A Study on Social Media Applications into Geo-Distributed Multi Clouds

E. Chinnarasu¹ H. Lookman Sithic²
¹Research scholar ²Associate Professor
¹,²Department of Computer Science and Engineering
¹,²Muthayammal College of Arts & Science, Tamilnadu India

Abstract— Social media applications are dominating the internet. A geo-distributed cloud is model for supporting large-scale social media applications with dynamic contents and demands. The defy lies in storing and relocation of the contents dynamically from different cloud and distributing them. This paper study the Cloud Rank Algorithm (CRA) for locally, optimal scaling media applications in a geo-distributed cloud. Group of geo-distributed cloud services is a trend in cloud computing which, by spanning multiple data centers at different geographical locations, can provide a cloud platform with much larger capacities. Such a geo-distributed cloud is ideal for supporting large-scale social media streaming applications with dynamic contents and demands, owing to its abundant on-demand storage/bandwidth capacities and geographical proximity to different groups of users. By exploiting social influences among users, this paper proposes efficient proactive algorithms for dynamic, optimal scaling of a social media application in a geo-distributed cloud. The QoS of the OSN service is better if more users have their data hosted on clouds of a higher preference.

Keywords: Cloud Rank Algorithm, Geo-Distributed Multi Clouds

I. Introduction

A. Cloud:

A cloud refers to a distinct IT environment that is designed for the purpose of remotely provisioning scalable and measured IT resources. The term originated as a metaphor for the Internet which is, in essence, a network of networks providing remote access to a set of decentralized IT resources. Prior to cloud computing [2] becoming its own formalized IT industry segment, the symbol of a cloud was commonly used to represent the Internet in a variety of specifications and mainstream documentation of Web-based architectures. This same symbol is now used to specifically represent the boundary of a cloud environment, as shown in Figure 1.

Fig 1: The symbol used to denote the boundary of a cloud environment.

It is important to distinguish the term “cloud” and the cloud symbol from the Internet. As a specific environment used to remotely provision IT resources, a cloud has a finite boundary. There are many individual clouds that are accessible via the Internet. Whereas the Internet provides open access to many Web-based IT resources, a cloud is typically privately owned and offers access to IT resources that is metered.

Much of the Internet is dedicated to the access of content-based IT resources published via the World Wide Web. IT resources provided by cloud environments, on the other hand, are dedicated to supplying back-end processing capabilities and user-based access to these capabilities. Another key distinction is that it is not necessary for clouds to be Web-based even if they are commonly based on Internet protocols and technologies.

B. A Different Way To Deliver Computer Resources:

A revolution is defined as a change in the way people think and behave that is both dramatic in nature and broad in scope. By that definition, cloud computing is indeed a revolution. Cloud computing is creating a fundamental change in computer architecture, software and tools development, and of course, in the way we store, distribute and consume information. The intent of this article is to aid you in assimilating the reality of the revolution, so you can use it for your own profit and well being.

C. Layers: Computing as a commodity:

The cloud concept is built on layers, each providing a distinct level of functionality. This stratification of the cloud's components has provided a means for the layers of cloud computing to becoming a commodity just like electricity, telephone service, or natural gas. The virtual machine monitor (VMM) provides the means for simultaneous use of cloud facilities VMM is a program on a host system that lets one computer support multiple, identical execution environments. From the user's point of view, the system is a self-contained computer which is isolated from other users. In reality, every user is being served by the same machine.

II. LITERATURE REVIEW


This paper discusses the concept of Cloud Computing to achieve a complete definition of what a Cloud is, using the main characteristics typically associated with this paradigm in the literature. More than 20 definitions have been studied allowing for the extraction of a consensus definition as well as a minimum definition containing the essential characteristics. This paper pays much attention to the Grid paradigm, as it is often confused with Cloud technologies. We also de-scribe the relationships and distinctions between the Grid and Cloud approaches.
computing is a new computing paradigm in which both hardware and software are provided to users over the Internet as services, in the form of virtualized resources [8].


In this paper we present Kingfisher, a cost-aware system that provides efficient support for elasticity in the cloud by (i) leveraging multiple mechanisms to reduce the time to transition to new configurations, and (ii) optimizing the selection of a virtual server configuration that minimizes the cost. We have implemented a prototype of Kingfisher and have evaluated its efficacy on a laboratory cloud platform. Our experiments [5] with varying application workloads demonstrate that Kingfisher is able to (i) decrease the cost of virtual server resources by as much as 24% compared to the current cost unaware approach, (ii) reduce by an order of magnitude the time to transition to a new configuration through multiple elasticity mechanisms in the cloud, and (iii), illustrate the opportunity for design alternatives which trade-off the cost of server resources with the time required to scale the application.

C. X. Cheng and J. Liu, “Load-Balanced Migration of Social Media to Content Clouds,” in Proc. of ACM NOSSDAV, June 2011:

Social networked applications have been more and more popular, and have brought great challenges to the network engineering, particularly the huge demands of bandwidth and storage for social media. The recently emerged content clouds shed light on this dilemma. Towards the migration to clouds, partitioning the social contents has drawn significant interests from the literature. Yet the existing works focus on preserving the social relationship only, while an important factor, user access pattern, is largely overlooked. In this paper, by examining a large collection of YouTube video data, we first demonstrate that partitioning the network entirely based on social relationship would lead to unbalanced partitions in terms of access. We further analyze the role of social relationship in the social media applications, and conclude that user access pattern should be taken into account and social relationship should be dynamically preserved.

III. ARCHITECTURE DESIGN FOR CLOUD AND ITS METHODOLOGY

A. Cloud architecture:

Cloud computing has recently gained significant popularity as a cost-effective model for delivering large-scale services over the Internet [10]. In a Cloud computing environment, infrastructure providers (namely, cloud providers) build large data centres in geographically distributed locations to achieve reliability while minimizing operational cost [1]. The service providers (SPs), on the other hand, leverage geo-diversity of data centres to serve customers from multiple geographical regions. Today, large companies like Google, Yahoo and Microsoft have already adopted this model in their private clouds, offering a wide range of services to millions of users worldwide. As Cloud computing technologies become mature, an increasing number of companies are expected to adopt this model by moving into clouds.

Fig. 2: Geo-distributed cloud model.

Beyond mere server scalability, some other elements need to be taken into account that affect the overall application scaling potential. For instance, load balancers (LBs) need to support the aggregation of new servers (typically, but not necessarily, in the form of new VMs) in order to distribute load among several servers. Amazon already provides strategies for load balancing your replicated VMs via its Elastic Load Balancing capabilities [9]. LBs and the algorithms that distribute load to different servers are, thus, essential elements in achieving application scalability in the cloud.

Having several servers and the mechanisms to distribute load among them is a definitive step towards scaling a geo-distributed cloud application. However, there is another element of the data centre infrastructure to be considered towards complete application scalability. Network scalability is an often neglected element that should also be considered [7]. By exploiting social influences among users, this paper proposes efficient proactive algorithms for dynamic, optimal scaling of an online social network application in a geo-distributed cloud. Our key contribution is an online content migration and request distribution algorithm with the following features: (1) future demand prediction by novelty characterizing social influences among the users in a simple but effective epidemic model; (2) one shot optimal content migration and request distribution based on efficient optimization algorithms to address the predicted demand, and (3) a D(1) - step look-ahead mechanism to adjust the one-shot optimization results towards the offline optimum.

B. Existing System:

Compared to conventional CDNs, cloud based CDNs have the benefit of cost efficient hosting services without owning infrastructure. Resource provisioning and replica placement in cloud CDNs involve a number of challenging issues, mainly due to the dynamic nature of demand patterns. To deal with this dynamic nature, this work proposes a set of novel algorithms to solve the joint problem of resource provisioning and caching for cloud-based CDNs [6] [11] with an emphasis on handling the dynamic demand patterns. In existing work, a two-step algorithm framework introduced the Differential Provisioning and Caching algorithm. The intuition is that to minimize the total cost, we first maximize the demands supported by the unexpired
resources. Minimizes the total cost of renting new resources such that all remaining demands can be served.

C. Proposed System:

A cloud server [3] [4] has to spend a significant amount of time in initializing its configuration before it is ready to use. The demand rates or the access patterns are rapidly changed, we propose greedy and proactive algorithms for dynamic, as detailed in the following subsections. These proactive assign content placements. Provisioning solutions can be finally determined by content placement decisions. In this work reinforcement learning algorithm is proposed to effectively handle the dynamic [12] varying of user demands, and time varying user demands. Reinforcement learning is mainly used for proactive management of Virtual machine capacity. Markov decision process is used to model the reinforcement learning process.

D. Geo Distributed Cloud:

We consider geo-distributed clouds [9] in online social network applications. As shown in Fig. 1, N cloud servers locate in different geographic regions. Each region \{1,..,N\} has one server. The server in the ith region is denoted by i S. The service capacity is evaluated by the number of virtual machines, a server has. Thus, the service capacity is limited. We consider time is slotted. At a different time slot, servers have different available service capability due to the dynamic allocation.

![Fig. 3: Geo-Distributed architecture](image)

We assume the computing (storage) servers inside a cloud site have similar hardware configurations, and charge the same prices for usage. Hardware configurations and usage charges are likely to be different across different cloud sites. We take into account the following three types of charges to a cloud consumer: storage cost to keep data on the storage servers, rental fee of VMs to run the application, and charges for incoming/outgoing traffic to/from each cloud site. The former two are charged by usage time on a per unit time rate.

IV. PERFORMANCE EVALUATION

We evaluate the performance of our online algorithm, by building a prototype system on Amazon Elastic Compute Cloud (EC2) under realistic settings.

A. Prediction Accuracy:

In our epidemic model, we set the values of \( \eta_c \) and \( \gamma_c \) for each video c by matching the resulting evolution of the video popularity with that captured by the traces. We found our model matches the traces best when \( \eta_c \) is set to a value around 0.5 and \( \gamma_c \) is chosen in the range of \([0.9, 0.99999]\), for each video c. When fitting an ARIMA model, we collected 96-hours’ user requests in a single dry run. The original series of the number of requests becomes stationary after being differenced twice, and we therefore chose an ARIMA(p,2,q) model; and after carefully checking the partial autocorrelations, an autoregressive model of p = 3 and q = 0 is applied. In Fig. 3, the solid blue curve plots the actual viewing request number in a time span of 48 hours, following the synthesized traces we applied. The dotted red curve corresponds to the ARIMA prediction results, using the ARIMA(3,2,0) model. The black square dots represent the prediction results using our epidemic model for 5 consecutive time slots, made at the time slots marked by ‘+’: e.g., the first five square dots are prediction results done at \( t = 1 \) for the next 5 time slots, the next batch of five square dots are prediction results done at \( t = 14 \) for the next 5 time slots, and so on. For better readability of the figure, we only show the prediction results made at selected time slots of \( t = 1, 14, 27 \) and 40, respectively. We can observe that predicted numbers using our epidemic model follow the actual numbers quite well, especially within a 4-hour look-ahead window (i.e., the first four square dots in each batch are well aligned with the blue curve). However, the ARIMA model fails to capture the social influence among users and performs poorly.

![Fig. 4: performance on sample video](image)

B. Scalability:

The geo-distributed cloud infrastructure for better scalability, lower management overhead, and proximity to users. The private cloud may or may not be part of the federated cloud. As a cloud consumer, the application provider deploys its Web service on the VMs on the computing servers and video files in the storage servers.

![Fig. 5: Scalability analysis](image)
sometimes already sufficient to achieve a large part of optimization. It demonstrates the scalability of our approach in fig. 8. We use METIS to partition our original data set into several partitions, and then apply our approach to each partition independently while neglecting the inter-partition interactions. Doing so saves up to 85% (in the 5-partition case) of the total execution time, and only degrades the total cost of the optimal data placement by less than 8%, compared with running our approach directly on the original dataset. This success roots in the community structure of OSN social relations and interactions; thus even neglecting 45% social relations and the associated 22% interactions of the original dataset only has a slight influence on the optimization.

V. Conclusion

This study describes the proactive, online algorithm to scale social media streaming applications for operating in geo-distributed clouds. Exploiting the fundamental social influences among the users, we build a simple, successful epidemic model to forecast future viewing demands for proactive service deployment. Aiming at operational cost minimization with service delay guarantees, resourceful methods are proposed to solve the one-shot optimization, and a novel (t)-step look ahead mechanism is designed with guarantees to adjust the one-shot optimum to the offline optimum, which is based on solid theoretical analysis. Our large-scale evaluations on an emulated distributed cloud over the Amazon EC2 platform under realistic settings confirm the excellent performance of our online algorithm in pursuing the ultimate optimal replication and request dispatching solutions.

References