Send Orders for Reprints to reprints@benthamscience.ae

Research on the Application of NS2 Network Simulation Based on Clustering Algorithm

Mao Shengli*

School of Computer, Hubei Polytechnic University, Huangshi 435003, China

Abstract: In this paper, we researched the application of NS2 network simulation based on clustering algorithm. With the development of Internet, more and more information transferred by network. Thus the increasing of bandwidth cannot keep up with the demand of information transmission. The content distribution technique is a novel network technology that efficiently delivers the content to end user on behalf of origin web sites. It works by distributing content to the nearest server, so content will closer to users, and avoid congestion. In this paper, a simulation model of content delivery networks based on NS2 is put forward, and compared with C/S mode, we found that the system performance is more better than C/S.

Keywords: NS2 Network Simulation, collaborative algorithm, clustering algorithm, simulation platform.

1. INTRODUCTION

NS2 is abbreviated from Network Simulator. It is a discrete event simulator which is able to simulate TCP, routing, multi-cast and many other protocols on variable layers over both wired and wireless (local and satellite) networks. It is written in C++ and uses OTc1 (an object oriented exteNSion of Tc1 or Tool Command Language) as s command and configuration interface. NS-2 denotes the second version of NS since it has some substantial changes from NS v1.

In 1989, NS appears as a variant of Real Network Simulator and has experienced substantial changes for these years [1, 2]. The development of NS earned support from the VINT (Virtual INTnet) project funded by the US DefeNSe Advanced Research Projects Agency (DARPA) in 1995 and the CONSER project funded by NSF later. Now, NS is daily maintained by USCISI. Every version update of NS adopts codes contributed by researchers from all over the world due-to its open architecture and good extendibility. From this point of view, NS is a production of all involved researchers. In summary, NS-2 has several characteristics.

A discrete event simulator. The core of it is a din-Crete event scheduler implemented by a C++ class of Scheduler which is in charge of recording the current simulation time, scheduling the events in event queue, creating new events and setting their launching time, etc.

Fruitful components. NN is an object-oriented simulator. In its distinction, a great deal of network entities and protocols have been modeled. Entities like link, queue, packet, node, etc, and protocols Eke TCP, FP, MPLS and ad-hoc compose the components library. Most users can start their work directly by making good use of these components.

Separate object model and split-language programming. NN is written in C++ with an OTc1 interpreter as an end. The simulator supports a C++ class hierarchy (called compiled hierarchy) and a similar OTcl class hierarchy (called interpreted hierarchy) [3] at the same time, and there is a one-to-one correspondence between them. NN utilizes C++ in detailed protocol implementation taking advantage of its running efficiency and utilizes OTc1 in simulation configuration since it can be changed very quickly and interactively although it nuts much slower. Open and free. One can freely download all codes of NN from its web site and extend it as he want.

Most bandwidth resources coNSumed by mass of content, so single network cannot bear. With the increasing demands of new application, content delivery has more challenges. As one of kernel technologies of delivery, content delivery networks are paid more attention by academy and industrial since 1998 [4]. By using distributed cache, load balancing, traffic engineering, etc, NS2 builds a distributed overlay network on internet, in order to push content from information sources to the edge equipment of network. On one hand, user can access the content quickly from the nearest server, by which reducing end to end delay and improving the quality of service. On the other hand, NS2 can break through performance bottlenecks of central server, reduce the flow of backbone network, ease high-throughput content delivery pressure on the backbone network, and increase the system capacity.

2. THE FRAME WORK

There are two servers in NS2, the original server and cache server. The initial file stores in the original server. It is

1874-4443/15

updated by the content provider. The cache server stores copy of content file, which can be official version that user request, and the original server update content file by communicating to cache server.

In short, NS2 is a number of network entities that work together. It copies and caches data in different network, in order to realize transparent of end to end, and improve the efficiency of content distribution. The working principle of NS2 is to avoid bottlenecks and links, which affect the speed and stability of data transmission, to ensure that the content delivery faster and more stable. By judging user proximity and server load, NS2 provides services for user in an efficient way. Compared with traditional content delivery technique, NS2's main characteristics are stressing the importance of content distribution in networks, and content distribution completed by the ICP [5].

A Network Distribution of NS2 Network distribution of NS2 is not complex, from which we can see there are three entities in NS2 system [6]: content provider, NS2 provider and end user. Content provider is the source of resources; it is URL address of the resources, which belonged to original server. NS2 provider is organization or company, provides how to accelerate service. End user is end users, accessing to NS2 provider. In order to copy and cache data, NS2 provider arranges to cache server in different areas. Cache server of NS2 is also named edge server or surrogates. NS2 delivers content to edge server, and edge server caches content, when users request, it can redirect to the closer optimal edge server, who deliveries content to users. Additionally, edge server send billing information to NS2's billing system to upload traffic and basis of billing.

NS2 mainly provides the following services: content storage and management; content distribution between edge servers; cache management; distribution resources of static, dynamic, and streaming media; resume and processing solutions; services of monitoring and measurement, etc. And NS2 mainly contain NS the following four components:

a) Content delivery component: containing origin server and replica server;

b) Request-routing component: redirecting user's request to find suitable edge server, and getting the newest content with distribution component;

c) Distribution component: distributing content from original server to edge server, at the same time, ensuring consistency with the cache;

d) Accounting component: recording user access behavior and use of NS2 servers, for network traffic statistics, as the basis for NS2 network charging, or a third party billing. NS2 contains tcl/tk, otcl, NS, tclcl [7]. TC1L1 is an open scripting language, used to program the NS2; tk is a GUI development tool of tcl, used to help user to develop GUI in a graphical environment; otcl is an extension for object-oriented based on tcl/tk, has its own class hierarcby; NS is the core of package, object-oriented simulator, using octl as a front end, and programmed by C++; tclcl provides interfaces for NS and otcl. In order to observation and analysis of simulation results, NS2 provides optional Xgraphy and Nam. Before simulation, we must analysis the levels designed and simulation. If the existing network components can meet the needs of simulation, to write simulation scripts directly, or to extend [8].

Usually, simulation steps of NS2 as follow:

a) Networking model, configure the network topology.

b) Establishing business model, binding equipment agreement, building communications traffic model, etc., included.

c) Setting trace object. In order to save the simulation process, trace object records changes of network elements or Internet phenomena during simulation process in trace file.

d) Interpreting and performing the prepared otcl script with NS2.

e) Observing and analyzing network simulation running process with Nam or X graph.

f) Adjusting the configuration parameters, do the simulations process again.

Compared to WRR(Weighted Round Robin) and URL Hashing, LARD (Locality-Aware Request Distribution) algorithm has more advantages. For example, the throughout is higher than the other two algorithm in severing static files, cache hit ratio, rate of unused nodes. LARD must have a mapping table of services and back-end service node set. When a request arrives, distributing a server with lightest load. Switch will check load imbalance of request set in Ks, if happened, it will modify mapping table.

T_{low} and T_{high} are the key parameters to determine whether the server is serious load imbalance. If one server's load is higher than T_{high}, and there is at least one server, whose load is lower than T_{low}, then the request will be diverted to the lower one. If a server, whose load is two times higher than T_{high}, request will be diverted to the lowest server in system, whether there has a server whose load is lower than T_{low} or not. If all the servers' load is two times higher than T_{high}, efficiency will be reduced, so the total number of effective connections in system must be limited, and the value of T_{high} and T_{low} must be appropriate set. Supposing the number of servers is n, the largest number of the total effective connections is L, we can get:

$$\tau_{ij}(t+n) = \rho\tau_{ij}(t) + \Delta\tau_{ij} \tag{1}$$

Where ρ evaporation is a coefficient such that $(1-\rho)$ represents the of trail between time t and t+n

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{\ k} \tag{2}$$

Then we have:

$$L = (N-1) * T_{high} + T_{low} - 1 \tag{3}$$

Thus there always has a server, whose load is lower than T_{low}. In order to improve the throughput of system, T_{low}

should be high enough. If the value of T_{low} is T_{high} minus T_{low} is very important obtained, the value of so the value of T_{high} should set as high as possible. Assuming known T_{low}, the maximum of request delay minus the minimum of request delay is D, the average request delay is R, and then we can get:

$$T_{high} = \frac{(T_{low} + D/R)}{2} \quad (4)$$

The fundamental principle of optical burst switching has two aspects: one is aggregation or assembly, i.e., packets from client networks are aggregated and assembled to bursts (large packets) which travel through the OBSN and are disassembled at egress nodes. The other is synchronism, i.e., payloads are not transferred along with their headers like in Optical Packet Switching (OPS), and furthermore, they runs on different channels (wavelengths). The "headers" containing routing and other necessary information are called Burst Header Pocket (BHP) or control packets. The channels they run on are called control channels. The "payloads", i.e., bursts, run on data channels. Usually, a burst is left behind its corresponding BHP for a little while, called offset time. BHPs are processed at core nodes and try to reserve bandwidth for the commit; bursts.

Given a set of ri towns, the TSP can be stated as the problem of finding a minimal length closed tour that visits each town once. We call d_{ij} the length of the path between towns i and j. In the case of Euclidean TSP, d_{ij} is the Euclidean distance between i and j (i.e., $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$). An instance of the TSP is given by a graph(N,E), where N is the set of towns and E is the set of edges between towns (a fully connected graph in the Euclidean TSP). Let $b_i(t)$ (i=1,2,...,n) be the number of ants in town at time t and let $m = \sum_{i=1}^{n} b_i(t)$ be the total number.

The issues of wavelength assignment (WA) and channel scheduling are very important in optical networks, but as we know, even in the latest distribution of NS, multi-Channel link has not been supported yet. So, we have to consider how to simulate a WDM link at first.

Basically, a standard NS link is composed of a queue object, denoted by queue_ , a delay element, denoted by link_, and several trace elements. Among them, queue_ simulates the behavior of output buffer while the bandwidth and transmission delay are simulated by link_ . To keep compatible with NS, we create a new type of mufti-server queue, called WvAssign, to simulate wavelength assignment and order tasks epeci6c for, a WDM lick. And a new C++ class, only de1ay, is created just to simulate the transmission delay without tJte bandwidth. The number of wavelength channels and bandwidth per wavelength are configured by the assign object. WvAssign is also a composite object. It consists of a wavelength classifier, denoted by wvc_ , and several fiber interfaces, denoted by fbh_. wvc_ is an object of a new C++ class, Wave Classifier, dived from the native class Classifier.

A classifier in rrs is such an object class which has several output putts. All elements in fbh_ are installed to the putts of wvc_ indexed from 0, 1,…, wvnwm_1, where wvnum_ is the number of wavelength channels. Note that we do not try to create a completely new type of link for WDM link; we just replace the queue type. Additionally, almost all the mechanisms of OBS core nodes, like channel scheduling, burst forwarding, etc., are dependent on the functions of WvAssign queue and the wavelength classifier inside it.

Considering the main three kinds of Simulation module, we select CGU-III module including five kinds of service types. And install the module into the NS2; achieve the simulation platform for Simulation based on NS2 successfully. On this basis, we build and simulate the Simulation wireless network with four nodes; finally, we descript the simulation results simply, and take rtPS type as an example to analyze the network delay performance of service. It provides the basis for in-depth research of Simulation networks simulation.

3. BURST PROCESSING

In this section, BHP processing and bandwidth reservation are introduced. A new C++ class, BHP Agent, derived from Agent is specially created for generating and processing BHPs.

Originally, three aspects are concerned about the bandwidth reservation: BHP processing, routing and wavelength assignment. For simplicity, we install the BHP Agent onto one port of the Wave Classifier object in WDM link so as to omit the routing part and condensate on the BHP processing. In addition, two queues are added before and behind the BHP Agent object (denoted by hagent_) to simulate the BHP processing more accurately. The queue before the agent is called input queue, denoted by iq_; the other is called output queue and denoted by o1_. The functions of these two queues include:

a) simulate the processing delay dynamically and more accurately;

b) buffer BHPs and decrease the loss ratio of BHPs;

c) modifying the type of iq_ can schedule BHPs differently.

The internal architecture of WvAssign queue for OBS and the procedure of BHP processing. When a BHP arrives, wvc_ first checks if it has been processed (in electrical domain); if not, the BHP will be sent to the port of (wvnum} where ices and hagent_ are installed; if yes, the BHP will be sent to the port of available control channel. The BHP is processed in hagent_. The BHP tells hagent_ when the corresponding burst will come and how large it is, then hagent_ tries to reserve wavelength channel for it by calling several member methods of the wavelength classifier object, wvc_ . if no eligible channel is found, the BHP will be discarded; otherwise, a new reservation record will be appended in a table, called Burst Forwarding Tirble (BFT, which is located in wvc- and serves as rules for burst forwarding. Before BFfPs are processed at hagent_, they are buffered at ices first.

By default, ices is a Drop Tail queue, but one may change it to other types, e.g., Class Based Queue (CBQ), to realize BHP differentiated scheduling which is an important way to implement service differentiation.

Those BHPs which succeed in reserving bandwidth will enter the output buffer oc_ where they wait until any control channel is idle, then the BHPs will be sent back to wvc_ . The difference is they have been marked as processed ones for this time. In fact, the actual bandwidth reservation is implemented in wvc_. It schedules channels by examining the reservation states of all output ports. Two channel scheduling algorithms. First Fit and First Fit with Void Filling (First Fit-VF), have been implemented for the time being and the interfaces have been left for adding new algorithm implementations.

4. SIMULATION AND EXPERIMENTAL ANALYSIS

At present many research institutions have developed the simulation module based on NS2 the most famous module of NS2 are NIST (National Institute of Standards and Technology) developed by the United States and CGU-III developed by Taiwan's Chang Gung University. Both of them are developed independently, so the architectures are completely different. In addition, South Korea KAIST (Korea Advance Institute of Science and Technology) has published WiBro modules yet. The comparison of the functions of three main modules is shown in Table **1**.

Table 1. A comparison sheet of different simulation model.

Module Type	Standards (IEEE802.16)	Qos	PHY
NIST	16d and some 16e Module	Best Effort	OFDM
CGU-III	16d	5 kinds of basic service type	OFDMA
KAIST	16d (based on 802.11)	Best Effort	DSSS

As shown in Table **1**, with the regard of Qos, only CGU-III provides five kinds of basic services, they are UGS, rtPS, nrtPS and BE, the others only support service of BE. It is necessary to consider different types of services in actual simulation, it is better to choose CGU-III module.

CGU-III module is based on IEEE 802.16-2004 standard, at the same time includes the unique ertPS (Extended-real-time Polling Service) service of IEEE802.16e-2005 standard, and has been tested completely under the version of NS2.29. The core of module is constituted by three components; they are CS sub-layer; MAC CPS sub-layer and PHY layer. The external network data received by CS access point is transmitted/mapped to MAC layer data by CS sub-layer, then send the date to public sub-layer. Public MAC sub-layer contains the main function of the MAC layer, the function includes broadband request, connection and maintenance established, etc. The physical layer mainly provides the OFDMA technical support.

The CGU-III module provides a source package, so we can install simulation module in the NS2 easily, CGU-III source package includes three folders, and they are mac-802-16; common and queue. Copy all the files in the folders to the specified directory, and then change the contents of Makefile files, finally re-compile the module to complete the installation. Similar to IEEE 802.11 wireless network simulation module, we can use the tcl scripting language to generate the simulated Simulation network. When we write tcl scripts to set the network we should pay attention to simulation parameters, such as the channel of Mobile Node; the type of Antenna, MAC, LL; the number of nodes and the source and type of service between nodes, etc. After simulation, we can use the name files to render the simulation, and to use trlfile to analyze the performance of Simulation network. For the speed of simulation limited, we just take four nodes, one is BS, others are SS, In addition, we use Wireless Channel; 802_16 of the MAC type, LL's link layer type, omni-directional antenna and the Topological boundaries is set to 600 X 600.

Service source: establish UGS, rtPS, and BE service between the BS and SS, in which the service source of UGS and rtPS using CBR data streams, BE's source business source data stream using FTP.

Queue Algorithm: use the Pri-Queue priority queue algorithm. Routing Algorithm: use AODV routing protocol.

After writing tcl script according to simulation parameters designed, the network simulation begins when NS2 command line starts running. If there is not any error appeared during the simulation, then NS2 will produce two files after simulation, they are not file and the file.

Animation Demo NAM of NS2 uses nom file to rendering simulation by visual approach, it is convenient to intuitively analyze how the data packets transmitted from the source node to destination node. The Simulation network with four notes is shown in Fig. (**1**).

NS2 as a free open source simulation tools in the component library, flexibility and versatility, etc, has more advantages. In this paper, a simulation model of NS2 based on NS2 is put forward; the system mainly consists of source server, edge servers and content redirection system. The basic working principle of the system is: data is distributed to edge servers by data distribution subsystem; user's request is navigated to the local server by redirection system. If there is no request content, local server will get request content from source server.

Source server is site owner, manages site files and distributes content to other server. Source server of one site

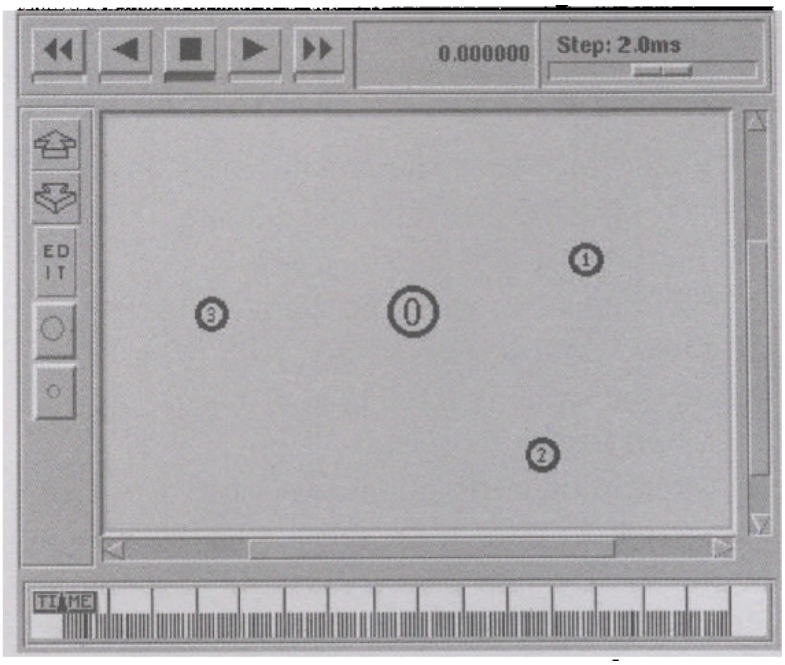

Fig. (1). Simulation network topology.

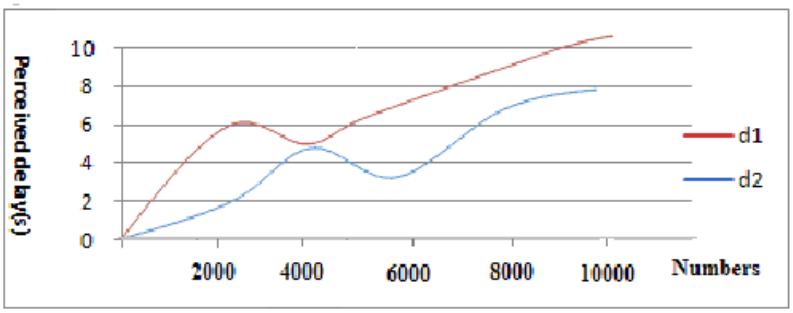

Fig. (2). Perceived delay of use.

must keep communicating with other servers at any time, for other servers are not available all the time and some backup servers are needed to secondary storage. Not only backup server has the contents of site, but also does the edge server. Every site has some edge servers, whose goal is optimize performance. According to the case of request load and QOS, the backup server makes site as closer to the potential users. Last but not least, a redirection server is needed, which based on redirection subsystem of HTTP or DNS. The function is monitor status of source server, edge server and backup server, to locate request to the nearest edge server optimally. In one word, as long as a redirect server is available, the site can be accessed.

Topology of NS2 is six subnets. Building a management system in one subnet is available, which includes distribution system and redirection system, and a cache server in each three-level subnets, so there are 64 cache servers. Business of VOD streaming media on demand service is simulated in the paper, the size of each video file is 30MB, and total number is ten thousand. User requests are distributed to client module evenly; arrival rate is 10r/s.

Poisson distribution and the performance parameters as follows: source server load (bit/s), each cache server load (bit/s), perceived delay of user(s). The result of C/S mode compared with our mode is shown in Fig. (**2**). User's perceived delay of C/S mode is dl, and our model is d2. We can see that dl is bigger than d2 all the time, so our model is better than C/S mode.

CONCLUSION

In this paper, a new NS2-based simulation platform for OBSN, called OBSS, is presented. Its architecture and features are introduced, and a simulation example of TCP

performance over OBSN is demonstrated. The most impressive feature of OBSS is its compatibility with the original architecture of NS2, which enable it very fit for simulating heterogeneous networks in which OBSN plays a role of the transport core. At the same time, interfaces to OBS-related elements have been left beforehand, like fiber delay lines (FDLs), channel scheduling algorithms, etc. Thus, it is very easy for one to extend it further. Therefore, OBSS will be very helpful studying the Optical Burst-Switched networks, especially its end-to-end performances.

CONFLICT OF INTEREST

The author confirms that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

This work is supported by the education project of Hubei Province Office (Project Title: Study on the application of NS2 network simulation based on Cluster).

REFERENCES

[1] Q. Zhao, G. Zhao, and J. Y. Wang, "University education level of information and comprehensive measure Cooperative Research", *Intelligence Magazine,* no. 6, pp. 101-102, 2004.

[2] J. Tai, W. Meleis, and J. Zhang, "*Adaptive Resource Allocation for Cloud Computing Environments under Busty Workloads*", Northeastern University, Boston*, USA*, 2013, pp. 978-987.

[3] Y. Zhang, and L. Qiao, "*Virtual instrument develop environment Lab Windows/CV16.0*," Publishing House of Electronics Industry, Beijing, 2013, pp. 123-130.

[4] J. Liu, P. Bai, and X. Tang "*Virtual Instrument Design Based on Lab Windows/CVI*," Publishing House of Electronics Industry, Beijing, 2003, pp. 111-113.

[5] J. Liu, Z. Shen, and F. Guo, "*Modern Test Technique and System Integration*," Publishing House of Electronics Industry, Beijing, 2005, pp. 398-400.

[6] K. Xu, Y. Liu, and W. Liu, "The design and implementation of security access proxy in database application system," *Computer Engineering and Application,* no.1, pp. 105-107, 2011.

[7] J. Zhang, and Y. Dai, "Design and implementation of network encrypted database system based on proxy," *Computer Engineering and Application,* no. 18, pp. 196-198, 2002.

[8] X. Wang, and S. Wang, "Qin Bo Research on database encryption and verification," *Journal of Xian University of Technology,* vol. 18, no. 3, pp. 263-268, 2012.

Received: September 16, 2014 Revised: December 23, 2014 Accepted: December 31, 2014