Leopold-Franzens-University
Innsbruck

Institute of Computer Science
Distributed and Parallel Systems

Extension of the link state routing functionality of the IRV-Tool

Bachelor Thesis

handed in to Muhammad Ali

Philipp Gschwandtner

Innsbruck, February 23rd, 2007

Abstract

The purpose of this thesis is to extend the functionality of the IRV-Tool written by Christian Sternagel. The IRV-Tool allows the visualization of routing algorithms in networks while not requiring the user to know very much about the protocols in use. Therefore it gives easy access to experimentation with routing protocols and algorithms.

Diese Arbeit beschäftigt sich mit der Erweiterung der Funktionen des IRV-Tools von Christian Sternagel. Das IRV-Tool ermöglicht es, Simulationen von Routing-Algorithmen in Netzwerken durchzuführen, ohne dass der Benutzer dabei die genauen Details der verwendeten Protokolle kennen muss. Somit erhält man eine einfache Möglichkeit mit Routingprotokollen und –algorithmen zu experimentieren.

Table of Contents

Chapter 1

Introduction

This chapter tries to deliver insight on the question what routing exactly is and why there is need for routing protocols. Furthermore the two routing protocols supported by IRV-Tool – RIP and OSPF – as well as their differences are discussed.

1.1. Routing and the IRV-Tool

Routing is a very important aspect of today's networks. Typically networks consist of many nodes that can be divided into two groups. For one thing there are hosts: They are usually personal computers, laptops, PDA's or some other piece of equipment that represent the end of a network connection (since they normally only have a single connection to the network, which is also assumed by the IRV-Tool and this thesis). Basically the only job they have to do is send and receive packets. For another thing there are routers: In general they have at least two connections to one or more networks. Their responsibility is to ensure the network's consistency and forward packets according to certain rules (e.g. "fastest way from A to B", etc…). Routing is the technique by which packets are forwarded across a network. In small networks it might be possible to solve these problems by having administrators manually enter routes and also change them to adapt to new situations. But for

bigger networks a more sophisticated solution is needed, especially if the configuration or state of the network changes frequently.

There are many routing protocols which facilitate the process of bringing a network to a consistent state. They are simply standards that define ways on how routers may complete this process without the need of manual intervention. This is very important when a network comes online for the first time for example after an electrical power outage. But there are many situations in which a network can become inconsistent (e.g. if a connection becomes unreliable or fails completely) and in which maintenance is needed.

The IRV-Tool[1] (Internet Routing Visualization) written by Christian Sternagel is an easy-to-use program to simulate and visualize how routing protocols work. One can easily design a network scenario by simply placing nodes (hosts or routers) and linking them with each other through connections that can have user-defined costs. The tool supports saving & loading scenarios as well as the use of two routing protocols, RIP (Routing Information Protocol, also known as Distance Vector Routing) and the more complex OSPF (Open Shortest Path First, also known as Link State Routing) along with some of their features. It also supports the sending of packets to actually see the route they take on their way through the network. In addition, the IRV-Tool allows to set starting and cycle times for routers to manually create critical situations for example, in which RIP might take too long or fail completely to converge to a consistent state. Another very useful feature is the terminal, supporting batch-like commands to run scripts and automate sequences.

1.2. Motivation

The reason for the focus of this thesis is the author's interest in computer networks and the IRV-Tool itself. The program is very easy to use and allows students to quickly simulate network situations and see how routing protocols respond to certain scenarios. There are many network simulators like ns[2], which are very powerful and

[1] http://www.welzl.at/research/tools/irvtool/index.html, GPL Licence

[2] Network Simulator, http://www.isi.edu/nsnam/ns/

capable of simulating and visualizing (using nam) complex computer networks. But unlike IRV-Tool, ns requires very detailed specifications of the network to be simulated (along with some knowledge of the TCL programming language).

The program written by Christian Sternagel takes computer networks to a more abstract level where one does not need to bother much about the length of packets or which type of packet queues are used. But the IRV-Tool does not fully implement all features of the protocols supported, therefore making it impossible to simulate many interesting situations and limiting its application area. The purpose of this thesis is to extend this functionality, allowing a wider spectrum of scenarios to be analysed.

1.2. Differences between RIP and OSPF

RIP is a very simple routing protocol of the "distance vector" family of routing protocols (which is the main reason for its common use, since there is little difficulty in its implementation compared to other routing protocols). It leads to routing tables with entries that consist mainly of four values: The destination, the connection to be used, the cost of this connection and a time value (which is important to distinguish between old and new records). Upon receiving a packet, a router compares the packet's destination with the destinations in its own routing table and resends the packet on the according connection. RIP in its base design defines the cost of each connection as 1 and the cost of a path consisting of many connections as their sum – the "hop count". The maximum number of hops was 15, 16 was regarded as infinity – therefore "unreachable".

Routers update these tables by simply receiving routing entries of other routers. Every router has at least some local knowledge – its own routing entry ("loopback", with cost = 0) and the number of outgoing connections to be more precisely.[3] This information is broadcasted to all of its neighbours, which then add an entry for this router with the sum of the original cost (in this case 0) and the cost of the connection over which the information has been received.

[3] [Huit 00], p. 86

Due to the way RIP works, every router only contains information about the connections to its neighbours and no information about the remaining network. This makes RIP simple, but fault-prone by design. It leaves the possibility of undesirable loops (exactly due to the fact that routers only know the next hop but not the whole route of a packet) or bouncing-effects (extensions like “triggered updates”, “split horizon” or “split horizon with poisonous reverse” try to minimize these problems). See [Huit 00], p. 95-97, for more detailed descriptions of the problems and on how these solving techniques work.

OSPF (Open Shortest Path First), a protocol of the link state family, generally avoids these inconsistencies by a simple fact: Every router using OSPF has routing entries for the whole network, containing every single connection. For example any router has information about a link between two others even if it is not directly connected to them. Therefore, undesired loops are impossible since routers both have a map of the whole network and perform a complete calculation of the best routes for any packet, not just the next hop.

But there are more differences between OSPF and RIP that make the link state protocol the more complex but better choice. Apart from distinguishing between routers and hosts, it uses three different types of subprocotols (Hello-, Exchange- and Flooding-protocol) for routing-relevant packets to keep the exchange of information as efficient as possible. For a more detailed explanation of the purpose of the subprotocols, see chapter 3 of [Stern 04] or [Huit 00], p. 145 - 151. Another major problem of RIP, especially when dealing with large networks, is the fact that it is limited to a maximum hop-count (number of nodes along the path of a packet) of 15. A RIP network that spans more than 15 hops is considered unreachable. OSPF on the other hand has no limitation for the maximum number of hops, thus allowing its use for very large networks. Nevertheless plain OSPF that is used across the whole network is not the best way to use it. To limit the explosion of link state updates over the whole network – which would greatly increase data traffic as well as decrease the networks applicable performance – OSPF supports the definition of logical networks (also called “areas”). This keeps the router’s databases small (see chapter 2 for more information).

Chapter 2

Extension

In this chapter we will discuss the extent of OSPF's implementation in the IRV-Tool and the extensions added within the scope of this thesis. Moreover example scenarios are given to show the implications of these extensions.

2.1. Current implementation

At this time, the IRV-Tool supports OSPF in its basic version (with one additional feature, called "multiple paths", which enables load-balancing using different routes for different packets which are all having the same source and destination address – this can increase the throughput of a network greatly). The routers use a link state database that consists of entries with six values:

source	destination	connection	cost	time	lollipop-number

Figure 2.1: The format of the link state database used by routers

It simply contains two entries for every connection (assuming that data transfer is possible in both directions), both its nodes as well as its cost. Apart from that, the time value and the lollipop- or sequence number from the corresponding link state

advertisement are saved to distinguish between old and new records, as well as to remove records that have reached a maximum age.

The routers use the SPF-Algorithm[4] to find the best route (in terms of least costs) for packets and forward them on the corresponding link.

After start-up, the routers use the Hello-protocol to check that the links are operational and to elect a designated router (responsible for originating link state advertisements) and a backup router. These Hello-packets are sent in intervals, adjustable by IRV-Tool's OSPF-variables option dialog (default are nine time units). After a connection between two routers has successfully been established, they must synchronize their databases. This initial synchronization is done by the use of the Exchange-protocol. It selects a master and a slave router, whereas the master then sends its database in an exchange packet to the slave, which then returns an exchange packet containing its own database. By the time the routers are finished with this process, the network is fully operational – provided that the state of the links doesn't change. However if the state of a link does change (e.g. a connection fails completely), the router responsible for this link will broadcast this new information using the Flooding-protocol. Its purpose is to quickly spread new link state records across the network.

2.2. Multiple Metrics

Today's smaller networks, at home or at any company, are usually made up of (almost) homogeneous network components. This means that the bandwidth, delay and other possible parameters that affect data transfer across the network are approximately the same for every connection. But in bigger networks, or in those that are used for high-technology applications, there are often connections that vary in many aspects.

Let's think of an example in real life. The Internet is the biggest network in the world, itself made up of (proportionally) smaller subnets. Overland, there is no

[4] Shortest-Path-First-Algorithm, also known as Dijkstra-Algorithm, see [Huit 00], p. 125

difficulty in linking these subnets together, but concerning the intercontinental connection there are far less options. Facing two of them, there are satellite links and submarine cables: Satellite links are known for their high bandwidth, making them a very good choice for television and radio broadcasts. However, one must not forget that – using geosynchronous satellites – the signal has to travel the distance of approximately 36.000 kilometres twice in order to get from ground station to ground station. Along with tasks like error-correction (although nowadays this small part of time consumption can be neglected), the delay of the signal will be increased up to 250 milliseconds, making this type of network link ineffective for time-critical applications like voice-over-ip for example. Submarine cables on the other hand have excellent attributes when it comes to delay times, but are somewhat weak concerning reliability (earthquakes for example often damage cables in the Pacific Ocean). These examples are the most prominent, but one also has to deal with these aspects when trying to maximize the performance of a network.

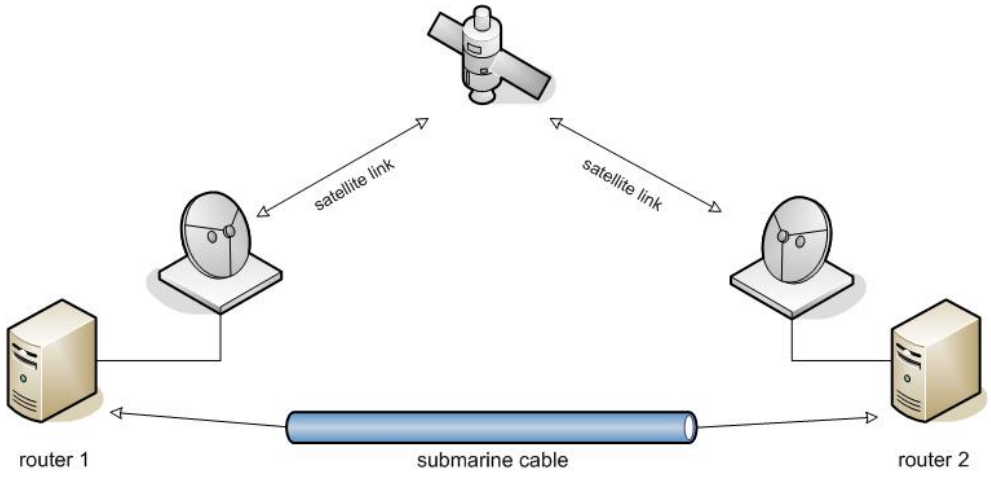

Figure 2.2: A network scenario with different types of connections.

Basic routing protocols like RIP are unable to distinguish between these attributes, since only one type of cost can be used (usually the delay or a pre-calculated average value). OSPF however supports more than one type of cost (these costs are also called metrics), to provide a more detailed map of the network. Therefore, routers can choose the best route more precisely, taking into account the type of data to be transferred. [Huit 00] mentions four different definitions concerning the best route for packets:

- The largest throughput
- The lowest delay
- The lowest cost
- The best reliability, defined as the lowest packet loss probability

To support multiple metrics, the link state database of the IRV-Tool (represented by a two dimensional array with six columns) as been extended to three dimensions to hold the additional metrics. The number of metrics is defined in *Consts.OSPFNUMBEROFMETRICS*, currently set to 4 – their indices are defined in *Package.M_DELAY*, *Package.M_THROUGHPUT*, *Package.M_RELIABILITY* and *Package.M_COST*. Packages now contain a new attribute, which is the metric that is used by any router to choose the best route for their delivery. However, it must be noted that while delay and cost are preferably very small along a route, throughput and reliability should of course be maximized. Since the SPF-Algorithm – by design - always calculates the minimal cost, the difference of throughput and reliability with *Consts.INVERSEMETRICMA*X (with a value of 2^{16} -1) is used respectively for saving and calculating. Hence, a user-specified throughput of 1 will be saved internally as $2^{16} – 2$ (the user does not need to know about this calculation, since it is done by every user interface provided by the IRV-Tool).

This extension of IRV-Tool's OSPF functionality can be activated by choosing "Multiple Metrics" in the OSPF options-menu. Afterwards, it is possible to change the metric hosts will use (default is *Package.M_DELAY*) by both the corresponding option dialog as well as the terminal with the following syntax (after having logged in):

```
set metric <delay | throughput | reliability | cost>
```

In addition to their address, the chosen metric will be displayed by hosts as one of four letters ("D", "T", "R", or "C"). Connections will display all four metrics instead of just one cost – they can be set either via the option dialog or using the terminal provided by the IRV-Tool. The syntax is

```
set <connection> metrics <delay throughput reliability cost>
```

For example the following command sets the metrics of connection no. 3 to delay = 1, throughput = 2, and so on.

```
set 3 metrics 1 2 3 4
```

Routers now also display four metrics for each connection in their routing tables.

To simulate the delay vs. throughput scenario, the IRV-file satellitesubmarine.irv can be used. Host 1 sends a TV-broadcast to host 3 that requires high throughput, whereas host 2 sends a time critical packet (a telephone call for example) to host 3 that requires short delay times. One can visually track the packets, which will take different routes by reason of the different metrics specified for the connections 9, 10 and 11, 12.

2.3. Multiple Areas

The biggest advantage of OSPF over RIP is the fact that every router holds information about the whole network and therefore is able to compute complete routes and not just the next hop. But this is also one of the bigger drawbacks at the same time, because when the size of the network increases, the size of the link state database increases as well (just as the volume of the routing messages and the effort of the best-route computation). As previously mentioned, a link state database has two entries for each connection there is. That means for a network containing five routers, each with separate connections to all others (thus four connections per router), the distance vector table for RIP would consist of eight entries whereas the link state table for OSPF would hold 20 entries. When applying OSPF to bigger networks, memory and computation requirements quickly become excessive and might cancel any advantages of OSPF over RIP.

To overcome this problem, the simplest (and also classical) solution is “hierarchical routing”. This represents a partitioning of the network, splitting it into smaller, independent parts which are all connected by a backbone. In OSPF, these independent parts are called “areas” and the backbone is called “backbone area”. Identification numbers are assigned to these areas with 0 being the backbone area. Each area can be seen as an autonomous network, the routers of which only hold routing information about the links of the own area and not that of others (excluding the area border routers as described later). Hence, route computation is only done

within the area and the flooding protocol basically also stops at the boundaries of an area. The complexity of the routing protocol is thus proportional to the size of the corresponding area, not to the size of the whole network.

When using areas, every router in the network belongs to at least one area (called Area Router, AR) – those who belong to more than one (usually two with one being the backbone area) are called Area Border Routers (ABR). Their purpose is to connect different areas and link the network together. Let's face an example:

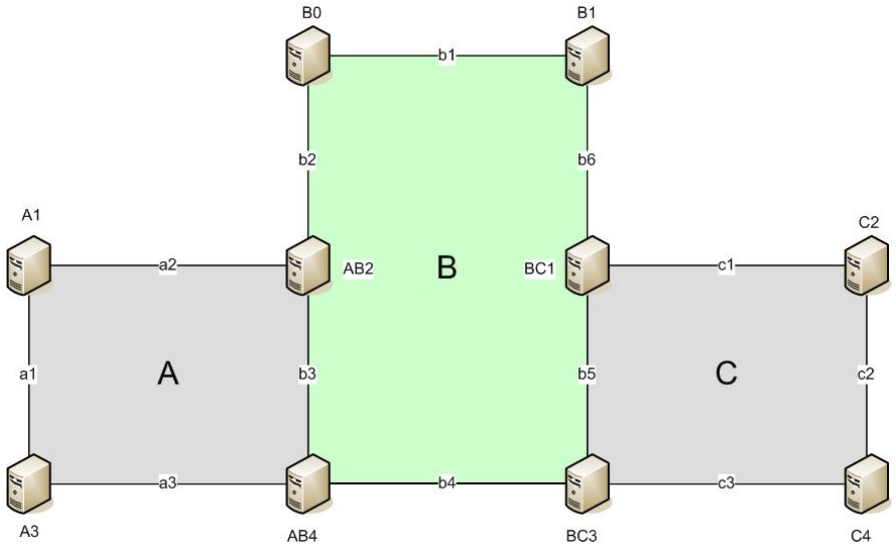

Figure 2.3: A sample network with three areas, A, B and C whereas B is the backbone area.

This sample scenario contains two areas A, C and a backbone area B. The routers A1, AB2, A3 and AB4 are part of A, with the links a1, a2 and 3. The routers BC1, C2, BC3 and C4 belong to the area C with the links c1, c2 and c3. The backbone area consists of the routers B0, B1, BC1, AB2, BC3 and AB4 with the links b1 to b6. BC1, AB2, BC3 and AB4 are area border routers while all others are area routers.

The purpose of this splitting is to reduce the size of the database, e.g. so that A1 only needs to know about the routers in area A and not about every router in the whole network. This consequentially raises the question: How can A1 reach other routers in other areas? To answer this question, we take a closer look at the entries of link state databases. Until now, there has only been one type of entry, which represents a connection from one router to another. To support areas and effectively reduce the amount of memory required, a summarization is needed. This means that the

destination of this new summary record will not be a router but a whole area. These entries – emitted by area border routers – are simply called “summary links” (while the router-to-router entries are naturally called “router links”) and their metric is equal to the length of the route from the corresponding ABR to the network.

In our example, the database of A will contain the following:

- the usual link state records for the links a1, a2 and a3 sent by A1, AB2, A3 and AB4
- the summary records sent by AB2 and AB4 for the backbone area B and area C

AR’s inside an area will still exchange and flood router link records as usual, but unlike summary records, they are not flooded across different areas. The way of routing is now very simple: If router A1 receives a packet with router C4 as its destination, it simply compares its own area with the destination area – since they are different, it will route the packet according to its destination area and not its destination address (= the address of the specified router inside area C). As soon as the packet has been received by a router inside area C, the router will notice that the destination area of the packet and its own area are the same and it will route the packet according to normal OSPF specifications.

By the nature of this solution for the explosion of resource requirements, one might expect problems already encountered with RIP (due to the fact that contrary to basic OSPF, not the whole network is mapped within a router) like loops for example. Although the computation for the metric of a summary record is indeed similar to the one used by distance vector protocols (the metric will be equal to the sum of the metrics along backbone path leading to the area), it does not bear the same risk since there is a strict hierarchy where areas are only connected to each other by the backbone area.

The IRV-Tool has been adapted to support multiple areas by changing the array representing the link state database from six rows to seven, to hold a new value called *TYPE*. This way of implementing OSPF’s “multiple areas”-feature has been

chosen for its relatively small complexity (there is also the possibility of using a second, entire independent link state database for summary records). The table header now looks like this:

type	source	destination	connection	cost	time	lollipop-number

Figure 2.4: The new header of the link state database used by the IRV-Tool to support multiple areas.

The new row either holds the value *LinkStateTable.TYPE_LINK* if the record is a normal link record or *LinkStateTable.TYPE_SUMMARY* if the record is a summary record. Furthermore the class *Router* contains a new attribute called *areas* which holds the ID of all areas this router is a member of. By default, every router belongs to area 0 which represents the backbone area. Normally routers do not require a summary record in their table containing their own area, however using such a summary record greatly simplifies the implementation. Moreover these summary records contain “0” as connection, since they are only used for flooding and not for routing

The ID of the area is displayed in the graphical user interface along with the address of the router using the notation “X.Y” where X is the area the router is part of (if the router is part of more than one area, “ABR” is displayed instead of the area’s ID) and Y is the router’s address. The areas of a router can be changed by either using the option dialog of the corresponding router or the built-in terminal of the IRV-Tool (after logging in to the router). The syntax for changing the areas is:

```
set areas <0-9>
```

Multiple areas can be specified by separating them with whitespaces, for example the following command will set a router to be an ABR that is part of the backbone and area 3:

```
set areas 0 3
```

The order in which the areas are specified does not matter.

2.4. Smaller bug fixes and changes

Apart from the OSPF extensions, a few other bug fixes and smaller changes have been made to the program.

- The terminal now scrolls automatically when displaying new output for a better user convenience.
- It is now impossible to enter negative values for connection metrics, neither as RIP costs nor as OSPF metrics.
- Connections now only display “INF” when the specified cost is greater than 16 only if RIP is the protocol in use (OSPF does not have this limitation).

Chapter 3

Conclusion

3.1. IRV-Tool extended

The IRV-Tool now supports multiple metrics as well as multiple areas, two features of OSPF that enhance its capability to offer best routing qualities. With the extended version of the program, one is able to simulate a greater variety of network scenarios by simply specifying more details. This makes the IRV-Tool even more suitable for simulation and visualization of network settings (for example for teaching at the university).

3.2. Problems

The part of this thesis involving the most work was integrating the new features into the existing program. The features alone follow very simple principles and are easy comprehend. Implementing them without limiting or damaging any other components of the program however is very difficult.

3.2. Compatibility issues

Since the changes that have been made to the IRV-Tool are rather sizable (added new attributes to the Java classes representing network components and extended the set of features the status of which – whether activated or not – is also saved to *.irv files), it is impossible to run older simulation scenario files created with previous versions of the IRV-Tool. Vice versa, it is impossible for older versions of the IRV-Tool to open files that have been created with the extended version of it.

Table of Figures

All figures have been created by the author.

References

[Huit 00] Christian Huitema, Routing in the Internet, 2nd edition, Prentice Hall PTR, Upper Saddle River, 2000

[Stern 04] Christian Sternagel, IRV-Tool, Innsbruck, 2004 (http://www.welzl.at/research/tools/irvtool/irvt-doku.pdf)

[CISC 06] N.n., OSPF Design Guide, Document ID 7039, 2006 (http://www.cisco.com/warp/public/104/1.pdf)