RESOURCE ALLOCATION AND TASK SCHEDULING OPTIMIZATION IN CLOUD-BASED CONTENT DELIVERY NETWORKS WITH EDGE COMPUTING

Approved by:

RESOURCE ALLOCATION AND TASK SCHEDULING OPTIMIZATION IN CLOUD-BASED CONTENT DELIVERY NETWORKS WITH EDGE COMPUTING

A Praxis Presented to the Graduate Faculty of the

School of Engineering

Southern Methodist University

in

Partial Fulfillment of the Requirements

for the degree of

Doctor of Engineering

with a

Major in Engineering Management

by

Yang Peng

B.S., Information Engineering, Beijing University of Posts & Telecommunications, 1995
M.B.A, Business Administration, Cranfield University, 2000
M.S., Management Information System, Northern Illinois University, 2001

December 21, 2019

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor Prof. Dick Barr for the continuous support of my Doctor of Engineering study and related research, for his patience, motivation, and immense knowledge. He continually and convincingly conveyed a spirit of discovery in regard to research and scholarship, and an excitement in regard to teaching. His guidance helped me in all the time of research and writing of this praxis. Without his guidance and persistent help this praxis would not have been possible.

Besides my advisor, I would like to thank the rest of my praxis committee: Prof. Eli Olinick, Prof. Jeff Tian, Prof. Harsh Gangammanavar and Dr. John Medellin, for their insightful comments and encouragement, but also for the helpful questions which incented me to further my research from various perspectives.

In addition, a thank you to Professor Volkan Otugen, who introduced me to Engineering Management of Information System of Southern Methodist University, and whose enthusiasm and support of my Doctor of Engineering study and related research.

I thank my fellow colleagues in the following entity, UR Inc. for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the experiments.

Last but not the least, I would like to thank my family for supporting me spiritually throughout writing this praxis and my life in general.

Peng, Yang B.S., Information Engineering, Beijing University of Posts & Telecommunications, 1995
M.B.A, Business Administration, Cranfield University, 2000
M.S., Management Information System, Northern Illinois University, 2001

Resource Allocation and Task Scheduling Optimization
in Cloud-Based Content Delivery Networks
with Edge Computing

Advisor: Dr. Richard Barr

Doctor of Engineering degree conferred December 21, 2019

Praxis completed November 25, 2019

The extensive growth in adoption of mobile devices pushes global Internet protocol (IP) traffic to grow and content delivery network (CDN) will carry 72 percent of total Internet traffic by 2022, up from 56 percent in 2017 [1]. Cloud-based CDN with edge computing (EC) provides a distribution of cloud computing capabilities to the edge network.

In this praxis, Interconnected Cache Edge (ICE) based on different public cloud infrastructures with multiple edge computing sites is considered to help CDN service providers (SPs) to maximize their operational profit. The problem of resource allocation and performance optimization is studied in order to maximize the cache hit ratio with available CDN capacity.

The considered problem is formulated as a multi-stage stochastic linear programming model that involves jointly optimizing the resource allocation and network performance. The problem is challenged in reality since the multi-cloud SPs have dynamic price strategies in different regions, tasks could be time sensitive, and busy-hour traffic model is hard to simulate. To overcome these challenges, the praxis proposes a method to decompose the problem into (i) a resource-allocation problem with fixed task-offloading decisions and (ii) a performance optimization problem that optimizes the cache hit ratio, round-trip time (RTT) and edge processing time corresponding to the resource allocation.

The praxis addresses the problem using optimization solvers of the General Algebraic Modeling System (GAMS) and proposes a broker scheme (ICE: Interconnected Cache Edge) using cloud-based CDN with edge computing architecture to maximize expected profit. Experimental design shows that ICE performs closely to the optimal solution and that it significantly improves the CDN profitability and network performance over traditional approaches.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

This praxis is dedicated to my mother and father, Shaoqin Qian and Ningshi Peng, who first taught me the value of education and life skills.

I also dedicate this praxis to my daughter, Xiaohan Peng, who is just a little girl who grows up to be my best friend I laugh with, and love with all my heart.

Chapter 1

Introduction

The extensive growth in adoption of mobile devices has led to a continuous increase in the average number of devices and connections per household and per capita. According to a recent report from Cisco [1], the number of devices connected to Internet Protocol (IP) networks will be more than three times the global Population by 2022 and smart phone traffic will exceed personal computer (PC) traffic. Annual global IP traffic will reach 4.8 Zettabytes (ZBs) per year by 2022 and global IP traffic will increase threefold over the next five years. Busy-hour Internet traffic is growing more rapidly than average Internet traffic. These are leading to greater investment in *Content Delivery Network* (CDN) and *Edge Computing* (EC) technologies.

A CDN is a system of geographically distributed servers (network) that delivers pages and other web content to a user, based on the geographic locations of the user, the origin of the webpage, and the content-delivery servers. CDN service is effective in speeding the delivery of the content of websites that have high traffic and a global reach. To minimize the distance between the users and the website's origin server, the *cache servers* store a cached duplicate version of the website's content in multiple *points of presence (PoPs).* The closer the CDN cache servers are to the users geographically, the faster the content can be delivered instead of using the origin server (Figure 1.1).

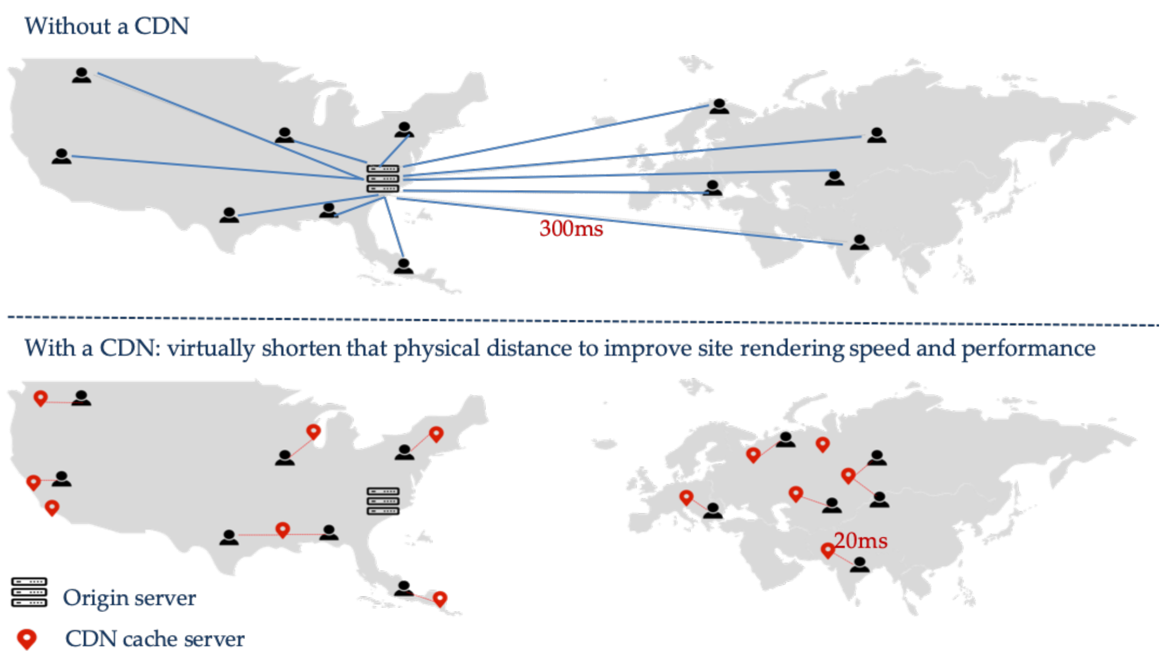

Without a CDN
300ms
With a CDN: virtually shorten that physical distance to improve site rendering speed and performance
20ms
Origin server
CDN cache server

Figure 1.1. How do CDNs work?

Current changes in mobile and video traffic topology strengthened the role of CDNs in data and content delivery. CDNs will carry 72 percent of total Internet traffic by 2022, up from 56 percent in 2017 (Figure 1.2). The delivery algorithms and scheduling methodologies used by CDNs could be more important than the speeds and latencies offered by the service providers for network performance [2].

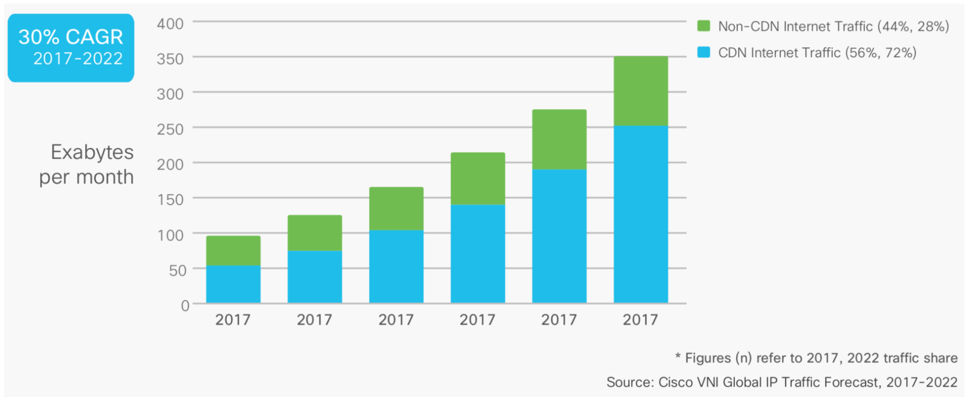


Figure 1.2. Global CDN Internet traffic, 2017 and 2022 [1]

Increasing utilization of network resources is one of the biggest challenges for mobile network operators. Internet bandwidth demands from smart phones, users' wearable devices, and the Internet of Things (IoT) force network operators to enhance and upgrade capacities of existing network resources continuously. With the increased computational power and expanded storage capacity of the mobile devices, creative application scenarios become realistic, although requiring much better infrastructure for a good user experience [3]. *Edge computing*, in which computing and storage nodes are located in close proximity to mobile devices to deliver highly responsive network services for mobile computing and content delivery, has been proposed by European Telecommunications Standards Institute [4] to reduce network stress by shifting computational and storage efforts from the core network to the edge network. As a consequence, devices deployed at the edge could not only act as access points but also could resolve many user requests. Multi-Access Edge Computing (MEC) has emerged as a solution to bring computing and content storage to the edge of a mobile network, even to the radio-access part of it [5].

1.1 Motivation

In the traditional CDN architecture invented by Akamai [6], data and content are dynamically replicated to cache servers in selected geographical regions to serve users in close proximity (Figure 1.3). *Service Providers* (SPs) have little flexibility on allocating underlying resource (e.g., networking, computation and storage) to offer a flexible pricing scheme for the users. As the busy-hour traffic continues to grow more rapidly than ever, the traditional CDN architecture fails to provide scalability [7]. The peer-to-peer (P2P) CDN architecture depends on the end-users (peer nodes) to store data and content and share among peers to gain scalability, but this can cause problems with privacy, copyright protection, and version control.

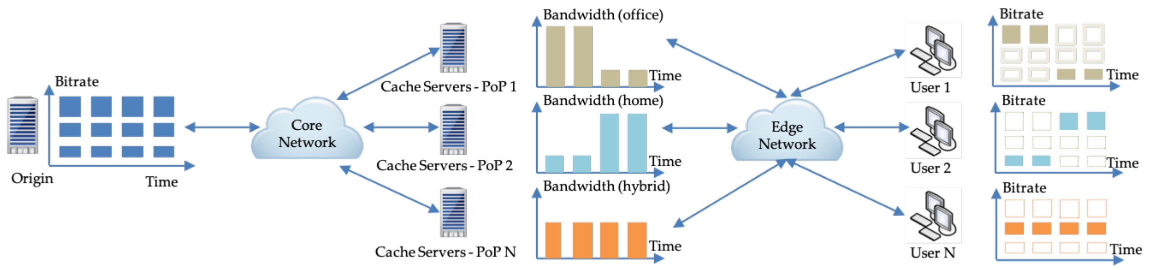


Figure 1.3. Traditional CDN architecture

Cloud computing or *cloud* is the on-demand availability of computer system resources (especially data storage and computing power) and is generally used to describe data centers available to many users over the Internet [8]. *Cloud-based CDN* is based on a cloud-computing infrastructure that provides scalable and on-demand resource allocation to provide cost-efficient, secure content distribution. Amazon CloudFront and Google Cloud CDN [9, 10] have suggested that a CDN could built on the infrastructure of a public cloud; SPs or *content providers* (CPs) could build CDN infrastructure by renting virtual machines (VMs) to deploy CDN services.

Cloud-based CDN leverages globally distributed edge points of presence (PoPs) on public cloud to accelerate content delivery for websites and applications. To deliver data and content faster to the end users while reducing serving costs, CPs tend to use cloud-based CDNs [10]. Before starting to use cloud-based CDNs, CPs need to select the regions to install cache servers and invest the *capital expenditure* (CAPEX) to establish the CDN infrastructure. With a cloud-based CDN architecture, infrastructure providers (InPs) establish virtualized server, storage, and network on demand and cloud-based CDN service providers construct cloud-based CDNs to serve the end users by setting up cache servers based on the virtualized server, storage, and network.

The cloud-based CDN — in which cache servers are provided by virtualized server, storage, and network on public cloud — dramatically changes traditional CDN services. For example, Verizon Digital Media Services leverage Google Cloud Platform (Figure 1.4) to move content between Verizon and Google directly without traversing other networks, thereby providing increased availability and reduced latency for important content. Verizon Digital Media Services customers that use Google Cloud Platform can save more than 65 percent on their cloud egress costs for an easier, more cost-effective delivery path that is optimized to move and scale content between Google Cloud Platform and Verizon's CDN [10]. Verizon Services' global footprint combined with Google's powerful network infrastructure offers users a better experience.

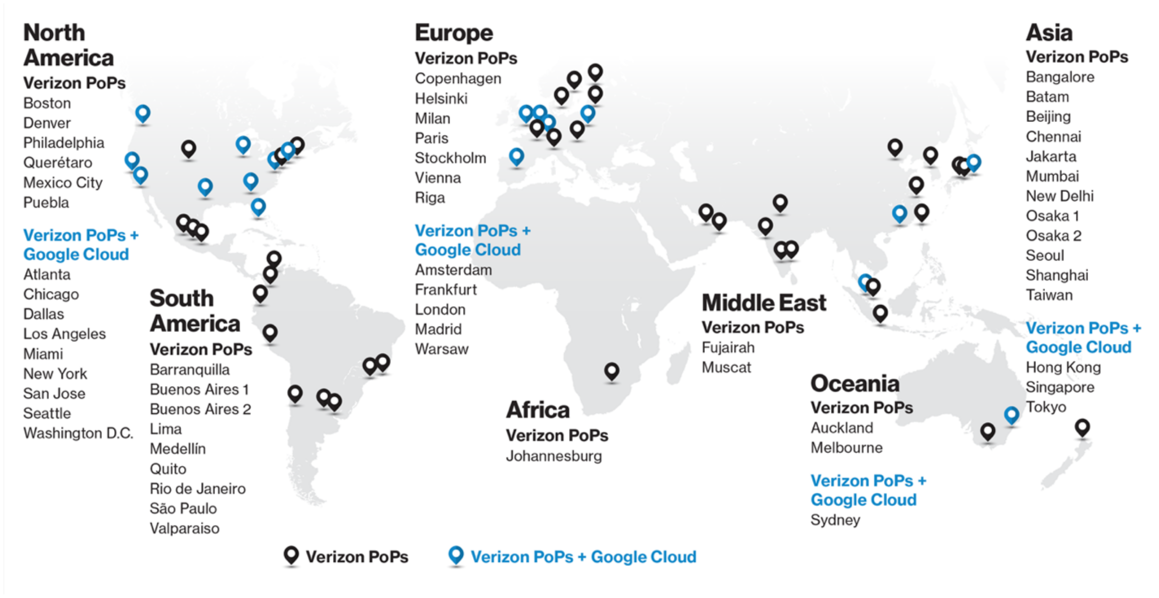


Figure 1.4. Verizon Digital Media Services Connects with Google Cloud Platform

Edge computing is a distributed computing paradigm that brings computer data storage closer to the location where it is needed. To alleviate the network resource limitation of the SPs and reduce the network latency, edge computing, which provides mobile devices in close proximity with low-latency and low-cost data exchange, is a promising approach. Before starting to use edge computing, cloud computing is the suggested scheme and the workload and traffic for mobile applications are offloaded to the centralized cloud or datacenters. In this way, the execution time of the mobile applications and the energy consumption of the mobile devices are reduced. However, due to the cloud or datacenters deployed distantly from the mobile devices, offloading the mobile applications to the remote cloud or datacenters occupies substantial network bandwidth, resulting in high network latency. Edge computing can push datacenters or containers with allocated resources, base stations, and access points to the edge of a multi-access network, thereby providing nearby resources for the mobile devices [5]. In this way, edge datacenters or containers connect to mobile devices via a Local Area Network (LAN) that supports high bandwidth and low network latency. Hence, edge computing can reduce offloading latency and make the network more efficient.

In computing, a task is a unit of execution or a unit of work. To fully utilize the computing resources, task scheduling aims to distribute tasks in order to make them more efficient in the use of limited computing resources. Although average Internet traffic has maintained a steady growth pattern, busy-hour traffic continues to grow more rapidly [1]. Therefore more computing resources and network capacity are required for SPs to satisfy the users' rush-hour demand than in the past. Video is the underlying reason for accelerated busy-hour traffic growth because of its consumption patterns: higher peak-to-average ratio. Moreover, real-time video—such as live video, ambient video, and video calling—has a higher peak-to-average ratio than on-demand video [1]. Thus, optimized task scheduling could reduce the processing time of the tasks, average the utilization of computing resources, and improve the network performance [11].

1.2 Basic Structure and Economics of CDN

The basic building blocks of CDN infrastructures are PoPs (points of presence), cache servers, and solid-state and hard-disk drives (SSD and HDD) (Figure 1.5). PoPs are regional data centers that hold multiple servers and routers responsible for caching, connection optimization, and other content-delivery features. Cache servers are responsible for the storage and delivery of cached files to accelerate content load time and reduce bandwidth consumption. Cache servers act as a repository for website content, providing local users with accelerated access to cached files. The closer a cache server is to the end user, the shorter the connection time needed for transmission of website data. Each cache server typically holds multiple storage drives and high amounts of random-access memory (RAM) resources.

Cached files are stored on SSD and HDD or in RAM. RAM is the fastest and used to store the most frequently-accessed items.

Figure 1.5. Basic building blocks of a CDN

A CDN is typically a multi-tenant infrastructure [12]: its resources could be shared among multiple CPs (Figure 1.6). The transit network among origin servers, cache servers, and users allows network traffic to transit.

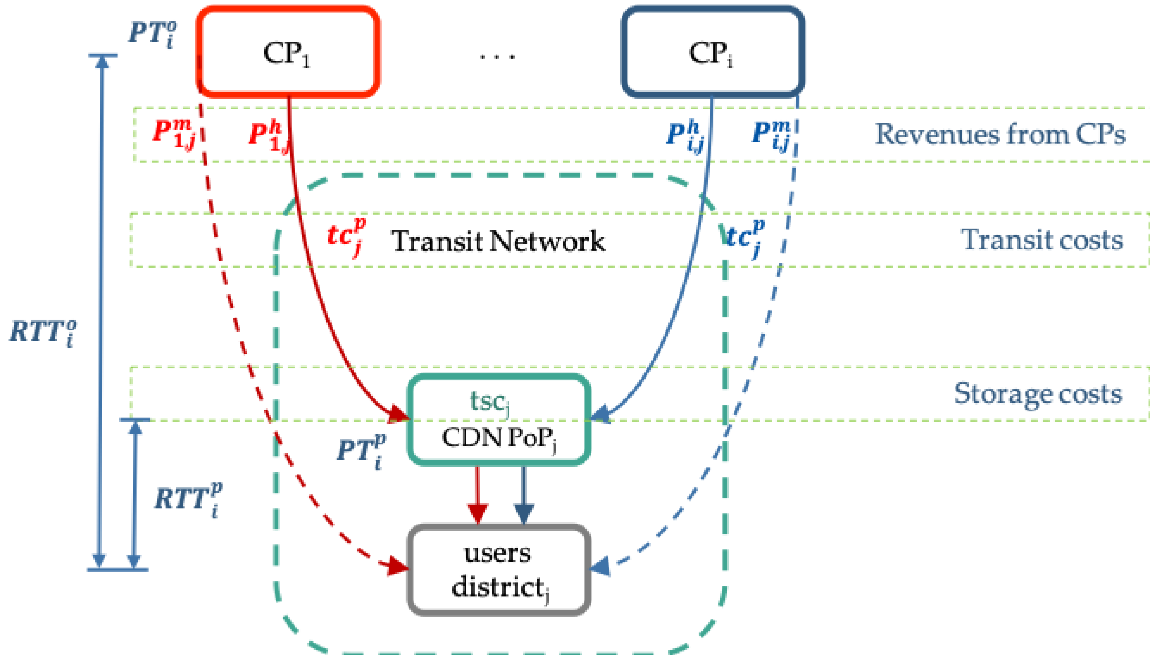


Figure 1.6. Prices and costs of CDN

In describing the various CDN components, I is the set of content providers, J is the set of PoPs or districts, and K is the set of regions, in the CDN system. In Figure 1.6, CP_i, $i \in I$, denotes the CPs that will be charged by SPs at the following unit prices:

1. $P^h_{i,j}$, 000s of USDs per G (k\$/G) for CP_i, $i \in I$, if content delivered from PoP j, $j \in J$ to users (cache-hit);

2. $P^m_{i,j}$, 000s of USDs per G (k\$/G) for CP_i, $i \in I$, if content delivered from origin server to users in district j, $j \in J$, (cache-miss).

SPs have two types of costs: tsc_j, $j \in J$, the total infrastructure fixed costs of PoP_j and tc^p_j, $j \in J$, the transit bandwidth variable costs between PoP_j, and the users.

Storage costs are related to the storage capacity, aggregating hardware, and data centers' costs. Bandwidth costs are typically priced per megabit per second per month, and the customers are often required to commit to a minimum volume of bandwidth with a minimum term of service, usually using a 95^{th} percentile burstable billing scheme. Some bandwidth agreements provide Service-level Agreements (SLAs) which purport to offer money-back guarantees of performance.

Burstable billing is a method of measuring bandwidth based on peak use, which allows usage to exceed a specified threshold for brief periods of time without the financial penalty of purchasing a higher committed information rate (CIR, or commitment) from an Internet service provider (ISP). The 95^{th} percentile burstable billing measures/samples the bandwidth from the switch or router and recorded in a log file. At the end of the month, the top 5% of data is thrown away and the next highest measurement becomes the billable use for the entire month. Based on this model, the top 36 hours (top 5% of 720 hours) of peak traffic is not taken into account when billed for an entire month. Conversely, if peak traffic only appears for a brief instant and no additional traffic is generated, the billing amount can be substantially higher than average usage billing.

The key metrics to evaluate the performance of CDN services are:

1. Round-trip time (RTT) — the duration for a network request to go from a starting point to a destination and back again to the starting point. RTT^o_i, $i \in I$, is the

round-trip time between CP_i's origin server and the users while RTT_i^p, $i \in I$, is the round-trip time between cache servers in the PoPs and the users.

2. Cache-hit ratio — the proportion of the requests serviced by cache servers.
$$Cache\ hit\ ratio = \frac{Number\ of\ cache\ hits}{Number\ of\ cache\ hits + Number\ of\ cachemisses}$$

3. Edge-processing time — the time spent by the cache servers that replaces the processing time by the origin servers.

1.3 Cloud-based CDN Organization and Economics

CDNs have played a valuable role in hosting and distributing content to users for decades while public cloud providers own a number of globally distributed data centers that are expanding continuously [13]. A cloud-based CDN could be designed to set up CDN cache servers on top of several cloud operators, such as Amazon AWS service, Microsoft Azure, Google Cloud, or OpenStack-managed cloud. Cache servers built on multiple public clouds can be used to serve the CDN users. The costs and performance of the cache servers on different clouds or in different regions could be very different (Figure 1.7).

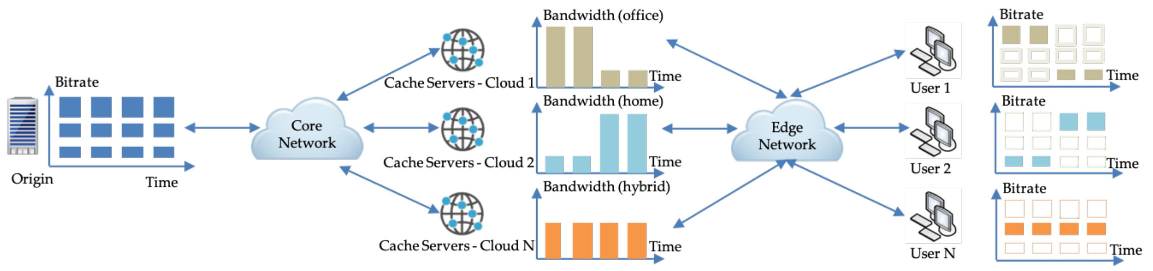


Figure 1.7. Cloud-based CDN architecture

Cloud-based CDN brings the following benefits:

1. Lowered barrier to entry for SPs by eliminating the massive CAPEX required to setup cache servers at selected geographical regions,

2. Increased bargaining power of CPs or users to use cloud-based CDN services with usage-based fee, and

3. Leveraged the marginal capacity of the infrastructure of public cloud to reduce the costs of CDN services

To serve users from different regions, SPs could rent virtual machines from multiple public cloud infrastructure providers (InPs) located in different regions for content caching and video streaming as illustrated in Figure 1.7. Each region, which corresponds to a cloud-based cache server, provides data and content delivery services with the lowest latency possible. All the cloud-based cache servers would be interconnected and for end users of the regions without cloud-based cache servers, data or content needs to be pulled from cache servers in other regions or original servers. This will degrade the user experience and affect the CPs' revenue streams. Therefore, it is necessary for the cloud-based CDN architecture to optimize the resource allocation, such as the selection of geographical regions, the capacity of cache servers built for different regions, and the bandwidth resourced in different regions. Cloud-based CDN services could be provided by several cloud operators and CPs can use virtualized cache servers provided at different regions. The costs and performance of cloud-based CDN services by different operators or in different regions could be very different. It is important to design the cloud-based CDN to maximize profit for SPs with required performance and optimized resource costs.

Set K represents the set of public cloud administration domains or regions in the cloud-based CDN system. CPs could be served by the cloud-based CDN as in Figure 1.8. CP_i, $i \in I$, that will be charged by SPs at the following unit prices:

1. $C^h_{i,k}$, 000s of USDs per G (k\$/G) for CP i, $i \in I$, if content delivered from cloud k, $k \in K$, to users (cache-hit);

2. $C^m_{i,k}$, 000s of USDs per G (k\$/G) for CP i, $i \in I$, if content delivered from origin server to users in region k, $k \in K$, (cache-miss).

SPs have two types of variable costs, pcc_k, $k \in K$, the total infrastructure costs of public $cloud_k$, $k \in K$, and tc^c_k, $k \in K$, the transit bandwidth costs between $cloud_k$, $k \in K$, and the users.

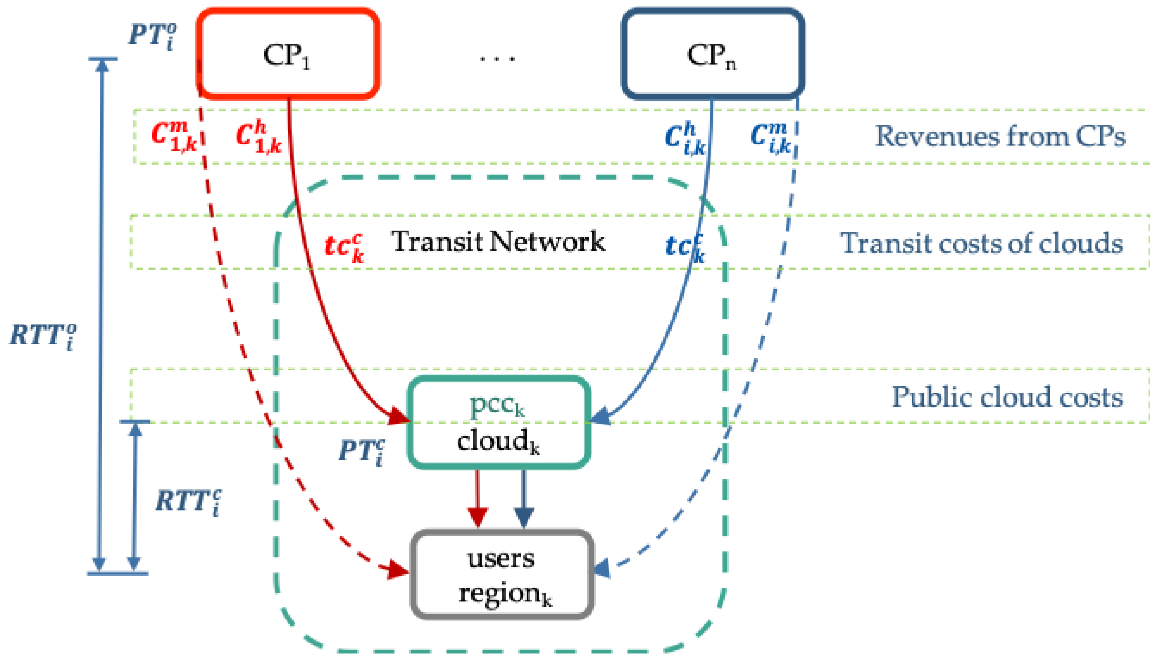


Figure 1.8. Prices and costs of Cloud-based CDN

RTT_i^c, $i \in I$, is the round-trip time between the cache servers on the public clouds and the users for CP_i, $i \in I$. PT_i^c, $i \in I$, is the processing time by the cache servers on the clouds for CP_i, $i \in I$.

1.4 Edge Computing Organization and Economics

The aim of edge computing is to deliver compute, storage, and bandwidth much closer to data inputs and/or end users. Cloud edge computing can mitigate the effects of widely distributed sites by minimizing the effect of latency on the applications. Edge computing first emerged by virtualizing network services over WAN networks and the rapid growth of mobile devices have driven the need for services at the network edge in close proximity to the end users [4]. In the traditional cloud computing systems where remote public clouds are utilized, the execution of mobile applications substantially increases the latency and the burden on the back-haul networks. As traffic from wireless and mobile devices will account for 71 percent of total IP traffic by 2022 [1], the back-haul networks will need further investment and upgrading to sustain network performance. Edge Computing has been suggested to the network operators to deploy edge servers directly at the local wireless Access Points (APs) or at the cellular Base Stations (BSs) using a generic-computing platform. Thereby the mobile applications could be executed in close proximity to users to minimize the back-haul network usage (Figure 1.9).

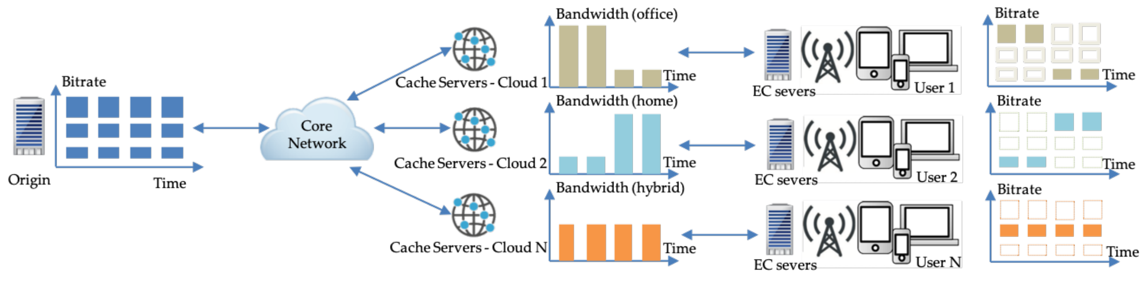


Figure 1.9. Cloud-based CDN with Edge Computing

This praxis suggests virtual machines (VMs) as the core visualization mechanism used by multi-tenant cloud-based CDN and edge computing sites to share disk space and CPU. Cloud-based edge computing moves the focus from PoP-based data center to more lightweight virtualized resources, distributed cloud to bring services to the users.

Task offloading from mobile devices to edge computing could generate extra overheads in terms of latency and energy consumption due to the communication required between the mobile devices and the edge computing servers. However, edge computing combines the speed of Cloud-based CDN with the benefits of cloud to enable a new generation of networking and move processing closer to the user. Edge Computing brings the following benefits:

1. Mobile devices with limited resource offloaded tasks to enable novel applications such as augmented reality, autonomous vehicles and image processing.

2. Edge Computing eliminated the need of routing data through the core network to increase performance of networks and save investment on core network.

3. Edge Computing supported large differences in site size and scale, from data center scale down to a single device.

Set E represents the set of edge computing sites or districts in the cloud-based CDN with edge computing system. CPs could be served by the cloud-based CDN with Edge Computing as (Figure 1.10). CP_i, $i \in I$, that will be charged by SPs at the following unit prices:

1. $C^h_{i,k}$, 000s of USDs per G (k\$/G) for CP i, $i \in I$, if content delivered from cloud k, $k \in K$, to users (cache-hit);
2. $C^m_{i,k}$, 000s of USDs per G (k\$/G) for CP i, $i \in I$, if content delivered from origin server to users in region k, $k \in K$, (cache-miss);
3. $E^h_{i,e}$, 000s of USDs per G (k\$/G) for CP i, $i \in I$, if content delivered from edge computing site e, $e \in E$, to users (cache-hit);
4. $E^m_{i,e}$, 000s of USDs per G (k\$/G) for CP i, $i \in I$, if content delivered from origin server to users in district e, $e \in E$, (cache-miss).

SPs have four types of variable costs, pcc_k, the infrastructure costs of public $cloud_k$, $k \in K$, ecc_e, the infrastructure costs of edge computing $site_e$, $e \in E$, tc^c_k, the transit bandwidth costs between $cloud_k$, $k \in K$, and the users, and tc^e_e, $e \in E$, the transit bandwidth costs between edge computing $site_e$, $e \in E$, and the users.

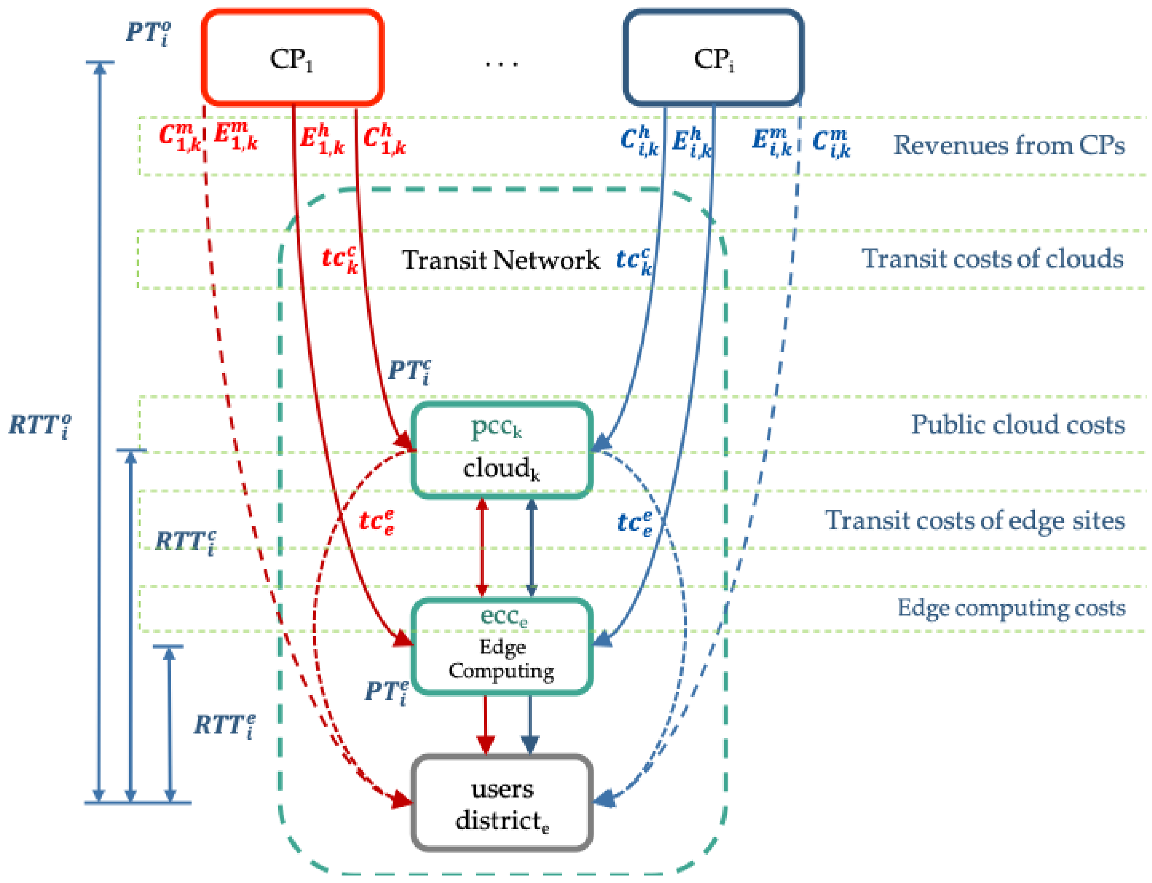


Figure 1.10. Prices and costs of Cloud-based CDN with Edge Computing

RTT_i^e, $i \in I$, is the round-trip time between the edge computing sites and the users. PT_i^e, $i \in I$, is the processing time by the edge computing sites.

1.5 Illustration of the Problem

In this praxis, we studied the profit maximization problem for cloud-based CDN with edge computing system from a CDN service provider's point of view and the performance optimization problem for mobile Internet applications from a content provider's perspective. The massive consumption of mobile Internet drives the requirement on content delivery and network optimization. The problem is a multi-stage resource allocation problem since at the beginning of every month, SPs need to make decisions on capacity costs and uncertain demand (Table 1.1).

Table 1.1. Multi-Stage Resource Allocation Planning

Multi-Stage Resource Allocation			
	Stage 1 (beginning of the month)	During the month	Stage 2 (end of the month)
Known at this time	-Selling price of CDN bandwidth -Costs and capacities for origin server -Costs and capacities for PoPs/cache servers -Costs and capacities for cloud-based cache servers -Costs and capacities for edge computing sites -Costs and capacities for bandwidth -Possible demand scenarios -Possible distribution of users -Processing time for each server -Round-trip time for each route -Performance requirement	Demand scenario becomes known	-Actual demand of bandwidth for each CP -Actual demand of each PoP -Actual demand of cloud for each region -Actual demand of edge computing sites -Total processing time for each CP -Total round-trip time for each CP -Cache-hit ratio for each CP
Unknown	-Actual demand of bandwidth for each CP -Actual demand of each PoPs/ -Actual demand of cloud for each region -Actual demand of edge computing sites -Total processing time for each CP -Total round-trip time for each CP -Cache-hit ratio for each CP		
Decisions needed	-How much bandwidth for SP to order for each district -How much capacity for SP to build for each PoP -How much cloud infrastructure for SP to build for each region -How much edge computing capacity for SP to build		-How much bandwidth used for each region -How much resource used for each CP in each PoP -How much cloud resource used by CP for each region -How much edge computing resource used by CP in each district

Decisions to order bandwidth and establish certain capacity were made in first stage by the SPs before the realization of the demand of the CPs is known in second stage.

The problem is twofold - firstly, profit maximization with fast capacity expansion, and secondly, performance improvement with increased customer demand.

The objectives are profit maximization:

$maximize\ revenue - bandwidth\ costs - storage\ costs - cloud\ costs - edge\ computing\ costs$

and performance optimization:

$maximize\ cache\ hit\ ratio$ and $minimize\ round\_trip\ time + processing\ time$

The praxis formulates *a multi-stage stochastic linear programming model* for mathematically modeling cloud-based CDN with edge computing system to allocate certain resources in different regions and schedule different tasks to maximize profit with required performance. It considers all the possible futures or scenarios under a probabilistic framework. The problem is challenged in reality since the multi-cloud SPs have dynamic price strategies in different regions, customer demand could be seasonal, and busy-hour traffic model is hard to simulate. To overcome these challenges, this praxis proposes a method to decompose the problem into (i) a resource-allocation problem for SPs with fixed content distribution decisions and (ii) a performance improvement problem for CPs corresponding to the resource allocation.

The praxis introduces an Interconnected Cache Edge (ICE) platform that allows dynamic deployment of cloud-based CDN with edge computing system running across multiple administrative cloud domains and optimizes resource allocation in different geographic regions. Furthermore, the cloud-based CDN with edge computing system can improve the distribution performance by caching content nearby end-users and reduce the service latency by decreasing the total processing time of content.

1.6 Literature Review

1.6.1 Resource Allocation

Um et al. [14] proposed a suggestion for cloud-based CDN to enable virtual machines to be scaled to satisfy the dynamically changing resource demand of CDN services and evaluated the performance based on a simulation. Yala et al. [15] focused on how to appropriately decide on the amount of computing resources to allocate to a CDN-as-a-service (CDNaaS) task to satisfy Quality of Experience (QoE) and resource capacity constraints to drive a QoE-aware virtual CPU resource allocation algorithm. Sharmin et al. [16] developed a resource allocation algorithm to select suitable sizes of video blocks by the number of parallel streams provisioned in a single virtual CPU (vCPU) and explored how to balance the workload among the vCPUs.

Haghighi et al. [17] formulated a two-stage resource provisioning and cloud assignment based on dynamic large and small-scale fluctuations of user demand rates as well as considering a constrained minimum lease time for resources. Zheng et al. [18] mathematically formulated an Improved Heuristic Genetic Algorithm for Static Content Delivery in Cloud Storage (IHGA-SCDCS) based on a resource management model and cost model. Hu et al. [7] presented a community classification method and formulated a stochastic optimization framework to reduce the monetary cost with the same latency. Yala et al. [19] provided polynomial-time heuristics to derive an assignment of computing resources to a set of virtual instances and formulated a multi-objective optimization problem to balance among conflicting objectives.

Several works have investigated methods of resource allocating in cloud-based CDN. The work in [20] explained the multi-objective resource provisioning problem to minimize virtual server, storage, and network cost and maximize the network performance for the end-user. Iturriaga et al. [20] suggested a brokering model that a single cloud-based CDN is able to host multiple CPs applying a resource sharing strategy. Unlike [20], Benkacem et al. [13] introduced a CDN-aaS platform that allows dynamic deployment and life-cycle management of cloud-based CDN slices running across multiple administrative cloud domains. [13] formulated the virtual network function (VNF) resource placement problem as two linear integer problem models to minimize the cost and maximize the quality of experience (QoE) of the virtual streaming service.

Mobile edge computing (MEC) was proposed in several works to offload task. Mach and Becvar [21] addressed MEC resources can be utilized by operators and third parties with different cases and reference scenarios. Nguyen et al. [22] proposed a scheme to allocate resources of heterogeneous capacity-limited edge nodes to multiple competing services. Chen et al. [23] adopted a game theoretic approach to formulate the distributed computation offloading decision-making problem among mobile device users. Samanta et al. [24] designed an adaptive service-offloading scheme in MEC for both delay-tolerant and delay-constraint services to optimize latency and maximize revenue.

Joint task offloading and resource allocation not only improve the network performance but also save the energy consumption of the mobile devices. Chen and Hao [25] formulated the task-offloading problem as a mixed integer non-linear program by leveraging software defined network to minimize the delay while saving the battery life of mobile devices in ultra-dense network. Tran and Pompili [26] studied joint task offloading and resource allocation to maximize the reductions in task completion time and energy consumption. [26] mathematically formulated a mixed integer nonlinear program (MINLP) to resolve the resource allocation problem with fixed task offloading decision in MEC. Furthermore, Li et al. [27] proposed an on-line computation rate maximization (OCRM) algorithm for multi-user by jointly managing the radio, computational resources, allocating time for data transmission, and energy saving.

In summary, most of the existing works did not consider a method to enable coordinated and cooperative content delivery via internetworking among multiple administrative cloud domains and edge computing sites to maximize profit by dynamic resource allocation supported by multi-cloud with edge computing sites as considered in this praxis.

1.6.2 Task Scheduling

An efficient task-scheduling mechanism to allocate resource will improve the resource efficiency significantly. Yi et al. [28] formulated a Mixed Integer Linear Programming (MILP) with joint resource (computing, storage and network) provision to propose a best-fit heuristic algorithm with different task scheduling policies. This method minimizes the expenditure for each user to obtain enough resources for task execution while taking as many tasks as possible. As edge computing becomes an increasingly popular alternative to cloud computing for resource allocation, Li et al. [11] purposed data-placement optimization and task scheduling to reduce the computation delay and response time in cloud computing. In this method, containers were deployed as the smallest resource unit for task scheduling to fully utilize the storage in edge servers to improve the overall performance. As edge computing can provide a low-latency and cost-effective computing capacity for task execution, Shao et al. [29] introduced a replication management system with a specialized task scheduler for data placement based on a mixed integer programming formulation.

As Internet applications have shifted from simple web browsing to real-time video, it is more crucial for SPs to reduce the operational costs while increasing the performance and responding timely. Task scheduling plays an important role for live-video streaming distribution to maximize the performance. Many contribution focused on scheduling resource to be allocated to task with constraint. Melika Meskovic and Mladen Kos [30] solved the problem by the deadline requirement of the tasks and proposed a chunk scheduling algorithm for layered live-video streaming in mesh pull-based CDN-P2P network. Scoca et al. [31] proposed a score-based edge service scheduling algorithm to evaluate network and computational capabilities of edge nodes and match between tasks and resources.

As many CPs take public cloud as their primary infrastructure for content storage and large-scale computations, task assignment and scheduling is one of the main problems to be investigated in a cloud computing environment. Reddy and Kumar [32] recommended a task scheduling method based on Modified Ant Colony Optimization (MACO) algorithm which suggested to perform Multi Objective Task Scheduling (MOTS) process by assigning pheromone amount relative to corresponding virtual machine efficiency in a cloud computing environment. To reduce the response time, operational costs, and energy consumption, Naik et al. [33] addressed a hybrid multi-objective heuristic algorithm based on Non-dominated Sorting Genetic Algorithm-II (NSGA-II) and Gravitational Search Algorithm (GSA) called as NSGA-II & GSA to allocate virtual machines from different data centers for task scheduling.

Designing an efficient task scheduling strategy for multi-cloud system could be very challenging since the available computing capacity and network topology could be very complex. The work in [34] and [35] addressed the task scheduling problem for multi-cloud environment, Wang et al. [34] presented a multi-cloud supported resource allocation and scheduling optimization strategy through the big data analysis on daily CDN operation. This method designed a multi-cloud extension algorithm to schedule extra cloud resource to handle overload requests and use algorithm to shift tasks to additional cloud resource prepared. Kang et al. [35] proposed a Dynamic Scheduling Strategy (DSS) to integrate the Divisible Load Theory and node availability prediction techniques in a multi-cloud environment.

In summary, most of the existing works did not consider a method that determines the task scheduling decision for both the busy-hour traffic and average Internet traffic to leverage the resource allocation to maximize profit and optimize performance as considered in this praxis.

1.6.3 Cloud-based CDN with Edge Computing

The basic framework to construct a cloud-based CDN based on cloud computing was proposed in [36], [14]. [14] proposed a cloud-based CDN architecture to provide cost-efficient and elastic CDN services by multiplexing a number of video service applications with different service level agreements (SLAs) into a virtual machine. Several works have investigated methods of allocating the location of caching servers in the cloud-based CDN. Noriaki Kamiyama and Yutaro Hosokawa [37] proposed a method of optimally selecting the geographical regions to use cache servers within a single public cloud to maximize profit for SPs. Edge Computing enabled computing and storage infrastructure provisioned closely to the end-users. Cloud-Based CDN with Edge Computing such as MEC would improve the scalability [7] as the geographical regions selection would be more flexible.

Cloud-based CDN is the common network function for mobile devices to enhance media availability and distribution performance. Multi-access Edge Computing (MEC) architecture is designed to enhance or boost media services. The work in [5], [11] and [38] addressed performance-aware system to enhance multimedia services. Viola et al. [38] proposed a MEC proxy to perform a local cache to minimize the traffic between the CDN and the edge servers and shield from identified or predicted CDN malfunction. MEC proxy could reduce the CAPEX and ensure performance for the SPs.

MEC enables mobile devices suitable for latency-sensitive applications. Combining MEC with cloud-based CDN infrastructures enables mobile devices to take advantage of the virtually unlimited resource capacity of clouds. Non-time-critical tasks can be scheduled and offloaded to the clouds. Dreibholz et al. [39] introduced a baseline combining multiple cloud systems and MEC into a unified MEC-multi-cloud platform to provide guidelines for designing an autonomic resource provisioning solution. Unlike [39] and given MEC could be limited in terms of the high infrastructure deployment and maintenance cost, Wang et al. [40] proposed a smart, Deep Reinforcement Learning based Resource Allocation (DRLRA) scheme, which can allocate computing and network resources adaptively, reduce the average service time of MEC and balance the use of resources.

In summary, most of the existing works did not consider a holistic approach to maximize profits and optimize performance by the joint deployment of cloud-based CDN and edge computing as considered in this praxis.

1.7 Approach and Methodology

The praxis is organized as follows.

1. Chapter 1 introduces the challenges and reviews the related works.
2. Chapter 2 presents and formulates the problem.
3. Chapter 3 experimental designs the evaluation criteria and explores factors.
4. Chapter 4 analyses numerical results.

5. Chapter 5 concludes this praxis and identifies the future research directions.

The praxis formulates a multi-stage stochastic linear programming model for mathematically modeling cloud-based CDN with edge computing system to allocate certain resources in different regions and schedule different tasks to maximize profit with required performance. The approach includes:

1. A process for designing a conceptual mathematical model to provide the framework for identifying key factors (variables) relating to performance.
2. Illustrative applications of the multi-stage stochastic linear programming model with data for resource allocation and performance optimization.

The praxis applied the following methodology and process steps.

1. Determine the purpose of the model and the level of detail
2. Define the objective and as many constraints as possible
3. Identify the decision variables
4. Mathematically formulate the problem
5. Resolve the problem with the General Algebraic Modeling System (GAMS) solver
6. Experimental setup and test prototype for Interconnected Cache Edge (ICE)
7. Scale up the data and analyze the results
8. Conclude the study

1.8 Expected Contributions

To maximize profit in cloud-based CDN with edge computing system by joint resource-allocation and performance optimization, there are several key challenges that need to be addressed.

1. The cloud-based CDN resource allocation is more complex in reality than the cases studied in [20] and [13] due to the multi-cloud providers have different price strategies in different regions.

2. The complexity of the task scheduling decision is high as the applications could be time sensitive and the busy-hour traffic model is changing.

3. The optimization model should also take into account the inherent heterogeneity in terms of capabilities of cache server, availability of resources at edge, and performance requirements in different regions.

The main contributions of this praxis are summarized as follows.

1. Formulated the problem of resource allocation and performance optimization as a multi-stage stochastic linear programming model in a multi-supplier, multi-cloud with edge computing environment to maximize profit.

2. Proposed a method to decompose the problem into (i) a resource-allocation problem with fixed content distribution decisions and (ii) a performance improvement problem for different mobile Internet applications corresponding to the resource allocation.

3. Developed a broker scheme (ICE: Interconnected Cache Edge) for cloud-based CDN with edge computing system leveraging the multi-supplier and multi-cloud environment to maximize profit and improve performance.

Chapter 2

Problem Description and Formulation

There are mainly three systems on content delivery studied in this chapter: traditional CDN, cloud-based CDN and cloud-based CDN with edge computing. The praxis assumes the content is delivered to end-users based on their geographical locations and availability of resources for every systems. Performance and reliability have become the major factors that directly impact the user experience. For ease of reference, the notation for key system elements and parameters used in the article are summarized in Table 2.1.

Table 2.1: Summary of Notation for Key System Components and Parameters

Notation	Description
I	Set of content providers
J	Set of districts/PoPs
K	Set of regions/administrative public cloud domains
E	Set of edge computing sites/districts
D	Set of expected demand scenarios: idle-hour, normal-hour and busy-hour
CP_i	Content provider, $i \in I$
S_d^p	Probability of each demand scenario, $d \in D$
$LD_{i,d}$	Expected demand of CP i, $i \in I$, under scenario, $d \in D$
$pud_{i,j}$	Expected demand distribution for district j, $j \in J$, by CP i, $i \in I$
$P_{i,j}^h$	000s of USDs per G for CP i, $i \in I$, if content delivered from PoP j, $j \in J$ to users (cache-hit)
$P_{i,j}^m$	000s of USDs per G for CP i, $i \in I$, if content delivered from origin server to users in district j, $j \in J$ (cache-miss)
PCA_j	CDN capacity (G) for PoP j, $j \in J$
sc_j^p	Unit storage costs (000s of USDs per G) of PoP j, $j \in J$
tc_j^p	Bandwidth purchase costs (000s of USDs per G) for PoP j, $j \in J$
hc_j^p	Bandwidth holding costs (000s of USDs per G) for PoP j, $j \in J$
pc_j^p	Bandwidth penalty costs (000s of USDs per G) for PoP j, $j \in J$

tsc_j	Total infrastructure costs (000s of USDs per G) of PoP j, $j \in J$
$cud_{j,k}$	Expected demand distribution for region k, $k \in K$, by related district j, $j \in J$
$C^h_{i,k}$	000s of USDs per G for CP i, $i \in I$, if content delivered from cloud k, $k \in K$ to users (cache-hit)
$C^m_{i,k}$	000s of USDs per G for CP i, $i \in I$, if content delivered from origin server to users in region k, $k \in K$ (cache-miss)
CCA_k	CDN capacity (G) of cloud k, $k \in K$
tc^c_k	Bandwidth purchase costs (000s of USDs per G) for cloud k, $k \in K$
hc^c_k	Bandwidth holding costs (000s of USDs per G) for cloud k, $k \in K$
pc^c_k	Bandwidth penalty costs (000s of USDs per G) for cloud k, $k \in K$
cc^c_k	Unit public cloud costs (000s of USDs per G) for cloud k, $k \in K$
pcc_k	Total infrastructure costs (000s of USDs per G) of cloud k, $k \in K$
$eur_{e,k}$	Expected demand distribution for edge e, $e \in E$, in region k, $k \in K$
$E^h_{i,e}$	000s of USDs per G for CP i, $i \in I$, if content delivered from edge computing sites$e$, $e \in E$ to users (cache-hit)
$E^m_{i,e}$	000s of USDs per G for CP i, $i \in I$, if content delivered from origin server to users in district e, $e \in E$ (cache-miss)
CCA_e	CDN capacity (G) of edge computing site e, $e \in E$
tc^e_e	Bandwidth purchase costs (000s of USDs per G) for edge computing site e, $e \in E$
hc^e_e	Bandwidth holding costs (000s of USDs per G) for edge computing site e, $e \in E$
pc^e_e	Bandwidth penalty costs (000s of USDs per G) for edge computing site e, $e \in E$
ec^e_e	Unit site costs (000s of USDs per G) for edge computing site e, $e \in E$
ecc_e	Total infrastructure costs (000s of USDs per G) of edge computing site e, $e \in E$
$minhit$	Minimum cache-hit ratio requirement (%) for CP_i
$maxrtt$	Maximum round-trip time requirement (%) for CP_i
PT^o_i	Processing time (ms) of origin server for CP i, $i \in I$
PT^p_i	Processing time (ms) of PoPs/cache servers for CP i, $i \in I$
PT^c_i	Processing time (ms) of public cloud cache for CP i, $i \in I$
PT^e_i	Processing time (ms) of edge computing site for CP i, $i \in I$
RTT^o_i	Round-trip time (ms) from origin server to the users (cache miss) for CP i, $i \in I$
RTT^p_i	Round-trip time (ms) from PoPs/cache servers to the users (cache-hit) for CP i, $i \in I$
RTT^c_i	Round-trip time (ms) from public cloud cache to the users (cache-hit) for CP i, $i \in I$
RTT^e_i	Round-trip time (ms) from edge computing site to the users (cache-hit) for CP i, $i \in I$

2.1 Traditional CDN

2.1.1 System Model

The architecture of a traditional CDN is considered as shown in Figure 2.1.

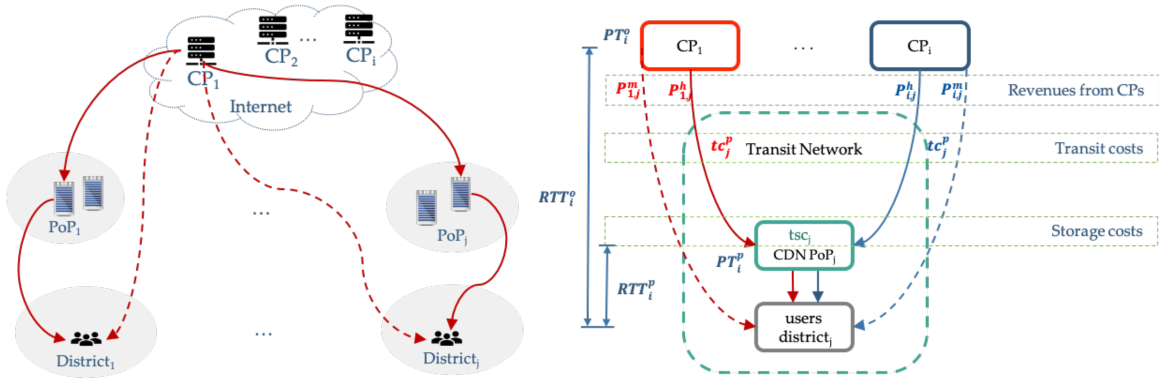


Figure 2.1. Traditional CDN Architecture

The praxis assumes that the traditional CDN service is provided to a set of CPs, I, that can use a set of CDN's PoPs/cache servers, J, to deliver content to a set of districts, J. Each district has one PoP and the district's users can only be served by the PoP in that district. A *cache-hit* is when the PoP's cache contains the requested content and the request is served by the cache servers in the requesting district. A *cache-miss* is when the PoP's cache does not contain the requested content and the request is passed along to the origin server. *Cache-hit ratio* is a measurement of the proportion or percent of content requests a cache is able to fill successfully of the requests it receives. For example, if a CDN has 95 cache hits and five cache misses over a given time-frame, then the cache-hit ratio is equal to 95 divided by 100, or 95%. CPs are charged for delivering content from origin servers to users (cache misses) or from cache servers to users (cache hits) at different unit selling prices.

Bandwidth costs, the unit purchase costs of delivering content from cache servers to users, depends on the districts of PoPs/cache servers reside. If the demand is less than the amount of bandwidth the service provider bought, the extra committed capacity will still need to be paid at a holding cost per unit. If there is more demand than the contracted capacity, the service provider must pay a penalty/higher cost per unit for the overage. On the other hand, storage costs, are mainly related to the storage capacity, aggregating hardware, and data centers' costs for each PoP. The minimal configuration of a PoP in a district is demonstrated as in Table 2.2.

Table 2.2. Minimal Hardware Configuration of a District PoP

	Configuration	Unit Price (USD)	Units	Subtotal (USD)
Switch	10G	3000	1	3000
LVS Server	Linux Virtual Server	2000	2	4000
Storage Server	48T	3500	4	14000
RAM Server	256G	4000	2	8000
SSD Server	12T	4500	2	9000
Total Costs (USD)				38000

The main risk for the traditional CDN is that it requires upfront investment on district PoPs to cover an unknown traffic peak level. This praxis sets up a consolidated CDN architecture with 20 high-capacity PoPs located in major data centers. With few PoPs, the maintenance and configuration propagations are rather effective, and the larger PoPs make sure that cache misses are kept low.

Round-trip time (RTT) is the duration in milliseconds (ms) required for a network request to go from a starting point to a destination and back again to the starting point. Reducing RTT is a major goal of a CDN, achieved by increasing the cache-hit ratios for the districts. Improvements in latency can be measured in the reduction of this round-trip time. Edge-processing time is the time spent by the cache server, which replaces the processing time by the origin server. The higher the cache-hit ratio, the more efficient the CDN.

2.1.2 Assumptions Made

The following assumptions are made in this model for designing and managing a traditional CDN system.

1. A two-stage stochastic model is appropriately to capture the CDN's uncertainties.
2. The model aims to maximize the expected profit of the CDN.
3. The decision for the SP to order bandwidth is made in first stage before the realization of the demand is known in second stage.
4. The decision for the SP to establish certain capacity for the PoPs is based on historical customer demand requirements.
5. The included demand scenarios and their corresponding probabilities of occurrence appropriately represent the demand uncertainty.

2.1.3 Multi-stage Recourse Planning

Table 2.3 illustrates the decision-making process of the multi-stage resource allocation planing for the traditional CDN system.

Table 2.3. Multi-Stage Resource Allocation Planning for Traditional CDN

Multi-Stage Resource Allocation for Traditional CDN			
	Stage 1 (beginning of the month)	During the month	Stage 2 (end of the month)
Known at this time	-Selling price of CDN bandwidth -Costs and capacities for origin server -Costs and capacities for district PoPs -Costs and capacities for bandwidth -Possible demand scenarios -Possible distribution of users -Processing time for each server -Round-trip time for each route -Performance requirement	Demand scenario becomes known	-Actual demand of bandwidth of each CP -Actual capacity demand of each PoP -Total processing time for each CP -Total round-trip time for each CP -Cache-hit ratio for each CP
Unknown	-Actual demand of bandwidth for each CP -Actual capacity demand of each PoP -Total processing time for each CP -Total round-trip time for each CP -Cache-hit ratio for each CP		
Decisions needed	-How much bandwidth for SP to order for each PoP -How much CDN capacity for SP to build for each PoP		-How much bandwidth used by each CP for each district -How much resource used for each PoP

Decisions to order bandwidth and establish certain capacity for the PoPs were made in first stage by the SPs before the realization of the demand of the CPs is known in second stage.

2.1.4 Decision Variables

The decision variables used in this model is described in the following table:

$XT_{i,d}$:	Gs of bandwidth used for CP i, $i \in I$, under scenario d, $d \in D$
PBT_j :	Gs of bandwidth the CDN should purchase at PoP j by SP, $j \in J$
$PC^h_{j,d}$:	Gs of cache-hit at PoP j, $j \in J$, under scenario d, $d \in D$
$PC^m_{j,d}$:	Gs of cache-miss at PoP j, $j \in J$, under scenario d, $d \in D$
$PC^n_{j,d}$:	Gs of bandwidth unused at PoP j, $j \in J$, under scenario d, $d \in D$
$DP^h_{i,j,d}$:	Gs of cache-hit for CP i, $i \in I$, at PoP j, $j \in J$, under scenario d, $d \in D$
$DP^m_{i,j,d}$:	Gs of cache-miss for CP i, $i \in I$, at PoP j, $j \in J$, under scenario d, $d \in D$

2.1.5 Mathematical Formulation and Explanation

Maximize Expected CDN Profit

$$\begin{aligned}\sum_{i\in I,j\in J,d\in D} DP^h_{i,j,d} * P^h_{i,j} * S^p_d + \sum_{i\in I,j\in J,d\in D} DP^m_{i,j,d} * P^m_{i,j} * S^p_d - \sum_{j\in J} PCA_j * sc^p_j \\ - \sum_{j\in J} PBT_j * tc^p_j - \sum_{j\in J,d\in D} PC^n_{j,d} * hc^p_j * S^p_d - \sum_{j\in J,d\in D} PC^m_{j,d} * pc^p_j * S^p_d\end{aligned} \tag{2.1}$$

subject to:

$$pdemand(d) : \sum_{i \in I} LD_{i,d} = \sum_{j \in J} PC^h_{j,d} + \sum_{j \in J} PC^m_{j,d} \tag{2.2}$$

$$pudeman(i,d) : LD_{i,d} \geq XT_{i,d} \tag{2.3}$$

$$ddemand(j,d) : \sum_{i \in I} LD_{i,d} * pud_{i,j} = \sum_{i \in I} DP^h_{i,j,d} + \sum_{i \in I} DP^m_{i,j,d} \tag{2.4}$$

$$capb(j) : PBT_j \leq PCA_j \tag{2.5}$$

$$capu(j,d) : PCA_j \geq PC^h_{j,d} + PC^m_{j,d} \tag{2.6}$$

$$cinv(j,d) : PC^n_{j,d} = PBT_j - PC^h_{j,d} \tag{2.7}$$

$$tcph(j,d) : PC^h_{j,d} = \sum_{i \in I} DP^h_{i,j,d} \tag{2.8}$$

$$tcpm(j,d) : PC^m_{j,d} = \sum_{i \in I} DP^m_{i,j,d} \tag{2.9}$$

$$hitrp(i,d) : \sum_{j \in J} DP^h_{i,j,d} \geq minhit * XT_{i,d} \tag{2.10}$$

$$tbght(i,d) : XT_{i,d} = \sum_{j \in J} DP^h_{i,j,d} + \sum_{j \in J} DP^m_{i,j,d} \tag{2.11}$$

$$Nonnegativity : XT, PBT, PC^h, PC^m, PC^n, DP^h, DP^m \geq 0 \tag{2.12}$$

The objective function simply maximize the expected profit and the following constraints should be followed:

1. total demand by all CPs under each scenario d, $d \in D$ = total cache hits and misses (2.2)

2. demand of CP i, $i \in I$ under scenario d, $d \in D \geq$ amount of bandwidth used by CP i, $i \in I$ (2.3)

3. demand of district j, $j \in J$ under scenario d, $d \in D$ = sum of cache hits and misses by CPs of PoP j, $j \in J$ (2.4)

4. bandwidth bought of PoP j, $j \in J \leq$ capacity of PoP j, $j \in J$ (2.5)

5. CDN capacity of PoP j, $j \in J \geq$ cache hits and misses of PoP j, $j \in J$ (2.6)

6. bandwidth unused of PoP j, $j \in J$ = bandwidth bought − used of PoP j, $j \in J$ (2.7)
7. cache hits of PoP j, $j \in J$ = sum of cache hits of district j, $j \in J$ (2.8)
8. cache misses of PoP j, $j \in J$ = sum of cache misses of district j, $j \in J$ (2.9)
9. cache-hit ratio of each CP must satisfy with the minimum cache-hit ratio requirement (2.10)
10. amount of bandwidth used by CP i, $i \in I$ = cache hits and misses by CP i, $i \in I$ (2.11)

Non-negativity of all decision variables is also required.

2.2 Cloud-based CDN

2.2.1 System Model

The architecture of a cloud-based CDN is considered as shown in Figure 2.2.

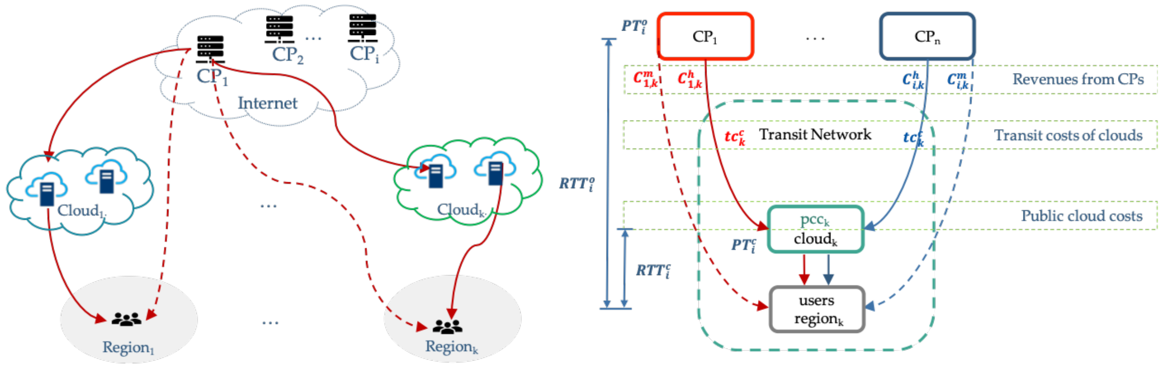


Figure 2.2. Cloud-based CDN Architecture

The praxis assumes that cloud-based CDN service is provided to a set of CPs, I, that can use a set of CDN's cloud-based cache servers on the public clouds, K, to deliver content to a set of regions, K. Each region has one public cloud administrative domain and the regional users can only be served by the cloud-based cache servers on the public cloud administrative domain in that region. A region covers several districts and a *cache-hit* is when the cloud cache contains the requested content and the request is served by the cache servers in the requesting region. A *cache-miss* is when the cloud cache does not contain the requested content and the request is passed along to the origin server. CPs are charged for delivering content from origin servers to users (cache miss) or from cloud-based cache servers to users (cache-hit) at different unit selling prices.

Bandwidth costs, the unit purchase costs of delivering content from cloud-based cache servers to users, depends on the regions of cloud-based cache servers reside. If the demand is less than the amount of bandwidth the service provider bought, the extra committed capacity will still need to be paid at a holding cost per unit. If there is more demand than the contracted capacity, the service provider needs not to pay a penalty or higher cost per unit for overage as long as the capacity of public cloud is still available. Unlike traditional CDN, cloud-based CDN offers a simple, pay-as-you-go pricing model without upfront fees or long-term-contract commitment since the decision to build certain cloud-based CDN capacity could be provisioned on public cloud in minutes. A typical configuration of a virtual cache server on public cloud is demonstrated as in Table 2.4.

Table 2.4. Typical Configuration of a Virtual Cache Server for Cloud-based CDN

CPU(vCPU)	RAM (GB)	System Disk (GB)	ESSD (GB)	Data Disk (GB)	Bandwidth (Mbps)	Costs (US$/Hour)
4	16	40	1000	4000	1000	35-40

Cloud-based CDN allows SPs to provision cache servers on the clouds quickly and provides the flexibility for CDN to scale up or down based on the demand from CPs. However, cloud-based CDN can hardly provide additional speed improvement in low-connectivity regions since the infrastructure resource is limited. Reducing RTT and edge-processing time could also be challenged for cloud-based CDN since both computing and connectivity resource on the public clouds will be limited and shared by many different users.

2.2.2 Assumptions Made

The following assumptions are made in this model for designing and managing a cloud-based CDN system.

1. A two-stage stochastic model is appropriately to capture the CDN's uncertainties.
2. The model aims to maximize the expected profit of the CDN.
3. The decision for the SP to order bandwidth is made in first stage before the realization of the demand is known in second stage.
4. The decision for the SP to establish certain capacity for the regional cloud capacity is based on historical customer demand requirement.
5. The included demand scenarios and their corresponding probabilities of occurrence appropriately represent the demand uncertainty.

2.2.3 Multi-stage Recourse Planning

Table 2.5 illustrates the decision-making process of the multi-stage resource allocation planing for the cloud-based CDN system.

Table 2.5. Multi-Stage Resource Allocation Planning for Cloud-base CDN

Multi-Stage Resource Allocation for Cloud-base CDN			
	Stage 1 (beginning of the month)	During the month	Stage 2 (end of the month)
Known at this time	-Selling price of CDN bandwidth -Costs and capacities for origin server -Costs and capacities for cloud-based cache servers -Costs and capacities for bandwidth -Possible demand scenarios -Possible distribution of users -Processing time for each server -Round-trip time for each route -Performance requirement	Demand scenario becomes known	-Actual demand of cloud-based resource -Total processing time for each CP -Total round-trip time for each CP -Cache-hit ratio for each CP
Unknown	-Actual demand of bandwidth for each CP -Total processing time for each CP -Total round-trip time for each CP -Cache-hit ratio for each CP		
Decisions needed	-How much resource for SP to order for each cloud		-How much bandwidth used for each CP in each region -How much resource used for each cloud

Decisions to order bandwidth and provision certain cloud-based cache servers were made in first stage by the SPs before the realization of the demand of the CPs is known in second stage.

2.2.4 Decision Variables

The decision variables used in this model is described in the following table:

$XC_{i,d}$:	Gs of bandwidth used for CP i, $i \in I$, under scenario d, $d \in D$
CBT_k :	Gs of bandwidth the cloud-based CDN should purchase on cloud k by SP, $k \in K$
$CC^h_{k,d}$:	Gs of cache-hit on cloud k, $k \in K$, under scenario d, $d \in D$
$CC^m_{k,d}$:	Gs of cache-miss on cloud k, $k \in K$, under scenario d, $d \in D$
$CC^n_{k,d}$:	Gs of bandwidth unused on cloud k, $k \in K$, under scenario d, $d \in D$
$RC^h_{i,k,d}$:	Gs of cache-hit for CP i, $i \in I$, on cloud k, $k \in K$, under scenario d, $d \in D$
$RC^m_{i,k,d}$:	Gs of cache-miss for CP i, $i \in I$, on cloud k, $k \in K$, under scenario d, $d \in D$

2.2.5 Mathematical Formulation and Explanation

Maximize Expected CDN Profit

$$\begin{aligned}\sum_{i\in I,k\in K,d\in D} RC^h_{i,k,d} * C^h_{i,k} * S^p_d + \sum_{i\in I,k\in K,d\in D} RC^m_{i,k,d} * C^m_{i,k} * S^p_d - \sum_{k\in K} CBT_k * tc^c_k \\ - \sum_{k\in K,d\in D} CC^h_{k,d} * cc^c_k * S^p_d - \sum_{k\in K,d\in D} CC^n_{k,d} * hc^c_k * S^p_d\end{aligned} \tag{2.13}$$

subject to:

$$\begin{aligned}
cdemand(d): \sum_{i\in I} LD_{i,d} &= \sum_{k\in K} CC^h_{k,d} + \sum_{k\in K} CC^m_{k,d} && (2.14)\\
cudeman(i,d): LD_{i,d} &\geq XC_{i,d} && (2.15)\\
rdemand(k,d): \sum_{i\in I} LD_{i,d} * \sum_{j\in J} pud_{i,j} * cud_{j,k} &= \sum_{i\in I} RC^h_{i,k,d} + \sum_{i\in I} RC^m_{i,k,d} && (2.16)\\
cacb(k): CBT_k &\leq CCA_k && (2.17)\\
cacu(k,d): CCA_k &\geq CC^h_{k,d} + CC^m_{k,d} && (2.18)\\
cinv(k,d): CC^n_{k,d} &= CBT_k - CC^h_{k,d} && (2.19)\\
tcch(k,d): CC^h_{k,d} &= \sum_{i\in I} RC^h_{i,k,d} && (2.20)\\
tccm(k,d): CC^m_{k,d} &= \sum_{i\in I} RC^m_{i,k,d} && (2.21)\\
hitrc(i,d): \sum_{k\in K} RC^h_{i,k,d} &\geq minhit * XC_{i,d} && (2.22)\\
cbght(i,d): XC_{i,d} &= \sum_{k\in K} RC^h_{i,k,d} + \sum_{k\in K} RC^m_{i,k,d} && (2.23)
\end{aligned}$$

$$Nonnegativity: XC, CBT, CC^h, CC^m, CC^n, RC^h, RC^m \geq 0 \qquad (2.24)$$

The objective function simply maximize the expected profit and the following constraints should be followed:

1. total demand by all CPs under each scenario d, $d \in D$ = total cache hits and misses (2.14)

2. demand of CP i, $i \in I$ under scenario d, $d \in D \geq$ amount of bandwidth used by CP i, $i \in I$ (2.15)

3. demand of region k, $k \in K$ under scenario d, $d \in D$ = sum of cache hits and misses by CPs of cloud k, $k \in K$ (2.16)

4. cloud-based bandwidth bought by region k, $k \in K \leq$ CDN capacity of public cloud in region k, $k \in K$ (2.17)

5. CDN capacity of public cloud in region k, $k \in K \geq$ cloud-based cache hits and misses of region k, $k \in K$ (2.18)

6. cloud bandwidth unused of region k, $k \in K$ = bandwidth bought − used of region k, $k \in K$ (2.19)

7. cache hits of public cloud in region k, $k \in K$ = sum of cache hits of region k, $k \in K$ (2.20)

8. cache misses of public cloud in region k, $k \in K$ = sum of cache misses of region k, $k \in K$ (2.21)

9. cache-hit ratio of each CP must satisfy with the minimum cache-hit ratio requirement (2.22)

10. amount of bandwidth on public cloud used by CP i, $i \in I$ = cloud-based cache hits and misses by CP i, $i \in I$ (2.23)

Non-negativity of all decision variables is also required.

2.3 Cloud-based CDN with Edge Computing

2.3.1 System Model

The architecture of a cloud-based CDN with edge computing is considered as shown in Figure 2.3.

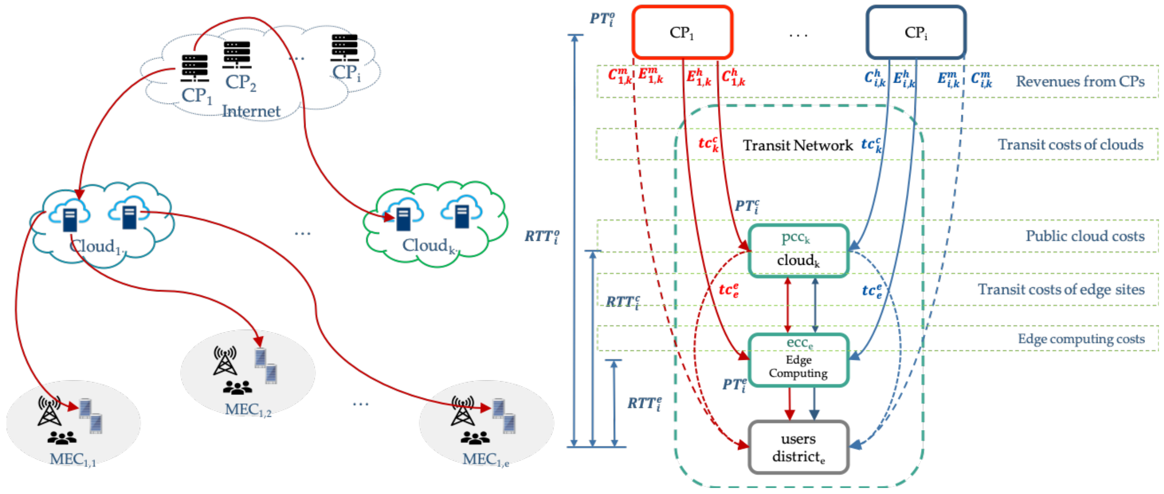


Figure 2.3. Cloud-based CDN with Edge Computing Architecture

The praxis assumes that cloud-based CDN with edge computing service is provided to a set of CPs, I, that can use a set of cloud-based CDN's cache servers, K, and a set of CDN's cache servers at the edge computing sites, E, to deliver content to a set of districts, E. Each region has one public cloud administrative domain to cover several districts and each district has an edge computing site. The regional users can be served by the cloud-based cache servers of that region or by the cache servers at edge computing sites covered by that region. A *cache-hit* is when the cloud-based or related edge cache servers contains the requested content and the request is served by the the cache servers in the requesting region. A *cache-miss* is when the cache servers do not contain the requested content and the request is passed along to the origin server. CPs are charged for delivering content from origin servers to users (cache miss) or from cache servers to users (cache-hit) at different unit selling prices.

Bandwidth costs, the unit purchase costs of delivering content from cloud-based cache servers or edge computing sites to users, depends on the regions or edge computing sites of cache servers reside. If the demand is less than the amount of bandwidth the service provider bought, the extra committed capacity will still need to be paid at a holding cost per unit. If there is more demand than the contracted capacity, the service provider needs not to pay a penalty or higher cost per unit for overage as long as the capacity of public cloud or edge computing sites is still available. Like cloud-based CDN, cloud-based CDN with edge computing also offers a simple, pay-as-you-go pricing model without upfront fees or long-term-contract commitment since the decision to build certain CDN capacity could be provisioned in minutes.

Cloud-based edge computing sites offer the advantage of not having to duplicate the processes already running on the host system and allow for more efficient distribution of the limited resources available on cache servers. Therefore this praxis assumes that the cloud-based CDN with edge computing system could obtain better network performance such as round-trip time.

2.3.2 Assumptions Made

The following assumptions are made in this model for designing and managing a cloud-based CDN with edge computing system.

1. A two-stage stochastic model is appropriately to capture the CDN's uncertainties.
2. The model aims to maximize the expected profit of the CDN.
3. The decision for the SP to order cloud-based and edge-based bandwidth is made in first stage before the realization of the demand is known in second stage.
4. The decision for the SP to establish certain capacity for the edge computing sites is based on historical customer demand requirement.
5. The included demand scenarios and their corresponding probabilities of occurrence appropriately represent the demand uncertainty.

2.3.3 Multi-stage Recourse Planning

Table 2.6 illustrates the decision-making process of the multi-stage resource allocation planing for the cloud-based CDN with edge computing system.

Table 2.6. Multi-stage Resource Allocation for Cloud-based CDN with Edge Computing

Multi-Stage Resource Allocation for Cloud-base CDN with Edge Computing			
	Stage 1 (beginning of the month)	During the month	Stage 2 (end of the month)
Known at this time	-Selling price of CDN bandwidth -Costs and capacities for origin server -Costs and capacities for cloud-based cache servers -Costs and capacities for edge cache servers -Costs and capacities for bandwidth -Possible demand scenarios -Possible distribution of users -Processing time for each server -Round-trip time for each route -Performance requirement by each CP	Demand scenario becomes known	-Actual demand of bandwidth -Actual demand of cloud-based resource -Actual demand of edge-based resource -Total processing time for each CP -Total round-trip time for each CP -Cache-hit ratio for each CP
Unknown	-Actual demand of bandwidth -Actual demand of cloud-based resource in each region -Actual demand of edge computing resource in each district -Total processing time for each CP -Total round-trip time for each CP -Cache-hit ratio for each CP		
Decisions needed	-How much resource for SP to order for each cloud -How much resource for SP to order for each edge site		-How much bandwidth used for each region/district by each CP -How much cloud-based resource used for each region -How much edge computing resource used for each district

Decisions to order bandwidth and establish certain capacity on public cloud and at edge computing sites were made in first stage by the SPs before the realization of the demand of the CPs is known in second stage.

2.3.4 Decision Variables

The decision variables used in this model is described in the following table:

$XE^{c}_{i,d}$:	Gs of bandwidth used by cloud-based cache for CP i, $i \in I$, under scenario d, $d \in D$
$XE^{e}_{i,d}$:	Gs of bandwidth used by edge computing cache for CP i, $i \in I$, under scenario d, $d \in D$
EBC_k :	Gs of bandwidth the cache servers should purchase on cloud k by SP, $k \in K$
EBT_e :	Gs of bandwidth the cache servers should purchase at edge computing site e by SP, $e \in E$
$EC^{h}_{k,d}$:	Gs of cache-hit on cloud k, $k \in K$, under scenario d, $d \in D$
$EC^{m}_{k,d}$:	Gs of cache-miss on cloud k, $k \in K$, under scenario d, $d \in D$
$EC^{n}_{k,d}$:	Gs of bandwidth unused on cloud k, $k \in K$, under scenario d, $d \in D$
$EE^{h}_{e,d}$:	Gs of cache-hit at edge computing site e, $e \in E$, under scenario d, $d \in D$
$EE^{m}_{e,d}$:	Gs of cache-miss at edge computing site e, $e \in E$, under scenario d, $d \in D$
$EE^{n}_{e,d}$:	Gs of bandwidth unused at edge computing site e, $e \in E$, under scenario d, $d \in D$
$RE^{h}_{i,k,d}$:	Gs of cache-hit for CP i, $i \in I$, on cloud k, $k \in K$, under scenario d, $d \in D$
$RE^{m}_{i,k,d}$:	Gs of cache-miss for CP i, $i \in I$, on cloud k, $k \in K$, under scenario d, $d \in D$
$DE^{h}_{i,e,d}$:	Gs of cache-hit for CP i, $i \in I$, at edge computing site e, $e \in E$, under scenario d, $d \in D$
$DE^{m}_{i,e,d}$:	Gs of cache-hit for CP i, $i \in I$, at edge computing site e, $e \in E$, under scenario d, $d \in D$

2.3.5 Mathematical Formulation and Explanation

Maximize Expected CDN Profit

$$\sum_{i\in I,k\in K,d\in D} RE^{h}_{i,k,d} * C^{h}_{i,k} * S^{p}_{d} + \sum_{i\in I,k\in K,d\in D} RE^{m}_{i,k,d} * C^{m}_{i,k} * S^{p}_{d} + \sum_{i\in I,e\in E,d\in D} DE^{h}_{i,e,d} * E^{h}_{i,e} * S^{p}_{d} + \sum_{i\in I,e\in E,d\in D} DE^{m}_{i,e,d} * E^{m}_{i,e} * S^{p}_{d} - \sum_{k\in K} EBC_k * tc^{c}_{k} - \sum_{e\in E} EBT_e * tc^{e}_{e} - \sum_{k\in K,d\in D} EC^{h}_{k,d} * cc^{c}_{k} * S^{p}_{d} - \sum_{e\in E,d\in D} EE^{h}_{e,d} * ec_e * S^{p}_{d} - \sum_{k\in K,d\in D} EC^{n}_{k,d} * hc^{c}_{k} * S^{p}_{d} - \sum_{e\in E,d\in D} EE^{n}_{e,d} * hc^{e}_{e} * S^{p}_{d} \tag{2.25}$$

subject to:

$$etde(d) : \sum_{i \in I} LD_{i,d} = \sum_{k \in K} EC^h_{k,d} + \sum_{k \in K} EC^m_{k,d} + \sum_{e \in E} EE^h_{e,d} + \sum_{k \in K} EE^m_{e,d} \quad (2.26)$$

$$eude(i,d) : LD_{i,d} \geq XE^c_{i,d} + XE^e_{i,d} \quad (2.27)$$

$$erde(k,d) : \sum_{i \in I} LD_{i,d} * \sum_{j \in J} pud_{i,j} * cud_{j,k} = \sum_{i \in I} (RE^h_{i,k,d} + RE^m_{i,k,d} + \sum_{e \in E} (DE^h_{i,e,d} + DE^m_{i,e,d}) * eur_{e,k}) \quad (2.28)$$

$$ebcb(k) : EBC_k \leq CCA_k \quad (2.29)$$

$$ebtb(e) : EBT_e \leq CCA_e \quad (2.30)$$

$$ebcu(k,d) : CCA_k \geq EC^h_{k,d} + EC^m_{k,d} \quad (2.31)$$

$$ebtu(e,d) : CCA_e \geq EE^h_{e,d} + EE^m_{e,d} \quad (2.32)$$

$$ecin(k,d) : EC^n_{k,d} = EBC_k - EC^h_{k,d} \quad (2.33)$$

$$eein(e,d) : EE^n_{e,d} = EBT_e - EE^h_{e,d} \quad (2.34)$$

$$tceh(k,d) : EC^h_{k,d} = \sum_{i \in I} RE^h_{i,k,d} \quad (2.35)$$

$$tcem(k,d) : EC^m_{k,d} = \sum_{i \in I} RE^m_{i,k,d} \quad (2.36)$$

$$teeh(e,d) : EE^h_{e,d} = \sum_{i \in I} DE^h_{i,e,d} \quad (2.37)$$

$$teem(e,d) : EE^m_{e,d} = \sum_{i \in I} DE^m_{i,e,d} \quad (2.38)$$

$$hitre(i,d) : \sum_{k \in K} RE^h_{i,k,d} + \sum_{e \in E} DE^h_{i,e,d} \geq minhit * (XE^c_{i,d} + XE^e_{i,d}) \quad (2.39)$$

$$rttre(i,d) : XE^c_{i,d} * RTT^c_i + XE^e_{i,d} * RTT^e_i \leq maxrtt * (XE^c_{i,d} + XE^e_{i,d}) \quad (2.40)$$

$$ecbgt(i,d) : XE^c_{i,d} = \sum_{k \in K} RE^h_{i,k,d} + \sum_{k \in K} RE^m_{i,k,d} \quad (2.41)$$

$$eebgt(i,d) : XE^e_{i,d} = \sum_{e \in E} DE^h_{i,e,d} + \sum_{k \in K} DE^m_{i,e,d} \quad (2.42)$$

$$Nonnegativity : XE^c, XE^e, EBC, EBT, EC^h, EC^m, EC^n, EE^h, EE^m, EE^n, RE^h, RE^m, DE^h, DE^m \geq 0 \quad (2.43)$$

The objective function simply maximize the expected profit and the following constraints should be followed:

1. total demand by all CPs under each scenario d, $d \in D$ = total cache hits and misses (2.26)
2. demand of CP i, $i \in I$ under scenario d, $d \in D \geq$ amount of bandwidth used by CP i, $i \in I$ (2.27)
3. demand of region k, $k \in K$ under scenario d, $d \in D$ = sum of cache hits and misses by CPs of cloud k, $k \in K$ (2.28)
4. cloud-based bandwidth bought by region k, $k \in K \leq$ CDN capacity of public cloud in region k, $k \in K$ (2.29)
5. edge computing bandwidth bought by district e, $e \in E \leq$ capacity of edge computing site e, $e \in E$ (2.30)
6. CDN capacity of public cloud in region k, $k \in K \geq$ cloud-based cache hits and misses of region k, $k \in K$ (2.31)
7. CDN capacity of edge computing site e, $e \in E \geq$ edge-based cache hits and misses of district e, $e \in E$ (2.32)
8. cloud bandwidth unused of region k, $k \in K$ = bandwidth bought $-$ used of region k, $k \in K$ (2.33)
9. bandwidth unused of edge computing site e, $e \in E$ = bandwidth bought $-$ used of district e, $e \in E$ (2.34)
10. cache hits of public cloud in region k, $k \in K$ = sum of cache hits of region k, $k \in K$ (2.35)
11. cache misses of public cloud in region k, $k \in K$ = sum of cache misses of region k, $k \in K$ (2.36)

12. cache hits of edge computing site e, $e \in E$ = sum of cache hits of district e, $e \in E$ (2.37)

13. cache misses of edge computing site e, $e \in E$ = sum of cache misses of district e, $e \in E$ (2.38)

14. cache-hit ratio of each CP must satisfy with the minimum cache-hit ratio requirement (2.39)

15. round-trip time of each CP must meet with the maximum round-trip time requirement (2.40)

16. amount of bandwidth on public cloud used by CP i, $i \in I$ = cloud-based cache hits and misses by CP i, $i \in I$ (2.41)

17. amount of bandwidth at edge sites used by CP i, $i \in I$ = edge-based cache hits and misses by CP i, $i \in I$ (2.42)

Non-negativity of all decision variables is also required.

Chapter 3

Experimental Design

3.1 The Experiment

The goal of the experiment is to maximize the profit for the CDN service providers through resource allocation and meet with the network performance requirement of the content providers who use the related CDN services.

In previous sections, the different CDN network architectures have been discussed and related system models for maximizing the profit through resource allocation were presented. This section continues with details of a series of statistical experiments designed to help service providers determine which of the CDN network architectures and related system models should be used.

Service providers usually setup CDN networks based on customers' contractual demand in the absence of comprehensive research that studies CDN infrastructure costs and network latencies under various scenarios. This praxis addresses this shortcoming through a rigorous statistical comparison of optimally engineered CDN infrastructure and resource allocation to design the best CDN network architecture or systems under a variety of situations and assumptions commonly found in practice.

The following problems were addressed in this praxis.

- How to maximize profit and cover all the likely customer demand for different scenarios?
- Which CDN network architecture or system should be deployed: traditional CDN, cloud-based CDN or cloud-based CDN with edge computing? The dependent variable is the expected profit gained from an optimally designed CDN network which is the primary decision-making metric for resource allocation and performance optimization.

3.1.1 Response Variables and Performance Evaluation Criteria

The experimental response, the expected profit gained by service providers would be reported in 1,000 US dollars, rounded to the nearest 1000^{th}. The expected profit depends on the revenue of the CDN services from CPs and costs for the SP to provide CDN services. The praxis assumes that the CPs are charged for delivering content from origin servers to users (cache miss) or from cache servers to users (cache-hit) at different unit selling prices. Bandwidth costs, the unit purchase costs of delivering content from cache servers or edge computing sites to users, depends on where the cache servers or edge computing sites reside. If the demand is less than the amount of bandwidth the service provider bought, the extra committed capacity will still need to be paid at a holding cost per unit. If there is more demand than the contracted capacity, the service provider needs not to pay a penalty or higher cost per unit for overage as long as the CDN capacity of public cloud or edge computing sites is still available.

3.1.2 Factors to be Explored

Many factors can affect the profit of a CDN network, including the service selling prices to the CPs, number and capacity of PoP cache servers, number and capacity of cloud cache servers, number and capacity of edge cache servers, network topology, capital expense (CAPEX) of the infrastructure, operational expense (OPEX) of the CDN network, number and size of the demands to be carried, number and size of the inventories to be held, performance requirement by the customers, the total amount of traffic on the network and capacities of each element.

The experiment assumes an existing network to cover 20 districts (roughly the size of a national network currently implemented by a service provider in China) carrying a total of 20 Terabits (Tb) per second of traffic, the bandwidth is dependent upon the infrastructure CAPEX invested for every district or region, and the infrastructure CAPEX is a linear function of its associated bandwidth. Customer demands assumed to be the different size for 3 scenarios such as idle-hours, normal-hours and busy-hours. Factors of interest, uncontrollable factors and nuisance factors are studied and several key factors have been selected for the experiment.

3.1.2.1 Factors of Interest

Most factors of interest are backed by industry research or empirical findings. The experiment also considered some factors by opinions of logical conclusions. The experiment was designed to focus on factors that are SP-controllable factors. In this experiment, the factors to be varied are: total customer demand to be carried, number of busy-hours, CDN system, number and capacity of cloud cache servers, number and capacity of edge cache servers, RTT requirement by the customers and cache-hit ratio requirement. The levels for these factors are shown in Table 3.1, and are described in detail in the sections to follow.

Table 3.1. Factors of Interest

Factors	Low-level			Neutral-level			High-level			Experimented
Total Customer Demand (G)	Idle	Normal	Busy	Idle	Normal	Busy	Idle	Normal	Busy	Yes
	2500	3500	4500	3000	4000	5000	3500	4500	5500	
Busy-hour Duration (%)	7.5%			10%			15%			Yes
Price of cache-hit to CPs	$2.5k/G			$3.0k/G			$3.5k/G			No
Price of Cache Miss to CPs	$2.5k/G			$3.0k/G			$3.5k/G			No
Number of PoP Cache Nodes	20			20			20			No
Capacity of PoP Cache (G)	5600			6000			7000			Yes
Number of Cloud Domains	4			4			4			No
Capacity of Cloud Cache Nodes (G)	5600			6000			7000			Yes
Number of Edge Computing Nodes	20			20			20			No
Capacity of Edge Computing Cache (G)	2500			3000			3500			Yes
CAPEX for Traditional CDN	$0.25k/G			$0.3k/G			$0.35k/G			No
CAPEX for Cloud-based CDN	$0.2k/G			$0.25k/G			$1k/G			No
CAPEX for Edge Computing Site	$0.07k/G			$0.1k/G			$0.15k/G			No
OPEX for Traditional CDN	$0.5k/G			$0.8k/G			$1.1k/G			No
OPEX for Cloud-based CDN	$0.5k/G			$0.8k/G			$1.1k/G			No
OPEX for Edge Computing Site	$0.2k/G			$0.4k/G			$0.8k/G			No
Number & Size of Inventories	10%			20%			30%			No
RTT Requirement by CPs	60ms			50ms			40ms			No
cache-hit Ratio	90%			95%			98%			Yes

- **Total Customer Demand** is very important to the profit gained from CDN. The general consensus is the CDN marginal revenue increases faster than its marginal cost of CDN bandwidth production giving the high upfront investment. The more total customer demand, the more profit gained.

- **Busy-hours Duration** is likely to affect the profit gained from CDN since burstable billing measures CDN bandwidth usage based on peak use. SPs are allowed bandwidth usage to exceed a specified threshold for brief periods of time without the financial penalty of purchasing a higher committed information rate (CIR, or commitment). The higher level of capacity required nowadays is expected to assist in better cache-hit ratio performance which theoretically leads to worse profitability.

- **Price of cache-hit to CPs** is the price charged to CPs while their site's content is successfully served from the cache and one of the main components for CDN revenue. It is difficult to vary by SPs since the market is competitive and transparent. Therefore it was excluded as a factor in the experiment.

- **Price of Cache Miss to CPs** is the price charged to CPs while their site's content is not successfully served from the cache and one of the components for CDN revenue. It is difficult to vary by SPs since the market is competitive and transparent. Therefore it was excluded as a factor in the experiment.

- **Number of PoP Cache Nodes** cannot be varied easily since it has been carefully planned and invested to cover 20 districts carrying a total of 20 Terabits per second of traffic for the peak time. Therefore it was excluded as a factor in the experiment.

- **Capacity of PoP Cache Nodes** is mainly related to the upfront investment for the traditional CDN.

- **Number of Cloud Domains** might not be varied easily due the management issues to interconnect among multiple clouds.

- **Capacity of Cloud Cache Nodes** could be more flexible adjusted and invested than the traditional CDN by leverage the incumbent capacity of public cloud's infrastructure. The capacity of cloud cache servers are much easier to scale to satisfy the potential demand, particular the demand of busy-hour duration, on the cloud-based CDN.

- **Number Edge Computing Sites** might not be varied easily since it has been carefully planned and invested to establish the base capacity on the edge.

- **Capacity of Edge Computing Nodes** is helpful to improve the CDN performance for the CPs. The more edge computing nodes, the better round-trip time could be obtained for CDN services. This could also improve the profitability for the SPs since the cost is controllable by the SPs as a linear function of the associated bandwidth used.

- **CAPEX of Traditional CDN** is the upfront CAPEX involved to build the cache capacity for traditional CDN. However, it is difficult to vary by the SP and therefore it was excluded as a factor in the experiment.

- **CAPEX of Cloud-based CDN** is a linear function of its associated bandwidth giving the requirement on bandwidth is less than the cloud capacity available. Unlike traditional CDN, there no huge upfront CAPEX involved as the infrastructure investment is covered by the public cloud. However, it is difficult to vary by the SP and therefore it was excluded as a factor in the experiment.

- **CAPEX of Edge Computing Site** is difficult to control form the SPs' perspective but controlled by the telecom carriers, and difficult to vary in an experiment. Therefore it was excluded as a factor in the experiment.

- **OPEX of Traditional CDN** is mainly the bandwidth costs charged by the telecom carriers to the SPs which is difficult to vary in an experiment. It is generally considered that SPs could obtain bandwidth from the telecom carriers with wholesale price and therefore it was excluded as a factor in the experiment.

- **OPEX of Cloud-based CDN** is also mainly the bandwidth costs charged by the telecom carriers to the SPs which is difficult to vary in an experiment. Therefore it was excluded as a factor in the experiment.

- **OPEX of Edge Computing Site** is mainly the bandwidth costs charged by the telecom carriers to the SPs which is difficult to vary in an experiment. Therefore it was excluded as a factor in the experiment.

- **RTT Requirement** is very important for CDN performance and it is an important metric in determining the health of a connection to diagnose the speed and reliability of CDN network. Reducing RTT is a primary goal of a CDN and controllable by SPs by investment on the edge sites. However, it is not easy change in level and therefore it was excluded as a factor in the experiment.

- **Cache-hit Ratio** is a measurement of how many content requests a cache is able to fill successfully, compared to how many requests it receives. A high-performing CDN will have a high cache-hit ratio and this is likely the affect the profit gained from CDN services.

3.1.2.2 Uncontrollable Factors

There are many uncontrollable factors since this experiment could not be conducted in a laboratory environment:

- Network congestion - reduce by increasing the capacity of network nodes or links which is carrying data in the experiment

- Seasonal peaks and troughs – reduce by using monthly or quarterly average and combined traffic models of CPs

- Promotion strategies of clouds – reduce by using monthly, quarterly or annually average data of public clouds

- Inaccuracy of district demand distribution - reduce by monthly data monitoring in practice

- Customer expectation of network performance - reduce by industry standards and guidelines

3.1.2.3 Nuisance Factors

Some factors may influence the experimental response, but they might not be directly interested in this experiment.

- Computer performance (the amount of useful work accomplished by a computer system)
- Network jitter (the variance in time delay in milliseconds (ms) between data packets over a network)
- High availability (ensure an agreed level of operational performance, usually uptime, for a higher than normal period)

3.1.3 The Hypotheses to be Investigated

The study's goal of comparing recovery techniques under combinations of experimental factors is achieved by gathering evidence and statistically analyzing the results to determine the truth or falsity of the following hypotheses. The hypotheses investigated are:

1. H_0 #1: The profit gained is the same for all CDN network architectures.
2. H_0 #2: The performance obtained is the same for all CDN network architectures.
3. H_0 #3: The profit gained is the same for all network capacity.
4. H_0 #4: The profit gained is the same for all total demand.
5. H_0 #5: The profit gained is the same for all busy-hour duration.
6. H_0 #6: The profit gained is the same for all performance requirement.
7. H_0 #7: The profit gained is the same for all network capacity with same total demand.
8. H_0 #8: The profit gained is the same for all network capacity with same busy-hour duration.

9. H_0 #9: The profit gained is the same for all network capacity with same performance requirement.

10. H_0 #10: The profit gained is the same for all total demand and busy-hour duration.

11. H_0 #11: The profit gained is the same for all total demand and performance requirement.

12. H_0 #12: The profit gained is the same for all busy-hour duration and performance requirement.

13. H_0 #13: The profit gained is the same for all network capacity with same total demand and busy-hour duration.

14. H_0 #14: The profit gained is the same for all network capacity with same total demand and performance requirement.

15. H_0 #15: The profit gained is the same for all network capacity with same busy-hour duration and performance requirement.

16. H_0 #16: The profit gained is the same for all total demand with same busy-hour duration and performance requirement.

Rigorously analyzing these postulations should uncover key factors in resource allocation. The next section describes the evidence that is used and how it was obtained for analysis.

3.2 The Design

This section describes the experimental setup for collecting data that can be used to allocate resource and optimize performance for traditional CDN, cloud-based CDN and cloud-based CDN with edge computing systems.

3.2.1 Evidence for the Analysis

The experiment would be executed in real-world conditions using the blocking aspects of CDN systems to gain more realistic results. The praxis used a full factorial design to execute an experiment on traditional CDN, cloud-based CDN and cloud-based CDN with edge computing.

The overall experiment was conceived with the expectation that some of the interesting factors could be screened out. While some of the factors could be considered in a 2-factor factorial design, some did not lend themselves to easy changes in level. For example, the selection of number and capacity of cloud cache or edge computing sites could be tedious and time-consuming.

The evidence-gathering process involves the following steps:

1. Generating CPs' demand sets
2. Setting up traditional CDN
3. Setting up cloud-based CDN
4. Setting up cloud-based CDN with edge computing
5. Setting up performance metrics
6. Generating input files for mathematical language program and optimizer
7. Optimizing resource allocation and performance

3.2.1.1 Generating Demand

Customer demand scenarios for ten CPs in the experiment and their corresponding probabilities of occurrence are demonstrated in Table 3.2.

Table 3.2. Customer Demand Scenarios for CDN Services

Factors	Low-level			Neutral-level			High-level		
Busy-hour Duration (%)	7.5%			10%			15%		
Normal-hour Duration (%)	67.5%			65%			60%		
Idle-hour Duration (%)	25%			25%			25%		
Total Customer Demand (G)	Idle	Normal	Busy	Idle	Normal	Busy	Idle	Normal	Busy
	2500	3500	4500	3000	4000	5000	3500	4500	5500
CP1	50	150	250	100	200	300	150	250	350
CP2	50	150	250	100	200	300	150	250	350
CP3	150	250	350	200	300	400	250	350	450
CP4	150	250	350	200	300	400	250	350	450
CP5	250	350	450	300	400	500	350	450	550
CP6	250	350	450	300	400	500	350	450	550
CP7	350	450	550	400	500	600	450	550	650
CP8	350	450	550	400	500	600	450	550	650
CP9	450	550	650	500	600	700	550	650	750
CP10	450	550	650	500	600	700	550	650	750

The likely demand distribution of CPs for 20 districts is demonstrated in Table 3.3.

Table 3.3. Likely Demand of Each CP in Every District

Scenarios	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16	D17	D18	D19	D20
CP1	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%
CP2	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%
CP3	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%
CP4	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%
CP5	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%
CP6	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%
CP7	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%
CP8	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%
CP9	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%
CP10	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%	5%

3.2.1.2 Setting up Traditional CDN

Traditional CDN is set up in the experiment as demonstrated in Table 3.4.

Table 3.4. Capacity (G) & Related Costs (k$/G) of Traditional CDN

Traditional CDN	Capacity (G)			Bandwidth Cost (k$/G)	Penalty Cost (k$/G)	Holding Cost (k$/G)	Storage Cost (k$/G)
	L	N	H				
PoP1	280	300	350	0.8	0.3	0.2	0.3
PoP2	280	300	350	0.8	0.3	0.2	0.3
PoP3	280	300	350	0.8	0.3	0.2	0.3
PoP4	280	300	350	0.8	0.3	0.2	0.3
PoP5	280	300	350	0.8	0.3	0.2	0.3
PoP6	280	300	350	0.8	0.3	0.2	0.3
PoP7	280	300	350	0.8	0.3	0.2	0.3
PoP8	280	300	350	0.8	0.3	0.2	0.3
PoP9	280	300	350	0.8	0.3	0.2	0.3
PoP10	280	300	350	0.8	0.3	0.2	0.3
PoP11	280	300	350	0.8	0.3	0.2	0.3
PoP12	280	300	350	0.8	0.3	0.2	0.3
PoP13	280	300	350	0.8	0.3	0.2	0.3
PoP14	280	300	350	0.8	0.3	0.2	0.3
PoP15	280	300	350	0.8	0.3	0.2	0.3
PoP16	280	300	350	0.8	0.3	0.2	0.3
PoP17	280	300	350	0.8	0.3	0.2	0.3
PoP18	280	300	350	0.8	0.3	0.2	0.3
PoP19	280	300	350	0.8	0.3	0.2	0.3
PoP20	280	300	350	0.8	0.3	0.2	0.3

3.2.1.3 Setting up Cloud-based CDN

Cloud-based CDN is set up in the experiment as demonstrated in Table 3.5.

Table 3.5. Capacity (G) & Related Costs (k$/G) of Cloud-based CDN

Cloud-based CDN	Capacity (G)			Bandwidth Cost (k$/G)	Holding Cost (k$/G)	Virtual Server Cost (k$/G)
	L	N	H			
Cloud1	1400	1500	1750	0.8	0.1	0.25
Cloud2	1400	1500	1750	0.8	0.1	0.25
Cloud3	1400	1500	1750	0.8	0.1	0.25
Cloud4	1400	1500	1750	0.8	0.1	0.25

3.2.1.4 Setting up Edge Computing Sites

Edge computing sites are set up in the experiment ans demonstrated in Table 3.6.

Table 3.6. Capacity (G) & Related Costs (k$/G) of Edge Computing Sites

Cloud	Capacity (G)	Edge Computing	Capacity (G)			Bandwidth Cost (k$/G)	Holding Cost (k$/G)	Edge Site Cost (k$/G)
			L	N	H			
Cloud 1	1500	EC Site1	125	150	175	0.4	0.1	0.1
		EC Site2	125	150	175	0.4	0.1	0.1
		EC Site3	125	150	175	0.4	0.1	0.1
		EC Site4	125	150	175	0.4	0.1	0.1
		EC Site5	125	150	175	0.4	0.1	0.1
Cloud 2	1500	EC Site6	125	150	175	0.4	0.1	0.1
		EC Site7	125	150	175	0.4	0.1	0.1
		EC Site8	125	150	175	0.4	0.1	0.1
		EC Site9	125	150	175	0.4	0.1	0.1
		EC Site10	125	150	175	0.4	0.1	0.1
Cloud 3	1500	EC Site11	125	150	175	0.4	0.1	0.1
		EC Site12	125	150	175	0.4	0.1	0.1
		EC Site13	125	150	175	0.4	0.1	0.1
		EC Site14	125	150	175	0.4	0.1	0.1
		EC Site15	125	150	175	0.4	0.1	0.1
Cloud 4	1500	EC Site16	125	150	175	0.4	0.1	0.1
		EC Site17	125	150	175	0.4	0.1	0.1
		EC Site18	125	150	175	0.4	0.1	0.1
		EC Site19	125	150	175	0.4	0.1	0.1
		EC Site20	125	150	175	0.4	0.1	0.1

3.2.1.5 Setting up Performance Metrics

Round-trip Time (RTT) from different sites to consumers is demonstrated in Table 3.7.

Table 3.7. Round-trip Time (RTT) for CPs

Round-trip Time (ms)	Origin Server	Cache Server/PoP	Cloud Cache	Edge Computing Site
CP1	200	30	60	10
CP2	200	30	60	10
CP3	200	30	60	10
CP4	200	30	60	10
CP5	200	30	60	10
CP6	200	30	60	10
CP7	200	30	60	10
CP8	200	30	60	10
CP9	200	30	60	10
CP10	200	30	60	10

Edge-processing Time (PT) for CDN services used cache on different sites is demonstrated in Table 3.8.

Table 3.8. Edge-processing Time (PT) for CPs

Processing Time (ms)	Origin Server	Cache Server/PoP	Cloud Cache	Edge Computing Site
CP1	100	20	25	10
CP2	100	20	25	10
CP3	100	20	25	10
CP4	100	20	25	10
CP5	100	20	25	10
CP6	100	20	25	10
CP7	100	20	25	10
CP8	100	20	25	10
CP9	100	20	25	10
CP10	100	20	25	10

3.2.2 Software and Computing Environment

- **GAMS v28.2.0** is the modeling system used in this experiment for mathematical programming and optimization. GAMS/CONOPT is the solver used to find the profit and performance optimization solution to each problem.
- **Design Expert v11.1.0.1 64-bit** by Stat-Ease is used to lay out the experiment with factors of interest to perform the statistical analysis. As a statistical software specifically dedicated to performing design of experiments (DOE), statistical significance of the factors is established with analysis of variance (ANOVA).
- **A MacBook Air** equipped with a 1.6GHz dual-core Intel Core i5 (Turbo Boost up to 3.6GHz, with 4MB L3 cache), 256GB PCIe-based SSD, and 16GB of 2133MHz LPDDR3 on-board memory is used in this experiment as the main computing environment.

3.2.3 Number of Observations

The goal was to execute the single replicate full factorial design for three CDN systems (blocks) and the following factors and response selected for this experiment. The following factors and response selected for this experiment.

- Factors: A = capacity, B = total demand, C = busy-hour duration, D = cache-hit ratio
- Aliases: A = A + BCD, B = B+ ACD, C = C+ABD, D = D+ABC, AB = AB+CD, AC = AC+BD, AD = AD+BC
- Defining Contrast: I = ABCD
- Generators: D=ABC
- Response: Profit gained from CDN
- Signal to noise and power: default

Three demand scenarios for ten CPs of 20 districts are tested on traditional CDN, cloud-based CDN and cloud-based CDN with edge computing systems for different factors and level combination. There is a total 243 test instances evaluated to reach this praxis' conclusions. Table 3.9 demonstrates the number of observations per factor and level combination).

Table 3.9. Number of Tests in Experiment

System	Nodes	Capacity(G)	Total Demand (G)			Busy-Hour (%)			Cache-hit Ratio(%)		
Traditional CDN	20	5600	2500	3500	4500	7.5%	10%	15%	90%	95%	98%
	20	6000	3000	4000	5000	7.5%	10%	15%	90%	95%	98%
	20	7000	3500	4500	5500	7.5%	10%	15%	90%	95%	98%
Cloud-based CDN	4	5600	2500	3500	4500	7.5%	10%	15%	90%	95%	98%
	4	6000	3000	4000	5000	7.5%	10%	15%	90%	95%	98%
	4	7000	3500	4500	5500	7.5%	10%	15%	90%	95%	98%
Cloud-based CDN with Edge Computing	20	2500	2500	3500	4500	7.5%	10%	15%	90%	95%	98%
	20	3000	3000	4000	5000	7.5%	10%	15%	90%	95%	98%
	20	3500	3500	4500	5500	7.5%	10%	15%	90%	95%	98%

3.2.4 Randomization

It is generally extremely difficult for experimenters to eliminate bias using only their expert judgment and the use of randomization in the observation collection order is common practice. We used the Design Expert software to randomize the runs as Table 3.10 (in standard order).

Table 3.10. Randomization of Experiment

	Factors			
Run	Capacity (G)	Total Demand(G)	Busy-Hour Duration(%)	Cache-hit Ratio (%)
1	+	−	+	+
2	−	−	+	−
3	+	+	+	−
4	+	−	−	−
5	−	+	−	−
6	+	+	+	+
7	−	+	+	−
8	+	−	+	−
9	−	+	−	+
10	−	+	+	+
11	+	+	−	−
12	+	+	−	+
13	−	−	−	+
14	−	−	+	+
15	+	−	−	+
16	−	−	−	−

3.3 The Analysis

3.3.1 Data Analysis Method

In this section a set of hypotheses are stated and are rejected or not rejected. 14 hypotheses are tested in the experiment.

3.3.2 Test Statistics Used

The results are tested using a full-factorial experimental design. Four factors of interests: capacity, total demand, busy-hour duration and cache-hit ratio, are investigated in the analysis.

3.3.3 Significance Levels

For all tests the level of significance, an α of 0.05 is used as the cutoff for significance. If the p-value is less than 0.05, we reject the null hypothesis that there's no difference between the means and conclude that a significant difference does exist. If the p-value is larger than 0.05, we cannot conclude that a significant difference exists.

Chapter 4

Experiment Test Results and Analysis

4.1 Summary of the Data

Solution by GAMS for the base case as Table 4.1 is presented in Appendix A.

Table 4.1. Base Case of Traditional CDN

Total Capacity (G)			Total Demand (G)			Busy-hour (%)	cache-hit Ratio (%)
Traditional CDN	Cloud-based CDN	Edge Computing Sites	Idle	Normal	Busy		
6000	6000	3000	3000	4000	5000	10%	95%

4.1.1 Traditional CDN

The monthly expected profit is \$3338.35 k USD for traditional CDN base case and the SP should order bandwidth by every PoP at the beginning of each month as demonstrated in Table 4.2.

Table 4.2. Optimum Results of Traditional CDN for SP

Monthly Expected Profit = \$3338.35 k USD									
Bandwidth Ordered by Every PoP at the Beginning of the Month									
S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
205G	140G	235G	250G	250G	250G	250G	250G	250G	250G
S11	S12	S13	S14	S15	S16	S17	S18	S19	S20
190G	250G	250G	250G	250G	250G	250G	250G	250G	230G

The optimum results for ten CPs served with a traditional CDN at the end of each month are demonstrated in Table 4.3. Cache-hit ratio, round-trip time and edge-processing time were calculated to evaluate the performance of traditional CDN.

Table 4.3. Optimum Results of Traditional CDN for CPs

Idle-hour							
CP	Demand (G)	Bought (G)	Cache_Hit (G)	Cache_Miss (G)	Cache_Hit_Ratio	RTT(ms)	PT (ms)
1	100	100	95	5	95%	38.5	24
2	100	100	100		100%	30	20
3	200	200	200		100%	30	20
4	200	200	200		100%	30	20
5	300	300	285	15	95%	38.5	24
6	300	300	300		100%	30	20
7	400	400	380	20	95%	38.5	24
8	400	400	380	20	95%	38.5	24
9	500	500	475	25	95%	38.5	24
10	500	500	475	25	95%	38.5	24
Normal-hour							
CP	Demand (G)	Bought (G)	Cache_Hit (G)	Cache_Miss (G)	Cache_Hit_Ratio	RTT(ms)	PT (ms)
1	200	200	190	10	95%	38.5	24
2	200	200	190	10	95%	38.5	24
3	300	300	300		100%	30	20
4	300	300	300		100%	30	20
5	400	400	390	10	98%	34.25	22
6	400	400	400		100%	30	20
7	500	500	475	25	95%	38.5	24
8	500	500	475	25	95%	38.5	24
9	600	600	600		100%	30	20
10	600	600	600		100%	30	20
Busy-hour							
CP	Demand (G)	Bought (G)	Cache_Hit (G)	Cache_Miss (G)	Cache_Hit_Ratio	RTT(ms)	PT (ms)
1	300	300	285	15	95%	38.5	24
2	300	300	285	15	95%	38.5	24
3	400	400	380	20	95%	38.5	24
4	400	400	380	20	95%	38.5	24
5	500	500	475	25	95%	38.5	24
6	500	500	475	25	95%	38.5	24
7	600	600	570	30	95%	38.5	24
8	600	600	570	30	95%	38.5	24
9	700	700	665	35	95%	38.5	24
10	700	700	665	35	95%	38.5	24

4.1.2 Cloud-based CDN

The monthly expected profit is \$4897.625 k USD for cloud-based CDN base case. The SP should order bandwidth by every cloud domain at the beginning of each month as demonstrated in Table 4.4.

Table 4.4. Optimum Results of Cloud-based CDN for SP

Monthly Expected Profit = \$4897.625 k USD			
Bandwidth Ordered by Every Cloud Domain at the Beginning of the Month			
Cloud 1	Cloud 2	Cloud 3	Cloud 4
1100G	1180G	1250G	1220G

The optimum results for ten CPs served with a cloud-based CDN at the end of each month are demonstrated in Table 4.5. Cache-hit ratio, round-trip time and edge-processing time were calculated to evaluate the performance of cloud-based CDN. Although the monthly expected profit from cloud-based CDN is better than the traditional CDN, the performance metrics such as round-trip time and edge process time are worse off.

Table 4.5. Optimum Results of Cloud-based CDN for CP

Idle-hour							
CP	Demand (G)	Bought (G)	Cache_Hit (G)	Cache_Miss (G)	Cache_Hit_Ratio	RTT(ms)	PT (ms)
1	100	100	95	5	95%	67	28.75
2	100	100	95	5	95%	67	28.75
3	200	200	190	10	95%	67	28.75
4	200	200	190	10	95%	67	28.75
5	300	300	285	15	95%	67	28.75
6	300	300	285	15	95%	67	28.75
7	400	400	380	20	95%	67	28.75
8	400	400	380	20	95%	67	28.75
9	500	500	475	25	95%	67	28.75
10	500	500	475	25	95%	67	28.75
Normal-hour							
CP	Demand (G)	Bought (G)	Cache_Hit (G)	Cache_Miss (G)	Cache_Hit_Ratio	RTT(ms)	PT (ms)
1	200	200	190	10	95%	67	28.75
2	200	200	190	10	95%	67	28.75
3	300	300	285	15	95%	67	28.75
4	300	300	285	15	95%	67	28.75
5	400	400	380	20	95%	67	28.75
6	400	400	380	20	95%	67	28.75
7	500	500	475	25	95%	67	28.75
8	500	500	475	25	95%	67	28.75
9	600	600	570	30	95%	67	28.75
10	600	600	570	30	95%	67	28.75
Busy-hour							
CP	Demand (G)	Bought (G)	Cache_Hit (G)	Cache_Miss (G)	Cache_Hit_Ratio	RTT(ms)	PT (ms)
1	300	300	285	15	95%	67	28.75
2	300	300	285	15	95%	67	28.75
3	400	400	380	20	95%	67	28.75
4	400	400	380	20	95%	67	28.75
5	500	500	475	25	95%	67	28.75
6	500	500	475	25	95%	67	28.75
7	600	600	570	30	95%	67	28.75
8	600	600	570	30	95%	67	28.75
9	700	700	665	35	95%	67	28.75
10	700	700	665	35	95%	67	28.75

4.1.3 Cloud-based CDN with Edge Computing

The monthly expected profit is \$6465.35 k USD for cloud-based CDN with edge computing base case. The SP should order bandwidth by every cloud domain and every edge computing site at the beginning of each month as demonstrated in Table 4.6.

Table 4.6. Optimum Results of Cloud-based CDN with Edge Computing for SP

Monthly Expected Profit = \$6465.35 k USD									
Bandwidth Ordered by Every Cloud Domain at the Beginning of the Month									
Cloud 1			Cloud 2		Cloud 3		Cloud 4		
455G			490G		480G		325G		
Bandwidth Ordered by Edge Computing Sites at the Beginning of the Month									
E1	E2	E3	E4	E5	E6	E7	E8	E9	E10
150G	150G	150G	150G	150G	150G	150G	150G	150G	150G
E11	E12	E13	E14	E15	E16	E17	E18	E19	E20
150G	150G	150G	150G	150G	150G	150G	150G	150G	150G

The optimum results for ten CPs served with a cloud-based CDN with edge computing at the end of each month are demonstrated in Table 4.7. Cache-hit ratio, round-trip time and edge-processing time were calculated to evaluate the performance of cloud-based CDN with edge computing. The monthly expected profit from cloud-based CDN with edge computing is greater than the traditional CDN and cloud-based CDN, the performance metrics such as round-trip time and edge process time are also better off.

Table 4.7. Optimum Results of Cloud-based CDN with Edge Computing for CP

Idle-hour							
CP	Demand (G)	Bought (G)	Cache_Hit (G)	Cache_Miss (G)	Cache_Hit_Ratio	RTT(ms)	PT (ms)
1	100	100	95	5	95%	19.5	14.5
2	100	100	95	5	95%	19.5	14.5
3	200	200	190	10	95%	19.5	14.5
4	200	200	190	10	95%	19.5	14.5
5	300	300	285	15	95%	19.5	14.5
6	300	300	285	15	95%	19.5	14.5
7	400	400	380	20	95%	47	22.75
8	400	400	380	20	95%	47	22.75
9	500	500	475	25	95%	47	22.75
10	500	500	475	25	95%	47	22.75
Normal-hour							
CP	Demand (G)	Bought (G)	Cache_Hit (G)	Cache_Miss (G)	Cache_Hit_Ratio	RTT(ms)	PT (ms)
1	200	200	190	10	95%	19.5	14.5
2	200	200	190	10	95%	19.5	14.5
3	300	300	285	15	95%	19.5	14.5
4	300	300	285	15	95%	19.5	14.5
5	400	400	380	20	95%	19.5	14.5
6	400	400	380	20	95%	19.5	14.5
7	500	500	475	25	95%	47	22.75
8	500	500	475	25	95%	47	22.75
9	600	600	570	30	95%	47	22.75
10	600	600	570	30	95%	47	22.75
Busy-hour							
CP	Demand (G)	Bought (G)	Cache_Hit (G)	Cache_Miss (G)	Cache_Hit_Ratio	RTT(ms)	PT (ms)
1	300	300	285	15	95%	19.5	14.5
2	300	300	285	15	95%	19.5	14.5
3	400	400	380	20	95%	19.5	14.5
4	400	400	380	20	95%	19.5	14.5
5	500	500	475	25	95%	19.5	14.5
6	500	500	475	25	95%	19.5	14.5
7	600	600	570	30	95%	47	22.75
8	600	600	570	30	95%	47	22.75
9	700	700	665	35	95%	47	22.75
10	700	700	665	35	95%	47	22.75

4.1.4 Data from the Experiment

Experiments were run for traditional CDN of a full factorial two level design in four factors: capacity, total demand, busy-hour duration and cache-hit ratio, with all 16 runs taken in random order (Table 4.8).

Table 4.8. Experiment Data of Traditional CDN

Run	Capacity (G)			Total Demand (G)									Busy-hour (%)			cache-hit (%)			Expected
No.	5600	6000	7000	2500	3500	4500	3000	4000	5000	3500	4500	5500	7.5%	10%	15%	90%	95%	98%	Profit (k$)
1			+		−										+			+	2154.650
2	−				−										+	−			2953.250
3			+								+				+	−			4338.350
4			+		−								−			−			2331.875
5	−										+		−			−			4557.425
6			+								+				+			+	3875.670
7	−										+				+	−			4758.350
8			+		−										+	−			2533.250
9	−										+		−					+	4093.485
10	−										+				+			+	4295.670
11			+								+		−			−			4137.425
12			+								+		−					+	3673.485
13	−				−								−					+	2372.375
14	−				−										+			+	2574.650
15			+		−								−					+	1952.375
16	−				−								−			−			2751.875

Experiments were run for cloud-based CDN of a full factorial two level design in four factors: capacity, total demand, busy-hour duration and cache-hit ratio, with all 16 runs taken in random order (Table 4.9).

Table 4.9. Experiment Data of Cloud-based CDN

Run	Capacity (G)			Total Demand (G)									Busy-hour (%)			cache-hit (%)			Expected
No.	5600	6000	7000	2500	3500	4500	3000	4000	5000	3500	4500	5500	7.5%	10%	15%	90%	95%	98%	Profit (k$)
1			+		−										+	−			4566.000
2			+								+				+	−			6171.000
3	−										+				+	−			6171.000
4	−										+		−			−			5989.875
5			+		−								−			−			4384.875
6	−				−										+	−			4566.000
7	−				−								−			−			4384.875
8			+								+		−					+	5368.975
9	−				−								−					+	3887.975
10			+		−										+			+	4065.200
11			+								+		−			−			5989.875
12			+		−								−					+	3887.975
13			+								+				+			+	5546.200
14	−				−										+			+	4065.200
15	−										+		−					+	5368.975
16	−										+				+			+	5546.200

Experiments were run for cloud-based CDN with edge computing of a full factorial two level design in four factors: capacity, total demand, busy-hour duration and cache-hit ratio, with all 16 runs taken in random order (Table 4.10).

Table 4.10. Experiment Data of Cloud-based CDN with Edge Computing

Test	Edge Capacity (G)			Total Demand (G)									Busy-hour (%)			cache-hit (%)			Expected
No.	2500	3000	3500	2500	3500	4500	3000	4000	5000	3500	4500	5500	7.5%	10%	15%	90%	95%	98%	Profit (k$)
1			+								+		−					+	7248.122
2	−										+				+	−			7541.125
3			+		−										+			+	5720.730
4	−				−										+			+	5343.095
5			+		−								−			−			5972.875
6			+		−										+	−			6162.250
7			+								+		−			−			7839.013
8	−										+		−			−			7367.500
9	−										+		−					+	6764.587
10	−				−										+	−			5821.875
11	−				−								−			−			5632.688
12			+								+				+	−			8030.525
13	−										+				+			+	6932.925
14	−				−								−					+	5156.697
15			+								+				+			+	7437.145
16			+		−								−					+	5531.115

4.2 Statistical Analysis and Conclusions

4.2.1 Traditional CDN

The praxis used the Design Expert software to analyze the data recorded in Table 4.8 for traditional CDN. Based on the half-normal probability plot, factors A, B, C, D, BD are studied as in Figure 4.1.

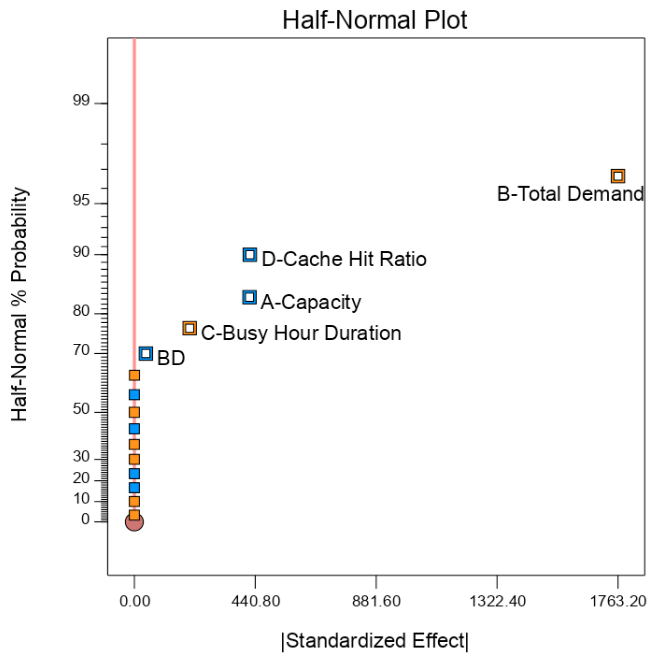
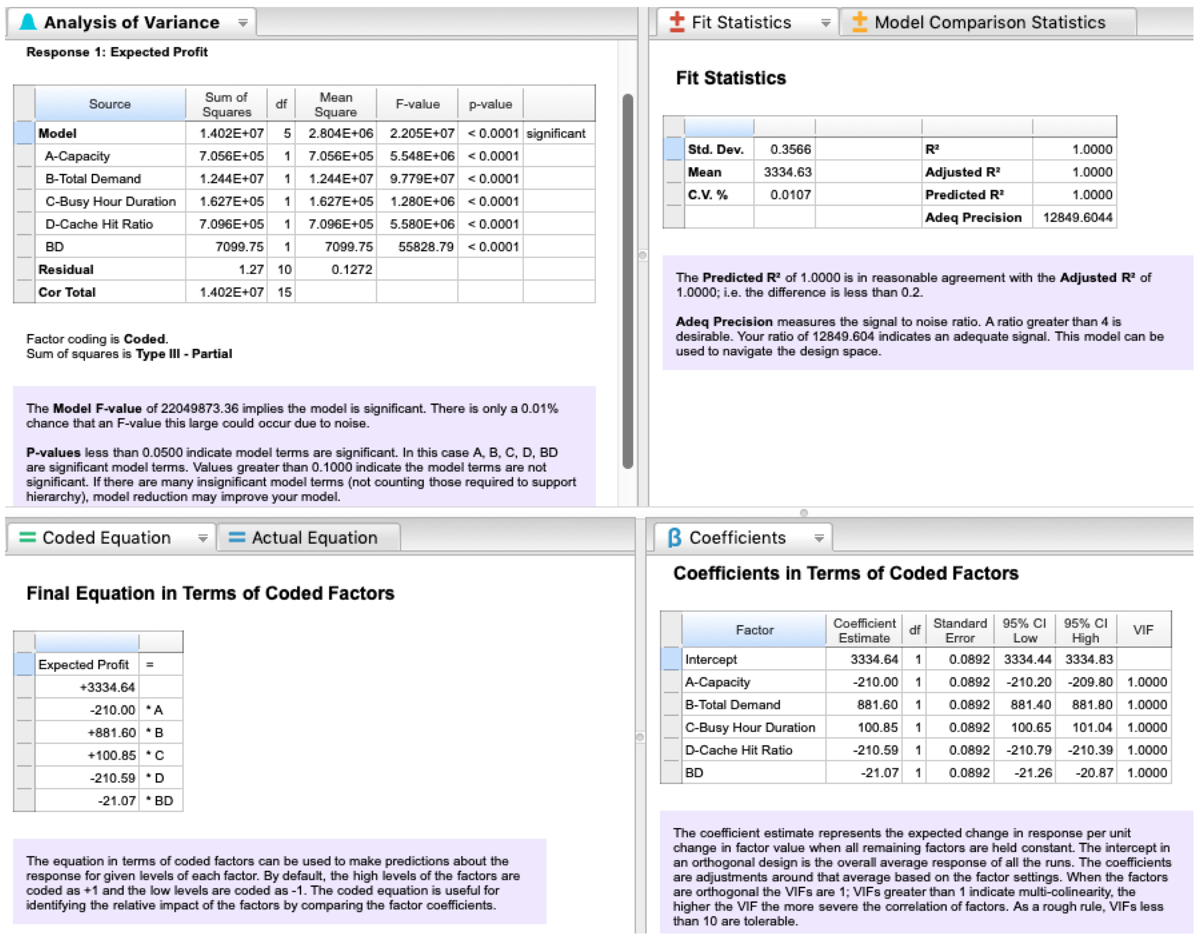


Figure 4.1. Half-normal Plot of the Experiment for Traditional CDN

According to the ANOVA results (Figure 4.2), the Model F-value of 22049873.36 implies the model is significant. There is only a 0.01% chance that an F-value this large could occur due to noise. P-values less than 0.0500 indicate model terms are significant. In this case A, B, C, D, BD are significant model terms. Thereby the vital few from the four factors experimented are capacity, total demand, busy-hour duration and cache-hit ratio requirement.



Response 1: Expected Profit

Source	Sum of Squares	df	Mean Square	F-value	p-value	
Model	1.402E+07	5	2.804E+06	2.205E+07	< 0.0001	significant
A-Capacity	7.056E+05	1	7.056E+05	5.548E+06	< 0.0001	
B-Total Demand	1.244E+07	1	1.244E+07	9.779E+07	< 0.0001	
C-Busy Hour Duration	1.627E+05	1	1.627E+05	1.280E+06	< 0.0001	
D-Cache Hit Ratio	7.096E+05	1	7.096E+05	5.580E+06	< 0.0001	
BD	7099.75	1	7099.75	55828.79	< 0.0001	
Residual	1.27	10	0.1272			
Cor Total	1.402E+07	15				

Factor coding is **Coded**.
Sum of squares is **Type III - Partial**

The **Model F-value** of 22049873.36 implies the model is significant. There is only a 0.01% chance that an F-value this large could occur due to noise.

P-values less than 0.0500 indicate model terms are significant. In this case A, B, C, D, BD are significant model terms. Values greater than 0.1000 indicate the model terms are not significant. If there are many insignificant model terms (not counting those required to support hierarchy), model reduction may improve your model.

Fit Statistics

Std. Dev.	0.3566	**R²**	1.0000
Mean	3334.63	**Adjusted R²**	1.0000
C.V. %	0.0107	**Predicted R²**	1.0000
		Adeq Precision	12849.6044

The **Predicted R²** of 1.0000 is in reasonable agreement with the **Adjusted R²** of 1.0000; i.e. the difference is less than 0.2.

Adeq Precision measures the signal to noise ratio. A ratio greater than 4 is desirable. Your ratio of 12849.604 indicates an adequate signal. This model can be used to navigate the design space.

Final Equation in Terms of Coded Factors

Expected Profit	=
+3334.64	
-210.00	* A
+881.60	* B
+100.85	* C
-210.59	* D
-21.07	* BD

The equation in terms of coded factors can be used to make predictions about the response for given levels of each factor. By default, the high levels of the factors are coded as +1 and the low levels are coded as -1. The coded equation is useful for identifying the relative impact of the factors by comparing the factor coefficients.

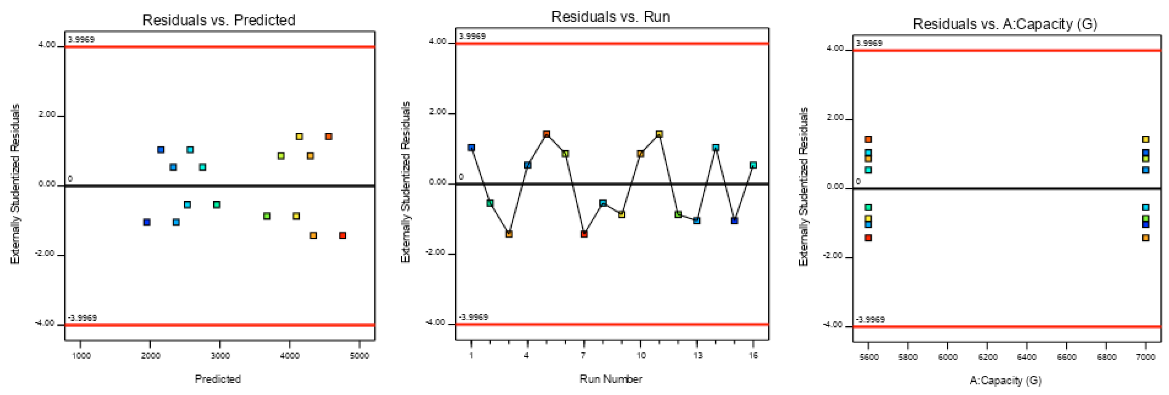
Coefficients in Terms of Coded Factors

Factor	Coefficient Estimate	df	Standard Error	95% CI Low	95% CI High	VIF
Intercept	3334.64	1	0.0892	3334.44	3334.83	
A-Capacity	-210.00	1	0.0892	-210.20	-209.80	1.0000
B-Total Demand	881.60	1	0.0892	881.40	881.80	1.0000
C-Busy Hour Duration	100.85	1	0.0892	100.65	101.04	1.0000
D-Cache Hit Ratio	-210.59	1	0.0892	-210.79	-210.39	1.0000
BD	-21.07	1	0.0892	-21.26	-20.87	1.0000

The coefficient estimate represents the expected change in response per unit change in factor value when all remaining factors are held constant. The intercept in an orthogonal design is the overall average response of all the runs. The coefficients are adjustments around that average based on the factor settings. When the factors are orthogonal the VIFs are 1; VIFs greater than 1 indicate multi-colinearity, the higher the VIF the more severe the correlation of factors. As a rough rule, VIFs less than 10 are tolerable.

Figure 4.2. ANOVA of Selected Factorial Model for Traditional CDN

According to the analysis of residuals (Figure 4.3), there is no indication of model inadequacy or violation of the assumptions.



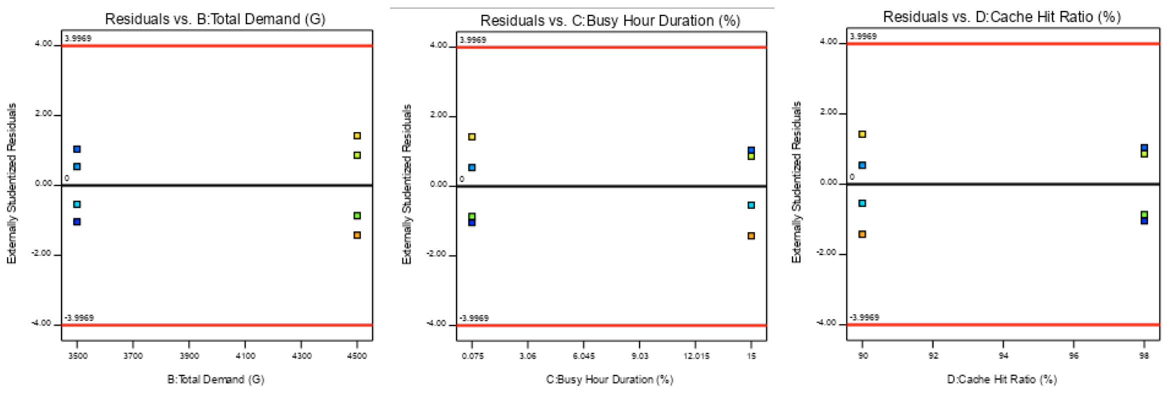


Figure 4.3. Residuals of the Experiment for Traditional CDN

4.2.2 Cloud-based CDN

The praxis used the Design Expert software to analyze the data recorded in Table 4.9 for cloud-based CDN. Based on the half-normal probability plot, factors A, B, C, D, AB, AC, AD, BC, CD, ABC are studied as in Figure 4.4.

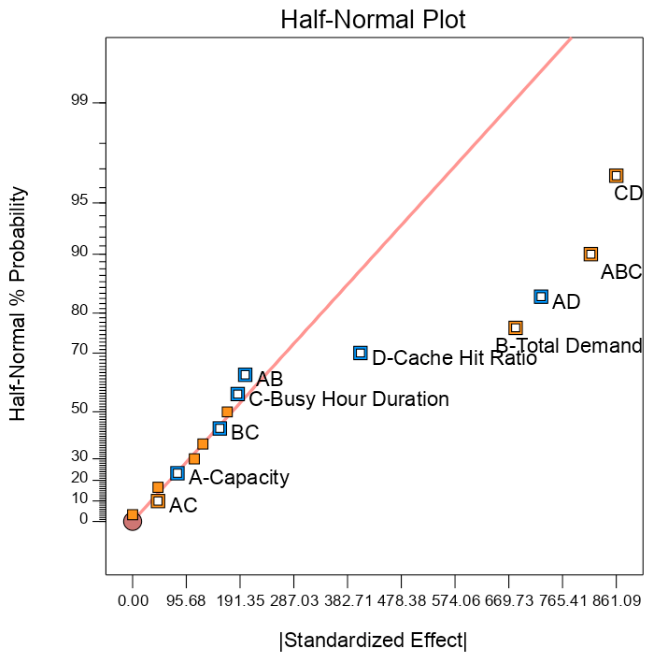
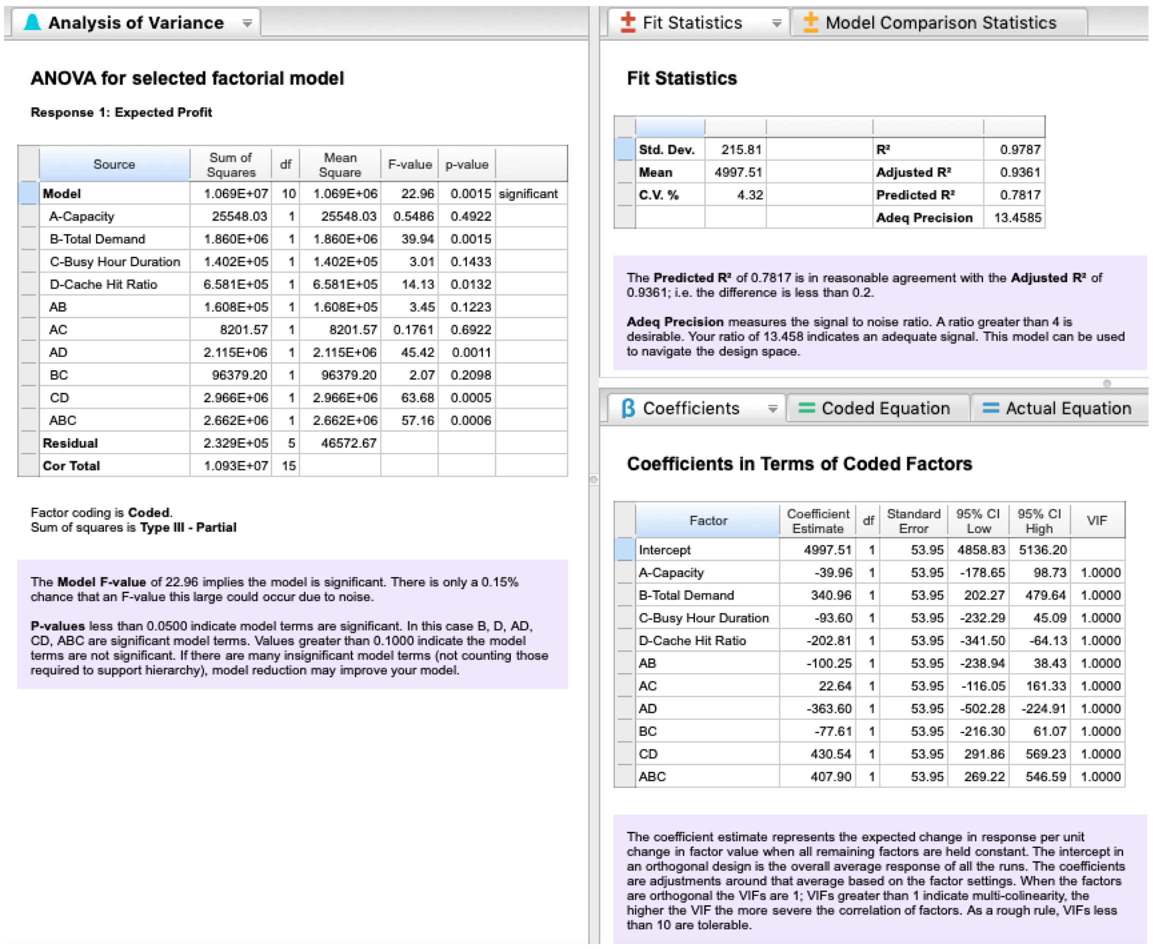


Figure 4.4. Half-normal Plot of the Experiment for Cloud-based CDN

According to the ANOVA results (Figure 4.5), the Model F-value of 22.96 implies the model is significant. There is only a 0.15% chance that an F-value this large could occur due to noise. P-values less than 0.0500 indicate model terms are significant. In this case B, D, AD, CD, ABC are significant model terms.



ANOVA for selected factorial model

Response 1: Expected Profit

Source	Sum of Squares	df	Mean Square	F-value	p-value	
Model	1.069E+07	10	1.069E+06	22.96	0.0015	significant
A-Capacity	25548.03	1	25548.03	0.5486	0.4922	
B-Total Demand	1.860E+06	1	1.860E+06	39.94	0.0015	
C-Busy Hour Duration	1.402E+05	1	1.402E+05	3.01	0.1433	
D-Cache Hit Ratio	6.581E+05	1	6.581E+05	14.13	0.0132	
AB	1.608E+05	1	1.608E+05	3.45	0.1223	
AC	8201.57	1	8201.57	0.1761	0.6922	
AD	2.115E+06	1	2.115E+06	45.42	0.0011	
BC	96379.20	1	96379.20	2.07	0.2098	
CD	2.966E+06	1	2.966E+06	63.68	0.0005	
ABC	2.662E+06	1	2.662E+06	57.16	0.0006	
Residual	2.329E+05	5	46572.67			
Cor Total	1.093E+07	15				

Factor coding is **Coded**.
Sum of squares is **Type III - Partial**

The **Model F-value** of 22.96 implies the model is significant. There is only a 0.15% chance that an F-value this large could occur due to noise.

P-values less than 0.0500 indicate model terms are significant. In this case B, D, AD, CD, ABC are significant model terms. Values greater than 0.1000 indicate the model terms are not significant. If there are many insignificant model terms (not counting those required to support hierarchy), model reduction may improve your model.

Fit Statistics

Std. Dev.	215.81	**R²**	0.9787
Mean	4997.51	**Adjusted R²**	0.9361
C.V. %	4.32	**Predicted R²**	0.7817
		Adeq Precision	13.4585

The **Predicted R²** of 0.7817 is in reasonable agreement with the **Adjusted R²** of 0.9361; i.e. the difference is less than 0.2.

Adeq Precision measures the signal to noise ratio. A ratio greater than 4 is desirable. Your ratio of 13.458 indicates an adequate signal. This model can be used to navigate the design space.

Coefficients in Terms of Coded Factors

Factor	Coefficient Estimate	df	Standard Error	95% CI Low	95% CI High	VIF
Intercept	4997.51	1	53.95	4858.83	5136.20	
A-Capacity	-39.96	1	53.95	-178.65	98.73	1.0000
B-Total Demand	340.96	1	53.95	202.27	479.64	1.0000
C-Busy Hour Duration	-93.60	1	53.95	-232.29	45.09	1.0000
D-Cache Hit Ratio	-202.81	1	53.95	-341.50	-64.13	1.0000
AB	-100.25	1	53.95	-238.94	38.43	1.0000
AC	22.64	1	53.95	-116.05	161.33	1.0000
AD	-363.60	1	53.95	-502.28	-224.91	1.0000
BC	-77.61	1	53.95	-216.30	61.07	1.0000
CD	430.54	1	53.95	291.86	569.23	1.0000
ABC	407.90	1	53.95	269.22	546.59	1.0000

The coefficient estimate represents the expected change in response per unit change in factor value when all remaining factors are held constant. The intercept in an orthogonal design is the overall average response of all the runs. The coefficients are adjustments around that average based on the factor settings. When the factors are orthogonal the VIFs are 1; VIFs greater than 1 indicate multi-colinearity, the higher the VIF the more severe the correlation of factors. As a rough rule, VIFs less than 10 are tolerable.

Figure 4.5. ANOVA of Selected Factorial Model for Cloud-based CDN

According to the analysis of residuals (Figure 4.6), there is no indication of model inadequacy or violation of the assumptions.

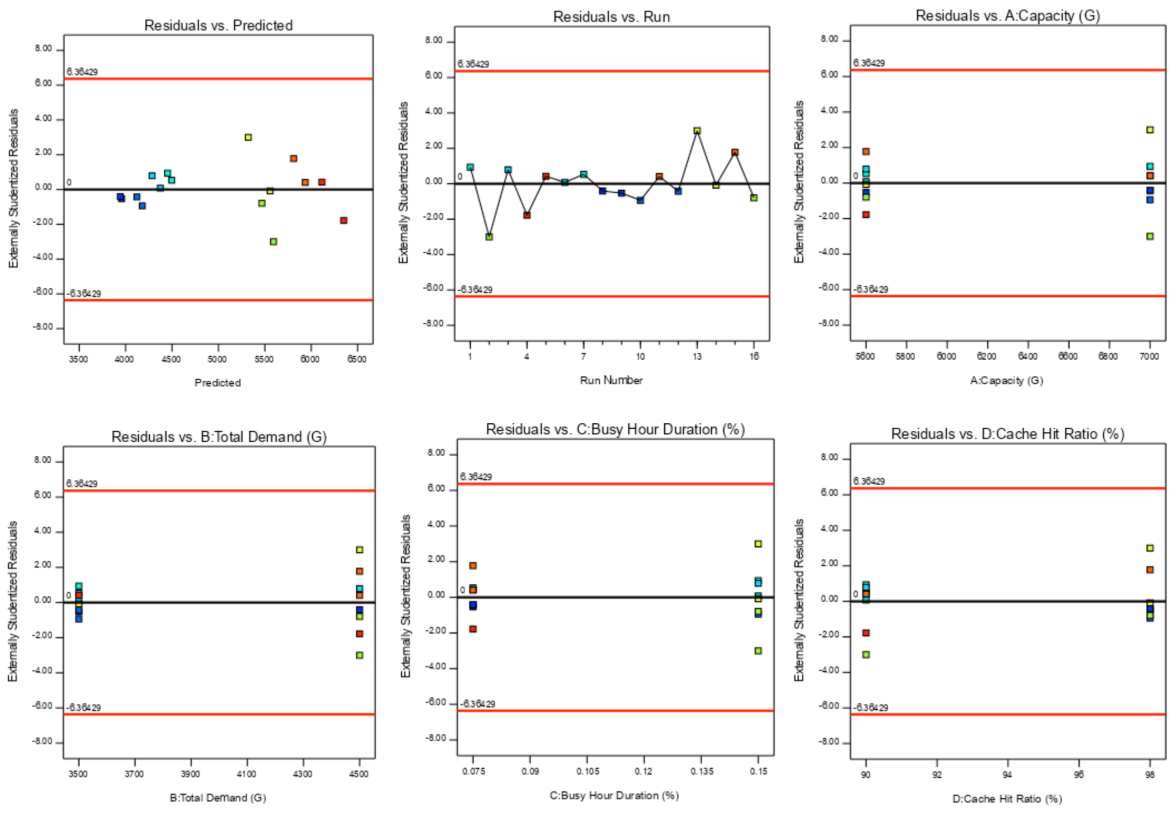


Figure 4.6. Residuals of the Experiment for Cloud-based CDN

4.2.3 Cloud-based CDN with Edge Computing

The praxis used the Design Expert software to analyze the data recorded in Table 4.10 for cloud-based CDN with edge computing. Based on the half-normal probability plot, factors A, B, C, D, AB, AD, BD are studied as in Figure 4.7.

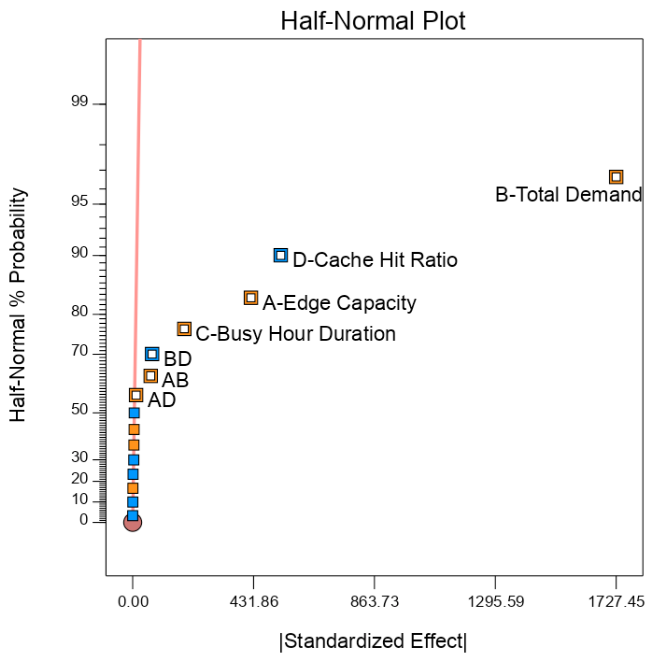
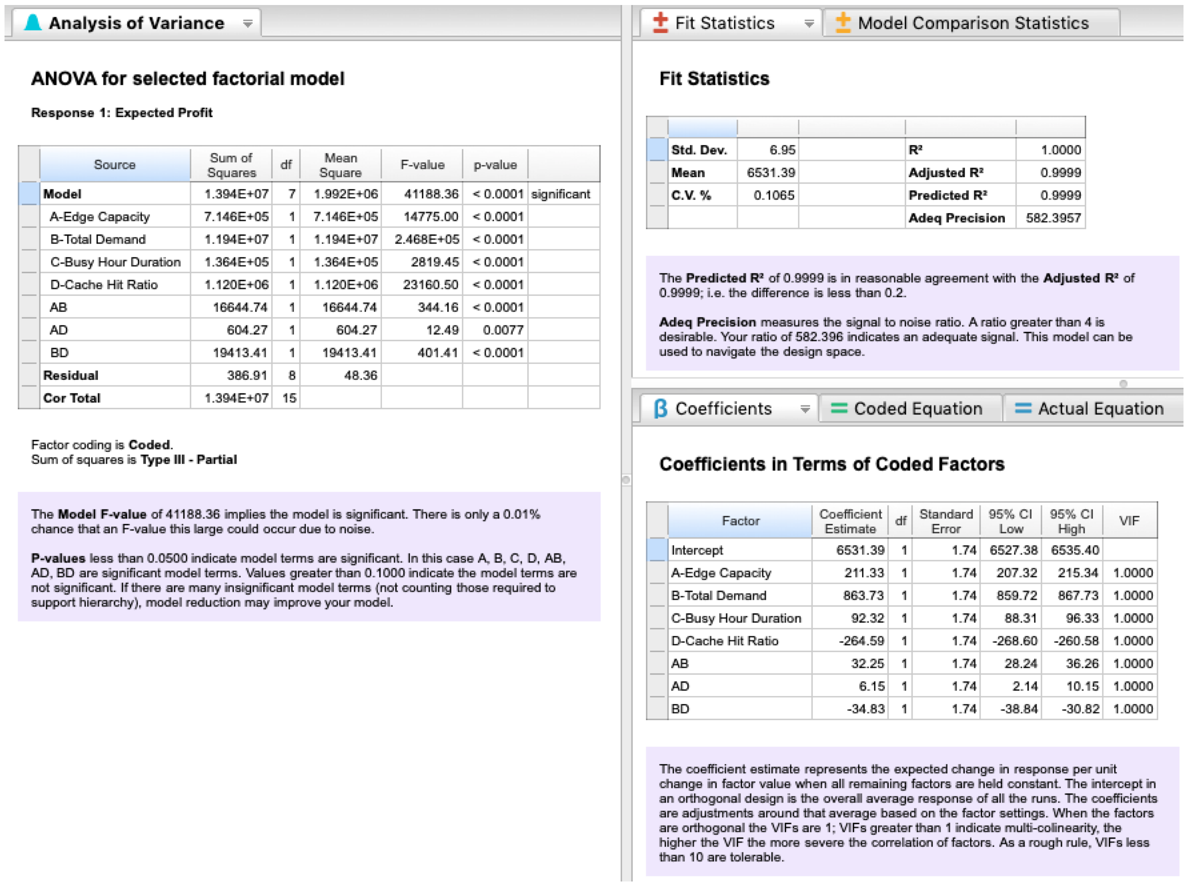


Figure 4.7. Half-normal Plot of the Experiment for Cloud-based CDN with Edge Computing

According to the ANOVA results (Figure 4.8), the Model F-value of 41188.36 implies the model is significant. There is only a 0.01% chance that an F-value this large could occur due to noise. P-values less than 0.0500 indicate model terms are significant. In this case A, B, C, D, AB, AD, BD are significant model terms.



ANOVA for selected factorial model

Response 1: Expected Profit

Source	Sum of Squares	df	Mean Square	F-value	p-value	
Model	1.394E+07	7	1.992E+06	41188.36	< 0.0001	significant
A-Edge Capacity	7.146E+05	1	7.146E+05	14775.00	< 0.0001	
B-Total Demand	1.194E+07	1	1.194E+07	2.468E+05	< 0.0001	
C-Busy Hour Duration	1.364E+05	1	1.364E+05	2819.45	< 0.0001	
D-Cache Hit Ratio	1.120E+06	1	1.120E+06	23160.50	< 0.0001	
AB	16644.74	1	16644.74	344.16	< 0.0001	
AD	604.27	1	604.27	12.49	0.0077	
BD	19413.41	1	19413.41	401.41	< 0.0001	
Residual	386.91	8	48.36			
Cor Total	1.394E+07	15				

Factor coding is **Coded**.
Sum of squares is **Type III - Partial**

The **Model F-value** of 41188.36 implies the model is significant. There is only a 0.01% chance that an F-value this large could occur due to noise.

P-values less than 0.0500 indicate model terms are significant. In this case A, B, C, D, AB, AD, BD are significant model terms. Values greater than 0.1000 indicate the model terms are not significant. If there are many insignificant model terms (not counting those required to support hierarchy), model reduction may improve your model.

Fit Statistics

Std. Dev.	6.95	**R²**	1.0000
Mean	6531.39	**Adjusted R²**	0.9999
C.V. %	0.1065	**Predicted R²**	0.9999
		Adeq Precision	582.3957

The **Predicted R²** of 0.9999 is in reasonable agreement with the **Adjusted R²** of 0.9999; i.e. the difference is less than 0.2.

Adeq Precision measures the signal to noise ratio. A ratio greater than 4 is desirable. Your ratio of 582.396 indicates an adequate signal. This model can be used to navigate the design space.

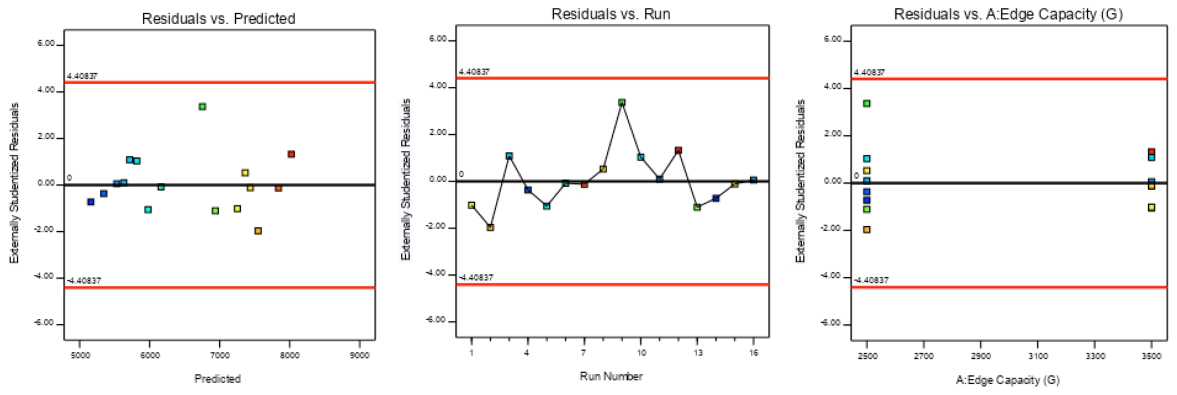
Coefficients in Terms of Coded Factors

Factor	Coefficient Estimate	df	Standard Error	95% CI Low	95% CI High	VIF
Intercept	6531.39	1	1.74	6527.38	6535.40	
A-Edge Capacity	211.33	1	1.74	207.32	215.34	1.0000
B-Total Demand	863.73	1	1.74	859.72	867.73	1.0000
C-Busy Hour Duration	92.32	1	1.74	88.31	96.33	1.0000
D-Cache Hit Ratio	-264.59	1	1.74	-268.60	-260.58	1.0000
AB	32.25	1	1.74	28.24	36.26	1.0000
AD	6.15	1	1.74	2.14	10.15	1.0000
BD	-34.83	1	1.74	-38.84	-30.82	1.0000

The coefficient estimate represents the expected change in response per unit change in factor value when all remaining factors are held constant. The intercept in an orthogonal design is the overall average response of all the runs. The coefficients are adjustments around that average based on the factor settings. When the factors are orthogonal the VIFs are 1; VIFs greater than 1 indicate multi-colinearity, the higher the VIF the more severe the correlation of factors. As a rough rule, VIFs less than 10 are tolerable.

Figure 4.8. ANOVA of Selected Factorial Model for Cloud-based CDN with Edge Computing

According to the analysis of residuals (Figure 4.9), there is no indication of model inadequacy or violation of the assumptions.



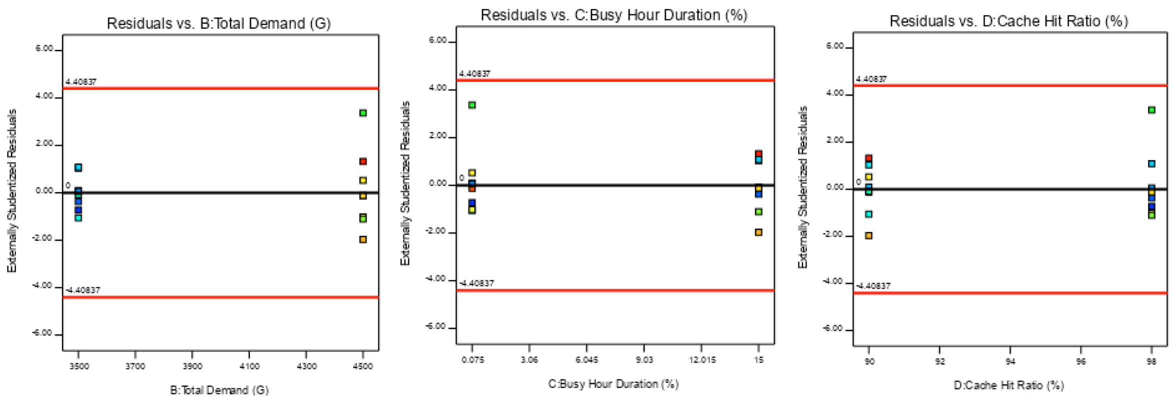


Figure 4.9. Residuals of the Experiment for Cloud-based CDN with Edge Computing

4.2.4 Factor Analysis and Hypotheses Results

H_0 #1 is rejected since different CDN network architecture has different profit gained by the SP. H_0 #2 is rejected since different CDN network architecture has different round-trip time results for CPs.

Table 4.11. Factor Analysis & Hypotheses Results

Hypotheses		Traditional CDN	Cloud-based CDN	Cloud-based CDN with Edge Computing
1-Factor	H_0 #3 (A)	rejected	not rejected	rejected
	H_0 #4 (B)	rejected	rejected	rejected
	H_0 #5 (C)	rejected	not rejected	rejected
	H_0 #6 (D)	rejected	rejected	rejected
2-Factor	H_0 #7 (AB)	not rejected	not rejected	rejected
	H_0 #8 (AC)	not rejected	not rejected	not rejected
	H_0 #9 (AD)	not rejected	rejected	rejected
	H_0 #10 (BC)	not rejected	not rejected	not rejected
	H_0 #11 (BD)	rejected	not rejected	rejected
	H_0 #12 (CD)	not rejected	rejected	not rejected
3-Factor	H_0 #13 (ABC)	not rejected	rejected	not rejected
	H_0 #14 (ABD)	not rejected	not rejected	not rejected
	H_0 #15 (ACD)	not rejected	not rejected	not rejected
	H_0 #16 (BCD)	not rejected	not rejected	not rejected

4.3 Findings

4.3.1 Traditional CDN

According to the further analysis of factors (Figure 4.10) for traditional CDN, the expected profit of traditional CDN could be improved by reducing the upfront capacity of PoPs, increasing the total demand, decreasing the busy-hour duration, or reducing the requirement for cache-hit ratio. The total demand and cache-hit ratio requirement affect the expected profit in the same way.

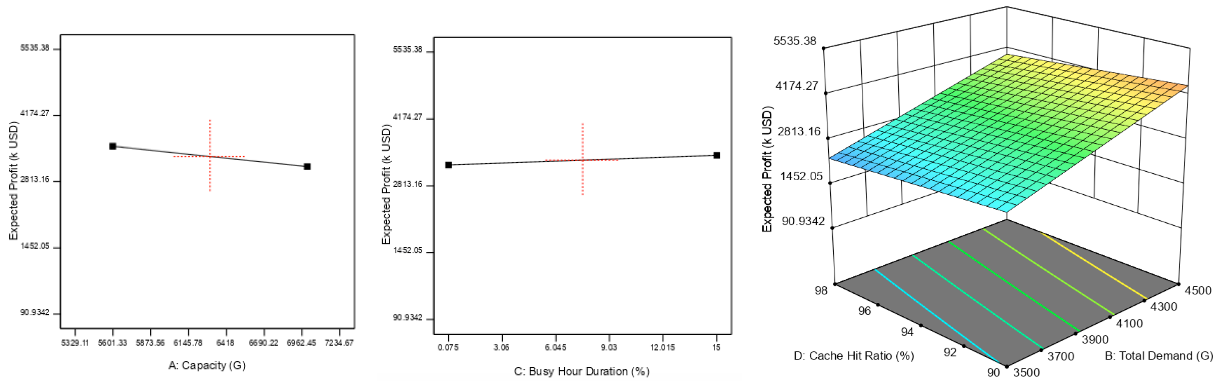


Figure 4.10. Factors of the Experiment for Traditional CDN

4.3.2 Cloud-based CDN

According to the further analysis of factor interaction (Figure 4.11) for cloud-based CDN, the expected profit of cloud-based CDN could be improved by increasing the total demand or reducing the requirement for cache-hit ratio.

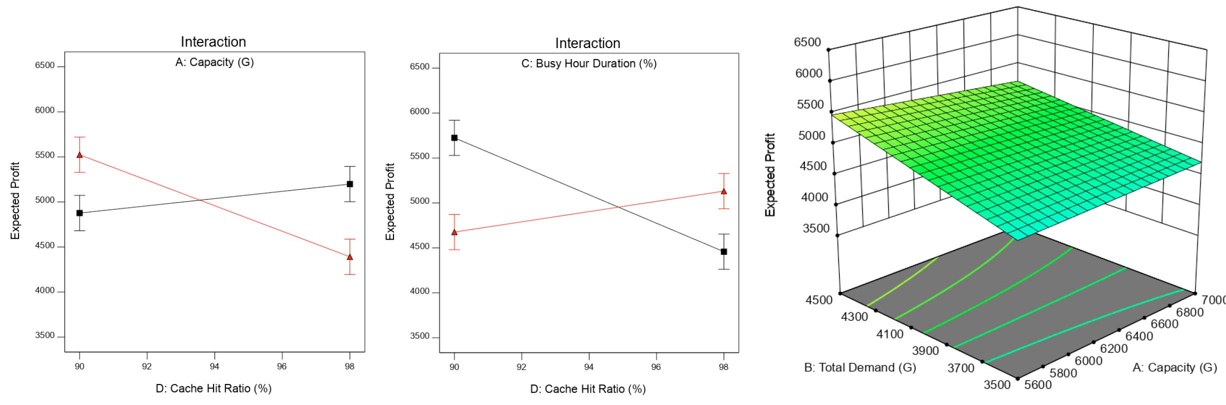


Figure 4.11. Factors of the Experiment for Cloud-based CDN

4.3.3 Cloud-based CDN with Edge Computing

According to the further analysis of factor interaction (Figure 4.12) for cloud-based CDN with edge computing, the expected profit of cloud-based CDN with edge computing could be improved by increasing the capacity of edge computing sites, increasing the total demand, decreasing the busy-hour duration, or reducing the requirement for cache-hit ratio. The capacity of edge computing sites and the total demand affect the expected profit in the same way.

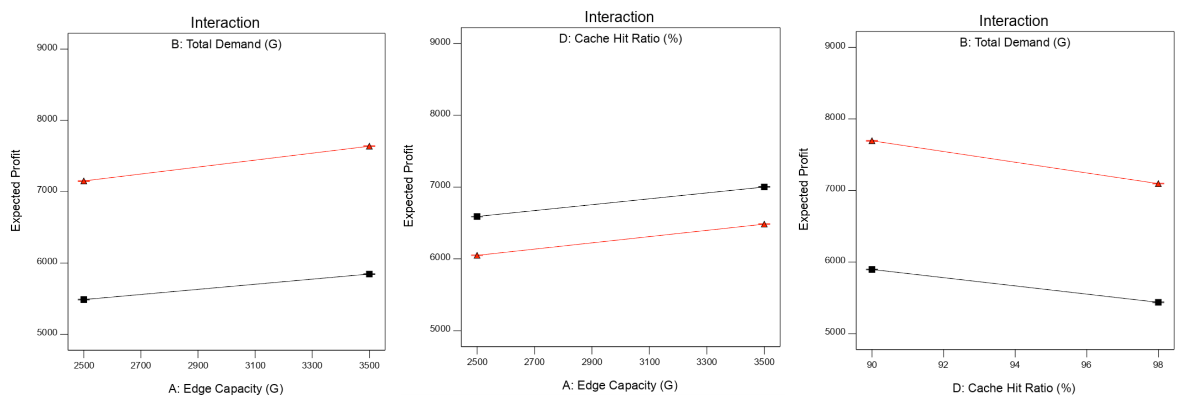


Figure 4.12. Factors of the Experiment for Cloud-based CDN with Edge Computing

4.4 Recommendations for Service Providers

According to the results of the experiments, service providers can optimize their CDN networks current in operation or in the planning stages. Recommendations regarding CDN network architecture, Capacity, total demand, busy-hour duration, cache-hit ratio requirement are presented as follows.

4.4.1 CDN Network Architecture

As previously discussed, cloud-based CDN would gain more expected profit since it saves more upfront investment than traditional CDN. Service providers that use traditional CDN or setup new CDN network should consider migrating to cloud-based CDN network architecture. Cloud-based CDN with edge computing is supposed to gain more expected profit than cloud-based CDN with better network performance in terms of round-trip time. Thereby service providers that use cloud-based CDN should consider adding edge computing sites to their CDN networks.

4.4.2 Capacity

As discussed in previous chapters, CAPEX of capacity is a linear function of its associated bandwidth for cloud-based CDN or cloud-based CDN with edge computation. Thereby service providers could invest little CAPEX upfront to leverage the infrastructure of public cloud and edge computing sites to improve the expected profit.

4.4.3 Total Demand

Total demand affects the profit gained mostly. Generally speaking, the higher the demand of CDN service, the lower of the average costs of CDN if the overall capacity is available. Service providers need to consider CPs' demand profiles to satisfy the need of the customers more efficiently. Moreover, service providers should have a good promotion plan to make sure the total demand is close to the capacity available to gain more profit.

4.4.4 Busy-hour Duration

The results from the experiments demonstrate that CP's demand of busy-hour has negative effects on expected profit. Different CPs might have different busy-hour profiles in practice and thereby service providers should design task scheduling strategies to reduce the peak demand for busy-hours duration or decrease the busy-hour duration.

4.4.5 Cache-hit Ratio Requirement

The results previously presented show that cache-hit ratio requirement by CPs has an important effect on expected profit. The higher the cache-hit ratio required, the lower the expected profit.

Service providers uses traditional CDN could hardly guarantee cache-hit ratio for every CP. Cloud-based CDN could satisfy with the cache-hit ratio requirement but the round-trip time might not be good enough. Service providers should migrate to cloud-based CDN with edge computing to get better performance and more expected profit.

4.5 Recommendation of Interconnected Exchange Cache

The deployment approach of the traditional CDN service providers involves placing their PoPs in numerous geographical locations worldwide. However, the requirements for providing high quality service through global coverage might be an obstacle for new CDN service providers, as well as affecting the commercial viability of existing ones.

The praxis introduces cloud-based CDN that CDN service providers can cooperate or leverage the global public cloud infrastructure in delivering content to the end users in a scalable manner. Such an open, cooperative model results in a coordinated and cooperative content delivery via interconnection among multiple public cloud administrative domains that could allow service providers to rapidly "scale-out" to meet anticipated increases in demand and remove the need for a given CDN to provision resources.

CDN services are often used by large enterprise customers and customers require performance commitments by signing Service Level Agreements (SLAs) with service providers. The praxis further introduces cloud-based CDN with edge computing to optimize the round-trip time (RTT) and edge-processing time for the end user requests. Economies of scale, in terms of cost effectiveness and performance for both providers and end users, could be achieved by leveraging existing underutilized infrastructure provided by public cloud and edge computing sites.

The praxis terms the technology for interconnection and inter-operation among cloud-based CDN nodes and edge computing sites as "Interconnected Cache Exchange" of CDNs or simply "ICE", which is defined as follows:

Definition of 'Interconnected Cache Exchange' – a peering arrangement formed by a set of autonomous clouds $cloud_1$, $cloud_2$, ..., $cloud_k$ and a set of edge computing sites ec_1, ec_2, ..., ec_e to provide facilities and infrastructure for cooperation among multiple public clouds and edge computing sites to allocate resources in order to ensure efficient service delivery. Each $cloud_k$ or ec_e is interconnected to other peers to exchange useful resources for performance optimization. The following issues are addressed by ICE:

- When to peer? The circumstances under which a peering arrangement could be triggered.

- How to peer? The strategy formulated to form an ICE arrangement among multiple clouds and edge computing sites.

- Who to peer with? The decision making mechanism used for choosing clouds and edge computing sites to peer with.

- How to manage to guarantee the performance in an effective way?

ICE requires fundamental research to be undertaken to address the core problems of satisfying total demand and performance requirement by multiple clouds and edge computing sites on a geographically distributed scale. Moreover, to ensure sustained resource sharing between service providers, ICE arrangements must ensure that sufficient incentive/profit exists for service providers.

The praxis presents an approach for ICE, cloud-based CDN with edge computing, to create an "open" CDN network architecture that scale well and can allocate resources among multiple clouds and edge computing sites to serve end users' requests with required performance.

Chapter 5
Conclusions and Future Research

5.1 Summary of Findings and Conclusions

The CDN industry is extremely competitive nowadays. New entrants aggressively challenge existing service providers while consumers continually demand more innovative and faster CDN services. To remain competitive, CDN service providers must provide rapid and reliable CDN network with better profitability. Although several systems introduced in previous chapters to provide CDN network services, very little conclusive CDN network research exists that considers relevant service provider CDN network design considerations. Also, new technologies are available, such as public cloud and edge computing, which provide enhanced CDN network capabilities but further complicate the CDN network systems selection process.

The praxis presents different CDN systems for solving CDN network profit maximization and performance optimization problems. three CDN network architectures are investigated. Mathematical model formulations to maximize expected profit for traditional CDN, cloud-based CDN and cloud-based CDN with edge computing are presented. Also, a calculation procedure for performance optimization in terms of round-trip time and processing time is described. The models are formulated as an Non-Linear Programming (NLP) problems to maximize the expected profit with performance requirement by network resource allocation.

A series of computational studies demonstrate the effectiveness of the models for determining which CDN network architecture is least costly given real-world constraints of the capacity, total demand, busy-hour duration, and cache-hit ratio requirement. The results found that the expected profit is substantially affected by the levels of the constraints and the selection of the least cost CDN network is dependent on the service provider network architecture, resource allocation and performance requirement. Information derived from the results of the experiments are used to determine the effects of CDN network architecture, capacity, total demand, busy-hour duration and cache-hit ration requirement.

A summary of the experiment results are as follows:

- In general, cloud-based CDN with edge computing system is the least expensive CDN architecture and traditional CDN system is the most expensive. However, the cost of recovery methods can be heavily influenced by the levels of the other factors such as capacity, total demand, busy-hour duration and cache-hit ratio requirement.

- Given that capacity established is considered a sunk cost by service providers, the results of this praxis conclusively demonstrate that the less capacity used, the more costly the CDN network. The capacity of bandwidth used in traditional CDN and cloud-based CDN is 5600G, 6000G and 7000G and the capacity of bandwidth used in edge computing is 2500G, 3000G and 3500G. The more the capacity of edge computing sites, the greater the expected profit gained by the service providers.

- The total demand of idle-hour, normal-hour and busy-hour used by the models are typical demand sizes supported by the service provider in practice. In general, the more the total demand, the more the expected profit if the capacity is still available.

- The busy-hour duration is tested using 7.5%, 10% and 15% for every 24 hours. The busy-hour duration will affect the peak demand of CDN service from CPs to service providers. The shorter the busy-hour duration, the better the profitability of CDN for service providers.

- The cache-hit ratio used by the models are 90%, 95% and 98% because they are typical requirement for the service providers by the content providers. In general, the expected profit of 90% cache-hit ratio is greater than 95%, which is greater than 98% cache-hit ratio. The results of the models indicate that the expected profit of using 90% cache-hit ratio in the CDN networks is the greatest for all CDN network architectures and total demands.

5.2 Contributions

The praxis proposed Interconnected Cache Exchange (ICE), a cloud-based CDN with edge computing platform for virtual caching to maximize profit by resource allocation and performance optimization. There are several key challenges that need to be addressed.

1. The cloud-based CDN resource allocation is complex in reality due to the cloud service providers have different strategies for different regions.

2. The complexity of the performance optimization decision is high as different customers have different requirement on round-trip time (RTT) and edge-processing time.

3. The optimization model should take into account the inherent heterogeneity in terms of CDN network capacity, total demand of different scenarios, likely busy-hour duration and cache-hit ratio requirement from the customers.

The main contributions of this praxis are summarized as follows.

1. Formulate the problem of resource allocation and performance optimization as a multi-stage stochastic linear programming model in a multiple clouds with edge computing sites environment to maximize profit.

2. Propose a method to decompose the problem into (i) a resource-allocation problem with fixed content distribution decisions and (ii) a performance improvement problem for different CDN network architectures corresponding to the resource allocation.

3. Develop a broker scheme (ICE: Interconnected Cache Edge) using cloud-based CDN with edge computing system to leverage the multi-supplier and multi-cloud environment to maximize profit and meet with the performance requirement.

The results of the models are valuable to CDN network service providers:

1. Allow technical and financial evaluation of CDN network architecture options.

2. Indicate the most cost-efficient CDN caching method based on network architectures.

3. Consider the real-world constraints of the capacity, total demand, busy-hour duration and cache-hit ratio requirement.

4. Eliminate the need for huge upfront investment for service providers.

5. Allow the evaluation of virtual caching by cloud-base CDN with edge computing.

6. Interconnected Cache Exchange (ICE) platform was developed and experimented.

5.3 Future Research

Optimizing the many parameters for the CDN architecture is challenging. There are tens of thousands of caching servers and each customer requires different metrics of performance with the network traffic changes over time. In fact, network traffic changes can happen within minutes and service providers increasingly rely on multiple cloud and edge computing sites deployments and frequently shift network traffic among them in practice.

A possible future avenue of research would be to consider the effect of cache sharing within minutes among multiple clouds and edge computing sites in virtualized CDN network architectures. In future work, we will focus on some improvements to the ICE platform, to develop a machine learning system/model that predicts the actual optimal caching decisions to further direct and benefit the service providers.

Appendix A

GAMS Code for Cloud-based CDN with Edge Computing

```
$Title Praxis ICE

$ontext
Cloud based CDN with Edge Computing problem.
Random parameter D has a discrete distribution.

$offtext

SETS    I       'Content Providers'                                 /1*10/
        J       'PoPs or Districts'                                 /S1*S20/
        K       'Clouds or Regions'                                 /C1*C4/
        E       'Edge Computing Site'                               /E1*E20/
        C       'Costs for Transit, Penalty, Holding, Storgage'     /CA, TC, PC, HC, SC, CC, EC/
        D       'Likely Demand Scenarios'                           /IH, NH, BH/
        T       'Time in ms'                                        /RTTO, RTTP, RTTC, RTTE, PTO, PTP,
            PTC, PTE/;

PARAMETER
       SP(D)    'Probability of Likely Demands'
        /  IH   .25
           NH   .65
           BH   .10 / ;

TABLE LD(I,D)   'Likely Demands of CPs'
                IH    NH    BH
           1    100   200   300
           2    100   200   300
           3    200   300   400
           4    200   300   400
           5    300   400   500
           6    300   400   500
           7    400   500   600
           8    400   500   600
           9    500   600   700
          10    500   600   700 ;


TABLE PUD(I,J) 'Likely Demand Distribution per CP by PoP'
                S1    S2    S3    S4    S5    S6    S7    S8    S9    S10   S11   S12   S13   S14   S15
                      S16   S17   S18   S19   S20
          1     .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          2     .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          3     .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          4     .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          5      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          6      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          7      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          8      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          9      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          10     .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05 ;

TABLE EUD(I,E)  'Likely Demand Distribution per CP by PoP'
                 E1    E2    E3    E4    E5    E6    E7    E8    E9    E10   E11   E12   E13   E14   E15
                     E16   E17   E18   E19   E20
          1      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          2      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          3      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          4      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          5      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          6      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          7      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          8      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          9      .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05
          10     .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05   .05
               .05   .05   .05   .05   .05  ;

TABLE PH(I,J)   '1,000 Dollars per Unit'
                 S1    S2    S3    S4    S5    S6    S7    S8    S9    S10   S11   S12   S13   S14   S15
                     S16   S17   S18   S19   S20
          1      3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0
               3.0   3.0   3.0   3.0   3.0
          2      3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0
               3.0   3.0   3.0   3.0   3.0
          3      2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8
               2.8   2.8   2.8   2.8   2.8
          4      2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8
               2.8   2.8   2.8   2.8   2.8
          5      2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5
               2.5   2.5   2.5   2.5   2.5
          6      2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5
               2.5   2.5   2.5   2.5   2.5
          7      2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2
               2.2   2.2   2.2   2.2   2.2
          8      2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2
               2.2   2.2   2.2   2.2   2.2
          9      2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0
               2.0   2.0   2.0   2.0   2.0
          10      2.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0
                2.0    2.0    2.0    2.0    2.0 ;

TABLE PM(I,J)   '1,000 Dollars per Unit'
                  S1     S2     S3     S4     S5     S6     S7     S8     S9     S10    S11    S12    S13    S14    S15
                       S16    S17    S18    S19    S20
          1       3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5
                3.5    3.5    3.5    3.5    3.5
          2       3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5    3.5
                3.5    3.5    3.5    3.5    3.5
          3       3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0
                3.0    3.0    3.0    3.0    3.0
          4       3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0
                3.0    3.0    3.0    3.0    3.0
          5       3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0
                3.0    3.0    3.0    3.0    3.0
          6       3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0
                3.0    3.0    3.0    3.0    3.0
          7       3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0
                3.0    3.0    3.0    3.0    3.0
          8       3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0
                3.0    3.0    3.0    3.0    3.0
          9       2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5
                2.5    2.5    2.5    2.5    2.5
          10      2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5    2.5
                2.5    2.5    2.5    2.5    2.5 ;

TABLE CH(I,K)   '1,000 Dollars per Unit'
                  C1     C2     C3     C4
          1       2.5    2.5    2.5    2.5
          2       2.5    2.5    2.5    2.5
          3       2.5    2.5    2.5    2.5
          4       2.5    2.5    2.5    2.5
          5       2.5    2.5    2.5    2.5
          6       2.5    2.5    2.5    2.5
          7       2.5    2.5    2.5    2.5
          8       2.5    2.5    2.5    2.5
          9       2.5    2.5    2.5    2.5
          10      2.5    2.5    2.5    2.5 ;


TABLE CM(I,K)   '1,000 Dollars per Unit'
                  C1     C2     C3     C4
          1       3.0    3.0    3.0    3.0
          2       3.0    3.0    3.0    3.0
          3       3.0    3.0    3.0    3.0
          4       3.0    3.0    3.0    3.0
          5       3.0    3.0    3.0    3.0
          6       3.0    3.0    3.0    3.0
          7       3.0    3.0    3.0    3.0
          8       3.0    3.0    3.0    3.0
          9       3.0    3.0    3.0    3.0
          10      3.0    3.0    3.0    3.0 ;

TABLE EH(I,E)   '1,000 Dollars per Unit'
                  E1     E2     E3     E4     E5     E6     E7     E8     E9     E10    E11    E12    E13    E14    E15
                       E16    E17    E18    E19    E20
```

```
        1     3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0
            3.0   3.0   3.0   3.0   3.0
        2     3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0
            3.0   3.0   3.0   3.0   3.0
        3     2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8
            2.8   2.8   2.8   2.8   2.8
        4     2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8
            2.8   2.8   2.8   2.8   2.8
        5     2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5
            2.5   2.5   2.5   2.5   2.5
        6     2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5
            2.5   2.5   2.5   2.5   2.5
        7     2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2
            2.2   2.2   2.2   2.2   2.2
        8     2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2   2.2
            2.2   2.2   2.2   2.2   2.2
        9     2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0
            2.0   2.0   2.0   2.0   2.0
        10    2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0   2.0
            2.0   2.0   2.0   2.0   2.0 ;

TABLE EM(I,E)   '1,000 Dollars per Unit'
              E1    E2    E3    E4    E5    E6    E7    E8    E9    E10   E11   E12   E13   E14   E15
                  E16   E17   E18   E19   E20
        1     3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5
            3.5   3.5   3.5   3.5   3.5
        2     3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5   3.5
            3.5   3.5   3.5   3.5   3.5
        3     3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2
            3.2   3.2   3.2   3.2   3.2
        4     3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2   3.2
            3.2   3.2   3.2   3.2   3.2
        5     3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0
            3.0   3.0   3.0   3.0   3.0
        6     3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0   3.0
            3.0   3.0   3.0   3.0   3.0
        7     2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8
            2.8   2.8   2.8   2.8   2.8
        8     2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8   2.8
            2.8   2.8   2.8   2.8   2.8
        9     2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5
            2.5   2.5   2.5   2.5   2.5
        10    2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5   2.5
            2.5   2.5   2.5   2.5   2.5 ;

TABLE CUD(J,K)  'Demand Distribution of Districts by Cloud'
              C1    C2    C3    C4
        S1    1
        S2    1
        S3    1
        S4    1
        S5    1
        S6          1
        S7          1
        S8          1
        S9          1
        S10         1
        S11               1
```

```
          S12                  1
          S13                  1
          S14                  1
          S15                  1
          S16                         1
          S17                         1
          S18                         1
          S19                         1
          S20                         1  ;

TABLE EUR(E,K)  'Demand Distribution of Districts by Cloud'
                 C1     C2     C3     C4
          E1     1
          E2     1
          E3     1
          E4     1
          E5     1
          E6            1
          E7            1
          E8            1
          E9            1
          E10           1
          E11                  1
          E12                  1
          E13                  1
          E14                  1
          E15                  1
          E16                         1
          E17                         1
          E18                         1
          E19                         1
          E20                         1  ;

TABLE CATC(J,C)  'Capacity & Related Costs in 1,000 Dollars per Unit'
                 CA     TC     PC     HC     SC     CC     EC
          S1     300    0.8    0.3    0.2    0.3
          S2     300    0.8    0.3    0.2    0.3
          S3     300    0.8    0.3    0.2    0.3
          S4     300    0.8    0.3    0.2    0.3
          S5     300    0.8    0.3    0.2    0.3
          S6     300    0.8    0.3    0.2    0.3
          S7     300    0.8    0.3    0.2    0.3
          S8     300    0.8    0.3    0.2    0.3
          S9     300    0.8    0.3    0.2    0.3
          S10    300    0.8    0.3    0.2    0.3
          S11    300    0.8    0.3    0.2    0.3
          S12    300    0.8    0.3    0.2    0.3
          S13    300    0.8    0.3    0.2    0.3
          S14    300    0.8    0.3    0.2    0.3
          S15    300    0.8    0.3    0.2    0.3
          S16    300    0.8    0.3    0.2    0.3
          S17    300    0.8    0.3    0.2    0.3
          S18    300    0.8    0.3    0.2    0.3
          S19    300    0.8    0.3    0.2    0.3
          S20    300    0.8    0.3    0.2    0.3;

TABLE CACC(K,C)  'Capacity & Related Costs in 1,000 Dollars'
                 CA     TC     PC     HC     SC     CC     EC
          C1      1500   0.8           0.1           0.25
          C2      1500   0.8           0.1           0.25
          C3      1500   0.8           0.1           0.25
          C4      1500   0.8           0.1           0.25 ;

TABLE CAEC(E,C)  'Capacity & Related Costs in 1,000 Dollars per Unit'
                  CA     TC     PC     HC     SC     CC     EC
          E1      150    0.4           0.1                  0.1
          E2      150    0.4           0.1                  0.1
          E3      150    0.4           0.1                  0.1
          E4      150    0.4           0.1                  0.1
          E5      150    0.4           0.1                  0.1
          E6      150    0.4           0.1                  0.1
          E7      150    0.4           0.1                  0.1
          E8      150    0.4           0.1                  0.1
          E9      150    0.4           0.1                  0.1
          E10     150    0.4           0.1                  0.1
          E11     150    0.4           0.1                  0.1
          E12     150    0.4           0.1                  0.1
          E13     150    0.4           0.1                  0.1
          E14     150    0.4           0.1                  0.1
          E15     150    0.4           0.1                  0.1
          E16     150    0.4           0.1                  0.1
          E17     150    0.4           0.1                  0.1
          E18     150    0.4           0.1                  0.1
          E19     150    0.4           0.1                  0.1
          E20     150    0.4           0.1                  0.1 ;

TABLE TI(I,T)  'RTT and PT in ms'
                  RTTO   RTTP   RTTC   RTTE   PTO   PTP   PTC   PTE
          1       200    30     60     10     100   20    25    10
          2       180    25     60     10     90    18    25    10
          3       200    30     60     10     100   20    25    10
          4       180    25     60     10     90    18    25    10
          5       200    30     60     10     100   20    25    10
          6       180    25     60     10     90    18    25    10
          7       200    30     60     10     100   20    25    10
          8       180    25     60     10     90    18    25    10
          9       200    30     60     10     100   20    25    10
          10      180    25     60     10     90    18    25    10 ;

Scalar    MINHIT   'Minimum Cache Hit Requirement'  / .95 /;
Scalar    MAXRTT   'Maximum RTT Requirement'  / 38 /;

Variable ZT        'Expected Profit for Traditional CDN in 1000 Dollars'
          ZC        'Expected Profit for Cloud based CDN in 1000 Dollars'
          ZE        'Expected Profit for Cloud based CDN with EC in 1000 Dollars';

Positive Variables
          PBT(J)         'Transit Bought at PoP J by SP at stage 1'
          PCH(J,D)       'Cache Hitted at PoP J of Likely Demand'
          PCM(J,D)       'Cache Missed at PoP J of Likely Demand'
          PCN(J,D)       'Cache Unused at PoP J of Likely Demand'
          XT(I,D)        'Transit used by CP I in Total at Stage 2'
          DPH(I,J,D)     'Cache Hitted of CP I at PoP J at Stage 2'
          DPM(I,J,D)     'Cache Missed of CP I at PoP J at Stage 2'

          CBT(K)         'Transit Bought at Cloud K by SP at Stage 1'
          CCH(K,D)     'Cache Hitted at Cloud K of Likely Demand'
          CCM(K,D)     'Cache Missed at Cloud K of Likely Demand'
          CCN(K,D)     'Cache Unused at Cloud K of Likely Demand'
          XC(I,D)      'Transit used by CP I of CCDN at Stage 2'
          RCH(I,K,D)   'Cache Hitted of CP I at Cloud K at Stage 2'
          RCM(I,K,D)   'Cache Missed of CP I at Cloud K at Stage 2'


          EBC(K)       'Transit Bought at Cloud K by SP at stage 1'
          EBT(E)       'Transit Bought at Edge E by SP at stage 1'
          ECH(K,D)     'Cache Hitted at Cloud K of Likely Demand'
          ECM(K,D)     'Cache Missed at Cloud K of Likely Demand'
          ECN(K,D)     'Cache Unused at Cloud K of Likely Demand'
          EEH(E,D)     'Cache Hitted at Edge E of Likely Demand'
          EEM(E,D)     'Cache Missed at Edge E of Likely Demand'
          EEN(E,D)     'Cache Unused at Edge E of Likely Demand'
          XEC(I,D)     'Cloud Transit Used by CP I of CCDN with EC at Stage 2'
          XEE(I,D)     'Edge Transit Used by CP I of CCDN with EC at Stage 2'
          REH(I,K,D)   'Cache Hitted of CP I at Cloud K at Stage 2'
          REM(I,K,D)   'Cache Missed of CP I at Cloud K at Stage 2'
          DEH(I,E,D)   'Cache Hitted of CP I at Edge E at Stage 2'
          DEM(I,E,D)   'Cache Missed of CP I at Edge E at Stage 2';

Equations ETPROFIT, PDEM(D), PUDE(I,D), DDEM(J,D), CAPB(J), CAPU(J,D), PINV(J,D), TCPH(J,D), TCPM(J,D),
     HITRP(I,D), TBGHT(I,D);

ETPROFIT..   ZT    =E= sum((I,J,D), DPH(I,J,D)*PH(I,J)*SP(D)) + sum((I,J,D), DPM(I,J,D)*PM(I,J)*SP(D))
     sum(J, CATC(J,'CA')*CATC(J,'SC'))
                         sum(J, PBT(J)*CATC(J,'TC'))   sum((J,D), PCN(J,D)*CATC(J,'HC')*SP(D))   sum((J,D),
                           PCM(J,D)*CATC(J,'PC')*SP(D));

PDEM(D)..      sum(I,LD(I,D))            =E= sum(J, PCH(J,D)) + sum(J, PCM(J,D));
PUDE(I,D)..    LD(I,D)                   =G= XT(I,D);
DDEM(J,D)..    sum(I,LD(I,D)*PUD(I,J))   =E= sum(I, DPH(I,J,D)) + sum(I, DPM(I,J,D));
CAPB(J)..      PBT(J)                    =L= CATC(J,'CA');
CAPU(J,D)..    CATC(J,'CA')              =G= PCH(J,D) + PCM(J,D);
PINV(J,D)..    PCN(J,D)                  =E= PBT(J)   PCH(J,D);
TCPH(J,D)..    PCH(J,D)                  =E= sum(I, DPH(I,J,D));
TCPM(J,D)..    PCM(J,D)                  =E= sum(I, DPM(I,J,D));
HITRP(I,D)..   sum(J,DPH(I,J,D))         =G= MINHIT * XT(I,D);
TBGHT(I,D)..   XT(I,D)                   =E= sum(J, DPH(I,J,D)) + sum(J, DPM(I,J,D));

Model   TCDN / ETPROFIT, PDEM, PUDE, DDEM, CAPB, CAPU, PINV, TCPH, TCPM, HITRP, TBGHT /;
Option  LIMROW=0,LIMCOL=0,SYSOUT=OFF;
Solve   TCDN USING NLP MAXIMIZING ZT;

Equations ECPROFIT, CDEM(D), CUDE(I,D), RDEM(K,D), CACB(K), CACU(K,D), CINV(K,D), TCCH(K,D), TCCM(K,D),
     HITRC(I,D), CBGHT(I,D);

ECPROFIT..   ZC    =E= sum((I,K,D), RCH(I,K,D)*CH(I,K)*SP(D)) + sum((I,K,D), RCM(I,K,D)*CM(I,K)*SP(D))
     sum((K,D), CCH(K,D)*CACC(K,'CC')*SP(D))
                         sum(K, CBT(K)*CACC(K,'TC'))   sum((K,D), CCN(K,D)*CACC(K,'HC')*SP(D))   sum((K,D),
                           CCM(K,D)*CACC(K,'PC')*SP(D));

CDEM(D)..      sum(I, LD(I,D))                        =E= sum(K, CCH(K,D)) + sum(K, CCM(K,D));
CUDE(I,D)..    LD(I,D)                                =G= XC(I,D);
RDEM(K,D)..    sum(I,RCH(I,K,D))+sum(I,RCM(I,K,D))    =E= sum(I, LD(I,D)*sum(J,PUD(I,J)*CUD(J,K)));
CACB(K)..      CBT(K)                                 =L= CACC(K,'CA');
```

```
CACU(K,D) ..    CACC(K, 'CA')                                    =G= CCH(K,D) + CCM(K,D);
CINV(K,D) ..    CCN(K,D)                                         =E= CBT(K)    CCH(K,D);
TCCH(K,D) ..    CCH(K,D)                                         =E= sum(I, RCH(I,K,D));
TCCM(K,D) ..    CCM(K,D)                                         =E= sum(I, RCM(I,K,D));
HITRC(I,D) ..   sum(K, RCH(I,K,D))                               =G= MINHIT * XC(I,D);
CBGHT(I,D) ..   XC(I,D)                                          =E= sum(K, RCH(I,K,D)) + sum(K, RCM(I,K,D));

Model   CCDN / ECPROFIT, CDEM, CUDE, RDEM, CACB, CACU, CINV, TCCH, TCCM, HITRC, CBGHT /;
Option  LIMROW=0,LIMCOL=0,SYSOUT=OFF;
Solve   CCDN USING NLP MAXIMIZING ZC;

Equations EEPROFIT, ETDE(D), EUDE(I,D), ERDE(K,D), EBCB(K), EBTB(E), EBCU(K,D), EBTU(E,D), ECIN(K,D),
     EEIN(E,D), TCEH(K,D), TCEM(K,D),
          TEEH(E,D), TEEM(E,D), HITRE(I,D), RTTRE(I,D), EBGHT(I,D), EEBGT(I,D);

EEPROFIT..     ZE     =E= sum((I,K,D), REH(I,K,D)*CH(I,K)*SP(D)) + sum((I,K,D), REM(I,K,D)*CM(I,K)*SP(D)) +
     sum((I,E,D), DEH(I,E,D)*EH(I,E)*SP(D))
                         + sum((I,E,D), DEM(I,E,D)*EM(I,E)*SP(D))    sum(K, EBC(K)*CACC(K,'TC'))    sum(E,
                              EBT(E)*CAEC(E,'TC'))
                           sum((K,D), ECH(K,D)*CACC(K,'CC')*SP(D))   sum((E,D), EEH(E,D)*CAEC(E,'EC')*SP(D))
                           sum((K,D), ECN(K,D)*CACC(K,'HC')*SP(D))    sum((E,D), EEN(E,D)*CAEC(E,'HC')*SP(D))  ;

ETDE(D) ..       sum(I, LD(I,D))                                    =E= sum(K, ECH(K,D)) + sum(K, ECM(K,D)) + sum(E,
     EEH(E,D)) + sum(E, EEM(E,D));
EUDE(I,D) ..    LD(I,D)                                             =G= XEC(I,D) + XEE(I,D);
ERDE(K,D) ..    sum(I, LD(I,D)*sum(J,PUD(I,J)*CUD(J,K))) =E= sum(I,(REH(I,K,D) + REM(I,K,D) +
     sum(E,(DEH(I,E,D)+DEM(I,E,D))*EUR(E,K))));
EBCB(K) ..      EBC(K)                                              =L= CACC(K,'CA');
EBTB(E) ..      EBT(E)                                              =L= CAEC(E,'CA');
EBCU(K,D) ..    CACC(K,'CA')                                        =G= ECH(K,D) + ECM(K,D);
EBTU(E,D) ..    CAEC(E,'CA')                                        =G= EEH(E,D) + EEM(E,D);
ECIN(K,D) ..    ECN(K,D)                                            =E= EBC(K)    ECH(K,D);
EEIN(E,D) ..    EEN(E,D)                                            =E= EBT(E)    EEH(E,D);
TCEH(K,D) ..    ECH(K,D)                                            =E= sum(I, REH(I,K,D));
TCEM(K,D) ..    ECM(K,D)                                            =E= sum(I, REM(I,K,D));
TEEH(E,D) ..    EEH(E,D)                                            =E= sum(I, DEH(I,E,D));
TEEM(E,D) ..    EEM(E,D)                                            =E= sum(I, DEM(I,E,D));
HITRE(I,D) ..   sum(K, REH(I,K,D)) + sum(E, DEH(I,E,D))   =G= MINHIT * (XEC(I,D) + XEE(I,D));
RTTRE(I,D) ..   MAXRTT * (XEC(I,D) + XEE(I,D))            =G= XEC(I,D)*TI(I,'RTTC') + XEE(I,D)*TI(I,'RTTE');
EBGHT(I,D) ..   XEC(I,D)                                            =E= sum(K, REH(I,K,D)) + sum(K, REM(I,K,D));
EEBGT(I,D) ..   XEE(I,D)                                            =E= sum(E, DEH(I,E,D)) + sum(E, DEM(I,E,D));

Model   ICE / EEPROFIT, ETDE, EUDE, ERDE, EBCB, EBTB, EBCU, EBTU, ECIN, EEIN, TCEH, TCEM, TEEH, TEEM,
     HITRE, RTTRE, EBGHT, EEBGT /;
Option  LIMROW=0,LIMCOL=0,SYSOUT=OFF;
Solve   ICE USING NLP MAXIMIZING ZE;

Parameter CPTCRep 'CP Summary Report for Traditonal CDN';
Loop (D,
CPTCRep(I,'Demand')      = LD(I,D);
CPTCRep(I,'Bought')      = XT.L(I,D);
CPTCRep(I,'Cache_Hit') = sum(J, DPH.L(I,J,D));
CPTCRep(I,'Cache_Miss')= sum(J, DPM.L(I,J,D));
CPTCRep(I,'Hit_Ratio') = sum(J, DPH.L(I,J,D))/XT.L(I,D);
CPTCRep(I,'RTT(ms)')    = sum(J, DPH.L(I,J,D))/XT.L(I,D)*TI(I,'RTTP') + (1    sum(J,
     DPH.L(I,J,D))/XT.L(I,D))*TI(I,'RTTO');
CPTCRep(I,'PT(ms)')      = sum(J, DPH.L(I,J,D))/XT.L(I,D)*TI(I,'PTP') + (1    sum(J,
     DPH.L(I,J,D))/XT.L(I,D))*TI(I,'PTO');
CPTCRep(I,'Revenue(k)')= sum(J, DPH.L(I,J,D)*PH(I,J)*SP(D)) + sum(J, DPM.L(I,J,D)*PM(I,J)*SP(D));
Display CPTCRep;);

Parameter PoPRepD 'PoP Summary Report';
Loop (D,
PoPRepD(J,'Capacity')   = CATC(J,'CA');
PoPRepD(J,'Demand')     = sum(I, LD(I,D)*PUD(I,J));
PoPRepD(J,'Bought')     = PBT.L(J);
PoPRepD(J,'Cache_Hit')  = PCH.L(J,D);
PoPRepD(J,'Cache_Miss')= PCM.L(J,D);
PoPRepD(J,'Unused')     = PCN.L(J,D);
PoPRepD(J,'Costs(k)')   = (PBT.L(J)*CATC(J,'TC') + PCN.L(J,D)*CATC(J,'HC') + PCM.L(J,D)*CATC(J,'PC') +
     CATC(J,'CA')*CATC(J,'SC'))*SP(D);
PoPRepD(J,'Hit_Ratio') = (PCH.L(J,D)+1)/(PCH.L(J,D) + PCM.L(J,D) + 1);
Display PoPRepD;);

Parameter CPCCRep 'CP Summary Report for Cloud based CDN';
Loop (D,
CPCCRep(I,'Demand')     = LD(I,D);
CPCCRep(I,'Bought')     = XC.L(I,D);
CPCCRep(I,'Cache_Hit') = sum(K, REH.L(I,K,D)) + sum(E, DEH.L(I,E,D));
CPCCRep(I,'Cache_Miss')= sum(K, REM.L(I,K,D)) + sum(E, DEM.L(I,E,D));
CPCCRep(I,'Hit_Ratio') = sum(K, RCH.L(I,K,D))/XC.L(I,D);
CPCCRep(I,'RTT(ms)')    = sum(K, RCH.L(I,K,D))/XC.L(I,D)*TI(I,'RTTC') + (1    sum(K,
     RCH.L(I,K,D))/XC.L(I,D))*TI(I,'RTTO');
CPCCRep(I,'PT(ms)')     = sum(K, RCH.L(I,K,D))/XC.L(I,D)*TI(I,'PTC') + (1    sum(K,
     RCH.L(I,K,D))/XC.L(I,D))*TI(I,'PTO');
CPCCRep(I,'Revenue(k)')= sum(K, RCH.L(I,K,D)*CH(I,K)*SP(D)) + sum(K, RCM.L(I,K,D)*CM(I,K)*SP(D));
Display CPCCRep;);

Parameter CloudRepD 'Cloud based CDN Summary Report';
Loop (D,
CloudRepD(K,'Capacity')   = CACC(K,'CA');
CloudRepD(K,'Demand')     = sum(I,RCH.L(I,K,D))+sum(I,RCM.L(I,K,D));
CloudRepD(K,'Bought')     = CBT.L(K);
CloudRepD(K,'Cache_Hit')  = CCH.L(K,D);
CloudRepD(K,'Cache_Miss')= CCM.L(K,D);
CloudRepD(K,'Unused')     = CCN.L(K,D);
CloudRepD(K,'Costs(k)')   = (CBT.L(K)*CACC(K,'TC') + CCN.L(K,D)*CACC(K,'HC') +
     CCH.L(K,D)*CACC(K,'CC'))*SP(D);
CloudRepD(K,'Hit_Ratio') = (CCH.L(K,D)+1)/(CCH.L(K,D) + CCM.L(K,D) + 1);
Display CloudRepD;);

Parameter CPECRep 'CP Summary Report for Cloud based CDN with EC';
Loop (D,
CPECRep(I,'Demand')     = LD(I,D);
CPECRep(I,'Bought')     = XEC.L(I,D) + XEE.L(I,D);
CPECRep(I,'Cache_Hit') = sum(K, REH.L(I,K,D)) + sum(E, DEH.L(I,E,D));
CPECRep(I,'Cache_Miss')= sum(K, REM.L(I,K,D)) + sum(E, DEM.L(I,E,D));
CPECRep(I,'Hit_Ratio') = (sum(K, REH.L(I,K,D)) + sum(E, DEH.L(I,E,D)))/(XEC.L(I,D) + XEE.L(I,D));
CPECRep(I,'RTT(ms)')    = sum(K,REH.L(I,K,D))/(XEC.L(I,D)+XEE.L(I,D)) * TI(I,'RTTC') +
     sum(E,DEH.L(I,E,D))/(XEC.L(I,D)+XEE.L(I,D)) * TI(I,'RTTE') +
     (1 (sum(K,REH.L(I,K,D))+sum(E,DEH.L(I,E,D))) /(XEC.L(I,D) + XEE.L(I,D))) * TI(I,'RTTO');
CPECRep(I,'PT(ms)')     = sum(K,REH.L(I,K,D))/(XEC.L(I,D)+XEE.L(I,D)) * TI(I,'PTC') +
     sum(E,DEH.L(I,E,D))/(XEC.L(I,D)+XEE.L(I,D)) * TI(I,'PTE') +
     (1 (sum(K,REH.L(I,K,D))+sum(E,DEH.L(I,E,D))) /(XEC.L(I,D) + XEE.L(I,D)))*TI(I,'PTO');
Display CPECRep;);
```

```
Parameter CloudRep 'ICE Cloud Summary Report';
Loop (D,
CloudRep(K,'Capacity')   = CACC(K,'CA');
CloudRep(K,'Demand')     = sum(I,REH.L(I,K,D))+sum(I,REM.L(I,K,D));
CloudRep(K,'Bought')     = EBC.L(K);
CloudRep(K,'Cache_Hit') = ECH.L(K,D);
CloudRep(K,'Cache_Miss')= ECM.L(K,D);
CloudRep(K,'Unused')     = ECN.L(K,D);
CloudRep(K,'Costs(k)')   = (EBC.L(K)*CACC(K,'TC') + ECN.L(K,D)*CACC(K,'HC') +
     ECH.L(K,D)*CACC(K,'CC'))*SP(D);
CloudRep(K,'Hit_Ratio') = (ECH.L(K,D)+1)/(ECH.L(K,D) + ECM.L(K,D) + 1);
Display CloudRep;);

Parameter EdgeRep 'ICE Edge Computing Summary Report';
Loop (D,
EdgeRep(E,'Capacity')   = CAEC(E,'CA');
EdgeRep(E,'Demand')     = sum(I, LD(I,D)*EUD(I,E));
EdgeRep(E,'Bought')     = EBT.L(E);
EdgeRep(E,'Cache_Hit') = EEH.L(E,D);
EdgeRep(E,'Cache_Miss')= EEM.L(E,D);
EdgeRep(E,'Unused')     = EEN.L(E,D);
EdgeRep(E,'Costs(k)')   = (EBT.L(E)*CAEC(E,'TC') + EEN.L(E,D)*CAEC(E,'HC') +
     EEH.L(E,D)*CAEC(E,'SC'))*SP(D);
EdgeRep(E,'Hit_Ratio') = (EEH.L(E,D)+1)/(EEH.L(E,D) + EEM.L(E,D) + 1);
Display EdgeRep;);

Display ZT.L, XT.L, ZC.L, XC.L, ZE.L, XEC.L, XEE.L;
```

BIBLIOGRAPHY

[1] Cisco, *Cisco Visual Networking Index: Forecast and Trends, 2017–2022*, tech. rep., 2019.

[2] Q. Fan, H. Yin, G. Min, P. Yang, Y. Luo, Y. Lyu et al., *Video delivery networks: Challenges, solutions and future directions*, 2018.

[3] M. Satyanarayanan, *The emergence of edge computing*, January, 2017. 10.1109/MC.2017.9.

[4] European Telecommunications Standards Institute (ETSI), *Mobile Edge Computing Introductory Technical White Paper*, tech. rep., Sep, 2014.

[5] Y. Hu, M. Patel, D. Sabella, N. Sprecher and V. Young, *Mobile Edge Computing: A key technology towards 5G*, Tech. Rep. 11, European Telecommunications Standards Institute (ETSI), Sep, 2015.

[6] R. Buyya, M. Pathan and A. Vakali, eds., *Content Delivery Networks*. Springer, 2008.

[7] H. Hu, Y. Wen, T.-S. Chua, J. Huang, W. Zhu and X. Li, *Joint Content Replication and Request Routing for Social Video Distribution Over Cloud CDN: A Community Clustering Method*, July, 2016. 10.1109/TCSVT.2015.2455712.

[8] Wikipedia, *Cloud computing*, *https://en.wikipedia.org/wiki/Cloud˙computing* (2019) .

[9] Amazon, *Amazon cloudfront*, *https://aws.amazon.com/cloudfront/* (2019) .

[10] Google, *Google Cloud CDN*, *https://cloud.google.com/cdn/* (April, 2019) .

[11] C. Li, J. Bai and J. H. Tang, *Joint optimization of data placement and scheduling for improving user experience in edge computing*, Nov, 2018. 10.1016/j.jpdc.2018.11.006.

[12] E. Gourdin, P. Maillé, G. Simon and B. Tuffin, *The economics of cdns and their impact on service fairness*, January, 2017.

[13] I. Benkacem, T. Taleb, M. Bagaa and H. Flinck, *Optimal vnfs placement in cdn slicing over multi-cloud environment*, 2018. 10.1109/JSAC.2018.2815441.

[14] T.-W. Um, H. Lee, W. Ryu and J. K. Choi, *Dynamic resource allocation and scheduling for cloud-based virtual content delivery networks*, 2014. 10.4218/etrij.14.2113.0085.

[15] L. Yala, P. A. Frangoudis and A. Ksentini, *Qoe-aware computing resource allocation for cdn-as-a-service provision*, in *2016 IEEE Global Communications Conference (GLOBECOM)*, IEEE, December, 2016.

[16] Z. Sharmin, A. A. Maruf and M. A. Razzaque, *Quality of experience- aware resource allocation for video content distribution to telco-cloud users*, in *2018 International Conference on Innovation in Engineering and Technology (ICIET)*, no. 1, IEEE, December, 2018.

[17] A. A. Haghighi, S. S. Heydari and S. Shahbazpanahi, *Dynamic qos-aware resource assignment in cloud-based content-delivery networks*, December, 2017. 10.1109/ACCESS.2017.2782776.

[18] Z. Zheng and Z. Zheng, *Towards an improved heuristic genetic algorithm for static content delivery in cloud storage*, 2018. 10.1016/j.compeleceng.2017.06.011.

[19] L. Yala, P. A. Frangoudis, G. Lucarelli and A. Ksentini, *Cost and availability aware resource allocation and virtual function placement for cdnaas provision*, 2018. 10.1109/TNSM.2018.2874524.

[20] S. Iturriaga, G. Goñi, S. Nesmachnow, B. Dorronsoro and A. Tchernykh, *Cost and qos optimization of cloud-based content distribution networks using evolutionary algorithms*, in *High Performance Computing*, vol. 979, Springer, Cham, March, 2019.

[21] P. Mach and Z. Becvar, *Mobile edge computing: A survey on architecture and computation offloading*, 2017. 10.1109/COMST.2017.2682318.

[22] D. T. Nguyen, L. B. Le and V. Bhargava, *Smart resource allocation for mobile edge computing: A deep reinforcement learning approach*, 2018. 10.1109/TCC.2018.2844379.

[23] X. Chen, L. Jiao, W. Li and X. Fu, *Efficient multi-user computation offloading for mobile-edge cloud computing*, 2015. 10.1109/TNET.2015.2487344.

[24] A. Samanta and Z. Chang, *Adaptive service offloading for revenue maximization in mobile edge computing with delay-constraint*, 2019. 10.1109/JIOT.2019.2892398.

[25] M. Chen and Y. Hao, *Task offloading for mobile edge computing in software defined ultra-dense network*, 2018. 10.1109/JSAC.2018.2815360.

[26] T. X. Tran and D. Pompili, *Joint task offloading and resource allocation for multi-server mobile-edge computing networks*, January, 2019. 10.1109/TVT.2018.2881191.

[27] C. Li, J. Tang and Y. Luo, *Dynamic multi-user computation offloading for wireless powered mobile edge computing*, January, 2019. 10.1016/j.jnca.2019.01.020.

[28] P. Yi, H. Ding and B. Ramamurthy, *Budget-minimized resource allocation and task scheduling in distributed grid/clouds*, in *2013 22nd International Conference on Computer Communication and Networks (ICCCN)*, IEEE, July, 2013.

[29] Y. Shao, C. Li, Z. Fu, J. Leyue and L. Youlong, *Cost-effective replication management and scheduling in edge computing*, January, 2019. 10.1016/j.jnca.2019.01.001.

[30] M. Meskovic and M. Kos, *Optimization of chunk scheduling algorithm in hybrid cdn-p2p live video streaming*, Sep, 2018. 10.1080/03772063.2017.1369911.

[31] V. Scoca, A. Aral, I. Brandic, R. D. Nicola and R. B. Uriarte, *Scheduling latency-sensitive applications in edge computing*, in *8th International Conference on Cloud Computing and Services Science*, 2018.

[32] G. N. Reddy and S. P. Kumar, *Maco-mots: Modified ant colony optimization for multi objective task scheduling in cloud environment*, January, 2019. 10.5815/ijisa.2019.01.08.

[33] K. Naik, G. M. Gandhi and S. H. Patil, *Multiobjective virtual machine selection for task scheduling in cloud computing*, in *Advances in Intelligent Systems and Computing*, vol. 798, Springer, Singapore, August, 2018.

[34] C. Wang, Z. Lu, Z. Wu, J. Wu and S. Huang, *Optimizing multi-cloud cdn deployment and scheduling strategies using big data analysis*, in *2017 IEEE International Conference on Services Computing (SCC)*, IEEE, June, 2017.

[35] S. Kang, B. Veeravalli and K. M. M. Aung, *Dynamic scheduling strategy with efficient node availability prediction for handling divisible loads in multi-cloud systems*, March, 2018. 10.1016/j.jpdc.2017.10.006.

[36] A. Moreira, J. Moreira, D. Sadok, A. Callado, M. Rodrigues, M. Neves et al., *Cost-effective replication management and scheduling in edge computing*, in *Proceedings of the 2011 Winter Simulation Conference*, 2011.

[37] N. Kamiyama and Y. Hosokawa, *Optimally designing virtualized cdn maximizing profit of content providers*, in *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2019.

[38] R. Viola, A. Martin, M. Zorrilla and J. Montalbán, *Mec proxy for efficient cache and reliable multi-cdn video distribution*, in *2018 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, IEEE, June, 2018.

[39] T. Dreibholz, S. Mazumdar, F. Zahid, A. Taherkordi and E. G. Gran, *Mobile edge as part of the multi-cloud ecosystem: A performance study*, in *27th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, vol. 979, IEEE, Feb, 2019.

[40] J. Wang, L. Zhao, J. Liu and N. Kato, *Smart resource allocation for mobile edge computing: A deep reinforcement learning approach*, 2019. 10.1109/TETC.2019.2902661.