Peered Content Delivery Network through Request Routing Peering System for Video-On-Demand

By

Kee Hwaai Shian

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

In partial fulfillment of the requirement

for the degree of

BACHELOR OF INFORMATION TECHNOLOGY (HONS)

COMMUNICATIONS AND NETWORKING

Faculty of Information and Communication Technology

(Perak Campus)

AUGUST 2012

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Dr Lau Phooi Yee who has given me this bright opportunity to engage in a Future Media Internet project. It is my first research project and it has been a very important experience for me in future career. A million thanks to you.

Finally I must say thanks to my parents, my family and my friends for their love, support, help and continuous encouragement throughout the course.

DECLARATION OF ORIGINALITY

I declare that this report entitled "**Peered Content Delivery Network through Request Routing Peering System for Video on Demand**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : ___________________________

Name : Kee Hwaai Shian

Date : 13 August 2012

ABSTRACT

Content Delivery Network (CDN) is an intermediate layer of existing internet infrastructure that helps to efficiently deliver the increase of multimedia content from content provider to a large number of geographically located clients (or end users). However, CDN providers have a limitation on geographically located surrogate servers, setup and maintenance of new surrogate because it is costly and inefficiently. Therefore a peered network has to form among CDN providers to increase the coverage to serve the increasing number of clients. Request Routing is an essential component for the peered CDNs because it enables the client's request to be rerouted to another CDN which may serve the client more efficiently. However, the current algorithm does not address the performance of request routing in peered CDN with Video on Demand (VoD). In this paper we propose a hierarchical architecture for peered CDNs to handle failure of intermediate nodes in between CDNs by having a backup of intermediate nodes and a request routing algorithm which makes decision on directing the request to the peered CDN or to be handled by the local CDN. The propose algorithm will depend on the bandwidth (connectivity) between the surrogate server and the client, the workload on the surrogate server and the content availability in the surrogate servers to determine the most suitable surrogate server to serve the client. The concept for peered CDN is to maximize the chance to serve the client. When one of the CDN receives a client's request about content that it does not has, it will redirect the request to another surrogate server which may or may not be in another CDN to serve the client rather than discard it. Apart from that, from the peering between CDNs, the number of surrogate servers would be increased, hence the workload could be shared among the surrogate servers and the chance to have closest surrogate servers to serve the client would be higher. In this project, we had performed some simulations where we vary the number of clients or surrogate servers and observe the change on performance based on the workload balance and closest surrogate server policy. In conclusion, workload balance policy is suitable to handle peak hour traffic and closest surrogate server policy is suitable to handle non peak hour traffic.

Table of Contents

LIST OF FIGURES

LIST OF ABBREVIATIONS

IPTV	Internet Protocol Television
CDN	Content Delivery Network
VoD	Video on Demand
IP	Internet Protocol
DVD	Digital Versatile/Video Disc
VCR	Video Cassette Recording
DNS	Domain Name System
NAT	Network Address Translation
URL	Uniform Resource Locator
IETF	Internet Engineering Task Force
CDNi	Content Delivery Network Internetworking
ALTO	Application Layer Traffic Optimization

1.0 Introduction

Since the widespread adoption of broadband access, everyone is able to access the Internet easily at any place, any time and on any devices. With this trend of Internet growth [7], comes the demand of delivery services of multimedia content. This demand shifted from the traditional way of delivery video content technologies to video delivery through Internet such as video conference over IP, IPTV and Video-on-Demand (VoD). With easy access of broadband technology, the number of Internet users skyrocketed. However, this leads to the increase of latency over the Internet. This proves to be a challenge for the delivery of video content over the existing Internet as most of them are in real-time transmission.

In this project, our main focus is on VoD. Firstly, what is VoD? VoD is an interactive multimedia service which delivers video content to the end user/subscribers [6]. The video will be delivered to the users with low start up delay and full interactivity. It provides end user with the possibility of watching the videos of their choice at any time and controlling them, such as pause, rewind, fast forward and etc. Users are able to pause and start the video any time as they want and fast forward/reverse search of hot scenes.

Formerly, a VoD system requires one to set up a set top box from a television to a digital video recorder, portable media player that is provided by the services provider. The video contents will then be delivered to the user through cables and satellites. The contents were downloaded whilst the users were watching the video and interacting with the video content. With the growth of broadband users, VoD were then opened up to possibility of being IP based.

There are a few issues that need to be addressed in the implementation of VoD services. Firstly there should be a huge storage capacity to accommodate the vast amount of media contents. Secondly, the bandwidth required for deliver the video content should be large enough to support delivery to multiple users at the same time. Not only on the provider's side, should users also have a minimum requirement of bandwidth. Thirdly, the video should be able to stream at any time and any place. Users that subscribed to the services should be able to watch the video at any time they want at any place that has a broadband connection. Lastly, users should be able to interact with the media content. However, existing providers don't provide most of them.

1.2 Background

As the Internet grow, so does the unmanageable amount of traffic which causes the lost of end user requests in the Internet. Then in 1998, Content Delivery Network (CDN) was proposed [1]. CDN is a network with the concept of geographically and strategically placing the surrogate servers (which is also known as proxy server) which has replicas of the same contents or services to various surrogate servers which could act as the origin server to deliver the contents. This aims to reduce the issue of overloads at the origin server and to increases the performance (efficiently, availability, reliability and accessibility) and scalability of the network.

Each end user request will be redirected to the best surrogate server. The best surrogate server is determined by calculating certain metrics, such as the bandwidth availability, connection availability and so on. A CDN consists of surrogate servers (which act as replica of the origin server), a distribution system (which handle the replication of the contents and services), a request routing system (which will redirect the end user request to the best surrogate server) and accounting system (which use to record down the usage of end users).

There are few request routing methods that is used by existing CDN providers to redirect the user's request to the best surrogate server. These include DNS based Request Routing, Transport Layer based Request Routing, Application layer based Request Routing, Content layer based Request Routing and NAT based Request Routing [3]. The first and second Request Routing is most commonly used by CDN providers [8]. DNS based Request Routing operate just like a normal DNS server. However, it takes several parameters into account, such as the availability of server as consideration to select the most appropriate server to serve the user. In Application based request routing, the request routing algorithm will redirect user's request URL address to the appropriate surrogate server to serve the user by considering several parameters that will affect the performance. Commonly these two types of approaches are used to perform request routing redirection.

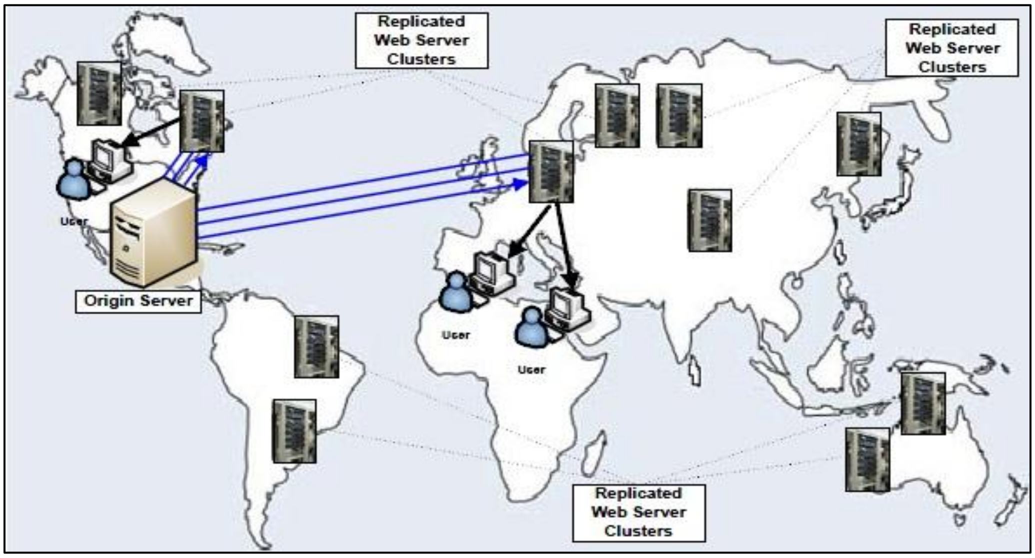


Figure 1-2-1: Example of CDN architecture

With the ongoing growth of Internet Technology, the delivery of contents and services can include text, voice, video and multimedia. The contents and services in the Internet become increasingly important and complex, especially for multimedia contents which requires real-time transmission. With the amount of congestion faced by the Internet, real-time guarantee is not easy. It requires an efficient and continuous delivery of contents. Therefore, most of the content and service providers turn to CDN providers for the delivery of their contents. However, CDN providers (such as Akamai) are not able to cover the whole world. Hence, end users who are out of coverage but have requests for content, they will have to be redirected to the suitable surrogate server even if the surrogate is far from the end user. This would then affect the performance. On the other hand, setting up more surrogate servers to increase coverage is not rational, due to it being very costly to do so.

To overcome this problem, CDN peering was proposed by some researchers, such as IETF proposed CDI, which stand for Content Delivery Internetworking [10] and proposed VO (Virtual Organisation)-based peering CDN [1]. In peering CDN, each of the existing CDN providers is connected to other existing CDN providers to increase the coverage of a single CDN. The end user's request will be redirected by the origin CDN to the CDN that could provide best delivery of content to the end users.

To reduce response time and achieve workload balance among the surrogate servers, request routing algorithm is important for a CDN. However, most of the request routing algorithm proposed did not focus on peering CDN or VoD service. For VoD, it requires interactive function during real-time streaming. However with existing request routing in CDN, such functionality is not there.

1.3. Scope and Objectives

The project aims to come up with a hierarchical network architecture and a request routing algorithm that supports VoD in terms of peering CDNs. The outcomes would provide a much more suitable approach to select the most appropriate CDN to deliver the service. Furthermore, the selection on CDN/CDNs would be base on the same request routing algorithm to select appropriate surrogate servers to deliver the VoD services to users.

The algorithm would consider the availability of contents and connections of the server to determine the appropriate surrogate servers. The end user request would be redirected to a few of the appropriate surrogate servers within a single CDN and also their peering CDN to provide delivery of video. The selected surrogate servers will handle the delivery of video content.

Before proposing the algorithm, e would study and analyse the existing algorithms in term of performance. From there, we would proceed to improve any weaknesses found with the existing algorithm. After that, we would look into further modification that could be done on to the current approach/algorithm in order to suit peering CDN and VoD. We would then propose an algorithm that would perform under both CDN and inter-CDN to support the delivery of VoD.

We would simulate the proposed algorithm in term of delivering video content to show that the algorithm is able to improve the performance of VoD on peering CDN.

The project scopes include:

1. Understanding the performance of current algorithm in request routing system.
    a. Study the existing design of the request routing system
    b. Analyse the existing request routing algorithm.

2. Propose a new algorithm for request-routing system

a. Study the peering system/algorithm for VoD services
b. Analyse the strength/improvement of the proposed approach.

3. Study and analyse the performance of the proposed approach
    a. Simulate different scenario for video delivery using the approach.
    b. Analyse the result of (a)
    c. State the improvement of the (b)

2.0 Literature Review

In [2], the authors proposed a fuzzy-based decision approach for supporting multimedia content request routing in CDN. The fuzzy-based decision is used to determine the appropriate replica server based on the bandwidth availability, hard disk availability and connection availability of the replica server. The author proposed this approach to overcome the other two request routing algorithms which are Random Choice (RC) and Round Robin (RR). As the authors point out, the previous two approaches have high amount of drop rate because they overlook bandwidth availability. The proposed approach by the authors required the request routing system to perform three steps in order to collect reliable information about the surrogate servers.

A Crisp Value was calculated by taking into account several parameters such as bandwidth availability, hard disk availability, and connection availability of the replica server. The CV is a combination of the current status of the replica server to determine whether the server is able to serve the requested task. For example when bandwidth availability, hard disk availability and connection availability is high, the attached value will be strongly recommended (Y). However, in the above paper the authors did not clearly addressed how the proposed approach is able to help enhance the performance of a CDN toward delivery multimedia content such as video. The authors merely stated the proposed approach is able to help enhance the selection of the most appropriate replica servers. Furthermore, the author merely improved drop rate rather than the performance of the CDN in term of delivery content.

TE (Traffic Engineering)-Friendly Content Delivery Request Routing was proposed in [5], the authors proposed a solution which consists of request routing by using anycast address and Multi-protocol Label Switching packet forwarding system. The authors proposed the use of MPLS to achieve TE objectives of the network. In TFRR system, the client address is treated as a network prefix network to achieve scalability. The client request was sent by MPLS label switching to an anycast address that represented a whole CDN and then the CDN server. The anycast is used for the purpose of load balancing among the surrogate server with a round robin order. Other than request routing mechanism, the author proposed TFRR algorithm to take traffic load value as input for server selection to achieve load balancing among the servers. However the proposed algorithms had an overhead request within the CDN due to use of anycast addresses to forward the request into the CDN for selection of surrogate server. Furthermore, the authors assumed that the contents were fully replicated within every surrogate servers that is in the CDN which did not support the fair use of storage.

In [8], the authors had proposed architecture for large scale of CDN and a scheme designed for content routing and content query. The proposed architecture is used to form a star-based connection among large-scale of CDNs. A few surrogate servers were grouped as a CDN cluster and one of the servers as the representative of the cluster. Each of the clusters is connected to form a star based architecture for the delivery of service/content. Requests are routed to other cluster of CDN surrogate servers and if the request fails to serve the local cluster. The request routing scheme proposed by the authors also discusses about using semi-hashing instead of hashing or directory based. Compared to hash and directory based schemes, the authors' proposed scheme had the advantages of smallest routing overhead and fully distributed without any central directory server. However the proposed architecture did not address the problem of failures on the representative server. The whole cluster would lose connection with the neighbour's cluster if the representative server was down.

In [9], the author had proposed a draft idea about use of ALTO (Application Layer Traffic Optimization) to improve the performance of CDNi request routing. ALTO is an approach for guiding the selection of resources providers to retrieve a requested resource. The ALTO server will provide sufficient information to guide the process of selection of resource providers. ALTO server is usually used in P2P application which provides the service provider the information when making selection on retrieving a resource. The author believes that the ALTO can potentially improve the performance of the existing request routing by reducing the steps in resolving the request in both DNS based and Application based. The upstream CDN will determine the best entry point of a downstream CDN based on the cost map from the downstream CDN and the upstream CDN, the ALTO map in upstream CDN and the location of the requested client. However, the author did not provide any evidence for improvement that can be achieved on existing scheme.

3.0: DESIGN AND ANALYSIS

3.1 Methods/Technologies Involved

First, we would simulate the proposed approach by using different kind of scenarios for request routing for video on demand services to show that the contribution of the proposed algorithm toward the video on demand service. In order to show differentiates/improvements from the previous approaches and algorithms, we would also simulate the previous approaches and algorithms as a comparison with our proposed approach. By doing the simulation of the previous approaches and algorithms, we could further understand the potential room to improve. However, there is few type of simulator (simulation software) in the commercial are available for this projects, which are OPNET and OMNeT++.

The parameters which would determine the network performance includes throughput, delay, latency and respond time. Throughput is a measurement of how much amount of packets in term of bytes had been flow in or out the network per second. Secondly, delay is a measurement of the time of packet to reach a destination after it sent out. Latency is a measurement about the condition of a network, whether the congestion is high or low. With higher latency (higher congestion) in a network, the packets travelled through the network would have higher delay. Respond time is the time used to receive the response packet after the request packet is sent. This could be affected by the latency of the network, the respond time of the server and etc. The simulation will base on this few measurements to observe the differences between each scenario in the simulated networks.

3.2 Request Routing Algorithm for Peer CDN

The ideal of the algorithm is to route the requests from the users to the most suitable surrogate server based on the proposed three parameters: (1) content availability; (2) works load of the server; (3) the bandwidth availability. We apply these three parameters to determine which surrogate server would be selected to be the most suitable to serve the users. The performance of the surrogate server is taken into consideration rather than consider the physical distances in between the serving surrogate servers and the users. Even when there exist a short physical distance in between the surrogate servers and the users; it would not be able to guarantee an efficient delivery of content. This is due to the conditions of the surrogate server, i.e. with a busy conditions, it would not be able to serve the users efficiently compared to other surrogate servers located further away from the users, though not busy. Therefore, this could be done by taking the workload of the surrogate server as one of the parameter. With the slower response time, the surrogate servers could be considers as though they are in busy condition.

Another parameter included is the availability of the content in the surrogate server. When the requested content did not contained in the surrogate server, we will assume that that particular surrogate server did not have the ability to serve the request (users). Therefore, the algorithm would skip the particular surrogate servers and search for the next surrogate server. The last parameter for this algorithm is the availability of bandwidth for the surrogate server. There would have two kinds of scenario: (1) surrogate server would be considers as slow response time when it is busy or; (2) the bandwidth in between the surrogate server and the user is inefficient (low bandwidth that might cause by the user is downloading from other resources currently). The first scenario will be taken care by the workload parameter of the proposed algorithm. On the other hand, the second scenario is refers to the availability of the bandwidth of the client site and surrogate server site. When the bandwidth is not enough for transfer the content, it would route the request to another surrogate server that have better bandwidth. This might be caused because of the surrogate server is transmitting content. Therefore it would affect the available bandwidth.

When the user's request arrived at the CDN side, the proposed algorithm will take the above three parameters as criteria to find the most suitable surrogate server and forward the request to that particular surrogate server. However, if there did not have any surrogate server that fulfills the requirement on these three parameters, the proposed algorithm will seek further for the suitable surrogate server on another CDN. The algorithm will repeat the above process again until it obtained the most suitable surrogate server. Therefore, when there did not exist any suitable surrogate server that able to serve the request, the request would be forward to its peer CDNs to look for most suitable surrogate server. Figure 1 shows the concept of peer CDN.

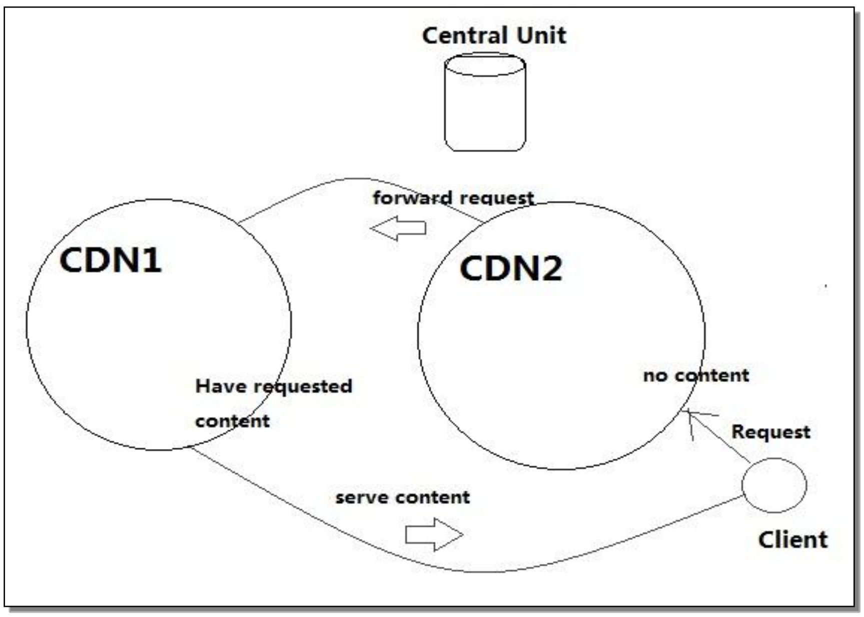


Figure 3-2-1: The Concept of Peer CDN

Besides that, the proposed algorithm would treat the peered CDN(s) as part of its network. The peered CDN provider would bring a group of surrogate servers to join into another CDN. In another words, there would be a big CDN that containing more than one CDN and each of them would have one request routing system but they work together and appear transparent to the end users. There would be an agreement about sharing resources in between those CDN providers before they merge their CDN together. It will taken care about the request routing within a CDN and also inter CDN. When the request could not able to serve by its intra CDN, it will work with the request routing system in another CDN to seek for the suitable surrogate server in the peer CDN. For request routing between CDNs, the request routing system in each CDNs will join as a request routing bond to serve the request that need to pass beyond a single CDN. It is likes few group of people that join/work together for a project and each of them elected a leader to deal with others group leader to achieve the goals. The request routing system in each CDN is just like the leader in the above example.

3.3 CDNsim

CDNsim is an open-source, modular and open-architecture of parallel discrete-event trace-driven CDN simulation system which based on OMNET++ simulation environment and the INET framework. INET framework is an extension of OMNET++ to provide basic network protocols functionality such as TCP/IP to build up a network. Furthermore, CDNsim uses OMNET++ for simulate the network operation such as TCP/IP transmission between nodes and discrete-event scheduling to control the event start and end time. CDNsim is developed to solve the limitations of previous simulators that just able to simulate real-world implementation with limited details in controlling such as use of static estimate for the network transfer time which is not realistic to represent the real time scenario. CDNsim is developed to provide realistic simulation for CDNs by simulate the event between surrogate server, the transmission using TCP/IP protocol and the main functionality of CDN. The TCP/IP protocols are taken out from INET framework and had been modified by the authors of CDNsim to suit CDN behaviors. The main purpose of CDNsim is to provide a testbed for research community and CDN developer to perform CDN evaluation and experimentation.

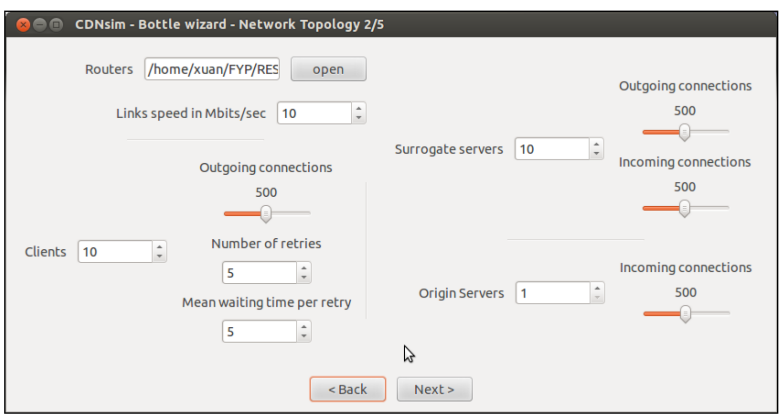


Figure 3-3-1: Sample of the options in CDNsim bottle wizard

In our project, we choose CDNsim as the simulation tool is due to: (1) it is open-sources; (2) it comes with the basic simulation model for CDN and; (3) it provides the flexibility to modify the simulation model. Therefore, we could use it to suit its network topology into our project. CDNsim had already provided the base simulation model for CDN comparing to OMNET++ which need to build the simulation model start from the ground. With CDNsim, we just need to change the parameters such as the number of surrogate server and the number of clients. Moreover, we are also able to control the number of objects and the content in each surrogate servers for the simulation. Therefore, we are able to assume the amount of content in each CDN. With these flexibilities, we could able to control the content (objects) within each CDN. Hence, during the simulation when the request reached CDN that does not have the requested object, the request would be re-routed to others peered CDN. Besides that, we are able to modify the simulation topology to suit our case. For instance, we can adjust the placement of the network devices such as clients unit, servers unit (origin and surrogate) and routers by editing the network diagram configuration file. Furthermore, CDNsim is much flexible in controlling the parameters and modification of the network topology comparing to OPNET++. With OPNET++ IT guru edition, we do not have much flexibility to change the parameters due to limited features are offered in IT guru edition. Furthermore, OPNET++ IT guru edition just offered some trial functionalities which might not be able to suit into this project situation. Extra functionalities required addition payment which will be costly to this project due to budget limitation.

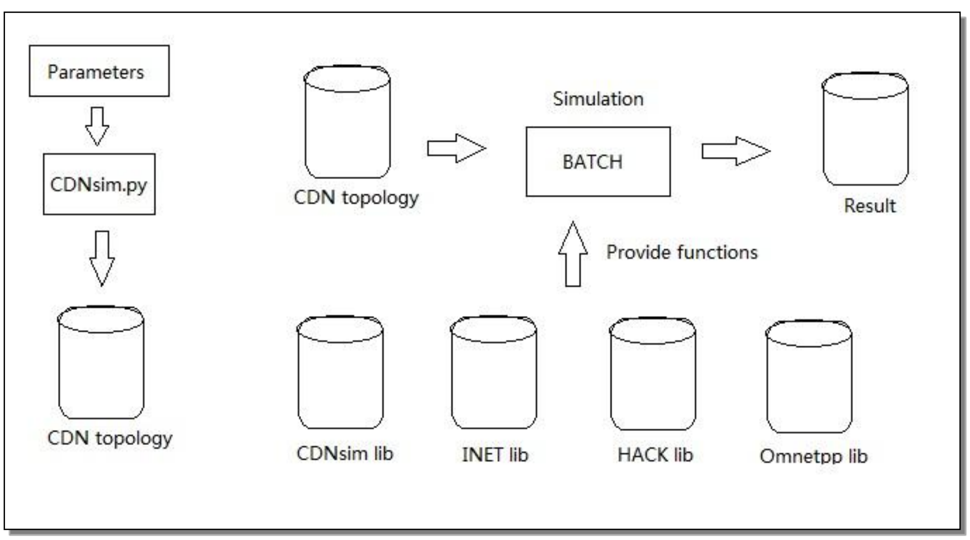


Figure 3-3-2: Processes of CDNsim model

To make the simulation model to suit the project case, we have to adjust part of the network topology. From the topology build from execute the CDNsim topology builder, we found that it is possible and flexible to make modification on the topology setting such as relation between all of the nodes, the amount of objects in the network, and the content that should be in the surrogate servers. With these flexibilities, we could modify the network topology to suit whatever case that we needed. Furthermore, CDNsim provided few policies on surrogate server management which are cooperative closest surrogate server, non-cooperative closest surrogate server, cooperative random surrogate server and cooperative workload balance among surrogate server. Cooperative mean the surrogate server will send the request to others surrogate server when it does not have the content more than send the request to the origin server and vice versa for non-cooperative. With closest surrogate server policy, the request will be forward to the nearest surrogate server for the clients. With workload balance policy, the system will balance the workload of each surrogate server and forward to the surrogate server that is in least workload. Random surrogate server policy is just forward the request randomly to a surrogate server. To suit our parameters, we would use cooperative surrogate server policy and cooperative workload balance among surrogate server as our policy for all of the simulations in this project.

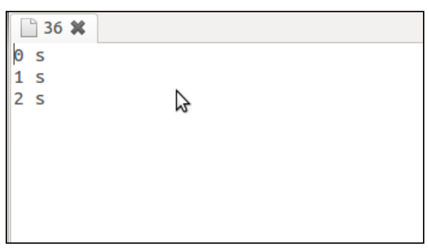


Figure 3-3-3: Sample of Server Content before the simulation

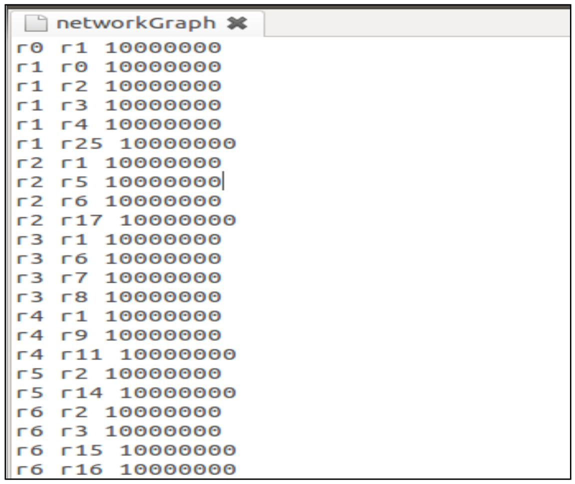


Figure 3-3-4: Example of the network graph

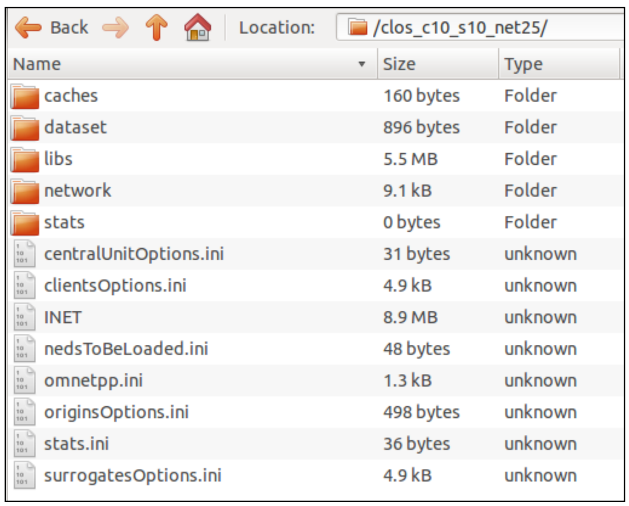


Figure 3-3-5: Example of the Network Topology for CDNsim

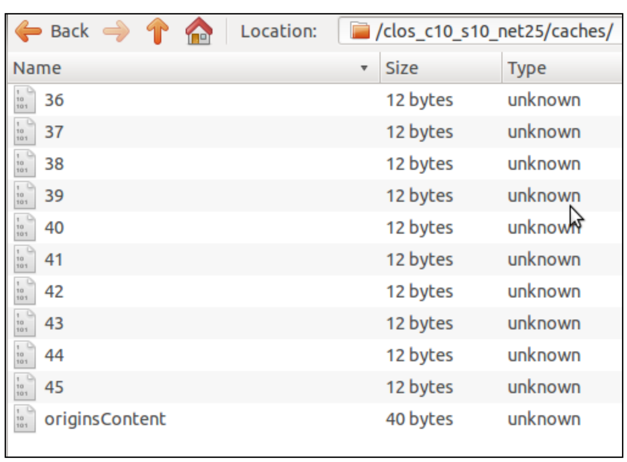


Figure 3-3-6: Example of the Cache Setting for Surrogate Server

However, we should make some assumptions before the simulation process. Firstly, we should make assumptions for each simulation case that how many surrogate servers is under each CDN. For instance, server 1 to server 5 are under CDN 1 and server 6 to server 10 are under CDN 2 to able us to simulate that there are request is flowing between 2 CDNs when the client request cannot be serve by the single CDN, in another words the single CDN does not have the object that requested by the client and the request will not serve by the origin server (this is due to CDN is aimed to reduce the workload for the origin server). Furthermore, we should also assume that the CDN providers have to sign up an agreement that indicate each of their CDN is joining together and will be working together to serve much more users before the peering between CDN started. The agreement is to agree up on the two CDN providers have to merge into a single CDN. There will have a request routing system normally running in single CDN but it will treat two of t he CDNs as one CDN and providing the request routing service in between both CDNs. In CDNsim, there is a central unit node that controlling all of the surrogate servers and route the request from the client. The central unit will treat the peer CDN as part of its CDN and serve the controlling and routing service. Following is a summary of the assumptions that should be make for each of the simulation case.

1. Assume range of server ids are for each CDN.
2. Assume that each peer CDN will be joined together and under control by a single central unit.

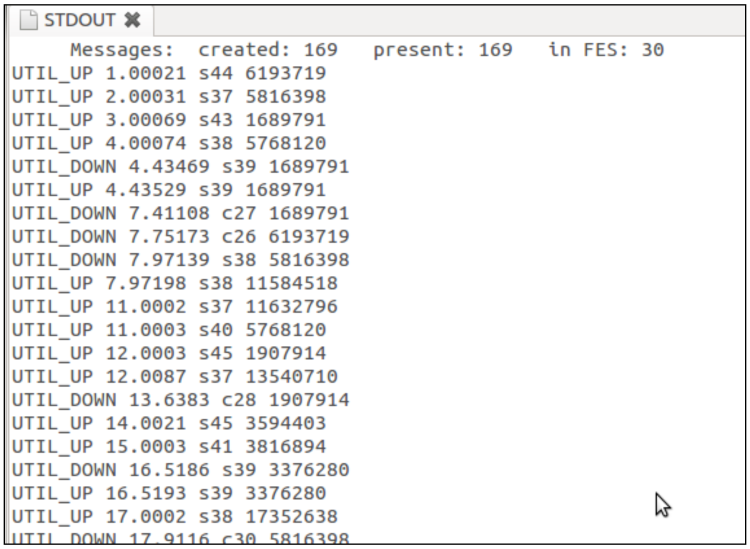

```
STDOUT
    Messages:  created: 169   present: 169   in FES: 30
UTIL_UP 1.00021 s44 6193719
UTIL_UP 2.00031 s37 5816398
UTIL_UP 3.00069 s43 1689791
UTIL_UP 4.00074 s38 5768120
UTIL_DOWN 4.43469 s39 1689791
UTIL_UP 4.43529 s39 1689791
UTIL_DOWN 7.41108 c27 1689791
UTIL_DOWN 7.75173 c26 6193719
UTIL_DOWN 7.97139 s38 5816398
UTIL_UP 7.97198 s38 11584518
UTIL_UP 11.0002 s37 11632796
UTIL_UP 11.0003 s40 5768120
UTIL_UP 12.0003 s45 1907914
UTIL_UP 12.0087 s37 13540710
UTIL_DOWN 13.6383 c28 1907914
UTIL_UP 14.0021 s45 3594403
UTIL_UP 15.0003 s41 3816894
UTIL_DOWN 16.5186 s39 3376280
UTIL_UP 16.5193 s39 3376280
UTIL_UP 17.0002 s38 17352638
UTIL_DOWN 17.9116 c30 5816398
```

Figure 3-3-7: Example for transmit and receive event of the simulation

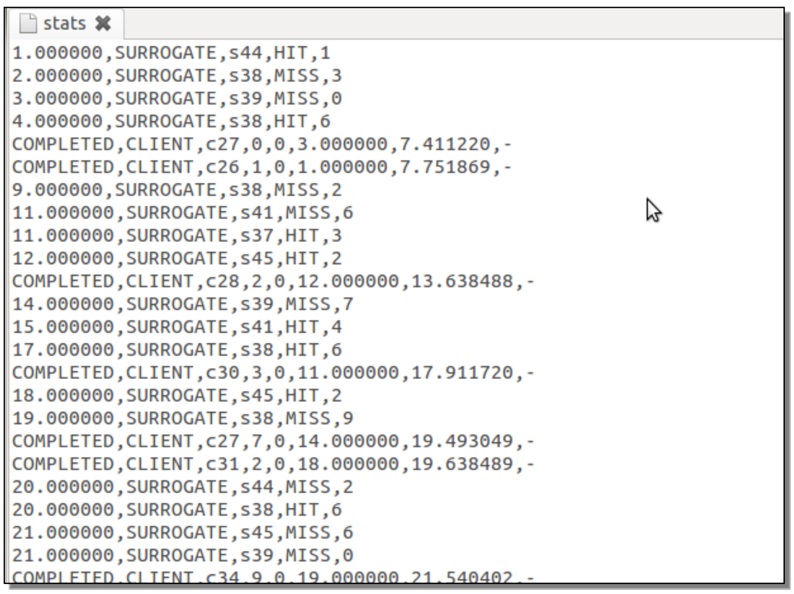

```
stats
1.000000,SURROGATE,s44,HIT,1
2.000000,SURROGATE,s38,MISS,3
3.000000,SURROGATE,s39,MISS,0
4.000000,SURROGATE,s38,HIT,6
COMPLETED,CLIENT,c27,0,0,3.000000,7.411220,-
COMPLETED,CLIENT,c26,1,0,1.000000,7.751869,-
9.000000,SURROGATE,s38,MISS,2
11.000000,SURROGATE,s41,MISS,6
11.000000,SURROGATE,s37,HIT,3
12.000000,SURROGATE,s45,HIT,2
COMPLETED,CLIENT,c28,2,0,12.000000,13.638488,-
14.000000,SURROGATE,s39,MISS,7
15.000000,SURROGATE,s41,HIT,4
17.000000,SURROGATE,s38,HIT,6
COMPLETED,CLIENT,c30,3,0,11.000000,17.911720,-
18.000000,SURROGATE,s45,HIT,2
19.000000,SURROGATE,s38,MISS,9
COMPLETED,CLIENT,c27,7,0,14.000000,19.493049,-
COMPLETED,CLIENT,c31,2,0,18.000000,19.638489,-
20.000000,SURROGATE,s44,MISS,2
20.000000,SURROGATE,s38,HIT,6
21.000000,SURROGATE,s45,MISS,6
21.000000,SURROGATE,s39,MISS,0
COMPLETED,CLIENT,c34,9,0,19.000000,21.540402,-
```

Figure 3-3-8: Example of the event log for the simulation

By using CDNsim, we able to obtain the requests flow between surrogate servers from the stats file under the stats folder and also the STDOUT file. Inside stats and STDOUT files contained the time client completely received the object (or failed to received), the time that surrogate server received the request and checking the content available and the next server that that previous content missed surrogate server forward the request. From these two files, we able to observe the requests were forwarded from the surrogate server to another surrogate server that under different or same CDN. Furthermore, we could also obtain the connections that have opened by each of the servers from the STDOUT file. Besides that, by modifying the files in dataset folder allow us to change the request time and the requested objects by each of the client. Therefore, we would able to control the flow of requests to each of the surrogate servers to suit our simulation scenario. For instance, we might need to burst the workload of a particular surrogate server or we want to force a client to request an object from a surrogate server that did not having it. Furthermore, by modifying the files inside cache folder allow us to control the availability of contents in each of the surrogate server. This is one of the approach of CDN (push based) to reduce the requests that will send to the origin servers and the surrogate act as the origin server to serve the users. By the flexibilities of CDNsim, we able to simulate the case which one of the surrogate server is serving multiple user at the same time by forcing customer to generate request to the same surrogate server (in same CDN) at the same time. This is to observe the request will be flow to the surrogate server in another CDN when the nearest surrogate server (in another CDN) is in busy condition.

4.0 Simulation and results

4.1 Case Study 1: Closest surrogate server

For this scenario case, a network topology with 46 nodes and links of 10Mbit/s were adopted. In addition, we used *closest surrogate server* policy for this scenario. We placed the origin server randomly at any part of the network due to origin server was not our main concern. The topology has two CDN with each CDN has 5 surrogate servers, 10 clients and 25 routers as the backbone network for the simulation. In this scenario, we assumed that server 36 till server 40 as CDN 1 and server 41 till server 45 as CDN 2. Moreover, there are only 10 objects in the simulation to let the simulation result has more clear view of the request flow from each CDN. We used both push-based and pull-based outsourcing strategies in this scenario. Before the simulation start, we had pushed certain objects into each surrogate server by editing the files inside the cache folder. This is to simulate the client request will be router to another CDN when the current CDN did not have the object based on closest surrogate server policy. The description of the pushed objects and the network topology are as below.

CDN 1: 0, 1, 2, 3, 4, 5, 6, 8

Server 36 with object id – 0, 1, 2

Server 37 with object id – 1, 2, 3

Server 38 with object id – 4, 5, 6

Server 39 with object id – 1, 6, 8

Server 40 with object id – 3, 5, 6

CDN 2: 0, 1, 2, 4, 7, 8, 9

Server 41 with object id – 1, 2, 4

Server 42 with object id – 4, 7, 9

Server 43 with object id – 0, 8, 9 Server 44 with object id – 0, 1, 4 Server 45 with object id – 1, 2, 7

Figure 4-1-1: Description of the contents in each server

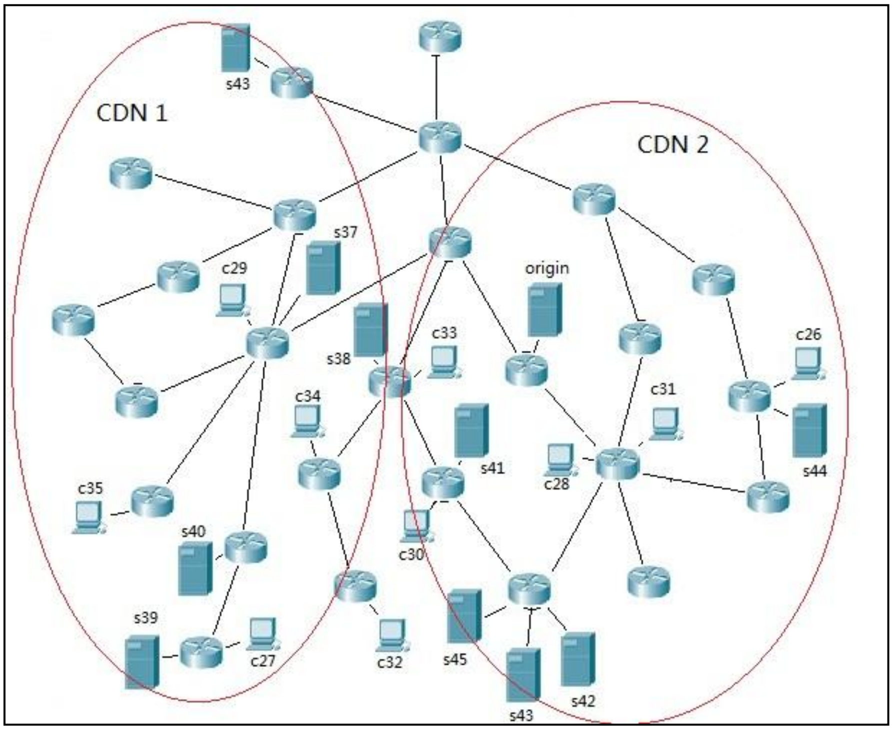


Figure 4-1-2: the network topology for this scenario

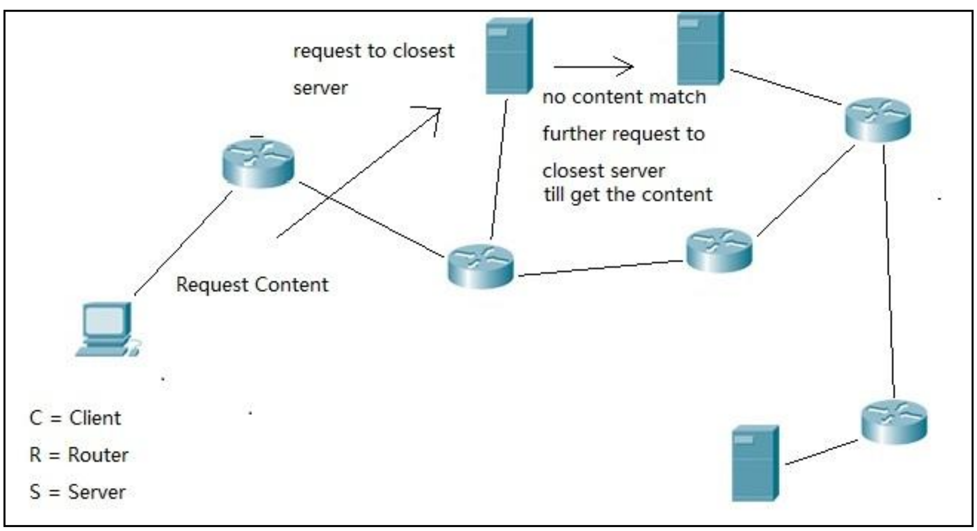
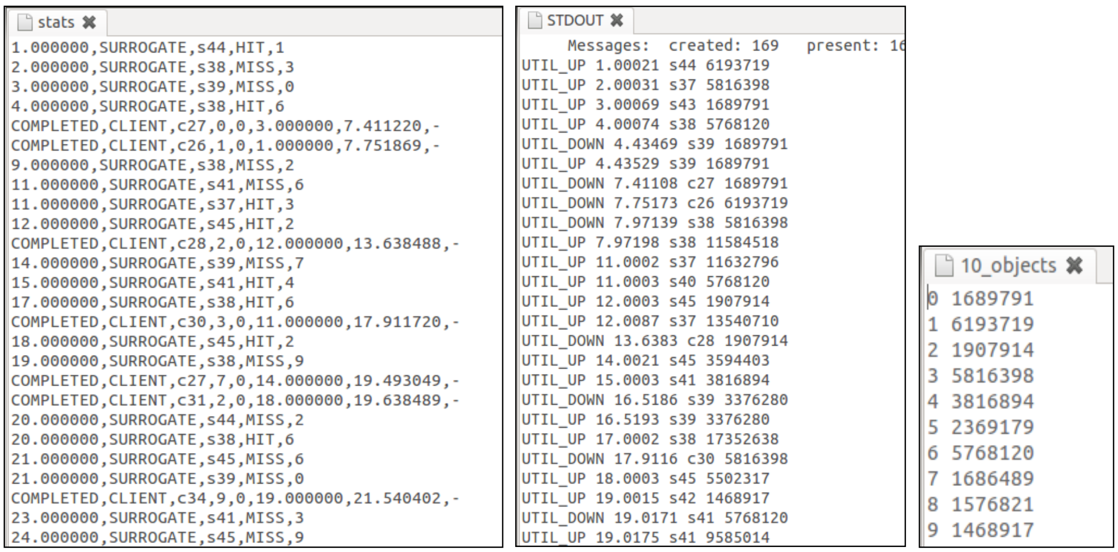


Figure 4-1-3: Concept of closest surrogate server

The following are the result for this simulation scenario. By referring to both STDOUT and stats files, we able to observe that there are request flowed from CDN 1 to CDN 2 when CDN did not have the result. From the result below, we can observe that at time 3, surrogate server 39 get a request of content 0 from the client. However, server 39 did not have the content. It had further requested the content from server 43 which from another CDN due to the closest surrogate first policy. Furthermore, the throughput value for this simulation is *54.3674Mbps* and having a *1%* of failure rate.



(a) (b) (c)

Figure 4-1-4: Traffic result: (a) stats file, (b) STDOUT file, (c) object file

4.2 Case Study 2: Workload balance

For this case study, a network topology with 46 nodes and links of 10Mbit/s that similar with Case Study 1, were adopted. However, we used *workload balance* policy for this scenario. It is based on the workload of the surrogate server when determining the most suitable surrogate server to serve the client. As mentioned in Case Study 1, we placed the origin server randomly at any part of the network due to origin server was not our main concern. And again, we assumed that the network topology consist of two CDN with each CDN has 5 surrogate servers and it contained 10 clients and 25 routers as the backbone network. In this scenario, we again assumed that server 36 till server 40 as CDN 1 and server 41 till server 45 as CDN 2. Moreover, as similar to previous scenario, there are only 10 objects in the simulation to let the simulation result has more clear view of the request flow from each CDN. We used both push-based and pull-based outsourcing strategies in this scenario. Before the simulation start, we had pushed certain objects into each surrogate server by editing the files inside the cache folder. This is to simulate the client request will be router to another CDN when the current CDN did not have the object based on workload parameter of the surrogate server. The description of the pushed objects and the network topology are same as Case Study 1.

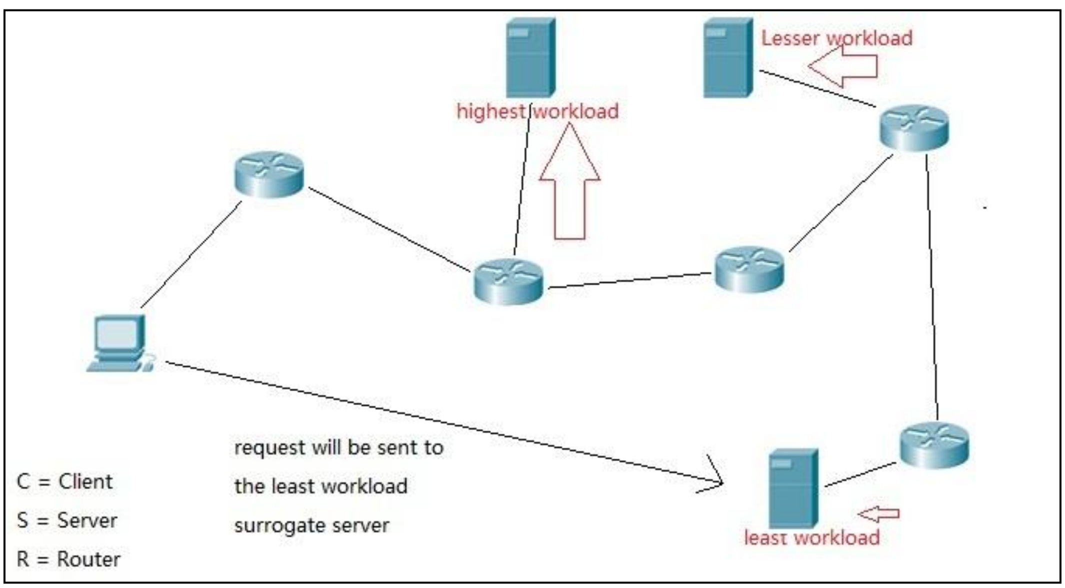
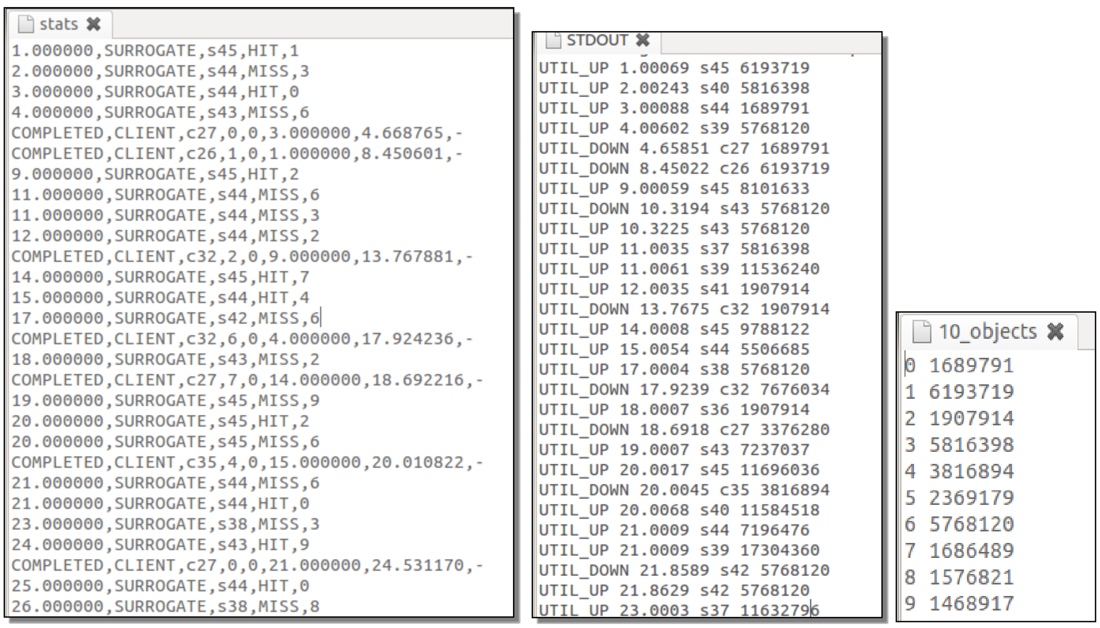


Figure 4-2-1: Concept of workload parameter

The following are the result for this simulation scenario. By referring to both STDOUT and stats files, we able to observe that there are request flowed from CDN 1 to CDN 2 when CDN 1 did not have the result or vice-versa. From the result below, we can observe that at time 2, surrogate server 44 which having the least workload get a request for content object 3 from the client. However, server 44 did not have the content and its CDN did not have object 3. Therefore, it had further requested the content from server 40 which from another CDN due to server 40 had the least workload and with the request object. Furthermore, the throughput value for this simulation is ***62.5019Mbps*** and the failure rate is ***0.5%***.



(a) (b) (c)

Figure 4-2-2: Traffic results: (a) state file, (b) STDOUT file, (c) object list

4.3 Case Study 3: Based on number of clients

For this Case Study, several simulations were simulated to observe the performance against the increasing of amount of clients. There are 50 routers as the backbone of the network, 10 surrogate servers which we assumed each CDN have 5 surrogate servers and 10Mbps for every connection between each of the nodes. Furthermore, there are only 10 different objects in this simulation and we pushed part of the objects to the surrogate servers from each CDN. The objects are partially assigned to each CDN. This is to enable the requests to be flow from CDN 1 to CDN 2 or CDN 2 to CDN 1 when they could not serve the requests.

CDN 1: 0, 1, 2, 3, 4, 5, 6, 8

CDN 2: 0, 1, 2, 4, 7, 8, 9

Figure 4-3-1: Content for each CDN of this Case Study

The following is the network topology that used in this Case Study. It shows the placement of the surrogate servers of each of the CDN. The clients was randomly placed in the network due to the client location is not important for this Case Study and the number of clients was increased in very simulation in this Case Study.

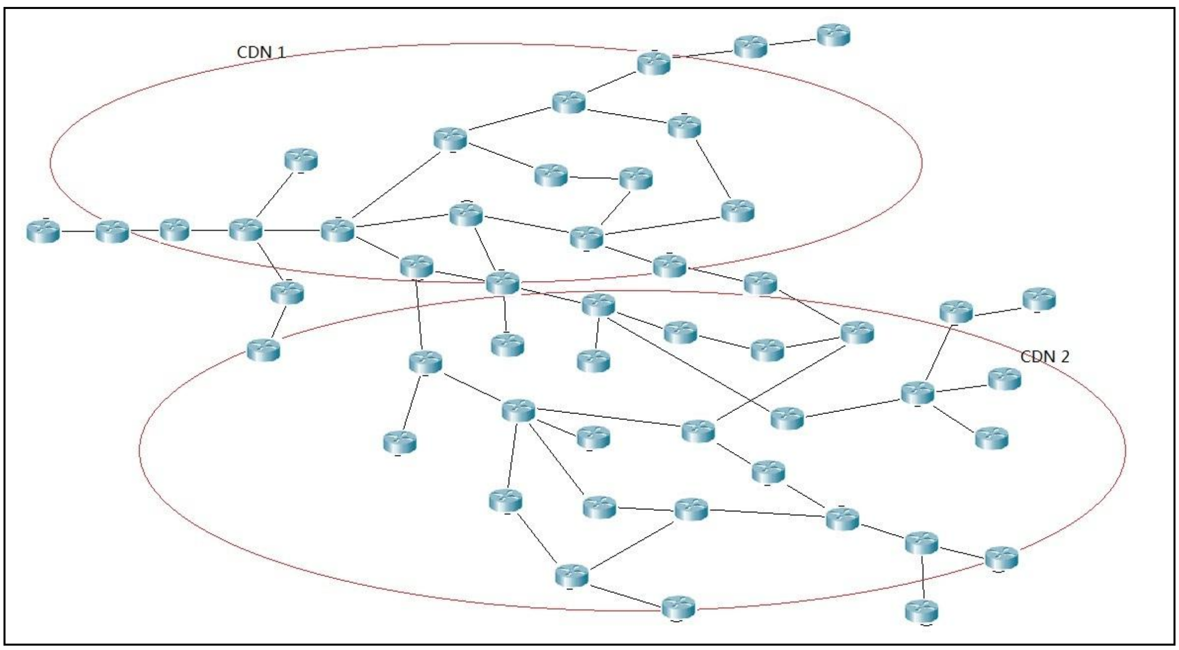


Figure 4-3-2: Network topology for this Case Study

For simulation 1, there are 2 sets of objects requests randomly assigned to the 10 clients for two different scenarios which are closest surrogate server and balance workload among surrogate servers. For simulation 2, which similar to simulation 1 using closest surrogate servers and balance workload as parameters and the number of clients increased to 20 with 3 sets of objects requests randomly assigned to the 20 clients. Thirdly, the number of clients increased to 30 and their objects requests is randomly assigned and the simulation will be done under closest surrogate servers and balance workload parameters. The number of clients increased 10 for each simulation until simulation 6 with 60 clients. All of the simulations are done under two different scenarios which are closest surrogate servers and balance workload among the surrogate severs. The following are the results for the throughput and delay for the two scenarios.

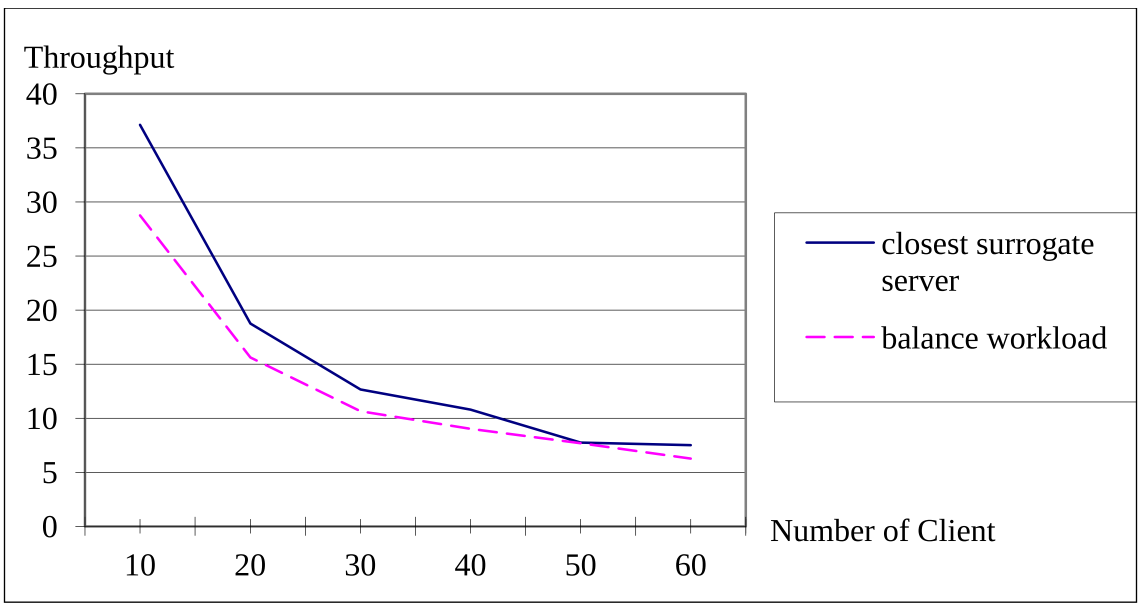


Figure 4-3-3: Throughput level versus number of clients

From figure 4-3-3, we able to observe the relationship in between the number of clients with the throughput level for the network. The increase of clients will pull down the throughput for both scenarios. However, it seems to have least decrease when the client number reached 30 clients and above. The throughput is remained around 8Mbps. From figure 4-3-3, we also able to observe that the throughput performance for closest surrogate servers are better than balance workload when the number of clients are least.

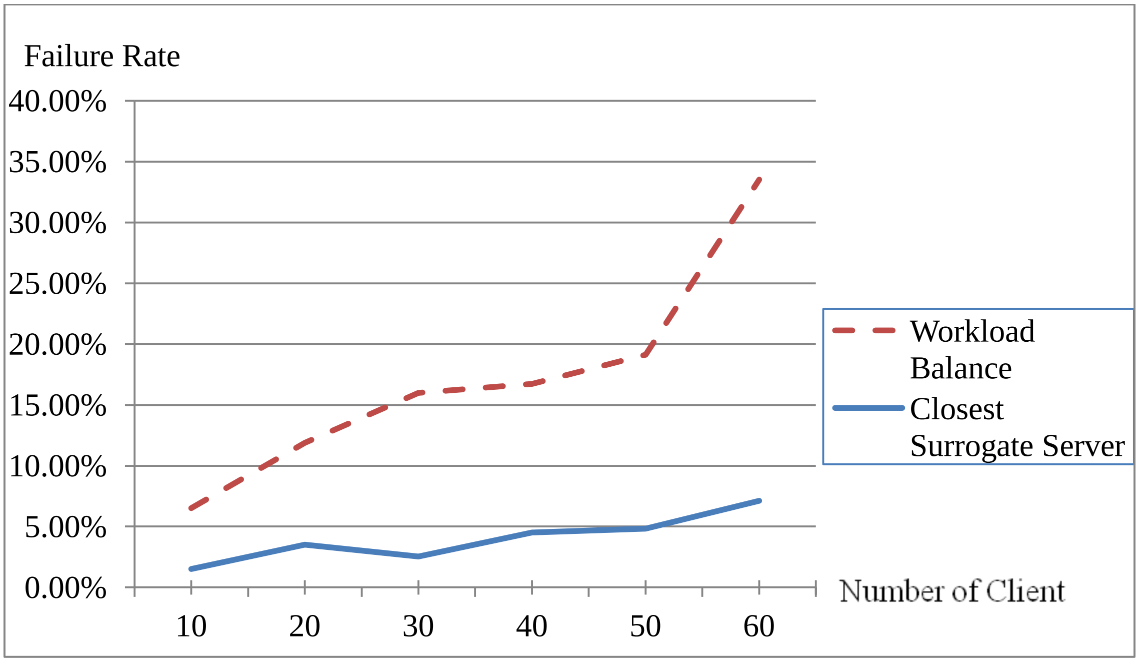


Figure 4-3-4: Failure rate versus number of clients

From figure 4-3-4, we able to observe that the failure rate for workload balance is higher than closest surrogate server. For closest surrogate server, the failure rate was not stabile during the client number in between 10 to 30. However, it changed to an increase form when the number of clients reached 30 and above with a slightly low increasing rate. On the others side, the failure rate is in an increasing form start for 10 clients. However, it reached a much more stabile form when the client number was around 30 to 50 clients. After that it increased dramatically. Therefore, from this figure, we can observe that the increase on number of clients will affect the failure rate when the number of surrogate server is fixed. However, closest surrogate server gave a much better performance compared to workload balance.

4.4 Case Study 4: Based on number of surrogate servers

For this case study, there are 50 routers serve as backbone of the network, 30 clients and 10 objects for the network topology for this Case Study. We had increased the number of the surrogate servers to observe the change of performance. For first simulation, we adopted 10 surrogate servers and the objects are randomly pushed to each surrogate servers for each CDN before the start of simulation. However, we specific the objects contained for each CDN as following.

CDN 1: 0, 1, 2, 3, 4, 5, 6, 8
CDN 2: 0, 1, 2, 4, 7, 8, 9

Figure 4-4-1: Content for each CDN of this Case Study

The following is the backbone of the network and the area of each CDN that used in this Case Study. The clients was randomly placed in the network due to the client location is not important for this Case Study. The surrogate servers were placed in the two areas shown in the following figure. This is to differentiate the surrogate servers for each CDN.

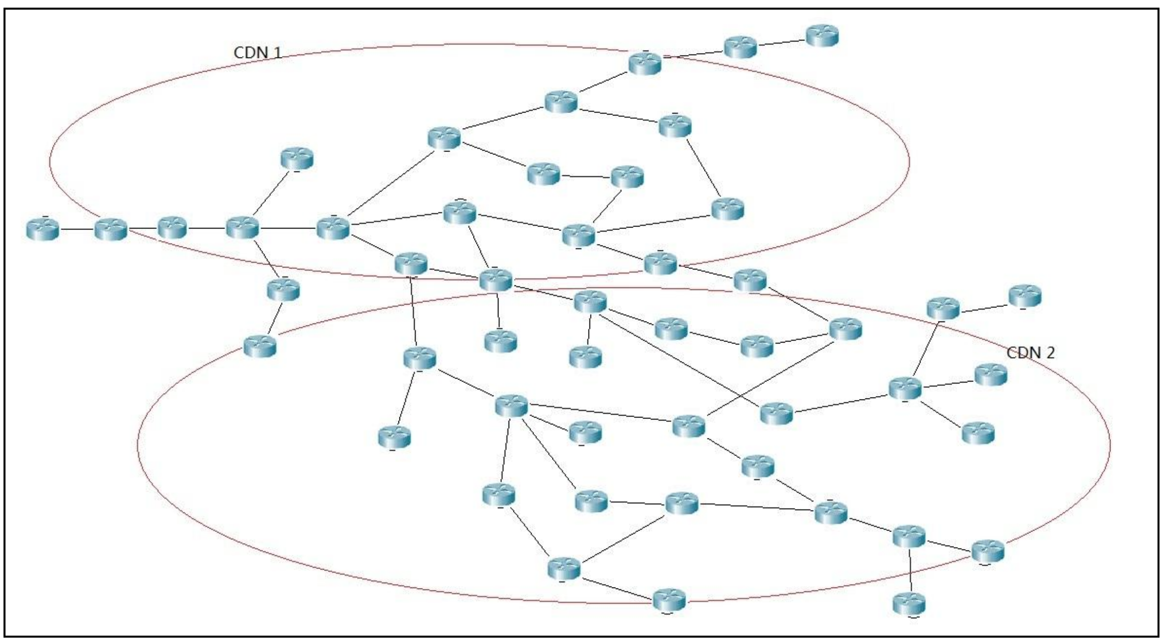


Figure 4-4-2: Network topology and area of each CDN for this Case Study

In the first simulation, we assume that 5 surrogate servers are from CDN 1 and 5 surrogate servers are from CDN 2. In simulation 2, the number of surrogate servers is increased to 20 surrogate servers with each CDN having 10 surrogate servers each. The number of surrogate for the following simulation were increased 10 for each simulation until it reached amount of 60 surrogate servers which are simulation 6. The following are the result for throughput level and delay for each of the simulation under two scenarios which are closest surrogate server and balance workload among the surrogate servers.

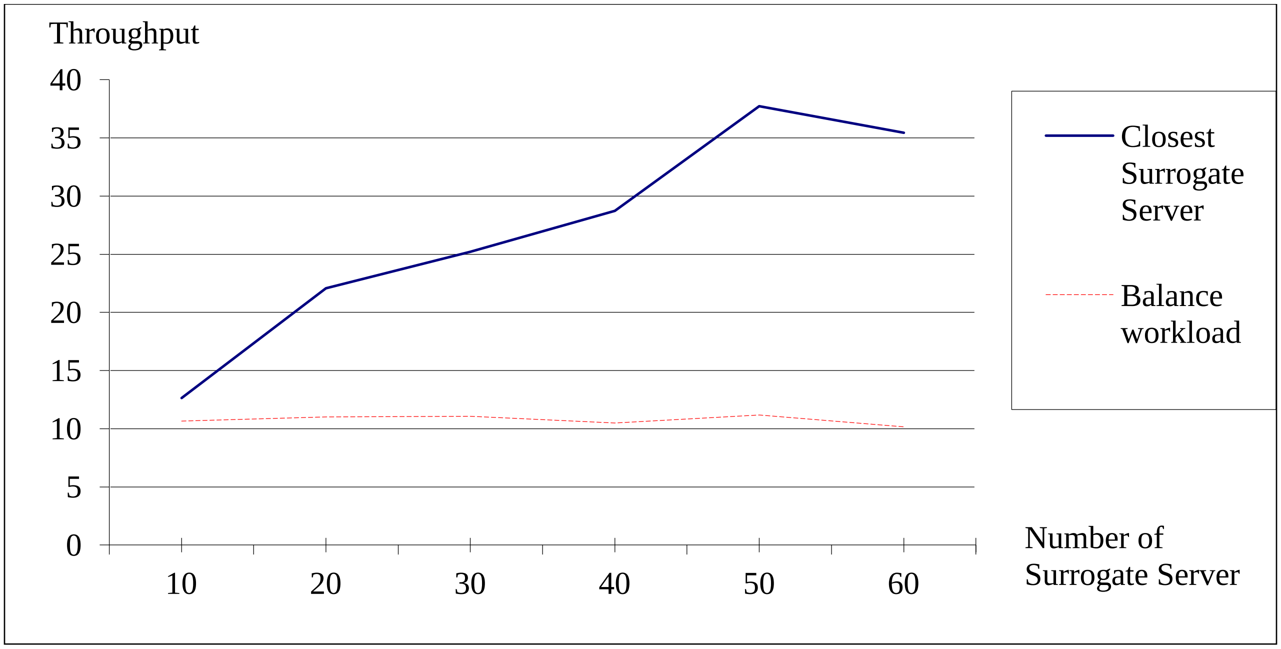


Figure 4-4-3: Throughput level versus number of surrogate servers

From figure 4-4-3, we observe the relationship in between throughput level and the amount of surrogate servers in both scenarios. In balance workload scenario, the throughput level remained slightly constant when the number of surrogate servers was increased. In another words, the increase of surrogate servers not much affected the throughput when balance workload is used. On the other hand, the throughput level increased rapidly when the amount of surrogate servers is increased under closest surrogate server scenario and started to be dropped when the number of surrogate servers increased to a number of 50. In another word, the throughput reached to the peak when there are 50 surrogate server serving 30 clients. Moreover, the throughput for closest surrogate server scenario is affected by the number of surrogate servers.

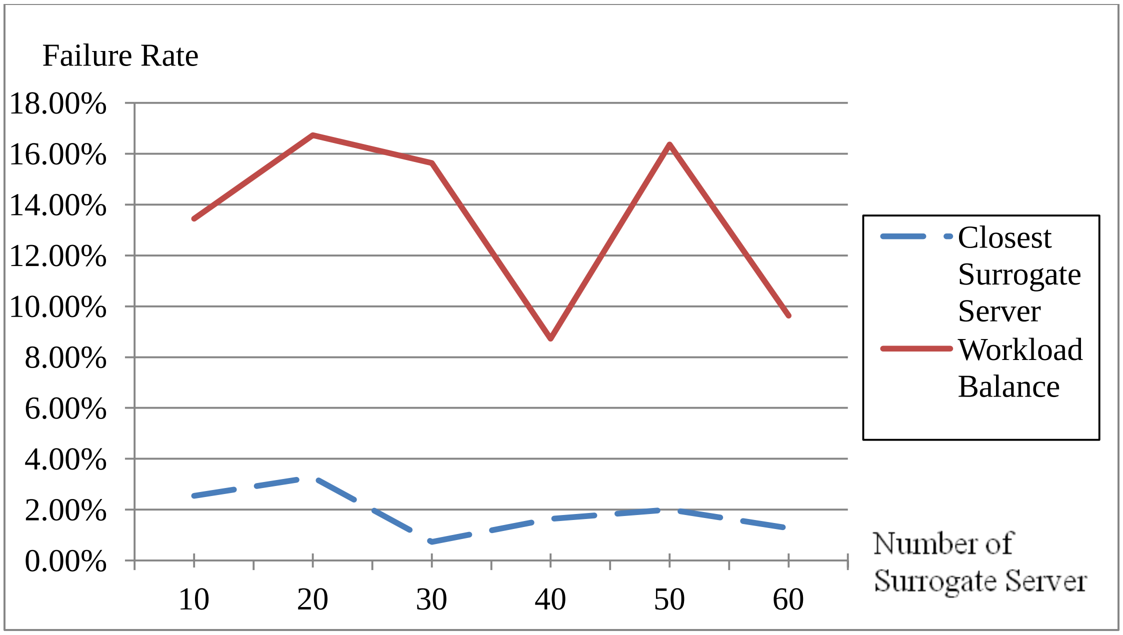


Figure 4-4-4: Failure rate versus number of surrogate Server

From figure 4-4-4, with 30 clients the failure rate is not stabile for both policies. When the surrogate servers increased, the failure rate increased until the number of surrogate servers reached 20 and started to drop till 40 surrogate servers. After that it increased again when 40 surrogate server and started to drop when 60 surrogate servers for workload balance. However, the peak failure rate was not over 17%. This might be caused by the placement of clients and surrogate server. We did not fix the location of clients and the surrogate servers, the placement of both nodes are randomly placed for every simulation experiment. Therefore, this might be the issue that affected the result. On the other hand, the failure rate slightly same with the workload balance for closest surrogate server. The failure rate slightly increased when 20 surrogate servers but after that it started decrease when 30 surrogate servers. After 30 surrogate servers, it changed back to increasing form till 50 surrogate servers and slightly decreased when 60 surrogate servers. For overall, the increase and decrease rate were not high. It reached a peak of 3% and having a plus or minus of 1 % increase or decrease rate.

5.0 Discussion and Conclusion

5.1 Discussion

The following section will discuss about the analysis on the data collected from the simulation experiments in the chapter 4. From Case Study 4.1 which used closest surrogate server policy, the data shown that there are traffic flowed in between surrogate servers that from different CDN. This is useful when the client on CDN 1 did not have the client's deserved content, it able to forward the request to its peer CDN to serve the client. This able to increase the success rate of serving client rather than discard the request. Besides that, the throughput of *54.3674Mbps* is quite high because in Case Study 4.1, the client request is separately for each of the client to observe the traffic in between surrogate server from different CDN. Therefore, surrogate server able to fully utilize the bandwidth of the connection in between surrogate server, backbone and the client to transfer the requested object. Besides that, it had a failure rate of *1%* due to the frequency of requests from the clients was not high. Thus, surrogate servers able to serve the client by handle the request to the closest surrogate server without much workload problem.

From Case Study 4.2 which used workload balance policy, the data shown that the traffic able to flowed to another surrogate server that have lesser workload in different CDN when the first CDN that received the request did not have the content. Thus, the workload from the clients is able to share among the surrogate servers which joined from two or more CDNs. The throughput is *62.5019Mbps* which is higher than Case Study 4.1 which have the similar network topology but different policy. This is due to the client request is sent in different time and the frequency of received request is not high. Thus, with balance workload among the surrogate server able to serve the client with higher utilize of bandwidth compared to closest surrogate server policy. Furthermore, the failure rate of this Case Study is *0.5%* is much lesser than the failure rate in Case Study 4.1. This is due to the requests is forwarded based on workload of surrogate server. This could be much lower the failure rate when the requests is not huge.

From the Case Study 4.3 which used 10 surrogate servers with change on number of clients under two scenarios, closest surrogate server and balance workload. From the throughput graph, the increasing of client will affect the throughput value of the performance of CDN. This is due to the simulation is done based on every single connections in between each node is 10Mbps, therefore when the number of clients increase, the amount of request would be increased as well. Furthermore, we had assumed that the part of the client were sent request at the same time. Thus, when the backbone network is full with others data traffic, then the available traffic to serve a particular client will be lesser. Therefore, the overall throughput would be slightly equal to the average bandwidth of all of the connection in between nodes when the bandwidth in between were fully been used. On the other word, when the number of client is increased over the ability of the bandwidth the average throughput will be lesser than the average bandwidth of the connection in between backbone, surrogate server and the client. From figure 4-3-3, with 30 clients and 10 surrogate servers is the starting point of the throughput become lesser decrease and it is the point that the throughput still slightly above the average bandwidth of the connection for both scenarios. Besides that, the failure rate is increased gradually for closest surrogate server scenario when the number of clients is increased. When the number of clients increased, the number of request is increased as well. Furthermore, in this case study we assume that the scenarios is happened during peak hour, we fixed most of the client sent request for same object at the same time to observe the behavior of surrogate server when high traffic flowed in. Therefore, with closest surrogate server policy, the client requests were forwarded to the nearest surrogate server and the surrogate server did not able to serve the huge number of request at the same time. This is the main reason that caused the failure rate increased when number of client increased. On the other hand, the failure rate for workload balance policy is much lesser compared to closest surrogate server and had much more stabile rate. This is due to the requests is forwarded equally to the surrogate servers based on the workload of the surrogate server.

From the Case Study 4.4 which used 30 clients on 50 routers as backbone to test the performance of both scenarios, closest surrogate server and workload balance. In addition, we want to simulate request traffic that is in peak hour, there would be most of the clients request same object at the same time. From figure 4.4.3, the throughput is increased when the number of surrogate server is increased. This is due to the available of closest surrogate server is increased while the number of client is fixed. On the others words, there are much more surrogate servers to serve a constant amount of request. Therefore, the bandwidth in between surrogate server and client is fully utilized with serving of fewer object compared to case study 4.3. However, with workload balance policy, the throughput is mainly remained the same while the surrogate servers is increased. This is due to the request is forwarded to surrogate server that with lesser workload but in much further logical distance. Therefore, with congestion in backbone network had caused the transfer of data become slower or lower. Besides that, closest surrogate server scenario having lesser failure rate compared to workload balance scenario due to there are much more surrogate server to serve less number of client. On the other hand, workload balance scenario have higher failure rate due to the request are equal forwarded to surrogate server that are further away but with less workload. This might cause the issue of high response time and thus the packet or object does not able to reach the client in time.

5.1.1 Simulations Scenario Comparison

	Case Study 1 Closest Surrogate Server	Case Study 2 Workload Balance	Case Study 3 Based on number of client	Case Study 4 Based on number of surrogate server
1. Efficiency	Throughput of 54.3674Mbps Less clients and requests Bandwidth shared with nearby clients	Throughput of 62.5019Mbps Less clients and requests Requests equally distributed	Number of client increased, throughput for closest surrogate server decreased in higher rate compared to workload balance	Number of surrogate server increased, throughput for closest surrogate server increased in high rate, workload balance not much changed
2.Failure rate	1% of failure	0.5% of failure	Number of client increased, failure rate for closest surrogate server increased in higher rate compared to workload balance	Number of surrogate server increased, failure rate for closest surrogate server much more stable than workload balance

Figure 5-1-1-1: Comparison on simulation scenario

Referring to figure 5-1-1-1, for the case study 1 and 2, it was found that with workload balance policy had better performance when number of clients is lesser and the backbone traffic not congested. For case study 3 and 4, it was found that closest surrogate server policy had better performance when the client requests is huge and workload balance policy had better performance when the number of surrogate server is greater than number of client. When the number of client increased, the peer CDN with closest surrogate server policy serve the client with better throughput and lowest failure rate. Besides that, when there are much more surrogate server than number of client (non peak hour), peer CDN with balance workload policy serve the client better than peer CDN with closest surrogate server policy.

5.1.2 Limitation

Due to the CDNsim had stopped update since year 2009; the GUI part of CDNsim does not able to work. Therefore, we failed to display the source code and also the network topology during the simulation. With lack of GUI platform, we faced several difficulties during the designation process. For instance, when we want to modify the network topology after generated by CDNsim we just able to view the source code of the network topology one by one from the network topology achieve file to adjust the topology to suit our case. The version of OMNET++ is outdated and been modified by the developers of CDNsim. The newest version of the GUI software that needed by the old version of OMNET++ did not support old version of OMNET++ anymore and we not able to found the old version of GUI software that needed by CDNsim. Therefore, we failed to install the old version of OMNET++ that provided by CDNsim developers which lead to no GUI during the modification and display of the source code. Furthermore, the OMNET++ and INET provided by the CDNsim developers had been modified by CDNsim developers, we could not able to move CDNsim to newest version of OMNET++ and also INET. Without GUI part of CDNsim, modification of network topology and simulation model become very time consume.

On the other hand, we also lack of skills and time to modify the whole CDNsim to greatly suit our object case. For instance, we are able to simulate the parameter separately to observe the change of performance of the system we not able to combine few parameters together to observe the more suitable result for our algorithm. Furthermore, the original CDNsim did not provide much information about the traffic flow inside the network during the simulation. It just provided little information of the simulation result. Furthermore, the result log file contained in the simulation result also did not provided much information such as there do not have the details of the time the client sent request and the time the client get the response from the server site. This is very important when we are talk about performance of the network such as delay. Without this information, there will be much more difficult to obtain deserved result.

5.2 Conclusion and future recommendation

In this project, we found that there are no such algorithms that specific in provide content request routing in CDN that suit for VoD. Besides that there are not much algorithms that provide request routing in peered CDN and specific the backup intermediate node in between each CDN when the primary intermediate node which provides the request routing to others CDN is down. The aim of this project is to propose a design of peered CDN architecture and algorithm to specific the request routing in peered CDN to provide request routing for VoD. The proposed algorithm is simulated by used of CDNsim on two policies, which are workload balance and closest surrogate server in four different case studies. From the simulation result, we could conclude that peered CDN is useful to increase the serving rate for a network system. This will also increase the VCR liked feature of VoD become much more reliable. However, the algorithm used for request routing should consider the peak hour and non peak hour when route the request. From the simulation results, workload balance policy is useful during peak hour due to its stability on throughput and failure rate compared to closest surrogate server policy. Besides that, during non peak hour closest surrogate server policy able to provide higher throughput and lower failure rate compared to workload balance policy. With higher throughput and stability of transfer data only able to provide much more reliable VCR liked VoD service. However, further research should be done on combination of these two policies, such as how to switch between policies and what kind of information the algorithm should be determine before the switch of policies. Furthermore, development and upgrade of CDNsim should be done to have much more related and detailed result when simulate case studies that related to CDN. The current version of CDNsim did not provide much information on the traffic result such as latency of the network, delay of the network and etc and the GUI of CDNsim had out of date.

6. References

[1]R. Buyya, A. M. K. Pathan, J. Broberg, and Z. Tari, "*A Case for Peering of Content Delivery Network",* IEEE Distributed System Online, Vol 7, No. 10, Los Alamitos, CA, USA, Oct. 2006

[2]J. B. Chen, and S. J. Liao, "*A Fuzzy-based Decision Approach for Supporting Multimedia Content Request Routing in CDN"*, in: Parrallel and Distributed Processing with Application (ISPA), 2010 International Symposium, Sept. 2010

[3]Md. H. Kabir, E. G. Manning, and G. C. Shoja, *"Request-routing Trends and Techniques in Content Distribution Network"*, in: Proceedings of the ICCIT, Dhaka, Dec. 2002

[4]E. Turrini, "*A Platform for Request Routing in Content Distribution Internetworks*", 7th Cabernet Radical Workshop, Bertinoro (FC), Italy, 13-16 Oct. 2002

[5]M. Jaseemuddin, A. Nanthakumaran, and A. Leon-Garcia, "*TE-Friendly Content Delivery Request Routing in a CDN*", Proc. IEEE International Communications. Vol 1.pp 323-330, June 2006

[6]Y. Dony, "*Video-on-Demand over Internet. A survey of Existing Systems and Solutions*", Facultes Universitaires Notre-Dame de la Paix, Namur, Faculty of Computer Science, academic year 2007-2008

[7]W. H. Dutton, and G. Blank, "*Next Generation Users: The Internet in Britain*", Oxford Internet Survey 2011 Report, Oxford Internet Institute University of Oxford, 1 St Giles, Oxford OX1 3JS, United Kingdom.

[8]J. Ni, D. H. Tsang, I. S. H. Yeung, X. Hei, "*Hierarchical Content Routing in Large-Scale Multimedia Content Delivery Network*", Department of Electrical & Electronic Engineering, Hong Kong University of Science and Technology, Hong Kong, 2003

[9]J. Seedorf, “*ALTO for CDNi Request Routing*”, IETF DRAFT, “*draft-seedorf-alto-for-cdni-00*”, October 19, 2011, expires April 22, 2012

[10]G. Bertrand, E. Stephan, G. Watson, T. Burbridge, P. Eardley, K. Ma, “*Use Cases for Content Delivery Network Interconnection*”, IETF DRAFT, “draft-ietf-dcni-use-cases-03”,January 30, 2012, expires August 2, 2012

[11]*OMNeT++ Network Simulation Framework*, http://www.omnetpp.org/home/what-is-omnet

[12]*Application and Network Performance with OPNET*, http://www.opnet.com

[13]K. Stamos, G. Pallis, A. Vakali, D. Katsaros, A. Sidiropoulos, Y. Manolopoulos, “*CDNsim: A Simulation Tool for Content Distribution Networks*”, ACM Transactions on Modeling and Computer Simulation, Vol. 20, April 2010.

[14] K. Stamos, G. Pallis, A. Vakali, D. Katsaros, A. Sidiropoulos, Y. Manolopoulos, "*CDNsim: A Simulation Tool for Content Distribution Networks*", ACM Transactions on Modeling and Computer Simulation, accepted, Feb. 2009.

7.0 Appendix

7.1 FINAL YEAR PROJECT BIWEEKLY REPORT

Trimester,Year: Year 3 Trimester 2	**Study week no.:** Week 4
Student Name & ID : Kee Hwaai Shian (1001017)	
Supervisor : Dr.LAU PHOOI YEE	
Project Title : Peered Content Delivery Network through Request Routing Peering System for Video on Demand	

WORK DONE First meeting with advisor to discuss on the simulation tools
WORK TO BE DONE Research on title on the Internet and in the library
PROBLEMS ENCOUNTERED Found several possible simulation tool
SELF EVALUATION OF THE PROGRESS Satisfactory progress

______________________________ ______________________________

Supervisor's signature Student's signature

FINAL YEAR PROJECT BIWEEKLY REPORT

Trimester,Year: Year 3 Trimester 2	**Study week no.:** Week 5
Student Name & ID : Kee Hwaai Shian (1001017)	
Supervisor **:** Dr.LAU PHOOI YEE	
Project Title **:** Peered Content Delivery Network through Request Routing Peering System for Video on Demand	

WORK DONE Decision on simulation tool
2.WORK TO BE DONE Study of the simulation tool and how to use of the tool
PROBLEMS ENCOUNTERED The tool is quite complex
SELF EVALUATION OF THE PROGRESS Satisfactory progress

__________________________ __________________________

Supervisor's signature Student's signature

FINAL YEAR PROJECT BIWEEKLY REPORT

Trimester,Year: Year 3 Trimester 2	**Study week no.:** Week 7
Student Name & ID : Kee Hwaai Shian (1001017)	
**Supervisor : ** Dr.LAU PHOOI YEE	
Project Title : Peered Content Delivery Network through Request Routing Peering System for Video on Demand	

WORK DONE Have deeper understand of the simulation tool
2.WORK TO BE DONE Search for input file that might change the parameters for simulation
PROBLEMS ENCOUNTERED The simulation tools does not have GUI form and less information is included
SELF EVALUATION OF THE PROGRESS Satisfactory progress

Supervisor's signature   Student's signature

FINAL YEAR PROJECT BIWEEKLY REPORT

Trimester,Year: Year 3 Trimester 2	**Study week no.:** Week 9
Student Name & ID : Kee Hwaai Shian (1001017)	
**Supervisor : ** Dr.LAU PHOOI YEE	
Project Title : Peered Content Delivery Network through Request Routing Peering System for Video on Demand	

WORK DONE Found the input files for changing the parameter
2.WORK TO BE DONE Search for possible change of parameter that suitable for the project
3.PROBLEMS ENCOUNTERED Not clear about the effect when the parameters changed.
SELF EVALUATION OF THE PROGRESS Satisfactory progress

_________________________ _________________________

Supervisor's signature Student's signature

FINAL YEAR PROJECT BIWEEKLY REPORT

Trimester,Year: Year 3 Trimester 2	**Study week no.:** Week 10
Student Name & ID : Kee Hwaai Shian (1001017)	
**Supervisor : ** Dr.LAU PHOOI YEE	
Project Title : Peered Content Delivery Network through Request Routing Peering System for Video on Demand	

WORK DONE Several simulation results had be done
2.WORK TO BE DONE Simulate the case with related to simulation experiments
3.PROBLEMS ENCOUNTERED The simulation tool did not have GUI, hard to modify and obtain result from text files that in all number and word.
SELF EVALUATION OF THE PROGRESS Satisfactory progress

___________________________ ___________________________

Supervisor's signature Student's signature

FINAL YEAR PROJECT BIWEEKLY REPORT

Trimester,Year: Year 3 Trimester 2	**Study week no.:** Week 11
Student Name & ID : Kee Hwaai Shian (1001017)	
Supervisor : Dr.LAU PHOOI YEE	
Project Title : Peered Content Delivery Network through Request Routing Peering System for Video on Demand	

WORK DONE Results had be obtained
2.WORK TO BE DONE Print out hardcopy
3.PROBLEMS ENCOUNTERED No
4.SELF EVALUATION OF THE PROGRESS Satisfactory progress

__________________________ __________________________

Supervisor’s signature Student’s signature

7.2 Softcopy

7.2.1 Poster

7.2.2 Report

7.2.3 Presentation Slides