Czech University of Life Sciences
Prague

Faculty of Economics and Management
Department of Management

Bachelor Thesis

Distributed Networking with a Cost-Effective Solution

Prepared by: Hardik Gondaliya

Thesis supervisor: Ing. Tomáš Vokoun

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

BACHELOR THESIS ASSIGNMENT

Hardik Rameshbhai Gondaliya

Informatics

Thesis title

Distributed Networking with cost effective solution

Objectives of thesis

The goal of this thesis is to analyze precisely that presently what are the risk challenges involved in networking when it comes to data security and budget. Current challenges are speech for the routing when it comes to other small industries of the department managing this. It Will include analyzing the small industries when it comes to earth the distributed M of and routing principles by considering security and cost-effective solution.

Specific Objectives

• To review what are the current practices used for the network, write routing, and how effective they are.

• To review what are the existing solutions and challenges faced by the network routing and budget analysis

• To propose a cost-effective analysis that can be helpful for networking.

Methodology

The procedure of gathering data for this study is to acquire details about the analysis of networking hazards, current methods of network routing, problems that have already been solved, and suggested cost-effective routing analysis. Both primary and secondary data sources will be used in the data collection techniques.

Primary data:

• Conduct surveys among small businesses and departments to learn more about their present network routing procedures, any difficulties they are currently facing, and how they view security and cost-effectiveness.

Secondary data:

• Review of the Literature: Conduct a thorough analysis of the current research, academic papers, business reports, and publications pertinent to the subject of networking hazards, current network routing procedures, difficulties, and cost-effective solutions.

• A case study with a focus on security and cost-effectiveness, examine case studies of small businesses or departments that have used network routing systems.

The proposed extent of the thesis

40-50 pages

Keywords

Network Security, Challenges, Technologies, Cost effective networking

---

Recommended information sources

MCMILLAN, Troy; EBRARY, INC. *Cisco networking essentials : e-book.* Indianapolis, Ind.: John Wiley & Sons, Inc., 2012. ISBN 978-1-118-09759-5.

ODOM, Wendell; HOGG, Scott. *CCNA Routing and Switching : official cert guide. ICND2 200-105.* Indianapolis, IN: Cisco Press, 2017. ISBN 978-1-58720-579-8.

Expected date of thesis defence

2023/24 SS – PEF

The Bachelor Thesis Supervisor

Ing. Tomáš Vokoun

Supervising department

Department of Information Technologies

Electronic approval: 19. 10. 2023

doc. Ing. Jiří Vaněk, Ph.D.

Head of department

Electronic approval: 3. 11. 2023

doc. Ing. Tomáš Šubrt, Ph.D.

Dean

Prague on 30. 11. 2023

Declaration

I declare that I have worked on my bachelor thesis titled " **Distributed Networking with a Cost-Effective Solution**" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the bachelor thesis, I declare that the thesis does not break any copyrights.

In Prague on 30th Nov 2023

Acknowledgement

I am deeply grateful for the support and encouragement I have received throughout my academic journey, which culminates in this thesis.

First and foremost, my heartfelt thanks go to my parents. Your unwavering belief in me, your sacrifices, and your endless love have been the bedrock of my achievements. I am forever indebted to you for your immeasurable support.

I would also like to extend my sincere gratitude to my thesis supervisor, whose guidance and expertise have been invaluable in shaping both this research and my overall academic growth. Your mentorship has been a guiding light throughout this journey.

My friends deserve special mention for their constant encouragement and companionship. You have been more than friends; you've been partners in this journey, providing support during challenging times and celebrating the victories, big and small.

I also wish to acknowledge the countless individuals who have assisted me, directly or indirectly, in the completion of this thesis. Your contributions, though they may seem small, have left a significant impact on my work and me.

This journey would not have been possible without each of you. Thank you for being part of my story and for helping me turn my aspirations into reality.

Abstract

This Study elucidates the challenges of broadcasting in networking, spotlighting the Broadcast Storm Problem (BSP) and its repercussions, particularly in Content Delivery Networks (CDN). Addressing redundancy, contention, and collision issues, the chapter examines broadcasting algorithms, such as simple flooding and probabilistic broadcasting, discussing their advantages and disadvantages. In the context of CDN, the text proposes modifications to these algorithms and introduces a clustered network structure with surrogates to alleviate BSP. Emphasizing the importance of reducing replication in CDN, the chapter underscores the potential efficacy of distance-aware and counter-based broadcasting approaches in mitigating BSP while enhancing reachability. The research focuses on optimizing Content Delivery Networks (CDN) to address flash crowd challenges by deploying replicas strategically using a population-based clustering algorithm. By modifying the K-Means algorithm based on population thresholds, the study aims to minimize deployment overhead and resource wastage. The algorithm redirects requests from underpopulated clusters to nearby surrogates, optimizing server utilization. Performance evaluation indicates comparable deployment costs with improved server efficiency in the population-based approach. This integration of distributed networking principles and CDN exemplifies a forward-looking strategy, promising cost-effectiveness, and performance excellence in the evolving digital landscape.

Keywords: Distributed Networking, Content Delivery Networks (CDN), Flash Crowd Challenges, Replica Deployment, Population-Based Clustering, K-Means Algorithm Modification, Server Utilization Factor, Deployment Cost Optimization, Resource Efficiency, Request Distribution

Distribuované sítě s nákladově efektivním řešením.

Abstrakt

Tato studie osvětluje výzvy v oblasti vysílání v síťových technologiích, zaměřuje se na problém broadcastové bouře (BSP) a její následky, zejména v obsahových distribučních sítích (CDN). Adresuje otázky redundance, soutěže a kolizí a zkoumá broadcastové algoritmy, jako jsou jednoduché rozšiřování a pravděpodobnostní vysílání, diskutuje o jejich výhodách a nevýhodách. V kontextu CDN navrhuje úpravy těchto algoritmů a představuje seskupenou síťovou strukturu se surrogate, aby zmírnilo BSP. S důrazem na důležitost snižování replikace v CDN zdůrazňuje kapitola potenciální účinnost přístupů ke vzdálenosti a protiopatření při broadcastingu při zlepšování dosažitelnosti. Výzkum se zaměřuje na optimalizaci Content Delivery Networks (CDN) k řešení problémů spojených s náhlým nárůstem návštěvnosti prostřednictvím strategického nasazení replik pomocí algoritmu shlukování založeného na populaci. Změnou algoritmu K-Means na základě prahů populace studie má za cíl minimalizovat náklady na nasazení a plýtvání zdroji. Algoritmus přesměrovává požadavky z málo obydlených shluků na blízké surrogáty, optimalizuje využití serveru. Hodnocení výkonu ukazuje srovnatelné náklady na nasazení s zlepšenou účinností serveru v přístupu založeném na populaci. Tato integrace principů distribuovaných sítí a CDN představuje strategii směřující vpřed, slibující nákladovou efektivitu a excelenci výkonu v se měnícím digitálním prostředí.

Klíčová slova: Distribuované sítě, Sítě pro distribuci obsahu (CDN), Výzvy náhlého nárůstu návštěvnosti, Nasazení replik, Shlukování založené na populaci, Úprava algoritmu K-Means, Faktor využití serveru, Optimalizace nákladů na nasazení, Efektivita zdrojů, Distribuce požadavků

Table of Contents

1. OBJECTIVES AND METHODOLOGY

1.1. OVERVIEW OF CONTENT DELIVERY NETWORK (CDN)

When it comes to managing massive amounts of internet traffic, the Content Delivery Network (CDN) is the foundation of the Internet. The vast proliferation of Internet multimedia applications has led to the massive generation of content, which CDN aims to govern. As a result, CDN has gained immense popularity. Currently, while we browse any news website, watch a YouTube video, shop online, or update a post on social media, we are all engaging with CDN, whether we realise it or not. A single server being the target of a large number of client requests created bottlenecks, which CDN was designed to avoid. CDN was first made available by Akamai in 1998 [1].

A content delivery network (CDN) is a group of geographically dispersed servers that repeat material from the origin server. We refer to these clone servers as surrogates or edge servers. To reduce user latency and bandwidth usage, the surrogates are strategically placed adjacent to the client across the worldwide network. The surrogate that is closest to the client in terms of distance receives the user request. The fundamental benefits of this structure are that it minimizes the traffic strain on the origin server and cuts down on the Round Trip Time (RTT) of the request. Since the majority of social media sites, like Facebook, Twitter, and YouTube, as well as e-commerce sites, handle their massive amounts of data via CDNs, the adoption of CDNs for real-world applications is growing quickly. The distinction between the traditional and CDN approaches is depicted in Figure 1.1.

Mobile operators, media and Internet ad firms, data centers, online music shops, ISPs, and others are among the early adopters of CDN [2]. Adopting CDN is beneficial for a number of reasons, some of them are as follows:

- **Faster content delivery:** In order to reduce latency and packet loss, CDN places surrogates as close to the client as feasible. Because there are less network jitters and spikes during streaming, the user's experience is improved.

- **Reduced server load:** The origin server experiences less strain when user requests are routed to the closest server, and as more surrogates get the requests, more bandwidth is available on the servers.

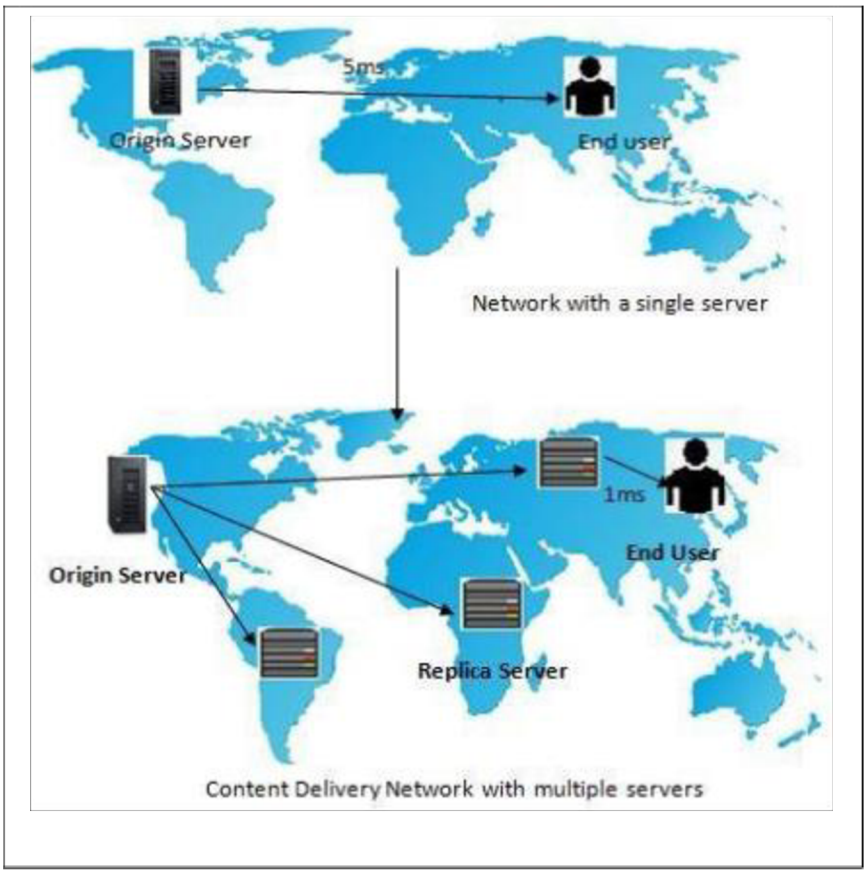


Figure 1.1 Content delivery network

- **100% availability and fail-over:** Thanks to the fact that CDN distributes all content from the origin server to multiple surrogates, it offers nearly 100% content availability. The material can be accessible from another server even in the event of a server failure.

But then CDN also has some limitations:

- **Determine the best server location:** Compared to the number of clients, the number of surrogates that must be used must be extremely small. Therefore, the primary concern is where to locate the servers in order to minimize packet loss and user latency.

- **Maintenance and support:** It is the responsibility of the CDN provider to make sure that content is updated constantly to ensure data consistency throughout the network. Considering also that the majority of businesses use services from outside CDN providers. Dependability and the availability of support are constant issues.
- **Cost:** The largest obstacle to CDN adoption is the high setup and maintenance costs. Using CDN for localized websites—where the majority of users live nearby—is an additional load because it rarely improves performance and rarely makes it worse.

1.1.1. History of CDN

In networking, using a proxy server is a highly widespread and outdated technology. There are several uses for different kinds of proxies. The client gets beyond firewall limitations by using forward proxies to access websites that are prohibited. The server uses reverse proxies to balance traffic and ensure high availability. A website's reverse proxy could be powered by multiple servers. The client request is received by the reverse proxy, which then routes it to one of the web servers. The clients are not given a clear picture of the full procedure. Content delivery networks employ the ad hoc notion of the reverse proxy.

CDN was first presented by Akamai as an MIT research project in 1998. The CDN was developed to handle Slashdot, sometimes known as flash crowds, which are sudden, large spikes in network traffic [3]. SpiceJet's summer sale serves as a straightforward illustration of flash crowd. As soon as an advertisement appears on radio, TV, and various newspapers, a large number of people attempt to access the website. However, the majority of the time there is a negative outcome since access to that particular page is blocked. This is due to the unanticipated increase in client requests creating a bottleneck for the unprepared website, which may cause the server to lag or even go down for a while. Hence, content delivery networks (CDNs) offer the greatest option as they replicate material from the origin server to several surrogates that are carefully positioned in multiple locations close to the clients.

Commercial CDN of the "90s has experienced tremendous development and technological modifications brought about by market forces, making it a mass-market technology. Three generations can be distinguished from the many evolutionary stages found in CDN [4].

- First generation CDN (1998-2001): Also referred to as static CDN because it handles downloaded files and static information. Requests for static content, such as photos or JavaScript files, are always routed towards the edges rather than the origin server by this content delivery network (CDN). However, CDN was mostly utilized by the corporate sector and was quite expensive in its early stages.
- Second generation CDN (2001-2010): Since it serves both static and dynamic content, the CDN used during this time is referred to as a dynamic CDN. Dynamic CDN covered availability in addition to performance, whereas static CDN simply addressed performance. This CDN's main clientele was the corporate sector.
- Third generation CDN (2010 onwards): Rich media and mobile data are added to the third generation CDN, also known as the multi-purpose CDN. The performance, availability, and security of the current CDN are its top priorities. Nowadays, nearly all websites use content delivery networks (CDNs) due to the considerable decrease in CDN service prices.

Following are some remarkable events in the field of CDN [5]

- Akamai released the first CDN for sale in 1998.
- Large-scale Internet service providers (ISPs) began building their own unique CDNs by 2001.
- It was predicted that CDN revenue would increase by 40% in 2005 due to the growth of streaming video platforms and Internet radio.
- Amazon introduced their own CDN in 2008.
- AT&T unveiled its cloud-based content delivery network (CDN) in 2011, enabling content to travel from 38 data centers across the globe over the network.
- Akamai's stock revenue skyrocketed to $345.32 million in 2012.
- YouTube, the leading website for creating videos, has deployed copies in more than 75 countries to serve its 1 billion users [6].

1.1.2. Research issues in CDN

With the development in technology, CDN has become a strong application delivery platform. However, there's always room for improvement, which expands the field of study in this field. The following list includes a few of the important issues:

- **Replica server placement:**

Performance, or how quickly and efficiently data can be sent to the client, is CDN's main priority. Data must be kept near to the user in order to do that. Data is replicated to many surrogates via CDN. The key query at hand is how many servers is the right amount to have and where to put those surrogates in order to maximize CDN performance. Numerous methods were put up by various researchers, and as the replica server problem is NP complete [7], a large body of study has been made possible.

- **Content selection and distribution**

The distribution of data to the surrogates is the next problem after servers are established. In actuality, CDNs are either push or pull based on how material is distributed. Pull-CDN retrieves the material from the origin as requests are received, while push-CDN moves the content forward and towards the edges [8]. Which data is cloned in which surrogate is the next issue. Full-site replication uses 100% replication across all servers, requiring a large amount of memory and frequent updates. Only embedded items are replicated to the edges via partial-site replication, and researchers have proposed several methods for determining which embedded objects are appropriate for a given server [6, 7].

- **Cache organization**

To reduce the time, it takes for a page to load, content caching distributes the content among multiple points of presence (POP) across the network rather than storing it in one place. The caching method and cache update are integrated in content caching in CDNs. When a cache misses, a surrogate notifies its neighbors. Only the request is forwarded to the origin if none of them responds positively. In this manner, the bandwidth used to reach the origin server is decreased. In order to guarantee current information serving, CDNs need periodically refresh all cached content.

Research is ongoing to maximize the outcome, and several caching and updating strategies for CDN have been proposed.

- **Routing mechanism**

Another CDN thrust area is the routing method. Since the main goal of CDN is to route client requests to the closest surrogate, request routing is crucial. Additionally, in cooperative pull-CDN, the surrogate's request is routed to all of its neighbors during a cache miss [9]. Even though there are a number of methods available to determine the optimal route for redirecting requests, work is still being done to improve the system.

1.2. DESIGN REQUIREMENTS

A global network of interconnected surrogates called content delivery networks (CDNs) is used to provide cached content to users where user proximity is the main consideration. The following list of distinct and crucial elements should be taken into account when building content delivery networks.

1.2.1. CDN system components

Content Delivery Networks utilize several surrogates dispersed throughout the network to duplicate the content. The three fundamental parts of a CDN are depicted in Figure 1.2.

- **Origin server:** The real material is saved on the origin server, also referred to as the content provider.
- **Surrogate servers:** Additionally referred to as edge servers or replica servers, surrogate servers are used by CDN providers. In order to relieve pressure on the original server and improve user experience by speeding up page loads, surrogates are used to copy the content of the original server. Every replica server often has several storage discs and a large amount of RAM.
- **End users:** Clients who submit queries to a certain website, which are routed to the surrogates and receive a response from CDN, are known as end users.

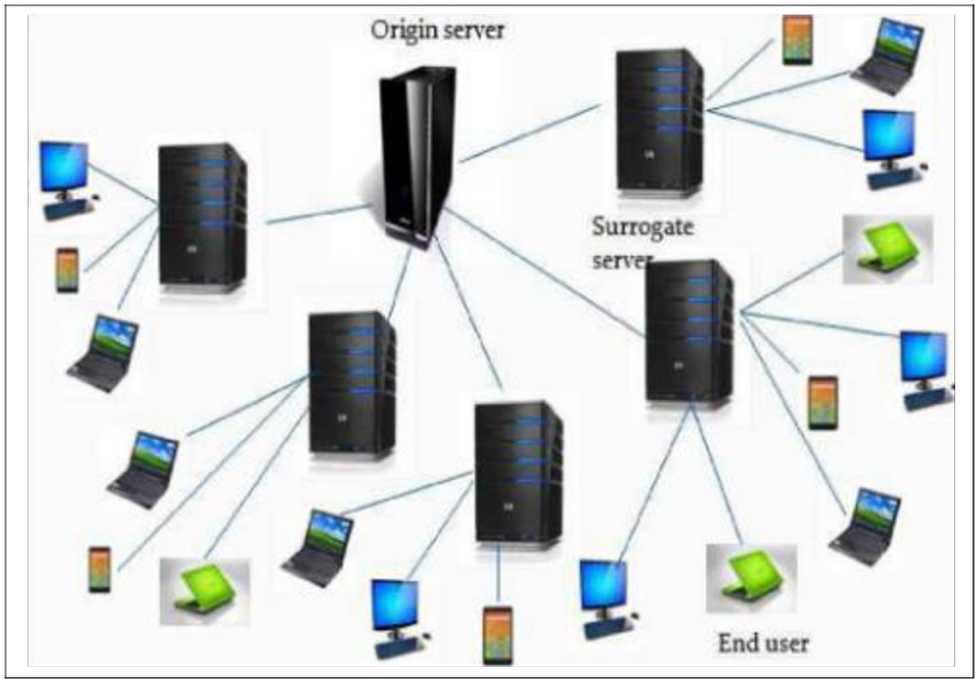


Figure 1.2 Content delivery network components

1.2.2. CDN architecture

The content provider keeps all of the website's data, and the surrogates, which are positioned in key spots, primarily duplicate static information because it loads slowly and doesn't need to be updated frequently. Figure 1.3 [10] depicts the CDN system architecture.

With reference to the Figure 1.3 the following steps are getting followed in CDN:

Step 1: When the user types www.abc.com, the client request is redirected to the DNS by the browser.

Step2: DNS replies with "abc," the IP address of the origin server.

Step 3: The user then asks the "abc" server for HTML.

Step 4: The HTML and the surrogate URL are returned by the server.

Step 5: The URL returns to the DNS once more.

Step 6: With the aid of a data collector, DNS resolves the URL and returns the IP address of the most accurate duplicate.

Step 7: Now, the request is sent to that particular server. Step 9 is when the server provides the requested content straight to the user if it has it.

Step 8: In the event of a cache miss, the request is routed back to the source; the material is retrieved, stored for later use, and then returned to the client (step 9).

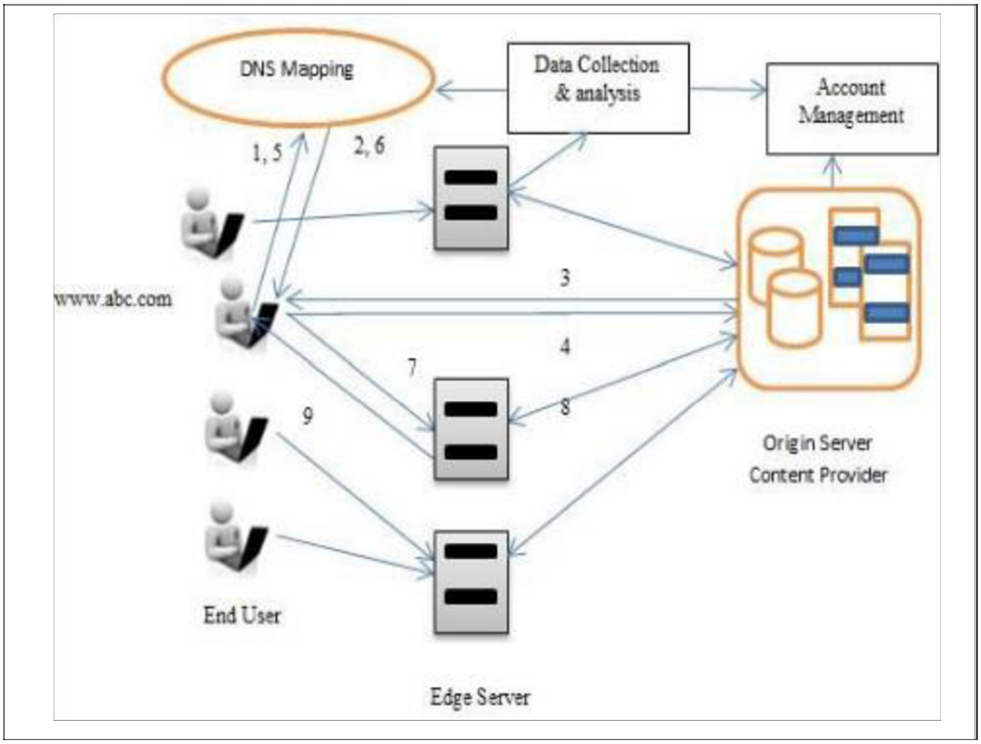


Figure 1.3 CDN system architecture

Reducing user latency and bandwidth usage to unload the origin is the main benefit of employing CDN technology. Aside from that, this CDN architecture also offers some other benefits.

- Downtime protection: The CDN infrastructure uses a significant quantity of networking and storage resources. It reduces the risk of a flash crowd while increasing the number of users with access. The same content is copied in multiple POPs, so even in the event of a surrogate failure, traffic is diverted to one or more alternative surrogates.

- Improved SEO: A website that uses a content delivery network (CDN) benefits from a faster page load time and can rank higher in search engine results pages (SERPs). Sites with fast speed due to content delivery network (CDN) will always be at the top of the list because Google utilizes website speed as a metric for ranking algorithms.

1.3. OBJECTIVES COVERED IN THE THESIS

The main goals of CDN's development were to minimize user latency, or the time it took for a page to load, and to reduce bandwidth usage in order to address the flash crowd issue. To do this, CDN establishes several replicas in various locations close to the users. The critical choices of how many surrogates to deploy and which approach to use in order to identify the optimal deployment locations will have a significant impact on CDN efficiency. While having an excessive number of copies can guarantee 100% data availability, it will also increase maintenance costs and storage requirements because each replica needs a large amount of RAM and storage space. Furthermore, it will take time for DNS to resolve the IPs, resulting in significant packet loss and latency. Determining whether to pick specific content or reproduce the complete content to all surrogates is another difficult task. It is necessary to implement the proper strategy while choosing material. The primary objective is to update the material in every surrogate around the world, regardless of the replication procedure. Another scenario is that a large number of request and acknowledgement messages are created within the CDN in the event of a cache miss under partial replication conditions, which may lead to a broadcast storm problem (BSP). This BSP will benefit from content updates in both the original server and any surrogates. Thus, this research focuses on proposing a density-based replica placement method for CDN, taking into account all these difficult considerations. The following goals have been taken into account:

- To research significant bandwidth savings issues and possibilities
- Investigation of scaling the distribution of content to the network edge and possibility of Resilient Mechanism.
- Investigation and improvement of broadcasting algorithm for content distribution.
- Improvement of the Minimized impact of flash crowd events w.r.t distributed content delivery.

1.4. CONTRIBUTION OF THE THESIS

Several efforts have been made to create an effective algorithm for placing replica servers. The first research focuses on optimizing bandwidth usage by using several surrogates spread over the network to offload the web server. In order to determine the area for improvement through a thorough comparative investigation of current server placement algorithms, the bandwidth savings possibilities study also includes replica server placement techniques. The study also looks at how content is distributed at network edges and attempts to determine the best broadcasting technique to use whenever there is a change in information, whether it comes from the source or one of the surrogates. This thesis' primary goal is to create a density-aware replica placement method that will outperform the current schemes in terms of server utilization and performance. This thesis has proposed parameters such as the utilization factor for surrogates and the Population Threshold for clusters in order to develop the algorithm.

1.4.1. Investigating bandwidth savings issues for CDN

The quantity of data flow between the server and the clients is restricted by bandwidth in the context of web hosting. Additionally, the fees that web hosting companies charge for this data transmission are known as bandwidth costs. Therefore, the same server's content will load for each HTTP request; the more hits received, the higher the bandwidth usage. A content delivery network (CDN) can handle this situation by replicating the original content to several surrogates and using the closest servers to serve HTTP requests. This reduces the amount of bandwidth used and the transmission path. Chapter 2 discusses various methods of optimizing bandwidth cost in CDNs, including surrogate placement, content distribution, and request routing.

1.4.2. Investigating appropriate broadcasting algorithm for CDN

A normal occurrence in networking and CDN is broadcasting. To avoid inconsistent data, the CDN must notify the other surrogates of any changes made to the original server. In a similar vein, any surrogate may have updates that need to be shared with the origin server and other surrogates. The simplest method of disseminating information is blind flooding, while broadcasting is the easiest.

However, this straightforward method will cause packet loss, contention, and collisions—a problem known as a broadcast storm (BSP). In order to prevent BSP, this thesis thoroughly examines the broadcasting algorithms employed in wireless sensor networks. In Chapter 3, it suggests a suitable counter-based probabilistic method for CDN.

1.4.3. Introducing population threshold for clusters

The most basic type of clustering is means. However, the clusters produced by the -Means algorithm might only have a small number of nodes, and adding surrogates to these clusters will just make maintaining servers, logs, and routing tables more difficult. Through the design of population-based clustering for replica server placement, this research attempts to address this problem. In order to achieve this, a new parameter known as population threshold has been added. It specifies the bare minimum of nodes required for a cluster to have its own server.

1.4.4. Defining utilization factor for surrogate

The effectiveness of the population threshold-based replica placement algorithm has been assessed using a parameter utilization factor. A surrogate's utilization factor indicates what proportion of the overall population it serves. This component determines whether or not a surrogate can fulfil requests from outside of its cluster. It also determines the servers' average utilization. The surrogate can serve more if the average utilization is lower than the individual utilization; if not, the server won't receive any more requests.

1.4.5. Designing Population-based clustering algorithm Load

This thesis presents the construction of a population-based clustering algorithm for replica server placement, utilizing a population threshold and utilization factor. Using the - algorithm, the algorithm creates a number of clusters from a given number of nodes. To determine how many clusters have a population below the threshold, a population threshold is applied. Requests made by these recognized clusters will be routed to adjacent servers rather than being handled by their own surrogates. These extra requests might be fulfilled by servers with utilization factors below average. Chapter 4 provides a description of the entire algorithm.

1.4.6. Density-aware replica placement algorithm incorporating strength of traffic.

Chapter 5 presents the final version of the replica server placement algorithm, which is called the density-aware replica placement algorithm. This algorithm now includes a new metric known as the strength of traffic load. The evaluation of the traffic load is as () Where are the link speed, packet length, and packet arrival rate? The traffic load must be less than 1 in order for the system to be stable. Therefore, in accordance with this technique, a server should first examine its utilization and traffic load before fulfilling a new request.

2. LITERATURE REVIEW

Distributed systems have completely changed the way that data and applications are handled, delivered, and accessed in the computing industry. The idea of decentralization, which replaces the conventional model of a single, monolithic server with a network of connected, distributed nodes cooperating, is at the core of this shift. The advent of material Delivery Networks (CDNs) is one of the most notable and revolutionary examples of distributed systems; they have completely changed how digital material and services are provided to users around the globe.

Distributed systems are defined by their capacity to perform a variety of computational tasks by combining the power of numerous linked devices or servers, which are frequently dispersed over different geographic areas. Numerous benefits come with this strategy, such as increased performance, fault tolerance, scalability, and dependability. In this regard, content delivery networks (CDNs) emerge as a standout example of a distributed system application, completely changing the online content distribution scene.

A content delivery network (CDN) is simply a network of strategically placed servers and edge nodes spread geographically throughout the world. Its main goal is to optimize the distribution of digital material, such as films, web pages, and other applications that require a lot of data. material delivery networks (CDNs) minimize latency, speed up load times, and make effective use of network resources by distributing material closer to end users. The application of distributed systems theory to the rising problems of an increasingly networked and content-hungry digital world has led directly to this revolution in content distribution.

There is a synergistic relationship between distributed systems and CDNs, with the distributed system offering the underlying architecture and infrastructure that drives CDNs. CDNs make use of the distributed system's characteristics to guarantee fault tolerance, load balancing, and effective

content delivery. By doing this, they greatly improve user experience and give content creators a dependable and affordable way to reach a worldwide audience.

We will go deeper into the underlying ideas and technology as we begin this investigation of distributed systems with a particular emphasis on CDNs, demonstrating how they work together to meet the needs of a data-driven, globally networked society. By lowering latency and network congestion and improving security and scalability, the combination of CDNs and distributed systems shows how decentralized architecture might influence the future of the digital world.

A crucial point of intersection in the context of contemporary digital infrastructure is the link between distributed networks and content delivery networks, or CDNs. The distribution of digital material and services over the internet is made possible by the complimentary roles that these two technological ideas play in being closely interconnected. In order to fully understand this relationship, it is necessary to investigate the fundamental ideas behind each and how they work together to satisfy the ever-increasing needs of our internet-driven society.

In the global digital landscape, distributed networks signify a paradigm shift in the ways that data is accessed, stored, and communicated. They represent a radical break from the conventional, centralized form of network architecture, in which content was served to users by a single server or a few data centers. Rather, distributed networks use a decentralized strategy to spread content and resources over a large number of geographically separated locations. They frequently make use of an extensive network of servers, edge nodes, and other infrastructure parts.

CDNs become an essential component of distributed networks in this environment. CDNs are specialized systems made to strategically place online content and application caches throughout the world to optimize content delivery. Reduced latency, faster load times, and an improved user experience are the main objectives of CDNs. They accomplish this by ensuring that available bandwidth is used efficiently, reducing the round-trip time of data packets, and delivering content from the server that is physically nearest to the end user.

Given the issues faced by the exponential growth of digital material and the diversified, often geographically scattered user base, the synergy between distributed networks and CDNs becomes evident. Here, distributed networks act as the general infrastructure that supports content delivery

networks (CDNs). Together, CDNs, which are an essential part of the dispersed network ecology, enable effective content delivery.

In order to lessen the burden on origin servers and minimize the stress on a single point of failure, CDNs make use of the concepts of distributed networks. To do this, content is replicated over a large number of edge servers or nodes that are dispersed throughout different geographic areas. The content delivery network (CDN) shrewdly routes a user's request to the closest edge server, cutting down on latency and streamlining the delivery process.

CDNs also benefit from distributed networks' innate scalability. By adding new nodes to the dispersed network, CDNs may easily extend their edge server infrastructure in response to the growing demand for digital content and services. The mutually beneficial link between content delivery networks (CDNs) and distributed networks is demonstrated by their capacity to scale horizontally, adjust to changing traffic patterns, and meet the various needs of content producers.

Moreover, CDNs' distributed architecture closely adheres to redundancy and fault tolerance principles, making them essential for ensuring high availability and dependability in content delivery. CDNs can divert traffic to operational edge servers or other points of presence within the distributed network in the case of a server loss or network disturbance, guaranteeing continuous service delivery.

An extensive overview of the history used in the field of content delivery networks is provided in this chapter. Along with a review of the literature, it also covers a variety of CDN topics, including the positioning of replica servers, the delivery and selection of content, content caching, and request routing. The chapter also discusses the cost of bandwidth for web hosting and how a CDN might save costs. Since the primary focus of this PhD thesis is replica server placement, a survey is conducted in order to conduct a comparative analysis of the replica server methods in CDN.

2.1. CONTENT DELIVERY NETWORK

The main goal of the evolution of the content delivery network was to provide customers with 100% data availability and low latency. The content, especially static content, is replicated by CDN among several surrogates that are placed around the network. In response to the increasing popularity and expansion, a number of organizations have already included CDN into their

operations. For instance, the popular website YouTube uses surrogates located in 75 different countries to service its 1 billion users [6, 11].

In order to lessen the effects of the Slashdot effect, also known as the flash mob problem, which is a sudden spike in network traffic, CDN was introduced [3]. CDN was introduced to the market by Akamai in 1982 [1]. When using CDN, the material is cached on the origin server and then copied to all other surrogates that are stored around the network at various key places in an attempt to relieve pressure on the origin [10, 12]. The CDN and its constituent parts are displayed in Figure 2.1. Initially, the content provider was in charge of managing the dynamic material, and the CDN exclusively hosted static content, such as pictures, ads, and media clips [13]. Currently, dynamic content like multimedia apps, video on demand, and interactive streaming media are the main sources of CDN popularity [14]. Delivering material to clients with minimal latency and adequate quality of service is the main goal of CDN. Request routing, content selection, cache management, and surrogate placement are the technologies used to accomplish this goal. The efficiency of the CDN is assessed using a number of performance metrics. The CDN's parameters include things like latency, cache hit ratio, bandwidth, packet loss, and CDN usefulness [1, 15, 16, 17].

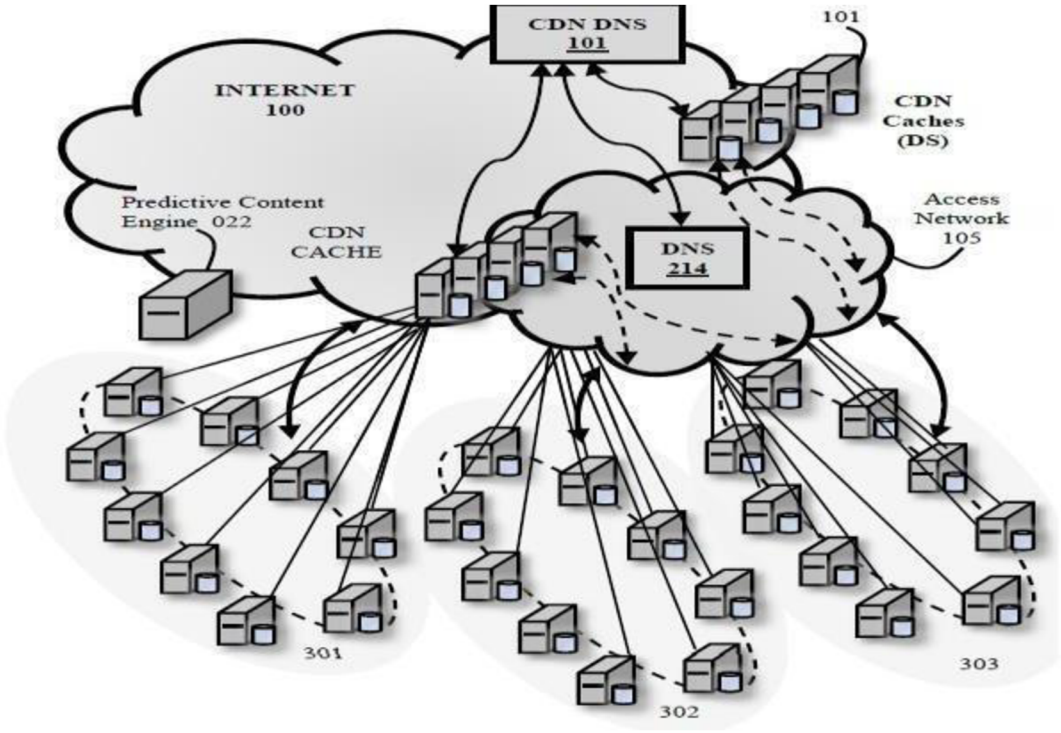


Figure 2.1 Content distribution network (CDN)

2.2. BANDWIDTH ISSUES AND POSSIBILITIES IN CDN

The quantity of data that may be exchanged between the server and the clients is actually limited by bandwidth in the context of web hosting. Additionally, the web hosting providers charge for this data transmission, which is known as bandwidth cost. When a client accesses any website without a content delivery network (CDN), all content loads directly from the origin server. Thus, the material is transferred from the same server and the HTTP requests are made to it for every client visit. An increasing number of website hits leads to a rise in data transfer, which raises the cost of transmission, or bandwidth. A number of performance metrics, including packet loss, deployment costs, server utilization, and user delay, can be used to assess bandwidth costs.

By putting itself in between the users and the website's hosting servers, a content delivery network (CDN) lowers bandwidth costs. To reduce latency, a CDN caches material from the origin server closer to the client. Requests now go to the closest surrogate rather than the web server, and content is loaded from there. This clearly shortens the transmission path and saves bandwidth costs. There are multiple methods in CDN to reduce data transmission rate. This is dependent on choices made regarding the number and location of surrogates, the outsourcing of material, the kind of content to be cached in each surrogate, and the method by which client requests are routed to the surrogate. This chapter focuses on the study and examination of various CDN components that can be used to address bandwidth difficulties, as stated in goal 1.

2.2.1. Replica server placement in CDN

Since the placement of replica servers greatly affects CDN efficiency, replica server placement is crucial to CDN establishment. Since replica server placement is an NP hard problem, there is no ideal solution [7]. Replica server issues were formerly classified as center placement issues. Based on graph theory, the -center problem and k-Hierarchically well Separated Tree (k-HST) offer the best solution for replica placement [8]. The Hierarchically well Separated Tree is a two-step process in which a random node is chosen at the first stage and is regarded as the parent node. Child partitions are made up of the remaining nodes within a certain radius of the parent node. The radius of the child division must be less than the radius of the parent partition. This process keeps going until the entire network is divided into a tree of partitions, with the parent node serving as

the root node and every other node acting as a leaf node. Even though these two methods yield the best results, their great computing complexity makes them ineffective in real-world situations.

Numerous heuristic approaches with relatively low computer cost and poor answers have been presented. Using a tree-based method and a dynamic programming algorithm, the network is viewed as a single, rooted tree made up of nodes [18]. In this case, the entire network T is split up into several smaller trees, and each little tree should include one replica. A node should not send requests to its sibling tree while using this strategy. Only the origin should ever be the target of the request. A greedy strategy chooses M sites out of N possible locations [19]. The first section analyses the costs related to each of the N sites, selecting the site with the lowest cost. The site with the second lowest cost is determined by analyzing the remaining N-1 locations in the subsequent iteration. This procedure is repeated M times. Sites are chosen at random using a random process, and the related costs are assessed. After ten iterations of this procedure, the most effective of these widely used methods—which rank all replicas in the top results—is chosen [19]. HotSpot is a website that displays the sites that generate the most traffic based on network load and traffic data [19]. HotZone is a two-step process wherein the Global Network Position (GNP) is used in the first phase to choose the network region where the surrogates would be placed. Subsequently, every node in every region is evaluated in order to identify possible locations for replica deployment [20]. Topology-informed replica placement, in which nodes are arranged according to their outdegree in descending order, was also discussed by M. Pathan et al. in [8]. Constrained Mirror Placement (CMP), one topology-informed technique, is introduced in [21] and is based on the optimization factor, Round Trip Time (RTT). Another topology-aware method called Max Fan Out chooses possible locations with the greatest number of outgoing edges [22]. The Flow Count Strategy was developed by Moises Rodrigues, who used an analyzer at each node to determine the amount of traffic that moves through them. Lastly, it selects nodes with the highest flow count, or nodes that produce the most traffic [23].

Evaluating the costs associated with individual locations becomes quite costly for a network with a large number of prospective sites. Instead, it would be more cost-effective to group nodes into clusters according to a predetermined standard. In order to determine the clients' approximate geographic locations, GeoIP clustering was first proposed [24]. Initially, the user log's IP addresses are taken out and used as input to the IP Geolocation service to determine the users' latitude and

longitude. Based on these geographic data, possible sites are sorted into groups using Fuzzy C-Means and subtractive clustering, and the optimal locations within the clusters are determined for replica deployment. A clustering-based method called NetClust has been proposed in [25]; it selects the replica position using the -Means clustering algorithm. The optimal matching of the clients and servers is carried out by the algorithm's second phase.

Either a single ISP or a multi-ISP strategy is used by the CDN provider [26]. Only a single ISP has a limited number of surrogates, and each heavily trafficked site often has one or two servers. Due to the servers' wide geographic coverage, this method results in higher user latency. Multiple ISPs set up multiple replica servers in various places. The waste of resources is apparent since fewer client inquiries are being attended to by the surrogates. The number of surrogates that can be deployed should be limited; otherwise, using too many servers can raise deployment costs and user latency [22].

2.2.1.1. Comparative study

N represents the entire number of possible sites, and M copies will be deployed among N locations, subject to the restriction N >> M. Popular replica placement algorithms (RPAs), which are given in Table 2.1 [27], are examined and contrasted in light of this claim.

Table 2.1: Comparison of Different RPAs

Approaches	**Computational Complexity**	**Optimization Factor**
Tree-Based	$O(N^3M^2)$	Tree path
Greedy	$O(N^2M)$	Minimum cost
Random	$O(NM)$	Random selection
HotSpot	$'O(N^2 + \min(NlogN$ $NM))$	Minimum Latency, Maximum Load

HotZone	$O(N.\max(\log N, M)$	Minimum Latency, Maximum load
Flow Count Strategy	$O(N^2M)$	Bandwidth Cross traffic
Max Fan Out	$O(N^2M)$	Network traffic
NetClust	$O(N+M)\,\lambda$	Minimum Latency and Deployment cost

The comparison demonstrates how extremely complex tree-based strategies are and how they can only be used in tree structures. Although the greedy approach produces results that are closer to ideal, it has a relatively high computational complexity [19]. The random technique does not yield an optimal solution, while having a relatively low complexity [19]. While the Flow Count method's complexity is comparable to that of the greedy method, it has a higher startup latency and overall network traffic [23]. However, it also greatly reduces the costly cross-trafficking of bandwidth that occurs between two independent systems. Because HotSpot is less sophisticated than greedy, it is more popular. Compared to other methods, NetClust's complexity is significantly reduced [25]. The number of iterations, which in turn depends on the number of clusters and the choice of beginning centroid, determines the complexity of this strategy. Therefore, choosing these two factors carefully lowers NetClust's complexity.

2.2.2. Content selection and delivery

The next issue is how content will be outsourced or replicated to every surrogate after the surrogates are deployed across the global network. Push- or pull-based approaches are used in content outsourcing [8, 28–33]. Section 3.3 provides a detailed examination of content distribution and outsourcing. Delivering accurate material to users with high availability and minimal response latency is the primary concern. Both complete and partial content deliveries are possible to the surrogates. The most straightforward method is "entire replication," in which every surrogate

receives an identical copy of all the content from the origin server. The main drawbacks of this strategy are the costs associated with storage and the regular updating of content. Partial replication is becoming more and more common due to its practical limitations, as it only duplicates embedded objects, allowing HTML pages to be read from the origin server [8, 9]. Figure 2.2 displays the content outsourcing techniques. Once more, the partial replication is divided into four categories: cluster, object, popularity, and empirical [9, 29, 30, 34]. The Top 10 strategy is another well-liked method for choosing content [35].

Table 2.2 provides a comparison of full- and partial-site replication.

Table 2.2 Comparison between Full-site and Partial-site Replication

Replication Strategy	**Advantages**	**Disadvantages**
Full-site replication	i. Easy implementation. ii.100% content availability.	i. huge storage space. ii. Frequent content update.
Partial-site Replication a. Empirical Based b. Object-Based c. Cluster-Based i. Per website- Based ii. Per URL-Based d. Popularity-Based	i. Occasional update in embedded data. ii. A reduced amount of storage space.	i. Latency may be larger than that of Full site.

2.2.3. Content caching

Caching methods and cache updates are combined to create content caching. In content delivery networks (CDNs), cache organization plays a critical role in the timely delivery of huge volumes of data to high-profile consumers worldwide. This is due to the frequency of content updates and the caching technique employed. There are two methods for caching content in a content delivery network (CDN): intra- and inter-cluster caching [8]. A variety of techniques, including query-based [36], digest-based [37], directory-based [38], hashing-based [39, 40], and semi-hashing-based [41, 42], can be used to carry out intra-cluster caching. Table 2.3 is a summary of all the caching strategies that were previously discussed.

When intra-cluster caching is unsuccessful, inter-cluster caching is carried out, and query-based clustering is the most effective method for doing so.

Table 2.3: Intra-Cluster Caching Techniques

Intra-cluster caching technique	Caching technique	Advantages/ Disadvantages
Query-based	When a cache miss occurs, the CDN server broadcasts a query to all other participating servers in the same cluster. It then waits for the last miss response from each surrogate before requesting an inter-cluster request.	Disadvantage: Significant delay query traffic.

Digest-based	Every surrogate keeps an updated digest of the content from all of its partner servers and notifies others when there are any changes to that content. The CDN server looks through its digest before rerouting each request in order to identify a specific collaborating surrogate.	Disadvantage: Huge update traffic
Directory-based	digest-based methodology in a centralized version. At a centralized server, the content digest is centrally maintained. Before rerouting any requests, all updates are sent to this centralized server, to which inquiries are also directed.	Disadvantages: Experiences bottleneck due to huge update and query traffic.
Hashing-based	Every collaborating surrogate maintains the same hashing function. Based on the URL, surrogate IP addresses, and hashing function of the content, a specific content delivery network (CDN) is assigned to host it. All queries are sent to this specific CDN server.	Advantage: Less implementation overhead. Disadvantage: Does not suit local requests.

Semi-hashing based	A certain CDN server utilizes a hash function to collaborate with other surrogates while using a tiny amount of its disc space to cache highly popular content.	Advantages: Less implementation overhead. Efficiency in high content sharing. Increased hit ratio.

For the end user to receive consistent and updated material, the cache update is crucial. As indicated in Table 2.4[43], there are several ways that content is updated, including periodic updates, update propagation, on-demand updates, and invalidation.

Table 2.4: Cache Update Techniques

Cache update schemes	Update mechanism	Disadvantages
Periodic update	Caches are regularly updated to guarantee that the content is current.	Unnecessary update traffic after each interval.
Update propagation	All surrogates receive the updated material whenever there is a change made to the origin server.	High update traffic.

On-demand update	The content is distributed according to its most recent update request.	Content does not get updated until requested. Back and forth traffic in between surrogate and origin server.
Invalidation	All caches receive an invalidation notification when the content is being updated at the origin, and their ability to access the updated content is restricted. Later, as needed, each cache must independently retrieve the updated content.	Inefficient in managing the content consistency.

2.2.4. Request routing

Request routing is a critical component of content delivery networks (CDNs) since it directs client requests to the proper server, avoiding congestion and resulting in significant reductions in response and download times [44]. Request routing is directly impacted by cache organization. When replication is complete, there is relatively little routing cost since there are fewer cache misses; when replication is partial, there are more misses, which causes requests to be rerouted. Algorithms and mechanisms for request routing make up request routing [45]. Request routing mechanism notifies the client of the edge server selection outcome after initially invoking the routing algorithm to choose an edge server in response to a request from the client. As shown in Table 2.5, request routing techniques in CDN are either adaptive or non-adaptive [46].

Numerous non-adaptive routing methods have been proposed by researchers in [40, 45, 47–51]. Round robin is the most basic non-adaptive technique, in which all client requests are sent to each

surrogate, and the load is then balanced between the servers. This technique takes into account the fact that all caches are capable of fulfilling any request and have comparable processing power. This method works great in a cluster network when every server is positioned in one area, but it is incongruous with a big distributed system where the surrogates are dispersed over different locations. In that case, it raises the cost of bandwidth as well as routing overhead. Other nonadaptive methods use a variety of variables to create their heuristics, such as the server load, the proportion of all client requests that are cached, the geographic locations of the clients, or a hash function based on the content URL, among others.

In [50, 52–55], a number of adaptive routing techniques are covered. Metrics including network proximity, client-server latency, bandwidth, and combinations of intra- and inter-AS distance and end-to-end latency are used to assess the state of the system as it is right now. For its extensive CDN, Akamai employs an extremely intricate adaptive routing system [56, 57].

Table 2.5: Request Routing Algorithms

Routing algorithms	**Routing technique**	**Advantages/ Disadvantages**
Adaptive	The selection of caches is contingent upon the state of the system at the time, which can be ascertained by assessing various factors such as server load, network congestion, etc.	Advantage: Able to change the behavior to cope with the situation and it can exhibit high system robustness. Disadvantage: Implementation is very complex.

Non-adaptive	Heuristics are used to choose the caches.	Advantage: Easy implementation. Disadvantage: Go well where the heuristics assumptions are met. But does not support system robustness.

The routing mechanism notifies clients about the replica servers that are chosen based on routing algorithms. The following criteria can be used to categorize routing mechanisms: Anycasting [60], CDN peering [7, 62], HTTP redirection [7, 60], URL rewriting [61], DNS-based request routing [7, 59], and global server load balancing [58].

2.3. APPLICATIONS OF CDN

Many industries, including advertising, mobile, media and entertainment, healthcare, education, online gaming, e-commerce, and government, employ CDN extensively. CNN and the BBC get their media from top CDN providers like LimeLight and Akamai [63]. Following its acquisition, Google began using its own CDN in addition to Akamai, replacing LimeLight CDN, which had previously been distributing the largest video-generating website, YouTube [63, 64]. Massive amounts of content are transferred over social networking sites like Facebook, Twitter, and WhatsApp, which employ S-CDN (Social content delivery network) at the moment. This network uses various techniques provided by Dropbox, FriendBox, CoDaaS, MetaCDN, Amazon S3, and other companies [65–70]. Telco CDNs, which are run by ISPs, are a new type of content delivery network that telecommunication service providers have introduced [71–73]. Table 2.6 presents a mapping between several CDN application areas and the surrogate placement techniques covered in section 2.2, since server placement strategy is the primary subject of this entire work.1. Because the tree-based technique follows a predetermined path to the root, it can be applied to email services. Any situation can be used with the greedy method. Since latency is the main optimization element, techniques like HotSpot, HotZone, Max Fan Out, and NetClust are appropriate in multimedia and real-time scenarios. Advertising organizations can utilize the GeoIP technique to

keep area-specific data to a specific server, as it allows for the retrieval of actual client locations. The Flow Count Strategy can be advantageous for autonomous systems since it lessens traffic between various networks. The circumstance and the goal of the CDN deployment always influence the choice of replica placement approach.

Table 2.6: Application Areas of Replica Placement Algorithms

Replica placement algorithms	Application areas
Tree-based	Email services
Greedy	Mobile content delivery, News, events updates, Multimedia Applications
Random	Advertisement content delivery
HotSpot	Streaming content delivery, Multimedia Applications
HotZone	Streaming content delivery, Advertisement content delivery, General websites delivery
Max Fan Out	Multimedia Applications
Flow Count Strategy	Streaming content delivery
GeoIP Clustering	Advertising

NetClust	Multimedia network, Streaming content delivery

2.4. CONCLUSION

The primary goal of this chapter has been to investigate CDN bandwidth-related difficulties. In addition to addressing the CDN's bandwidth problems, it looked into a number of ways to save bandwidth by utilizing the various CDN features, such as request routing, content caching, server placement, and content selection. Furthermore, many methods put out by multiple writers for CDN-related issues have also been examined. Chapters 4 and 5 will analyze the bandwidth cost in terms of deployment cost, server utilization and traffic load with respect to replica server placement. Sometimes bandwidth cost gets increased because of broadcast storm problem in CDN which is explained in Chapter 3.

3. PRACTICAL PART

Broadcasting is the concurrent transmission of an identical message to multiple recipients. Broadcasting is a fundamental operation in networking to resolve several issues like paging a particular host, finding a route to a particular host, and sending an alarm signal. Simple flooding or blind flooding can be considered as the easiest way for broadcasting where each and every incoming packet gets retransmitted to all neighbors except the one where it has come from. The broadcast signals may overlap with one another which finally results in high collision, high contention, and elevated redundancy, generally termed as Broadcast Storm Problem (BSP).

1. **Redundant rebroadcasting**: In blind flooding, when a message is broadcast, all the

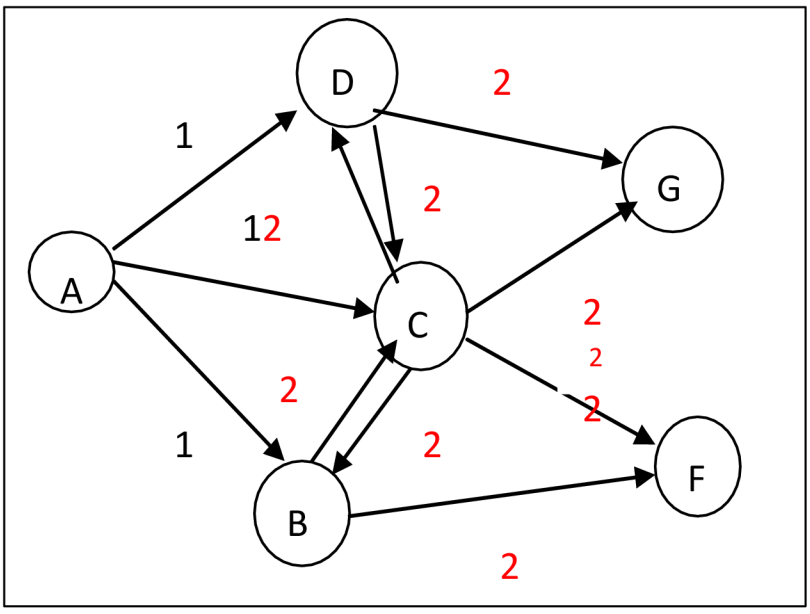
neighbors within the source's transmission range receive that message. When the message gets rebroadcast by any of the recipients, it leads to redundant rebroadcasting as most of its neighbors already have that specific message.

2. **Contention:** After receiving the message from the source, if all the neighboring nodes decide to rebroadcast the same message, the transmissions will contend with each other.

3. **Collision:** Due to heavy traffic (as all the neighboring nodes are rebroadcasting the same message) and absence of Request-To-Send and Clear-To-Send signals, collision will occur, resulting in packet loss.

Figure 3.1 illustrates the concept of Broadcast Storm Problem. In this diagram, A is considered as the source node and F is the destination node. Initially node A broadcasts a packet which is received by all its 1-hop neighbors which are B, C and D. Next, all the recipients will rebroadcast the same message to their 1-hop neighbors. As a result, B and C and also C and D will contend as B & C or C & D already have received from message A. Packets from B and C will collide at F and packets from C and D will collide at G. So, there is a high probability of losing the packet at node F due to the collision.

BSP also occurs in CDN. In the case of cooperative pull based CDN, during cache miss, the request is redirected towards the neighboring edges. If that specific content is not found, then again the data is directed to their neighbors, which is a time-consuming process. So, the easiest way is to



broadcast the request. But the broadcasting and rebroadcasting signals may overlap and collide which ends up in BSP. In CDN, BSP can be considered as a consequence of content scaling over the network. This chapter will focus on objectives 2 and 3 which are related to the investigation of content scaling and Broadcast Storm Problem.

Figure 3.1: The concept of broadcast storm problem

3.1. BSP IN CDN

BSP is a natural phenomenon in wireless sensor networks like MANET or VANET. It is not much talked about in the field of CDN. But the content outsourcing and content replication in CDN have a direct effect on BSP. The outsourcing and replication techniques for CDN are discussed in detail in section 3.3.

CDN works efficiently with a clustering approach. For example, N number of nodes in a CDN get grouped into K number of clusters and each of these K replicas maintain a record of neighboring surrogates, their locations and distance required to redirect requests in cache miss. The client request gets directed to the nearest surrogate by the request routing technique. The nearest surrogate is not always the best one i.e., it may not be consisting of the required content. In that case the request has to be redirected towards the nearest neighbor and so on. It includes a significant amount of delay. So, the easiest strategy is broadcasting the request to all the neighbors of the server where the miss has occurred. Another situation where broadcasting is required is when there is an update in the origin. Distance between replicas is an important factor in CDN broadcasting. If there is any update in a server R, that update may be broadcast using simple flooding, but it may happen that many of the surrogates will not receive the message as BSP is an obvious consequence to the simple flooding.

3.2. CONTENT DISTRIBUTION AND REPLICATION IN CDN

CDN is either Push-Put together or Pull-Based depending on respect to the technique followed for content dissemination. In Helpful Push-Based approach, the CDN pre-brings the substance to the substitutes from the beginning before they are gotten to. Here a planning is kept up with by the CDN supplier between the edge servers and the substance. Every client demand is directed to its

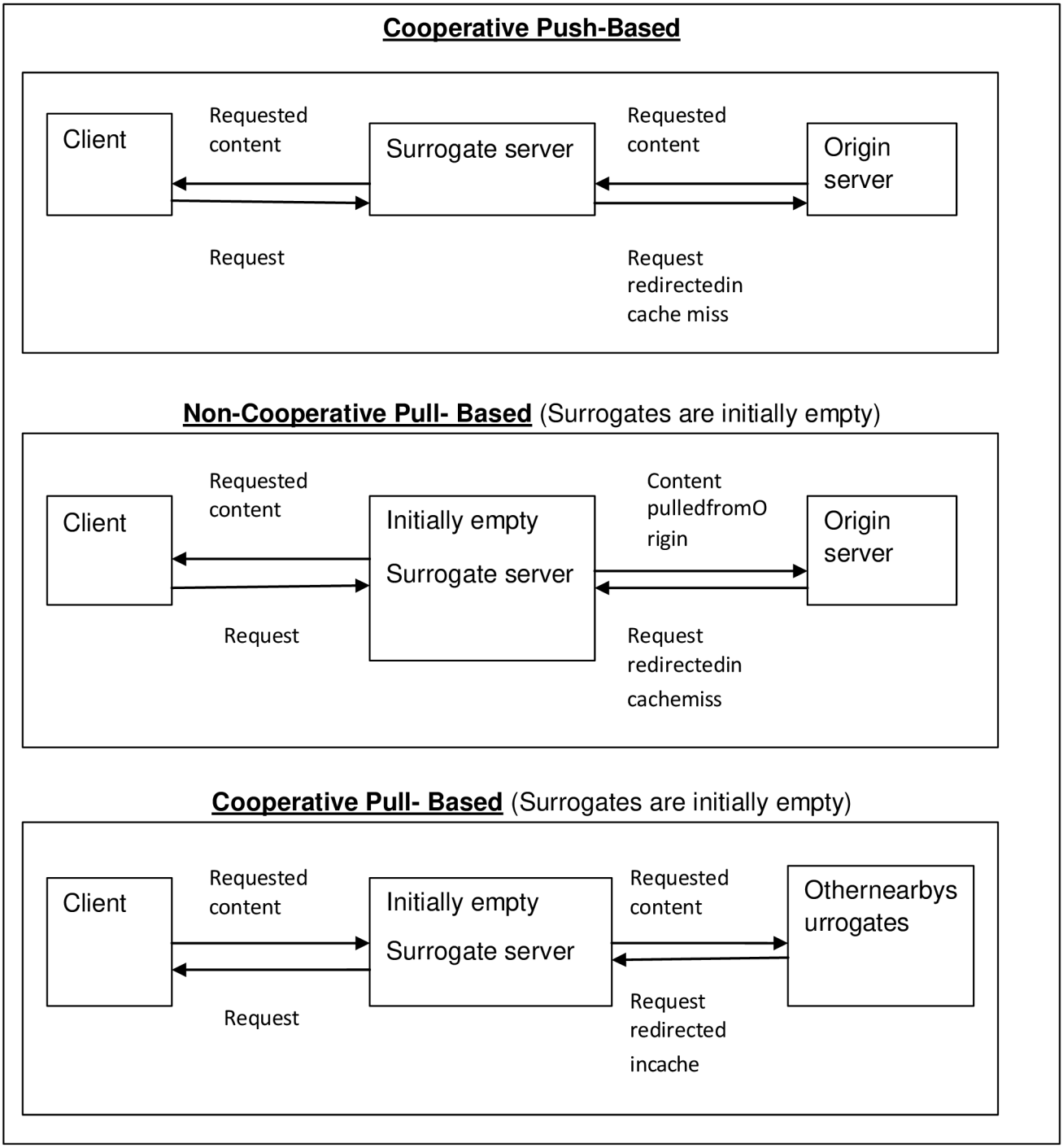
closest server and in reserve miss it goes to the beginning. Pull-based CDN is either Agreeable Draw Based or Non-Helpful Force Based. In Non-Agreeable Draw Based the client demand is coordinated to the closest server and in the event that the mentioned content is absent in that server, the solicitation is straightforwardly shipped off the beginning. The reaction is served to the client and furthermore get put away in the server. The Helpful Force Based is not the same as the non-agreeable as here, the client demand is diverted to the close by substitute in the event of store miss. Figure 3.2 shows the different substance re-appropriating strategies.

Presently in Non-Helpful Draw Based approach, the likelihood of BSP is nearly low as the solicitations get coordinated to the beginning server in reserve miss while for Agreeable Force Based, the opportunity is high as solicitations need to go through the network by means of adjoining hubs.

Next concern is replication of information that arrangements with which information ought to get copied in which imitation server. In CDN information is reproduced in two distinct ways: it is possible that it is full-site replication or halfway site replication as displayed in Table 2.2. In full-site replication, the whole beginning server informational collection gets copied in all proxies that guarantee practically 100 percent information accessibility from copy servers. Conversely, the halfway site replication copies just the implanted articles in copy servers and the base HTML page

is recovered from beginning.

Figure 3.2: Different content outsourcing techniques



It can be assumed that partial-site replication is more inclined to BSP as the number of cache miss

is high, but full-site replication has both pros and cons from BSP point of view. Becauseof100%replication, the cache miss is supposed to be low, and it also reduces the request routing. But at the same time frequent updates in origin increases the broadcasting of update traffic as the update messages have to be propagated to all the servers. The same is also applicable for pushbased CDN.

3.3. SOLUTION TO BSP

There are several parameters based on which broadcasting can be achieved. For simple flooding BSP is obvious and to reduce its effect the other approaches have been proposed.

3.3.1. Simple flooding:

In simple flooding the source node initiates the process by broadcasting a packet to all its neighboring nodes. Upon receiving the packet, all the recipients start rebroadcasting the packet to all its neighbors. Each node sends a packet only once. Since every single node is actively involved in sending, during a flood it is very likely that a BSP will occur.

3.3.2. Probabilistic broadcasting:

Probabilistic methodology is a better form of flooding where every hub broadcasts a message with a particular likelihood. Tattle based and Counter-based routing are awesome probabilistic broadcasting models. In this plan, the likelihood by which a particular hub will broadcast relies upon the quantity of its neighbors. The worth of is conversely relative to the quantity of neighbors i.e., assuming the Neighbor populace is high, the broadcasting likelihood gets diminished, else it gets raised.

i. Gossip1 (p): The source node will broadcast a packet with probability 1 to all its neighbors. But all the neighbors are allowed to broadcast the same message with predefined probability p exactly once. If any node receives that same packet for the second time, it will discard that with probability $1 - p$.
ii. Gossip1 (p, k): In this approach, one extra parameter k has been added. Here the first k hop neighbors of the source will broadcast the same packet with probability 1 and rest of the nodes will gossip with probability p.
iii. Gossip2 (p_1, k, p_2, n): This approach differs from the previous one as it considers

a threshold n on the number of neighbors. Initial k hop neighbors broadcast the message with probability 1. Rest of the node will gossip based on the value of$n$. If the neighbor population of a node is greater than$n$, it will gossip with probabilityp_1, else it will gossip with probability p_2 with a constraint $p_1 < p_2$.

iv. Gossip3 (p, k, m): Here, the first hop node transmits with probability 1. The remaining nodes send gossip with at least probability if they do not receive the same message from their neighbors.

v. Counter-based broadcasting: It is a well-known approach for broadcasting. Here an edge esteem is characterized as what ought to be the most extreme number a hub gets a similar packet to rebroadcast it. At first the count is set to 0. Every time the message is received, the value of count gets incremented by If it crosses the predefined threshold within a prescribed time period, then the rebroadcasting is prevented from that node.

vi. Density- based probabilistic counter plan: It is a crossover approach which consolidates the upsides of counter-based and straightforward probabilistic systems. Every single hub rebroadcasts a packet with likelihood which not entirely settled by the thickness for example the quantity of - jump neighbors. Here also the concept of count variable is applicable. Each node can remove the duplicate packets upon receiving the same message.

vii. Distance-aware counter-based broadcast: This scheme introduces distance parameter to determine the broadcasting probability. Two different Random Assessment Delays (RADs) are defined separately for interior nodes and border nodes. The RAD for border nodes is comparatively smaller than that of interior nodes which makes it clear that the rebroadcasting probability is higher at border than from interior.

viii. Dynamic probabilistic counter-based broadcast: According to the methodology, each hub sets its counter to an underlying worth, by and large 0. After getting a similar packet the counter gets expanded. Inside a particular RAD on the off chance that the counter worth surpasses the limit, a low sending likelihood gets produced and simultaneously an irregular number is likewise made in the scope of [0, 1]. In the event that the likelihood is more prominent than the arbitrary

number, just the broadcasting will happen, else there will no rebroadcast.

3.3.3. Distance-based broadcasting

In this scheme, the node-to-node distance is taken as a parameter based on which each node decides the forwarding probability.

i. Weighted p-persistence broadcasting: Here, the first hop node transmits with probability 1. The remaining nodes send gossip with at least probability if they do not receive the same message from their neighbors., $p_{AB} = \frac{d_{AB}}{TR}$ where d_{AB}represents the distance between A and B and TR represents the transmission range.

ii. Slotted-1 persistence broadcasting: When a node receives a message for the first time, it will wait for a predefined time slot T for the same message to receive. If it does not, then the message will get rebroadcast by probability 1, else the message will get rejected. iii. Slotted-p persistence broadcasting: It is almost similar to slotted-1 except the probability factor. Here, a node will rebroadcast a message received for the first time with probability$p$, if it does not receive any duplicate message within a time slot.

3.3.4. Neighbor knowledge-based broadcasting

Here each node adds its neighbor list to the packet header. The receiver checks the list, and it will rebroadcast the message only if there is an extra node in the receiver's neighbor list.

- **Flooding with self-pruning:** Before broadcasting a message, a node should include its neighbor list to its header. Upon receiving a packet, the receiver checks and compares its own neighbor list with the one given in the packet header. The message will be broadcast only if there is at least one extra node in the receiver's neighbor list, else the message gets dropped.

- **Scalable broadcasting:** Upon receiving a message for the first time, the receiver compares the neighbor list included in the packet with its own. If it finds any extra node to receive the message after rebroadcasting, then only it will wait for a RAD. It will only rebroadcast the message if it does not get any duplicate message within this RAD.

- **Dominant pruning:** Each and every node is considered to maintain the information up to

its 2-hop neighbor. While broadcasting a message, a node Includes a subset of its neighbor so that only a set of selected nodes will rebroadcast the message.

- **Double-covered broadcasting:** This approach selects some of the sender's 1-hop neighbors to retransmit the message, ensuring that all 2-hop neighbors receive the message, and all 1-hop nodes are covered by at least two forwarding nodes.

3.4. COMPARATIVE ANALYSIS OF BROADCASTING ALGORITHMS

All broadcasting approaches referenced in 3.4 have been utilized to reduce the BSP impact. Every single methodology has its own benefits and negative marks. A near report among the methodologies is acted in Table 3.1 to view as the most proficient one. Straightforward flooding heightens the likelihood of BSP. For portable networks like Vehicular Specially appointed Networks (VANET), Distance-based broadcasting can be applied where neighbor data can be obtained through satellite. Reachability gets upgraded with the utilization of Neighbor Information based Broadcasting. Subsequently, it will likewise build the intricacy. The Probabilistic methodology, especially counter-based and Tattle broadcasting are able enough with okay intricacy and can be valuable in a network.

Table 3.1: Comparison of Broadcasting Algorithms

Broadcasting Algorithms	**Advantages**	**Disadvantages**
Simple Flooding	Decentralization	Network Congestion, Redundant Transmissions, and Resource Utilization
Probabilistic Broadcasting		
Gossip1 (p)	Simple	Gossip is dependent on the number of neighbors of the source node.
Gossip1 (p, k)	Reachability is higher than Gossip1 (p). Traffic gets reduced up to 35% compared to simple flooding.	The probability p,k should be large enough (0.65- 0.75) to eliminate the possibility of premature message death.
Gossip2 (p_1, k, p_2, n)	Less probability of message expiry. Redundancy gets reduced as probability is controlled.	There will be no impact on regular networks.
Gossip3 (p, k, m)	Premature gossip death is prevented.	Latency due to retransmission may be an issue.

Counter-based	Simple. Inherent adaptability to local topology.	Can be applied in thick network. Counter-based turns into a straightforward flooding in scanty network and saved rebroadcast (SRB) gets diminished forcefully.
Density-based Probabilistic Counter scheme	Start to finish delay gets limited and it additionally increments conveyance proportion when contrasted with counter based.	Channel contention gets increased as every node is allowed to retransmit.
Distance Aware Counter-based	Due to the primary parameter Expected Additional Coverage (EAC), the reachability becomes very high almost 95%). Not sensitive to the network topology.	Computation of two different threshold values Counter and Distance, two different RAD values and two different probabilities for interior and border nodes made the system more complex.

Dynamic Probabilistic Counter-based	Dynamic probability computation method can be used for all types of networks. Channel contention gets reduced even at high broadcast injection rate.	Considers a scenario where the number of received duplicate messages exceeds the threshold value and calculate the forwarding probability.
Distance-Based Broadcast		

Weighted p-Persistence	Light weight strategy. Higher reachability	Generally applicable to mobile networks.
Slotted-1 Persistence	Likelihood to the hubs from the broadcaster expands the reachability.	Generally applicable to mobile networks.
Slotted-p Persistence	Higher reachability. Packet loss gets reduced by almost 90%.	Generally applicable to mobile networks.
Neighbor Knowledge-Based Broadcast		
Flooding with Self Pruning	Simple.	Does not show efficiency for all types of networks.
Scalable Broadcast	Neighbor knowledge is used efficiently.	Larger number of rebroadcasting nodes is required in order to improve the gain in performance.
Dominant Pruning	Elevated reachability with the use of Greedy Set Cover algorithm.	Not suitable for versatile climate as it doesn't utilize nearby data to determine next rebroadcasting hubs.
Double-Covered	Unfailing retransmission. Sensitive to nodes' mobility. During high mobility, because of wrong neighbor information, transmission rate becomes very low.	Number of retransmissions turns out to be exceptionally high as the shipper hub continues to rebroadcast when it doesn't get greatest number of affirmation from the sending hubs.

3.5. BSP SOLUTIONS FOR CDN

Several solutions available for BSP are mainly used for MANET and VANET. But those solutions can also be applied in CDN with some modifications. For a better understanding, let's consider a clustered network structure where *N* number of nodes is grouped into K number of clusters. Each cluster is facilitated with a surrogate. Each of these K replicas is expected to maintain the information of other replicas along with their locations, in-between distances etc. required for request routing. Instead of using flooding, other broadcasting techniques can be used to reduce the effect of BSP. Among the broadcasting approaches distance-aware is basically used for mobile networks. Neighbor-based approach will be more complex, and it may increase the delay and traffic because of huge number of retransmission and acknowledgement messages. Because of its simplicity, a counter-based approach can be applied in CDN. In cache miss, the surrogate broadcasts the request. Upon receiving that message, the recipients will confirm the ID of the sender whether it is the originator or not. If it is, then the recipient will send an acknowledgement in case of hit. If a source in the RAD does not receive the required number of acknowledgments, it resends the request. Several studies have proven that X = 3 can improve system performance.

3.6. CONCLUSION

This chapter examines the substance scaling procedures in CDN and BSP as the consequence of content replication which is referenced in objective no. 2 and 3. Various answers for BSP have been examined in this section. BSP is a serious worry as it might bring about information irregularity and information loss. Be that as it may, generally the current answers for BSP were proposed for remote sensor networks. The near concentrate on shows that the probabilistic methodology for broadcasting is more successful in any network. A counter-based system should be more appropriate to the CDN situation regarding higher reachability with moderately less packet loss. The impact of BSP can be limited by lessening the quantity of replication which has been finished in Section 4 by enhancing the quantity of proxies to be put.

4. RESULTS AND DISCUSSION

4.1. RESULTS

To curb the effect of flash crowd, CDN deploys several replicas at the user proximity. This approach definitely cuts down the bandwidth cost as the data transmission path gets shortened. The content against the client request gets loaded from the nearby surrogates in place of the origin server which in turn minimizes packet loss and delay in response. The primary concern for CDN designers is to find out the locations for deploying surrogates. Manual decision making is almost impossible because of the heterogeneous and dynamic nature of CDN. A huge chunk of internet traffic is generated especially due to the popular sites like YouTube, Facebook, Twitter, and LinkedIn. This gigantic unstructured raw data can only be well managed by unsupervised machine learning. This unsupervised machine leaning has only input data but no corresponding output variable and the main aim of this is to distribute the data or model in a structure so as to learn about the data in an efficient way. Clustering can be considered to be the most efficient unsupervised predictive model to structure a collection of mass volume of unlabeled data. In this approach the response variables are grouped into user- defined clusters based on several characteristics like distance, density etc. Here N number of nodes is divided into K number of clusters based on the Euclidian distance measured by computing the square of the distance between each pair of nodes and then adding the square and getting the square root of the sum result. And the mean position of each cluster will be selected for surrogate deployment. But this simplest unsupervised approach may produce clusters consisting of an insignificant number of nodes. Surrogates deployed in those clusters having few numbers of nodes are left underutilized which causes deployment overhead, maintenance overhead and wastage of resources, especially storage. This chapter modifies the K-Means algorithm based on the number of nodes within a cluster. A parameter named population threshold has been introduced for this purpose. Any cluster having population less than the threshold value will not have its own replica and any request from that cluster will be served by nearby surrogates. This way the number of servers can be optimized over the network. This chapter will explain the basic

principle of K-Means algorithm followed by population-based clustering, which is the modified K -Means based on population threshold. Next, it will describe the replica server placement algorithm using population-based clustering and finally the performance of the algorithm will be compared with the simple clustering algorithm considering the parameters like deployment cost and server utilization factor.

4.2. CLUSTER GENERATION USING K-MEANS.

Population-based clustering is based on the simplest unsupervised clustering approach KMeans algorithm. The main idea of this approach is to generate K number of clusters out of N number of data points based on some features.

All the notations used throughout this chapter are presented in Table 4.1

Table 4.1: Notations

Notation	Description
K	No of clusters
K'	No of optimized clusters i
C_i	^{h} cluster
N	Total number of nodes in the network
D	Number of clusters having nodes below population threshold
P_i	Number of nodes in C_i
R_i	Replica server placed C_i in
T_p	Population threshold.
A	Number of nodes in the largest populated cluster
B	Average number of nodes
$N_{K'}$	Total number of nodes in K'clusters
U	Average utilization factor of replica server
u_i ρ_i	Utilization factor of replica server for cluster C_i
C_d	
c_{d_i}	
μ_i	

	Ratio of population of cluster C_i to the total population.
	Average deployment cost
	Deployment cost for unit server in cluster C_i.
	Mean of cluster i

For CDN, the nodes or the client positions are considered as the data points and the location latitude and longitude are taken as features for clustering. Here a network is considered which consists of total N nodes and it is defined that K number of surrogates will be deployed with a constrain $K << N$. So, the primary objective is to find K center points to place the replicas. Initially K random locations are selected as the center points of K clusters. The selection will give better results if the centroids are selected as far as possible from each other. Next, each data point gets associated to one of the centers based on the Euclidian distance between the cluster centroid and the node. Obviously these are not the final clusters. Next step is to calculate the mean of each newly formed clusters and based on the mean values again new K clusters get formed and this process goes on until there is no change in the cluster membership. The entire process is written in the following algorithm.

Algorithm 4.1: Cluster generation using K-Means

Input: Coordinates of data points: Number of clusters to be formed

Output: Clusters along with their centroids ( ) and cluster members.

Step 1. Introduce centroids one for every one of the quantity of clusters

Step2. Assign each of the N data points or nodes to a cluster whose centroid is closest to the selected node.

Step3. Recalculate the new clusters' centroids μ_i.

Step4. Repeat steps 2 & 3 until there is no longer change in membership of all K clusters.

4.3. POPULATION-BASED CLUSTERING

In CDN, data gets disseminated through the origin server established by content provider to several

surrogates deployed at different strategically identified places by the service provider. The main aim behind this model is to protect the origin server from bottleneck situations when there is an upsurge in the client request. As the same data gets replicated at various places, CDN has to ensure the data inconsistency. So, any update in the origin server should get reflected in all the edges instantly and vice versa. The algorithm discussed in this chapter adopts the existing policies for data replication and request routing. Population-based clustering uses full-site replication which means all the surrogates will replicate the entire content of the origin server. For request routing it considers non-cooperative pull-based strategy which defines that in case of cache miss, the request will be directly redirected to the origin server and the response will be served to the client and also get cached to that particular surrogate for future reference. The replication strategy followed in this approach minimizes the possibility of cache miss. For content outsourcing, the routing strategy termed as non-adaptive request routing is used, where the URL entered by the client gets directed towards the DNS which provides IP address of the content provider in return. While the users send requests to the server, it sends the HTML content along with the IP address of the surrogate. The client request is finally redirected to the appropriate replica server. The proposed system model is depicted in Figure 4.1.

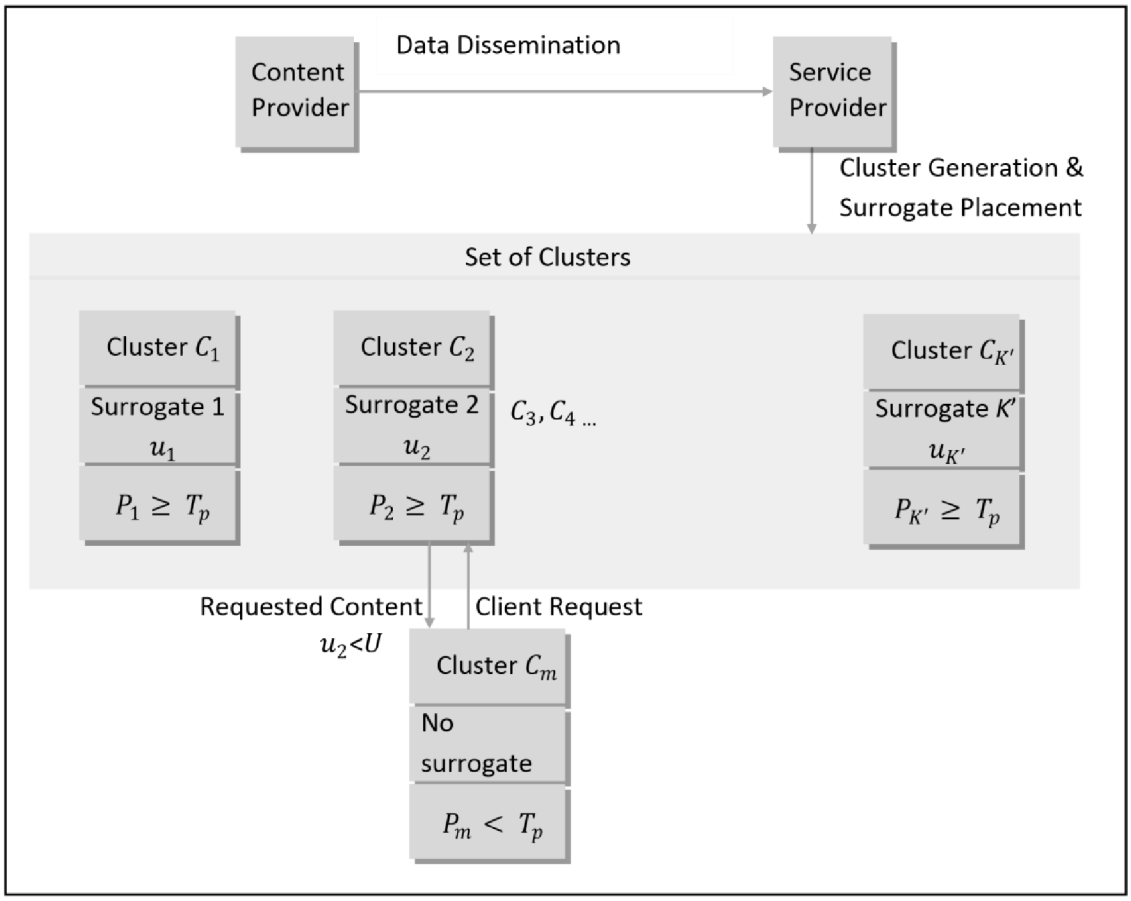


Figure 4.1 System model for Population-based replica placement algorithm

Now the main concern is to finalize the locations to deploy the surrogates. The simplest unsupervised clustering algorithm K-Means has been selected to generate the clusters out of the client locations and to find the centroid of each cluster as the position to place the replicas. KMeans has already been discussed in section 4.2 through which N number of nodes get grouped into K number of clusters. To ensure the proper utilization of all the replicas, the number of surrogates can be optimized based on the number of nodes within a cluster. A replica server placement algorithm has been designed based on population-based clustering. The entire algorithm has been divided into two steps. The initial step is to present a boundary called population limit which characterizes the base number of hubs a cluster ought to comprise to have its own edge server. Thus, the clusters having population beneath the edge won't get any proxy. Requests from these under-populated clusters will be redirected to the nearest surrogate. The second part of the algorithm will identify the surrogate which can serve requests coming from outside the cluster

based on parameter like server utilization factor. If the utilization of a surrogate is equal or greater than the average utilization then it is considered that the server is already overloaded, and it is not able to take any extra burden. So, the second step is dealing with assigning membership to the under- populated cluster nodes.

4.3.1. Parameters used.

The parameters used in this chapter are given below with their definitions.

a) Population Threshold (T_p): Population threshold (T_p) characterize the minimum number of nodes a cluster should consist of to have a substitute in it. Population threshold can be calculated by the ratio of the number of nodes in the highest populated cluster (A) and the average number of nodes in the network (B) multiplied by the number of clusters (K).

Where $B = \frac{N}{K}$ .....Eq 4.3

b) Optimized Cluster (K'): The under-populated clusters will get identified based on the value of population threshold (consider$D$) and replicas will not be deployed in those D clusters. So, total number of clusters K will be optimized into K' clusters with the constrain $K' \leq K$.

c) Average Utilization Factor (U): The average server utilization is estimated to be $U = \frac{100}{K'}$

The identified under-populated clusters can only be combined with those clusters with population greater than or equal to T_p and utilization factor less than U.

d) Utilization Factor of each surrogate (u_i): Utilization factor of an individual surrogate placed in cluster i can be evaluated from

$\mu_i = \mathrm{p}_i \times 100$ Eq.4.4

Where ρ_i is the population ratio and can be determined by

$\mathrm{p}_i = \frac{P_i}{N}$ Eq.4.5

With P_i as the number of nodes in cluster$i$.

e) Average Deployment Cost (C_d): The average deployment cost of surrogate can be evaluated as

$C_d = +\sum_{i=1}^{K} p_i c_{d_i}$ Eq.4.6

Where ρ_i is the population ratio of cluster C_i and c_{d_i} is the deployment cost for unit server in cluster C_i. For metro city C_{d_i}=6, while for important city it is 2 and 1 for general city.

4.3.2. Evaluation of Population threshold

Consider a network grouped into 12 clusters consisting of 15, 250, 695, 10, 100, 640, 830, 700, 20, 1050, 850 and 1000 nodes respectively. Here, A = 1050, N =- 6160 So, B = 6160/12 = 513.33.

Now the value of $T_p = \left(\frac{1050}{513.33}\right) \times 12 = 25$

If population threshold is applied, all the clusters having nodes below 25 will not get its surrogate. For the given network, 3 clustersC_1, C_4, C_9 get identified, with population 15, 10 and respectively. Now total number of surrogates $K = 12$ can be optimized to K'which is where $K'$the number of is under-populated clusters So, instead of placing 12 replicas 9 surrogates can be placed over the network. How the requests from these 3 identified surrogates will be served is discussed in 4.3.2.

$$K - D = 12 - 3 = 9$$

4.3.3. Memberships of under-populated cluster nodes

For the given network only 9 surrogates will be deployed. Requests from the rest of the 3 underpopulated clusters will be served by these 9 surrogates. Now membership of these underpopulated cluster nodes will be determined by the average server utilization (U) and utilization factor of each surrogate (u_i). The average server utilization can be evaluated by Eq.4.3.

So, any surrogate having utilization factor greater or equal to this average value is overburdened or saturated. So, any extra request will make it overloaded which will again result in user latency or packet loss. The individual surrogate utilization factor can be calculated using Eq.4.4.

The population-based clustering replica server approach considers that an under- populated cluster I can only be merged with a cluster J if the following two conditions get satisfied:

$P_j \geq T_p$ And $u_j < U$..... Eq.4.7

The procedure of assigning membership to the under-populated clusters is given in the following algorithm:

Algorithm 4.2: Assigning memberships to under-populated cluster nodes Input: $, A, P_i$ for K $i = 0,1, \ldots., K$Output: Number of clusters with population where; Server locations,

Were, $\boldsymbol{i = 1, 2, 3, \ldots \ldots \ldots., K'}$

Step 1: Evaluate average no. of nodes in the network.

$$\boldsymbol{B = \frac{N}{K}}$$

Step2: Calculate population thresholdT_p.

$$T_p = \frac{A}{B} \times K$$

Step3: Identify clusters with population less thanT_p.

Step4: Find the cluster centroid which is closest to the centroid of the under-populated clusterC_j.

Step5: Check if $P_i \geq T_p$ and$\mu_i < U$, then redirect the request of C_j to the edge server of C_i Else find the next nearest cluster centroid.

Step 6: Rehash Stage 4 and Stage 5 until a cluster is related to $P_i \geq T_p$ and the previously mentioned model with K = 12, D = 3 and K' = 9 is presented in Figure 4.2.

From Figure 4.2, following two important cases are discussed: Case I.μ_i > U: In Figure 4.2, Cluster C_1has 15 nodes which is below the population threshold. As $P_l < T_p$, no replica server will be placed in C_l. Now, consider that the nearest cluster of C_1 isC_{10}. The population of the cluster $C_1$is greater than threshold but (Calculation of utilization factor is shown in 4.4.2). As the server R_{10} is already over loaded, it cannot serve request from clusterC_1. The next nearest cluster of C_1 is C_2 with 250 nodes. As $P_2 > T_p$and utilization factor u_2(4.06) < U, request from C_1 can be served byR_2.

$$\mu 10(17.05) > U(11.1)$$

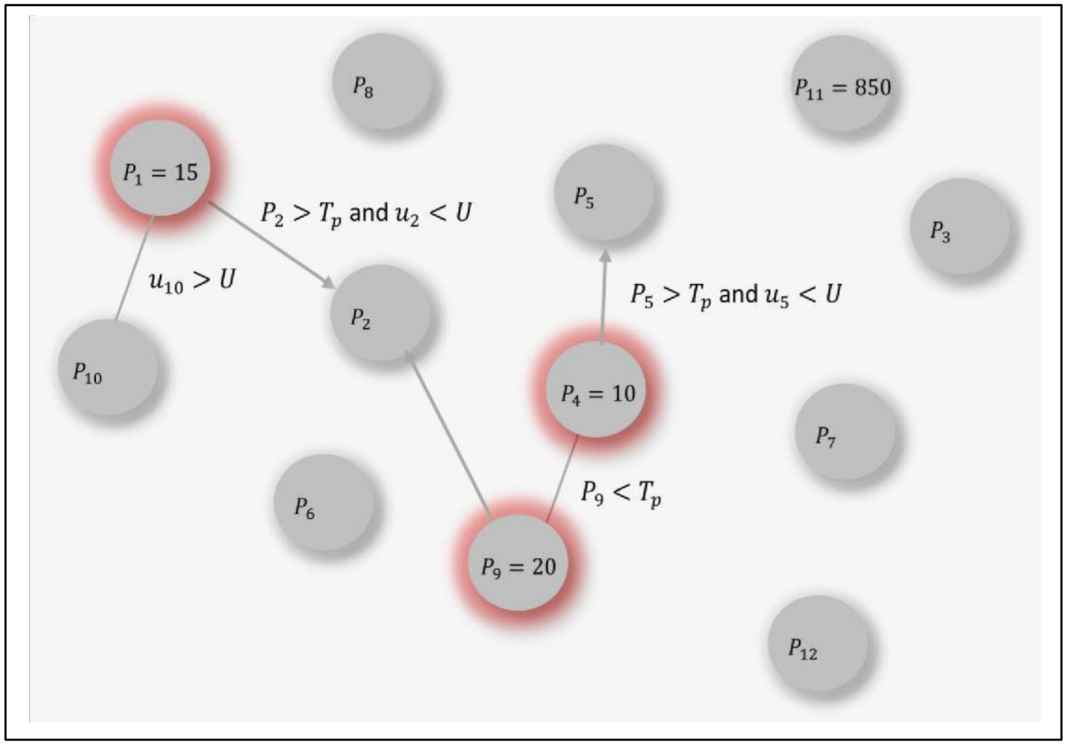


Figure 4.2 Network with K = 12 and K' = 9 with T_p = 25.

Case II. $Pi < Tp$: In Figure 4.2, cluster C_4 consists of 10 nodes. The cluster nearest to C_4 is C_9 whose population is also below the threshold. So, no C_4 request can be directed towardsC_9. The next nearest cluster C_5 is identified with P_5 = 100 and μ_s = 1.62. So C_4can be serviced through R_5.

4.4. PERFORMANCE PARAMETERS

The performance of the population-based replica server algorithm will be evaluated based on two performance parameters – deployment cost and server utilization factor. Both the parameters will be measured for the given algorithm and then compared to that of simple clustering technique. In case of simple clustering, the latitude and longitude of client locations get collected from the log file through IP Geolocation service. The entire network is then partitioned into a number of clusters and the mean position of each cluster gets identified for server placement.

4.4.1. Utilization factor

The parameter, utilization factor measures the utilization of each surrogate. For the given replica placement algorithm, the average utilization factor can be calculated as $= 100/K$, as K is

optimized to K. The same factor can be measured for individual replica by Eq. 4.4.m Consider the above example with K = 12, K' = 9 and T_p = 25. For simple clustering, the average utilization is $U = 100/12 = 8.33$. Whereas the same for population-based approach is $U = 100/9 = 11.11$. Considering the redirections $C_1 \rightarrow R_2$, $C_4 \rightarrow R_5$ and $C_9 \rightarrow R_2$ the utilization factor of each surrogate for both the approaches is calculated and shown in Table 4.2 using Eq.4.4.

Table 4.2 Utilization Factor for Simple Clustering and Population-Based Clustering

Server	Utilization factor for simple clustering	Utilization factor for population-based clustering
R1	0.2	Not deployed
R2	4.1	4.6
R3	11.3	11.3
R4	0.2	Not deployed
R5	1.6	1.8
R6	10.4	10.4
R7	13.5	13.5
R8	11.4	11.4
R9	0.3	Not deployed
R10	17	17
R11	13.8	13.8
R12	16.2	

The values in Table 4.2 show that in case of population-based clustering, only 9 surrogates get deployed and if we compare the values of the utilization factors for the surrogates R_2 and R_5 , it gets increased in population-based approach compared to simple clustering.

4.4.2. Deployment cost

One of the important factors to evaluate the CDN performance is deployment cost. The average deployment cost of surrogate can be evaluated using Eq.4.6. In the given approach all the surrogates are considered to be deployed in important cities and as per the Chinese Internet market, the unit price for individual surrogate deployment is 2.

Table 4.3 Population Ratio for Simple Clustering and Population-Based Clustering

Cluster	pi in simple clustering	pi in population-based clustering
C1	0.0024	Server not deployed
C2	0.0406	0.0463
C3	0.1128	0.1128
C4	0.0016	Server not deployed
C5	0.0162	0.0179
C6	0.1039	0.1039
C7	0.1347	0.1347
C8	0.1136	0.1136
C9	0.0032	Server not deployed
C10	0.1705	0.1705
C11	0.138	0.138
C12	0.1623	

Using Eq.4.7 and the values given in Table 4.3, the value of C_d is evaluated as 1 where total number of surrogates is 12 and the value of C_d in population-based clustering with number of cluster as 9.

4.5. PERFORMANCE EVALUATION

Performance of the given algorithm population-based replica placement algorithm is evaluated based on deployment cost (C_d) and server utilization factor (μ_i). Consider the previous example with K = 12, K' = 9 and T_p = 25 As the total number of nodes are same for both the approaches, deployment cost is almost similar for both, which is nearly equal to 1. But if utilization factor is considered, Figure 4.3 shows that the μ_i for surrogates R_2 and R_5 got increased in population-based compared to simple clustering.

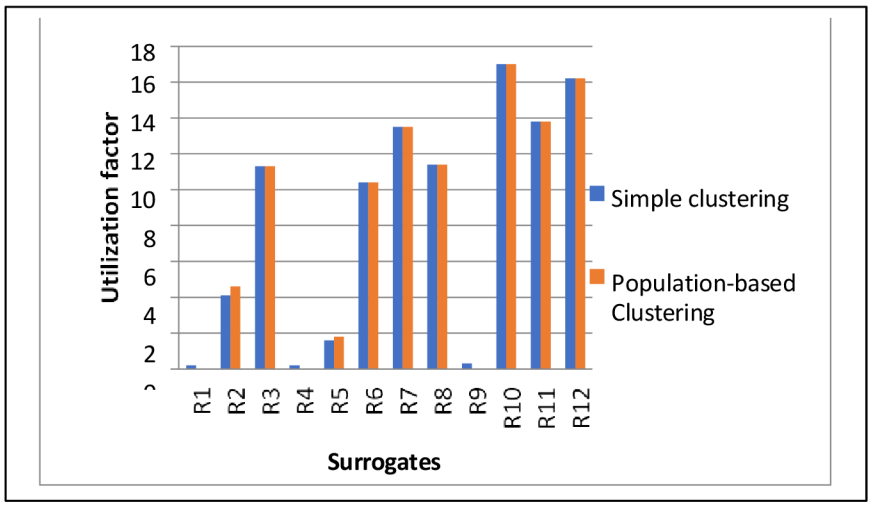


Figure 4.3 Comparison of utilization factor between Population-based and *N*-Means clustering.

For better understanding of the advantages of population-based clustering, consider a larger network consisting of 34 clusters having 100, 1270, 950, 55, 5500, 250, 900, 460, 4400, 1540, 115, 270, 1045, 1230, 716, 130, 1720, 940, 70, 380, 1000, 40, 2020, 340, 85, 310, 500, 1040, 3090, 520, 1200, 3200, 1050 and 2000 nodes, respectively.

For this example, N = 38436, A= 5500 and K = 34. As per Eq.4.1, the population threshold is $Tp = 5500/1130.29 \times 34$, which is equal to 166 and based on the value of threshold, 7 clusters got identified (C_1, C_4, C_{11}, C_{16}, C_{19}, C_{22} and C_{25}) with population less than 166. So, K = 34 will be optimized to $K = 34 - 7 = 27$. So, no surrogate will be deployed in those 7 under-populated clusters.

4.5.1. Utilization factor

For simple clustering, the average utilization factor for the above network is $U = 100/34 = 2.94$ whereas, in population-based $U = 100/27 = 3.7$ which means the surrogates are expected to serve more in the given approach. Any under-populated cluster requests will not be served by any server having surrogate utilization factor more than or equal to 3.7. The consideration for request redirection from under-populated clusters is as follow:

$C_1 \rightarrow C_{15}, C_4 \rightarrow C_{31}, C_{11} \rightarrow C_{21}, C_{16} \rightarrow C_3, C_{19} \rightarrow C_8, C_{22} \rightarrow C_{30}, C_{25} \rightarrow C_{13}$

The individual surrogate utilization can be calculated as $u_i = \rho_i \times 100$ which is given in Table 4.4.

Table 4.4 Surrogates Along with Their Utilization Factor Before Merging with Under Populated Clusters

Server	Utilization factor	Server	Utilization factor	Server	Utilization factor
R2	3.3	R13	2.72	R26	0.81
R3	2.47	R14	3.2	R27	1.3
R5	14.31	R15	1.86	R28	2.71
R6	0.65	R17	4.47	R29	8.04
R7	2.34	R18	2.45	R30	1.35
R8	1.2	R20	0.99	R31	3.12
R9	11.44	R21	2.6	R32	8.33
R10	4.01	R23	5.26	R33	2.73
R12	0.7	R24	0.88	R34	5.2

So, the utilization of the clusters to which the extra requests get redirected will change, which is

given in Table 4.5.

Table 4.5 Surrogates Along with Their Utilization Factor after Merging with Under-Populated Clusters

Server	Utilization factor	Server	Utilization factor
R15	2.12	R8	1.38
R31	3.27	R30	1.46
R21	2.9	R12	2.94
R31	2.81		

4.5.2. Deployment cost

It is viewed as that the urban communities where the proxies are conveyed are extremely significant urban communities where the unit cost for sending every substitute C_{di} is 2. For the given model, the absolute number of clusters is K = 34 which got limited into K' = 27 based on the worth of the population edge. Average deployment cost (C_d) for every one of the 34 substitutes is 1.98 while the equivalent for sending 27 servers is additionally assessed as 1.98.So, average deployment cost does not vary so much with the number of clusters as it basically depends on the number of nodes in the network.

4.5.3. Result and discussion

In 4.5.1 and 4.5.2, it is revealed that the average deployment costs for both the scenario (simple *N* -Means clustering and Population-based clustering) are same which is equal to 1.98. But average as well as individual utilization factors for surrogates vary in Population-based clustering. Utilization factor (μ_i) becomes null for C_1, C_4, C_{11}, C_{16}, C_{19}, C_{22}, C_{25}, as the population of these clusters are below the threshold. But the utilization for the servers to which the requests from underpopulated clusters get merged will be changed. Comparative study of the utilization factor for those surrogates is given in Table.4.6.

Table 4.6 Comparative Analysis of Utilization Factor

Replica Server	µi in standard K - Means	µi in proposed approach
R1	0.27	Will not be deployed
R4	0.14	Will not be deployed
R11	0.3	Will not be deployed
R16	0.34	Will not be deployed
R19	0.18	Will not be deployed
R22	0.1	Will not be deployed
R25	0.22	Will not be deployed
R3	2.47	2.81
R8	1.2	1.38
R13	2.72	2.94
R15	1.86	2.12
R21	2.6	2.9
R30	1.35	1.46
R31	3.12	3.27

The comparison in Table 4.6 shows that the utilization of the servers which are serving the extra requests from under populated clusters gets enhanced in population- based compared to that of simple K-Means clustering. The comparison is also represented in Figure 4.4.Also, as the number of clusters gets minimized in the population-based, it is obvious that replication cost, maintenance cost will be reduced and there is no need to maintain the log files required for under-populated clusters.

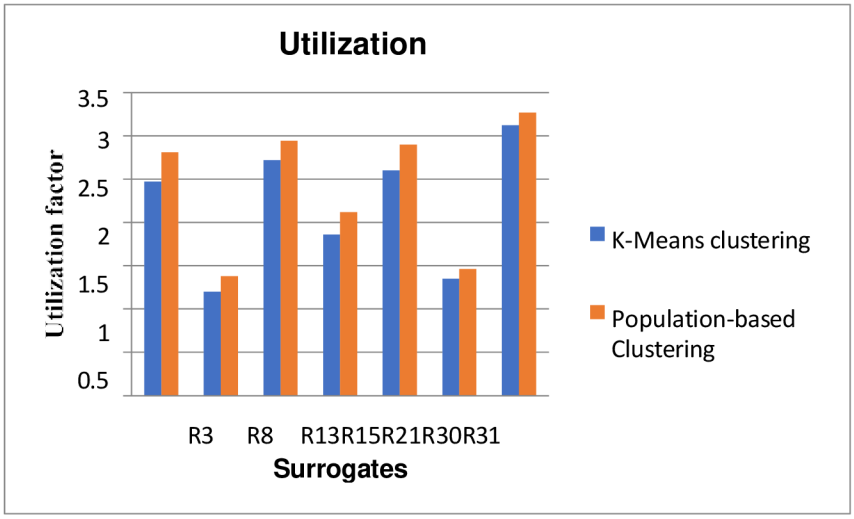


Figure. 4.4 Comparison of utilization factor between *K*- Means and Population-based clustering.

4.6. CONCLUSION

In conclusion, the intersection of Distributed Networking and Content Delivery Networks (CDN) reveals a symbiotic relationship that underscores the pursuit of a cost-effective and efficient digital infrastructure. Distributed Networking, with its decentralized approach to data processing and communication, aligns seamlessly with the goals of CDN, which aims to optimize content delivery by strategically distributing it across a network of servers. This synergy not only enhances the overall performance of digital services but also minimizes latency, accelerates data transfer, and ultimately reduces the associated costs.

By leveraging the principles of Distributed Networking, CDN capitalizes on a distributed architecture to store and deliver content closer to end-users, mitigating the challenges posed by centralized servers. This not only ensures faster access to data but also significantly diminishes the burden on individual servers, contributing to a more scalable and resilient network infrastructure. Moreover, the inherent flexibility of distributed systems allows for adaptive scaling, enabling CDN providers to dynamically adjust resources based on demand fluctuations, thereby optimizing operational costs.

The cost-effectiveness of this integrated approach becomes particularly evident in scenarios where a traditional, centralized server model would incur higher expenses for bandwidth, hardware, and maintenance. The distributed nature of CDN, guided by principles of Distributed Networking, promotes resource efficiency and resilience, resulting in a more sustainable and economically viable solution for content delivery in the digital landscape.

In essence, the marriage of Distributed Networking and CDN exemplifies a forward-looking strategy that not only meets the contemporary demands of a globalized digital environment but also offers a financially pragmatic solution. As the digital landscape continues to evolve, this harmonious relationship between distributed architectures and content delivery optimization promises to play a pivotal role in shaping the future of network infrastructure, ensuring both performance excellence and cost-effectiveness for businesses and end-users alike.

5. REFERENCES

1. M. Yu, L. Jose, and R. Miao, "Software defined traffic measurement with OpenSketch," in presented as the 10th USENIX Symp. Netw. Syst. Design Implement. (NSDI), 2013, pp. 29–42.
2. G. R. Cantieni, G. Iannaccone, C. Barakat, C. Diot, and P. Thiran, "Reformulating the monitor placement problem: Optimal network-wide sampling," in Proc. ACM CoNEXT Conf., Mar. 2006, pp. 1725–1731.
3. H. Xu, Z. Yu, C. Qian, X.-Y. Li, and L. Huang, "Minimizing flow statistics collection cost using wildcard-based requests in SDNs," IEEE/ACM Trans. Netw., vol. 25, no. 6, pp. 3587–3601, Dec. 2017.

4. A. Dixit, F. Hao, S. Mukherjee, T. V. Lakshman, and R. Kompella, "Towards an elastic distributed SDN controller," ACM SIGCOMM Comput. Commun. Rev., vol. 43, no. 4, pp. 7–12, Oct. 2013.
5. J. Xie, D. Guo, Z. Hu, T. Qu, and P. Lv, "Control plane of software defined networks: A survey," Comput. Commun., vol. 67, pp. 1–10, Aug. 2015.
6. A. Tootoonchian, M. Ghobadi, and Y. Ganjali, "OpenTM: Traffic matrix estimator for OpenFlow networks," in Proc. Int. Conf. Passive Active Netw. Meas., 2010, pp. 201–210.
7. S. R. Chowdhury, Md. F. Bari, R. Ahmed, and R. Boutaba, "PayLess: A low-cost network monitoring framework for software defined networks," in Proc. IEEE Netw. Operat. Manage. Symp. (NOMS), May 2014, pp. 1–9.
8. C. Yu, C. Lumezanu, Y. Zhang, V. Singh, G. Jiang, and H. V. Madhyastha, "Flowsense: Monitoring network utilization with zero measurement cost," in Proc. Int. Conf. Passive Active Netw. Meas., 2013, pp. 31–41.
9. Z. Su, T. Wang, Y. Xia, and M. Hamdi, "CeMon: A cost-effective flow monitoring system in software defined networks," Comput. Netw., vol. 92, pp. 101–115, Dec. 2015.
10. H. Tahaei, R. Salleh, S. Khan, R. Izard, K.-K. R. Choo, and N. B. Anuar, "A multi-objective software defined network traffic measurement," Measurement, vol. 95, pp. 317–327, Jan. 2017.
11. A. C. Myers, "JFlow: Practical mostly static information flow control," in Proc. 26th ACM SIGPLAN-SIGACT Symp. Principles Programm. Lang., 1999, pp. 228–241.
12. V. Mohan, Y. J. Reddy, and K. Kalpana, "Active and passive network measurements: A survey," Int. J. Comput. Sci. Inf. Technol., vol. 2, no. 4, pp. 1372–1385, 2011.
13. S. Sezer et al., "Are we ready for SDN? Implementation challenges for software-defined networks," IEEE Commun. Mag., vol. 51, no. 7, pp. 36–43, Jul. 2013.
14. X. Zhang, E. Tune, R. Hagmann, R. Jnagal, V. Gokhale, and J. Wilkes, "CPI2: CPU performance isolation for shared compute clusters," in Proc. 8th ACM Eur. Conf. Comput. Syst., 2013, pp. 379–391.
15. T. Koponen et al., "Onix: A distributed control platform for large-scale production networks," in Proc. OSDI, Vancouver, BC, Canada, 2010, pp. 1–6.

16. A. Tootoonchian and Y. Ganjali, "HyperFlow: A distributed control plane for OpenFlow," presented at the Internet Netw. Manage. Conf. Res. Enterprise Netw., San Jose, CA, USA, 2010.
17. P. Berde et al., "ONOS: Towards an open, distributed SDN OS," presented at the 3rd Workshop Hot Topics Softw. Defined Netw., Chicago, IL, USA, 2014.
18. J. Medved, R. Varga, A. Tkacik, and K. Gray, "OpenDaylight: Towards a model driven SDN controller architecture," in Proc. IEEE Int. Symp. World Wireless, Mobile Multimedia Netw., Jun. 2014, pp. 1–6.
19. S. H. Yeganeh and Y. Ganjali, "Kandoo: A framework for efficient and scalable offloading of control applications," presented at the 1st Workshop Hot Topics Softw. Defined Netw., Helsinki, Finland, 2012.
20. B. Pfaff, B. Lantz, and B. Heller, "OpenFlow switch specification, version 1.3.0," Open Netw. Found., Tech. Rep., 2012.
21. B. Pfaff, B. Lantz, and B. Heller, "OpenFlow switch specification, version 1.1.0," Open Netw. Found., Tech. Rep., 2011.
22. R. Ahmed and R. Boutaba, "Design considerations for managing wide area software-defined networks," IEEE Commun. Mag., vol. 52, no. 7, pp. 116–123, Jul. 2014.
23. S. Sharma, D. Staessens, D. Colle, M. Pickavet, and P. Demeester, "Fast failure recovery for in-band OpenFlow networks," in Proc. 9th Int. Conf. Design Reliable Commun. Netw. (DRCN), 2013, pp. 52–59.
24. S. Zander, G. Armitage, and P. Branch, "A survey of covert channels and countermeasures in computer network protocols," IEEE Commun. Surveys Tuts., vol. 9, no. 3, pp. 44–57, 3rd Quart., 2007.
25. J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, A. R. Curtis, and S. Banerjee, "DevoFlow: Cost-effective flow management for high-performance enterprise networks," presented at the 9th ACM SIGCOMM Workshop Hot Topics Netw., Monterey, CA, USA, 2010.
26. D. Sünnen, "Performance evaluation of OpenFlow switches," Ph. D dissertation, Dept. Inf. Technol. Elect. Eng., Swiss Federal Inst. Technol. Zürich, Switzerland, 2011.
27. D. A. Patterson and J. L. Hennessy, Computer Organization and Design MIPS Edition: The Hardware/Software Interface, 5th ed. New York, NY, USA: Elsevier, 2014, p. 800.

28. P. Megyesi, A. Botta, G. Aceto, A. Pescapé, and S. Molnár, "Challenges and solution for measuring available bandwidth in software defined networks," Comput. Commun., vol. 99, pp. 48–61, Feb. 2017.
29. K. Yi and Y. H. Ding, "32-bit RISC CPU based on MIPS instruction fetch module design," in Proc. Int. Joint Conf. Artif. Intell., 2009, pp. 754–760.
30. R. Izard. (2016). Fast-Failover OpenFlow Groups. [Online]. Available: https://goodlight.atlassian.net/wiki/display/goodlightcontroller/How+to+Work+with+Fast-Failover+OpenFlow+Groups
31. A. Botta, A. Dainotti, and A. Pescapè, "A tool for the generation of realistic network workload for emerging networking scenarios," Comput. Netw., vol. 56, no. 15, pp. 3531–3547, Oct. 2012.
32. Mininet. (2015). Mininet Version 2.2.1 ed. [Online]. Available: http://mininet.org/overview/
33. OpenvSwitch. (Feb. 17, 2017). Open vSwitch Version 2.5.5. [Online]. Available: http://openvswitch.org/releases/NEWS-2.5.2
34. BigSwitchNetworks. (2016). Floodlight v1.2. [Online]. Available: https://goodlight.atlassian.net/wiki/spaces/goodlightcontroller/pages/24805419/Floodlight+v1.2
35. M. Karakus and A. Durresi, "A survey: Control plane scalability issues and approaches in software-defined networking (SDN)," Comput. Netw., vol. 112, pp. 279–293, Jan. 2017.
36. C. L. Lim, A. Moffat, and A. Wirth, "Lazy and eager approaches for the set cover problem," presented at the 37th Austral. Comput. Sci. Conf., vol. 147, Auckland, New Zealand, 2014.
37. M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," ACM SIGCOMM Comput. Commun. Rev., vol. 38, no. 4, pp. 63–74, 2008.

6. TABLES OF FIGURES

7. TABLES OF TABLES

8. TABLES OF ABBREVIATIONS

CDN	Content Delivery Network
POP	Points of Presence
BSP	Broadcast Storm Problem
RADs	Random Assessment Delays
VANET	Vehicular Specially appointed Networks
MANET	Mobile Ad hoc NETwork