ET2438: “TCP/IP Internetworking”

laboration exercises : OPNET Simulation and Tuning TCP

Blekinge Institute of Technology
School of Computing
Department of Telecommunication Systems

Name: __________ Acronym: __________

Examiner: __________ Date: __________ Pass: __________

Contents

1 Introduction

1.1 Goals and ambitions

The practical part of the course ET2440 TCP/IP Internetteknik consists of three parts. This documents elaborates one of them, you will learn how to use OPNET. The other parts will be presented to you in another document.

Optimized Network Engineering Tools (OPNET) is a powerful network simulator. Its main purposes are to optimize cost, performance and availability of network simulations. The goal of this set of laboration exercises is to learn the basics of how to use the OPNET Modeler interface, as well as some basic modeling theory. The following tasks are considered:

- Build and analyze models.
- Configure the object palette with the needed models.
- Set up application and profile configurations.
- Model a Local Area Network (LAN) as a single node.
- Specify background utilization that changes over a time on a link.
- Simulate multiple scenarios simultaneously.
- Apply filters to result graphs and analyze the results.

After having completed the following exercises you will be able to:

- Understand the working of OPNET.
- Set-up your own simulation scenarios in OPNET.
- Extract experimental results from OPNET simulations.

Your OPNET skills will be tested with a challenge lab at the end of the course.

1.2 Prerequisites

You will need rudimentary skills in navigating the UNIX system, launching programs and listing directories. Being able to use either emacs or vi is a definite advantage, as is previous contact with other scripting Languages, such as *Bash*.

1.3 Formatting

The text uses different typefaces to distinguish between user input, the output of commands and menu options. The regular Courier typeface signifies user input, whereas italic Courier typeface denotes the output from a command. The former is also used for the source code listings. Furthermore, regular bold typeface is used for denoting a menu option or clickable Graphical User Interface objects and regular italic typeface denotes a value that must be filled out.

Example

Issuing the command:

`ls -l valid.xml`

will yield the following results:

`-rw-r--r-- 1 der dave 223 Mar 14 14:13 valid.xml`

Enter `your name` in the *name field*, and click on the **Exit** button to leave the workspace.

1.4 Deliverables

No deliverables are expected from the OPNET laborations nor are dedicated laboratories organized. You are expected to go through the OPNET labs in this document by yourselves. This can can be done either in one of our computer labs running OPNET (see the BTH website for room numbers) or on your own workstation. Hereby we offer you the possibility to play with OPNET at your disposal. At the end of the course a challenge lab is organized where you will have to demonstrate the gained OPNET knowledge and skills.

1.5 Examination

A challenge lab will be organized at the end of the course for this part of the laboration. Your gained knowledge in OPNET will be tested by simulating a proposed scenario. A sample scenario will be provided to make you familiar with the level of the examination.

The examination is conducted individually. Students are not allowed to communicate with others during the exam. There is only a limited amount of time allotted for the examination, and the lab examiner will not wait for those who must finish up in the last moment. Make sure your simulation is running without any problems. There will be no lab support, i.e. the lab examiner will not help solving potential problems with your simulation.

For the dates and times of the challenge lab refer to the course page on It's Learning. The data will be provided during the course.

1.6 Code of Ethics

Lab assistance by the lab supervisor will be provided using the course forum on *It's learning*.

The challenge lab, the OPNET exam, is to be performed *individually*.

Any other form of cooperation, copying and/or sharing of solutions will be considered a breach of the code of ethics and will warrant loss of credit and/or additional exercises without prior notice.

You are expected to be present at the challenge lab at the date assigned to you. You are only exempted from your duty in case of illness (with a doctor's note) or extraordinary circumstances. If one of these situations present themselves, then you should discuss the matter with the lab assistant as soon as possible.

1.7 Acknowledgement

This lab manual is established with laboration documentation writing by Alex Popescu and Tommy Svensson. The Layout of the document is designed by David Erman.

1.8 Outline of this Document

This document is outlined as follows. *Part 1* contains three tutorials to familiarize you with OPNET this introduction will prepare you for the challenge lab at the end of the course. Do not feel bound to this tutorials. Experimentation with the programs parameters are at your disposal.

The first part is followed by the practical part, *Part 2*. Here the details about the practical part are expounded. This will guide you through your assignment.

2 Part 1: The OPNET simulation tool

2.1 Obtaining OPNET

Optimized Network Engineering Tools (OPNET) is a closed source simulation program. Blekinge Institute of Technology (BTH) has a site license to provide students with OPNET access. OPNET is installed and available on all computers in computer room 2511. You can access this computer room with your access card anytime when there are no teaching activities going on. An academic version of OPNET is also available on `http://www.opnet.com/university_program/itguru_academic_edition/`. The academic version allows you to run OPNET on your own machine with a students license. Before downloading the academic version you must register yourself.

After installation OPNET, as any other program, is available from the *Programs* menu of your system. (**Start → All Programs → Simulators → OPNET Modeler 14.5 → OPNET Modeler 14.5**)

2.2 A large scale OPNET scenario

2.2.1 Goal of this lab

The goal of the laboration is to model a Wide Area Network (WAN) composed of several LANs. The main task is to model the BTH's WAN. BTH stretches over three locations in Blekinge, namely: Karlskrona, Ronneby and Karlshamn. Because there is no interest in modeling the details of each LAN you will use available LAN models to model the individual LANs as single nodes. The first step in setting up the WAN is to specify the overall context for the network with the Startup Wizard.

2.2.2 Methodology

The task that will be emphasized through this exercise is to determine how the bAcknowledgment (ACK)ground traffic is affecting File Transfer Protocol (FTP) traffic on the BTH network. To do this the FTP performance of the network will be modeled, first without background traffic and then with background traffic. The student should follow the following steps in the order described in this laboration manual in order to complete the laboration exercise:

1 Begin by starting up OPNET Modeler and create a new project. Select **File → New**.

2 Name the new project `acronym_LAN_Mod` and the scenario `no_back_util`. Here, acronym must be substituted with the student's own **acronym**!

3 Create an empty scenario for the initial topology.

4 Specify a map to use as a background for your network. Click **Choose From Maps** for Network Scale. Choose **Europe** or **World** from the list.

5 Select **Lan_Mod_Model_List** to be included in your network topology by clicking on the Include cell and changing the value from `No` to `Yes`.

6 Review your settings and finish the startup wizard. The workspace should now contain the specified map and the object palette.

7 Zoom in Sweden from the chosen map (until you are satisfied).

To work with Modeler's full set of node and link models would be overwhelming, so the object palette can be configured to show only a specific subset, or model list. Furthermore, you can use the standard model list, adapt it for your own needs, or make your own list. For this laboration **Lan_Mod_Model_List** was created. Now you will adapt that model list by adding the LAN node model to it. Open the configure palette dialog box by clicking the **Configure Palette** button in the object palette. The dialog box lets you change the object palette according to your needs.

8 Expand the **LAN** Node Model by clicking in the configure palette dialog box. A list of entries drops down.

9 Find **10BaseT_LAN** in the list and include it in your object palette with a right mouse click, **Add to Default Palette** . (See Figure 1)

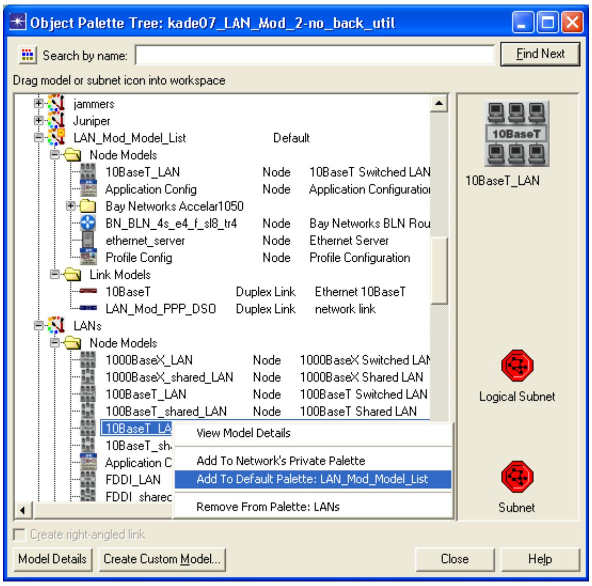

Figure 1: The object Palette

You will now configure the **Application Configuration Object** and the **Profile Configuration Object**. Before you begin constructing the network you must predefine the profiles and applications that will be used by the LAN. To configure the application configuration object, open the object palette in the case it is not already open and

10 Drag an **Application Config** object to the project workspace.

11 Right click on the object and select **Edit Attributes** from the pop-up menu.

By clicking on the question mark next to an attribute you will see a brief description of that attribute.

12 Set the *name* attribute to `Application Configuration`. (See Figure 2)

13 Change the **Application Definitions** attribute to `Default` by clicking in the attribute's **Value** column and selecting `Default` from the pop-up list.

By selecting `Default`, OPNET configures the application definition object to have eight standard applications, namely: Database Access, E-mail, File Transfer, File Print, Telnet Session, Video Conferencing, Voice over

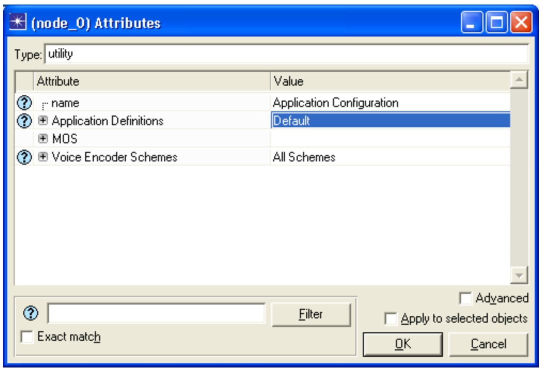

Figure 2: The Application Configuration

IP (VoIP) Call and Web Browsing. Close the Attributes dialog box by pressing the **OK** button. Now you must configure the **Profile Configuration Object**.

14 Drag a **Profile Config** object from the object palette to the workspace.

15 Right-click on the object and select **Edit Attributes**.

16 Set the name attribute to **Profile Configuration**.

17 Change now the **Profile Configuration** attribute by clicking in its **Value** column and selecting `Edit` from the drop down menu.

The profile configuration table box (see Figure 3) appears. You must define a new profile and add it to the table.

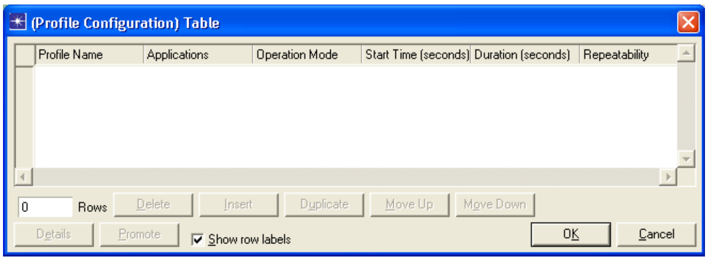

Figure 3: The Profile configuration box

18 Change the number of *Rows* to `1`.

19 Name the new profile `LAN Client`.

20 Click in the profile's **Start Time (seconds)** cell to open the **Start Time Specification** dialog box. (See Figure 4)

21 Select a `constant distribution` with a mean outcome of *100*.

You will be modeling FTP performance, thus that application should be included in the profile.

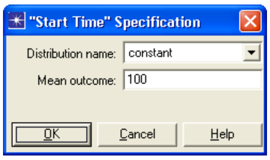

Figure 4: Start Time Specification

22 Expand in the **Profile Config** window the *Profile Configuration*, then the **LAN Client** and then the *Applications* list item. Choose `Edit` from the pop-up menu in the *Value* culomn.

23 Change the number of *Rows* to `1`.

24 Select **File Transfer (Heavy)** by clicking in the appropriate cell and selecting this type of application from the pop-up menu.

By selecting Default as the value for the application definition attribute in this object, you enable a list of applications. The list includes 16 entries, a heavy and a light version for each of the eight standard applications we mentioned earlier.

25 Set the start time offset to **uniform (0, 300)**.

The completed dialog box should look something like Figure 5. Verify it and then close the applications table dialog box as well as all the other dialog boxes you may still have open.

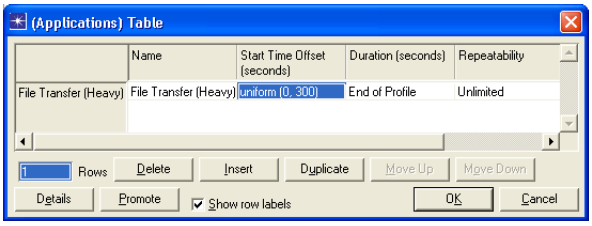

Figure 5: The Application Configuration

You are now ready to begin the construction of the BTH WAN. In this laboration scenario the network will contain three identical subnets placed over Karlskrona, Ronneby and Karlshamn. You can create the first subnet in Karlskrona, with its nodes inside it, and then copy the whole subnet to Karlshamn and Ronneby and then modify it further when needed. Subnets are useful when organizing your network model. Subnets can be nested within subnets to an unlimited degree. Now, open the object palette if not already open.

26 Place a **subnet** node over Karlskrona (in the workspace) and right-click to turn off duplication.

27 Set the name of the subnet to **Karlskrona**.

The extent of the subnet needs to be modified. The subnet extent is the geographic area covered by the subnet, which may be much larger than the actual area you wish to model.

28 Right-click on the Karlskrona subnet and select **Edit Attributes (Advanced)**.

29 Change the **x span** and **y span** attributes to `0.25`.

The unit of measure of these attributes is determined by the unit of measure of the top-level area, i.e., degrees in this case. In order to see what's inside subnets just double-click on that subnet icon and the Modeler will change the view. By default a subnet's grid properties is based on its parent subnet. You can change them to fit your network.

30 Open the Karlskrona subnet by double clicking on it and change its properties (**View** → **Background** → **Set Properties**) such that the units are set to `Meters` and resolution is `10 pixels/m`. Uncheck the **Visible** checkbox for `Satellite Orbits` and verify that **Drawing** is set to `Dashed` and **Division** is set to `10`.

The dialog box should resemble Figure 6.

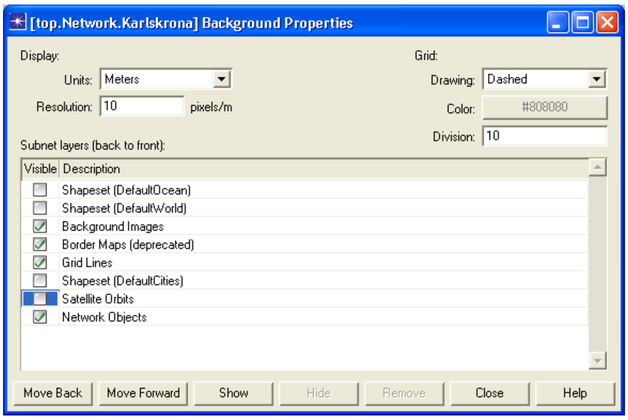

Figure 6: Subnet visual properties

The BTH network does not require modeling the precise nature of each node in each subnet, so you can represent the subnets with a LAN model.

31 Place a **10BaseT_LAN** in the workspace.

32 Right-click on the **10BaseT_LAN** and choose **Edit Attributes**.

You can change the attributes so that it represents a network with a certain number of workstations and a particular traffic profile.

33 Set the name to `Office_LAN`.

34 Choose **Edit** for the *Application: Supported Profiles* attribute in the *Applications* list item.

35 Change the number of **Rows** to `1`.

36 Change the profile name to **LAN Client**.

This LAN will now use the **LAN Client** profile you just created. This profile includes the file transfer application. The LAN will send traffic that models heavy FTP use. Verify that the *Number of Clients* attribute is set to 10 and close the attributes box. You have now modeled a ten workstations LAN inside the Karlskrona subnet. Furthermore, because this LAN model is composed of workstations and links only, it must be connected to a router. This router can then be connected to other routers in the network.

37 Drag a **BN_BLN_4s_e4_f_sl8_tr4** node from the object palette to the workstation near the **Office_LAN** node.

38 Name the new node router and connect it to the **Office_LAN** node with a **10BaseT** link.

The Karlskrona subnet is now fully configured. Because the subnets in Karlshamn and Ronneby are identical, you can now copy the Karlskrona subnet and place it over the appropriate geographic position.

In order to copy the subnet you must first return to the parent subnet. This is done either by clicking on the **Go to Parent Subnetwork** button in the toolbar or by right clicking in the workspace and choosing **Go to Parent subnetwork** from the pop-up menu. After returning to the parent subnet, select the subnet and copy it either by **Edit** → **Copy** or by pressing `<CTRL> + C`. Paste the subnet over Karlshamn and Ronneby by selecting **Edit** → **Paste** or by pressing `<CTRL> + V`. Rename the subnets accordingly, i.e., Karlshamn and Ronneby respectively. Next you should connect the Karlshamn and the Karlskrona subnets to Ronneby. To do so, select a **LAN_Mod_PPP_DS0** link in the object palette.

39 Draw a **LAN_Mod_PPP_DS0** link from **Karlskrona** to **Ronneby**. A dialog box appears asking which nodes in each subnet are to be endpoints of the link.

40 For **node a** choose the **Karlskrona.router** node and for **node b** choose the **Ronneby.router** node. (See Figure 7)

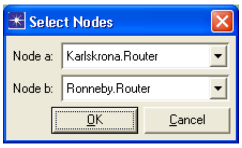

Figure 7: Selecting Nodes

Repeat this process, drawing a similar link between the Karlshamn and Ronneby subnets as well. Specify correctly the link endpoints. Your network should now closely resemble the one shown in Figure 8.

To complete the network configuration, the main office in Ronneby needs to have a switch and a server added to it. In order to configure the network in Ronneby double-click on the Ronneby subnet to enter its subnet view.

41 Place a **Bay Network Accelar1050** node and one **ethernet_server** node in the workspace and rename them to **switch** and **FTP** respectively.

42 Connect the router and the server to the switch with **10BaseT** links and close the object palette.

The FTP server needs to be configured to support the FTP application. Open the attributes dialog box for the FTP server.

43 Choose **Edit** for the *Application: Supported Services* in the *Applications* list entry.

44 Change number of **Rows** to 1.

45 Select **File Transfer (Heavy)** from the *Name* column pop-up menu.

46 Close the supported services dialog box, and the FTP attributes dialog box.

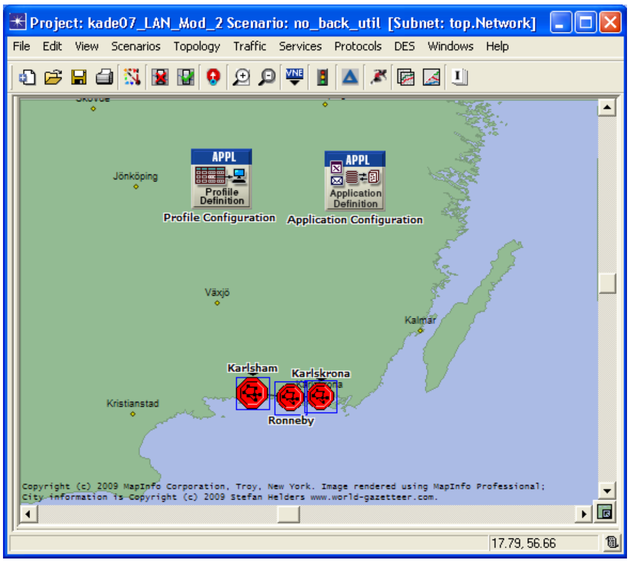

Figure 8: Graphical overview of the scenario.

Return to the parent subnet view and save the project. You have now created a model to act as a baseline for the performance of the network. Background traffic will now be added to the links connecting the cities. The results from the two scenarios will be later simulated and compared.

We begin by duplicating the current scenario in order to be able to compare the simulation results later.

47 Select **Scenarios** → **Duplicate Scenario**.

48 Name the new scenario to `back_util`.

Network studies show that traffic rises gradually over the course of the day as employees and students arrive and start using the network. In order to model the background utilization, you need to know the link type and time period for a particularutilization. Moreover, remember to have the same values for the background utilization for both directions of the link, i.e., we have duplex links.

49 Select the link between Karlskrona and Ronneby. Right-click on it and choose **Similar Links** from the pop-up menu.

50 Display the **Edit Attributes** dialog box for the link between Karlskrona and Ronneby. Expand *Traffic Information* and add a row. Change the background utilization of the link according by editing *Traffic Load (bps)* to the following:

- From 0 to 300 seconds: 30% background utilization.
- From 300 to 500 seconds: 40% background utilization.
- Over 500 seconds: 50% background utilization.

You must enter the link bandwidth here. To obtain this value you must calculate the utilization bandwidth in bps starting from the speed of the link.

51 Check the **Apply to Selected objects** check box in the attributes dialog box.

Check both links and make sure that both are configured properly.

Save the project. Now you have configured two different scenarios for the same network topology, i.e., one without background utilization and one with background utilization. You are now ready to collect data and analyze it. The relevant statistics for this laboration exercise are the following:

- Utilization statistics for the network links.
- Global FTP downloads time for the network. As you are still configuring the **back_util** scenario you will now collect the necessary statistics for this scenario.

52 Right-click in the workspace and select **Choose Individual DES Statistics**.

53 Select the **Global Statistics** → **Ftp** → **Download Response Time (sec)** statistic. (See Figure 9)

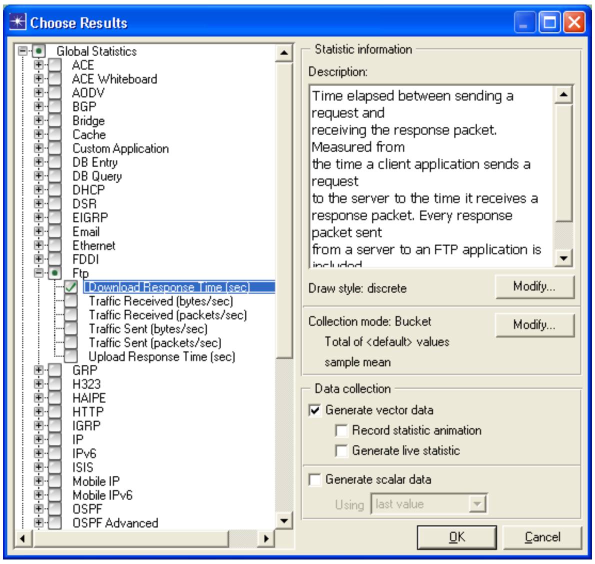

Figure 9: Marking Statistics

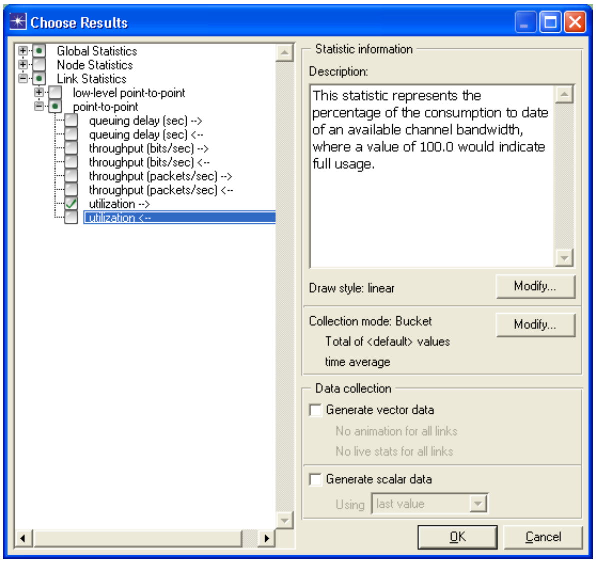

Figure 10: Marking Statistics

54 Select the **Link Statistics** → **point-to-point** → **utilization** –> statistic. (See Figure 10)

In order to compare the statistics in the **back_util** scenario with the **no_back_util** scenario, the same statistics must be collected in the **no_back_util** scenario as well. Change scenario and select the correct statistics by selecting **Scenarios** → **Switch To Scenario**, and then choose **no_back_util**.

55 Collect the same statistics as in the **back_util** scenario.

Save the project. The statistics are now ready to be collected by running the simulations. Instead of running each simulation separately, you can batch them together to run consecutively. To do so, select **Scenarios** → **Manage Scenarios**.

56 Click on the *Results* value for the **no_back_util** and **back_util** scenarios and change the value to collect from the pop-up menu.

57 . Set the simulation duration for each scenario to `30 minutes`.

OPNET Modeler will now run simulations for both scenarios. A simulation sequence dialog box shows the simulation progress. Shut down the dialog box when the simulations are done. Now, if everything went all right you are now ready to view and compare the results of the two simulated scenarios.

To view the results from two or more different scenarios against each other, you can use the **Compare Results** feature. With this topic you can also apply different built-in filters to the graphs.

58 In the **Compare Results** dialog box, select **Object Statistics** → **Choose From Maps Network** → **Karlshamn** ↔ **Ronneby[0]** → **point to point** → **utilization ->**. >>P17

Furthermore, you will also have to change the filter menu from **As Is** to **time_average**. This must be done because as utilization varies over the course of a simulation and it is therefore helpful to look at time average for this statistic.

59 Display the graphs.

2.2.3 Reflection

You may Want to look at the utilization of other links to determine the maximum utilization of any link. Look also at the global FTP response time. The simulation exercise is now completed. Try to explain the results obtained in the simulation. Why do the graphs look like they do? What are the most important parameters that influence network traffic as you use them in this simulation study? What is the influence of the application simulated in these results?

2.3 The TCP congestion algorithm in OPNET

Previous versions of Transport Control Protocol (TCP) start a connection with the sender injecting multiple segments into the network, up to the windows size advertised by the receiver. This is fine when the hosts are placed on the same LAN. But if there are routers and slower links between the sender and the receiver different problems can arise. Some intermediate router must queue the packets and it is possible for the router to run out of space in the queue. The algorithm to avoid this is called *slow start*.

Beginning transmission into a network with unknown conditions requires TCP to slowly probe the network to determine the available capacity, in order to avoid congesting the network with an inappropriate large burst of data.

Slow start adds another window to the sender's TCP, i.e., the congestion window, called *cwnd*. When a new connection is established with a host on another network, the congestion window is initialized to one segment (typically 536 bytes or 512 bytes).

The sender starts by transmitting one segment and waiting for its Acknowledgment (ACK). When that ACK is received, the congestion window is increased from one to two, and two segments can be sent. When each of those two segments is acknowledged, the congestion window is increased to four. This provides an exponential growth, although it is not exactly exponential because the receiver may delay its ACKs, typically sending one ACK every two segments that it receives. The sender can transmit up to the minimum of the congestion window and the advertised window. The congestion window is flow control imposed by the sender, while the advertised window is flow control imposed by the receiver.

At some point the capacity of the internet can be reached and an intermediate router will start discarding packets. This tells the sender that its congestion window has gotten too large.

Congestion avoidance is a way to deal with lost packets. Congestion can occur when data arrives on a big pipe (a fast LAN) and outputs on a smaller pipe (a slower WAN). Congestion can also occur when multiple input streams arrive at a router whose output capacity is less than the sum of the inputs. There are two indications of packet loss at a sender: a timeout occurring and the receipt of duplicate ACKs. However, the overall assumption of the algorithm is that packet loss caused by damage is very small (much less than 1%), therefore the loss of a packet signals congestion somewhere in the network between the source and destination. Although congestion avoidance and slow start are independent algorithms with different objectives, in practice they are implemented together. When congestion occurs TCP must slow down its transmission rate of packets into the network, and then invokes slow start to get things going again.

The combined congestion avoidance and slow start algorithms require that two variables are maintained for each connection:

1. A congestion window (*cwnd*).
2. A slow start threshold size (*ssthresh*).

The combined algorithm operates as follows:

1. Initialization for a given connection sets *cwnd* to one segment and sstresh to 65535 bytes. The initial value of *cwnd* must be less than or equal to 2 × SMSS bytes and must not be more than 2 segments. SMSS, Sender Maximum Segment Size, is the size of the largest segment that the sender can transmit. The initial value of *cwnd* may be arbitrarily high (some implementations use the size of the advertised window), but it may be reduced in response to congestion.
2. The TCP output routine never sends more than the minimum of *cwnd* and receiver's advertised window.
3. When congestion occurs one-half of the current window size is saved in *ssthresh*. Additionally, if the congestion is indicated by a timeout, *cwnd* is set to one segment. Congestion is indicated by a timeout or the reception of duplicate ACKs.
4. When new data is acknowledged by the other end, increase cwnd. The way in which *cwnd* is increased depends on whether TCP is performing slow start or congestion avoidance. If *cwnd* is less than or equal to *ssthresh*, TCP is in slow start, otherwise TCP is performing congestion avoidance. Slow start continues until TCP is halfway to where it was when congestion occurred, and then congestion avoidance takes over. This is done due to the recorded half of the window size that caused the problem.

As mentioned earlier slow start increases congestion window (*cwnd*) exponentially. Congestion avoidance on the other hand dictates that congestion window be incremented by *segsize* × *segsize* / *cwnd* each time an ACK is received, where segsize is the segment size and *cwnd* is maintained in bytes. This results in a linear growth of *cwnd*, compared to slow start's exponential growth. The increase in *cwnd* should be at most one segment each Round Trip Time (RTT) (regardless how many ACKs are received in that RTT) whereas slow start increments *cwnd* by the number of ACKs received in a RTT.

Fast retransmit is a modification of the congestion avoidance algorithm. The TCP sender should use fast retransmit algorithm to detect and repair loss, based on incoming duplicate ACKs. The fast retransmit algorithm uses the arrival of 3 duplicate ACKs (4 identical ACKs without the arrival of any other intervening packets) as an indication that a segment has been lost. After receiving 3 duplicate ACKs, TCP performs a retransmission of what appears to be the missing segment, without waiting for the retransmission timer to expire. The fast retransmit algorithm first appeared in the 4.3 BSD Tahoe release.

Congestion avoidance without slow start is performed after fast retransmit sends what appears to be the missing segment. It is an improvement that allows high throughput under moderate congestion, especially for large windows. In this case the reason for not performing slow start is that the receipt of duplicate ACKs tells TCP that more than just one packet has been lost. Since the receiver can only generate the duplicate ACK when another segment is received, that segment has left the network and is in the receiver's buffer. In other words there is still data flowing between the two ends, and TCP does not want to reduce the flow abruptly by going into slow start. The fast recovery algorithm appeared in the 4.3 BSD Reno release. The fast retransmit and fast recovery algorithms are usually implemented together as follows:

1. When the third duplicate ACK is received, set *ssthresh* to no more than onehalf the current congestion window, *cwnd*, but no less than two segments. Retransmit the missing segment, and then set *cwnd* to *ssthresh* plus 3 times the segment size. This increases the congestion window by the number of segments that have left the network and which the other end has cached.

2. Each time another duplicate ACK arrives, increment *cwnd* by the segment size. This inflates the congestion window for the additional segment that has left the network.

3. Transmit a segment (packet) if allowed by the new value of *cwnd* and the receiver's advertised window.

4. When the next ACK arrives that acknowledges new data, set *cwnd* to *ssthresh*. This ACK should be the acknowledgment of the retransmission from step a, one RTT after the retransmission. Additionally, this ACK should acknowledge all the intermediate segments sent between the lost packet and the receipt of the first duplicate ACK. This step is congestion avoidance, since TCP is down to one-half the rate it was at when the packet was lost.

2.3.1 Goal

The goals of this laboration assignment are as follows:

- To study the behavior and implementation of slow start and congestion avoidance algorithms.
- To study modifications to the congestion avoidance algorithm, namely fast retransmit and fast recovery.

2.3.2 Methodology and results

The student should follow the following steps in the order described in this laboration manual in order to complete the laboration exercise:

1 Create a new project. (**File** → **New**).

2 Name the project `acronym_TCP` and the scenario **NoDrop**. Here, acronym must be substituted with the student's own **acronym**!

3 Create an empty scenario with Europe as a map and no technologies included.

4 Open the internet toolbox object palette if it's not already open and add an **Application Config** object to the workspace. Rename this object to `Applications`.

5 Edit the *Application Definitions* of the object and set the *Application name* to `FTP_Application`. Choose the FTP application and set the following values (**Application Definitions** → **Row 0** → **Description** → **FTP**). (See Figure 11 and Table 1)

6 Add a **Profile Config** object to the workspace and rename it to **Profiles**. (See Figure 12)

7 Edit the *Profile Configuration* of this object by adding a new profile and naming it `FTP_Profile`.

8 Set *Operation Mode* to `Serial (Ordered)`, the *Start Time* must be `constant (100)` and the *Duration* should be set to the `End of the Simulation`. Moreover, *Repeatability* should be `Once at Start Time`.

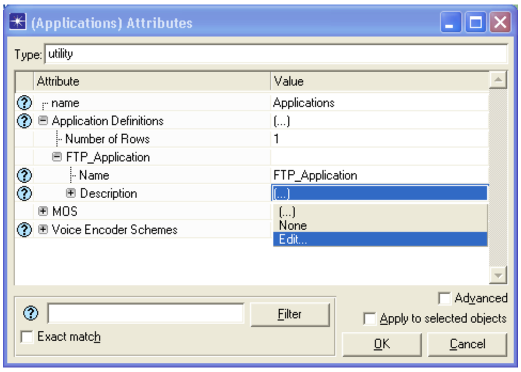

Figure 11: Application Definitions

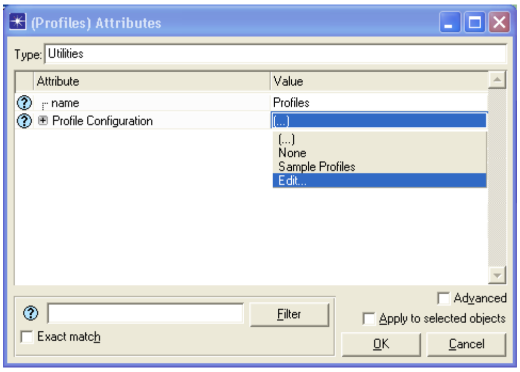

Figure 12: Profile Configuration

ATTRIBUTE	VALUE
Command Mix (Get/Total)	100%
Inter-Request Time (seconds)	constant (3600)
File Size (bytes)	constant (9000000)
Symbolic Server Name	FTP Server
Type of service	Best Effort (0)
RSVP Parameters	None
Back-End Custom Application	Not Used

Table 1: Configuration settings for the Application Definitions

9 *Applications* should be named **FTP_Application**. Set *Start Time Offset* to `constant (5)` and *Duration* to `End of Profile`. Set *Repeatability* to `Once at Start Time`. (See Figure 13)

After the configuration of the application and profile objects, we will create now the network for analyzing the behavior of the TCP protocol. Place a **subnet** over Paris in the workspace and name the subnet **Paris**.

Enter now the Paris subnet.

10 Place an **ethernet_server** in the workspace and rename it to `Server_Paris`. Place an **ethernet4_slip8_gtwy** router in the workspace next to the server and rename it to `Router_Paris`. Connect the server and the router with a **100BaseT** cable. (See Figure 14)

11 Open the server's attributes and add **FTP_Application** as a *Supported Service* in the *Applications* list item. Set the *Server Address* to `Server_Paris`.

12 Expand *TCP Parameters* and disable both *Fast Retransmit* and *Fast Recovery*.

Save your changes and exit the Paris subnet. Now place a **subnet** over Stockholm and set its name to `Sthlm`.

Enter now the Stockholm subnet.

13 Place an **ethernet_wkstn** in the workspace. Rename it to `Client_Sthlm`. Furthermore, place an **ethernet4_slip8_gtwy** router in the workspace next to the client and rename it to `Router_Sthlm`. Connect the client and the router with a **100BaseT cable**. (See Figure 15)

14 Edit the attributes of the client such that it uses the **FTP_Profile** created earlier and set the client address attribute to **Client_Sthlm**.

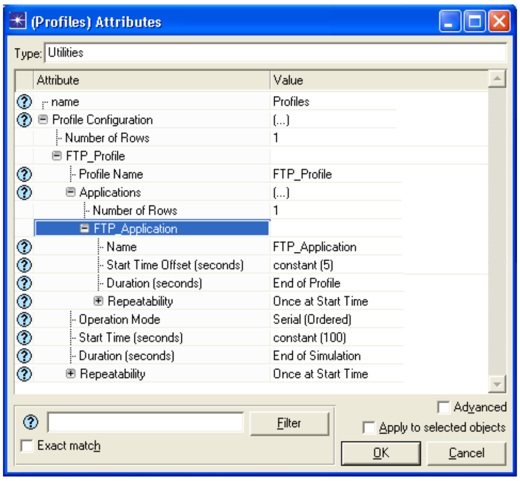

Figure 13: Application Attributes

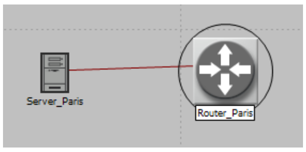

Figure 14: Connecting the Paris server with router

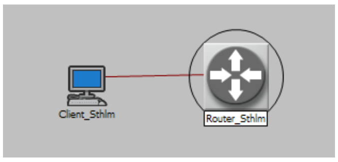

Figure 15: Connecting the Stockholm server with router

15 Set the symbolic name of *Application: Destination Preferences* to **FTP Server** and set the *Actual Name* to the name of the server in Paris.

Save your changes and exit the Stockholm subnet. We will model now the networks between Paris and Stockholm as an Internet Protocol (IP) cloud. Place an **ip32_cloud** on the workspace between Stockholm and Paris and set its name to **Europa_Internet**.

16 Connect the router in the **Paris** subnet to the **Europa_Internet** IP Cloud with a **PPP_DS3 cable**.

17 Connect the router in the **Sthlm** subnet to the **Europa_Internet** IP Cloud with a **PPP_DS3 cable**.

The IP Cloud is now configured and your configuration should resemble the one depicted in Figure 16. Save your project.

Before running the congestion avoidance simulation you will need to select the statistics that we want to analyze later. Thus, for the server in Paris you will need to select the *Congestion Windows Size (bytes)* statistic (**Node Statistics → TCP Connection → Congestion Windows Size (bytes)**).

18 Right click on the *Congestion Window Size (bytes)* and select Change Collection mode. Check the *Advanced* checkbox in the pop-up dialog and change *Capture mode* to **all values**.

Run the simulation 10 minutes. Visualize now the results of the simulation.

19 View the results by choosing: **Object statistics → Choose From Maps Network → Paris → Server Paris → TCP Connection Congestion Window Size (bytes)**. >>P13

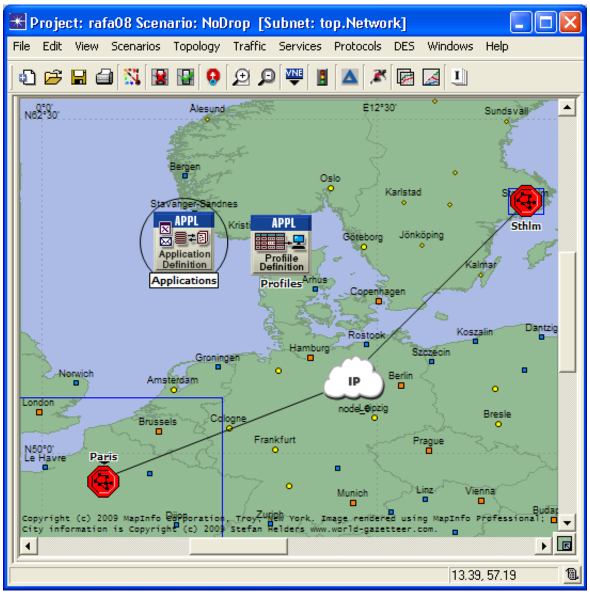

Figure 16: Outline of the European scaled Scenario

Two additional scenarios will be created to study the behavior of *fast retransmit* and *fast recovery* algorithms. The network just created was assumed to be perfect with no packet loss. In the following scenarios packet loss will be introduced. In the following scenarios packet loss will be introduced.

20 Duplicate the scenario and name it `Tahoe`.

21 Edit the attributes of the IP cloud such that we now have a packet discard ratio of 0.5%.

22 Edit the attributes of the server in Paris such that the TCP parameters will have the *Fast Retransmit* algorithm enabled.

Save your new scenario (and the project). Now you will need to duplicate this scenario to enable the fast *Retransmit Recovery* as well.

23 Duplicate the scenario and name it `Reno`.

24 Edit the attributes of the server in Paris such that the TCP parameters will have the *Fast Recovery* enabled and set to **Reno**.

Save the new scenario as well and rerun the simulation for all three scenarios simultaneously. Visualize now the results.

The first graph illustrates the **NoDrop** scenario which has no packet loss. The second graph illustrates the **Tahoe** scenario which has 0.5% packet loss. When congestion is indicated by a timeout, *cwnd* is set to one segment. In other words, slow start is performed. The third graph illustrates the **Reno** scenario which also has 0.5% packet loss. The *cwnd* size does not drop to zero as in the **Tahoe** graph. *Fast recovery* is performed instead of *slow start*.

2.4 Investigating OSPF

2.4.1 The OSPF Protocol

The Open Shortest Path First (OSPF) protocol is an Interior Gateway Protocol (IGP) used for routing in IP networks. As a link state routing protocol, OSPF is more robust against network topology changes than distance vector protocols such as Routing Information Protocol (RIP), Interior Gateway Routing Protocol (IGRP), and Enhanced Interior Gateway Routing Protocol (EIGRP). OSPF can be used to build large scale networks consisting of hundreds or thousands of routers. OSPF uses the Dijkstra's algorithm to compute the shortest path to a destination. The algorithm calculates the shortest path to each destination based on the cumulative cost required to reach that destination. The cumulative cost is a function of the cost of the various interfaces needed to be traversed in order to reach that destination. The cost (or the metric) of an interface in OSPF is an indication of the overhead required to send packets across that interface. The cost of an interface is calculated based on the bandwidth – it is inversely proportional to the bandwidth of that specific interface (i.e., a higher bandwidth indicates a lower cost). For example, the cost of a T1 interface is much higher than the cost of a 100 Mbit Ethernet interface because there is more overhead (e.g., time delays) involved in crossing a T1 interface. Some characteristic features of the OSPF protocol are:

- Link-state based.
- Runs directly over IP.
- Interior or border gateway protocol.
- Multiple paths to each destination load balancing.
- Link-attribute based costing costing is statically assigned.

2.4.2 Goal

The goals of this laboration assignment are as follows:

- and simulate the OSPF routing scheme in an IP network.
- Implement a simplified static routing scheme.
- Implement a simplified OSPF routing scheme.

2.4.3 Methodology

1 Create a new project (**File** → **New**).

2 Name the project `acronym_OSPF` and the scenario to `NoAreas`. Here, acronym must be substituted with the student's own **acronym**!

3 Select an empty scenario.

4 Select Office and set both the **x span** and **y span** to `200`. Do not include any technologies.

Review the values and click OK. Open the object palette and change the palette to **routers**. (See Figure 17)

5 Place 10 **slip8_gtwy**'s in the workspace.

6 Change the object palette to **internet_toolbox**.

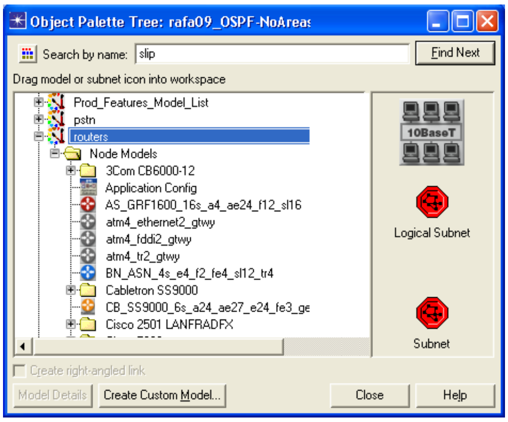

Figure 17: Selecting routers from the *Object Palette*

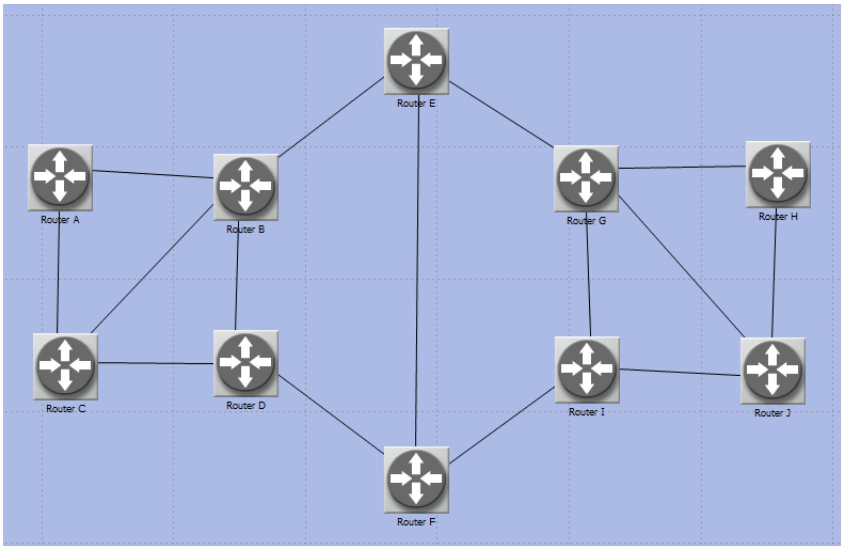

Figure 18: Overview of the network

7 Connect all the routers using **PPP_DS3** links and rename the nodes according to Figure 18.

We need to designate the interfaces of all routers that use the OSPF protocol. By default, RIP is used on every router interface. There are three ways to configure router interfaces to use a particular set of routing protocols which are depicted in Table 2.

Table 2 is for informational use only!

The easiest way to designate routing protocols is via the **Configure Routing Protocols** operation from the **Protocols** → **IP** → **Routing** menu.

This operation has the same effect as manually setting the interface routing protocol attributes, but with the added advantage of being able to configure multiple interfaces at the same time.

NOTE: *The previous setting on the interface is overwritten each time this operation is used!*

8 Assign OSPF as the routing protocol for your network. You should select the **All interfaces (including**

METHOD	CHARACTERISTICS	WHEN TO USE
Protocols → IP → Routing → Configure Routing Protocols	• Any number of interfaces can be configured at the same time • Overwrites the **IP Routing Parameters → Interface Information → Routing Parameters** attribute • Multiple routing protocols can be specified.	In most cases
IP Routing Parameters → Interface Information → Routing Protocols	• Only one interface can be configured at a time.	When one wants to add a protocol to those already designated on a particular interface.
IP Dynamic Routing Protocol simulation attribute	• Does not modify router attributes. • Overrides the routing protocols configured on the router interfaces for the duration of the simulation. • One routing protocol used on all interfaces • When this attribute is set to *Default* the protocols specified on the router interfaces are used.	You have configured the routing protocols in your network but want to see the effects of running a single protocol throughout the network.

Table 2: Means of assigning link costs.

loopback) radio button in the appropriate selection box and save the project.

A Routing Domain Legend appears in the bottom left corner on the workspace. All links should have a green **O** attached to it. This indicates that **OSPF routing protocol** is used over that link.

The **Protocols → IP → Addressing → Auto-Assign IP Addresses** operation assigns a unique IP address to the connected IP interfaces whose IP address is currently set to auto-assigned. This operation does not change the value of manually set IP addresses.

9 Auto-assign IP addresses for your network.

Cost is specified on a per interface basis and is used as the basis for the shortest path route calculation. There are two ways of setting this cost attribute for each interface.

- *Per-interface*: The interface information table is located by right clicking on a router and selecting the **Edit attributes** option. One can manually specify the cost of an interface by editing the value with the desired cost setting. For example, the default value of **Auto Calculate** can be over-written by any positive integer cost value: When set to auto calculate, the formula used to calculate the cost is based on

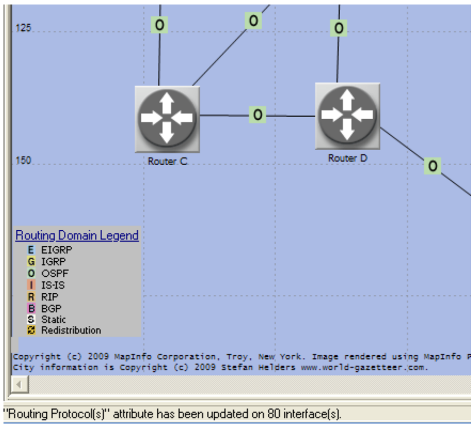

Figure 19: Clarification of the assigned protocols

the interface speed and another configurable attribute called **Reference Bandwidth**:

$$\text{Interface cost} = \frac{\text{Reference Bandwidth}}{\text{Interface Bandwidth}}.$$

NOTE: The default value for the reference bandwidth is 1000 Mbps; consequently, it will cost 1 000 000 000 / 100 000 000 = 10 to traverse a 100 Mbps Ethernet interface while it will cost 1 000 000 000 / 1 544 000 = 647 to cross a T1 serial line interface. The default for interface bandwidth is computed dynamically using the data rate of the connected interface. It can be over-written by using the bandwidth setting in the ***Protocols → IP → Routing → Configure Interface Metric Information*** *table.*

- *Globally for all interfaces*: If wanted to change the interface cost across all interfaces, then, rather than individually setting them on each interface, one can use the model-wide cost configuration option using the following menu option: **Protocols → OSPF → Configure Interface Cost**. This operation will allow for choosing one of the following two cost configuration options:

  1. the reference bandwidth will be set for all routers. All interfaces will be set with a cost value of auto calculate.
  2. all interfaces will be set with the specified cost value, i.e., the interface/bandwidth settings will be ignored.

In this laboration exercise you can use different bandwidths on the links to set different costs or set the cost globally. What would be the interface bandwidths if you would like to get the costs illustrated in the Figure 20?

10 Configure the cost for the different interfaces according to the figure below. Choose whatever method you want, as described in the previous paragraph, but make sure the cost configuration scenario corresponds to the one in Figure 20.

Now is time to configure the traffic demands between several routers in your network. (**Traffic → Create Traffic Flows → IP Unicast**)

11 Create a traffic demand between router B and router D with traffic originating from router B.

12 Create a traffic demand between router C and router J with traffic originating from router C.

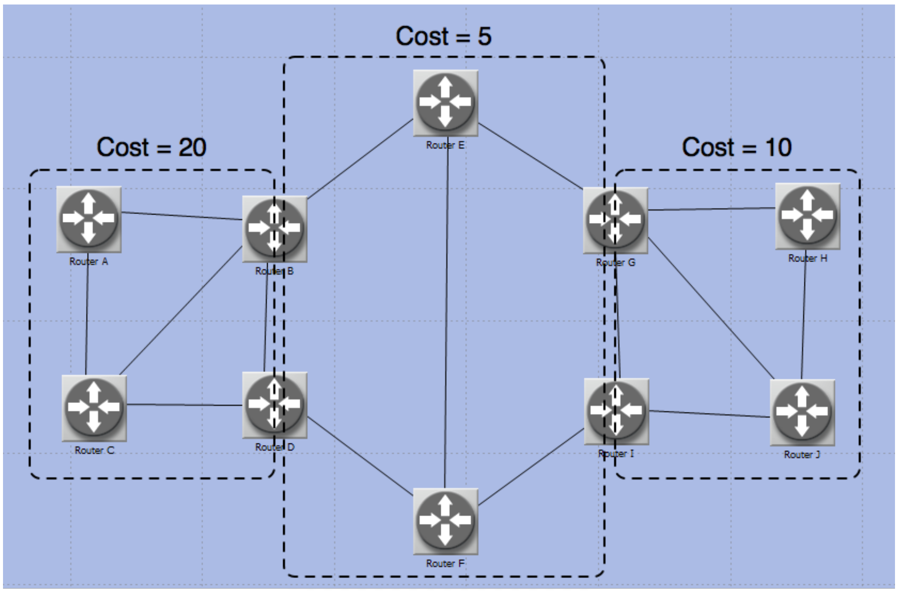

Figure 20: Assigned OSPF link cost

The paths of the traffic demands should be now visible. To hide the traffic demands select **View** → **Demand Objects** → **Hide All**. Save the project.

In the scenario you just created all routers belong to the same level of hierarchy, i.e., one area. No load balancing where enforced for any routers. Two new scenarios must be now created in order to implement an areas scenario and a load balancing scenario. The major addition in OSPF configuration, relative to other protocols, is that the OSPF routing domain can be divided into smaller segments called areas. This reduces memory and computational load on the routers. Each area is numbered and *there must always be an area zero,* which is the OSPF backbone. All other areas attach to the backbone either directly or via *virtual links*.

An area should contain no more than about 50-100 routers for optimum performance. A router that connects to more than one area is called an Area Border Router (ABR).

13 Duplicate current scenario (**Scenarios** → **Duplicate Scenario**) and name the new scenario **Areas**.

You must now partition the network into areas. This is a physical partitioning in the sense that an interface can belong to only one area. The distinct interfaces of the same router may still belong to separate areas. This can be achieved through **Protocols** → **OSPF** → **Configure Areas** dialog.

14 Partition the OSPF network into different areas such that the links with cost 20 belong to area 1, links with cost 5 belong to area 0 and links with cost 10 belong to area 2 (for reference see the cost figure on previous page).

15 Verify your area partitioning by visualizing the different areas in different colors (**View** → **Visualize Protocol Information** → **OSPF Area Configuration**).

Save your new scenario. Now you will need to create a load balancing scenario. Load balancing is a concept that allows a router to take advantage of multiple best paths (routes) to a given destination. If two routes to the same destination have the same cost, the traffic will be distributed equally among them.

16 Go back to the **NoAreas** scenario (**Scenarios** → **Switch To Scenario** → **NoAreas**).

17 Duplicate the scenario and name the new scenario **Balanced**.

18 Select both router C and router J and enable them to act as load balanced routers (**Protocols** → **IP** → **Routing** → **Configure Load Balancing Option**). Be sure that *Packet-Based* option is chosen and do not forget to save your changes. (See Figure 21)

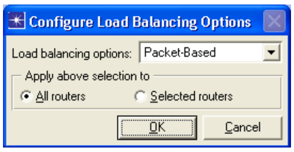

Figure 21: Packet-BasedLoad Balancing

Configure OPNET to run the simulation for 30 minutes for each scenario. Display the simulation results for each scenario (**Protocols** → **IP** → **Demands** → **Display Routes for Configured Demands**).

2.4.4 Reflection

Why does the traffic flow look this way? Explain the results for each scenario.