APPENDIX – A

CDNSIM – SIMULATION TOOL

Content distribution networks (CDNs) have gained considerable attention in the past few years. Hence there is need for developing frameworks for carrying out CDN simulations. CDNs developed by Stamos et al.(2010) is a modeling and simulation frameworks for carrying out CDN simulations. It has been designated to provide a realistic simulation for CDNs, simulating the surrogate servers, the TCP/IP protocol, and the main CDN functions. The main advantages of this tool are its high performance, its extensibility, and its user interface, which is used to configure its parameters. CDNs developed provides an automated environment for conducting experiments and extracting client, server, and network statistics. The purpose of CDNs developed is to be used as a testbed for CDN evaluation and experimentation. The statistical information stored in log files can be useful in computing the values of performance measures and quantify the performance of CDN. This CDNs developed tool is quite useful to both the research community to experiment with new CDN data management and distribution techniques, and useful for CDN developers to evaluate profits on prior certain CDN installations.

Summary of CDN features

- Cooperative push based content management policy.
- Non-Cooperative push based content management policy.
- Cooperative pull based content management policy.
- Non-Cooperative pull based content management policy.
- LRU cache replacement policy.
- STATIC cache policy.
- TCP/IP networking.
- Wizard for creating self-contained simulations (bottles).
- Utility for executing unattended simulations.
- Utility for automatically generating results’ reports.
- Utility for extracting statistics related to net-utility.
- Utility for converting Apache log files into CDNsim trace files.
- Extensible by implementing modules in the form of libraries.

**CDNsim Implementation**

CDNsim is implemented by extending the preceding architectural components. In this section we present the network topology modeling and the available CDN services.

**CDNsim Network Topology Modeling**

The network topology of CDNsim consists of a set of nodes (generic nodes) which are interconnected via network links. Each node contains a compilation of where the internal structure of each node is described as follows:

**Client**

The client is the initiator of requests to CDN. It contains the request-generation service which preloads the clients’ requests. The requests are fulfilled and the serviced content is retrieved using the consumer unit of the content-transfer service. Clients are assigned to surrogate servers in order to submit their requests using the client-redirection service.
**Surrogate server**

It contains the mixed content-transfer service because it acts both as server and client. Additionally, it includes the surrogate server-redirection service for detecting alternative servers to pull content and a local cache of finite capacity.

**Origin server**

The origin server wraps the serving unit of the content-transfer service and a cache that contains all the available content.

**DNS redirector**

The DNS redirector includes the serving units of all the redirection services. The intermediate router nodes are provided by INET (they are excluded from the generic node architecture). When routers are configured, they are used as “black boxes,” which retransmit network packets according to the current network protocol. The INET library includes options for retry timeouts, retry counts, delays concerning the Address Resolution Protocol (ARP), network datagram sizes, and so on. It should be noted that network nodes are not aware of the content of the packets. We make use of the NED language, which is suitable for modeling node hierarchy and connections, essentially building a network topology. Thus, the surrogate servers and client placement in the network is a matter of providing the desired configuration using NED. Detailed information about the NED language can be found in the OMNeT++ manual. The speed and propagation delay for each link can be modified (the user may use either real or artificial measurements, and it is simulated by the INET framework. TCP/IP is the main protocol used to perform communications between services. TCP/IP and all lower-level protocols (i.e., network layer) are provided by the INET framework.
CDNsim Services

The CDNsim implementation supports the following CDN services.

Client redirection service

This service manages the client redirection process. The clients are redirected to the nearest surrogate server in terms of network distance. This distance metric is based on the Dijkstra algorithm, but it can be extended to include more sophisticated methodologies such as Alzoubi et al. (2007) that include information about the load of each server. The client side of this service runs at the clients, while the server side runs at the DNS redirector. Upon a request, the client is redirected by the DNS redirector to the appropriate surrogate server by advertising the IP address and listening port.

Surrogate server (cooperative/noncooperative) redirection service

This service takes place during a cache miss. Specifically, the surrogate servers use one of the two instances (cooperative/noncooperative) of this service. If the cooperative service is activated, the surrogate servers are redirected through the DNS redirector to the closest surrogate server that contains the requested object. But when the noncooperative service is activated, the surrogate servers are directed through the DNS redirector to the origin server.

Request-generation service

This daemon service runs at the clients. At the beginning of the simulation it loads the corresponding trace file containing the requests to be made. The requests are sorted by timestamp, and each one is scheduled appropriately. The scheduled requests are performed by the client’s content-transfer service. CDNsim does not generate traffic according to some specific
methodology or algorithm. The traffic patterns are defined in the trace file which can be real (i.e., Apache logs) or artificially generated by a third party tool such as Medisyn [Tang et al. 2003]. CDNsim executes the user-specified traffic.

Content-transfer service

This service manages the requests for content (i.e., video, audio, text, HTML pages, etc). The clients send requests to the CDN while surrogate servers attempt to satisfy them. This service implements a set of TCP applications responsible for uploading and downloading files. The number of these applications set the connection capacity of the service, and thus the “strength” of a surrogate server. A generic interface is offered that can be tweaked in order to support services such as VOIP and streaming media. The CDNsim user could use TCP dumps that contain all the TCP traffic of streaming services between peers in the form of a trace file. A TCP dump must be preprocessed in order to fit the CDNsim assumptions and message transmission logic.
APPENDIX – B

WAXMAN MODEL

In Waxman model, the nodes in the graph represent the communication end points, and the link between the end points represents the physical link. Nodes in the graph are randomly scattered on a rectangular coordinate grid. Each node is placed at a location with integer coordinates. Links are added to the network by considering all possible pairs of nodes. The existence of the link is decided using the probability function considering how far apart the two nodes are and how many links are expected to exist in the network. For a given pair of nodes, (u, v), the edge is added with the probability that depends on the distance between two nodes.

In the random graph, edge probability can be given by the following formula:

\[
Pr(u,v) = \mu \exp\left(-\frac{d(u,v)}{\beta \times L_1}\right)
\]

(A1)

Where \(L_1\) is the maximum distance between any two nodes in the graph and \(d(u, v)\) is the distance between \(u\) and \(v\). The values of \(\mu\) and \(\beta\) are chosen from the range of [0, 1]. Parameter \(\mu\) can be used to control the value of the average degree of the random graph. If \(\mu\) becomes larger, there will be more links on the network; that is, the links will become denser. Large values of \(\beta\) increase the density of longer edges relative to small ones. In this thesis \(\mu\) and \(\beta\) values are taken as 0.5.
Sequences of dynamic request for entering the node (surrogate server) are generated according to a simple probability model $P_c(mt)$ (Waxman 1988; Kadirire and Knight 1995), and defined as follows:

$$P_c(mt) = \alpha (n - mt) / \alpha (n-mt) + (1- \alpha) mt$$

(A2)

Where $P_c(mt)$ is the probability that an event is the addition of a node or deletion of the node, $mt$ is the number of nodes in the Dominating set, ‘n’ is the total number of nodes in the network and is $\alpha$ is a parameter in $(0,1)$. The value $\alpha$ is the fraction of nodes in the connection at equilibrium. Sequences of dynamic request for leaving the node are $(1-P_c(mt))$. 