Statistical Convergence Investigation of Routing Protocols

Frank Bohdanowicz, Marcel Jakobs, Christoph Steigner

University of Koblenz
Germany
{bohdan,zimon,steigner}@uni-koblenz.de

***Abstract*—The performance of routing algorithms can only be compared if they undergo a comprehensive convergence analysis. In this paper, we present a new approach for convergence analysis in order to evaluate the benefit of a new distance vector routing algorithm, which is no longer affected by the well-known Counting-to-Infinity (CTI) problem which still occurs in topology loops. The newly developed *Routing with Metric-based Topology Investigation (RMTI)* protocol only uses event-triggered updates. Thus, the convergence time and the update traffic can be reduced. Convergence properties of RMTI are compared with the *Routing Information Protocol (RIPv2)* under the impeded condition of provoked CTIs. In this way, the performance benefit of RMTI can be shown in comparison to RIPv2. The major focus of this paper are the approaches to measure the convergence time of routing protocols in a newly developed test environment. Special effort is directed at minimizing measurement perturbations by the separation of the online data capturing task from the offline data evaluation task. The results show that this convergence measurement method is of universal quality. RMTI is a newly competitive intra-domain routing protocol which can perform filtering policies in contrast to other intra-domain routing protocols.**

***Keywords*- distance vector routing; metric-based topology investigation; routing loops; counting to infinity problem; routing evaluation**

I. Introduction

Routing is the exchange of routing information between routers to provide them with reachability information about destination nodes in the computer network. Decentrally running routing protocols can repair interrupted links if the network topology contains enough loops. In case of a link failure, loops provide alternative links to the same destination node in a network.

However, loops in the topology are also the main reason for the occurrence of routing loops. A routing loop is a circular trace of a routing update message which returns to the same router, either directly from the neighbor router or via a loop topology. It is crucial to detect and to prevent routing loops since they can consume a large amount of network bandwidth and impact the end-to-end performance of the network.

All common routing protocols cope more or less efficiently with the routing loop problem as a routing system is a distributed system. Therefore, new routing protocols should deal with topology loops as well as routing loops in an efficient way in order to provide convergence and stability for the network.

The newly developed *Routing with Metric-based Topology Investigation (RMTI)* protocol [2] can detect routing loops and omit the Counting-to-Infinity (CTI) problem. The well-known CTI-problem stands for the characteristic routing-loop situation in RIP-networks. In a CTI situation the routing metric is constantly increased due to circulating routing update messages in a routing loop. RMTI accelerates the convergence time and reduces the update traffic by only using event-triggered updates together with neighbor-alive-notifications. In contrast to this new approach the common RIP protocol is mainly based on periodic update messages.

To compare the convergence time of routing protocols, a new test environment has been developed. This test environment consists of the following components (see Figure 1):

- A computer network based on virtual linux machines connected by software bridges.
- An online data capturing facility to collect characteristic data during a test run.
- A script-based event generator capable of triggering failure events.
- A topology generator able to generate many different but regularly structured topologies (see Figure 4) and random topologies with pre-defined constraints.
- An offline statistical analysis tool to visualize the results of a test run using plots and graphs.

Network failures which cause CTIs can be triggered, recorded and evaluated with the test environment. The update traffic together with a time stamp is recorded during the online data capturing phase. In the offline data analysis phase, the events, starting with the first network failure up to the end of the convergence process, are analyzed.

A major concern of this research is to include and compare the CTI-problem of common RIP algorithms into the convergence analysis. A convergent state in a distributed routing system is given when all forwarding tables in the routers contain the optimal next hop entries for the given network topology. The convergent state is not given if the forwarding tables do not offer the optimal next hop entries due to a certain network failure.

This paper is structured as follows: In Section 2, other approaches of convergence tests and routing protcols are reflected. In Section 3, the newly developed test environment is introduced. In Section 4, the tested routing protocol is presented. In Section 5, an overview about the routing algorithms tested so far is shown. In Section 6, the tested topologies and the convergence measurement technique is described. Furthermore the results of the convergence analysis are discussed in this section. The paper ends with an outlook of further research items.

II. Related Work

Newly developed routing protocols or enhancements to existing routing protocols are usually tested in simulation test environments. They only analyze a certain part of the respective protocol in order to reveal its features. The authors of [11] and [26] analyze their protocol enhancements by simulation techniques. The test environment described in this paper is not based on simulation, but on emulation technique. It can analyze routing software which is fully implemented and ready to be used, e.g., in company networks.

The test environment is based on the *Virtual Network User Mode Linux (VNUML)* system [20]. A system especially developed for protocol testing in a virtual machine infrastructure. Virtual machines are generated with *User Mode Linux (UML)*[6][7] and are connected together by software bridges [25]. In order to generate many different network topologies and link failure situations, the VNUML system has to be extended by an automatic script software. This script software can also collect a great deal of data within log files.

Currently, there is hardly any research on convergence analysis in deployed networks. Therefore this newly approach is presented in the next section. Several enhanced distance vector protocols have been proposed in order to avoid routing loops as well as the CTI problem. In contrast to RMTI, many of these approaches increase the amount of information exchanged among the nodes of a network.

The Ad hoc On-demand Distance Vector (AODV) protocol [16] by Perkins extends the distance vector information originally based on subnet N, next hop NH and distance D, to a 4-tuple (N,NH,D,SEQ), where SEQ denotes the sequence number. This approach is loop free [15] but it is not compatible with RIP due to the required protocol changes.

The Enhanced Interior Gateway Routing Protocol (EIGRP) used by Cisco is based on the DUAL algorithm proposed by Garcia-Luna-Aceves. DUAL provides loop-free paths at every moment, which was proven in [10]. EIGRP is a Cisco proprietary routing protocol and not compatible with RIP because of a completely different protocol design.

A solution called Source Tracing has been introduced by Cheng *et al* [5] and Faimann [9]. This approach provides additional information in updates and routing tables by adding a first-hop indication to the path. In this way loops can be recognized recursively.

These protocols avoid the CTI problem because they provide loop freedom. Likewise, they are not compatible with the RIPv2 standard or are proprietary approaches.

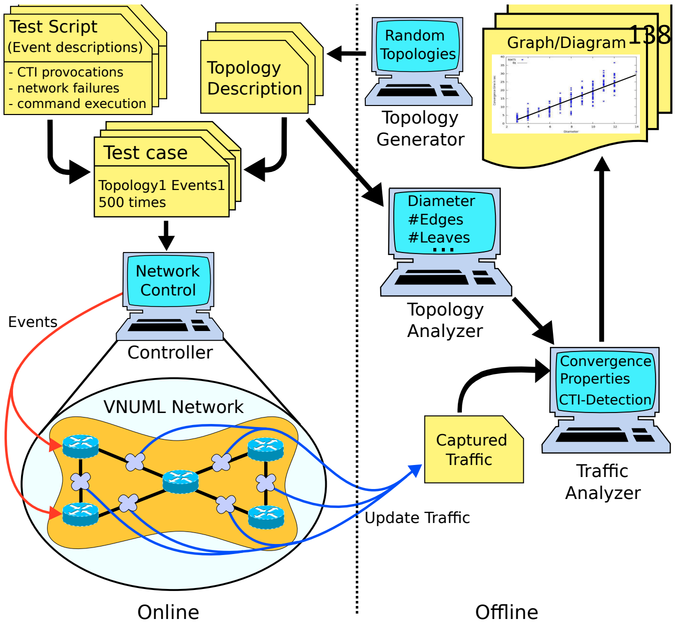

Fig. 1. Test Environment

In contrast to these approaches, we aim to provide a solution that has a complete and sound backward compatibility with every existing implementation of RIPv2 [12]. The enhanced knowledge is based on information already provided by the RIP protocol. Therefore, a new RMTI router can be deployed at selected nodes in an existing RIPv2-domain.

III. Test Environment

Various tests with different topologies have been carried out in order to investigate the convergence properties of different routing protocols. For an automatic execution of these test runs a script software was written to perform the following operations which

- starts many different VNUML network scenarios in succession
- uses a test script of certain network failure situations (e.g., link corruption, router collapse)
- controls each VNUML scenario during runtime
- causes runtime failures in a predicted manner
- collects data needed for the offline evaluation
- evaluates the captured data during an offline phase with respect to convergence time and network traffic.

The measurements are carried out by the analysis of automatically generated tcpdump files [21] after the test run. Specific network data are logged and analyzed during each test run. This approach prevents measurement pertubations as only specific data are captured during runtime.

The separation of the test script file and the topology discription file (see Figure 1) allows for flexible design of test runs. The test script files define the topologies and the frequency of test runs. Each test run is divided in an *online* and *offline* phase (see Figure 1) in order to distort the runs as less as possible by the data capturing. This test environment is needed to provide measurement results that are as close to reality as possible in order to analyze convergence properties of routing algorithms.

The tested routing protocol implementations are part of the widely used Quagga Routing Software Suite [17]. The RMTI algorithm is implemented as a re-engineered version of the Quagga RIPv2 router. The test environment can carry out up to 1000 and more test runs where each run can be fully controlled and analyzed.

During a test run all characteristic actions and changes are saved with a corresponding timestamp for later synchronization. When the online test phase is finished the captured data are evaluated by a statistic analysis in the offline phase.

The test environment is developed for evaluating complex and comprehensive test runs without any human interaction. The environment can be configurated for many different test runs which cover various network failure situations. Each step of a test run can be pre-determined precisely by the test script. Many test cases including random or previously defined link and router failures can be performed.

To be able to test various failure scenarios that could occur in networks, the test environment allows for the opportunity to cause packet losses or transmission delays for single or multiple devices of one or more routers. In order to expose the distance vector routing algorithms to Counting-to-Infinity situations, the test environment can cause these situations by retaining certain update messages on certain router ports.

The high degree of automation of the test environment enables to gather a great deal of result data that can be statistically analyzed. Together with the network environment for the test cases, very detailed insight into the behavior of common routing algorithms can be obtained. The results of the test runs can be summarized and then visualized as diagrams. With this approach, the relationship of the various test parameters as well as a performance comparison between the different routing algorithms can be shown. The evaluation of these results enables the user to depict these relationships as mathematical functions. These functions are approximated using regression analysis. Additionally, the representation of frequency distributions such as in Figure 13 is supported. The test environment can average the results before analyzing them with statistical methods. It can create a network graph of the network topology even if the topology is randomly generated.

Before the test starts, topology properties of the network like diameter, articulations, leaves and number of nodes, edges and circles [8] are analyzed offline and saved in the topology description (see Figure 1). During the runtime of the test, all characteristic events are saved as routing update packets with a timestamp for the offline analysis after the run. After the runtime the convergence properties, such as convergence time or traffic amount, are calculated with the saved update traffic and information about the topology tested.

As the test environment is modularly designed it can be used for the comparison and testing of other intra domain routing algorithms as well. For this purpose, only the specifications for the offline analysis module have to be modified. This can be done by identifying the update packet which leads to the convergence of the network.

A. Detailed description of test and measurement procedure

To measure the convergence time of a certain network, the network has to be changed from a convergent state to an inconvergent state by changing the status of certain links or routers in order to cause a link or router failure.

In order to calculate convergence properties of the network after a test run, the protocol analyzer *tcpdump* [21] is used to capture the existing network traffic. To get proper results the routing daemons in the routers have to reach a convergent state before the status of certain links or routers is changed to cause a failure. Then the sequence of state changes is recorded starting with the failure state and ending with the next convergent state. When the network has reached its convergent state, the run is automatically stopped and the next run follows.

After the online capturing of the router states is completed, the offline evaluation of the states can be executed. To evaluate the measurement, the captured update traffic has to be treated in the same way as the routing algorithms do in order to calculate the forwarding table entries. That means that the offline evaluation module repeats the processing steps of the routing algorithm.

The timestamped update packets of all subnets are sorted chronologically for evaluation. To identify the starting point of the convergence time t_{conv}, the first packet with the information about an unreachable subnet is selected and its timestamp saved for later use (as first timestamp t_f). As well, the timestamp of the first update packet can be taken as first timestamp to analyze the coldstart[1] behaviors of a routing algorithm.

In order to find the timestamp of the update packet leading to convergence (the last timestamp t_l), all update packets are re-processed from the start.

Whenever a routing table is updated, the timestamp of the appropriate update packet is saved. As the network must be in a convergent state when no routing table is updated any more, the last timestamp t_l must belong to the update packet that led to this state. This timestamp is chosen for convergence time calculation.

The convergence time t_{conv} is now defined as:

$$t_{conv} = t_l - t_f \text{ with:} \tag{1}$$

t_{conv} = convergence time,
t_l = last timestamp,
t_f = first timestamp

Adding up the packet sizes of the whole traffic sent between the first timestamp t_f and the last timestamp t_l shows the traffic volume produced by routing updates during the convergence procedure.

[1] all routers in the considered network were rebooted and start with empty routing tables

IV. IP Networks and Routing Protocols

Within computer networks, routers are special nodes which provide the whole network with information about the location of subnets. The IP protocol uses packet forwarding to deliver data packets from the source to the destination. Packet forwarding means that a node only knows the adjacent node, which is closer to the destination node. Therefore, every router has to know which adjacent router is closer to the subnet of the destination node. A router maintains a forwarding table listing the subnets in connection with the corresponding next hop router and the metric which represents a distance to the destination node. Routing is the process of completing and maintaining an accurate forwarding table. It is a distributed network-wide process which offers some properties such as dynamic adaptation to network changes, scalability and system stability. At any time, a forwarding table of a router must correctly describe the location of subnets and how to get there in order to prevent misguided data packets. There is always a small time gap beginning with a network failure and ending with the detection of the failure by the routers. Keeping the network free of invalid routing information during this time is a challenge every routing protocol has to cope with. It may occur that routing updates between routers become invalid on their way through the network and forwarding loops can take place. In practice, all common routing protocols are affected by forwarding loops [24]. A forwarding loop corresponds to a loop in the forwarding process of data packets. Thus, a routing loop corresponds to a loop in the routing process caused by circulating routing update packets. A routing loop is in close connection with a forwarding loop because the routing process has a direct impact on the forwarding process.

A. Relevant Items of the RIP Protocol

The Routing Information Protocol (RIP) is considered as the classical representative of the distance vector protocol family. It employs the hop count as a metric to a destination node in the network (see Figure 2).

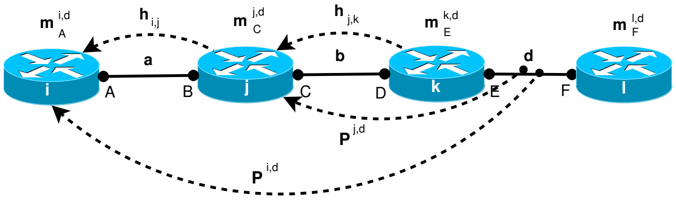

Fig. 2. Network model, $h_{i,j}$ = hop from router i to router j, $P_{i,d}$ = path from router i to subnet d with metric $m_{i,d}$.

RIP works like this: If a RIP router i receives a routing update from an adjacent RIP router j to a subnet d with the metric $m^{j,d}$ via interface A of router i, this indicates the existence of a corresponding path $P^{j,d}$ from RIP router j to subnet d. Router i does not know the complete path $P^{j,d}$, but knows the number of hops of this path. Then, from the view of router i, there will be a path $P_A^{i,d}$ with metric $m_A^{i,d} = m^{j,d} + 1$ by appending a hop $h_{i,j}$ from i to j to the path $P^{j,d}$. Therefore, router i knows the metric $m_A^{i,d}$ and the next hop towards subnet d (see Figure 2). If a RIP router has a valid path to subnet d, it will reject all equivalent or inferior paths to the same subnet. RIP prevents the proliferation of routing loops by implementing a limit on the number of hops allowed in a path from the source to a destination node. The maximum number of hops allowed is 15. This hop limit also limits the size of the diameter of a RIP network. A hop count of 16 is called infinity. It is considered an infinite distance and used to deprecate inaccessible routes. RIP mainly detects suddenly inaccessible routes by the expiration of timers due to missing confirmations. RIP uses three timer intervalls:

- an update timer (default 30 sec) for periodically sending out routing updates,
- a timeout timer (default 180 sec) to detect invalid entries in the routing table and
- a garbage collection timer (default 120 sec) to delete invalid entries from the routing table.

The timeout timer of an entry is re-initialized when the entry is confirmed by an incoming routing update. Should the timeout timer of the entry expire, the corresponding route is marked as unreachable with metric infinity (16), and the garbage collection timer is started. The garbage collection timer is stopped when the entry is confirmed by a valid routing update. If the garbage collection timer expires, the entry will be deleted from the routing table. RIP periodically sends routing updates every 30 seconds containing the whole routing table to all adjacent routers. As it is not useful to claim reachability for a subnet to the neighbor from which the route was learned, the *split horizon scheme* is defined in the RIP specification [12]. Split horizon omits routes learned from one neighbor in updates sent to that neighbor in order to avoid loops between two neighbor routers on one link. Another enhancement, Split horizon with poisoned reverse, includes such routes in updates, but sets their metrics to infinity. Routing updates are also sent with every change in the routing table only containing the affected entries. These event-driven routing updates are called triggered updates. RIP uses two kinds of message types, RIP updates and RIP requests which have the same message structure (see Figure 3) but differ in their processing. RIP updates are used to announce routing information to the adjacent routers. RIP requests are used to ask adjacent routers for routing information. The messages can be recognized by the value in the command field of the message structure. As RIP only uses one message structure type, it is very easy to implement and to deploy RIP in contrast to complex routing protocols like Open Shortest Path First (OSPF)[14]. The simple deployment of RIP is one reason why it is still widely used.

B. RIP Request Messages

A RIP request is used to ask for a response containing all or part of a router's routing table. Requests and their treatments are part of the standard RIP specification [12]. Usually, RIP requests are sent as multicast messages to neighbors by routers which have just booted and try to fill in their routing tables.

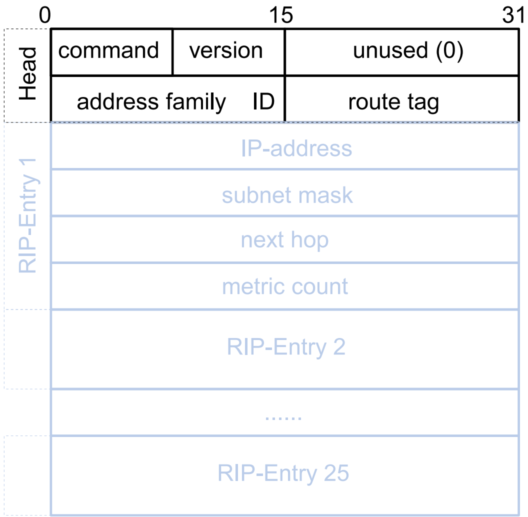

Fig. 3. The RIP message structure of the update and the request message as specified in [12].

Therefore, the most common request messages causes all neighbor routers to transmit all route entries in their routing table. Also single entries can be requested. An option which is mainly used for monitoring, e.g., by network monitoring tools.

The RIP request message is processed entry by entry (see Figure 3) and can have 25 entries at its maximum before a second message is needed. If there are no entries in a RIP request message, the requested router does not response. If there is exactly one entry in the request, and it has an address family identifier of zero and a metric of infinity (IP-Address 0.0.0.0 and metric 16), then this is a request to send the entire routing table. Except for this special case, the processing of the RIP request is quite simple. For each entry, the destination is looked up in the router's routing database and, if the route is listed, the route's metric is put in the metric field of the request message (see Figure 3). If there is no explicit route to the specified destination, the infinity value is put in the metric field. Once all the entries are filled in, the value in the command field is changed from request to response and the datagram is sent back to the requestor.

V. The RMTI Protocol

The RMTI router evaluates redundant routing updates from different directions to the same destination. The RMTI protocol is implemented on top of the Routing Information Protocol (RIP). Thus, our RMTI-protocol is downward-compatible to RIP. As the RMTI extension does not change the message structure but only the processing algorithm, the RMTI technique can also enhance other distance vector routing protocols as well.

A. *RMTI optimizations*

In distance vector routing the Counting-to-Infinity problem (CTI) reflects the routing loop problem. A group of RIP-Routers can get into an unstable state as soon as a CTI occurs. This problem cannot be solved by the well-known *split-horizon scheme*, because this approach can prevent CTIs in virtual loops but not in real loops. A real loop can be found in networks where at least three routers are connected to each other via separate links. Therefore the RIP protocol is restricted to a small network diameter in order to limit the impact of the CTI update sequence. This impact has two aspects:

1. wrong routing updates are proliferated and
2. payload packets are misguided.

The RMTI protocol avoids CTIs as well as routing loops and forwarding loops. If all routers in a network are RMTI-routers the infinity metric is no longer necessary. The RMTI protocol and its enhancements to RIP are described in [2][3][19]. The RMTI algorithm is an optimization that affects convergence time and update traffic. The update traffic is reduced by sending small neighbor-alive-notifications instead of periodic routing updates. An incoming neighbor-alive-notification confirms all routes received from this neighbor. Changes in the network will be advertised by event-triggered updates which are already part of the standard RIPv2 specification [12]. The neighbor-alive-notification is small in contrast to the ordinary periodic routing update whose size depends on the amount of subnets in the network. Moreover, due to the smaller neighbor-alive-notifications either

- the entire routing update traffic can be reduced or
- the sending interval and the convergence time of the network can be reduced.

There are two options to optimize the RMTI protocol either to have less update traffic or to reduce the convergence time.

B. *RMTI with neighbor-alive-notifications*

The periodic routing updates are replaced by small neighbor-alive-notifications which accelerates the convergence time and the reliability of RMTI. In order to keep RMTI downward compatible to RIP, the neighbor-alive-notification is performed by empty RIP Request messages.

In correspondence to the RIP specification, a RIP router does not have to reply to a Request with no entries. An RMTI router can identify its neighbor as RMTI router, if it receives empty requests. The neighbor-alive-notification is very small and, is periodically sent via multicast to all neighbor RMTI routers. There is a time and a traffic optimization of the RMTI. The time optimized RMTI sends the neighbor-alive-notification every five seconds in order to detect link failures immediately. This optimization speeds up the convergence time. The traffic optimized RMTI's sending period is larger than 5 seconds in order to reduce the update traffic between the routers. Should an RMTI router detect an empty request from a neighbor router, it accepts this router as an RMTI router and does the following:

- Firstly, the timeout timers of all route entries in the routing table from this neighbor are reduced to ,e.g., 10 seconds. So, if the entries are not confirmed within 10 seconds they will be marked as unreachable.

- Secondly, every time a neighbor-alive-notification is received from the neighbor router, the corresponding route entries in the routing table will be confirmed at once and their timeout timer will be re-initialized.

The burden of periodical routing updates which contain the whole routing table could be omitted. The tedious routing update which transmits the whole routing table every 30 seconds is replaced by a small neighbor-alive-notification sent every 5 seconds. If a route is not confirmed by a neighbor in RIP for 180 seconds, it will be marked as unreachable. If a neighbor is not confirmed in RMTI within 10 seconds, all route entries from this neighbor will be marked as unreachable. Individual entry changes in the routing table are transmitted by the event-driven triggered updates further on.

The neighbor-alive-notification allows for *speeding up* the convergence time of the network in case of a link failure due to lower timeout and short-intervalled alive notifications. Since the triggered updates become most significant for the RMTI routing process, the sender needs an acknowledgment of the triggered updates. Our acknowledgment approach is based on *split horizon with poison reverse*, which is part of the RIPv2 specification [12]. All routes received by a triggered update are immediately acknowledged with metric infinity.

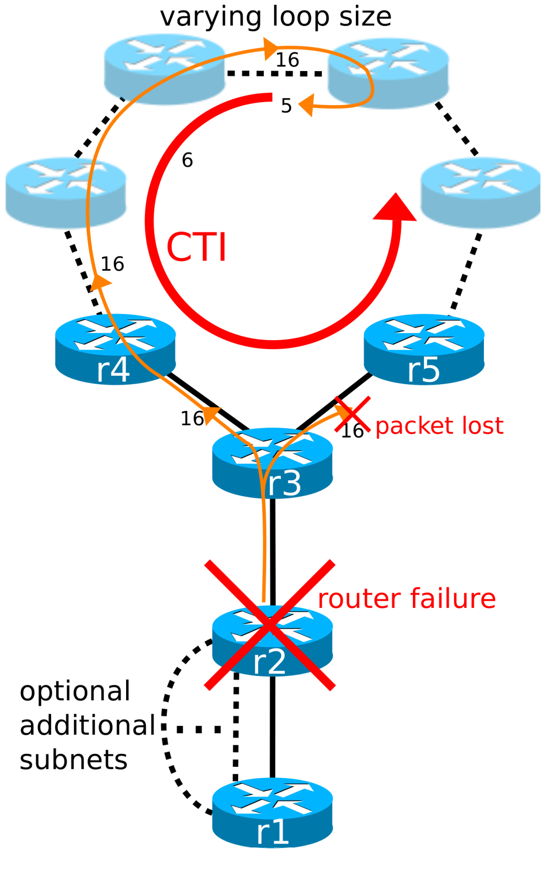

Fig. 5. Y-Topology class

VI. Convergence Measurement

In this section, the tested topologies, the measurement methods, and the results of the convergence properties of the RIP and RMTI algorithms are presented.

A. Tested Topologies

A simple network topology in which the CTI problem can occur is shown in Figure 5. This topology consists of a topology loop and a row of routers $r1$, $r2$ and $r3$. Router $r3$ is the node connecting the topology loop with the row. The respective subnets are peripherally located outside the topology loop connected to router $r1$ and $r2$. Routing information about these subnets can flow into a CTI sequence within the topology loop after a link failure between router $r2$ and $r3$.

As depicted in Figure 5, a CTI can occur if the information of the unreachable subnet will be lost or just simply delayed in one path of the topology loop. If the link between router $r2$ and $r3$ fails, the subnets located between $r2$ and $r1$ are unreachable for $r3$. Router $r3$ does not receive any more updates and the timeout for these subnets expires. The routes in $r3$'s forwarding table are marked as unreachable with metric infinity. Then, router $r3$ sends a triggered update with metric infinity to router $r4$ and $r5$ and advertises this new routing information. Router $r4$ forwards the information about the unreachable subnets to the next router. In Figure 5 router $r5$ does not receive the routing information from $r3$ due to a transmission failure and the routers behind $r5$ in the topology loop do not receive the information about the unreachable subnets. At this point, the old invalid routing information prevails over the current routing one. The routers which have lost their route to the subnets accept the old routing information as new alternative routes. The well-known *split horizon scheme* cannot avoid the CTI situation. *Split horizon* is an extension to RIP which prevents a router from advertising routing information back to the router from which it was received. Router $r4$ gets the invalid routing information indirectly from router $r5$ and sends the information to $r3$. Router $r3$ accepts the invalid routing information from router $r4$ as new and valid alternative route to the subnets via $r4$. Then the CTI sequence starts. As long as the CTI occurs, a malicious routing loop is established.

For the test runs, several topologies were used which allow conclusions about expected correlations of a particular topology property and convergence properties. These topologies are presented in Figure 4. For the evaluation, over 300 topologies with a total of over 5000 test runs are analyzed. Due to the high computational complexity, the test topologies have been limited to a maximum size of 50 routers and 100 subnets. In order to obtain meaningful results it is important that any correlations between individual topologies are considered.

The following topologies are used in the test environment in order to analyse routing protocols (see Figure 4 and 5):

- the y-topology,
- the crown topology,
- the square topology,
- the Arpanet, described in [27], and
- the Internet 2 topology, used in [28].

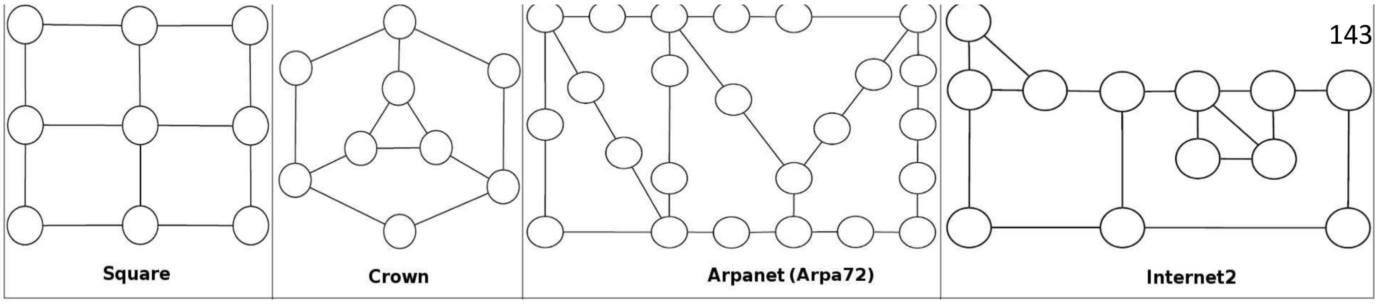

Fig. 4. The tested topologies: square, crown, arpanet, and internet2

B. General Convergence Properties

In order to compare different routing algorithms, it is crucial to know how they behave in general. Coldstart tests are performed to measure convergence time and update traffic volume from the point in time where all router daemons are started until the network reaches the convergent state for the first time. For this test case, no CTI is provoked as the CTI situation is a special case. This test scenario was designed to test the propagation speed of the routing updates being the worst case scenario for convergence time and traffic.

These measurements covered over 300 (mostly randomly generated) topologies showing that the convergence time for both algorithms mainly depends on the network topologies diameter d. As shown in Figure 6 the convergence time increases in this test environment by a factor of 2.5 along with an increasing diameter d. However, the update traffic volume of both algorithms increases with polynomial growth with the number of subnets in the network (see Figure 7).

Test cases show that RMTI and RIP reache their convergent state at the same time (see Figure 6). If the time-optimized RMTI (Section V-A) is used, it does not affect the convergence time because the time benefits correspond to the detection of a topology change. With traffic improvement the RMTI needs 10 percent less traffic than RIP[2] (see Figure 7). The traffic volume for the time-optimized RMTI is the same as for RIP.

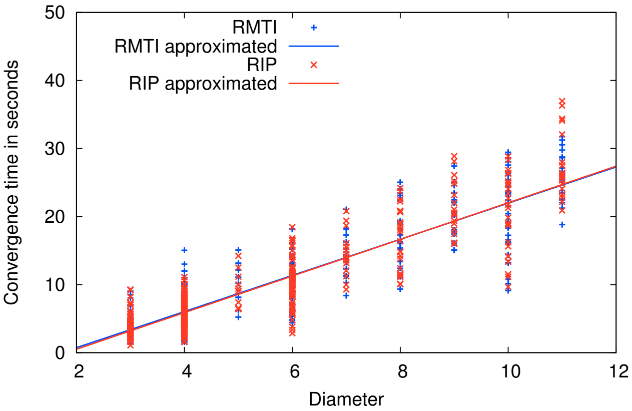

Fig. 6. Convergence time for RIP and RMTI at coldstart

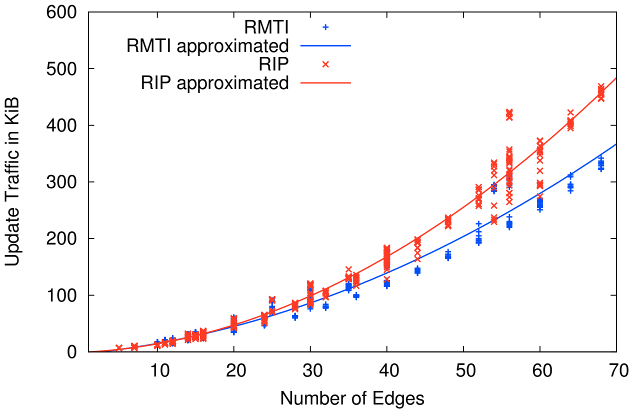

Fig. 7. Traffic comparison of optimized RMTI and RIP at coldstart

The analysis of the measurements shows that an even better equation for the approximation of convergence time t_{approx} is achievable. Besides the network diameter d, the number of nodes n in the network topology require to add the product $0.16 \cdot n$ to the convergence time. The variance is halved from 7.28 to 3.56. Now the approximated convergence time t_{approx} from coldstart can be calculated for every network using the following Equation (2):[3]

$$t_{approx} = 2.5 \cdot d + 0.16 \cdot n - 5 \pm 1.89 \text{ [seconds] with:} \quad (2)$$

t_{approx} = convergence time,
$d > 1$ = diameter of network topology ,
n = Number of nodes in the network topology

In Figure 8 and 9 the dependencies between the number of edges (subnets) are shown for the square and the crown topology.

In Figure 10 and 11 the dependencies between the network diameter and the convergent time in seconds are shown for the square and the crown topology.

Fig 12 depicts the amount of traffic as function of the number of verticies (router nodes) and the number of edges (subnets) within a topology in relation to the traffic.

[2] To achieve this result, the first 60 seconds from coldstart are measured instead of stopping the measurement at the convergent state.

[3] The values may differ a bit depending on the test environment

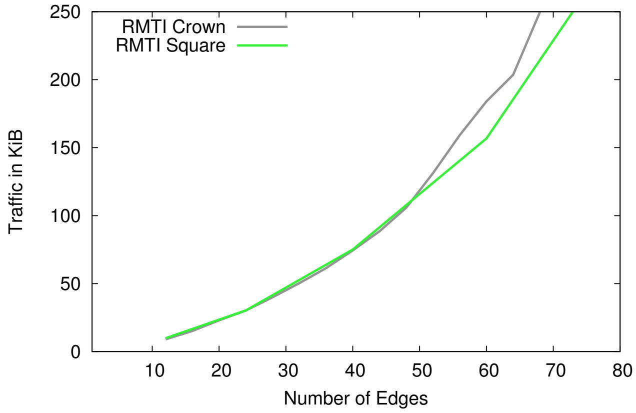

Fig. 8. Traffic comparison between the topologies square and crown with RMTI at coldstart.

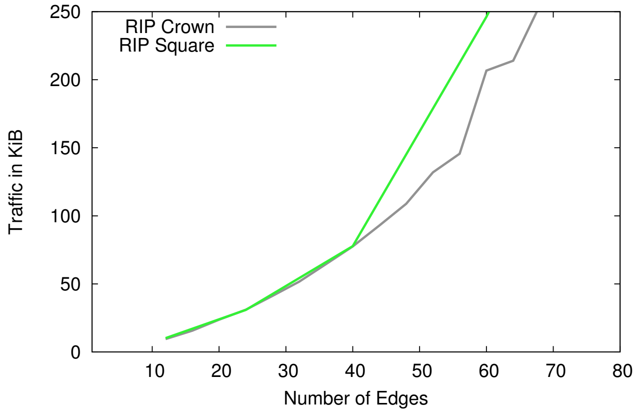

Fig. 9. Traffic comparison between the topologies square and crown with RIP at coldstart.

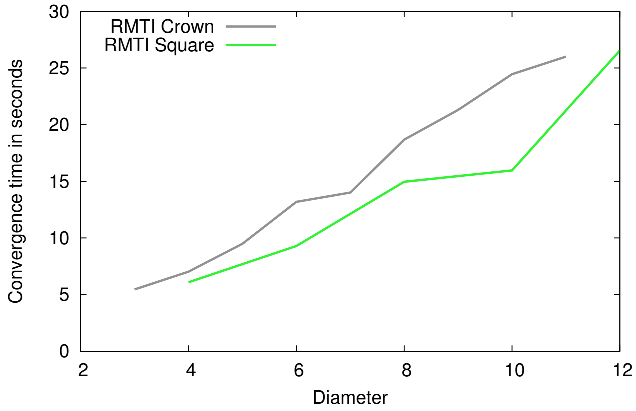

Fig. 10. Convergence time comparison between the topologies square and crwon with RMTI at coldstart.

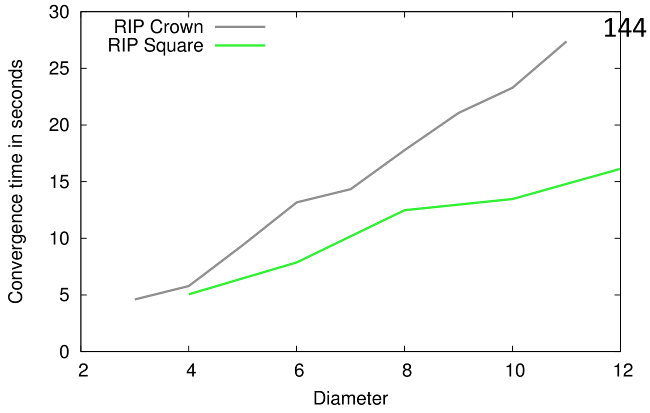

Fig. 11. Convergence time comparison between the topologies square and crwon with RIP at coldstart.

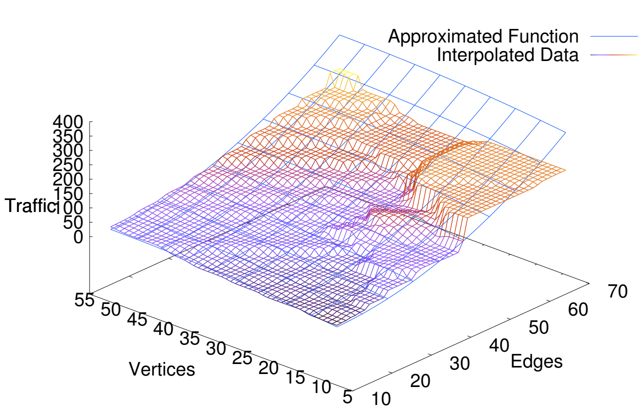

Fig. 12. Traffic comparison in relation to the growing number of edges (subnets) and verticies (router nodes).

C. Importance of CTI avoidance

Despite the *Split Horizon* algorithm, CTIs can occur under certain circumstances when the network contains loops and links are flapping or routing updates are lost. To show the importance of CTI avoidance for distance vector algorithms, some test runs have been performed that show the CTI frequency for links with different packet losses.

As CTIs can occur in Y-Topologies (Section 4) they have been tested with a loop size of 3 where a failure was provoked on one subnet. If the failure update of this subnet is announced to either upper router $r4$ or $r5$ (Figure 5), a CTI occurs. The bigger the loop size, the more probable is the occurrence of a CTI. This test case is a best case scenario because there are more subnets to pass and more possibilities to lose the information about the failed subnet in bigger loops.

As the results illustrate in Figure 13, the CTI frequency F_{CTI} increases with the routing update packet loss L_p (in percent) in a linear way.

The general rule is:

$$F_{CTI} = 0.55 \cdot L_p \pm 0.62 \text{ [\% (absolute)] with:} \qquad (3)$$

F_{CTI} = CTI frequency in percent,
L_p = packet loss in percent

With an update packet loss L_p of 0.01%, a CTI is provoked with a probability of 0.0055% or averaged at every 180st

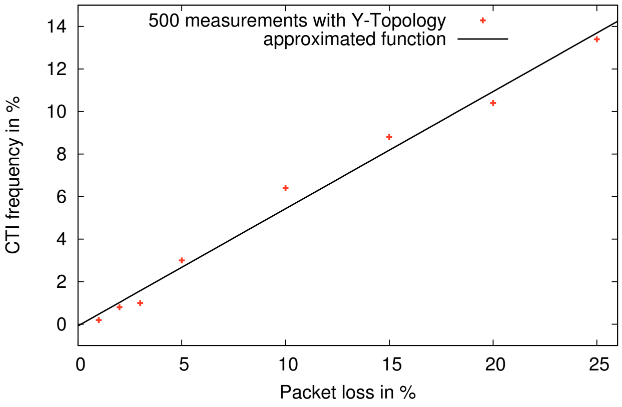

Fig. 13. CTI frequency depending on Packet loss

failing subnet.

D. CTI Duration

The impact of CTI situations become apparent by examining their duration. As long as the CTI situation lasts, all payload traffic for the corrupted subnet is cycling in the loop. This is a burden for all routers and subnets located in the loop.

Since RMTI can detect routing loops and avoid CTI occurrences it eliminates this burden. To quantify these benefits the convergence time of RIP and RMTI have been measured in case of a CTI situation. As CTIs only occur in loops several Y-Topologies with different loop sizes (Figure 5) have been tested.

To provoke a CTI router, $r2$ (Figure 5) is shut down. This leads to an unreachability of a certain subnet. Router $r3$ is forced to propagate the new infinity metric of the unreachable subnet only to one of its neighbors ($r4$ or $r5$) so that this kind of update transfer must result in a CTI.

For each topology tested the time was measured starting when a router detected a failing subnet and ending when the network was convergent again. Hence, these results are independent of the RMTI optimization (time or traffic - see Section V-A), because time optimization only speeds up the topology change detection.

The result graph (Figure 14) shows that the benefit of RMTI is between 30 and 120 seconds. The average CTI duration with increasing loop size increases with a factor of 18.6 for RMTI. That is nearly one fourth of RIP's growth factor which is 74. The average CTI duration of all tested topologies with RMTI only needs 30% of the average RIP CTI duration. The irregularities (see Figure 14) of the RIP results are generated when the update timer of one router expires and affects the CTI in reaching infinity (metric 16). This only happens for certain loopsizes together with specific timer settings. For these tests the standard RIP update timer of 30 seconds was used. With other timer settings this behavior occurs with other loop sizes. RMTI can converge so much faster because RMTI stops the CTI at router $r3$ (Figure 5).

Starting the measurement with the topology change, RIP converges slower because of its long timeout timer of 180 seconds.

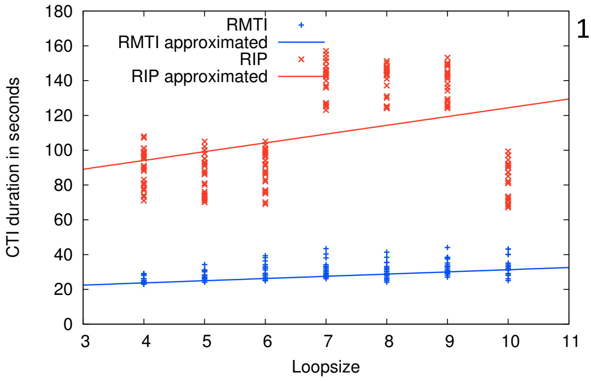

Fig. 14. CTI duration

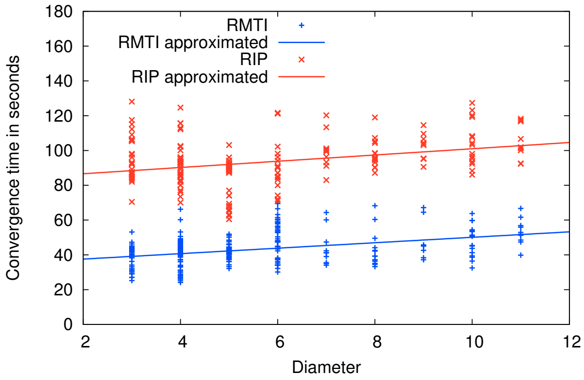

Fig. 15. Convergence time for random router failures

E. Common Convergence Behavior

As the most common topology changes do not provoke CTIs (Figure 13), the convergence properties were also measured in case a router fails without CTI provocation. Therefore, a random router has been shut down in each test run. As shown in Figure 15 the RMTI algorithm with time optimization (Section V-A) converges 50 seconds faster on average than the normal RIP from the time point where the topology change happens. With a traffic optimized RMTI the convergence time is the same as with RIP, but there is a noticeble update traffic reduction.

VII. Conclusion and Future Work

To demonstrate the advantages of new routing algorithms in contrast to common intra domain routing algorithms, comprehensive tests have to be carried out. In practice, a variety of network topologies and a succession of critical events have to be automatically generated by random generators. Test situations concerning the capability of a routing algorithm to re-organize itself after a topology change have to be covered.

With the newly developed test environment, one can show the benefits of the new RMTI algorithm in comparison to RIP. The flexible design of the test environment allows us to test RIP, RMTI and other intra domain routing protocols. The RMTI algorithm shows two important advantages: the possibility to avoid CTI situations and to converge much faster than other distance vector algorithms in case of a topology change. The ability of RMTI to choose whether to optimize a fast topology change detection or a traffic reduction (or a mixture of both) makes the test environment adaptable to the specific needs of many different networks. With the VNUML script, it is also possible to run larger networks on computer clusters which will support the results for those networks. The high degree of automating the test environment and the ability to gather large amounts of measurement data is a big advantage for analyzing all kinds of routing protocols. Another issue that we will deal with in our future research is the performance analysis of the RMTI algorithm with payload traffic and the comparison with other intra domain routing protocols.

We have shown that we can obtain a variety of characteristic routing parameters and various relationships between them.

References

[1] F. Bohdanowicz, M. Jakobs, and Ch. Steigner, *Statistical Convergence Analysis of Routing Protocols*, in Proceedings of the Nineth International Conference on Networking (ICN 2010), Les Menuires, France, 2010.

[2] F. Bohdanowicz, H. Dickel, and Ch. Steigner, *Routing with metric-based Topology Investigation*, International Journal on Advances in Internet Technology, 2009, vol 2 no 1.

[3] F. Bohdanowicz, H. Dickel, and Ch. Steigner, *Metric-based Topology Investigation*, in Proceedings of the Eighth International Conference on Networking (ICN 2009), Cancun, Mexico, 2009.

[4] Ch. Steigner, H. Dickel, and T. Keupen, *RIP-MTI: A New Way to Cope with Routing Loops*, in Proceedings of the Seventh International Conference on Networking (ICN 2008), Cancun, Mexico, 2008.

[5] C. Cheng, R. Riley, S.P.R. Kumar, and J. J. Garcia-Luna-Aceves, *A loop-free extended bellman-ford routing protocol without bouncing effect*, ACM Sigc. Symp. Commun. Arch. and Prot., pp. 224-236, 1989.

[6] J. Dike, *User Mode Linux*, Prentice Hall, 2006.

[7] The user-mode-linux project page, URL: http://user-mode-linux.sourceforge.net, 18.01.2011.

[8] C. Berge, *Graphs and Hypergraphs*, North-Holland Publishing Company, 1973.

[9] B. Rajagopalan and M. Faiman, *A new responsive distributed shortest-path routing algorithm*, ACM Sigcomm Symposium Commun. Arch. and Protocols, pp. 237-246, 1989.

[10] J.J. Garcia-Luna-Aceves, *Loop Free Routing Using Diffusing Computations*, IEEE Transactions on Networking, 1993.

[11] K. Levchenko, G. M. Voelker, R. Paturi, and S. Savage, *XL: An Efficient Network Routing Algorithm*, Proc. Sigcomm 2008, August, 2008.

[12] G. Malkin, *RIP Version 2*, RFC 2453, 1998.

[13] V. Fuller, T. Li, *Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan*, RFC 4632, 2006.

[14] J. Moy, *OSPF Version 2*, RFC 2328, 1998.

[15] C. E. Perkins, *Ad hoc networking*, Addison-Wesley, Amsterdam 2001.

[16] C. E. Perkins, E. Belding-Royer, S. Das, *Ad hoc On-Demand Distance Vector (AODV) Routing*, RFC 3561, 2003.

[17] The quagga project page, URL: http://www.quagga.net, 18.01.2011.

[18] Y. Rekhter, T. Li, S. Hares, *A Border Gateway Protocol 4*, RFC 4271, 2006.

[19] A. Schmid and Ch. Steigner, *Avoiding Counting to Infinity in Distance Vector Routing*, Telecommunication Systems 19 (3-4): 497-514, March - April, 2002, Kluwer Academic Publishers.

[20] VNUML project home page, URL: http://www.dit.upm.es/vnuml, Technical University of Madrid (UPM), 18.01.2011.

[21] tcpdump project home page, http://www.tcpdump.org, 18.01.2011.

[22] OpenWRT Project home page, http://www.openwrt.org, 18.01.2011.

[23] Andrew S. Tanenbaum, *Computer Networks*, 3rd ed., Prentice Hall PTR, 1996, pp. 358-359

[24] Z. Zhong, R. Keralapura, S. Nelakuditi, Y. Yu, J. Wang, C. Chuah and S. Lee, *Avoiding Transient Loops Through Interface-Specific Forwarding*, Transactions on Networking, IEEE/ACM, Volume: 15, Issue: 6, Dec. 2007.

[25] F. Galan, D. Fernandez, W. Fuertes, M. Gmez and J. Lpez de Vergara, *Scenario-based Virtual Network Infrastructure Management in Research and Educational Testbeds with VNUML: Application Cases and Current Challenges*, Annals of Telecommunications, special issue on Virtualization: a path for the future Internet, vol. 64(5), pp. 305-323, May 2009.

[26] B. Premore, *An Experimental Analysis of BGP Convergence Time*, ICNP '01: Proceedings of the Ninth International Conference on Network Protocols, IEEE Computer Society, 2001.

[27] Frank, Howard, Robert E. Kahn und Leonard Kleinrock, *Computer communication network design: Experience with theory and practice*, in Proceedings of the spring joint computer conference (AFIPS '72 Spring), May 1972.

[28] Gunes, Mehmet H. und Kamil Sarac, *Inferring Subnets in Router-level Topology Collection Studies*, in Proceedings of the 7th ACM SIGCOMM conference on Internet measurement (IMC '07), Oktober 2007.

[29] *Internet 2 Fact Sheet*, URL: http://www.internet2.edu/resources/AboutInternet2.pdf,18.01.2011.