Internet Topology Generation for Large Scale BGP Simulation *

J.M. Fourneau, H. Yahiaoui

PRiSM, 45 Avenue des États-Unis. 78035 Versailles, FRANCE
E-mail: `jmf@prism.uvsq.fr, yaho@prism.uvsq.fr`

Abstract

It is now well known that the Internet inter-domain routing protocol (BGP) suffers from multiple problems of safety: first, reliability because of its reactions to links or routers breakdowns, reliability also when incoherent local policies obstruct the convergence of the routing algorithm or flood the network with obsolete update messages. More recently, BGP routers breakdowns during some worms propagation have also demonstrated a safety problem. We study new versions of this protocol which takes into account some information added in the protocol, some measurements and tomography. We develop a large scale simulation model to study how few smart routers may help. We also design a tool to build large graphs from Internet maps to represent the topology.

1 Introduction

The main issue of the Internet routing protocols is to provide worldwide connectivity. The Internet connects thousands of Autonomous Systems (AS) operated by various institutions such as Internet service providers, universities, companies or organisations. Within an AS, routing is controlled by intra-domain protocols such as OSPF or RIP. The Autonomous Systems are connected by dedicated links or public network access points. They exchange information about routes using the Border Gateway Protocol (BGP) [22]. BGP is an inter-domain routing protocol that allows AS to apply local policies for the route selection and the propagation of control information. Such a hierarchical routing follows the hierarchical and distributed nature of the Internet. But, the routing protocols have not been designed to manage the large number of hosts we now have in Internet. The network is fault tolerant as most of its functions are distributed. However, new types of faults like worm propagation, distributed denial of service and attacks of the DNS servers have shown that the network is much less reliable and secure that we may have thought. Furthermore, the lack of guarantee prohibits most of the value added services.

Clearly, a hierarchy of routing protocols is necessary because Internet is a collection of smaller networks with their own rules. And the inter AS routing protocol has a large impact on delays experienced by packets, due to the routes selected. The main property for a routing algorithm on a large network is to deal with the dynamics of the topology, the available links and the connected routers. As the whole Internet is dynamic, the routing tables must change frequently. BGP routers must inform their neighbours when paths change due to a failure or a correction. And they must propagate this information when they receive it. Thus the routing protocols have to manage these updates efficiently. Many authors (for instance, Govindan [9], Labovitz [15], [16]) have reported BGP oscillations for routes. Labovitz et al. report in [15] that the number of exchange messages is several orders of magnitude larger than the number of real faults. These oscillations imply a considerable loss of bandwidth but also larger delays. Furthermore, it is still possible to have an infinite loop, and BGP does not converge if the AS use general rules of routing (see Govindan et al. [21]). Of course the TTL, timers and retransmit techniques will insure that packets never stay too long in the network and that they will be sent again. We advocate that this bandwidth lost will have been better used by a more efficient routing protocol and smart routers.

Thus, BGP suffers typically from a reliability problem since an elementary breakdown of router causes serious transient problems. But BGP and the routers also exhibit safety problems which threaten Internet connectivity. The propagation of worms using random IP addresses recently caused important breakdowns of BGP routers [25][26]. Indeed the random IP address is more likely in cache (due to the size of the BGP table) and the router is stressed by a large number of cache misses and crashes.

1.1 Motivation

BGP thus suffers from multiple problems of safety: first, reliability by its reactions to links or routers breakdowns, reliability also when incoherent local policies obstruct the convergence of the routing algorithm [21] or causes an overload of the network because of the flooding by obsolete update messages, and also safety when attacks on other components of the network stress the router and causes a breakdown. Note that IP spoofing technique may also be used to provide wrong information to a BGP router as the routes are advertised by TCP packets. Classical cryptographic techniques have already been proposed to make secure the exchange of informations [14].

Modeling Internet and its protocols belongs to the most challenging simulation problem we must address. BGP is a complex protocol due to the large number of freedom given to the operators. Furthermore, Internet topology has a large influence on the performance of routing algorithms and new problems related to malicious usage to the network (worms, virus, distributed denial of service) are more and more frequent. The purpose of project SR2I is to improve the reliability of the routing infrastructure. To achieve this goal, we will study how to improve the routing and flooding algorithms, design correct local policies and perform statistical controls to detect traffic variations. We will show how the monitoring of the network may help to understand the real topology and load. Active and passive tomography will recover more information about the network, These techniques can also help to limit the propagation of worms and maybe the Distributed Denial of Service. The main idea is the construction of a "more realistic" image of the networks and the routes to limit the exchanges of messages, accelerate convergence and improve BGP and routers robustness. To fully understand the problems and how to solve them, we must check on a large scale new versions for the protocol or new constraints on the topology or the configuration. And simulation is the only tool available for analysis of protocols on a large scale.

For instance, several extensions of BGP have been proposed recently to improve the stability and the safety. Most of the researches about BGP focused on the *stability* of the protocol. Assume that a computer network is started from an initial coherent state (where known paths to the destination do exist, where routers are properly configured, etc.), then a particular BGP instance (*i.e.* a particular route policy configuration of BGP in the network) is stable if it reaches a final global state where all nodes have a stable path toward the destination. Unfortunately, knowing if a particular instance of BGP is stable is an NP-complete problem (see [10]). Thus, proposals to guarantee stability include a global sufficient condition on the system (see [12]), a local condition on the BGP routers' policies (see [8]), and a dynamic additional algorithm that is run on each BGP router (The Safe Path Vector Protocol as defined in [11]).

Recently, several authors have tried to apply the *self-stabilizing* property to BGP [1]. This property means the following: assume that a computer network is started from an arbitrary initial configurations, (where known paths can be arbitrary and where routers can be completely misconfigured), then a self-stabilizing BGP solution guarantees, as soon as faults cease, that the network eventually reaches an initial coherent state, and then a stable final configuration, assuming a stable BGP is run. To the best of our knowledge, only two self-stabilizing versions of the BGP protocol exist. The first one, [1], is a self-stabilizing version of the Safe Path Vector Protocol of [11]; it assumes realistic communications between nodes, and provides all available routes, but and has high memory requirements. In contrast, [2] has smaller memory requirements than [1], but uses a theoretical reliable communication model, and may removes routes that fit all policies of the system. One important issue would be to design a self-stabilizing version of BGP that is realistic (*i.e.* implementable) and that does not have high resources consumption.

Another approach consists to study the self-stabilizing properties of the BGP protocol itself. Previous approaches to guarantee BGP stability provide sufficient conditions on the acceptable routing policies. If such conditions are satisfied, the resulting protocol is stable (*i.e.* starting from an initial well known configuration, it provides a loop free route to every autonomous system). Since BGP is designed to accept the arrival and departure of autonomous systems dynamically (along with their corresponding routing policies), it should be able to start from an arbitrary configuration (resulting from a brutal modification of the autonomous systems graph topology). It is still unclear whether the sufficient conditions that have been provided so far for stability are also sufficient conditions for self-stabilization. In a high level communication model, at least two approaches (see [12, 8]) proved to be self-stabilizing. Further investigation is needed to characterize the minimal possible sufficient condition for stability and self-stabilization.

However, none of the currently proposed self-stabilizing versions of BGP has been tested in a realistic environment (*e.g.* in a large simulator). It is needed to run benchmarks to analyze and compare non-stabilizing and self-stabilizing versions of BGP according to several criteria: fault-tolerance, speed of convergence toward a loop-free routing, memory consumption. Our project is a part of a large collaborative research action with some specialists of *self-stabilizing* theory to provide them a testbed to check on a large scale new extensions of the routing algorithm.

But we also want to check more practical problems which are not completely related to BGP. First we must have an accurate representation of the AS topology because it has a large influence on performance and also on instability and message flooding. Furthermore, BGP uses TCP to connect the peers. Thus BGP suffers the same congestion problem as TCP. When the network is saturated because of usual congestion or distributed denial of service attacks, the BGP packets may experience the same problems as data packets. As BGP has its own timer to detect that a link is dead, missing packets due to congestion can create new withdraw message which use the network and increase the congestion. Clearly, we may have some domino effect. Note that the correlation between some worms propagation and BGP instability which has been reported before [26] may also be related to this load and congestion problem. Thus the simulation model must carry some information on the load of the network.

Thus we want to study how to improve BGP and the routers. The main idea is to use few "smart" or "intelligent" routers which will help all the AS. Actual routers are rather dumb and it is not that difficult to increase their processing power. This new routers will perform some active and passive measurements and use tools based on tomography to understand the network topology. Even if they do not fully understand everything, we hope that they can give and propagate some useful control informations. For instance, actual BGP implementations use a withdraw message which states that a path is now missing. But it does not explain which links has disappeared. Thus we can select a new path which also uses the dead link and this new path will be canceled in the future. A smarter router can add to the withdraw message more information on the links. This is consistent with today BGP protocol as it is assumed that we can add information after a path and that this information is propagated with the path even if the router is dumb.

The main problem is to understand how the network will behave if few routers implement this more efficient implementation while most of them do not. Again the only possible solution is to build a large scale model of the network and to use some simulations.

1.2 BGP Simulators and Internet Maps

BGP simulators are not rare. Several projects have developed some abstract model of the protocol to check its properties. For instance, SSF.OS.BGP4[19] (implementation of the BGP protocol on SSFNet simulator) was developped to study BGP stability behavior in different network topologies, when confronted to various events. The simulator contains a lot of features from the protocol specification, and even includes implementation-specific aspects. However, the model did not incorporate user traffic's effects on BGP. Morever, the simulation model is too "rich": the memory and CPU consumption required makes it hardly usable on large topologies. Unlike the former simulator, C-BGP[20] was designed to work on topologies of several thousands of ASes. This simulator, built as an efficient decision process simulator, allow experimentations on BGP protocol and attributes. However, it lacks temporal dimension, as the simulator does not implement BGP timers, nor messages delays (link crossing delay, router CPU waiting). Another large scale BGP simulator was proposed in [13]. Its aim was to run BGP simulations on several thousands nodes topologies by keeping the temporal delays introduced by the BGP timers. But, like all the previous simulators, this one uses Logical AS topologies, ignoring IBGP interactions and the effects of AS-internal event on the convergence. Same goes for the RouteSim BGP simulator[17]. Althought the model was abstracted enought to allow large topologies simulation, it lacked realistic temporal delay model, not allowing the observation of user traffic variation consequences on BGP signalisation. Another way in BGP simulator conception was explored by the creators of BGP++. By integrating a *real* BGP implentation into a simulator, the simulations gained a realness never obtained before, but the low level of abstraction made memory and CPU consumption too large to allow large topologies simulation. Clearly, all the simulators already designed are useless for our own purpose, even if they help us to design our simulation model. Thus we have to build our own simulator. Similarly, Internet topology has recently became a hot topic. Tools based on traceroute and ICMP protocols can be used to census the Internet links but the maps we can obtain is not complete and suffers from several problems : some links are missing and some links are aggregated. But we must know the exact topology to model the exchange of update messages between BGP nodes. We address this problem in the tool we have developed.

The paper is as follows. In section 2, we describe the simulator model of the BGP protocol, the network topology, the network load and the router configuration. Note that we only describe the simulator model, not the actual implementation. Then in section 3, we present the tool we have designed to generate Internet graphs from maps already obtained (see for instance the Route Views project [27]).

2 Large Scale BGP Simulation Model

We present in this section a simulation model for the inter-domain protocol, BGP. BGP had been the *de facto* inter-domain routing protocol for more than a decade and our model is mainly an abstraction of the real protocol. That is, some elements are kept as described in the protocol specification. Some others elements are simplified or reduced so they can fit the modelisation process needs. Lastly, some elements are completely ignored, since they don't impact the behaviour we are interested in. And, because this simulation model was created to produce the same *instabilities* as the ones observed on the network, we tried to give the components of the simulator behaviour as close as possible from real components on the network. For example, the real BGP speakers can crash due to an overload of traffic. This is modeled in the simulator by giving the speakers the possibility to cease their functioning. This affects the traffic of BGP signalisation and produces some interesting behaviour of instability.

An important point about the simulation of BGP instability is the ability for the simulator to produce the same *kinds* of instability as the ones observed on the Internet and generating them with the same proportions. This is an important matter as it affects and explains many choices done in the process of modelisation. For that reason, many aspects of the simulation model were straightly deduced from instabilities observation on the interdomain network. For example, the need to model a more complex AS was driven from the fact that BGP traffic observation showed that an important part of exchanged BGP messages springs from the use of a particular routing policy type. This type of policy has almost no effect on single connected ASs. Also, the examination of instabilities and their impact on the user traffic give us hints on how to model some other details of the simulator as temporal delays affecting the messages.

Hence, the simulation model was born from the combination of elements deduced from interdomain traffic watching and real BGP specification abstraction. Combined together, the following elements form our large scale BGP simulation model:

1. The first component is a *realistic* model of *BGP speakers* (BGP communicating entity). The simulation model speaker must follow as closely as possible the model of BGP speaker described in the BGP specification. Or, at least, the modeled speaker must produce the same behaviour as a real one in the same conditions.

2. The second component of the simulation model is the simulation topology. To produce the same kind of instabilities as those produced in the Internet, the simulations must run on a topology very close to the real topology (Internet interdomain topology) in term of shape and size.

3. The third component is directly related to simulation topologies. In almost all the BGP simulators, the topologies are graphs of ASs (Autonomous Systems) where each node represents a hole autonomous system, but is modeled as an unique speaker. Since some interesting behaviours of instability are caused by the fact that the AS is composed from several interconnected speakers, our simulation model uses a model of Autonomous System that is more complex than the usual model. We choose to model each AS as a graph of internal speakers. Every speaker being connected to speakers from other ASs. This way, the ASs can have multiple links between them (as it is commonly found in the Internet) and the message generation and outgoing in each single AS will be more realistic.

4. The fourth component of our model includes the routing policies and the way they are integrated to the simulations. Since BGP does not use a shortest path first algorithm to choose its paths, but permits to each AS to choose freely the way it selects its paths (by using routing policies), it is important to include in the simulator these routing policies to watch the impact of different policies on the routing stability.

5. The fifth component of the simulation model adds, when complemented by the use of BGP timers, the temporal dimension to the simulations. This dimension is introduced by the simulation of the delays imposed on BGP messages going through the network: the delay of transmission of the packets, the waiting time for each packet when arriving at the receiver and the processing time of the packet at that speaker.

To obtain the desired behaviour from the simulator, i.e., routing instabilities and slow convergence, the five components must be combined. Since, the deficiency of one of them can prevent the behaviour we would like to observe.

The following subsections describe each component by specifying each of its aspects.

2.1 BGP Speakers Modelisation

The first element considered in the conception of a BGP instabilities simulation model is the elementary node of the simulation and the way it communicates with its neighbours. In BGP, this elementary node is called a *speaker*. The speaker model we conceived is not far from most of the simulation models created before. However, it contains some elements modeled completely differently, integrated by suppressing the complexity of the element, or adopted most integrally as described in the BGP specification. In particular, the sessions management system of our model is extremely simplified but allow us to simulate events as speaker crashes or reboot, new links establishment or links failures. As shown in figure 1, the BGP speaker model is composed from several active components:

- **The *Ingress* Filter:** The role of this component is to receive messages from neighboring speakers and filtering them. The process of filtering messages is done by using routing policies. Hence, each time the *Ingress Filter* receives a messages from a peer X of the speaker, the filter applies the routing policy corresponding to X to the received message. This way, messages can be rejected, accepted or modified before acceptance. The filtered input messages are put

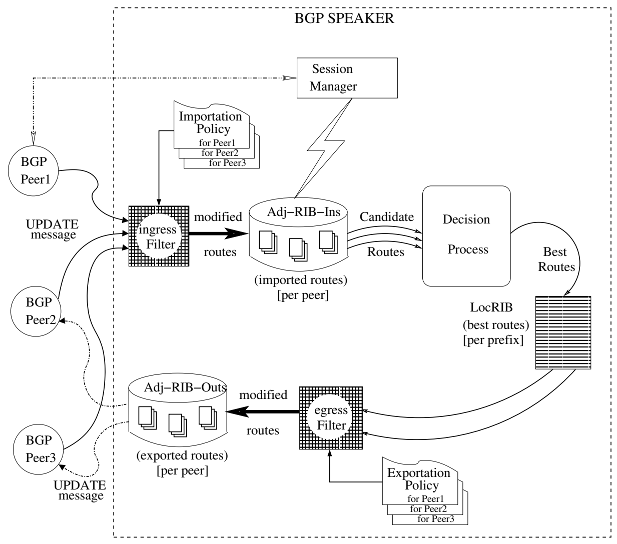

Figure 1. BGP Speaker Simulation Model

in the Adj-RIB-In associated to the peer who sent the message.

- **The Decision Process:** This component is invoked each time the speaker receives a new message or loses its best route to a destination. By looking in every Adj-RIB-In of the speaker, this process can choose the *best* path to a certain destination. This particular path is marked as the best route to the destination, stored in the Loc-RIB and transmitted to the *Egrss* Filter for exportation. The best route selection method is explained in the next subsection.

- **The *Egress* Filter:** This component works the same way as the *Ingress* filter, but it applies *exporting* policies to the routes that the speakers wants to be announced to its neighbours. As for the *Ingress* filter, the policies give the possibility to modify the routes before exportation and they can stop certain routes from being announced. A copy of the filtered exporting route is put in the Adj-RIB-Out of each peer to which the route is to be announced.

- **The Sessions Management Unit:** This component manages the sessions between the speaker and its peers. This unit keeps for each of the peers a **state of the session**. This state allows us to simulate different situations for the speakers and the links. For example, we may produce situations where only the links fails, where one of the speakers cease from functioning . . .

In addition to those active components, the speakers contains data storage elements:

- The **Adj-RIB-Ins** contain the imported routes. One Adj-RIB-In is associated to every peer of the speaker. The entries of those tables are the messages carrying the routes.

- The **Adj-RIB-Outs** contain the routes to be announced. One Adj-RIB-Out is associated to every peer of the speaker. The entries of those tables are different from those in the Adj-RIB-Ins: the sender identity changes, as well as other potential route attributes.

- The **Loc-RIB** contains the best route for each known destination. Its entries are pointers to routes in the Adj-RIB-Ins.

- The **Policy Tables** contains the routing policies of the speaker. These are sets of rules, stipulating the actions to be done when some conditions on the routes (imported or exported) are met.

Since our objective is to simulate topologies as big as the Internet, we must assume several simplifications in the model of the speaker. First, only one type of message is left. The three other kinds of messages would serve in sessions management. Since the session management is simplified and left to a specialised unit, the messages are of no use. The remaining message type, UPDATE messages, will carry the routes exchanged between the speakers. These messages will comport the attributes ”LocalPref”, ”AS-Path”, ”MED”, ”OriginatorID” and ”ClusterList”. The LocalPref attribute is coupled to a path (message) when it enters an AS. The message carries this attribute, unchanged, while it is in the same AS. When the routing information is about to be published to another AS, the border speaker must suppress this attribute from the exported message.The ASPath carries an ASN sequence representing the path from destination AS to the route announcer AS. The med attribute allow speakers to differentiate between many existent links binding a couple of ASs. Finally, the attributes OriginatorID and ClusterList are used in that particular case of ASs with Routes Reflectors. An important matter impacting the stability of the inter-domain routing relies on the fact that BGP uses timers to reduce the volume of messages exchanged on the network. In the simulation model, we incorporate two kinds of temporal limitation for the announcements:

- The *Minimum Route Advertisement Interval* (MRAI) is a timer described in BGP specification to limit the rate of route announcements from a speaker. Each time a speaker announces a route, it must set this timer and can’t announce any other route about that destination until the timer allows that. By using such a limitation, speakers can’t send multiple close announcements about one destination, limiting this way the network bandwidth used by BGP and suppressing many potential erroneous announcements. Because this timer affects the convergence of BGP, we decided to put it in the simulation model, exactly how it is in real speakers. The announcement of a route will set the timer for each peer (for **all** the destinations exported to that peer, as it is used in all the commercial implementations of BGP). During the waiting period, the new messages about a particular destination will erase the old ones. This way, the timer will trigger the announcement of the most recent route.

- The *Withdrawal Rate Limiting* is a technique of announcement limitation used in several BGP implementations. The BGP specification advise to not apply the rate limitation (by MRAI) to the route withdrawals. However, several well distributed implementations apply this limitation. It consists in applying the same kind of limitation to route announcements **and** to route withdrawals. This way, only the most recent route withdrawal for a destination will be distributed. Thought, several simulations showed the bad effects of this method, we choose to put it in our model, to reflect the state of BGP implementations in the real world, hoping to obtain the most realistic reactions.

In the late 90's, labovitz discovered that the volume of BGP traffic was several times the one expected for the Internet [15]. One of the reasons for that, was the adoption by the BGP routers manufactures, of a technique of announcement that reduces the memory consumption. When a router sends an announcement, it never keeps the last announced route, thus saving memory. This method caused the routers to send several times the same announcements, producing a huge volume of BGP messages. This particular BGP implementation (Stateless BGP) caused a great burden on the Internet. Unfortunately, not all the manufacturers accepted to implement a last-message-save. In our model, this behaviour of BGP speakers is include, to observe the impact of bad-behaving speakers on global convergence and eventually on other speakers.

BGP decision process The decision process included in the simulation model is directly extracted from the specification. Some steps are simply removed because the elements (message attributes) theses steps are based on, were removed from our abstraction. First, we describe the exact content of each BGP message:

1. The sender identity: IP address of the sender (simulation attribute PEER_ID) and his ASN (simulation attribute PEER_ASN).
2. The withdrawn routes: is a set of prefixes withdrawn by the sender (variable length simulation attribute WITHDRAWN).
3. The attribute LOCAL_PREF: only for the messages within an AS.
4. The attribute AS_PATH: a sequence of ASNs describing the path from the sender's AS to the destination's AS.
5. The attribute MED: a value differentiating routes from multiple sessions between two ASs.
6. The attribute ORIGINATOR_ID: only for Route Reflectors based ASs; This attribute is used to prevent messages loops within an AS.
7. The attribute CLUSTER_ID: only for Route Reflectors based ASs; This attribute is used to prevent messages loops within an AS.

When a speaker receives a new message, it first pass it through the *ingress* filter, then it put it in the appropriate Adj-RIB-In, deleting any older message about that destination and deleting the new message if the speakers ASN is found upon the as_path attribute. If any modification impacting the route for a certain destination is detected by the speaker (new route, old best route deletion, old best route withdrawal), it launches the decision process to choose a path to the destination. This process entries are a set of all the candidate routes for that destination; the process passes by several steps, rejecting less wanted routes at each step:

1. Eliminate routes whom local_pref attribute value are strictly lower than the largest local_pref among the candidate routes.
2. Eliminate routes whom as_path lengths are exactly lower than the longest as_path among candidate routes.
3. Eliminate from each group of routes originating from the same AS, the routes that possess med attribute values greater than the lowest med value from that AS.
4. Eliminate routes, coming from internal peers, whom the AS IGP metric to the announcing peer is greater than the lowest IGP metric for a internal imported route.
5. Choose only the route announced by the external peer with lowest BGP ID.
6. Choose only the route announced by the internal peer with lowest BGP ID.

Passed this process, only one route is remaining, it is placed in the Loc-RIB and exported to peers.

BGP Sessions Management Unit In our BGP simulation model, the sessions can have two states: **up** and **down**. The switch between the two states is achieved by the sessions management unit (SMU), which is the component simulating different effects of external events on BGP sessions. It can be told to re-initiate sessions, as well as completely run down a router (close all its sessions). It also simulates some of network load effects on BGP sessions, by imitating BGP comportment in link failure events: a *HoldTimer* is affected to every up session. This timer is initiated each time the speaker receives a message from the session peer. In case the peer does not send messages until the timer expires, the SMU computes a **session loss probability**. When this probability exceeds a threshold, the SMU simulates the real BGP HoldTimer expiration: it runs the session down, removing all the routes imported from the peer from the Adj-RIB-In and sending a message of session run-over to the peer's SMU. If the Loc-Rib is affected by the session stop, the decision process is launched to find a new best route for destinations that lost their best path.

2.2 Simulations Topologies

Network simulations are affected by the network topologies used in simulations. This is also true in BGP simulations, where the topology's shape as well as the topology's size impact the behaviour of the simulator. The use of realistic BGP network topologies is an important element affecting the instabilities behaviour and volume. In almost all the BGP simulators, the topologies used are **Logical BGP ASs Topologies** (figure 2). The nodes of these topologies are complete BGP ASs regarded as single simple nodes. The complex routers topology within each AS, is abstracted to a single speaker who deals with all the communications from that AS. This kind of topologies is obtained by processing *BGP logs* from various Internet looking glasses, or by active measures (rarely used). The processing can split a physical AS into several logical simple ASs, to reflect the network complexity inside that AS. Our BGP simulation model's aim is to produce the same types of instability as in the Internet with relatively the same amount and proportion. That is why we intent to use a more realistic topology model: the **BGP Sessions topologies**(figure 3). This kind of topology uses as elementary node the **BGP speakers** of each AS. This speakers are internally interconnected within each speaker and have links between the ASs. This way, no need to split the autonomous systems into logical ASs, the topology is a more realistic one, as the ASs can possess several sessions between each other (unobtained on a ASs topology). Produced from BGP ASs topology and information from router level Internet map, this kind of topology is **inferred** by an algorithm described in section 3.

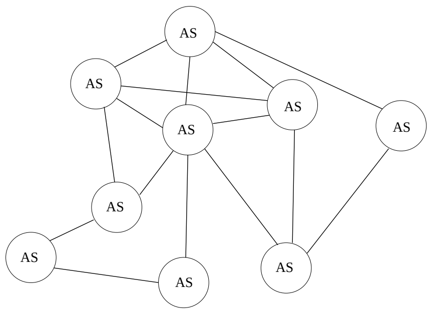

Figure 2. Logical BGP ASs Topology

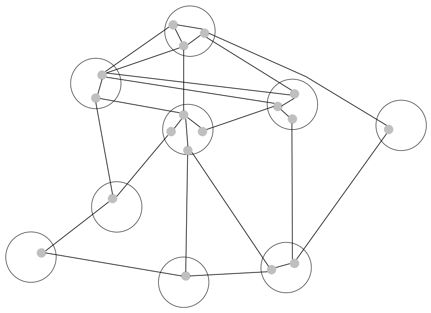

Figure 3. BGP Sessions Topology

2.3 Autonomous Systems Modelisation

Moving from a simplified AS structure (one speaker by AS) to a more realistic one (several speakers by AS) requires a more elaborate AS model. Following the BGP specification instructions, the speakers within an AS must be interconnected by **IBGP sessions** and communicate with external speakers using **EBGP sessions**. To obtain a more true to life instability behaviour and to capture additional instability starters, the AS model internal structure can be made from **Full Mesh** interconnected speakers (normal IBGP connection approach) or can be formed from **Routes Reflectors** and their client speakers.

Full Mesh Autonomous Systems Used in relatively **small** ASs, this interconnection fashion comes from the cycle avoiding rule **in** the AS:"an internally imported route must not be exported to internal peers". To use this rule and keep the routing correctness within the AS, each speaker must establish an IBGP session with every other speaker in the AS. Moreover, each AS must possess an IGP weights graph reflecting the physical internal AS organisation. This graph could be used in the forth step of decision process and exploited to produce IGP-MED mapping policies.

Route Reflection Autonomous Systems Used in relatively **large** ASs, or to test IGP-MED mapping policy impact on the inter-domain routing, this kind of interconnection uses a special type of speakers: the routes reflectors. While all the other speakers of the AS apply the IBGP cycle avoiding rule, those speakers can, under certain circumstances, export internally learn routes to internal peers. And, as for the Full Mesh Autonomous Systems, the Route Reflections ASs must posses an IGP weights graph, to allow special policies and decision process application.

2.4 Routing policies Modelisation and Implementation

Routing policies are the way by witch the simulations achieve realness. In deed, on the actual Internet, one can hardly find an AS who does not disable the default BGP route choice (shortest path first) by adopting a source-driven local_pref value attaching policy. The routing policies implement the routing strategy adopted by the ASs; they are the mean by which the transit service agreement are achieved. In the simulation model, routing policies are implemented as sets of **decision rules**. A speaker owns a set of decision rules for every peer. One set is an ordered list of rules:

$$< DecisionCriteria \Rightarrow Action >$$

where:

- $DecisionCriteria$ is a logical combination of conditions on route (filtered message) attributes.
- $Action$ represents the operation to be done. The operations can be attributes modification, route acceptance, or route denial.

Each time a route is filtered, the set's decision criteria are successively examined; and each time a condition is full filled, the corresponding action is executed, before carrying on rules examination.
An important matter in BGP simulation is the routing policies adopted in simulation and where it should be put on the topology. From observation, Hao and Kopol identified three kinds of policies on actual world Wide Internet [13]:

1. Transit policies: allow speakers to offer a transit service to some other ASs. These policies are applied by *Internet connectivity providers*
2. Local route selection policies: are policies used commonly by customers to specify their preferred routes according to the source announcing the route.
3. AS prepending (AS padding) policies: are policies applied by ASs that try to influence other ASs choices by enlarging their exported routes. This is done by prepending several times their ASN on the as_path attribute. Thus, the ASs that apply a shortest path first selection can be influenced in their best route selection process.

For now, there is no sure information on where these policies must be put to produce Internet's inter-domain behaviour. But we can wisely distribute the policies by inferring the associations between the ASs. In fact, our model uses an algorithm to try to guess the association between two ASs. First, we build a hierarchy of ASs by examining the ASs topology (the real Internet possesses 4 or 5 levels). After that, we label each link with ”provider-customer relationship” if the ASs belong to different hierarchy levels, else we label it with ”peering relationship”. Then we deliver the policies to an AS according to the link's label:

- if the relationship is ”customer-provider” and the AS is the highest in topology hierarchy then the speakers with this link must apply a transit policy, exporting then it's clients routes and it's providers routes to the client and importing the client's routes freely.
- if the relationship is ”customer-provider” and the AS is the lowest in topology hierarchy then the speakers with this links must apply policies to export only it's clients routes to its provider and accept clients routes and providers routes.
- if the relationship is ”peer-to-peer” then the speakers must only export and accept clients routes. They must not exchange providers routes.

For the AS prepending policies, there is, meanwhile, no information on the exact localisation of the speakers applying this kind of policy, we can only apply them arbitrary, in the simulations. However, it is interresting to know the effects of such policies on the inter-domain routing and particularly on the routing instability.

2.5 Temporal Delays Modelisation

In real networks, BGP messages going from a speaker to another take a certain time before being processed by the receiver. This time period between the message emission and the end of processing is the sum of three periods of time: link crossing time, waiting time at the receiver and processing time. In the simulation model, two kinds of delay are charged on the message:

Link crossing delay for messages between external peers (the delay in AS-internal link crossing is neglectable);

Speaker's CPU charge induced delay is a delay inflicted to every entering message of a speaker. This delay models the waiting time induced by other activities of the router (packets forwarding, IGP, . . .).

The classical approach to model the link crossing delay consists in choosing a random value in an interval and letting the message wait the drawn time period. This approach do not allow the study of user traffic volume augmentation effects on interdomain routing, for example. Thus, we adopted in the model a method of link crossing delay estimation based on links congestion. This congestion touches both the packets loss and the round trip time of the link. We can, thus, *estimate* the link crossing delay using those two variables:

$$CrossingDelay = \frac{PacketSize}{LinkThroughput}$$

$$LinkThroughput = \frac{PacketMaxSize}{RTT \times \sqrt{PacketLossRate}}$$

where: $CrossingDelay$ is the link crossing time for a BGP message, $LinkThroughput$ is the link's transmission capacity that is calculated using theRTT (Round Trip Time) and the $PacketLossRate$. To apply this model, we need, for each link between a couple of speakers, the RTT and the packet loss rate. By doing this, we can simulate congestion on regions of the Internet and observe the effects of this congestion on BGP convergence.

The second delay charged on messages is the CPU charge induced delay. It comes from the fact that the almost all the real Internet's speakers are normal routers that have, in addition to the BGP processing, several jobs running (packets forwarding, IGP processing, monitoring, other BGP messages processing, . . .). The result is a *waiting time* for the BGP messages arriving at a speaker. We modeled this waiting time by adding two components:

1. the first component models the delay induced by BGP processing. BGP messages arriving at a speaker share the same waiting queue. Thus, an arriving message must wait for the preceeding messages to be processed. This delay is computed by counting the BGP messages passing by a speaker during a period of time. Then, the following formulae is used to compute the BGP treatments-induced delay for an incoming message:

   $$D_{BGPProcess} = \gamma \times N \times T_S^2$$

   where: N is the number of BGP message by time unit, T_S is the time needed to process one BGP message and γ is a parameter depending on the packets arrival.

2. the second component models the delay induced by router's other jobs. This second component is function of the network activity. Packets number increase will cause a more important delay on BGP messages. Meanwhile, it is not possible to obtain exactly this parameter behaviour, we choose to assign it a fixed value. By increasing this value, it will be possible to study the impact of network activity increase on BGP convergence and stability.

The addition of the two components will give each message the waiting time until the CPU is free to process it.

3 AS Topologies and BGP Sessions Topologies Creation Tool

As told in subsection 2.2, our simulation model uses a particular kind of network topologies: the BGP Sessions Topologies. The elementary node of these topologies are BGP speakers, not BGP ASs. These speakers are connected *within* the ASs using IBGP sessions and between the ASs using EBGP sessions. That is, multiple BGP sessions can attach a couple of ASs and speakers within the same AS can choose different routes for the same destination. This disposition, widely met in real network, is rarely adopted in simulations. The lack of information on Internet's real connectivity incite us to try to infer this kind of topology from disseminate information. First we must have a BGP ASs Topology. This is obtained by compiling data gained from various Internet looking glasses. Next, we must adjoin complementary information to this topology. In our BGP Sessions topology generation algorithm, those information consist of router-level connectivity between the ASs. Thought difficult to obtain (by traceroutes or by another mean), this kind of information allow us to burden links using the number of router-level links joining the ASs as weight for the links. Third, we reduce the links weights by dividing all the weights by a common factor computed to obtain the right final number of BGP sessions. The final action consists in the creation of BGP speakers for each AS and the creation of BGP sessions between them. Thus, the algorithm of BGP sessions topologies construction is given by:

1. **Weights computing:** Given a router level topology, we must gather routers by AS and compute for each BGP link between a couple of ASs the number of links joining routers from the couple of ASs. This number will be the initial weight given to the AS link.

2. **Weights reduction:** In deed, if there is four router-level links between two ASs, it does not mean that four BGP sessions tie together the ASs. We must, then, reduce the links weights by dividing them by a common factor, to estimate the number of BGP sessions connecting a couple of speakers. The factor is computed by the formulae:

   $$factor = \frac{\sum_i LinkWeight_i}{\frac{LinksNumber}{1-MissingLinksPercentage}}$$

where: $LinkWeight_i$ is the weight of the i^{th} link of the topology,$LinksNumber$ is the total number of links in the AS topology and $MissingLinksPercentage$ is the proportion of missing links in ASs topologies built from BGP logs. In deed, the AS topology built from those logs are incomplete: they miss several links being in the actuel network. In particular, multiple links between ASs are gathered in the produced AS topology. The $MissingLinksPercentage$is an estimation of the proportion of links that do not appear in the topology despite their presence in the real network. Recent research's estimated that this proportion vary in the interval 20% to 50% of the number of links being in the topology of ASs. The reduction must produce integer number, so the non-integers devision product is rounded to the closest non-zero integer value.

3. **BGP speakers creation:** In each AS, we must look for the link with the maximum weight. We, then, associate to that AS a number of speakers equal to this link's weight. Adjoin to each AS link a counter initialised with that link's weight.

4. **EBGP sessions establishment:** We repeat the same actions as long as the AS posses at least one counter who's value is greater than zero. First, we must choose a non-completely connected speaker. For each non-zero counter, we establish an EBGP sessions between the speaker and a neighboring AS speaker, but carefully choosing the neighboring speaker as to avoid establishing multiple sessions between a couple of speakers. The counters are decremented on the two sides, each time a sessions is successfully established.

5. **IBGP sessions establishment:** When all the EBGP sessions are established, we must establish the sessions within the ASs. For Full Mesh ASs, there must be one IBGP session between each couple of speakers of the AS. For the Route Reflection ASs, so far, there is no specified way of constructing internal AS structure. We must, then, select random speakers that will be Route Reflectors and select among the remaining speakers the clients of each route reflector. There must be one IBGP session between a R.R. and its client, but the route reflectors, as well as the speakers that do not belong to a cluster, must be connected using an IBGP sessions full mesh.

The BGP sessions topology generation pseudo-code is given in the following table. Thought the produced topology is not identical to the real interdomain network (whose opaqueness restricts information gathering), it entails topology features used to produce interesting instability behavior.

```
BGP Sessions Generation PseudoCode
// 1- By some mean, give to every AS link a weight.
// Let this weight be $LinksWeight_i$ for link i
2-$factor = f(ASTopology, MissingLinks\%)$
3-For every link i in the AS topology, do:
    *$LinksWeight_i = LinksWeight_i / factor$
    *if ($LinksWeight_i < 1$)
        $LinksWeight_i = 1$
4-For every node $AS_j$ in the AS topology, do:
    *Look for maximum weighted link $m$
    *Create $m$ speakers for $AS_j$
5-For every node $AS_j$ in the AS topology, do:
    *Look for a non-zero AS-link $m$
    *If there is not, pass to the next AS.
    *Select a speaker not yet connected to
    the neighbour $AS_h$
    *Select a speaker in $AS_h$ not yet
    connected to the speaker of $AS_j$
    *Establish the EBGP session
    *Reduce counters
6-For every node $AS_j$ in the AS topology, do:
    // Full Mesh only
    $n = numberOfSpeakersofAS_j - 1$
    * For every speaker of $AS_j$, do:
        * Connect speaker to $n$ others
        * $n = n - 1$
7-Topology generation completed.
```

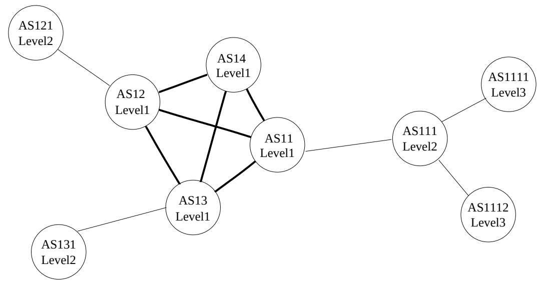

Figure 4. Generation Tool Input AS Topology

This algorithm has been implemented in a tool for BGP sessions topologies generation. Thought in early stage of elaboration, it allows ASs topology manipulation (addition and suppression of nodes and links, . . .). For now, the first step in our algorithm is not full filled using router-level information. We use instead, random generated weights values, looking meanwhile for a way to obtain reliable, complete and consistent router-level data.

The given figures show examples of tool generated BGP Sessions Topologies. Figure 4 shows the input topology, that is, the AS topology fed to the algorithm to produce the BGP Sessions Topology. Figures 5, 6 and 7 show the

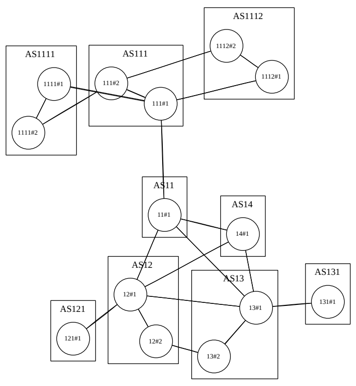

Figure 5. Tool generated BGP Sessions Topology (Missing Links Percentage=20%)

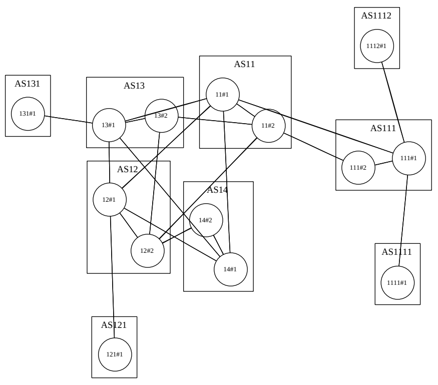

Figure 6. Tool generated BGP Sessions Topology (Missing Links Percentage=40%)

impact of the parameter *Missing Links Percentage* on the topology generation.

The tool also implements an interdomain network hierarchy discovery algorithm. It exploits a particularity of Internet maps: at the highest level, ASs are (almost) entirely interconnected. The ASs of this level (Tier1 providers) are connected to some lower level ASs (Tier2 providers), that provide connectivity service for lower level ASs, . . . The algorithm tries to identify the biggest connex component in the ASs graph. This problem being a NP-complete problem, we use an heuristic to gather ASs to form the biggest clique we can. The ASs belonging to the clique are told to be Level 1 hierarchy ASs. Next, ASs who do not belong to the clique but are connected to Level 1 ASs are said to belong to the Level 2 hierarchy. Next, ASs attached to level 2 ASs are labelled with Level 3 hierarchy and so on. For example, figure 4 shows for every AS node, the hierarchy level it belongs to, deduced by the method described. By identifying the level of hierarchy of each AS, this algorithm give us opportunity to generate individual routing policy for every speaker of an AS. As described in subsection 2.4, the type of routing policy applied for route importation (resp. exportation) can be deduced from speakers and peers AS hierarchy levels.

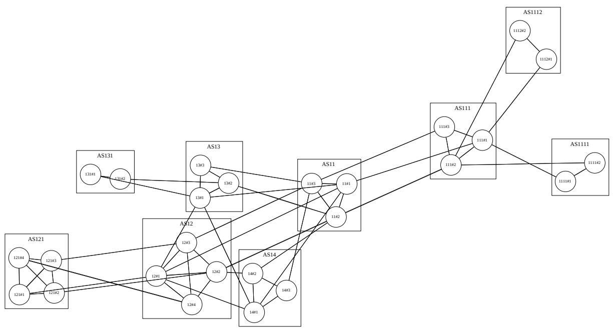

Figure 7. Tool generated BGP Sessions Topology (Missing Links Percentage=60%)

4 Conclusion

BGP had been the official Internet interdomain routing protocol for the last decade. Its wide utilisation, its relatively open implementation and the liberty given for route selection and information propagation, made the entire Internet rely on it for interdomain routing information propagation. However, this routing protocol have not been designed to manage the increasing number of hosts present on the actual Internet. BGP suffers from a reliability problem: observations of Internet BGP signalisation showed great instability, as well as sensitiveness to user traffic volume fluctuations. Moreover, BGP exhibits safety problems, as proved by recent worm's propagation influence on BGP infrastructure. BGP problems resolution requires simulation: we must possess a simulation model able to produce BGP instability before trying to find new protocol modifications to remove it. We must also possess realistic Internet topologies since the experimentation of protocol simulation and modification gives more reliable information when the topology is very close to the real network topology. In this paper, we presented a large scale BGP simulation model. One important component of this model is the topology used for simulation. We exposed the type of topologies needed to produce realistic behaviours, as well as the tool actually used to produce these topologies. The next step will be the addition of routing policy generation capabilities to our tool, while trying to find a way to gain reliable and consistent router-level topologies.

References

[1] Y. Chen and A.K. Datta and S. Tixeuil, ”Stabilizing Inter-domain Routing in the Internet”, Europar 2002, LNCS 2400, Paderborn, Germany, pp 749-752

[2] J. A. Cobb and M. G. Gouda and R. Musunuri, ”A Stabilizing Solution to the Stable Path Problem”, Self-Stabilizing Systems 2003, 6th International Symposium, SSS 2003, Lecture Notes in Computer Science 2704, pp 169-183, San Francisco, CA, USA

[3] S. Delaët and D. Nguyen and S. Tixeuil, “Stabilité et Auto-stabilisation du Routage Inter-Domaine dans Internet”, RIVF 2003, Studia Informatica Universalis, pp 139-144, 2003, Hanoi, Vietnam Ed. Suger

[4] E.W. Dijkstra”, ”Self stabilizing systems in spite of distributed control”, ”Communications of the ACM”, V 17, 1974, pp 643-644

[5] X. A. Dimitropoulos, G. F. Riley, ”Creating Realistic BGP Models”, 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer Telecommunications Systems, ’03, page 64.

[6] S. Dolev, ”Self-stabilization”, 2000, The MIT Press

[7] T. D. Feng, R. Ballantyne, and L. Trajkovic. Implementation of BGP in a network simulator. In Applied Telecommunication Symposium, ATS ’04, pages 149-154, Avril 2004.

[8] L. Gao and T. Griffin and J. Rexford, ”Inherently Safe Backup Routing with BGP”, IEEE INFOCOM’01, V 1, pp 547-556, 2001

[9] R. Govindan and A. Reddy, An analysis of Internet inter-domain topology and route stability, IEEE INFOCOM, 1997, pp 850-857.

[10] T. Griffin and G. Wilfong, ”An analysis of BGP convergent properties”, ACM Sigcomm 99, 1999.

[11] T. Griffin and G. Wilfong, ”A Safe Path Vector Protocol”, IEEE INFOCOM’00, pp 490-499, 2000

[12] T. Griffin and FB. Shepherd and G. Wilfong, ”Policy disputes in path-vector protocols”, International Conference on Network Protocols (ICNP’99), 1999.

[13] F. Hao and P. Koppol. An Internet scale simulation setup for BGP. SIGCOMM Comput. Commun. Rev., 33(3) :43-57, 2003.

[14] S. T. Kent, ”Securing the Border Gateway Protocol: A Status Update”, 7th IFIP TC-6 TC-11 Conference on Communications and Multimedia Security, ’03.

[15] C. Labovitz, G. R. Malan, and F. Jahanian, “Internet Routing Instability”, ACM SIGCOM 97, France, pp 115-126

[16] C. Labovitz, G. R. Malan, and F. Jahanian, “Origins of Internet Routing Instability”, IEEE INFOCOM 99, USA

[17] J. Nykvist and L. Carr-Motyckova. ”Simulating convergence properties of BGP”, In Proceedings of the 11th International Conference on Computer Communications and Networks (ICCCN). IEEE, Octobre 2002.

[18] R. Perlman, ”Interconnections: Bridges, Routers, Switches, and Internetworking Protocols”, 2000,Addison-Wesley Longman

[19] B. J. Premore. An Analysis of Convergence Properties of BGP using Discrete Events Simulation. PhD thesis, Dartmouth College, Hanover, New Hamshire, Mai 2003.

[20] B. Quoitin. Synthesis and recommendations for inter-domain traffic enginneering, Novembre 2003. Information Society Technologies.

[21] K. Varadhan and R. Govindan and D. Estrin, ”Persistent route oscillations in inter-domain routing”, Computer Networks (Amsterdam, Netherlands, 1999, volume32, number 1, pp 1–16,

[22] Y. Rekhter and T. Li, *”A border Gateway Protocol4 (BGP-4)”, RFC 1771. 1995.*

[23] G. Varghese, ”Self-stabilization by counter flushing”, International ACM Conference on Principles of Distributed Computing, 1994, pp 244-253

[24] G. Varghese and M. Jayaram, The Fault Span of Crash Failures, 2000, Journal of the ACM, V 47, N 2, pp 244-293

[25] L. Wang, X. Zhao, D. Pei, R. Bush, D. Massey, A. Mankin, S.F. Wu, and L. Zhang, ”Observation and Analysis of BGP behavior under stress”, ACM SIGCOMM USENIX IMW, pp 183-195, 2002

[26] M. Liljenstam , Y. Yuan, BJ. Premore and D. Nicol, ”A Mixed Abstraction Level Simulation Model of Large-Scale Internet Worm Infestations”, IEEE MASCOTS 2002, USA, pp 109-116.

[27] ”*The Route Views Project*”, http//:www.antc.uoregon.edu/route-views/