explicit mathematical representation of the objectives to identify an optimal solution for the control variables [39]. Such complex systems are often segmented into multiple autonomic managers, each of which deals exclusively with a specific sub-problem, a cluster of servers, a specific concern or an individual objective. This allows a system that would be extremely hard to understand and manage as a whole to be broken down into smaller and more comprehensible components. A range of different approaches have been proposed for the definition and design of autonomic managers and the coordination of their actions to achieve system-wide objectives.

This thesis is concerned with autonomic approaches to the problem of coordinating autonomic manager behavior and its optimization with respect to a number of high level objectives. Paper I [40] describes an investigation into the coordination of two vertical and horizontal autoscalers to achieve cost-effective resource allocation for a specific application. Paper II [41] takes a broader approach, describing a data center-level study on the coordination of four autonomic managers (admission control, elasticity control, placement control, and a fault tolerance engine) with respect to a high level objective. Paper III [42] describes another data center-level study in which a P2P approach is used to address the same problem of resource management. In this approach, the optimization of the high level objective is formulated as a problem of coordinating the actions of numerous autonomic agents that manage clusters of servers rather than the four autonomic managers introduced in Paper II.
3.1 Paper I

Paper I [40] discusses a re-packing approach for coordinating the trade-offs between horizontal and vertical elasticity decisions that must be made to manage the capacity acquired by an elastic application in a cost-effective and on-demand fashion.

Adapting application’s resource set to changes in load can be quick and cheap in terms of reconfiguration costs following horizontal elasticity decisions; because they only add the extra capacity on the currently deployed VMs, however the resulted resource set could be far from optimal for the aggregated capacity over time. On the other hand, vertical elasticity decisions require frequent, costly and time-consuming reconfiguration of the deployed resource set, such as VMs On/Offs; but they maintain the optimality of the resource set. Therefore there is an inevitable trade-off between costs and benefits of the two methods.

This paper investigates how combining the benefits of vertical and horizontal elasticity can improve the cost efficiency of the resource set when scaling is performed in terms of both the number and the size of VMs. A cost-benefit analysis is presented that takes the current configuration as well as the cost and durability of a reconfiguration into account to determine the appropriate trade-off between horizontal and vertical scaling. This analysis is then used to inform and make a re-packing decision.

We evaluated our re-packing approach in combination with different auto-scaling strategies. The effects of varying different parameters on re-packing decisions were investigated, including workloads, the cost and durability of reconfigurations, and the type of application being run. It was found that by performing a cost-benefit analysis, we could decide when and how to replace the non-optimal set with a new optimal set in a way that reduced the total cost of resource utilization by up to 60% over the application’s life time.

3.2 Paper II

Paper II [41] addresses the challenge of managing cloud resources in a holistic way to satisfy the business level objectives of an IP. The paper adopts a top-down approach to the development of a unified cloud resource management system, which is outlined in Figure 1. In this approach, the management process is divided across a collection of low level controllers with very distinct responsibilities. Each autonomous controller performs a specific managerial task such as admission, elasticity control, VM Placement, or monitoring and fault management.

The key problem to address is that the individual controllers are quite complex and despite their distinctly separate responsibilities, they are far from independent of one-another. Consequently, a decision made by one can have profound effects on the responses of others, thereby affecting the optimality of the resulting solution with respect to the high level objective. For example,
increasing the server consolidation level (number of VMs per physical host) or making the elasticity control more restrictive may allow the admission controller to be less restrictive in accepting services. Therefore, optimizing the overall system behavior to achieve a Business Level Objective (BLO) requires some degree of coordination between these actions.

Paper II proposes a business-oriented governance model for coordinating such conflicting goals and tuning the system to strive toward a high level objective or BLO. The paper provides a unified view of several managerial challenges, mostly from the perspective of the infrastructure provider, followed by a review of the available solutions. The paper also presents a novel formulation of the problem of controller coordination with a list of the major challenges that must be addressed.

3.3 Paper III

Paper III [42] proposes a Peer to Peer (P2P) resource management framework for cloud data centers. The main objective of the study was to develop a resource management solution that is scalable with respect to both the number of physical servers and incoming Virtual Machine (VM) requests while remaining computationally practical. The framework consists of an agent community that interacts in a goal-oriented P2P fashion and a gossip protocol for information dissemination, discovery and optimization.

Unlike Paper II, Paper III adopts a bottom-up approach in which the system is decomposed into many interacting elements, each of which performs both functional and managerial tasks. In this case, the purpose of the decomposition is not to separate the concerns of each element but to ensure that the scale of
the task assigned to each element is sufficiently modest that each autonomic element can simply solve the problem within its own local scope.

In this approach, the high level objective is achieved emergently via the cooperation of these autonomic elements. A conceptual overview of the P2P approach is presented in Figure 2.

![Figure 2: Resource Management (P2P Approach)](image)

As part of the work, we focus on the problem of resource allocation. We propose a gossip-based resource allocation protocol that aims to maximize data center utilization and profit by ensuring that active nodes are highly utilized and decreasing power consumption by putting the remaining nodes into energy saving mode.

The new protocol’s performance was evaluated with respect to data center utilization, node utilization, number of hops, rejection ratio and profit. We highlight the impacts of request propagation and allocation distribution on the protocol’s performance. We also studied the scalability of our approach by evaluating its performance for different numbers of servers. Promising results were achieved, indicating that the approach can be scaled up to systems of at least 5000 nodes with 9000 placement requests arriving during the simulation time if using policies that provide efficient request propagation.

Finally, we present a re-consolidation process for optimizing the allocation of currently running VMs. The re-consolidation process was designed to redistribute the allocation of data center resources in order to enable the efficient utilization of active nodes. Re-consolidation was shown to increase node utilization by up to 10%. This reduces the data center’s operational costs by minimizing its power consumption and thereby increases its profitability.
Chapter 4

Future work

Our preliminary results on P2P resource management are promising in terms of performance, suggesting that such approaches are practical and offer significant advantages when tackling large scale resource management problems. The work presented in this thesis will be extended by building on the results reported in Paper III [42]. Specifically, the P2P approach will be investigated further and applied in new contexts. Four key goals will be addressed in these studies:

1. **Investigating more efficient techniques for optimization**
   We are currently formulating the resource allocation problem as an optimization problem and using heuristics for discovery and optimization. Heuristics are capable of generating moderately optimal solutions within a short execution time at a low computational cost. However, a more in-depth evaluation of alternative optimization techniques is required to increase the quality of the optimized solution.

2. **Further evaluation of the scalability of the P2P approach**
   Paper [42] describes an investigation into the scalability of the proposed P2P approach with respect to the numbers of servers and incoming VM requests. Our results indicate that the framework scales well up to at least 5000 servers and 9000 VM requests. However, further tests will be required to evaluate the applicability of the approach on larger scales. We are also interested in studying the strong scalability of the system in cases where the number of VM requests increases while the number of servers remains constant.

3. **Extending the P2P approach to support different functionalities**
   As mentioned previously, the P2P framework is designed to support different cloud functionalities including admission control, failure detection and management, and VM and application placement. We have partially addressed the resource allocation and placement problem on the VM level. However, we want to adapt the framework to manage other cloud
functionalities. It would also be desirable to investigate ways of unifying these different management functionalities within the new framework to achieve a high level business objective.

4. **Extending the P2P approach to support different BLOs**

   We currently formulate the allocation problem in order to fulfill two separate business objectives: data center utilization and profit. It would be useful to develop methods that could accommodate more complex business objectives such as fairness, energy efficiency or the minimization of SLA violations.

   The framework could also be extended to support a more complex set of constraints such as affinity and anti-affinity constraints when allocating VMs, or consolidation rate and utilization limits that could be used to govern overbooking in a data center.
Bibliography


