Load Balancing
Master Thesis

Panagiotis Stabernas

30/6/2010

Introduction

The WLANs for the past few years have become very popular. Their advantages come in various forms, but the most valuable advantage is the fact that one doesn't need any more wires to connect to the Internet. More over, a number of stations can connect to the access point, wilst in the past the only way to achieve such numbers was through switches.

This evolution however doesn't come without a price to pay. In the case of IEEE802.11g which is the standard used much more than any other, the station can transmit at 54 Mbps. However, depending on the distance between the station and the access point, the rate can vary. In this case, the station can only download at the full capacity of the rate that it connects. This becomes even more complex when we have many stations that connect to the same station, which may cause a great deal of congestion and decline of the download rate of each station.

This problem can be solved, even partially, by the introduction of a second access point within the range of the first, thus reducing the load of the first access point by having stations connecting to the second one. But now we have another obstacle to overcome. And that's the distribution of load among the access points in such a way that it comes to a balanced point.

So here comes the question that describes this problem. How can we distribute this load between the access points in such a way that it is balanced?

Load Balancing

An Introduction To Load Balancing

The problem that we study is the load distribution among access points in the most efficient way. The standard way to connect wireless hosts (laptops, netbooks, smartphones, etc.) to access points is to connect to the access point with the strongest signal. This is a very good way to acquire the best bitrate possible, but it's not the optimal way to distribute the load among the available access points.

For example, let's assume that there are 2 access points according to the IEEE802.11g standard that transmit at the same rate of 54 Mbps and there are 8 stations (laptops, SmartPhones, etc.) that can connect at both stations at the rate of 54 Mbps. All stations download at a rate of 9 Mbps and we have 6 stations that connect to one access point and 2 at the other. In this scenario, let's assume that there is a 9th station that enters the network which can connect at both stations at 54 Mbps. It is quite obvious that it is at its best interest to connect at the access point that has 2 stations connected at it. But there is no extra information that the station can acquire and use besides the fact that both stations are within range and that both have a very strong signal.

This is a very simple scenario, but it clearly shows the weaknesses of the algorithm that we use today to connect to access points. This obviously means that we must introduce an extra piece of information that can be used by all the stations in such a way that the load among the access points will be distributed evenly. This means that we need to study better algorithms than the distance algorithm to achieve the goal of load distribution.

Algorithms

Various researchers have addressed this problem in various ways. We preferred to study the algorithms that are less invasive in both clients and access points. The proposed algorithms are the following:

1. *Distance algorithm*

   With this algorithm the clients connect to the closest to them access point (i.e. to the one with the strongest signal).

2. *Theta algorithm*

   Using this algorithm, we use a metric of ϑ (theta) which is the following:

   $$\vartheta = \sum_{r} \frac{N_r}{r}$$

   where r is the rate and N_r the number of stations connecting to this access point at rate r. With this algorithm the clients connect to the access point with the largest ϑ at which they have the smallest contribution of load according to the equation above.

3. *LBA*

   This algorithm introduces the LBA (Load Balancing Agent) to the access points which is used to create three distinct states at each access point:

   (a) Under-loaded
   The access point can accept clients that either enter the network or are roaming from neighboring access points.

   (b) Balanced
   The access point can only accept clients that enter the network.

   (c) Over Loaded
   The access point can't accept any clients that either enter the network or are roaming from neighboring access points. At this point, the access point forces the handover of current stations (clients) to reduce its load level

Each algorithm proposes an approach that is the less possibly invasive. The *theta algorithm* uses a module implemented on a Linux workstation which uses the SNMP (Simple Network Protocol) to communicate to each access point and obtain the number of stations connected at each access point and the corresponding transmission rate from the access points to the stations. This information is retrieved periodically. Finally, the *LBA algorithm* implements the LBA as a user space process in Linux.

Distance

The *distance algorithm* proposes that the user will connect to the closest possible access point, meaning that the station will connect to the access point with the strongest signal. Although this seems reasonable enough, the scenario proposed on the *Load Balancing* section clearly shows that there is a great deal of flaws in this algorithm. However, it is reasonable to use this algorithm as a point of reference for the other algorithms since this is the algorithm that is used nowadays to connect to an access point.

Theta Algorithm

The *theta algorithm* proposes a very simple but yet very clever approach to this matter. This algorithm suggests that the load of an access point is proportional to the number of stations connected to it at the same rate r.

This algorithm, introduces the metric of ϑ (theta) which can be calculated for each access point according to the following equation:

$$\vartheta = \sum_{r} \frac{N_r}{r}$$

where r is the rate and N_r the number of stations connecting to this access point at rate r.

This metric derives from the calculation of the throughput x of an access point, which is expressed from the following equation:

$$x \approx \frac{1}{a \sum_r \frac{N_r}{r} + b}$$

When using this algorithm the clients connect to the access point with the largest ϑ, at which they have the smallest contribution of load according to the equation above. This metric assumes that the stations take advantage of the full capacity of the rate at which they connect to the access point.

In order to use this algorithm, it is proposed to implement it in a software module, running on a Linux workstation, and use the SNMP (Simple Network Management Protocol) to obtain all the necessary information to calculate the module's ϑ. The station that tries to connect to the network at that time, at

a rate r, will connect at the station at which it creates the largest ϑ, meaning that it creates the smaller load.

LBA

The *LBA algorithm* is having the Load Balancing Agent introduced to the network. In order to decide whether the network needs to be balanced, it uses the load balancing index β, which is calculated by the following equation:

$$\beta = \frac{(\sum B_i)^2}{(n \sum B_i^2)}$$

where B_i is the throughput at the access point i, and n is the number of neighboring APs over which the load is being distributed.

If this load balancing index β equals to 1 then the network doesn't need any balance. The balance index is 1 when all the access points have the same throughput and tends to 1/n when the throughput is the same throughput severely unbalanced.

There are three states that an access point can be:

1. Under-loaded
   The access point can accept clients that either enter the network or are roaming from neighboring access points.

2. Balanced
   The access point can only accept clients that enter the network.

3. Over Loaded
   The access point can't accept any clients that either enter the network or are roaming from neighboring access points. At this point, the access point forces the handover of current stations (clients) to reduce its load level.

The balance index quantifies the balance among neighboring access points, but it is not used to determine the access points state among overloaded, under-loaded or balanced. For this matter, the load at each access point is compared with the average load L calculated as follows:

$$L = \frac{\sum B_i}{n}$$

where B_i is the throughput at the access point i, and n is the number of neighboring APs over which the load is being distributed.

The access points with load below L are declared *under-loaded* and thus can accommodate more traffic. The access points with load above L are divided into two groups: *overloaded* and *balanced*. *Overloaded* access points are those which load exceeds the average by δ, while balanced access points are those which load exceeds the average by less than δ. The rational behind the parameter δ is to

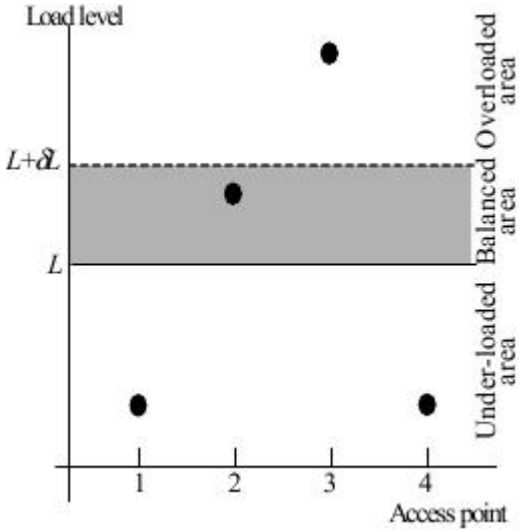

Figure 1: Access points' state with respect to the average load

force the transfer of load from the most overloaded access points to those with load below the average L. The access points that are slightly over the average should not receive more load.

Figure 1 (above) shows an example with four access points. The gray area in the figure highlights the balanced area. AP2 is balanced, while AP3 is overloaded. AP3 should transfer some load to either AP1 or AP4. The parameter δ reduces the number of load transfers, but it also permits some unbalance since only APs exceeding the average load by δ will transfer some load.

Simulation

OMNeT++

Introducing OMNeT++

OMNeT++ is a component-based, modular and open-architecture discrete event network simulator. The most common use of OMNeT++ is the simulation of computer networks, but it is also used for queuing network simulations, and other areas as well.

OMNeT++ is popular in academia for its extensibility (due to its open source model) and plentiful on-line documentation.

OMNeT++ represents a framework approach. Instead of containing explicit and hardwired support for computer networks or other areas, it provides an infrastructure for writing such simulations. Specific application areas are catered by various simulation models and frameworks, most of them open source. These models are developed completely independently of OMNeT++, and follow their own release cycles.

INET Framework

The INET Framework is an open-source communication networks simulation package for the OMNeT++/OMNEST simulation environment. The INET Framework contains models for several Internet protocols: UDP, TCP, SCTP, IP, IPv6, Ethernet, PPP, IEEE 802.11, MPLS, OSPF, and several other protocols.

This framework was used as our baseline in order to create the algorithms that will achieve the load balance. More specifically, we used the Lan80211 example that can be found in the examples folder of INET framework.

Simulator

In order to measure the effectiveness of each algorithm, we have decided to create a simulator using the OMNeT++ (version 4.0p1). The simulator has the following characteristics:

1. 2 access points

2. 8 stations (clients)

3. 2 servers that connect to the access points

The access points are created according to the IEEE802.11g standard. Each of them transmits to the corresponding stations according to each algorithm.

Scenarios

There are two scenarios that we study. At the first scenario all the stations connect at the same rate. Basically, they are all at the middle of the distance between the two access points having 6 of them closer to the one access point, and the other 2 closer to the other access point. At the second scenario, the stations are at various points within the common radius of the 2 access points and they connect at different rate. The basic scenario is that the servers stream a live video to the clients.

Same Rate

We study each algorithm in this scenario by assuming that all the stations connect at the same bitrate at the corresponding access point. In this way we have the results that are represented by 3 charts. The first chart shows the packets delivered throughout the simulation time, the second one shows the first few seconds until no more stations connect to the access points and the third one shows the end of the live streaming. All these charts can be found in the appendix section of this thesis.

Here we will comment on the results that we pulled from each simulation.

1. *Distance*

   The results about the distance algorithm show that the access point 1 has the most load according to the packets per time that there are throughout the simulation time.

2. *Theta Algorithm*

   The theta algorithm is much better than the distance algorithm in this scenario. The load is distributed among the access points evenly.

3. *LBA*

   The LBA algorithm has the same results as the theta algorithm.

From the charts that can be found in the appendix (figures) we realize that the *distance* is not the most efficient way to balance the load among two access points. The other two algorithms balance the load in a more efficient way. For example. if we decide to sum up the whole throughput and then distribute it to the 2 access points, then we see that both *theta* and *LBA* algorithms distribute the load in the most efficient way.

Various Rates

In this scenario the clients connect to various rates to the access points

Client	Rate (Mbps) - AP1	Rate (Mbps) - AP2
Client1	54	24
Client2	24	11
Client3	11	2
Client4	24	54
Client5	54	24
Client6	24	11
Client7	11	2
Client8	24	54

1. *Distance*

   The results about the distance algorithm show that the access point 1 has the most load according to the packets per time that there are throughout the simulation time.

2. *Theta Algorithm*

   The theta algorithm is much better than the distance algorithm in this scenario. The load is distributed among the access points evenly.

3. *LBA*

   The LBA algorithm shows the same contribution to the balance of load as the *theta algorithm*.

The above results show us exactly what we expected. Both *theta* and *LBA* are better than the distance algorithm, concerning the load distribution.

Improving the Theta Algorithm

We decided that we should improve the theta algorithm. In order to do so, we used a different ϑ than the one used before. The new ϑ is as following:

$$\vartheta = \sum_{r} \frac{B_r}{r}$$

where r is the rate and B_r the throughput of the of stations connecting to this access point at rate r. With this algorithm the clients connect to the access point with the largest ϑ at which they have the smallest contribution of load according to the equation above.

1. *Same rate*

   In the case where all stations connect at the same rate at the access points, the load is distributed evenly among the two access points.

2. *Various rates*

   In the case where all stations connect at the various rates at the access points, the load is distributed evenly among the two access points.

The results show us that there is room for improvement of the *theta algorithm*. This is something that one can study as a future work.

Conclusions

Concluding

Interpreting the results of our simulations

We studied the algorithms described above and then created a simulator according to the scenarios we mentioned. The results show that the LBA an the theta algorithm are much better than the distance algorithm. However, the theta algorithm shows very promising signs for better improvements than the LBA.

The theta algorithm takes under consideration the rate that the stations connect, which is a very important aspect of the wireless network. More rate means faster download (usually). The most important aspect is that the *theta algorithm* lets the stations connect to the access points and then it doesn't handover them to another access point.

The LBA algorithm, on the other hand, takes under consideration the load of every access point and redistributes the stations connected to it, even the ones connected only a few seconds ago. In our opinion, the handover creates more disadvantages than the advantages that it tries to promote. Thus, the *theta algorithm* is considered a much better approach.

Future Work

As future work, we would propose a way to improve the theta algorithm taking under consideration both the access points' load *and* the throughput. It's our firm belief that a metric that has these two characteristics, implemented in a better way than ours will be the best solution to the load balancing problem.

Moreover, we would propose an actual implementation of the above algorithms, including the improved theta algorithm.

Appendix

Ned Files

```
package inet.examples.wireless.lan80211;
import inet.networklayer.autorouting.FlatNetworkConfigurator;
import inet.nodes.wireless.WirelessAPSimplified;
import inet.nodes.wireless.WirelessHostSimplified;
import inet.world.ChannelControl;

network Lan80211
{
   parameters:
       int playgroundSizeX;
       int playgroundSizeY;
   submodules:
       host0:  WirelessHostSimplified {
           @display("p=238,174;r=,,#707070");
       }
       host9:  WirelessHostSimplified {
           @display("p=238,174;r=,,#707070");
       }
       host1:  WirelessHostSimplified {
           @display("p=186,205;r=,,#707070");
       }
       host2:  WirelessHostSimplified {
           @display("p=179,215;r=,,#707070");
       }
       host3:  WirelessHostSimplified {
           @display("p=177,144;r=,,#707070");
       }
       host4:  WirelessHostSimplified {
           @display("p=197,215;r=,,#707070");
       }
       host5:  WirelessHostSimplified {
           @display("p=221,193;r=,,#707070");
       }
       host6:  WirelessHostSimplified {
           @display("p=203,216;r=,,#707070");
```

```
        }
        host7:  WirelessHostSimplified {
            @display("p=240,86;r=,,#707070");
        }
        host8:  WirelessHostSimplified {
            @display("p=303,148;r=,,#707070");
        }
        ap1:  WirelessAPSimplified {
            @display("p=213,174;r=,,#707070");
        }
        ap2:  WirelessAPSimplified {
            @display("p=263,174;r=,,#707070");
        }
        channelcontrol:  ChannelControl {
            playgroundSizeX = playgroundSizeX;
            playgroundSizeY = playgroundSizeY;
            numChannels = 2;
            @display("p=61,46");
        }
        configurator:  FlatNetworkConfigurator {
            networkAddress = "145.236.0.0";
            netmask = "255.255.0.0";
            @display("p=140,50");
        }
}
```

Initialization Files

```
[General]
#debug-on-errors = true
network = Lan80211
tkenv-plugin-path = ../../../etc/plugins

cmdenv-event-banner-details=true
cmdenv-module-messages=true
cmdenv-express-mode=false
cmdenv-output-file=out

output-vector-file = ${resultdir}_7/${configname}-${runnumber}.vec
output-scalar-file = ${resultdir}_7/${configname}-${runnumber}.sca

*.playgroundSizeX = 600
*.playgroundSizeY = 400
**.debug = true
**.coreDebug = false
**.channelNumber = 0
**.mobility.x = -1
**.mobility.y = -1

# channel physical parameters
*.channelcontrol.carrierFrequency = 2.4GHz
*.channelcontrol.pMax = 20.0mW
*.channelcontrol.sat = -110dBm
*.channelcontrol.alpha = 2

# access point
**.ap1.wlan.mac.address = "10:00:00:00:00:00"
**.ap2.wlan.mac.address = "20:00:00:00:00:00"
**.host*.**.mgmt.accessPointAddress = "10:00:00:00:00:00"
**.mgmt.frameCapacity = 100

# mobility
**.host*.mobility.x = -1
```

```
**.host*.mobility.y = -1
**.host*.mobilityType = "NullMobility"
**.host*.mobility.changeInterval = truncnormal(2s, 0.5s)
**.host*.mobility.changeAngleBy = normal(0deg, 30deg)
**.host*.mobility.speed = truncnormal(20mps, 8mps)
**.host*.mobility.updateInterval = 100ms

# udp app
**.numUdpApps = 1
**.host0.udpAppType = "UDPVideoStreamSvr"
**.host0.udpApp[*].videoSize = 10MB
**.host0.udpApp[*].serverPort = 3088
**.host0.udpApp[*].waitInterval = 10ms
**.host0.udpApp[*].packetLen = 1000B

**.numUdpApps = 1
**.host9.udpAppType = "UDPVideoStreamSvr"
**.host9.udpApp[*].videoSize = 10MB
**.host9.udpApp[*].serverPort = 3088
**.host9.udpApp[*].waitInterval = 10ms
**.host9.udpApp[*].packetLen = 1000B

**.host*.udpAppType = "UDPVideoStreamCli"
**.host*.udpApp[*].serverAddress = "host9"
**.host*.udpApp[*].localPort = 9999
**.host*.udpApp[*].serverPort = 3088
**.host1.udpApp[*].startTime = 0
**.host2.udpApp[*].startTime = 2000ms
**.host3.udpApp[*].startTime = 4000ms
**.host4.udpApp[*].startTime = 6000ms
**.host5.udpApp[*].startTime = 8000ms
**.host6.udpApp[*].startTime = 10000ms
**.host7.udpApp[*].startTime = 12000ms
**.host8.udpApp[*].startTime = 14000ms
**.host*.udpApp[*].startTime = 0

# ping app (host[0] pinged by others)
*.host0.pingApp.destAddr = ""
*.host9.pingApp.destAddr = ""
*.host*.pingApp.destAddr = ""
**.pingApp.interval = 10ms
*.host0.pingApp.startTime = 0s
*.host9.pingApp.startTime = 0s
*.host1.pingApp.startTime = 0s
*.host2.pingApp.startTime = 2000ms
*.host3.pingApp.startTime = 4000ms
```

```
*.host4.pingApp.startTime = 6000ms
*.host5.pingApp.startTime = 8000ms
*.host6.pingApp.startTime = 10000ms
*.host7.pingApp.startTime = 12000ms
*.host8.pingApp.startTime = 14000ms

# nic settings
**.mac.address = "auto"
**.mac.maxQueueSize = 14
**.mac.rtsThresholdBytes = 3000B
**.mac.bitrate = 2Mbps
**.wlan.mac.retryLimit = 7
**.wlan.mac.cwMinData = 7
**.wlan.mac.cwMinBroadcast = 31

*.ap1.wlan.radio.bitrate = 54Mbps
*.ap2.wlan.radio.bitrate = 54Mbps
*.host0.wlan.radio.bitrate = 54Mbps
*.host9.wlan.radio.bitrate = 54Mbps
**.radio.bitrate = 2Mbps
**.radio.transmitterPower = 20.0mW
**.radio.carrierFrequency = 2.4GHz
**.radio.thermalNoise = -110dBm
**.radio.sensitivity = -85mW
**.radio.pathLossAlpha = 2
**.radio.snirThreshold = 4dB

# relay unit configuration
**.relayUnitType = "MACRelayUnitNP"
**.relayUnit.addressTableSize = 100
**.relayUnit.agingTime = 120s
**.relayUnit.bufferSize = 1MB
**.relayUnit.highWatermark = 512KB
**.relayUnit.pauseUnits = 300 # pause for 300*512 bit
                              # (19200 byte) time
**.relayUnit.addressTableFile = ""
**.relayUnit.numCPUs = 2
**.relayUnit.processingTime = 2us

[Config Streaming1]
description = "video stream"
#*.numHosts = 3

[Config Streaming2]
description = "n hosts"
# leave numHosts undefined here
```

Charts - Algorithms

Here are all the charts that were created as part of the simulation experiments. Each one has a description that helps us realize to which case it refers to.

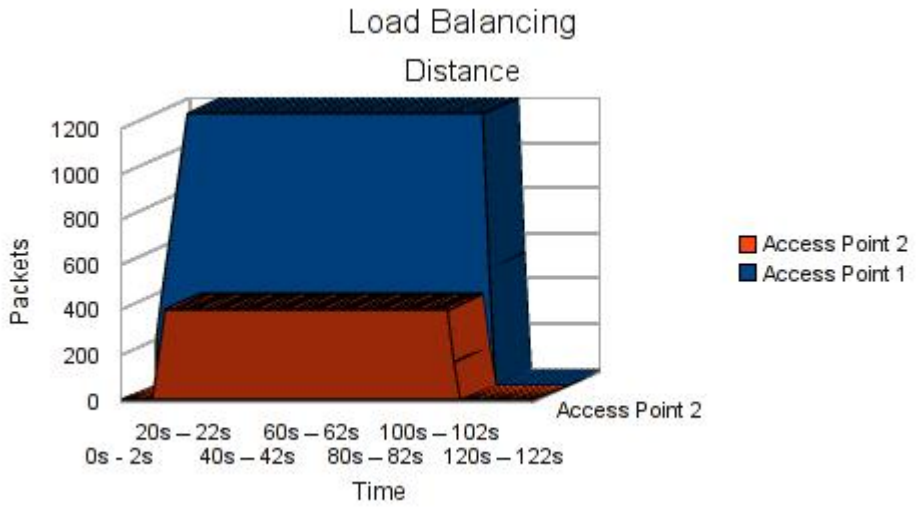

Figure 2: Same Rates - Distance algorithm (full time experiment)

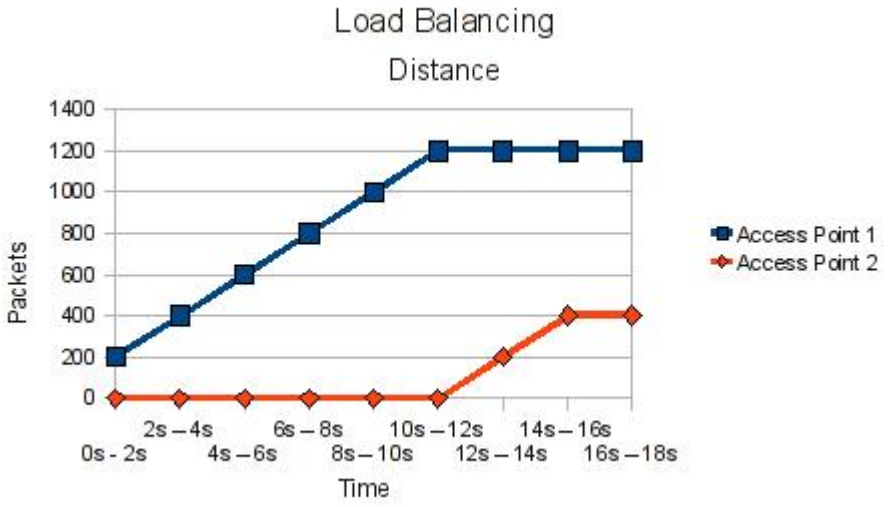

Figure 3: Same Rates - Distance algorithm (first 18 seconds till balance)

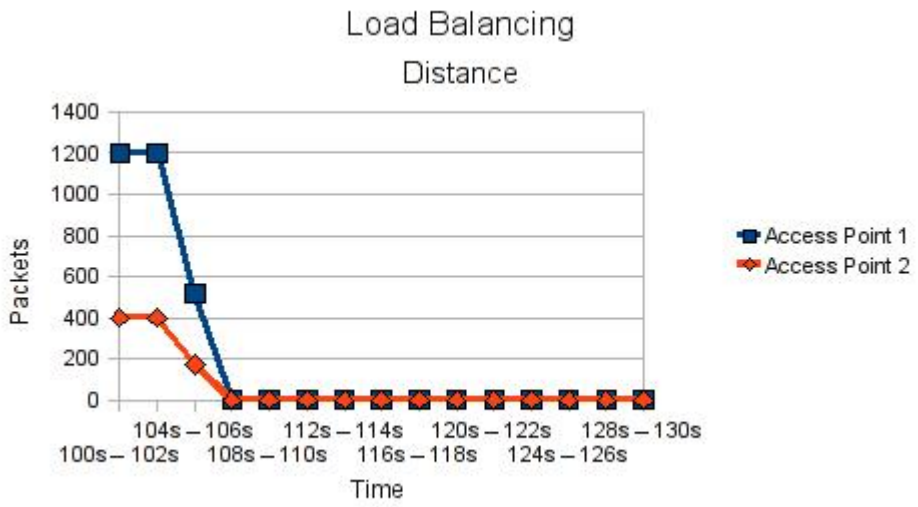

Figure 4: Same Rates - Distance algorithm (last 8 seconds till finish)

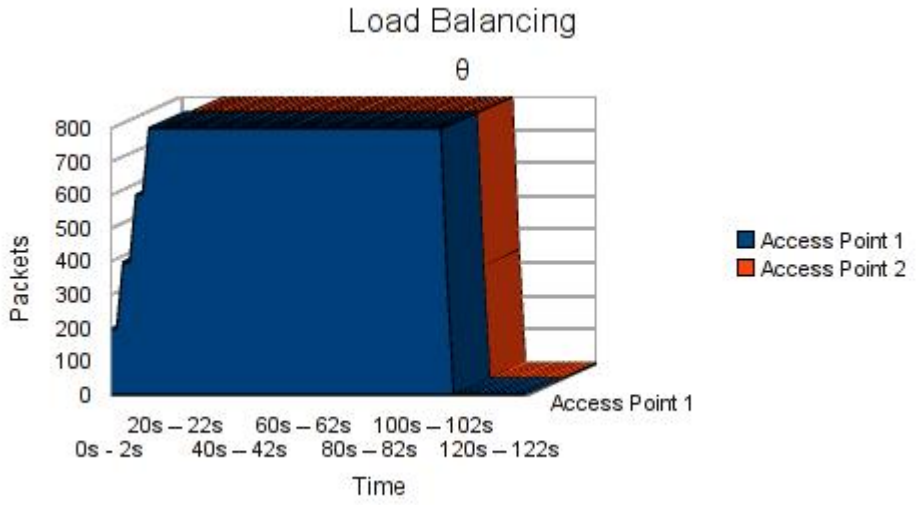

Figure 5: Same Rates - Theta algorithm (full time experiment)

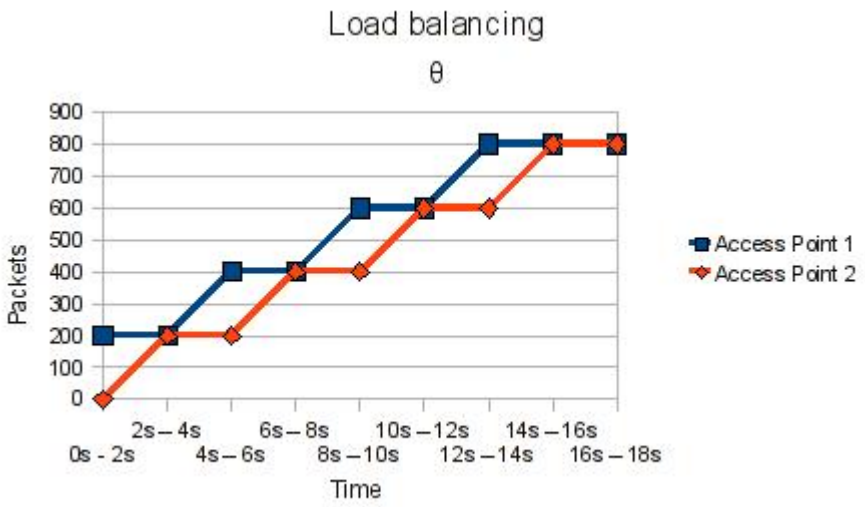

Figure 6: Same Rates - Theta algorithm (first 18 seconds till balance)

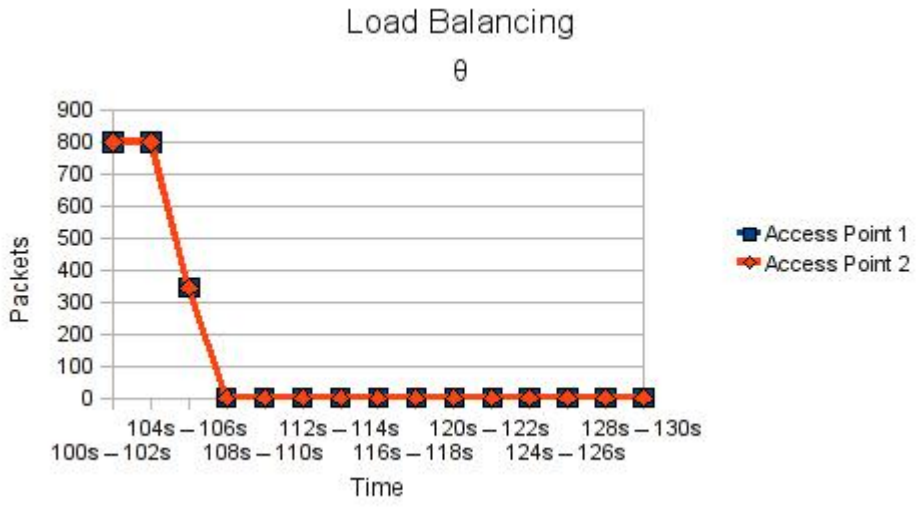

Figure 7: Same Rates - Theta algorithm (last 8 seconds till finish)

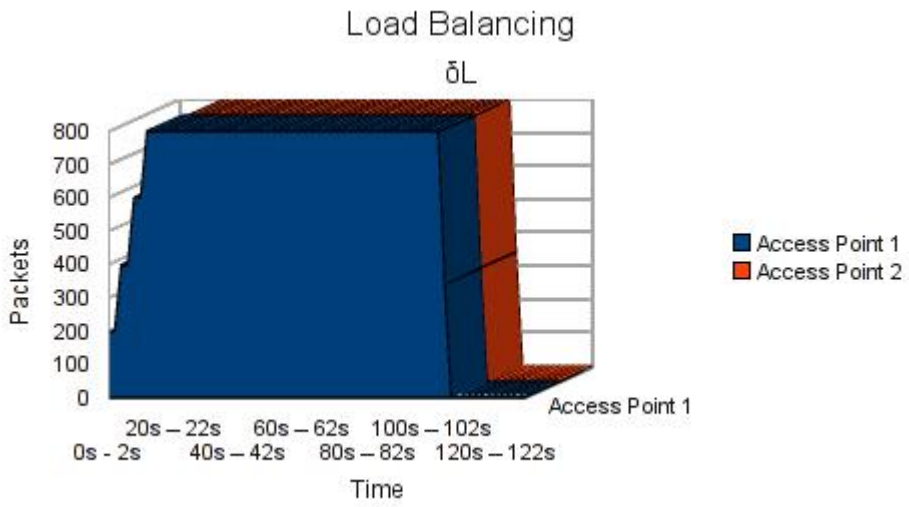

Figure 8: Same Rates - LBA algorithm (full time experiment)

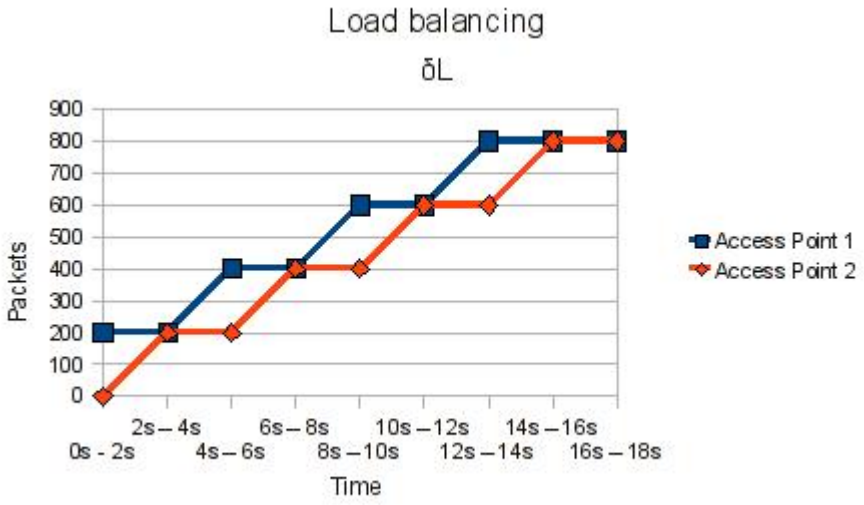

Figure 9: Same Rates - LBA algorithm (first 18 seconds till balance)

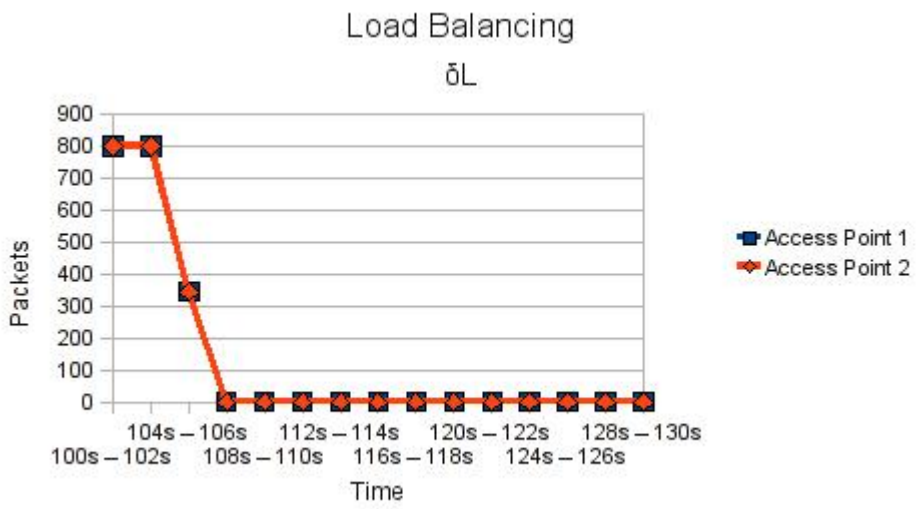

Figure 10: Same Rates - LBA algorithm (last 8 seconds till finish)

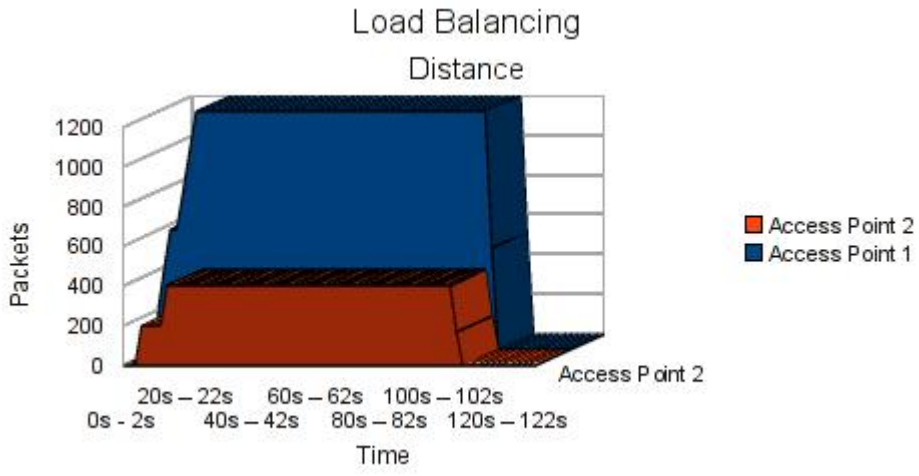

Figure 11: Various Rates - Distance algorithm (full time experiment)

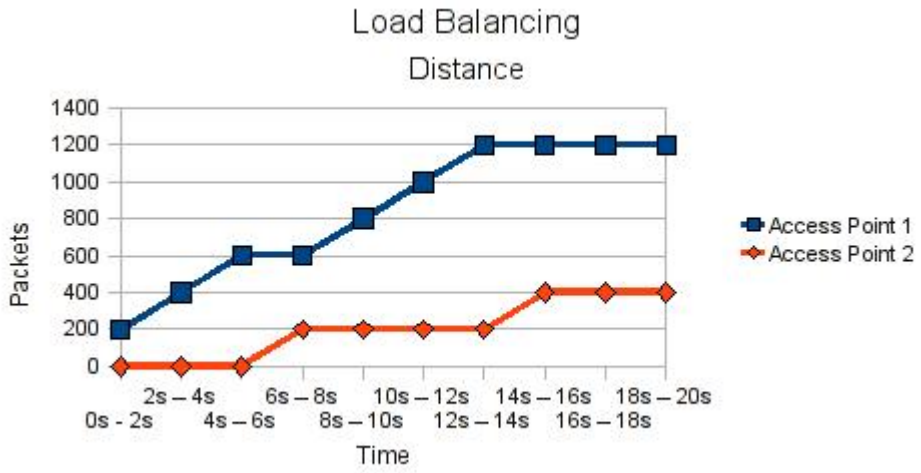

Figure 12: Various Rates - Distance algorithm (first 18 seconds till balance)

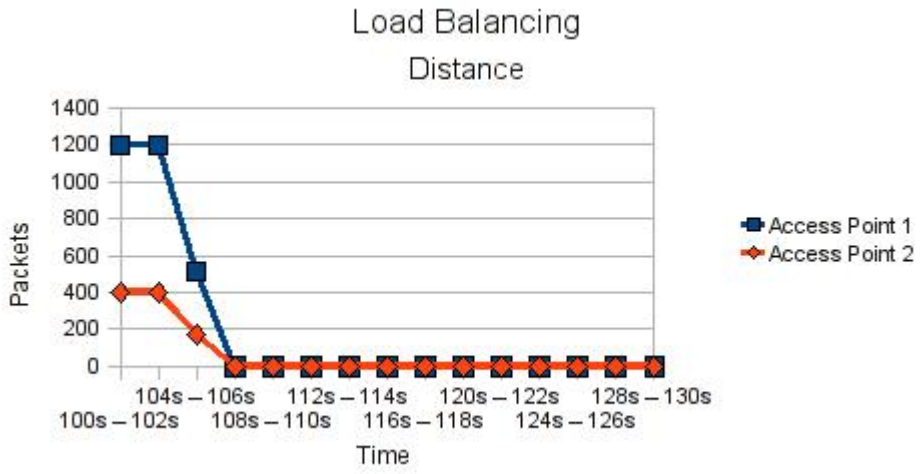

Figure 13: Various Rates - Distance algorithm (last 8 seconds till finish)

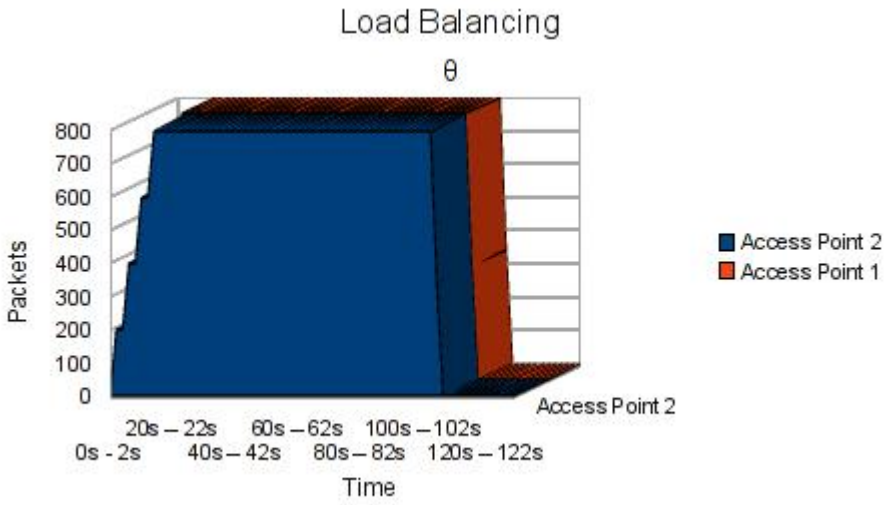

Figure 14: Various Rates - Theta algorithm (full time experiment)

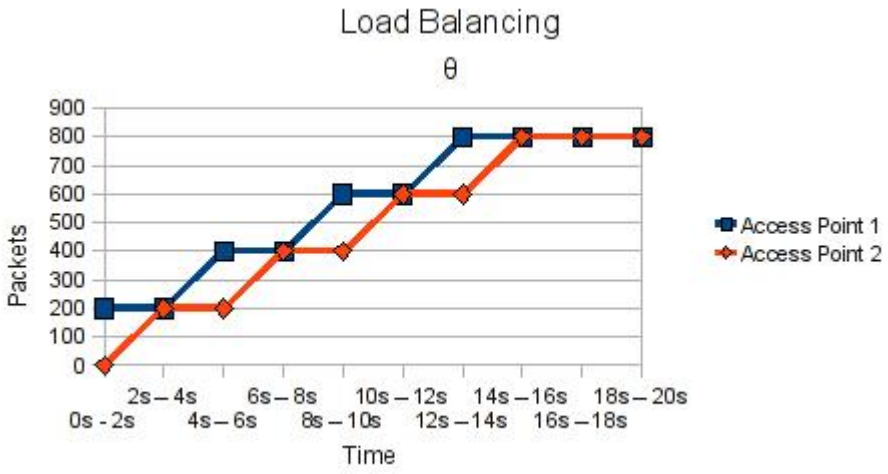

Figure 15: Various Rates - Theta algorithm (first 18 seconds till balance)

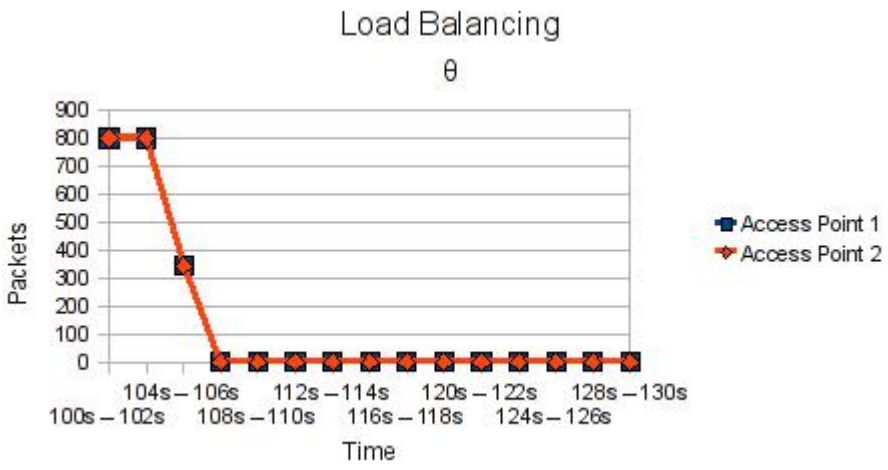

Figure 16: Various Rates - Theta algorithm (last 8 seconds till finish)

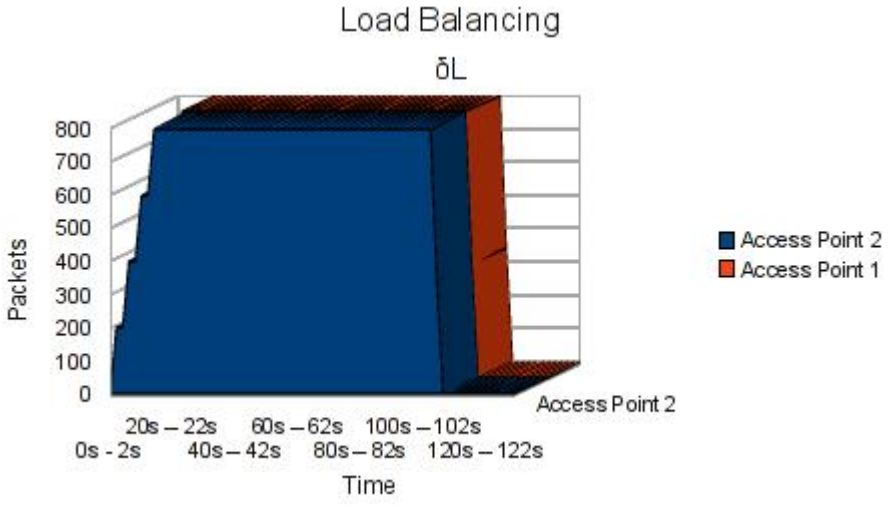

Figure 17: Various Rates - LBA algorithm (full time experiment)

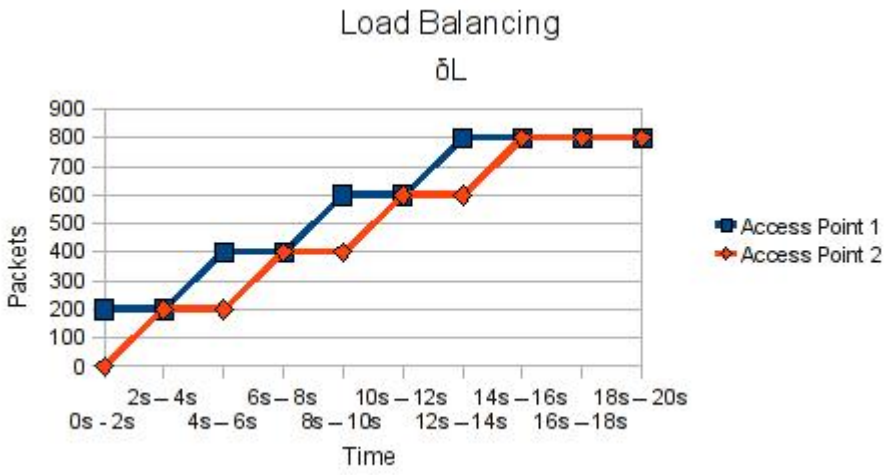

Figure 18: Various Rates - LBA algorithm (first 18 seconds till balance)

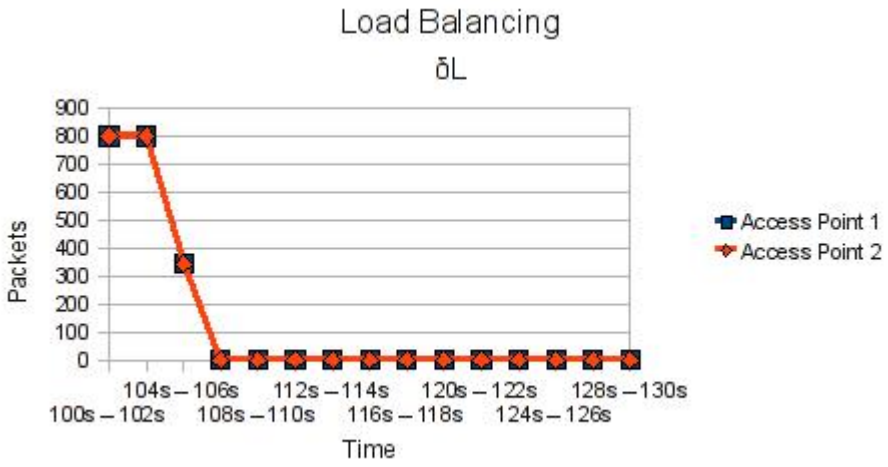

Figure 19: Various Rates - LBA algorithm (last 8 seconds till finish)

Charts - Improved Theta Algorithm

Here we present the charts of the improved *theta algorithm*.

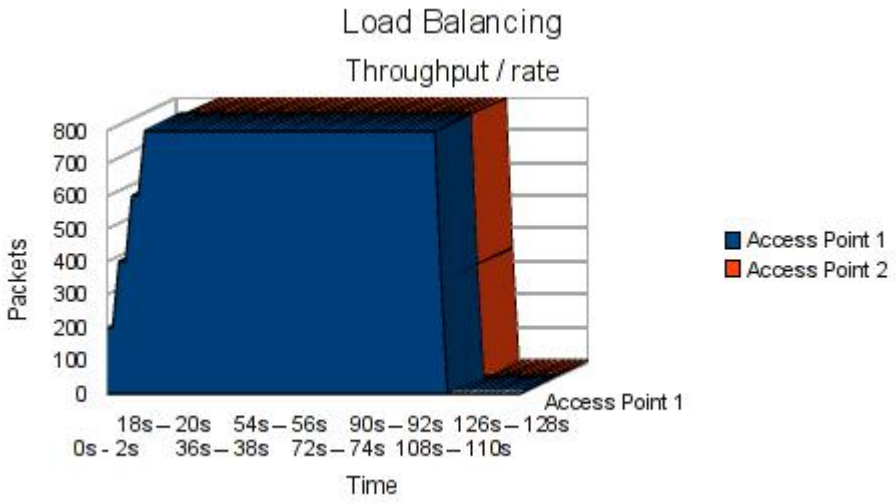

Figure 20: Same Rate - Improved Theta algorithm (full time experiment)

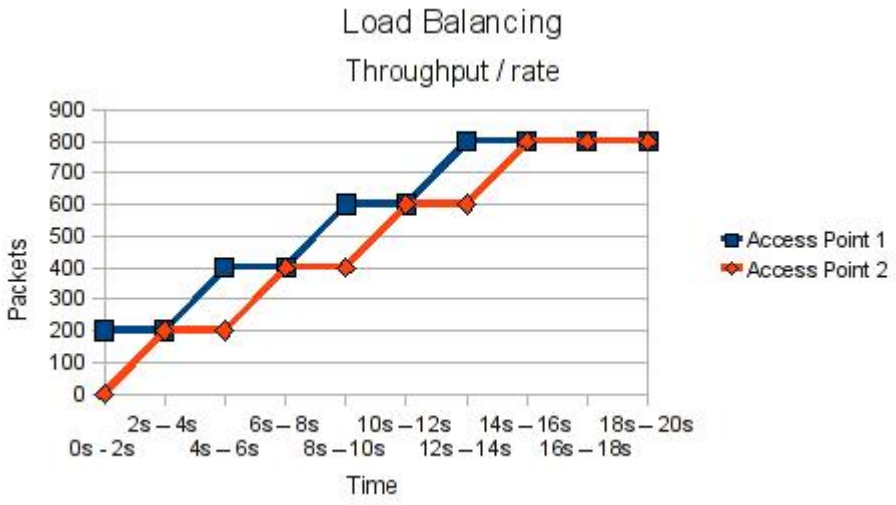

Figure 21: Same Rate - Improved Theta algorithm (first 20 seconds experiment)

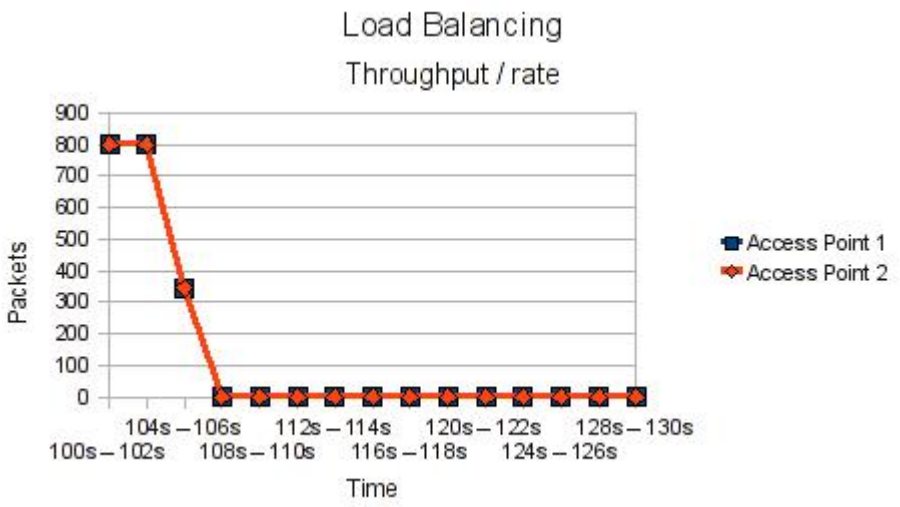

Figure 22: Same Rate - Improved Theta algorithm (last 8 seconds experiment)

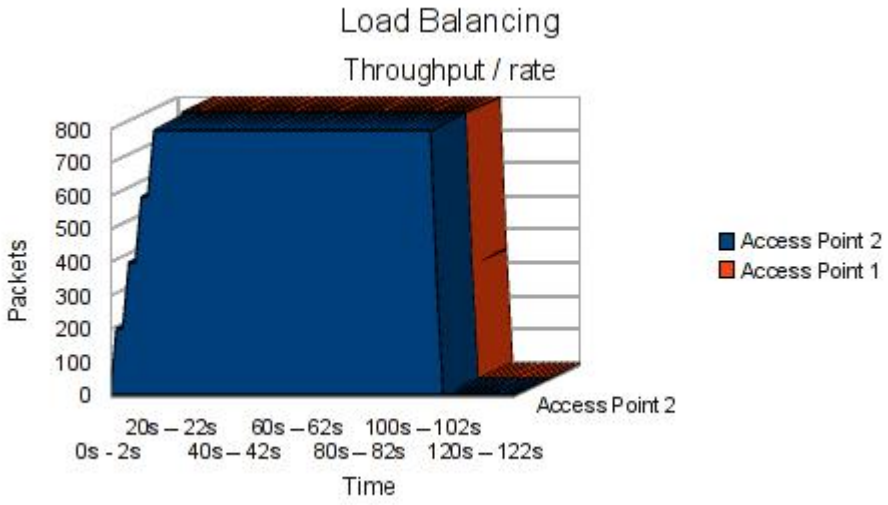

Figure 23: Various Rates - Improved Theta algorithm (full time experiment)

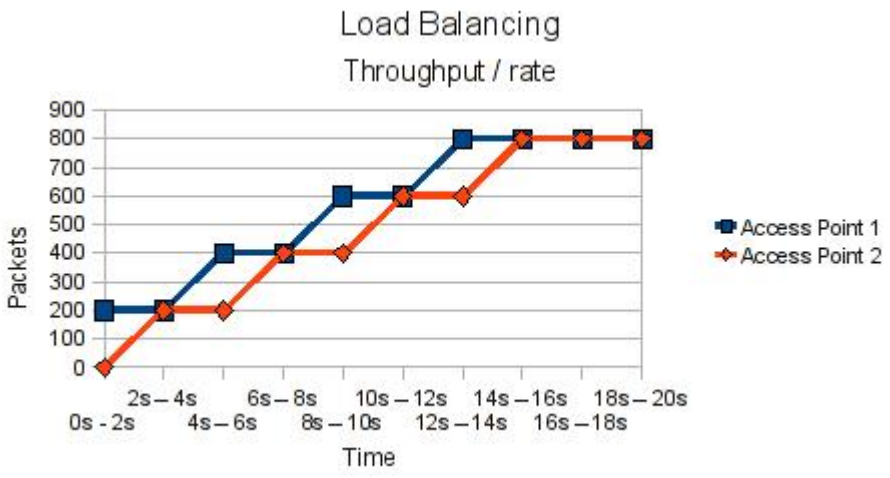

Figure 24: Various Rates - Improved Theta algorithm (first 20 seconds experiment)

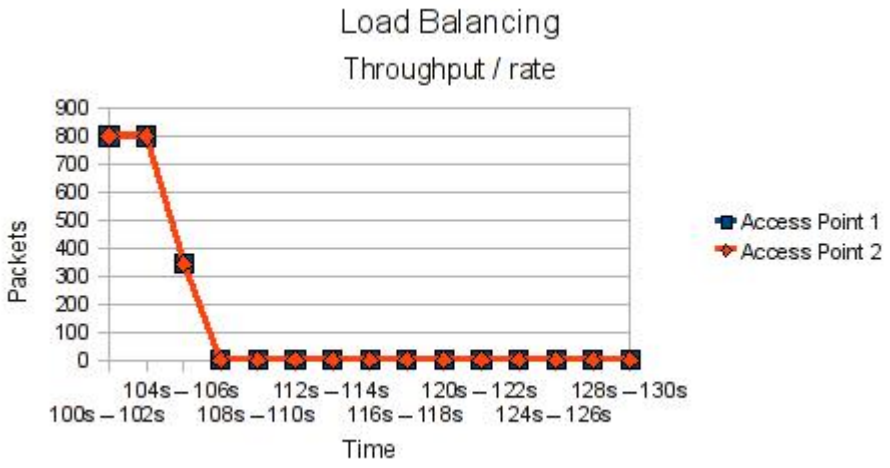

Figure 25: Various Rates - Improved Theta algorithm (last 8 seconds experiment)

Links

- OMNeT++ official site: `http://www.omnetpp.org/`
- OMNeT++ 4.1 Tic-Toc tutorial: `http://www.omnetpp.org/doc/omnetpp41/tictoc-tutorial/`
- A screencast demo of using the OMNeT++ IDE: `http://www.omnest.com/webdemo/ide/demo.html`
- INET Framework official site: `http://inet.omnetpp.org/`
- INET Framework documentation: `http://inet.omnetpp.org/doc/INET/neddoc/index.html`

NOTE: *The above links are valid until this time that these lines are written. Bear in mind that depending on the release of new versions of OMNeT++ these links may change.*

Bibliography

[1] Velayos, H. Aleo, V. Karlsson, G. Load balancing in overlapping wireless LAN cells *Communications, 2004 IEEE International Conference on.* Pages: 3833-3836, Vol.7, 20-24 June 2004.

[2] Vasilios A. Siris and Theodoros Dionisiou Load Balancing among Access Points in Multi-Rate Wireless LANs

[3] Murad Abusubaih and Adam WoliszAn Optimal Station Association Policy for Multi-Rate IEEE802.11 Wireless LANs *Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems.* Pages: 117-123, 2007.

[4] Velayos, H. Mas, I. Karlsson, G. Overload Protection for IEEE 802.11 Cells *Quality of Service, 2006. IWQoS 2006. 14th IEEE International Workshop on.* Pages: 149-158, 19-21 June 2006

[5] Michael Tüxen, Irene Rüngeler, Erwin P. RathgebInterface connecting the INET simulation framework with the real world *SIMUTools 2008. March 3-7, 2008*

[6] Juan-Carlos Maureira, Olivier Dalle, Diego Dujovne.Generation of Realistic 802.11 Interferences in the Omnet++ INET Framework Based on Real Traffic Measurements *Quality of Service, 2006. IWQoS 2006. 14th IEEE International Workshop on.* Pages: 149-158, 19-21 June 2006

[7] Canali Claudia, Colajanni Michele and Lancellotti Riccardo (2010). Resource Management Strategies for the Mobile Web *Mob. Netw. Appl.*, 15, 2: 237–252.

[8] Stamos Konstantinos, Pallis George, Vakali Athena, Katsaros Dimitrios, Sidiropoulos Antonis and Manolopoulos Yannis (2010). *CDNsim: A simulation tool for content distribution networks ACM Trans. Model. Comput. Simul.*, 20, 2: 1–40.

[9] Payton Jamie, Julien Christine, Roman Gruia-Catalin and Rajamani Vasanth (2010). Semantic self-assessment of query results in dynamic environments *ACM Trans. Softw. Eng. Methodol.*, 19, 4: 1–33.

[10] Yasar Ansar-Ul-Haque, Mahmud Nasim, Preuveneers Davy, Luyten Kris, Coninx Karin and Berbers Yolande (2010). Where people and cars meet: social interactions to improve information sharing in large scale vehicular networks *SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing:* 1188–1194.

[11] Brogle Marc, Barthlomé Sebastian and Braun Torsten (2010). Quality of service for multicasting using NICE *SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing:* 670–677.

[12] Liu Zhenhua and Xu Wenyuan (2010). Zeroing-in on network metric minima for sink location determination *WiSec '10: Proceedings of the third ACM conference on Wireless network security:* 99–104.

[13] Miao Huawei, Ooi Chia Ching, Wu Xiaowen and Schindelhauer Christian (2010). Coverage-hole trap model in target tracking using distributed relay-robot network *SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing:* 1299–1304.

[14] Yadav Kuldeep and Srinivasan Avinash (2010). iTrust: an integrated trust framework for wireless sensor networks *SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing:* 1466–1471.

[15] Hautakorpi Jani and Mäenpää Jouni (2010). Load balancing for structured P2P networks using the advanced finger selection algorithm (AFSA) *SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing:* 655–662.

[16] Oikonomou Konstantinos, Kogias Dimitrios and Stavrakakis Ioannis (2010) A study of information dissemination under multiple random walkers and replication mechanisms *MobiOpp '10: Proceedings of the Second International Workshop on Mobile Opportunistic Networking:* 118–125.

[17] Schell Frank, Schaf Andreas, Dinger Jochen and Hartenstein Hannes (2010). Assessing identity and access management systems based on domain-specific performance evaluation *WOSP/SIPEW '10: Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering:* 253–254.

[18] Lacage Mathieu, Ferrari Martin, Hansen Mads, Turletti Thierry and Dabbous Walid (2010). NEPI: using independent simulators, emulators, and testbeds for easy experimentation *SIGOPS Oper. Syst. Rev., 43, 4:* 60–65.

[19] Di Sorte D., Femminella M. and Reali G. (2009) QoS-enabled multicast for delivering live events in a Digital Cinema scenario *J. Netw. Comput. Appl., 32, 1:* 314–344.

[20] Nou Ramon, Kounev Samuel, Julií Ferran and Torres Jordi (2009) Autonomic QoS control in enterprise Grid environments using online simulation *J. Syst. Softw., 82, 3:* 486–502.

[21] Arns Markus, Buchholz Peter and Müller Dennis (2009) OPEDo: a tool for the optimization of performance and dependability models *SIGMETRICS Perform. Eval. Rev., 36, 4:* 22–27.

[22] Pustina Lukas, Schwarzer Simon, Gerharz Michael, Martini Peter and Deichmann Volker (2009) A practical approach for performance-driven UML modelling of handheld devices - A case study *J. Syst. Softw., 82, 1:* 75–88.

[23] Horváth András, Horváth Gábor and Telek Miklós (2009) A traffic based decomposition of two-class queueing networks with priority service *Comput. Netw., 53, 8:* 1235–1248.

[24] Macedo Mário, Grilo António and Nunes Mário (2009) Distributed Latency-Energy Minimization and interference avoidance in TDMA Wireless Sensor Networks *Comput. Netw., 53, 5:* 569–582.

[25] Mukherjee Sankar and Singh Jyoti Prakash (2009) Time slot assignment for interference reduction in cluster based sensor network *ICAC3 '09: Proceedings of the International Conference on Advances in Computing, Communication and Control:* 679–684.

[26] Pujol-Ahulló Jordi, García-López Pedro, Sànchez-Artigas Marc and Arrufat-Arias Marcel (2009) An extensible simulation tool for overlay networks and services *SAC '09: Proceedings of the 2009 ACM symposium on Applied Computing:* 2072–2076.

[27] Xu Ming-wei, Wu Qian, Xie Guo-liang and Zhao You-jian (2009) The impact of mobility models on mobile IP multicast research *Int. J. Ad Hoc Ubiquitous Comput., 4, 3/4:* 191–200.

[28] Wang Hui, Agoulmine Nazim, Ma Maode, Li Yajun and Wang Xiaomin (2009) Network Lifetime Optimization by KKT Optimality Conditions in Wireless Sensor Networks *Wirel. Pers. Commun., 49, 2:* 179–196.

[29] Benmammar Badr, Jrad Zeina and Krief Francine (2009) QoS management in mobile IP networks using a terminal assistant *Int. J. Netw. Manag., 19, 1:* 1–24.

[30] Singh, Jyoti Prakash and Dutta, Paramartha (2009) Temporal behavior analysis of mobile ad hoc network with different mobility patterns *ICAC3 '09: Proceedings of the International Conference on Advances in Computing, Communication and Control:* 696–702.

[31] Milani, Simone, Calvagno, Giancarlo, Bernardini, Riccardo and Zontone, Pamela (2009) Cross-Layer joint optimisation of FEC channel codes and Multiple Description Coding for video delivery over IEEE 802.11e links *Int. J. Internet Protoc. Technol., 4, 1:* 32–43.

[32] Cuevas, Ruben, Cabellos-Aparicio, Albert, Cuevas, Angel, Domingo-Pascual, Jordi and Azcorra, Arturo (2009) fP2P-HN: A P2P-based route optimization architecture for mobile IP-based community networks *Comput. Netw., 53, 4:* 528–540.

[33] Dietrich, Isabel and Dressler, Falko (2009) On the lifetime of wireless sensor networks *ACM Trans. Sen. Netw., 5, 1:* 1–39.

[34] Yang, Shuigen, Luo, Hongbin, Qin, Yajuan and Zhang, Hongke (2009) Design and Evaluation of DNS as Location Manager for HIP *Wirel. Pers. Commun., 48, 4:* 605–619.

[35] Law, Yee Wei, Palaniswami, Marimuthu, Hoesel, Lodewijk Van, Doumen, Jeroen, Hartel, Pieter and Havinga, Paul (2009) Energy-efficient link-layer jamming attacks against wireless sensor network MAC protocols *ACM Trans. Sen. Netw., 5, 1:* 1–38.

[36] Hoes, Rob, Basten, Twan, Tham, Chen-Khong, Geilen, Marc and Corporaal, Henk (2009) Quality-of-service trade-off analysis for wireless sensor networks *Perform. Eval., 66, 3-5:* 191–208.

[37] Weingärtner, Elias, vom Lehn, Hendrik and Wehrle, Klaus (2009) A performance comparison of recent network simulators *ICC 2009: IEEE International Conference on Communications.*

[38] Durán, Ramon J., de Miguel, Ignacio, Merayo, Noemí, Fernández, Patricia, Lorenzo, Rubén M. and Abril, Evaristo J. (2009) Joint optimization of delay and congestion in wavelength-routed optical networks using genetic algorithms *Photonic Network Communications.*

[39] Binh, Le Nguyen, Binh, Le Huu and Tu, Vo Thanh (2009) Routing and Wavelength Assignment and Survivability of Optical Channels in Ultra-high Speed IP over DWDM Networks Under Constraints of Residual Dispersion and Nonlinear Effects *IJCSNS International Journal of Computer Science and Network Security, 9, 2.*

[40] Braun, Torsten, Staub, Thomas and Gantenbein, Reto (2009) VirtualMesh: An Emulation Framework for Wireless Mesh Networks in OMNeT++ *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++* (hosted by SIMUTools 2009).

[41] Minkenberg, Cyriel and Herrera, German Rodriguez (2009) Trace-driven Co-simulation of High-Performance Computing Systems using OMNeT++ *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[42] Warneke, Daniel and Rerrer-Brusch, Ulf (2009) Exchangeable, Application-Independent Load Balancing for P2P Simulation Frameworks *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[43] Kozlovszky, Miklós, Balaskó, Ákos and Varga, András (2009) Enabling OMNeT++-based simulations on Grid Systems *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[44] Gamer, Thomas and Mayer, Christoph P. (2009) Large-scale Evaluation of Distributed Attack Detection *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[45] Seggelmann, Robin, Rüngeler, Irene, Tüxen, Michael and Rathgeb, Erwin P. (2009) Parallelizing OMNeT++ simulations using Xgrid *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[46] Jonsson, Kristjan (2009) HttpTools: A Toolkit for Simulation of Web Hosts in OMNeT++ *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[47] Lewandowski, Andreas, Köster, Volker and Wietfeld, Christian (2009) A new dynamic co-channel interference model for simulation of heterogenous wireless networks *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[48] Ariza, Alfonso, Casilari, Eduardo and Cabrera, Alicia Triviño (2009) An architecture for the implementation of Mesh Networks in OMNeT++ *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[49] Höckner, Sören, Lagemann, Andreas and Nolte, Jörg (2009) Integration of Event-Driven Embedded Operating Systems Into OMNet++ – A Case Study with Reflex *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[50] Maureira, Juan-Carlos, Dujovne, Diego and Dalle, Olivier (2009) Generation of Realistic 802.11 Interferences in the Omnet++ INET Framework Based on Real Traffic Measurements *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[51] Dreibholz, Thomas, Zhou, Xing and Rathgeb, Erwin (2009) SimProcTC – The Design and Realization of a Powerful Tool-Chain for OMNeT++ Simulations *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[52] Hurtado-López, J., Casilari, Eduardo and Ariza, Alfonso (2009) Enabling IEEE 802.15.4 Cluster-Tree Topologies in OMNeT++ *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[53] Wessel, Karl, Swigulski, Michael, Köpke, Andreas and Willkomm, Daniel (2009) MiXiM The Physical Layer An Architecture Overview *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[54] Rousselot, Jérome and Decotignie, Jean-Dominique (2009) A High-Precision Ultra Wideband Impulse Radio Physical Layer Model for Network Simulation *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[55] Lichte, Hermann and Weide, Jannis (2009) Modeling Obstacles in INET/Mobility Framework: Motivation, Integration, and Performance *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[56] Bredel, Michael and Bergner, Martin (2009) On The Accuracy of IEEE 802.11g Wireless LAN Simulations Using OMNet++ *OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*