University of Warsaw
Faculty of Mathematics, Computer Science and Mechanics

VU University Amsterdam
Faculty of Sciences

Joint Master of Science Programme

Maciej Wojciechowski

Student no. 209510 (UW), 1735691 (VU)

Border Gateway Protocol Modeling and Simulation

Master's thesis
in COMPUTER SCIENCE

Supervisors:

Benno Overeinder
NLnetLabs, Amsterdam

Guillaume Pierre,
Maarten van Steen
Dept. of Computer Science,
VU University Amsterdam

Janina Mincer-Daszkiewicz
Institute of Informatics,
University of Warsaw

July 2008

Oświadczenie kierującego pracą

Potwierdzam, że niniejsza praca została przygotowana pod moim kierunkiem i kwalifikuje się do przedstawienia jej w postępowaniu o nadanie tytułu zawodowego.

Data Podpis kierującego pracą

Oświadczenie autora (autorów) pracy

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam również, że niniejsza praca jest przedstawiona w ramach wspólnego programu magisterskiego Uniwersytetu Warszawskiego i Vrije Universiteit w Amsterdamie. Praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem tytułu zawodowego w wyższej uczelni.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Data Podpis autora (autorów) pracy

Abstract

Border Gateway Protocol (BGP) is de facto the only inter-domain routing protocol of the Internet. Although the protocol is reasonably simple, due to the size of its current deployment many anomalies can be observed. Such anomalies are for example very long convergence time or path haunting. Some techniques (e.g., like route flap damping) have been applied to tackle the observed problems but they have side-effects that were not anticipated before. There is still a strong need for better BGP protocol understanding.

In this thesis we present a new, ambitious approach to BPG simulation. Instead of focusing on intra-domain communication, network and protocol are highly abstracted in order to allow for large-scale simulation. We describe our model of the BGP protocol along with its implementation. The implementation is validated in order to show to what extent our model resembles the real-world. Many tracks of future research are shown as well as many possible uses of this kind of approach to BGP simulation.

Keywords

BGP, inter-domain routing, Internet, modeling, simulation, emulation, Autonomous System

Thesis domain (Socrates-Erasmus subject area codes)

11.3 Informatics, Computer Science

Subject classification

C.2.2 [Network Protocols]: Routing
C.2.4 [Distributed Systems]: Distributed applications
I.6.3 [Simulation and Modeling]: Applications
I.6.8 [Types of Simulation]: Distributed

Contents

1. Introduction . . . 5
- 1.1. Background . . . 5
- 1.2. BGP . . . 5
  - 1.2.1. Prefix Routing . . . 5
  - 1.2.2. Autonomous System . . . 6
  - 1.2.3. Basic Operations . . . 6
  - 1.2.4. Routing policies . . . 7
- 1.3. BGP Instability . . . 7
  - 1.3.1. Project Motivation . . . 8
- 1.4. Outline . . . 9

2. Current state of BGP research . . . 11
- 2.1. BGP Monitoring . . . 11
- 2.2. BGP Simulation . . . 11
  - 2.2.1. Simulation Software . . . 11
  - 2.2.2. Conclusions . . . 13

3. The Model of the BGP Protocol . . . 15
- 3.1. Current State of the Internet . . . 15
- 3.2. Network Modeling . . . 16
  - 3.2.1. Network Links . . . 16
  - 3.2.2. Autonomous Systems . . . 16
- 3.3. Routing . . . 17
  - 3.3.1. Basic Operation . . . 17
  - 3.3.2. Protocol Abstraction . . . 17
  - 3.3.3. BGP modeling . . . 19
  - 3.3.4. Summary . . . 21

4. Design and Implementation . . . 23
- 4.1. Design Requirements . . . 23
- 4.2. Application of the Model . . . 24
- 4.3. Simulation framework . . . 24
- 4.4. Hardware Environment . . . 25
- 4.5. Programming Environment . . . 25
- 4.6. Execution . . . 26
  - 4.6.1. Communication . . . 26
  - 4.6.2. Coordinator . . . 26
  - 4.6.3. Compute Node . . . 27
  - 4.6.4. AS Implementation . . . 27
  - 4.6.5. Execution Flow . . . 28
- 4.7. Summary . . . 29

5. Validation Description . . . 31
5.1. Model Validation . . . 31
5.2. Validation Techniques . . . 31
5.3. Experiment Framework . . . 32
5.4. Validation Approach . . . 32
5.5. Experiment Description . . . 33
5.5.1. BGP Beacon . . . 33
5.5.2. Route Views . . . 33
5.5.3. Signal Duration and Relative Convergence Time . . . 34
5.6. Experiment Setup . . . 35
5.6.1. Runtime Environment . . . 35
5.6.2. Simulation Parameters . . . 35
5.6.3. External factors . . . 37
5.6.4. Time Measurement . . . 37
5.6.5. Single Experiment Description . . . 38
5.6.6. Experiment Instance Description . . . 38
5.7. Summary . . . 40

6. Results Discussion . . . 41
6.1. Experimental Setup . . . 41
6.1.1. Graphical Data Representation . . . 41
6.1.2. Beacons . . . 41
6.2. Obtained Results . . . 42
6.2.1. Simulated Behavior . . . 42
6.2.2. Parameters Sensitivity . . . 43
6.2.3. Parameters Used . . . 44
6.3. Results Comparison . . . 44
6.3.1. "BGP Beacons" Experiment Discrepancy . . . 45
6.3.2. Similar and Different Characteristics . . . 45
6.3.3. Withdrawals Propagation . . . 46
6.4. What-if Analysis . . . 47

7. Conclusions . . . 49
7.1. Impact of This Project . . . 49
7.1.1. Proof of Concept . . . 49
7.1.2. Real-world Resemblance . . . 49
7.2. Future Research . . . 50
7.2.1. Validation and Calibration . . . 50
7.2.2. Model Properties Study . . . 50
7.2.3. Possible Usage . . . 50

Bibliografia . . . 53

Chapter 1

Introduction

1.1. Background

One of the many remarkable qualities of the Internet is that it has scaled so well to its current size [33]. The Internet evolved from the inter-connection of independent networks and is still constantly evolving—number of networks, relations between them, and their connectivity are changing all the time. With the inter-connection of independent networks (domains), a mechanism is needed to route data between the different domains.

For fixed small size networks, routes can be created statically, but for bigger and constantly changing networks we need a protocol that can create, change, and withdraw routes according to the state of the network at a particular moment. Because of the sheer size of the inter-connected networks and the dynamics of changes occurring between them, static routing is not good enough since it cannot adapt itself to these changing conditions.

The Border Gateway Protocol (BGP) is a protocol that allows dynamic route creation and maintenance such that reachability is preserved. In contrast to static routing, BGP can automatically adjust routing configuration based on network path distance information received from other users of the protocol. BGP is a rich protocol that has many ways to sustain nodes or network failures as well as changes in the network topology. The main goal of the protocol is to maintain connectivity between inter-connected networks such that traffic can be routed to its destination.

1.2. BGP

As of today BGP is the core inter-domain Internet routing protocol. Its main objective is to exchange information about network connectivity and reachability between inter-domain entities called Autonomous Systems.

1.2.1. Prefix Routing

In order to route a data packet to a particular network, the destination network must be identified and located. BGP uses Prefix Routing to address networks. Network prefix is a general idea of assigning an identifier to a network in such a manner that for each data packet it is clear whether its destination lies inside a given network. On the other hand, prefix does not give us any information about how to reach its destination network—this information has to be obtained using the BGP protocol. For BGP to maintain connectivity means to be able to know how to route a data packet to the destination.

1.2.2. Autonomous System

Prefix routing is done between independent domains, also called Autonomous Systems (AS). It is the basic abstraction of independent entities comprising routers, networks, and policies. BGP can be seen as a protocol that maintains information on how data packets should be routed between different Autonomous Systems. Although an AS may consist of many hundreds of networks and thousands of routers, from an external point of view its behavior has to be consistent.

Every AS has a globally unique number (sometimes referred to as Autonomous System Number, or ASN) associated with it; this number is used in both the exchange of exterior routing information (between neighboring ASes), and as an identifier of the AS itself [26]. ASN is only an identifier, it does not reveal any information about the AS.

At the present time there are more than 27000 registered ASes communicating with each other using the BGP protocol [23].

1.2.3. Basic Operations

BGP is a protocol used for maintaining routing information between ASes. Each AS is connected to a number of other ASes (called neighbors or peers) and exchanges its routes with them according to AS-specific policies. After receiving information an AS may propagate it to its own neighbors. This lets the information spread like a gossip.

Most intra-domain routing protocols are based on Dijkstra's algorithm. In contrast, BGP is vector-path based, which means that information is not periodically broadcasted and that BGP routers do not have the full topological view of the network. Each router knows only how to reach its direct neighbors and through which neighbor particular Network Prefixes can be reached.

For each Prefix Network, an AS can use only one path as its default path. It can propagate this path to its neighbors, so that they can use the AS as transit for their traffic routing. In addition to the default path, all alternative paths leading to a given Network Prefix received from the other neighbors have to be stored. The reason is to be able to quickly restore the connectivity when the default path becomes unavailable—an AS may then start to use another path to route data packets and maintain connectivity even when a link has failed or network topology/reachability has changed.

Technical Description

Routing information sent between peers in BGP has two forms: announcements and withdrawals. A route announcement indicates that a router has either learned of a new network attachment or has changed policy to prefer another route to a network destination. Route withdrawals are sent when a router makes a new local decision that a network is no longer reachable [16]. Upon receiving a message each node decides what to do with it (i.e., change the routing table) and whether to propagate the message further. Both of these decisions are made according to AS policies. Effectively, BGP is a peer-to-peer protocol that distributes routing information and keeps routing tables up-to-date.

In contrast to other routing protocols, BGP does not periodically flood the network with routing information, but sends messages in an incremental manner. In theory messages should be sent only when a new route is announced, updated, or withdrawn. Such information should be propagated to all other ASes (of course according to policies).

BGP stores routes to other hosts as lists of ASes that have to be traversed by the packet to reach the destination, and IP address of the next hop on the path. When a new announcement is received, policies are evaluated to see whether the new route is better than the current one.

If this is the case, then the old route is replaced by the recently received. Next, if policies allow it, the route is announced to the neighbors.

Effectively, route update is equivalent to route withdrawal or announcement. BGP requires every router to store information received from every neighbor as well as information sent to other neighbors. Only one route can be used for one destination and it is prohibited to propagate a route different from the one actually used.

For routing purposes, storing only the next-hop IP addresses would be sufficient but BGP requires to store whole ASes path. This allows routers to choose better paths (by length) and avoid loops on routing paths—when an AS receives a route announcement it first checks if it is not already on the routing path, in which case, the route is not evaluated.

1.2.4. Routing Policies

BGP itself is a vector protocol in the purest form. When different routes for a given prefix are available, the shorter one is considered to be the better one. Although this behavior seems to be reasonable, it is not always desirable for a particular AS operator. For that reason BGP allows to create a set of rules that determine which route is the best and, what is more important, whether a route is suitable to be forwarded to a neighbor. Such set of rules are called policies.

In general, the relations between peers can be divided into two categories: (i) customer-provider, and (ii) peering. In customer-provider relation one peer is a connectivity supplier for the other—the provider provides transit services for which the customer has to pay. In peering relation, both peers are considered equal and exchange traffic between their networks, in general without money being involved.

Of course one AS can have many providers and/or customers as well as many peerings. In addition some peers and/or providers may be considered better and traffic should be directed to their networks even if the AS path length is suboptimal. With policies an Autonomous System can decide which neighbors to use for which prefixes, while still having other paths available as backup.

In fact routing policy is a way how routing decisions are made in the Internet today [12].

1.3. BGP Instability

Although the basic BGP protocol principle (exchanging information about existing routes between routers) is quite simple, the protocol exhibits very complex behavior. It is not uncommon that large-scale deployments of protocols introduce side-effects that are not seen in small-size deployments. It can be attributed to the fact that messages are processed by the servers and small interactions are becoming more significant when the amount of peer interaction patterns is increasing [1]. In large-scale deployments even anomalies that are very rare can have a big impact on the overall behavior.

Routing instability can be described as rapid change in network reachability information and topology information. Some of these changes are the results of genuine events in the network but it has been observed that surprisingly many are results of network link failures, configuration errors, and software bugs. Such information is then propagated throughout the network, generating large amounts of sometimes unnecessary traffic. This in turn degrades network performance and the overall efficiency of the Internet infrastructure [17].

Over the years some anomalies have been studied [17, 16]:

Redundant information Although the problem has been tackled and some improvements have been developed, still vast part of exchanged messages is pathological or redundant.

Uniform distribution There is not a small set of ASes or routes that are responsible for generating or propagating unnecessary announcements or withdrawals. It is therefore assumed that the protocol (and/or its implementations) is responsible for the pathological behavior and not only misconfigurations of particular routers [27].

Some techniques have been developed to improve the stability of the protocol:

Route flap damping When a route is advertised, withdrawn and then re-advertised it is considered to be flapping. Flapping routes are expensive because routers have to compute new routes every time a message is received. This leads to higher load on the router and may lead even to crashes in some cases. Therefore some BGP implementations make use of so called route damping—a penalty is assigned to a flapping route and if the penalty exceeds a given limit, such route is considered to be invalid. The penalty is decreased over time. If the route stops flapping it is advertised again—in which case the penalty is completely withdrawn.

Message aggregation There is often a correlation between updates and withdrawals. Quite often when a route is being withdrawn there is an immediate announcement - the route is just changing or behaves pathologically. To decrease the BGP traffic volume it is then advantageous not to propagate the message immediately, but to buffer it for some time and aggregate the information. If a route is withdrawn and then reannounced just the same, it is correct to discard both messages because the overall status is the same. Such approach decreases the traffic volume as well as computation on BGP routers.

Unfortunately both route damping and message aggregation introduce delays to route propagation. Therefore although the protocol and the routes get more stable, the convergence speed decreases. It has been studied that in fact route damping can significantly exacerbate the convergence times for relatively stable routes [19]. It is not clear whether this mechanism should be still used as routers are much more powerful now and fast route convergence is very important.

1.3.1. Project Motivation

Although a large fraction of the world's economy relies on correct BGP behavior, there is little understanding of its global behavior. It is therefore important to understand the behavior of BGP, as these issues are threatening the growth of the Internet. As of today our understanding of the protocol and the causes of its pathological behavior are not good enough to be able to predict how BGP will behave in the future. BGP is an essential part of today's Internet and therefore it is crucial that the protocol is able to maintain routing information even when the conditions change.

It is not clear how much of the traffic exchanged between BGP routers is pathological and how serious this problem is. The issue is being studied by the IRTF RRG group because finding answer to these questions might be crucial for the future growth of the Internet. Knowledge whether the increasing update load is caused by the growth of the Internet, increasing interconnection between ASes, or by more diverse policies (or maybe a conjunction of these factors) may allow us to take countermeasures: either try to tackle the causes of the problem or redesign some parts of BGP protocol. We want to be able to see how future changes of these factors will affect BGP's dynamic behavior.

To date, two main approaches of understanding the global BGP behavior have been explored: (i) detailed, small-scale simulations; and (ii) monitoring of the real deployment. We want to explore a different direction: model the global behavior of BGP in a scalable fashion (which implies focusing on the overall picture) to study the impact of diverse parameters on the global properties of the system.

Simulation may even lead us to find new problems in BGP's behavior that have not been observed before. For example issues that we have overlooked or that have not yet occurred but are likely to occur in the future. We need to search for limitations of the BGP deployment size. As BGP is such a crucial part of the Internet, it is extremely important to have a strong tool for analyzing the protocol and its behavior.

We are interested in finding answers to many unanswered questions about BGP dynamic behavior:

- How does the network and its dynamic properties behave if we change the network topology and/or its interconnectivity?
- How do different policies and techniques (e.g., route damping) affect the convergence time and routing instability?
- What is the impact of number of prefixes and nodes number on the protocol behavior?

In the IETF "BGP Stability Improvements," Internet-Draft [14], Li and Huston propose a few solutions to improve BGP stability. It is assumed that one of them should be chosen and then deeply examined. A BGP simulator could contribute significantly to evaluate these ideas and show empirically whether theoretical predictions are actually true.

There is a lot of BGP-related data available on the Internet. Big parts of network topologies can be reconstructed using ISPs looking glasses, Internet research-oriented Associations like CAIDA [5], or RIS databases [29]. We would like to have a simulator capable of simulating a BGP network of today's size, with about 27,000 ASes and about 250,000 prefixes [23]. We believe that simulating BGP on a small scale or with regular network (like cliques, meshes, and alike) does not give us enough information about the protocol's real behavior.

We believe that by studying the dynamics of BGP's behavior we will be able to contribute to the BGP research area. Our study may help increase the understanding of the protocol by AS operators, Regional Internet Registries (RIRs), BGP researchers and the Internet society. Data obtained from our simulations may help evaluate currently used BGP techniques and develop new ones. Possible hypotheses for further study are, e.g., band-stop filtering, path length damping, or optimal path hysteresis [14].

1.4. Outline

Chapter 2 provides information about current state of the BGP research. Both analytical studies as well as available BGP simulators are described, along with their advantages and weaknesses. We describe our abstract model of the BGP protocol in Chapter 3. Details of the protocol, along with their impact on the BGP dynamic behavior are discussed. Afterwards, in Chapter 4, we present the design of the simulator, along with the implementation in Java. In order to show the accuracy of our simulator, we perform a validation experiment (described in Chapter 5) by studying BGP Beacons behavior and comparing the results with those from a real-world study. Chapter 6 discusses the results, showing similarities and giving explanations for observed differences. Contribution to the BGP research field, possible future research, and improvement possibilities are described in Chapter 7.

Chapter 2

Current state of BGP research

Because the BGP protocol is playing such a significant role in today's Internet, the protocol behavior has been extensively monitored and analyzed [16, 15, 10, 11, 13]. The basic objective in BGP (exchange routing information between peers) is very simple but the scale of protocol deployment makes its behavior very complex. Over the years many studies have been conducted which can be roughly divided in two categories: monitoring and simulations.

2.1. BGP Monitoring

For monitoring the behavior of BGP, various tools are available like RouteViews [30], Looking Glasses, and RIS [29]. By using these tools it is possible to obtain some partial information about BGP router status, messages exchanged, and BGP traffic volume. CAIDA [5] monitors the state of BGP peers and its relationships trying to identify them and reconstruct a graph of the BGP network topology. As BGP monitoring and analysis is based on real-world data collection, study of future behavior and what-if scenarios is difficult or even impossible.

2.2. BGP Simulation

Because the protocol is too complex to be described analytically, BGP research quite often involves simulation. For the simulation we have to model the protocol and abstract some of its parts that are not relevant for the particular study. There is always a trade-off—on the one hand the more we abstract the less accuracy, on the other hand the simulation gets faster and can in fact simulate BGP on larger scale. Although data obtained from simulation is less accurate than real-world data collection (monitoring), simulation does allow to study future behavior and what-if scenarios.

2.2.1. Simulation Software

There are a number of BGP simulators publicly available. Although the simulators differ in certain points they show some significant resemblance: BGP is always literally or almost literally implemented.

Discrete Event Simulation

In his PhD thesis "An analysis of convergence properties of the border gateway protocol using discrete event simulation," Brian J. Premore [24] proposes to implement and run BGP protocol on top of the SSFNet [32] discrete event simulator. The author decided to create an as exact BGP implementation as possible. Although the author suggests that his implementation allows us to observe BGP behavior, the results contained in the thesis cannot be treated as

representative for large networks. He analyzed a few different network topologies: line, ring, focus, and clique topology for at most 30 hosts. Neither the used topologies nor the host numbers let us draw conclusions about actual BGP properties in Internet-like networks. Also his statement that although the Internet is not a clique it can be modeled as composition of line, ring, and clique topologies is neither proved nor convincing.

Another discrete event simulator is **BGP++** [2]. It is a BGP simulator using the Zebra bgpd daemon [34] on top of the ns-2 network simulator [21]. The BGP++ simulator actually simulates the network itself (in ns-2) at a very low level. BGP is not abstracted but runs as it is—it is a full-featured BGP implementation (Zebra) that is used in the real world. ns-2 is an event-driven network simulator, so the fact that BGP++ is a simulator built on top of ns-2 makes BGP++ an event-driven simulator as well. BGP++ runs BGP without any abstraction and is therefore perfectly suitable for simulating BGP in relatively small networks, with deep insight in each of the communication aspects (such as TCP congestion level). BGP++ also generates a very accurate state of each simulated router. This makes BGP++ a very useful tool to test and predict router states in changing conditions in small networks.

Unfortunately prefix tables are very big (each of the BGP nodes has to store information for each of the prefixes) and the amount of computation involved in a full BGP implementation is not negligible. Concluding, BGP++ is not suitable to study networks with many routers because of memory and CPU requirements.

C-BGP

C-BGP [6] is a BGP simulator addressing routing policy evaluation. The backbone network is not simulated in detail as in BGP++ but C-BGP still uses a full BGP implementation. It is possible to see the exact state of each node (with all BGP parameters) during the simulation and each BGP decision by evaluating all policies, preferences, and tie-breaking rules. According to the project homepage *"C-BGP is aimed at computing the outcome of the BGP decision process in networks composed of several routers"*.

During design and modeling of C-BGP, some things that are normally part of the BGP protocol were omitted. This involved for example the TCP level, connection creation, keep-alive messages and some parts of BGP Finite State Machine (as defined in IETF specification [26]) [25]. This is a step in the right direction: do not use resources and time to compute things that are not important for the simulation outcome.

Surprisingly C-BGP allows to simulate big networks by exploiting an interesting BGP property: with only few exceptions, computations made for different prefixes are independent with respect to the resulting nodes state. Therefore it is possible to do the computations for a part of prefixes, store them on hard drive, do the computation for another part of the prefixes and so on. This approach allows for the simulation of large networks of BGP routers, but unfortunately does not allow observation of protocol dynamics. When the prefixes are splitted in distinct parts and run separately, a lot of information is lost: the traffic between the nodes, their load and convergence speed. Those attributes depend strongly on all information that is exchanged at a particular point of time. Nevertheless this approach is very interesting, especially if we consider simulating different parts of prefix space on separate machines. C-BGP would then simulate BGP in parallel.

The simulation approach used in C-BGP is slightly different from BGP++. While BGP++ is an event-driven simulator, C-BGP is state driven—most important from its perspective is the steady-state of BGP routers and not how state changes over time. Another difference is that normally network simulators introduce some level of nondeterminism (as the network traffic is not deterministic). This is not the case in C-BGP: for every run with the same input data the output and the whole simulation run is exactly the same.

2.2.2. Conclusions

Both decision process simulator (**C-BGP**) and available discrete-event simulators simulate BGP at the protocol level in full detail. One can say that these simulators are prefix-oriented—the most important thing is that information for each prefix is correct at every moment of the simulation and as exact as possible.

The **BGP++** author points out that conventional simulators exhaust resources of a single machine and that it limits the simulation to small size networks. He is truly aware that results from such simulations are not inductive of the Internet's wide trends. What is suggested is to try exploiting parallel and distributed simulations, which is a very reasonable idea. What is not considered is lowering the simulation memory and CPU need by simplifying the problem complexity [9].

Important characteristics to observe (given the research goals set out in Chapter 1) is the volume of BGP messages exchanged between ASes, and time it takes for a message to converge. This means that we would like to be able to observe the network traffic, convergence speed, node load, but not necessarily the exact states. Of course, one cannot neglect the prefixes and BGP computations, but we want to abstract many BGP features and aspects which do not really influence the protocol's dynamic behavior. Although full-featured BGP solvers (like BGP++ or C-BGP) are more accurate, we think that some accuracy can be traded for less memory and CPU resources, giving way to simulation of large networks (i.e., the current Internet AS network).

Using discrete-event simulators like **SSFNet** might be interesting but analyzing large-scale deployment of protocols using discrete-event simulators is very hard, because of the need to order events correctly in parallel or distributed execution. Given that there are more than 27,000 heavily inter-connected BGP nodes and more than 250,000 prefixes, it is easy to spot that there will be billions of messages exchanged between these nodes during the simulation. We have to look for another, broader approach that would allow us to actually process such amounts of data.

Monitoring and analyzing the BGP protocol gives us very good understanding of the protocol performance during the time of monitoring. We can then analyze collected data and even try to analytically look for the root causes of the problems. What we cannot do using monitoring is predict how things will change when certain aspects of the protocol change. That seems to be a very important for the future growth of the Internet.

We believe that although some substantial simulation-based BGP research was conducted, none considered what BGP actually is: a simple P2P protocol, with a very big number of participating peers, which sometimes behaves pathologically. Trying to understand that pathological behavior using small or regular networks will probably not give insight in the complex large-scale behavior.

Chapter 3

The Model of the BGP Protocol

The challenge of designing a model of inter-domain routing is the appropriate level of detail of the protocol and the AS topology router network. In order to perform the simulation, we have to introduce a model that abstracts or ignores those BGP parts which are not relevant for our study (like link delay) while focusing on those aspects that truly affect BGP dynamics (like specific timer settings or Route Flap Damping).

3.1. Current State of the Internet

In the perspective of this MSc. research project, the Internet is not a regular network where every participating entity is equal and can freely communicate with other entities. In fact, the Internet is a network of autonomous networks (Autonomous Systems) that are connected with each other. Each autonomous network can consist of thousands of hosts and span large geographical regions.

One of the main goals of the Internet—global connectivity and reachability—is achieved because autonomous networks exchange traffic with each other. As not every network is directly connected to all others, a way to exchange information about Internet topology (inter-AS connections) and paths for forwarding packets—a routing protocol—is needed.

As presented in Section 1.2.3, most intra-domain routing protocols (inside the autonomous network) are based on Dijkstra's algorithm. In such protocols, information is being periodically propagated to other peers so that in the end every participating router knows the full topology of the network. With full knowledge of the graph, optimal paths can be computed. This is feasible due to the relatively small size of the network. However, link-state algorithms have issues with respect to scaling: memory and CPU requirements to handle large link databases are not insignificant. Also for very large networks, the amount of data exchanged by the routers can swamp the network. An example of the most popular protocols using full network topology are IS/IS and OSPF.

In inter-domain routing, full topology knowledge would be very expensive in terms of CPU memory usage as well as bandwidth costs. Inter-domain routing protocols have to be much more robust and scalable than intra-domain routing protocols as they work in a much larger environment.

Intra- and inter-domain routing also have different principles. The main goal of a intra-domain routing protocol is to fulfill technical demands, whereas inter-domain routing protocols are more complex as they have to reflect political and business relationships between the networks and companies involved.

BGP is the only inter-domain routing protocol available at this time. It is a protocol where peers make decisions based only on a limited view of the global network topology—the path to a destination point. Protocols, like BGP, where decisions are made according to a path to

the destination without knowing the full graph topology are called vector-path protocols.

3.2. Network Modeling

Routing algorithms operate on the network where data must be transfered from source to destination. With modeling and simulating BGP, a model of the network topology it operates on is also required. The network model includes the network links and the ASes.

3.2.1. Network Links

ASes are connected by TCP/IP links, where data is carried over various types of wired media (copper/optical fiber), switches, routers, etc. Bandwidth, signal propagation, contention at the switches and routers results in communication delays and dropped packages. Although these network characteristics influence the operation of BGP, the precise timing in milliseconds of the events is not of importance, but rather the occurrence of the event itself.

Also the topology of a particular AS network has influence on the propagation time inside that AS (especially when this AS spans multiple continents).

Though one can see that the cable and router delay may have some impact on for example message propagation time the difference is still in order of milliseconds. In reality BGP convergence studies [18, 3] show that convergence time is in the order of hundreds of seconds and does not distinguish values below 1 second.

Detailed simulation of data links is a very complicated task (done in for example the ns-2 simulator [21]) and doing that limits the size of networks that can actually be simulated. As BGP's self-introduced delay is a few magnitudes higher than the one introduced by the network we claim that we can actually ignore the link-delay and still design a valid BGP model within accepted accuracy boundaries.

Another thing that we do not want to model explicitly is connection creation and all kind of keep-alive messages. Both connection creation messages (as for example TCP Syn packets) and keep-alive messages are only important if something goes wrong (for example if a keep-alive message does not arrive on time the connection is considered to be broken). As long as everything works well, modeling them would introduce only noise without actually changing the behavior of the modeled protocol. Therefore connection-administration related messages are ignored and connection failures can be introduced manually if needed.

3.2.2. Autonomous Systems

The biggest simplification that we opt for is no explicit modeling of the intra-AS network topology. That means that in the model each AS is modeled as simple, atomic entity without taking its complexity and geographical span into account. There are two reasons for doing that:

1. There are about 27,000 ASes in the current Internet, which is already a lot. Trying to model their internal topology would require us to simulate hundreds of thousands of entities. That does not seem to be feasible.

2. The topology of each autonomous network is proprietary information of the entity controlling it. Obtaining and/or inferring such information is a complicated task and if we would like to model the today's Internet we would not be able to get creditable data.

   On the other hand, connections between particular ASes are in most cases not proprietary information and they can be inferred from observations using Route Views monitors, RIS routers, and traceroute utilities. CAIDA [5] publishes the inferred AS graph with (simplified) relations between ASes on a regular basis.

We opt for modeling the BGP network on a high level: without taking link capacity and delay into account, ignoring packets such as keep-alive messages and treating each AS as an atomic entity. Due to these simplifications, we should be able to apply our model to very big networks (like todays Internet) and run simulations upon them. This very high abstraction level is where our simulator differs from currently available ones—they cannot simulate big networks due to the amount of details they are taking into account.

Although the AS network topology is not modeled, its size is taken into account. If an AS consists of hundreds of routers, iBGP can take significant time to propagate information inside its network. Convergence time for big ASes is therefore not negligible—the time it takes to propagate a message inside an AS is modeled by a mathematical function with configurable parameters. This is of course a simplification but it still allows to make a distinction between big ASes (with hundreds or thousands of peers) and small one (with only one BGP router).

3.3. Routing

3.3.1. Basic Operation

The main BGP objective is to propagate routing messages within the AS network. Such a message is either an announcement—propagating that a route has changed or there is a new route available, or withdrawal—propagating that a route is no longer available. Upon receiving a message each BGP peer has to decide what to do with it: just store it, discard it, install and/or propagate it. Decisions are made according to policies which are expressed using many different parameters in BGP router configuration.

If a decision was made that a new route is the best one for a new prefix, the route can be propagated or (what can be surprising at time) that the old one should be withdrawn without propagating the new one. Routers have to know what they have received from other routers (RIB-In), what they have installed as default routes (FIB), and what they have sent to other routers (RIB-Out).

If all BGP routers would send messages as soon as they can, the convergence time would be at the level of tens or hundreds of milliseconds. In reality there are some factors slowing BGP down—in order to decrease the amount of messages exchanged and the noise.

Timer limiting the amount of messages is such a factor. Its main objective is to bound the number of messages sent from one peer to another. Such timer states how much time has to elapse before a next message concerning the same prefix can be sent—effectively slowing down the convergence time. In the BGP such timer is called Minimum Route Advertisement Interval (MRAI) and describes the minimum time interval between two consecutive prefix updates sent to the same peer. Another factor slowing BGP down is route flap damping technique which suppresses propagation of information about routes that are considered to be unstable. Because of these factors, the messages sometimes have to wait before they can be sent to other peers—this is the trade-off between faster convergence time and amount of bandwidth consumed by the BGP protocol itself.

3.3.2. Protocol Abstraction

In order to model and simulate the BGP protocol we have to decide which parts of it are relevant and which parts do not have much impact on its global behavior. Like in the previous section the main factor for choosing whether something is important is whether it is relevant for answering the questions about BGP dynamic behavior and impact of the network size on this behavior.

BGP's main principle is fairly simple: propagate routing information between peers to maintain connectivity and reachability, trying to keep forwarding paths short. In fact BGP

is much more complicated than that. It has many properties that can be set in order to change its behavior. One of the distinctive protocol characteristics is that it is policy-driven and not every path will be propagated to every neighbor, and not always the shorter path is selected for propagation. It has some techniques for traffic regulation like MRAI—a parameter that states how often two peers can exchange data, and the route flap damping—a penalty mechanism for bad behaving routes. Besides that BGP is not a single-threaded protocol - it is being run by thousands of peers at the same time and their interactions also impact its behavior. Though it is easy to understand the protocol's main principle, it is really hard to spot what is responsible for its dynamic behavior.

Finite State Machine

The current version of BGP is described in RFC4271. The data structures and Finite State Machine (FSM) described there are conceptual and do not have to be implemented precisely as described there, as long as the implementations support the described functionality and exhibit the same externally visible behavior [26].

According to the standard, a BGP router is a (FSM) and its state changes when an event occurs. From our point of view the FSM is not needed to successfully model the BGP protocol. In fact, the FSM's main interest is in the connection creation, keep-alive messages and connection termination. As we have stated in previous section we do not intend to model these network events. Hence, we do not need the FSM to take care of them. Our main interest is sending updates and the route propagation part of BGP, not router or connection maintenance. Of course our modeled BGP router also would have to have a state but it does not have to resemble the complete BGP Finite State Machine.

MRAI Timers

The parameter MinRouteAdvertisementIntervalTimer (MRAI) determines the minimum amount of time that must elapse between an advertisement and/or withdrawal of routes to a particular destination by a BGP speaker to a peer [26]. The implementation of this timer is very different depending on the router vendor [18]

- Cisco routers use 30 seconds as default (plus some random number to prevent self-synchronization)
- Juniper routers use 0 seconds—no limits on how often information concerning one prefix can be sent

An interesting thing is that both vendors implement MRAI timers on a per-peer and not a per-prefix-per-peer basis. This makes it much less memory-consuming and is still compliant with the RFC defining the BGP protocol.

As MRAI timers can introduce serious delays in the BGP message propagation time, we cannot simply ignore them. We do not have to model them on a per-prefix-per-peer basis (as BGP implementations do not do that either) but they are a very important factor influencing BGP's dynamics.

Route flap damping

In Section 1.3 we discussed how route flap damping works and how the convergence of the flapping route is affected. Many studies (e.g., [4] and [19]) have shown that in fact a route can be considered flapping at some point in the network as a result of a single withdrawal or re-announcement. Since a route propagation can be suppressed even up to one hour, flap damping has a very serious impact on both connectivity and convergence time.

Route flap damping is believed to be harmful [4] and its usage is currently discouraged by the RIPE community [28]. Nevertheless many routers seem to be still using route flap damping and it is believed to be the cause of the long convergence time observed for some routes.

We are interested in the impact of the route flap damping on the BGP protocol and therefore in order to simulate its behavior we have to include it in our model.

BGP parameters

As was stated before there are a lot of BGP related data that do not have any, or not significant impact on the BGP's dynamic behavior. As an example one might look at the NEXT-HOP property of BGP announce message. This property contains the IP address of the next router to contact to forward the packet.

Routers have not been modeled, only ASes and their peer/transit relations. Packet forwarding will therefore not be done and there is actually no need to know the first router on the BGP path. It is of no use to model it then.

Many factors in BGP are policy parameters, which are difficult to infer. It is very hard to state in advance which of these parameters might be interesting. Because of that we want to postpone this decision to the last moment—the simulator running time—and allow to specify any kind of policy, taking into account both legitimate BGP parameters as well as totally arbitrary ones. As decision making varies from peer to peer, there has to be a possibility left for peers to express all kind of policies they might want.

3.3.3. BGP modeling

This subsection will explain how the parts of the protocol that are considered important are modeled.

BGP Message

The only thing that we require from a BGP message in our model is that it is capable of transmitting information from one peer to another. There is absolutely no need to stick with BGP messages as described in RFC4271 [26]. On the other hand we do not want to change the BGP message semantics as this might have impact on the measurements (if we would like for example to count the number of messages exchanged). That means that our modeled message cannot contain announcements for two different prefixes if they have different routes (just like the real BGP message).

In our model a BGP Message contains only information about sender, a list of withdrawn prefixes, a new route and a list of prefixes for which that route applies. As in normal BGP either the withdrawal or the announce part might be omitted.

ASes, ASNs and Prefixes

We want to model each AS as a separate and independent entity capable of communicating with other ASes by sending BGP Messages and making decisions according to its own policy. Still we treat each AS as one router and model it as such.

ASNs (Autonomous System Numbers) and Prefixes are longer than they could be. We want them to take as little space as possible (for sending messages over the network and for storing routes) so in our model all ASNs and Prefixes are enumerated and each AS has access to an entity capable of translating enumerated number into a real ASN or Prefix.

FIB, RIB-In and RIB-Out

As stated before, FIB is the store containing default routes for given prefixes (installed routes). RIB-In contains information about the last route for a particular prefix received from a particular neighbor, whereas RIB-Out contains information about the last routes sent to particular neighbors.

BGP makes a distinction between FIB and RIB, because only the FIB is used to forward packets and it is therefore essential to have a faster FIB than RIB. In our case—no forwarding—there is no need for a faster FIB. In fact, every time information from FIB is needed, information from RIB-In is required as well. This is why in the model FIB and RIB-In are treated as one Prefix Store. The Prefix Store for each prefix knows the routes received from the neighbors as well as the currently installed route (if any).

RIB-Out has a very nice property—it can be recomputed. To recompute RIB-Out it is enough to apply policies to FIB and see which route would be propagated to whom. As it turns out the simulator is memory- and storage-bound so it is useful not to store RIB-Out explicitly but recompute it every time it is needed.

Policy

BGP policy can be specified by many available configuration parameters. Some of them are in the BGP standard, some are vendor-dependent. It was already stated that policies need to be very flexible. This is why we opt for modeling policies as "black boxes" capable of answering only two questions:

1. Is route x from peer A better than route y from peer B?

2. Can route x from peer A be advertised to peer B?

Each peer can then have it's own "black box" and therefore enforce its own policies.

MRAI and Route Flap Damping

Both MRAI timers and Route Flap Damping are not really abstracted—they are modeled just as they are.

For MRAI all computations are done as normally, but before a route is propagated it is checked how much time has elapsed since the last message was sent. If the pause was not long enough, the message is suppressed.

Also Route Flap Damping modeling is very straightforward—if a route is considered to be damped it is simply not taken into account during calculations.

iBGP Convergence Time

To model iBGP convergence time an exponential function is used. The function has two parameters: (i) maximum convergence time, and (ii) number of neighbors for which this value is reached. The function is as follows: $maxValue^{min(maxNeighbors,asSize)/maxNeighbors}$ where $asSize$ is size of the AS.

For one AS this function gives a constant time that each update will spend inside the AS before getting propagated to its neighbors. It is very important to preserve the order of the updates: two updates from the same peer to the same router have to be processed in the same order, because otherwise the system could converge to the wrong state if for example an announcement and a withdrawal for the same prefix were swapped.

3.3.4. Summary

In this chapter a model of the BGP protocol was presented. In order to shrink the model to a computable size, many aspects were abstracted, simplified or even ignored. Still, an effort was made to include everything that was considered important for answering important questions about BGP's dynamic behavior.

The model covers receiving messages, applying policies, and forwarding routing paths. Important characteristics and parameters that dominate the dynamic behavior are incorporated. Other parameters and details of less prominent influence are abstracted from. The cost in accuracy (of the model) will be evaluated in the next sections by validating the model and simulation in a series of experiments and comparing the results with real-world observations.

Chapter 4

Design and Implementation

In order to perform simulations using the model designed in the previous chapter, it has to be implemented as a computer program. There are many ways how the model can be implemented, but in order to achieve the goals—being able to analyze BGP's behavior in large networks—a few requirements have to be added to the design and implementation of the simulator.

4.1. Design Requirements

The design of the simulator is driven by the requirement for scalability of the system. If we want to be able to successfully run instances of the model (as a simulator) on a large scale, careful choices have to be made with regard to the implementation.

KISS (**K**eep **I**t **S**imple **S**tupid) is a design principle that implies that simplicity should be the key goal and unnecessary complexity should be avoided. This principle is applied to obtain a minimal but sufficient simulator, one that will be capable of helping us to find answers to important questions about the dynamic behavior of BGP. This design principle has already been applied to the model—by ignoring many not important aspects of the protocol—and will be applied further on.

Design requirements of our simulator include:

Scalability We want to be able to simulate BGP on a large scale. This means simulating AS networks of size comparable with today's Internet, and even simulation of larger networks. In order to achieve high scalability, solutions that introduce bottlenecks have to be avoided and the simulator has to be able to efficiently use available resources.

Efficiency In order to simulate a large network, the simulator has to be efficient. We do not impose any particular performance numbers, but every design and implementation decision must be made with efficiency in mind with respect to computational and communication complexity as well as memory usage.

Relaxed accuracy Given the highly abstracted model, results will be less accurate. This is clearly a trade-off which has to be carefully balanced. We are neither interested in the BGP state of particular nodes, nor in the exact contents of the messages exchanged between peers. In our study, we are interested in statistical data and/or impact of significant policy changes.

Extensibility It is impossible to clearly separate the significant and not so significant parts of the protocol beforehand and also we do not a priori know the kind of experiments that we might want to conduct. Simulation software must be therefore extendable in order to be able to include components needed for further studies. We include things that

we consider necessary but leave "open" anything that might be considered significant later. This leads to a design using loosely coupled components so that each one of them can be easily changed, adjusted, exchanged, or removed. This is of course not always possible, but should be considered when making design decisions.

4.2. Application of the Model

The model described in the previous chapter maps naturally to a simulator running a large number of ASes using message passing for information exchange—in our case modeled BGP messages. ASes are logically separated from each other as much as possible, just like they are separated in the real BGP network.

For ease of modeling and distributed execution we opt for process-oriented simulation model semantics [31]. That means that each process behaves like an independent program, has a well-defined algorithm, can generate and process entities (messages), and multiple instances can be run in parallel. In our simulator, each AS behaves like an independent process, runs independently from the others and communicates with them using messages. This implies that even if ASes are to share the same resources, they should be unaware of it and not make any decisions based on knowledge other than available local information about network topology.

ASes communicate with each other using the abstracted network described in Section 3.2. This means that although ASes have communication links between each other, the only type of message they are allowed to exchange is a valid BGP update—announcement or withdrawal. An AS designed in such a way behaves just like a BGP router. The only knowledge about the surrounding world it has are ASes to which it is connected.

We believe that building the simulator on top of independently running ASes is a simple, yet a very powerful way to simulate BGP, which provides the flexibility to extend the simulator in the future.

4.3. Simulation Execution Environment

Application of the model to process-oriented simulation described above does not imply a particular execution mechanism for the simulation. Two viable approaches are considered for the discrete event system under study:

Discrete-event simulation The simulation method executes events in chronological order. This requires events to be ordered by a scheduler. The "time" is always moved to the first event in the event-queue. In our case, an event in the simulator is a BGP update.

A few discrete-event simulators were described in Chapter 2. By using such an approach, a very good accuracy level is achieved (as for example C-BGP has even deterministic output) but it is limited in scalability by the effectiveness of the Parallel Distributed Event Simulator (PDES). In Section 4.1 we have stated that scalability is one of the driving factors in designing our simulator, so we have to look for a more scalable approach.

Emulation The main difference between discrete-event simulation and emulation is lack of strict event ordering in the latter. The main idea in emulation is to let the simulated entities run in quasi-real time and react to generated events when they occur. Causality follows from execution order in real-time (wall clock time).

In our case it implies that each AS is running concurrently and reacts to received messages. When processing of the received message triggers sending of new messages

(e.g., prefix announce propagation) such message is sent directly to the recipient without delay other than the one introduced by the AS itself, for example, by MRAI timers. This means that when a message is sent between two ASes only two entities are involved—sender and recipient—and no distributed global coordination is needed.

Emulation gives the entities impression of being run in the real-time, without any bounds from the simulator. Scaled time is used to control the progress of the simulation. Very good scalability level is achieved thanks to the fact that ASes do not need to communicate with each other in order to continue the simulation.

Since we want to simulate very large networks—tens of thousands of peers—we consider the scalability property of the emulation approach to be a significant advantage and therefore decided to use it as the simulation framework for the simulator.

To achieve the effect that each AS runs in quasi-real time, each AS is encapsulated in a lightweight thread. Even if all threads are not executed at the same time, the thread scheduler can still make the threads behave like they were executed all at the same time.

4.4. Hardware Environment

The design of the simulator allows for a distributed execution in order to study the BGP behavior in large-scale AS networks. In our project, a homogeneous cluster is assumed for parallel execution of the simulation. Such a cluster can consist of many computers called nodes, connected with each other by a high-speed low-latency network, capable of running a parallel application on many of them simultaneously. Homogeneity means that each of the nodes (computers) used have the same or comparable hardware resources available. Thanks to that, two similar tasks should on average take the same amount of time on any of the nodes, and synchronization overhead of two processes running on different machines can be kept relatively small.

By using an appropriate level of abstraction during the implementation, it is possible to hide the fact that the nodes are connected by a computer network and have the impression of running the application on one big virtual computer.

We want our application to scale and to be extendable, therefore the network communication overhead has to be kept minimal and any central component must by avoided. Such a component could become a bottleneck when more resources would be available to the simulator. We opt for a model where if one entity wants to send a message to another one, no other parties has to be involved in the process—interactions between two parties should be kept local.

4.5. Programming Environment

Java was chosen as the programming language to implement the simulator. As an object-oriented programming language, Java helps to write well-designed programs and what is important—it makes code refactoring easy. Our simulator is highly experimental software and having well maintained code is of great value. Besides that, Java interfaces are well designed for implementing loosely coupled components, enforcing very strong functionality and implementation separation. Each entity described in Section 3.3.3 is simply hidden behind a Java interface and has one or more implementations.

Another argument for using a fully object-oriented language to implement the model is that the process-oriented simulation approach maps naturally to object-oriented languages. This is due to strong functionality-implementation separation and ability to clearly specify interactions between independent entities at the language level.

4.6. Execution

Execution of the simulator is performed on many compute nodes, all of which are running the same program (Single Instruction Multiple Data—SIMD). At the start of the simulation every compute node reads its configuration from configuration files and learns which ASes it will be simulating in the particular execution.

Besides computation nodes there is also one special node: the coordinator. This node is not directly involved in the computation (BGP simulation) but is responsible for controlling the course of the execution, introducing new events and necessary synchronization. The coordinator ensures that the simulation is executed according to the given scenario.

4.6.1. Communication

Communication between the compute nodes is done using TCP channels. As each compute node hosts many simulated ASes, which can have neighbors hosted by different compute nodes, it is required that every compute node can freely send a message to another one. Therefore we create a TCP channel between every two compute nodes, so that they can freely exchange messages without involving any other node.

Besides having a TCP connection to every other node, each node is also connected to the coordinator. This link is used for monitoring and controlling the compute nodes' state as well as ASes hosted by it.

Messages exchanged between two normal computation nodes can only be BGP updates: two nodes do not have any other possibility of direct interaction with each other. ASes residing on the same compute node are unaware of this collocation on one node. The only way they can exchange a message is over the TCP link (even if the recipient is on the same physical machine as the sender).

Messages to and from the coordinator can never be BGP updates because the coordinator is not taking direct part in the BGP simulation and does not host any AS. Such messages are arbitrary commands sent to and from the coordinator which are simply executed by the recipient.

Using the Java builtin RMI as communication library failed, because RMI is unable to handle the amount of communication generated by the simulator. Besides that, RMI uses either standard Serialization or Externalization. Both of them are considerably slower than the serialization currently built into the simulator which is highly aware of the type of serialized data and can omit many unnecessary checks.

4.6.2. Coordinator

Although the coordinator is a central authority, it is not a bottleneck for the simulator, as it does not take part in the simulation. The coordinator is an external entity, modeling some environmental events, effectively controlling the execution of the experiment. The load of the coordinator node during the experiments is negligible.

The main role of the coordinator is to make the simulation management easier and take the management tasks away from the compute nodes. The responsibilities of the coordinator are as follows:

Introduce and trigger events ASes do not have to know when to start announcing or withdrawing a new prefix, when to connect to a new neighbor and so on. It is the coordinator's duty to deliver appropriate commands on time to the ASes.

Synchronize Compute nodes have to synchronize at the beginning of each experiment in order to know when they can start connecting to each other. It is also important

that every node knows the time that is considered to be the start of the simulation (which depends on the time it takes them to start at the beginning). Unfortunately not everything can be broadcasted by the coordinator—some data (e.g., AS network description) is loaded by the compute nodes from files at the start, as sending it to each node would take too much time.

Monitor It is important to know the state of the nodes during the simulation. In order to be able to accurately simulate BGP we cannot permit the nodes to get overloaded by the amount of data to process or send through the network. It might also be interesting to know some characteristics of the ASes' behavior (e.g., process load or memory usage). Such data can be periodically sent to the coordinator and stored in human- or machine-readable form.

Process results Normally every experiment run on a simulator generates some kind of results. Though compute nodes can write their results directly to the files on hard drives, it is more convenient to send the data back to the coordinator instead. It can be processed, filtered and written in human- or machine-readable form by the coordinator.

Finishing the simulation Whether at the end or unexpectedly, the simulation eventually has to stop if stop conditions are met. Compute nodes have to be aware of it because they have to release resources they are using (e.g., close log files so that no data is lost). The coordinator notifies all nodes and waits for them to shutdown.

Having the coordinator is not a "must" for the simulator but it helps to keep the management and scheduling apart from the core of the simulation—BGP updates propagation. Though it is a single component, it does not have negative effect on the scalability. On the contrary—by introducing it we have taken some work away from the computation nodes and made the simulator scale even better. Distributed coordination is a complex task and putting all coordination responsibility in hands of one entity, while not allowing this entity to become a bottleneck, is a significant simplification.

4.6.3. Compute Node

Because most of the management job is done by the coordinator, the design of a compute node is straightforward. Its main task is to host simulated ASes and give them means to communicate with each other.

It is the compute node's responsibility to maintain the TCP connection with other compute nodes. All ASes hosted by one compute node share the same TCP connection but are unaware of it. Also the possible transmission delay cannot affect the AS—from its point of view a message is sent instantaneous even if it is buffered for some time before being really sent.

We use Java's asynchronous I/O to forward messages efficiently. Messages from ASes are put into a queue before sending so that they do not have to wait for it to be actually sent.

It is important to point out that the communication between different compute nodes and between the nodes and the coordinator is done independently. We want to be sure that even if links between nodes are heavily congested and the buffers are full, we will still be able to communicate with the coordinator (e.g., to report the heavy load).

4.6.4. AS Implementation

Using an object-oriented programming language (like Java) to implement an AS, means that the entities described in the model must be mapped to he programming language level. Our main implementation principle is high interface–implementation separation. Thanks to this we will be able to freely exchange components (and/or their implementation) of which an AS is made. Base components of our AS implementation are:

PrefixStore — responsible for altering the state of the router when a new BGP update is received. PrefixStore is capable of processing route announcements and withdrawals. The purpose of this class is to encapsulate the decision making algorithm (as described in [26]) according to the policy of the particular AS. PrefixStore uses PrefixCache as database and propagates its decision to OutputBuffer.

PrefixCache — models RIB and FIB in our simulator. It behaves like a map that for every prefix returns a data structure containing all the information about a prefix known to the AS (i.e., all the routes received from all neighbors). PrefixCache is used by PrefixStore but hides its implementation such that the PrefixStore does not really know how the prefixes are stored. In fact only a part of the prefixes is stored in memory and the rest is stored on the external storage implemented by DiskStorage—prefixes are sent to the storage when there are too many prefixes in memory and read out when needed.

DiskStorage — a helper class encapsulating the storing algorithm, hiding its implementation from PrefixCache. Its responsibility is to write and read prefixes from storage when it is asked to do so. Currently prefixes are stored on the hard drive but there is also an implementation that stores them in memory, compressed using GZIP algorithm. However, compression has proven to be much too slow for our purposes, it is a good illustration how one can change the implementation of one component without affecting others.

OutputBuffer — encapsulates propagation algorithm. Whenever PrefixStore installs (or removes) a new route there is a possibility that the route will have to be propagated to some of the AS's neighbors. PrefixStore simply notifies OutputBuffer about the changes made and flushes the changes after all the data from a BGP update has been processed. OutputBuffer processes the received information, makes decisions to which neighbors to propagate it (according to the policies), aggregates prefixes and withdrawals in BGP messages, and sends them to the neighbors (without actually knowing where they are).

FlapStore and MRAIStore — two stores maintaining information about the timers. We described MRAI timers and Route flap damping technique above. MRAI suppresses sending of a message to a neighbor if not enough time has elapsed since the last announcement was sent, and Route flap damping suppresses a route if it is considered to be flapping. FlapStore and MRAIStore maintain information about such suppressed objects. Their responsibility is to make sure that appropriate actions will be taken as soon as the suppression time expires.

4.6.5. Execution Flow

In order to show what the execution flow looks like we have to make a distinction between (i) communication flow and (ii) AS flow. The first one deals with what happens between sending of a BGP message by one AS and receiving of this message by another AS, while the latter deals with how a BGP update is processed by the destination AS.

Communication Flow

In order to send a message to another AS, the sender has to use the communication capabilities of its compute node. The flow is as follows:

- A prepared BGP message is added to the sending queue, where it is serialized into bytes and written to the buffer.
- At this moment the sending AS considers the message to be delivered.

- The compute node tries to send the buffer contents to the receiving compute node using asynchronous I/O.
- The receiving compute node deserializes the message, finds the recipient AS, adds the message to its incoming queue, and notifies it that there is a new message to process.

Execution Flow

Each AS performs the following task in an infinite loop:

- Wait for an object in the incoming queue.
- If the object is an incoming BGP Update:
  - Iteratively add information about each received prefix and withdrawal to the PrefixStore.
  - The PrefixStore updates FIB and RIB (according to the policies) and eventually flushes the OutputBuffer.
  - OutputBuffer decides which messages should be sent to which neighbors (enforcing the policies) and which to deffer (because of the MRAI timer).
- If the object is information about an expired MRAI timer:
  - Send deferred messages to the appropriate neighbor.
- If the object is information about an expired Route flap damping penalty:
  - Mark the appropriate route as eligible and run the decision process. Eventually propagate the new route to the neighbors.
- If the object is a message sent from the coordinator to the compute node:
  - Execute the message.

4.7. Summary

The design and implementation of the simulator is driven by requirements such as scalability, extendibility and efficiency. We have decided to implement the model as a parallel application running on a homogeneous cluster. Java was used a programming language, because of its object-oriented approach, good specification–implementation separation, direct mapping from process-oriented simulation and portability. The KISS principle was applied to keep implementation as simple as possible.

A validation experiment will be conducted to show to what extent the simulator reflects the BGP behavior and indicate which parts of the protocol need more detailed modeling.

Chapter 5

Validation Description

żółw zażółcił gęslą jaźń!!! In previous chapter we have presented the design and implementation of the BGP model capable of simulating BGP on a Internet-size scale. In order to make it a useful tool to draw conclusions about BGP's behavior, it needs to be calibrated and it has to be shown that it is in fact a valid BGP simulator. That means we need to show to what extent the results obtained from the experiments conducted using our simulator resemble real BGP behavior.

5.1. Model Validation

During the model design phase a number of decisions affecting the accuracy of the simulator have been made. An AS internal network is not modeled but treated as a single router, parts of the protocol like keep-alive messages or connection creation are not included in the model, and network link problems (i.e., transmission delay and link congestion) are not taken into account. We argued that the overall impact of these factors on the protocol's dynamic behavior is negligible. In order to show that our assumptions are indeed correct, we need to show that the designed model of BGP protocol is correct with respect to some particular characteristics.

Besides abstracting the protocol we have also used a less strict simulation execution method. The simulator is not event-driven because it would make it not scalable. We have opted for emulation instead, knowing that the resulting simulator will not be as accurate as available event-drive simulators. This is due to the high non-determinism level in the simulator execution introduced by the quasi-real-time execution model and it cannot be a priori stated how much inaccuracy in our simulator is inducted by the fact that it is run in parallel without strict event ordering.

We will demonstrate that, although we have abstracted or omitted parts of the protocol and used a non-standard approach to the BGP simulation problem, the designed model in fact reflects some interesting BGP characteristics.

5.2. Validation Techniques

One of the classical papers in the validation field is the one written by Naylor and Finger [20] which describes recipies for model validation. One of the proposed ways is to compare the model input-output transformations to corresponding input-output transformations of the real system. This can be done only when the input-output transformations for the real system is available, which in fact is rarely the case (if it was possible do arbitrary input-output transformations using the real model there would be no need to build the simulator). If there is absolutely no possibility of studying input-output transformations of the real system to test

how representative the simulations output is, one has to resort to analytical studies of the model and expert reviews of output data.

Even though BGP behavior has been studied and analyzed extensively, a deeper understanding of BGP phenomenas is still missing in the community. Many of anomalies have been named but it is not clear what is in fact causing them. This makes it hard to use experts reviews to validate the output of the simulator as the expected output is not known.

There is also no possibility of running arbitrary input-output transformations on the real BGP network—it is a very big, heterogeneous network of great importance to the Internet and the world's economy. It is therefore not possible to perform experiments that would affect the overall BGP performance. Because of that we concentrate on BGP experiments that can be conducted on the real network without affecting its overall performance and do not require too many entities to participate.

5.3. Experiment Framework

Our BGP model was not intended to be a full BGP simulator, reflecting every aspect of the protocol and network behavior. Our interest was limited to studying BGP dynamic behavior, not looking at a particular router state but rather at the globally visible protocol characteristics.

By means of validation we want to show with respect to which properties our simulator is accurate and to what extent. Such knowledge could be later used to observe how exactly those properties affect the simulation output and allow to get a much deeper protocol understanding.

Possible BGP characteristics that can be studied using our simulator include properties such as:

- Convergence time with respect to various parameter settings and techniques used (e.g., MRAI timers, route flap damping).
- Impact of new techniques (such as studied in [22] and in [7]) on the amount of messages exchanged between peers, convergence time, connectivity, etc.
- Impact of properties such as average AS path length, AS network size (with respect to the amount of peers as well as to interconnectivity level) on BGP dynamic behavior.

5.4. Validation Approach

Because BGP is such an important factor of today's Internet it has been monitored for many years now. CAIDA [5] studies network topologies at many levels, the RIPE RIS project [29] and University of Oregon's Route Views [30] have been monitoring BGP updates exchanged between peers. Many people have analyzed this data and tried to draw conclusions about the BGP behavior (e.g., [3] and [18]). Our approach is to try to run such an analysis of BGP on our simulator and see to what extent the results from the conducted studies match with the results from the simulation. In order to do so, a study has to be chosen that is reproducible and can be run outside the real BGP network environment—if the cause of the observed behavior is known it can be introduced into the simulator and the resulting behavior can be observed.

The main goal of rerunning such an analysis is to show which observed BGP properties hold and to what extent. It is also very important to see which properties do not hold—such information could be used to improve the simulator in the future—we could for example conclude that we have abstracted some parts of the protocol too much and would have to reconsider decisions we made. Using validation one can measure how well and how accurately we captured the dynamics of the protocol in the model.

5.5. Experiment Description

We have decided to use "BGP Beacons" study by Mao et al. [18]. They analyzed BGP beacons behavior using Route Views monitors. Although they studied issues such as, interarrival times, and different BGP implementations, we will only concentrate on convergence time study.

The study concentrates on observing special BGP prefixes—so called BGP beacons. Beacons are announced and withdrawn at known times and treated as signals going through the network. Because of the limited access to BGP routers, Route Views monitors have been used to study the signal duration and relative convergence times. The purpose of the study was to get a better understanding of how information is propagated inside the real-world BGP network.

The main goal of the validation experiment is to compare the distribution of the BGP beacons signal duration observed in the real BGP network with results from the simulator. By analyzing the results it will be possible to say whether the convergence characteristics of the simulator are correct, increasing the confidence in the correct design of the model.

5.5.1. BGP Beacon

To study BGP dynamics, cause and consequence relations are needed. Although a lot of data from BGP monitors is available, it is hard to reproduce the observed behavior. This is due to the fact that what monitors observe is a result of unforeseen events happening somewhere in the network. It is hard to infer what was the root cause of an observed event. Instead of recomputing the root causes using the observed behavior, people have decided to tackle the problem from the opposite direction: introduce a known event at a known time into the network and observe the behavior caused by it. This means that instead of inferring the cause of an event by studying its consequences, one can study consequences of the known cause.

A BGP beacon is a special network prefix. Its main task is not to maintain connectivity to the network it describes but to give the possiblity to monitor the BGP updates caused by announcements and withdrawals. Such prefix is periodically announced and withdrawn at known times. When the time of the prefix announcement is known, it is possible to measure its convergence time using available BGP monitors. Because the time between consecutive events (i.e., announcements and withdrawals) is relatively small—2 hours—there are enough samples available to draw conclusions about beacon prefixes convergence time for both announcements and for withdrawals.

5.5.2. Route Views

The University of Oregon's Route Views project was originally conceived as a tool for Internet operators to obtain real-time information about the global routing system from the perspective of several different backbones and locations around the Internet [30]. Route Views are monitors that collect data from different sources by peering with them. Routes received from peering sessions are never passed on or used to forward packets and Route Views servers do not advertise any prefixes themselves.

Because Route Views only receive BGP updates and never send any, effectively they can be seen as "black holes" or "sinks" to which many BGP routers send announcements and withdrawals (just as to normal neighbors) but with the only task to store them. Consequently, there should not be any AS path containing a Route Views AS number as transit AS.

All the updates received by Route Views are published online, allowing researchers to analyze them and study various aspects of the BGP behavior. In particular, Route Views can be used to study BGP beacons convergence time—the time elapsed between the beacon advertisement/withdrawal and the time the beacon is seen by Route Views servers.

5.5.3. Signal Duration and Relative Convergence Time

Mao et al. [18] studied BGP beacons convergence time by means of two different metrics, namely (i) relative convergence time, and (ii) signal duration time. We will study both parameters using the simulator and compare the results with the ones from the real BGP network to measure to what extent convergence time distribution is preserved.

Signal Duration

Mao et al. used the concept of *input signal* to describe both announcements and withdrawals of the BGP beacons (at their originating hosts). The BGP network was treated as a giant nondeterministic signal transducer where each input signal causes output signals that can be observed at many locations. The term *output signal* was used to describe incoming updates, which were a consequence of the input signal, as observed by one host. A sample output signal (from host AS2) could for example be:

Time	Type	AS Path
3	A	AS2 AS3 AS4 AS5
7	A	AS2 AS3 AS6 AS5
10	A	AS2 AS3 AS5

In this situation, the first update (as seen from Route Views server peering which peers with AS2) of the output signal came in at time 3 and the last one came in at time 10. We define the **signal duration** as the time elapsed between the first and the last update in the output signal from the Route Views servers perspective. In this case the signal duration time is 7. If there is only one update in the output signal the signal duration is 0.

The signal duration can be seen as the time it takes for the input signal to converge to a stable state for the observed AS.

Relative Convergence Time

Normally one Route Views server peers with more than one AS. Signal duration is computed independently for each peer but apart from that, Mao et al. define also *relative convergence time*. Assume a Route Views server receives output signals from two different peers (for the same input signal)—AS2 and AS10:

From AS2:			From AS10:		
Time	Type	AS Path	Time	Type	AS Path
3	A	AS2 AS3 AS4 AS5	8	A	AS10 AS13 AS27 AS5
7	A	AS2 AS3 AS6 AS5	13	A	AS10 AS13 AS25 AS5
10	A	AS2 AS3 AS5	23	A	AS10 AS13 AS5

Signal duration for AS2 is 7 and for AS10 15. Relative convergence time for a peer X is the time between the first update in the output signal from any peer and the last update in the output signal from peer X. So the relative convergence time for AS2 is 7 $(10-3)$ whereas for AS10 it is 20 $(23-8)$.

Figure 5.5.3 illustrates the difference between normal convergence time, relative convergence time and signal duration.

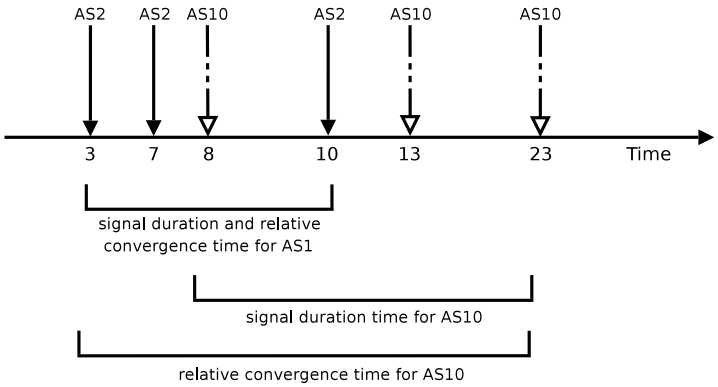


Figure 5.1: Illustration of difference between signal duration, relative convergence and convergence time.

5.6. Experiment Setup

In [18] the authors analyzed BGP beacons signal duration time and relative convergence time of three BGP beacons announced and withdrawn from three different ASes. For every beacon both the announcement and the withdrawal times were analyzed. We will rerun the experiments in the simulator trying to recreate an environment similar to those in which the original experiment was conducted.

5.6.1. Runtime Environment

The BGP Simulator is intended to run on a homogeneous cluster. To perform the experiment the Distributed ASCI Supercomputer 3 (DAS3) at Vrije Universiteit Amsterdam was used. This cluster consists of 85 dual-processor dual-core nodes (4 CPU cores per compute node). The validation experiment was executed using 65 nodes—64 compute nodes and 1 coordinator node.

To rerun the experiments we need a network, similar to the on which the reference measurements were made. CAIDA network of AS relationships is used. Unfortunately the original analysis was conducted in 2002-2003 and the oldest available CAIDA network is from January 2004. This network may be considerably bigger (especially with respect to AS's interconnection) than the original one.

The network for January 2004 obtained from CAIDA consists of 16301 ASes. They are distributed randomly between compute nodes, which gives 254 or 255 ASes hosted on each one of them.

5.6.2. Simulation Parameters

The result of each experiment is in fact the combined impact of many affecting factors. Model design, its abstraction level, simulator implementation and language runtime environment (namely JVM) all affect the results obtained by the simulation. Apart from such "hard" factors, there are also many simulation parameters which can be treated as the input of the simulator and have impact on its output. One can distinguish BGP model specific parameters like MRAI timers settings or route flap damping thresholds, and environmental parameters like noise level in the network. Each one of them has its impact and each one of them has to be carefully studied and set in a way that is most compliant with the real-world state at the time of the original experiment. All these parameters define a specific instantiation of the model and with the simulator a specific experiment instance.

Route flap damping At the time of the original BGP Beacons experiment (2003), route flap damping was already considered harmful [4] however convergence time distribution graphs indicate that it was still widely used. In order to recreate the original environment route flap damping has to be present in our simulations.

There is a difference in standard route flap damping settings used by Juniper and Cisco routers, which affects the amount of flaps necessary to suppress a route. We do not have extensive information about the router type distribution and leave the setting of how many percent of routers is Cisco-like as one of the input settings. The distribution of router types among ASes is random.

MRAI MRAI describes the minimal time between consecutive route announcements sent by one peer. In the simulator this timer is implemented on a per-peer rather than on a per-peer-per-prefix basis, just as it is done by router vendors (to our best knowledge). Just like for route flap damping there is a difference between MRAI default values for Cisco and Juniper routers—Cisco uses 30 seconds as default setting, while Juniper routers by default do not suppress advertisements before sending. Again the percentage of routers having 30 seconds timers is left as input setting while distribution among routers is random.

Policies Policies are the way how decisions are made in the BGP network. Unfortunately when using only BGP monitors like RIS or Route Views it is hard to infer the exact policies used by the peers. The easiest possible policy is to always prefer the shortest path, no matter where it came from. This is obviously not the case in the Internet. The policies used in the simulator make decisions based on AS path lengths, type of relation between the sender of the route and the receiver, and some random factor (to avoid ties). Definitely policy decision making in the simulator requires more study in order to reflect real-world situations better.

Network noise level Although BGP is an incremental protocol and updates should be sent only when a new event occurs in the network, due to the protocol nature and network size, there is a considerable amount of constantly exchanged updates. From the perspective of other updates, constant exchange of information can be treated as "network noise" and may have serious impact on how long it takes for new updates to propagate to a stable state. One of the reason is that "noise" triggers the MRAI timers and makes other announcements wait until the MRAI timer expires.

Although updates exchanged between BGP peers in the real networks can be observed by RIPE RIS or Route Views monitors, it is not an easy task to generate events creating a comparable level of noise. The simulator generates noise using very a simple solution: random ASes in the network propagate bogus prefixes every X seconds. Convergence time of these prefixes is not measured—their only purpose is to create noise and make the runtime environment reflect the reality more accurately. The time between consecutive bogus prefix announcements is also an input parameter.

Time scaling As we have stated before the simulation is not being run in real-time but in scaled time. The scaling factor tells how much faster (or slower) the simulation time goes with respect to real time.

It is possible the run the experiment without scaling the time but the resulting simulations would take very long and because of the low load, the cluster usage would not be effective. Therefore the time is scaled such that the experiment can be run much more faster but without having significant impact on the output. Therefore it is very important to find a good balance between good compute node load and the time it takes

to run an experiment. Effectively, time scaling makes the simulator perform exactly the same operations, only faster.

The simulator is software executed on physical machines using physical network for the communication. Both the Java Garbage Collection mechanism as well as access to hardware devices can generate delays in the order of up to few milliseconds. If the time scaling factor would be too big—for example in the order of hundreds—tens milliseconds would expand to a few seconds. Few seconds difference in processing time by single router is not a negligible difference. Because the nodes overloading could have very bad impact on the simulation results there are many performance counters in the simulator, which monitor if the processing delays are inside an acceptable interval. Time scaling factor has to be chosen in such a way that the overloading never occurs. Experiments have shown that depending on the noise level, reasonable scaling factor is between 15 and 50.

The described model parameters have considerable impact on the results obtained by the simulation. Because of the lack of extensive study of these parameters in real life, simple approximations have been used (like random MRAI and route flap damping types distribution among peers), accepting the fact that it makes the results less accurate. The task of better inferring real values and their distribution is left for future research.

5.6.3. External factors

One of the biggest and most important factors affecting accuracy of the simulation is the AS topology. It is important to understand how the network graph can influence obtained results.

Some accuracy is lost already by treating each AS as one router and by ignoring intra-AS connections. This is a deliberate modeling decision made to scale the size of the problem down.

As we have written before we use the network data from CAIDA [5]. Network inferred using algorithms like the one used by CAIDA can miss up to 90% of peering connections [8]. It is not clear to us to what extent lack of peering connections can influence the convergence property of the BGP protocol. Therefore it cannot be assumed that the network used for the experiments is complete and that it truly describes the real BGP network.

Besides the incompleteness of the network, there is another important factor affecting the experiment results: the reference analysis was done at the end of 2002 and at the beginning of 2003 but the first network available at CAIDA is from January 2004. We cannot a priori state to what extent network from January 2004 differs from the network from January 2003. It might be possible to prune some AS connections using different data available at CAIDA to obtain an AS topology more similar to the one from 2003.

We have to be very careful when trying to draw conclusions from the obtained results, because the data used for out simulation is different from the original and we cannot tell to what extent.

5.6.4. Time Measurement

In order to compute the signal duration and relative convergence, the simulator has to be capable of measuring time. All ASes, the compute node itself and the coordinator have access to current simulation (scaled) time. Only the scaled time is used during simulation, not the real-world time.

At the beginning of the simulation each compute node is notified which moment in time is treated as simulation time 0. This is necessary to be sure that the time of the events that occurred on different compute nodes are comparable.

In the reference study the authors did not have access to each router and therefore had to use Route Views to monitor outgoing updates. Although we have direct access to each AS, in the simulator results are collected using Route Views monitors just like they were in the real network. The Route Views monitor is implemented as a BGP router but with a different PrefixStore—instead of real BGP implementation it only saves information about when particular prefix was received.

After the experiment is finished all data from the two Route Views monitors is sent back to the coordinator node. The coordinator processes the data and computes signal duration time and relative convergence time for each prefix on every monitor. Information about bogus prefixes (noise) is not sent back to the coordinator.

5.6.5. Single Experiment Description

We call one run of the whole simulator on a cluster a single experiment. The input of the experiment is the network topology, parameters (e.g., MRAI, time scale). The output is signal duration and relative convergence time data.

After all nodes have been set up and all initial settings have been made, we start introducing beacon prefixes into the network. Each beacon prefix is given enough time to propagate to a state were its interference with other beacons is minimal. Unlike with normal beacons we are not restricted to a few beacon prefixes and therefore for each probe (prefix announcement/withdrawal) we use different prefixes—it makes it easier to monitor the events, as we do not have to use sophisticated techniques to distinguish different input signals for the same beacon.

To monitor withdrawals we have to first propagate prefixes as fast as possible—without taking care of possible interactions. Afterwards we reset their timers (so that we can register the time we first see an update in the withdrawal's output stream) and start withdrawing them one by one.

Just as in the referenced work, we measure signal duration and relative convergence time using data for exactly the same prefixes.

An experiment consists of measurements of 100 introduced and 100 withdrawn prefixes.

5.6.6. Experiment Instance Description

One execution of the simulator, performing an experiment is called an experiment instance. Instance input consists of (i) AS network, (ii) list of events that should be executed by the coordinator, and (iii) model and simulation properties. The output of the simulator depends on the type of the performed experiment—data that needs to be recorded is different for average AS path length measurement than for signal duration time. The simulator output for reruns of the "BGP Beacons" experiment is a list of times at which Route Views monitor has received BGP updates from its neighbors.

AS Network

At the start of the simulation, every compute node reads the AS network graph from a file on the hard drive. The file consists of a list of ASes as well as links between them. Each link has an annotation that says whether it describes a customer-provider or a peer-peer relation as well as its direction. The coordinator does not have to read the whole graph—its only interest is the distribution of ASes between compute nodes, not connections between them.

Events List

The events that have to be introduced into the network are stored in a separate file. This file is read only by the coordinator, the entity responsible for introducing new events into the network. An event is an arbitrary object that will be executed by the coordinator at a scheduled time. There are various types of events:

AnnounceEvent This event describes which AS should start announcing (or withdrawing) a list of given prefixes at a given time. The coordinator delivers this information to the appropriate AS and this AS starts sending information about the prefix to its neighbors. Effectively, AnnounceEvent is a way to introduce events into the AS network.

NoiseEvent An event describing that the coordinator should start introducing noise into the system (by triggering bogus prefix announcements).

LastSeenEvent Processing this event results in a request to the Route Views monitors to send back information about arrival times of updates for prefixes described in the event. When the information is received it is stored on the hard drive. Effectively this event saves the output of the simulator in disk files.

ResetDataEvent Executing this event results in resetting data stored by Route Views monitors. This is necessary for withdrawals' signal duration and convergence time measurement (after the announcement converges to steady state, before the withdrawal input signal is introduced).

Events executed by the coordinator describe one full simulation scenario and determine the whole experiment. At the programming level an event is a simple Java object implementing the Event interface. For different types of experiments different types of events may be necessary—they can be easily created by providing different implementations of the Event interface.

Properties

Many model as well as simulation settings are externalized from the code. This means they are not part of the simulator's source code but rather are its input. Such parameters include for example MRAI timers (model) or noise level (simulation) settings. Changing the property value can have dramatic impact on the simulation results (as will be shown in the next chapter). Important properties that create a model instance are as follows (the meaning of the parameters is described in 5.6.2):

timeScaler Describes how much faster the simulation time should go with respect to the real time. A timeScaler value n means that with every second the simulation time advances by n seconds.

mraiProc Says how many percent of the ASes should have MRAI timers. Setting this value to 70 means that 70% of ASes will have a MRAI timer with 30 seconds threshold (hosts are picked on random basis).

flapPercentage Says how many percent of the ASes should have the route flap damping technique turned on.

flapDistribution Says how many percent of the ASes, that have route flap damping technique turned on, should have Cisco-like behavior. The rest of the ASes will have Juniper-like behavior.

iBGPmaxValue and iBGPMaxNeighbors Those parameters describe the values to be used when computing the delay caused by iBGP.

noiseSleepTime Describes the time (in milliseconds) between consecutive bogus prefixes announcements.

Each experiment instance input consists of an AS network topology, an event list and properties settings. Based on this input the simulator generates the output (experiment results).

Output

The simulation output of the "BGP Beacons" experiment is stored in a file of the following structure:

monitorName; peerASId; prefix; firstSeen; lastSeen; totalSeenTimes

Each line says that prefix "prefix" from "peerASId" was first time seen at time "firstSeen" and last time seen at time "lastSeen" by monitor "monitorName". Between "firstSeen" and "lastSeen", "prefix" was seen "totalSeenTimes" times. Such data is sufficient to study the distribution of both signal duration and relative convergence times.

Output generation is not included in simulator's core source code but is a result of processing of one of the events—LastSeenEvent for "BGP Beacons" experiment—by the coordinator.

5.7. Summary

In order to show validity of the simulations, a validation experiment was performed. Mao et al. [18] was used as reference experiment and the simulator setup and configuration was described.

In the next chapter the results of the simulation will be compared with results obtained from real-world beacons observation. Similarities will be shown and some explanations will be given for the spotted discrepancies.

Chapter 6

Results Discussion

Many experiments were performed using the setup and different possible parameter values presented in the last chapter. In this chapter, results of the calibration will be presented and compared with the anticipated outcome. Impact of the input parameters on obtained results will be studied and one example of possible "what-if" analysis will be presented.

6.1. Experimental Setup

6.1.1. Graphical Data Representation

Mao et al. [18] used Cumulative Distribution Function (CDF) graphs to present the results of their analysis. The vast majority of observed signal duration and relative convergence time values fall into a 0–180 seconds interval and the authors have therefore decided to plot CDF only for values falling into this interval. In order to be consistent with their representation, we will present the simulation results using CDF as well, showing only the 0-180 seconds interval.

Besides cumulative distribution graphs, signal duration time histograms will be presented. Reason for this is that there is a significant discrepancy in the results presented by Mao et al. using histograms and CDF in their "BGP Beacons" paper. Their histograms match our results and expectations, unlike their CDF graphs.

6.1.2. Beacons

In the reference experiment three BGP beacons were studied.

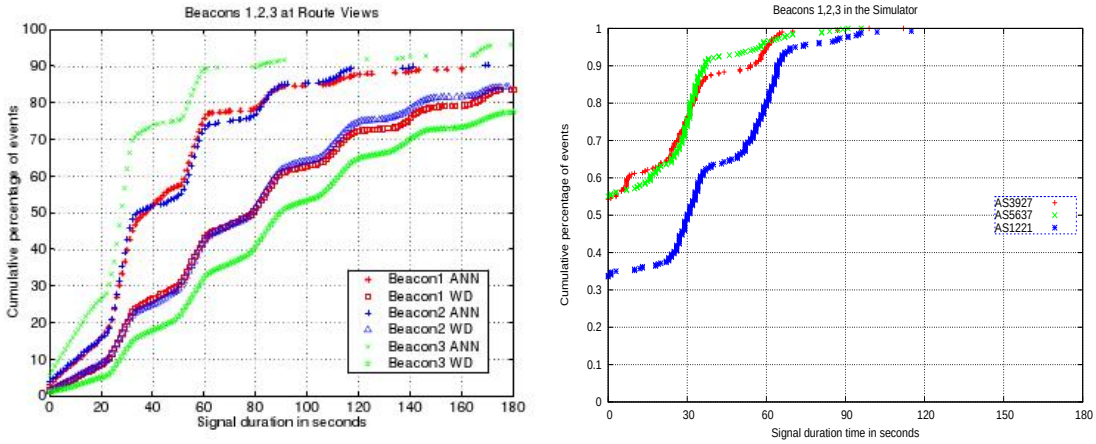
- Beacon 1: 198.133.206.0/24 maintained by Randy Bush, introduced by AS3927, having AS2914 and AS1239 as Tier-1 upstreams
- Beacon 2: 192.135.183.0/24 maintained by David Meyer, introduced by AS5637, having AS3701 and AS2914 as Tier-1 upstreams
- Beacon 3: 203.10.63.0/24 maintained by Geoff Huston, introduced by AS1221, having AS1221 as Tier-1 upstream[1]

It is very important to note that the connectivity of Beacon 1's host was changing during and after the experiment was conducted. Additionally, because the simplified network used in our experiment is about one year younger than the one used in original experiment, Beacon 1 has different connectivity and results for this beacon may vary strongly from the one observed in 2002–2003. Nevertheless, we are much more interested in trends than in exact values and studying Beacon 1's behavior can still be very useful.

[1] AS1221—Telstra Internet, is Tier-1 provider itself

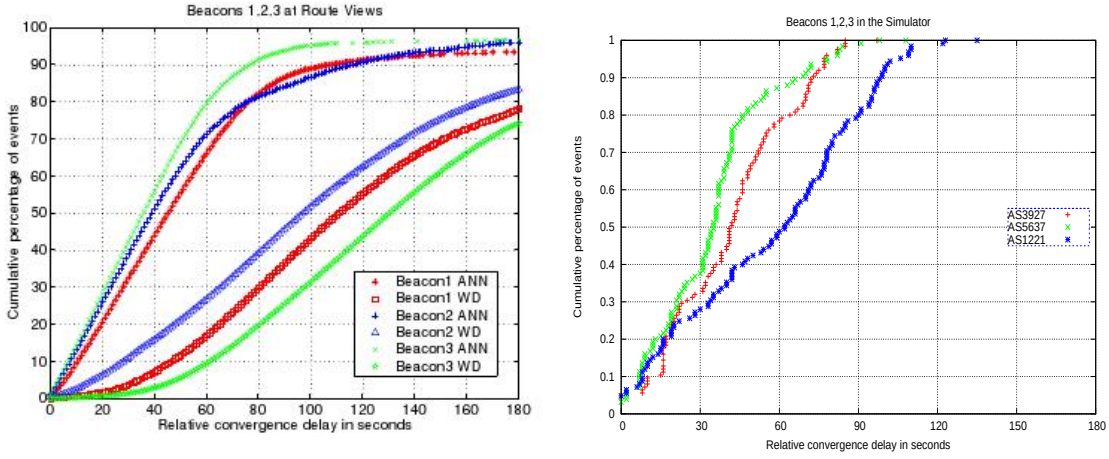
6.2. Obtained Results

6.2.1. Simulated Behavior



(a) (b)

Figure 6.1: Cumulative distribution of the signal duration for all three beacons from the reference study [18] (a) and from the simulator (b).XXX



(a) (b)

Figure 6.2: Cumulative distribution of the relative convergence time for all three beacons from the reference study [18] (a) and from the simulator (b).

Figures 6.1a and 6.2a present results obtained by Mao et al. [18]. We want to concentrate on the announcement distribution, especially on signal duration time. One interesting thing are repeated sudden "jumps" every 30 seconds. Those jumps are normally attributed to MRAI timers—time between first and last update received from the same peer is often a multiple of 30 seconds. What is also very distinctive and intriguing is the very small percentage of signals with 0 seconds duration.

Figures 6.1b and 6.2b present our simulator's output for announcements for all three beacons. Although the curves shape is very similar, one significant difference can be observed—the percentage of signals with 0 seconds duration from the simulators results is much bigger

(between 30% and 60%) than the one observed in the reference experiment (less than 10%). In both cases large percentage of values is a multiple of 30 seconds—this is the influence of MRAI timers.

6.2.2. Parameters Sensitivity

As described in Section 5.6.2 each model instance is made out of input parameters such as MRAI timers percentage, route flap damping settings and so on. Those parameters influence the simulator's output and it is interesting to study to what extent the simulator's output changes for different input settings.

MRAI

The percentage of routers with MRAI timers bigger than 0 influences the simulator output in two ways:

Shorter signal duration times The less routers have MRAI timers set, the less updates are suppressed for 30 seconds and signal duration time decreases.

30 seconds intervals The percentage of routers having MRAI timers set changes the percentage of signals with duration that is a multiple of 30 seconds. The more MRAI timers, the bigger the "jumps" observed on the graphs. In fact, if every router has MRAI timer set almost every signal duration time is a multiple of 30 seconds, and the CDF graph is no longer continuous—consists of points every plotted every 30 seconds.

Noise Level

Constant update exchange between peers influence the MRAI impact. When a bogus prefix is propagated to a neighbor, the MRAI timer suppresses the next updates to the same peer for 30 seconds. It is therefore possible that the first update, that should be propagated to Route Views monitor, gets suppressed and before the timer expires a new, better update is received. The first update does not get propagated and Route Views monitor receives only one update for a given beacon. Signal length decreases from 1 to 0 and signal duration from 30 seconds to 0 seconds.

In the observed results most updates have a signal length of either 1 or 2, and a higher noise level significantly increases the percentage of signals with duration time 0.

Route Flap Damping

Route flap damping is considered to be responsible for extremely long signals duration time [18]. This is a result of suppressing a route when it is falsely considered to be flapping. Route Flap Damping technique was given as main reason for the presence of not negligible percentage of signals that have a duration time longer than 180 seconds in [18] (as can be observed on Figure 6.1a).

Route flap damping is implemented in our model and the percentage of routers using this technique is one of the simulation parameters. Although many routes get damped, we did not observe a significant influence of route flap damping on the signal duration. This is caused by the fact that it is very rare (in the simulator) that the route that gets undamped is propagated globally. In most cases, the neighboring routers do not consider the newly received undamped route to be better than the currently installed one.

iBGP Convergence Time

Normally an AS consists of more than one router. iBGP is used to exchange routing information between routers being in control of the same AS. The proposed BGP model treats each AS as a single router, without taking its internal network into account. Because the size of the AS network might have an impact on the iBGP convergence time, the parameter was externalized and is treated as simulator input.

iBGP convergence time has significant impact on the signal duration time and this impact depends on the beacon: large ASes (with more than 700 neighbors) have longer iBGP convergence time and if they are often on the best AS path, such paths are may not be found fast enough. Longer paths, but traversing only smaller ASes can be found faster and more updates have to be generated before the network converges to the final state. Depending on the path explored different prefixes can "suffer" longer or smaller delay in founding the best AS path.

iBGP convergence time seems to play an important role in the simulation results—although it does not significantly change the CDF curves shapes, it has big impact on the exact values. It is therefore necessary to study iBGP convergence in the real BGP network and its impact on the simulator as it might improve the simulator accuracy.

6.2.3. Parameters Used

Many different parameter settings have been tested. They were evaluated both by looking at simulator behavior and output, as well as by its sanity. Many different configurations generate reasonable output. The results of the experiments differ, but all show the same characteristics. The parameters used are:

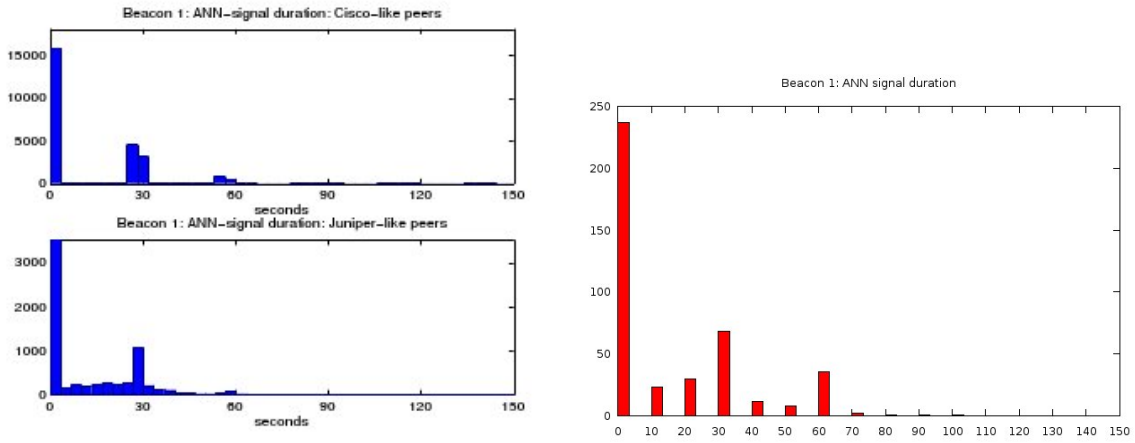
- 65% of routers having MRAI timers.
- 10 seconds of iBGP convergence time for 1500 neighbors with logarithmic distribution: $iBGPConvergence(asSize) = log_{(1500^{1/10})} asSize$ seconds.
- 65 percent of hosts having route flap damping, 80% of them being Cisco-like.
- Bogus prefix announcement introduced once every 121 seconds (by a random AS).

It has to be stressed, that these parameters are a result of the calibration but it does not mean that they perfectly reflect the situation in the real BGP network.

For real BGP beacons a two hour pause is used to assure that two consecutive beacon updates do not interfere with each other. To achieve better result reproducibility, a two hour pause is used in the simulator as well. In fact, the observed simulator behavior is extremely stable and the CDF graphs for e.g., 30 and 200 prefixes do not differ significantly. This is a very important simulator property, as it shows that although the execution order is highly non-deterministic, it does not have that much impact on the results—it shows that using emulation instead of discrete-event simulation does not introduce too much noise into the results.

6.3. Results Comparison

The obtained results exhibit both similarities as well as differences to the ones observed in the real BGP network. Some of these differences can be attributed to a discrepancy found in the "BGP Beacons" paper.



(a) (b)

Figure 6.3: Beacon 1's announcement signal duration distribution for Cisco-like and Juniper-like peers as observed by Mao et al. (a) and as observed by the simulator (b).

6.3.1. "BGP Beacons" Experiment Discrepancy

Careful study of the "BGP Beacons" paper has been done and a discrepancy has been found. The CDF plot for signal duration time (see Figure 6.1a on page 42) does not match with the histogram for Beacon 1 (see Figure 6.3a). The histogram was meant to present the difference between signal duration time distribution between Cisco-like and Juniper-like peers. It shows a very large percentage of signals with duration time 0, small percentage of signals with duration time 60, and very small percentage of longer signals. This results clearly contradict the results presented on Figure 6.1a, where very few signals have a duration time 0 and more than 10% have a duration time above 60 seconds. In fact the CDF graph generated by the simulator presented on Figure 6.1b matches the histograms much better.

We do not know the reason why the results presented in the histograms do not match those presented in the CDF graphs and we do not know which result is actually correct.

6.3.2. Similar and Different Characteristics

The characteristic that clearly matches in the simulator and in the real "BGP Beacons" observation is the 30 second interval in relative signal duration. This interval is visible both on histograms and CDF graph in the reference paper, as well as in the simulated results. Furthermore, these "jumps" are visible also when input parameters are changed—they disappear only when the amount of routers without MRAI timers becomes very low[2].

The biggest difference is the percentage of signals with duration time 0. There is a discrepancy in the reference paper with respect to this characteristic and it is therefore unclear which behavior is the actually observed. There is a need to perform experiment similar to the one described in [18] to assess the factual state.

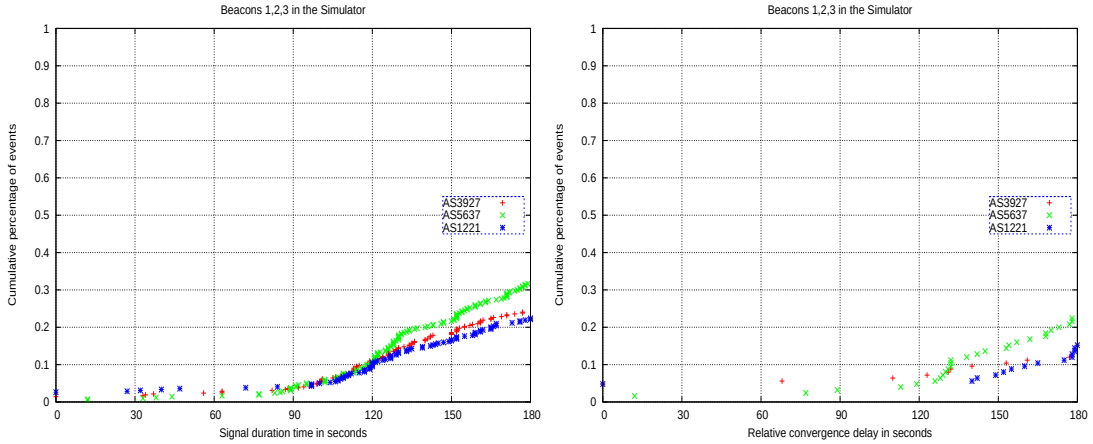
Another characteristic that is different in the results of the simulations and in the paper is the relation between prefixes. In [18] it was Beacon 3 that had the shortest signal duration time and in simulator results it is Beacon 3 to have the longest signal duration. It has to be reminded that Beacon 1's connectivity has been changing during and after the experiment, and the network used for simulations is in fact a year later. We do not know whether this difference is implied by wrong BGP modeling, incorrect input parameter settings, external

[2]The signal duration time converges to 0 then, because consecutive updates come extremely fast.

factors like available BGP network, or an error in the reference paper. Once again, redoing the original analysis of the "BGP Beacons" could give answer to this question and lead to improvements of the behavior of the simulator.

Although signal duration time characteristics seem to be different (especially with respect to percentage of signals with duration time 0), relative convergence time distribution (see Figure 6.2a on page 42 and Figure 6.2b on page 42) are matching. This is surprising as it implies that ASes in the simulator see the updates arriving for the same time duration. It is therefore strange that they see updates arriving with the same distribution but generate different output signal—the observed signal duration is different in the reference paper and in the simulation. This is yet another premise that in fact the CDF graph (Figure 6.1a on page 42) presented by Mao el al. is erroneous.

6.3.3. Withdrawals Propagation



(a) (b)

Figure 6.4: Cumulative distribution of signal duration (a) and relative convergence time (b) for withdrawal signals for all three beacons as observed by the simulator.

In rerunning the "BGP Beacons" experiment we have concentrated ourselves primarily on announcements study. The main reason for doing so is that running experiments with withdrawals requires much more time and finding appropriate input parameters is therefore more difficult. Nevertheless, withdrawals experiments were conducted as well (see Figure 6.1b on page 42 and Figure 6.2b on page 42). Clearly the results for both signal duration and relative convergence time do not match those observed by Mao et al. in [18]. The reason for it is extreme BGP path hunting (as described in [14]) observed during the simulations.

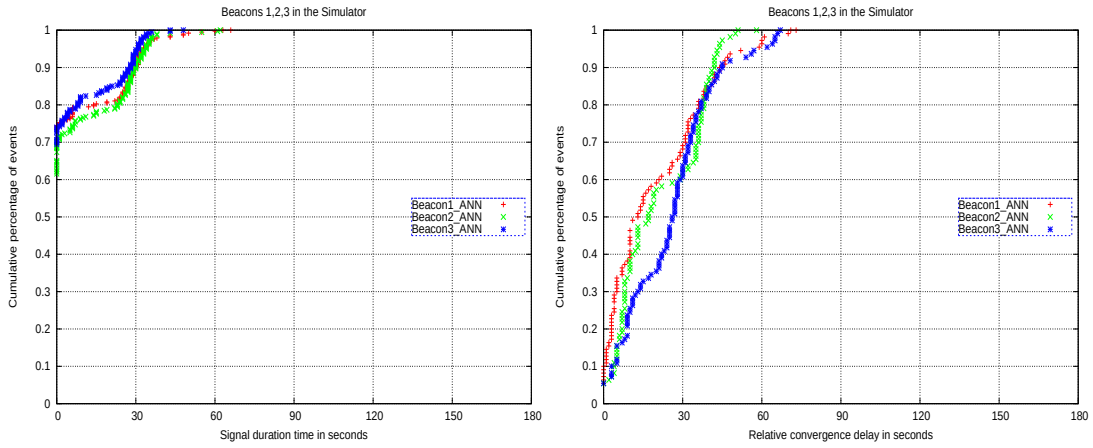
BGP path hunting (aka BGP path exploration) is a well known phenomena. It can occur both during prefix announcement and withdrawal but is more visible during the latter. When a node receives a withdrawal it invalidates the route and chooses a different one from the RIB (each router stores last route received for a given prefix from its peers). The new route is then propagated to the neighbors, but after some time this route gets invalidated as well; another route starts to be propagated and so on, until all paths get explored or withdrawn. In theory path hunting can lead to an extreme amount of BGP updates exchanged (up to $n!$ in full graph). It has been studied that, in real BGP deployments, scale of the phenomena is much not that significant—adding on average only 2 or 3 updates to a prefix withdrawal [14] output signal.

Severe path hunting can be observed in the simulator when a withdrawal is being sent. While normally average AS path length is about 4 to 5, during path hunting the AS path length can get as long as 30. This is definitely more than observed in real BGP network and we do not have a definitive answer why path exploration occurs on such a large scale in the simulated network. There is definitely a need for further analysis of this problem as in fact path exploration during withdrawals might indicate wrong propagation of announcements as well.

6.4. What-if Analysis

One of the main reasons for modeling and simulating BGP is "what-if" analysis. Because the simulator still needs calibration, more validation experiments and its input parameters need to be researched, results of such analysis may not be creditable. Nevertheless it is interesting to see how the simulator behaves in a different environment.

To show the idea of "what-if" analysis, the BGP beacons experiment was performed on a bigger AS network. Besides the network everything is the same: beacons, beacon hosts, observation points, etc. The only difference is the number of the ASes and their interconnections. The CAIDA network from January 2008 was used instead of the network from January 2004. The 2004 network contained only about 17,000 ASes where 2008 network contains more than 24,000 ASes.



(a) (b)

Figure 6.5: Cumulative distribution of signal duration (a) and relative convergence time (b) for announcement signals for all three beacons for the network from January 2008 as observed by the simulator.

Figure 6.5a shows signal duration and relative convergence time distribution for all three beacons announcements. Surprisingly, the average signal duration decreased significantly and so did relative convergence time. Also the percentage of ASes seeing only one update (signal length 0) is higher.

One might expect BGP to converge slower in bigger networks but there is another big difference in BGP networks from 2004 and 2008—its density. It seems that BGP is converging much better in denser networks and network density is more important than its size. There is no doubt that this property should be further researched and such research is a perfect application of our model.

Of course Route Views monitors is not the only way one can study BGP convergence.

While in the real network access to BGP routers is highly restricted, during the simulation it is possible to analyze how the propagation behaves "step by step" and find out how the decisions are really made. Such close look at BGP can greatly contribute to the understanding of the protocol's behavior.

Chapter 7

Conclusions

In this thesis we have presented a heavily abstracted BGP model and its implementation—a large scale BGP simulation. A validation experiment has been presented to show the accuracy of the model and its implementation. Although the experiment has shown some discrepancies between observed real-world and simulated behavior, many similarities were found. The model reflects many BGP characteristics and results obtained with the simulator match with observed phenomena (e.g., periodicity of signal duration connected to MRAI timers length).

7.1. Impact of This Project

7.1.1. Proof of Concept

One of the most remarkable things shown in this project is a new approach to BGP simulation. In contrast to other BGP simulators, BGP was heavily abstracted and many BGP parameters were omitted. This allowed to shrink the problem size (with respect to the number of participating entities) and made computation feasible. Of course, every decision whether a part of the protocol or the network is essential might introduce accuracy loss. Now when it is clear that large-scale BGP simulation is feasible, one can go back and re-examine the decisions made for the model, to improve the simulator's accuracy.

Another important and new contribution in the field of BGP simulation is using emulation instead of event-driven simulation. This allows to build a truly scalable and efficient simulator capable of simulating tens of thousands of ASes with very short single prefix propagation time. This project shows that BGP can be simulated in a distributed manner and opens doors to even more distributed simulations: on heterogeneous cluster, spanning many locations with thousands of cores and terabytes of memory available.

7.1.2. Real-world Resemblance

During the validation experiment many model properties (like MRAI timers, iBGP convergence time settings) have been studied. In most cases the impact of single properties on simulator's results was just as anticipated. There are few reasons why we could not show full similarity of the real-world observations and results of the simulator. The reason are both practical and model limitations such as: not deeply enough studied model parameters' real-world values, different BGP network or too abstracted protocol parts, and lack of consistency of results presented in the reference work. Nevertheless, the type of distribution observed in the simulation results is similar to the one observed by Mao et al. in [18]. This proves that although the simulator still needs a lot of calibration and validation, the overall properties of BGP dynamic behavior match.

7.2. Future Research

7.2.1. Validation and Calibration

Rerunning the "BGP Beacons" experiment has shown discrepancies between experiment results and those observed in the real-world. There is still need for further simulator calibration in order to improve the resemblance to a real BGP network behavior. Besides that, the experiment covers only a small part of the BGP protocol and simulator capabilities. Many more things can be studied, like for instance: total signal duration time, impact of different updates crawling the network on each other, behavior of the network in the presence of node failures and so on. Nevertheless, the simulator is capable of running the experiments suggested in Chapter 1: showing how new (and old) techniques influence BGP dynamic behavior.

The reference "BGP Beacons" experiment was conducted more than five years ago. The Internet is growing fast and has changed very much during these years. It might be helpful to conduct a similar BGP beacons study on today's network and compare that with the results of the simulator.

7.2.2. Model Properties Study

During the model design, few parts of the BGP protocol or network were left as model parameters—treated as simulator input. It has been shown in the previous chapter, that these parameters have very big impact on the observed behavior. They have been modeled using either very simple random-based distributions or simple mathematic functions. There is a strong need to research these properties values and distribution in order to provide better input for the simulator. If the simulator's input was more creditable its results would also be more creditable.

Unfortunately, it is hard to infer the exact value of many of these parameters (like for example routers having MRAI timers and exact values of these timers). Probably many of them can be inferred by studying Route Views [30] or RIPE RIS [29] monitors data. However, this will not be a basic task and requires serious research.

7.2.3. Possible Usage

Right now, BGP simulator still requires a lot of investigation in order to be able to give creditable answers to questions about BGP dynamic behavior. After extensive calibration effort, model properties study and implementation verification, the simulator can become a very useful tool for predicting the future BGP behavior with respect to many factors, like for example: network interconnectivity and size growth, policy changes, protocol changes and statistical properties on the overall behavior.

The simulator is capable of simulating networks much bigger than today's Internet, regarding both prefix table size and AS network size. It scales very well when more hardware resources are available.

One specific difference between real-world observation and simulations is the ability to study the exact behavior of the latter. It is possible to study the behavior of every node separately, which can lead to a better understanding of the interaction between peers in BGP. One can "play" with the simulator and see how various settings and implementations influence the nodes, the protocol and the whole system behavior. To our best knowledge no other tools exist that can facilitate such large-scale studies.

We think that with this project we made an significant contribution in the BGP research field. The presented model and its implementation need a lot of improvements but already shows much resemblance with the real-world behavior. Although the results cannot be treated as final, they give a direction for future studies which will improve the simulators behavior.

We hope that further development will lead to a tool capable of answering serious BGP-related questions and in the end—to protocol improvement.

Bibliography

[1] Y. Bar-Yam. *Dynamics of Complex Systems*. Studies in Nonlinearity. Addison-Wesley, Reading, MA, 1997.

[2] BGP++: A C++ implementation of BGP for ns-2 and GTNetS network simulators. `http://www.ece.gatech.edu/research/labs/MANIACS/BGP++/`.

[3] S. Bürkle. BGP convergence analysis. Master's thesis, Universität des Saarlandes, June 2003.

[4] R. Bush, T. Griffin, and Z. M. Mao. Route flap damping: Harmful? `www.nanog.org/mtg-0210/ppt/flap.pdf`, Oct. 2002.

[5] CAIDA: Cooperative Association for Internet Data Analysis. `http://www.caida.org/home/`.

[6] C-BGP: Efficient solver for BGP. `http://cbgp.info.ucl.ac.be/`.

[7] J. Chandrashekar, Z. Duan, Z.-L. Zhang, and J. Krasky. Limiting path exploration in BGP. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, Miami, FL, Mar. 2005.

[8] X. Dimitropoulos, D. Krioukov, M. Fomenkov, B. Huffaker, Y. Hyun, kc claffy, and G. Riley. AS relationships: Inference and validation. *ACM SIGCOMM Computer Communication Review*, 37(1):29–40, Jan. 2007.

[9] X. A. Dimitropoulos and G. F. Riley. Creating realistic BGP models. In *Proceedings of the 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer Telecommunications Systems (MASCOTS 2003)*, pages 64–70, Orlando, FL, Oct. 2003.

[10] N. Feamster, H. Balakrishnan, and J. Rexford. Some foundational problems in interdomain routing. In *Proceedings of the 3rd Workshop on Hot Topics in Networks (HotNets-III)*, San Diego, CA, Nov. 2004.

[11] T. G. Griffin and B. J. Premore. An experimental analysis of BGP convergence time. In *Proceedings of the 9th International Conference on Network Protocols (ICNP'01)*, pages 53–61, Riverside, CA, Nov. 2001.

[12] J. Hawkinson and T. Bates. Guidelines for creation, selection, and registration of an Autonomous System (AS). RFC 1930 (Best Current Practice), Mar. 1996.

[13] G. Huston. Analyzing the Internet's BGP routing table. *Internet Protocol Journal*, 4(1):2–15, Mar. 2001.

[14] G. Huston. BGP stability improvements. `http://tools.ietf.org/html/draft-li-bgp-stability-01`, June 2007. (work in progress).

[15] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet routing convergence. *IEEE/ACM Transactions on Networking*, 9(3):293–306, June 2001.

[16] C. Labovitz, G. R. Malan, and F. Jahanian. Internet routing instability. In *Proceedings of the ACM SIGCOMM '97 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 115–126, Cannes, France, Sept. 1997.

[17] C. Labovitz, G. R. Malan, and F. Jahanian. Origins of Internet routing instability. In *INFOCOM*, pages 218–226, 1999.

[18] Z. M. Mao, R. Bush, T. G. Griffin, and M. Roughan. BGP beacons. In *IMC '03: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, pages 1–14, New York, NY, USA, 2003. ACM.

[19] Z. M. Mao, R. Govindan, G. Varghese, and R. H. Katz. Route flap damping exacerbates Internet routing convergence. *SIGCOMM Comput. Commun. Rev.*, 32(4):221–233, 2002.

[20] T. Naylor and J. Finger. Verification of computer simulation models. *Management Science*, 14(2):92–101, 1967.

[21] The Network Simulator NS-2. `http://www.isi.edu/nsnam/ns/`.

[22] D. Pei, M. Azuma, D. Massey, and L. Zhang. BGP-RCN: improving BGP convergence through root cause notification. *Comput. Netw. ISDN Syst.*, 48(2):175–194, 205.

[23] BGP Routing Table Analysis Reports. `http://bgp.potaroo.net/`.

[24] B. J. Premore. *An analysis of convergence properties of the border gateway protocol using discrete event simulation.* PhD thesis, Dartmouth College, Hanover, NH, USA, 2003.

[25] B. Quoitin, L. Iannone, C. de Launois, and O. Bonaventure. Evaluating the benefits of the locator/identifier separation. In *Proceedings of MobiArch (ACM SIGCOMM Workshop)*, Kyoto, Japan, August 2007.

[26] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271 (Draft Standard), Jan. 2006.

[27] J. Rexford, J. Wang, Z. Xiao, and Y. Zhang. BGP routing stability fo popular destinations. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurement*, pages 197–202, Marseille, France, Nov. 2002.

[28] RIPE Routing Working Group Recommendations on Route-flap Damping. `http://www.ripe.net/docs/routeflap-damping.html`, 2006.

[29] RIPE NCC projects: RIS—Routing Information Service. `http://www.ripe.net/ris/`.

[30] University of Oregon Route Views Project. `http://www.routeviews.org/`.

[31] T. Saydam. Process-oriented simulation languages. *SIGSIM Simul. Dig.*, 16(2):8–13, 1985.

[32] Scalable Simulation Framework. `http://www.ssfnet.org/homePage.html`.

[33] I. van Beijnum. *BGP.* O'Reilly, Sebastopol, CA, USA, 2002.

[34] GNU Zebra: Free routing software distributed under GNU General Public License. `http://www.zebra.org/`.