Improving Internet-wide Routing Protocols Convergence with MRPC Timers

Anthony Lambert
Orange Labs
Paris, France
anthony.lambert@orange-ftgroup.com

Marc-Olivier Buob
Orange Labs
Paris, France
marcolivier.buob@orange-ftgroup.com

Steve Uhlig
TU Berlin/T-Labs
Berlin, Germany
steve@net.t-labs.tu-berlin.de

ABSTRACT

The behavior of routing protocols during convergence is critical as it impacts end-to-end performance. Network convergence is particularly important in BGP, the current interdomain routing protocol. In order to decrease the amount of exchanged routing messages and transient routes, BGP routers rely on MRAI timers and route flap damping. These timers are intended to limit the exchange of transient routing messages. In practice, these timers have been shown to be partly ineffective at improving convergence, making it even slower in some situations.

In this paper, we propose to add a timer mechanism to routing protocols, that enforces an ordering of the routing messages such that path exploration is drastically reduced while controlling convergence time. Our approach is based on known results in generalized path algorithms and endomorphism semi-rings. Our timers, called MRPC (metrics and routing policies compliant), are set independently by each router and depend only on the metrics of the routes received by the router as well as the routing policies of the router. No sharing of information about routing policies between neighboring ASs is required by our solution. Similarly to the case of routing policies that may lead to BGP convergence problems, arbitrary routing policies can also make it impossible to enforce an ordering of the messages that will prevent path exploration to occur. We explain under which conditions path exploration can be avoided with our timers, and provide simulations to understand how they compare to MRAI.

Categories and Subject Descriptors

C.2 [**Computer Systems Organization**]: Computer Communication Networks, Network Protocols, Internetworking

General Terms

Design, Theory

Keywords

Path algebra, Routing, Convergence, Path exploration

1. INTRODUCTION

Convergence problems have been identified in BGP since almost a decade [18, 21]. Two delay mechanisms have been introduced in BGP to reduce the number of routing messages exchanged between routers: MRAI and route flap damping. *MRAI* (Min Route Advertisement Interval Timer) [33] is a timer that determines the amount of time that must elapse between consecutive announcements from a BGP router to a neighbor. As a router expects learning an improved route before its MRAI timer expires, MRAI supposedly addresses path exploration. Route flap damping [37] addresses a different problem: dynamics generated by faulty and flapping hardware. Route flap damping filters the propagation of consecutive routing updates belonging to a prefix that are following each other over too small time periods, which is a sign of instability. Finding the right values of these timers that fit most situations appears impossible [5, 9, 15, 25, 26], as routing dynamics depends on the type of failures, their location, and on the network topology. The main issue with MRAI is that it slows down at the same time the propagation of obsolete and valid paths. Route flap damping may also slow down the propagation of messages not rooted from instabilities but from normal convergence, making the problem of slow convergence in the Internet even more acute in some situations [5, 26].

In this paper, we discuss for the first time in a formal and generic way, the problem of routing protocols dynamics, through carefully designed timers. We show that if timers are used to properly schedule the order in which routing messages propagate across the network, by design, network convergence time can be made arbitrarily small without requiring unnecessary routing messages to be exchanged between routers. The additional benefit of proper scheduling of routing messages is improved consistency of forwarding during convergence.

Our approach is radically different from most existing works [3, 8, 30, 31, 38] that have focused on identifying obsolete paths and preventing those from further propagating in the network. These approaches, even though pragmatic fixes to today's routing protocol shortcomings and potentially effective, do not address the cause of the problem of unnecessary exchange of routing messages, only its consequence. Only [4] has proposed to delay BGP route messages according to the carried AS paths. Unfortunately, [4] does not rely on a generic theoretical background as we do, so cannot deal with multiple path metrics or routing policies as we do.

We present timers that address the cause of the exchange of unnecessary routing updates, called MRPC, for *metrics and routing policies compliant*. MRPC timers are set independently by each router and depend on the metrics of the routes received by the router as well as the routing policies of the router. Our solution does not require that neighboring ASs share information about their routing policies. Along with MRPC timers, our work completes the existing algebraic framework through which the behavior of routing protocols can be abstracted [17], by adding a scheduling mechanism to tackle the exchange of routing messages during convergence.

After providing some background information on routing protocols (Section 2), we illustrate with a simple and intuitive example our timers (Section 3.1). We give the theoretical building blocks of our approach in Section 3.2. In Section 3.3, we describe in a general setting how to compute our MRPC timers, and prove under which conditions they can be computed. As our timers are aimed at being used in real routing protocols, we apply our framework to specific routing algebras in Section 4. This leads us to consider MRPC timers for the particular case of BGP in Section 5. To understand the benefits of our solution, we compare in Section 5 the performance of MRPC timers with the current solution implemented in the Internet, MRAI. We also explain how our timers work under general topological events in Section 6 as well as in the case of flooding protocols. Related work is presented in Section 7, before concluding in Section 8. Note that the proofs of the propositions and theorems stated this paper can be found in [22].

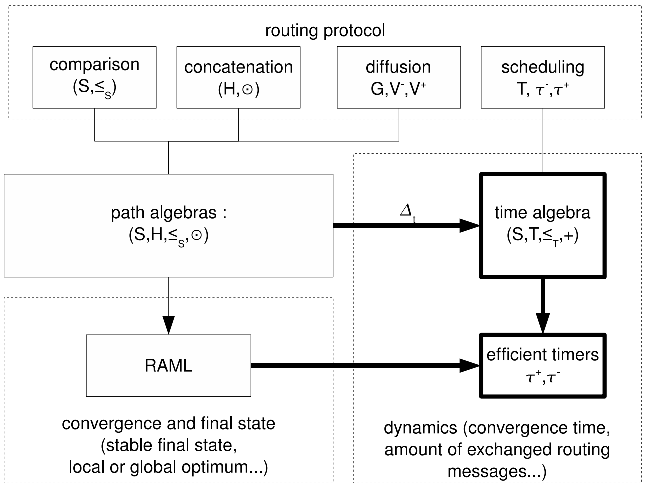


Figure 1: A routing protocol model.

2. BACKGROUND

2.1 Routing protocol basics

Routing protocols aim at building forwarding paths towards any destination of a network. This is achieved by exchanging routing information between neighboring routers through routing adjacencies or sessions. Routers announce to their neighbors routing information related to the destinations they can reach and based on this information, they build a routing table selecting for every destination, the neighbour that should be used to forward traffic. The two main types of routing protocols are distinguished by the topological knowledge routers can have about the network:

- *Flooding*: Routers exchange the state of the adjacencies they share with their neighbors. These *link states* are then flooded to all routers of the network. This enables any router to build a map of the whole network, which is then used to compute shortest paths to any destination. When a router learns new routing information, it recomputes its best paths in case some would have been modified and forwards the received link states to its neighbors. These protocols are called link-state protocols. OSPF and IS-IS are examples of such routing protocols.

- *Incremental*: When a router learns routing information from a neighbour, it applies an *import routing policy* which might filter or modify it. Then, if this information triggers the recomputation of part of its routing table, the router may announce new routing information to its neighbors according to its *export routing policy*. Routing information in the form of distance or path vectors is propagated hop-by-hop across the network, not flooded. Routers do not know the whole network topology. RIP or EIGRP are examples of such protocols. The most widely used is the BGP (Border Gateway Protocol) [33], the current interdomain routing protocol.

Routing information is triggered by topological dynamics: link, router or destination appearance or disappearance, metric change... When a routing protocol converges from a state to another because of some topological modification, two issues may arise. Note that those issues arise because routers do not have a global view of the topology and when they withdraw routes, they try to immediately switch to an alternative route, potentially obsolete, instead of waiting for convergence. With our solution, we ensure that the ordering in which messages propagate will not lead routers to use obsolete routes whose propagation might lead to routing loops for example.

Routing loops. Transient routing loops may appear in flooding and incremental routing protocols. Transient loops appear during the convergence of routing protocols, when routers use a topology or paths that correspond to obsolete network connectivity information. Solutions to ensure loop-free routing have been proposed recently [12, 24, 29, 32]. These solutions require that routers ask and wait for clearance from their neighbours before re-routing. The approach we propose in this paper is not addressing the general problem of routing loops. Our solution guarantees that routing loops due to obsolete paths do not occur during convergence, because with our ordering of the messages a router cannot propagate a message that would create such a loop. Our solution does not require the exchange of special messages between routers, but is based on distributed computation of timers by each router. Furthermore, we do not require all routers to implement our timers, allowing partial and incremental deployment in the Internet.

Path exploration. Path exploration is specific to incremental protocols. Path exploration happens when a router has more than one neighbour announcing a path to a given destination. Depending on the arrival order of these announcements, the router might explore transient paths before learning and converging to its best path. Because routers rely on each other's paths in incremental protocols, a router exploring paths may trigger the exploration of paths by its neighbors. This phenomenon may spread across the network, leading to an explosion of the number of messages exchanged and may dramatically slow down network convergence.

2.2 Routing protocol models

A routing protocol can be formally described by four main mechanisms:

1. *Comparison:* Each path metric may be compared and ordered according to a decision process. This decision process can usually be modeled with a total pre-order relation (see [17, 19]).

2. *Concatenation:* Each path metric related to an incoming (resp. outgoing) route may be modified according to an import (resp. export) policy.

3. *Diffusion:* Each router may accept a subset of the incoming routes and forwards them to a subset of the neighboring routers.

4. *Scheduling:* Each router may delay routing messages.

The most well-known framework in the networking community to abstract the mechanisms of routing protocols is Tim Griffin's "metarouting" [17, 19]. Metarouting proposes to model routing protocols using algebraic structures, in order to prove mathematically whether a routing protocol converges or not. This framework, called RAML (Routing Algebra Meta Language), focuses on the final state reached by the network protocol: does the considered routing protocol converge towards a local or a global optimum, or does not converge at all. For this, RAML considers the comparison and concatenation operators (see Figure 1). Those two operators capture the way routers rank routes from best to worse (using one or multiple metrics), and how routing information is modified by routers while it propagates through them. The limitation of restricting the design of routing protocols to those two mechanisms is similar to understanding the behavior of computer programs using static analysis only: most of the dynamic and transient properties during the execution of the routing algorithm are ignored.

Understanding the way routers choose their paths is important to guarantee that the routing protocol will converge, or that it will lead to a unique choice of the paths [16]. This is especially true in the case of path vector protocols, like BGP, that make it possible to use local routing policies. Once a routing protocol is designed in such a way that it is guaranteed to converge, even if only under certain conditions [16], the next step in the design should tackle the dynamical properties of the protocol: how many messages need to be exchanged under topological changes or how much time it takes to converge. The latter aspects have been mostly untouched in the Internet, even though we have reasons to worry about the convergence time of BGP [21] and the load in terms of the number of routing messages exchanged.

For the first time, we tackle in this paper the problem of defining a timer mechanism that will enforce an ordering of routing messages such that the number of exchanged messages is minimal during the convergence of the protocol. The novelty of our approach is to derive a very simple and intuitive timer mechanism directly from known results in generalized path algorithms and endomorphism semi-rings [14]. To keep the content of this paper accessible to the reader uninterested in the abstract formalism of endomorphism semi-rings, we limit to mathematical content and leave technical details to [22]. The novelty of this paper is summarized in bold on Figure 1, with the time algebra and the efficient timers.

3. MRPC TIMERS

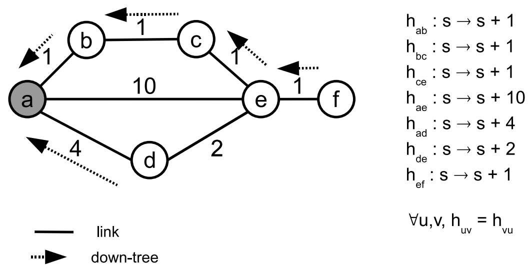
3.1 Overview

We first concentrate on path exploration avoidance in the case of a router announcing a destination to the whole network. Flooding protocols do not suffer from path exploration, as their routing messages describe link state used to build a map of the network topology, based on which paths are computed. In incremental protocols on the other hand, routers exchange path or distance vectors, that may trigger path exploration until they learn their best possible path. Path exploration can be avoided if each router is able to learn its best possible route first, i.e. if the arrival order of the routes is ideal. In this case, the number of messages exchanged can be significantly decreased. The natural way to impose a particular arrival order of the routing messages consists in delaying routing information propagation, especially of the messages that will delay convergence.

Our MRPC timers are designed to delay routing messages according to the path metric carried in the routing message as well as the routing policy configured by the router. If the routing protocol does not have a notion of routing policies (e.g., RIP), then routers will only rely on the path metric. If the routing protocol allows to express routing policies (e.g., BGP), then MRPC timers can be based on both the path metric and its routing policies. When policies are used, each router configures one timer per inbound session[1], depending on the router's import policy, and one timer per outgoing session, depending on its export policy. An import (export) policy is the set of rules that modify a learned (advertised resp.) path.

Shortest paths.

Let us first illustrate the basic idea with an example that illustrates the problem of path exploration. We consider a path vector routing protocol using the usual shortest path algebra, where computing the path length consists in summing the weights of the crossed sessions, and where shortest paths are preferred. h_{uv} represents the fact that the path metric of the route is incremented after crossing the arc from u to v. Router a announces to the whole network a reachable destination (see Figure 2). A router only forwards its best path to its neighbours.



a
b
c
d
e
f
1
1
1
1
10
4
2
$h_{ab} : s \rightarrow s + 1$
$h_{bc} : s \rightarrow s + 1$
$h_{ce} : s \rightarrow s + 1$
$h_{ae} : s \rightarrow s + 10$
$h_{ad} : s \rightarrow s + 4$
$h_{de} : s \rightarrow s + 2$
$h_{ef} : s \rightarrow s + 1$
$\forall u,v,\ h_{uv} = h_{vu}$
link
down-tree

Figure 2: Router e prefers path (e, c, b, a) over path (e, d, a) over path (e, a) to reach router a.

Under these assumptions, router e prefers path (e, c, b, a) over (e, d, a) over (e, a) to reach router a. We therefore want to make sure that router e will first learn path (e, c, b, a) so that its neighbours (router f especially) will not explore nor even hear about e's transient paths. Clearly, if each router delays updates using a constant timer for each neighbour (similarly to MRAI), then the ideal scheduling will not be achieved. This behavior is illustrated on Figure 3: each routers waits for a constant period T before propagating its best route to its neighbours. As a consequence, router e's shortest path is the last to be learned as it is the longest in terms of hops and router f explores transient paths before converging too. Intuitively, if each router delays routing updates on a per-neighbor basis such that the induced delays are a multiple k of the weights installed on its inbound arcs (see Figure 4), router e would learn path (e, c, b, a) after $3 \times k$ time units, path (e, d, a) after $6 \times k$ time

[1]Routers exchange routing messages with their neighbours on *adjacencies*, also called *sessions*. In this paper, we use directional adjacencies or sessions corresponding to the direction of the propagation of routing messages, called *arcs* in the remainder of the paper.

units and path (e, a) after $10 \times k$ time units (with k a common scaling factor). Therefore, router e would immediately converge to its best path, other paths would then be learned but not re-announced to neighbours and router f would not learn about them nor explore them.

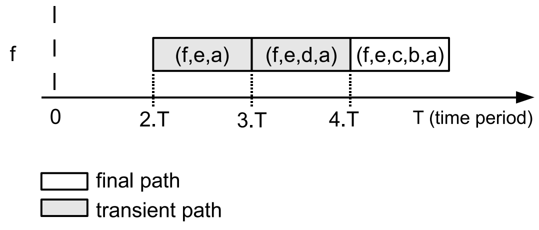


Figure 3: Constant timers: router f hears about router e's transient paths and explores them before converging.

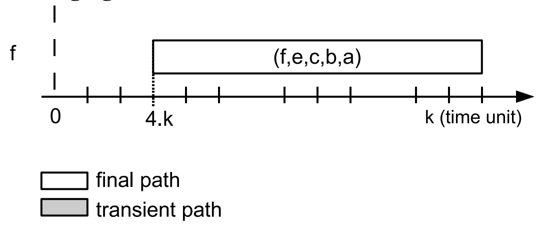


Figure 4: MRPC timers: router f converges immediately without hearing about router e's transient paths.

3.2 Generalized path algorithms

Let first examine quite informally how some routing protocols can be modeled with endomorphism semi-rings (for more details see [14]). Let $G(V, E)$ be a directed graph representing the network topology, and S the metric set related to a path in G according to some routing protocol. We complete S if necessary with ∞_S, the *infinite metric*, which we will associate to paths to unreachable destinations for instance. As explained in Section 2.2, we need a comparison and a concatenation mechanisms to model our protocol. Comparison is obtained by defining a total pre-order[2], $\preceq_S$, over S. $\preceq_S$ models how metrics are compared according to the protocol decision process to decide which route is better. For example, on Figure 2 $(S, \preceq_S)$ is $(\mathbb{N}, \leq)$.

To model concatenation, we rely on H, the set of *concatenation functions* which is a particular subset of endomorphisms over $(S, \preceq_S)$. We use these functions to model how the metric associated with a route is changed when the route propagates over an arc. From now on we will call h_{uv} the concatenation function associated to the arc $(u, v) \in E$, as illustrated on Figure 2. On Figure 2, $H = \{h_{ab}, h_{bc}, h_{ce}, h_{ae}, h_{ad}, h_{de}, h_{ef}\}$. We also define two particular endomorphisms: $\infty_H : s \mapsto \infty_S$ the function that always returns ∞_S, and $0_H : s \mapsto s$ the identity function of S. Finally, we consider $\odot$ the reversed composition operator defined by

$$\forall h, h' \in H, h \odot h' = h' \circ h.$$

where $\circ$ denotes the usual function composition operator. This operator is useful to represent the metric associated to a path as the composition of the metrics of the arcs of the path. Then our routing protocol can be modeled by $\mathcal{A} = (H, S, \preceq_S, \odot)$, an algebraic structure referred to as an *endomorphism semi-ring* [14]. The key point of relying on such algebraic structures is that Gondran and Minoux [14] have shown that they allow generalizations of path algorithms, i.e. very general models of routing protocols.

[2]A pre-order is a reflexive and transitive binary relation. An order must also be anti-symmetric.

Gondran and Minoux [14] proposed two generalizations of Dijkstra's algorithm and proved they converge to an optimal state, leading to a shortest paths tree if $\mathcal{A}$ is an endomorphism semi-ring. In the direct generalization, $\preceq_S$ is required to be a total order. With this algorithm, each vertex is examined only once, in the same order as Dijkstra's algorithm is visiting vertices. In the greedy version of the algorithm, on the other hand, $\preceq_S$ is only required to be a pre-order. In this case however, vertices may be examined several times.

3.3 Properties enforced by MRPC timers

We want to build a timer $\tau_{uv} \in T$ for each arc (u, v), which depends on the path metric of the routes sent by u to v, where

$$T = \{\tau : S \rightarrow \mathbb{R}^+ \cup \{+\infty\}\}.$$

We distinguish 0_T, the null-timer that does not delay routes at all, and ∞_T, the infinite timer that filters a route, two particular timers such that: $\forall s \in S, 0_T(s) = 0; \forall s \in S, \infty_T(s) = +\infty$.

Moreover, we define the operator $+$ and the binary relation $\prec_T$ as follows:

$$\forall \tau, \tau', \tau" \in T, \tau" = \tau + \tau' \Leftrightarrow \forall s \in S, \tau"(s) = \tau(s) + \tau'(s);$$

$$\forall \tau, \tau' \in T, \tau \prec_T \tau' \Leftrightarrow \forall s \in S, \tau(s) < \tau'(s).$$

We are looking for a transformation which maps a concatenation function h_{uv} onto a timer function τ_{uv}. We denote by $\Delta_t : H \rightarrow T$ this transformation.

We now present properties we require Δ_t to satisfy. These properties ensure an ideal convergence behavior, i.e. no path exploration nor loop if the routing protocol can be modeled with an endomorphism semi-ring. They lead to a set of equations that allow us to compute MRPC timers.

For the interested reader, the proofs of all propositions stated in this section can be found in [22]. They are skipped in this paper due to space limitations, as well as not to burden readers uninterested in the mathematical details.

Arrival time ordering

PROPERTY 1. *Routing messages must reach a given router in the same order as its path preferences.*

We can prove that Property 1 is equivalent to:

$$\forall h, h' \in H, h \prec_H h' \Rightarrow \Delta_t(h) \prec_T \Delta_t(h')$$

where $\prec_H$ is the pre-order induced by $\prec_S$ over H. This gives us the first equation to compute MRPC timers. This proposition claims that any concatenation function can be transformed into a positive delaying function while preserving the $\prec_H$ ordering. Moreover, we can deduce from this theorem that messages following different paths of equal metrics must be equally delayed.

Timer summability

PROPERTY 2. *The propagation time along a path must be equal to the sum of the propagation times along each arc of the path.*

We can prove that Property 2 is equivalent to:

$$\forall h, h' \in H, \Delta_t(h \odot h') = \Delta_t(h) + \Delta_t(h').$$

This gives us the second equation used to compute MRPC timers. Additionally, we obtain two particular bounds from this property:

$$\Delta_t(0_H) = 0_T \text{ and } \Delta_t(\infty_H) = +\infty_T.$$

These bounds give us the third and final equation used to compute MRPC timers.

Confidentiality

PROPERTY 3. *A router must be allowed to define MRPC timers only based on its own routing policy and/or the metric related to the path of the received route.*

This property is critical, especially in today's Internet where routing policies play such an important role and must be kept confidential. A router must be able to delay routing messages according to its own routing policies and the path metric of the received route, without revealing its policies to its neighbours. This is an additional constrain we enforce for our MRPC timers, it is not necessary per se to compute MRPC timers.

We prove that the property can be satisfied whenever the timer τ_{uv} can be split into two separate timers:

1. An outgoing timer $\tau^+_{uv} = \Delta_t(h^+_{uv})$ applied by u to messages sent to v,

2. An incoming timer $\tau^-_{uv} = \Delta_t(h^-_{uv})$ applied by v to messages learned by u.

If routers u and v belong to different ASs, they do not have to share their routing policies. They simply apply their own.

3.4 Summary

Let $(H, S, \preceq_S, \odot)$ be an endomorphism semi-ring describing a routing algebra in a graph $G(V, E)$. Each router installs a timer per incoming and per outgoing arc, according to the transformation Δ_t. Δ_t is used by all routers of the network, and it transforms a routing policy (modeled by a concatenation function) into a timer, i.e. a function that maps a route to a delay. The transformation Δ_t generates MRPC timers if and only if the previous properties hold. These properties allows us to find the transformation Δ_t for a given instance of path algebra that models a given routing protocol. Section 4 describes how to build MRPC timers for several basic path algebras, as well as more complex ones.

4. MRPC TIMERS AND PATH ALGEBRAS

4.1 Base algebras

In the case of simple algebras as defined in metarouting [17], computing MRPC timers can be done directly based on the properties mentioned in Section 3.3. Table 1 presents the MRPC timers corresponding to the following algebras:

- $(\mathbb{R}^+ \cup \{+\infty\}, \leq, +)$ is the *usual* (additive) shortest path algebra.

- $([1, +\infty], \leq, \times)$ is the *multiplicative* algebra. We multiply metrics instead of adding metrics to compute path lengths. Packet losses on a path are the product of the losses on each arc of the path.

- $([0, 1], \geq, \times)$ is the usual *availability* algebra. A path is better if its availability is higher ($\geq$), and the availability of a path is the product ($\times$) of the availabilities of its arcs.

- $(\mathbb{R}^+ \cup \{+\infty\}, \geq, min)$ is the *bandwidth* algebra. A path is better if it has more bandwidth ($\geq$). The bandwidth of a path depends on the bandwidth of the bottleneck link (min).

- $(SEQ, \preceq_{SEQ}, .)$ is the *sequence* algebra. A path is described by a sequence, e.g., the identifiers of the routers crossed on the path to the destination. A path is shorter if its corresponding sequence length (obtained through $|\cdot|$) is shorter. The operator '.' consists in concatenating sequences carried by the path metric and the metric installed on the concatenated link. '.' may also return ∞_S if a loop is detected.

4.2 Neighbors preference classes algebra applied to BGP

Now, let us introduce routing policies as they are typically modeled in the literature. Here we provide one way of defining MRPC timers for such routing policies. The reader should be aware that MRPC timers are not tied to these routing policies. What matters for not break the endomorphism semi-ring property is to have a finite number of neighbour classes and that a strict and consistent ordering of those neighbours be used by all ASs. Let us explain what that means based on simplified routing policies similar to those in use in the Internet.

In the Internet, ASs are commonly connected through one of the following economical relationships:

- *Customer to provider ($c2p$)*: the customer pays its provider for both its incoming and outgoing traffic.

- *Peer ($peer$)*: the two connected ASs exchange for free the traffic having their customers as destination.

- *Provider to customer ($p2c$)*: the provider is paid for the traffic it sends or receives from its customer.

Therefore, an AS prefers sending its traffic to its customers, then to its peers and finally to its providers[3]. As a consequence, interdomain paths usually verify the following regular expression first presented in [13]: $(c2p)^*(peer)?(p2c)^*$, i.e. a path can be made of 0 or more $c2p$ relationships, followed by 0 or 1 $peer$ relationship, followed by 0 or more $p2c$ relationships.

Based on the previous regular expression, BGP routes are allowed to propagate on the following inbound-outbound arc pairs: $\{(c2p, c2p), (c2p, peer), (c2p, p2c), (peer, p2c), (p2c, p2c)\}$. These pairs induce MRPC timer values as shown in Table 2, where k is a global constant factor used by all ASs.

When the neighbour class on its inbound and outbound arcs are both $c2p$ (resp. $p2c$), then an AS immediately forwards routing announcements. On the other hand, the more different the preference between the neighbour classes of the inbound and outbound arcs, the more the routing messages are delayed. For instance, a message is delayed more if it crosses a $(c2p, p2c)$ arc pair (by $2.k$ time units) than a $(c2p, peer)$ or $(peer, p2c)$ pair (both k time units). Finally, routing messages are not propagated when local preferences pairs correspond to AS relationships combinations that violate the routing policies ($+\infty$ timer).

[3]These relationships are typically configured in BGP routers through the local-pref attribute, whose value reflects the interest for an AS to use a neighbouring AS to send traffic. An AS usually sets high local-pref values to routes learned from customers, intermediate values to routes learned from peers, and small values to routes learned from providers.

	Algebra	MRPC timer
Shortest paths	$(\mathbb{R}^+ \cup \{+\infty\}, \leq, +)$	$s \mapsto k.\lambda$
Multiplicative algebra	$([1, +\infty], \leq, \times)$	$s \mapsto k.ln(\lambda)$
Availability	$([0, 1], \geq, \times)$	$s \mapsto -k.ln(\lambda)$
Bandwidth	$(\mathbb{R}^+ \cup \{+\infty\}, \geq, min)$	$s \mapsto k.(s - \lambda)$ if $s - \lambda > 0, 0$ otherwise
Sequence	$(SEQ, \preceq_{SEQ}, .)$	$s \mapsto k.\lvert\lambda\rvert$
Neighbor preference classes	$(\mathbb{N}_k \cup \{+\infty\}, \geq, \otimes_{NPC})$	see Section 4.2

Table 1: Some base algebras defined in [17] and their corresponding MRPC timer. We denote by λ the metric installed on the arc. We denote by $k > 0$ a global constant factor used by the all routers of the network.

	$c2p$	$peer$	$p2c$
$c2p$	0	k	$2.k$
$peer$	$+\infty$	$+\infty$	k
$p2c$	$+\infty$	$+\infty$	0

Table 2: Example delay generated by an AS. This duration depends on the inbound AS relationship (line) and on the outbound AS relationship (column).

Such timers can be implemented using BGP communities [7], which are already widely used [10]. It consists in tagging incoming i (resp. outgoing j) eBGP routes according to the corresponding peering agreement. A router forwarding a route through an eBGP session would then examine the couple (i, j) and delay the BGP message according to the timers defined above.

The AS relationships model we have defined here is a simplified view of real policies implemented in the Internet [6, 28]. In particular, it does not take into account the case when one wants to distinguish between neighbours of a given type class (e.g. provider, customer, peer). A customer may want to distinguish between two providers for instance, one being its primary and the other used only as a backup. To take this into account the propagation pattern would need to be changed as $(c2p_{primary}|c2p_{backup})^*(peer)?(p2c)^*$ for instance, meaning a $c2p$ agreement can be either established with a primary *or* a backup provider. But because no strict preference can be enforced between $c2p_{primary}$ and $c2p_{backup}$ in such a pattern, the resulting algebraic structure would no longer have the required properties to ensure that no path exploration occurs. More precisely one cannot be sure that for a destination all paths from the primary would be received before those from the backup, which might cause exploration. We explain in section 4.2.1 how this drawback can be overcome.

We insist on the following fundamental point: there is nothing we can do to prevent path exploration if routing policies give too much freedom and do not allow to define a total order on the paths that propagate. This is not a limitation of MRPC timers, but of the ordering that can be expected from the routing messages propagation in a given routing protocol. We raise this point especially for the design of future Internet routing protocols. If convergence properties are considered important for the routing protocol, then being able to model the routing protocol with an endomorphism semi-ring would ensure that the amount of path exploration can be engineered with a mechanism such as MRPC timers.

4.2.1 Lexicographic product of MRPC timers

So far we have considered base algebras where the set S was defined over simple metric spaces. Some routing protocols, e.g. BGP, use multiple route metrics. They are ordered by importance and evaluated through a *decision process* according to a lexicographic comparison: the first attribute is used to rank the routes, then if several routes have the best metric value, the second attribute is used, and so on until a single best route remains. We show in this section how to build MRPC timers in complex algebras that model the lexicographic comparison, as introduced in [17,19]. For the interested reader, additional proofs related to this section can be found in [22].

Let us remind the reader that MRPC timers can be trivially found for all metrics used in BGP, except for local-pref and MED, thanks to base algebras (Section 4.1). In the case of local-pref used to implement routing policies, we refer to Section 4.2. For the MED attribute, it cannot be considered cleanly unless all MED values used inside an AS are known. We believe that it is better to let MED introduce some path exploration than trying to introduce timers that are based on wrong or incomplete information about the MED values that will be received from a neighbouring AS.

Let $\mathcal{A}_1 = (H_1, S_1, \preceq_1, \odot_1)$ and $\mathcal{A}_2 = (H_2, S_2, \preceq_2, \odot_2)$ be two endomorphism semi-rings. Let Δ_1 and Δ_2 be two MRPC timer transformations related to $\mathcal{A}_1$ and $\mathcal{A}_2$ and T_1 and T_2 the related timer sets. Assume that S_1 is a *discrete set* and that $S'_2 = S_2 \backslash \{\infty_{S_2}\}$ is a *bounded set*.

We call *quantum of* Δ_1, denoted q_1, the minimal difference between any two delays induced by finite timers over T_1 and *width of* Δ_2, denoted w_2, the maximal delay computed by any finite timer over T_2.

To preserve an adequate arrival ordering, we scale timers over T_1 such that the delays induced by timers over T_2 become negligible compared to those over T_1. Namely the resulted timers are as follows:

$$\tau = k_1.\Delta_1(h_1) + k_2.\Delta_2(h_2)$$

with $k_1 > 0$ and $0 < k_2 < k_1.q_1/w_2$.

For instance if we consider the first two steps of the BGP decision process[4], we are in this case. The algebra built from peering relationships is discrete and the length of AS paths can be bounded. For example, most AS paths in the Internet are shorter than 9 hops. Suppose you decide to delay messages by 0.1 time unit per AS hop in the AS path. Then timers over the relationships algebra must be such that the difference between the delays induced by any two of them is at least greater than 9×0.1.

One can choose to delay messages crossing $(c2p, c2p)$ or $(p2c, p2c)$ pairs by 0 time units, those crossing $(c2p, peer)$ or $(peer, p2c)$ pairs by 1 time unit and those crossing $(c2p, p2c)$ pairs by 2 time units. This ensures that for any destination all paths from customers are known before those of peers which are themselves known before those from providers, no matter the length of the AS path the message has been propagated on[5]. Then inside a same class of

[4]Prefer the path with the highest local-pref, if not unique then prefer the one with the shortest AS path length.

[5]As long as this path is shorter than 9 hops. Otherwise in this very example it might involve path exploration.

paths (those received from providers for instance), paths are known in the order of the the length of their AS path, from the shortest to the longest.

A possible solution to the drawback encountered in Section 4.2 would be to use AS path prepending. It consists in adding multiple times an AS number to the AS path, to make that path look worse. A customer having a primary and a backup link would for instance prepend paths received on the backup link. The effect of prepending is that the timers acting on the AS path length will delay more the prepended paths. Therefore, even if all paths from providers are delayed as they should, those received on the backup link would be more delayed than those received on the primary link. This would avoid exploration if the extra delay due to prepending is large enough.

4.2.2 Dealing with network propagation times

The propagation time between any two routers in the network in terms of medium transmission, queuing and computation is not null. Therefore, MRPC timers must be designed so that their duration is an order of magnitude larger than propagation times inside networks. One can model propagation times by an algebra S_{prop} which is bounded by w_{prop}, the worst propagation time between any two routers. As a consequence, as far as the routing protocol used in the network can be modeled by an algebra S_1, discrete or at least that can be discretized, then by using the lexical product, we can scale timers associated to S_1 in such a way that the arrival order they impose is not broken by propagation times.

Such an approach is reasonable as propagation and convergence inside an AS and across ASs generally take place at different timescales. Today, MRAI timers for eBGP sessions are themselves an order of magnitude larger than those on iBGP sessions. It must be noticed however that using our formalism further steps could be modeled considering the lexicographic product of more than only two algebras, similarly to [19]. If an AS does not want to include intradomain details in its MRPC timers, it may just include this part within the network propagation time as we do. If the timers are properly designed, the only drawback should be some path exploration occurring *inside* the AS.

5. MRPC VS. MRAI TIMERS

In this section, we want to show that it would be beneficial to replace the current MRAI timers in BGP by MRPC timers. Even if simplified routing policies and not all steps of the BGP decision process are used to design MRPC timers, they will improve convergence properties quite a lot, even though they may not improve much the total convergence time. We remind the reader that MRPC timers are not designed to make convergence fast, but to ensure that no path exploration will occur in the ideal case, or that a bit of path exploration will occur if the endomorphism semi-ring properties do not hold. As will be explained in Section 6.2, MRPC timers can also be used in link state routing protocols to prevent micro-loops from appearing during FIB updates.

The current default values of MRAI in the Internet are defined in RFC4271 [33]. The suggested value for MRAI on eBGP sessions is 30 seconds, while it is 5 seconds on iBGP sessions. These values have been found to be too high, both empirically [25] and based on simulations [15]. Recently, [20] has suggested the use of MRAI values of 5 seconds or less on eBGP sessions, and 1 second or less on iBGP sessions. As pointed out at the IETF in response to [20], the issue of MRAI is not its value, but the fact that one value does not fit all situations [15]. We will confirm in this section that whatever the actual value of MRAI used, the basis fact remains that MRAI is a mechanism that does not explicitly address path exploration and the number of routing messages exchanged during convergence. A proper ordering of the routing messages during convergence is necessary to keep low the number of messages exchanged in all situations.

We have developed a simulator that computes routing message propagation and that triggers each routing event at a specified time. We have built an Internet AS graph made of $29,146$ ASs and $78,934$ links, as observed by trace collectors [1], and inferred the AS relationships [2]. We assume that each AS implements a routing policy consistent with the inferred AS relationships.

Keep in mind that the sole purpose of our simulations is to compare MRPC timers to MRAI to understand their respective performance. We cannot expect to capture with simulations the full complexity of routing protocols behaviour in the Internet. We only use the simulations as a way to compare two different timer mechanisms on the same topology.

We install a random constant delay belonging to $[0, 1[$ seconds for each AS router. This random delay models the propagation time required to traverse a given AS. We run four different sets of simulations:

1. *MRAI-30:* At the beginning of this simulation, we set for each AS a random value belonging to $[0, 30[$ seconds corresponding to the next expiration time of its MRAI timer. Then, every 30 seconds, its MRAI expires and the AS forwards its best route if it has been improved during the last MRAI round.

2. *MRAI-5:* The current recommended value in RFC4271 [33] for MRAI is 30 seconds on eBGP sessions, so the higher bound of 30 seconds was chosen in the previous setting. The recommended MRAI values have recently been revised in [20] to be 5 seconds or less for eBGP sessions and 1 second or less for iBGP sessions. In this setting we do similarly as in MRAI-30, but the first expiration time of the timer belongs to $[0, 5[$ instead of $[0, 30[$. Then the MRAI timer expires every 5 seconds.

3. *MRPC-10-20:* We install on each eBGP session an MRPC timer. Such a timer is obtained by using the lexical product (see Section 4.2.1) with the local preference algebra and the AS path algebra (see Section 4). In order to bound the AS path algebra, we decide to set the maximal AS path length to 10. If some AS paths are longer than this bound, then the only consequence is that some path exploration may occur. This value and the upper bound of the propagation time between two ASs (set to 0.1 s) therefore gives us the duration of the different timers. First, an AS delays a message 1 second per AS hop in the AS path. Then, depending on its inbound-outbound AS relationship pair, an AS will add an extra delay of 10 seconds (community belonging to $(c2p, peer)$ or $(peer, p2c)$) or 20 seconds (community belonging to $(c2p, p2c)$).

4. *MRPC-5-10:* The MRPC-10-20 setting is actually not comparable in convergence time to the MRAI-5 one that uses far smaller timers. The latter leads to smaller convergence times. MRPC timers can be scaled down almost arbitrarily, by changing how much delay is added per AS hop and per inbound-outbound AS relationship pair. The MRPC-10-20 delays are divided by two in this experiment, with 0.5 second per AS hop and 5 ($(c2p, peer)$ or $(peer, p2c)$) and 10 seconds ($(c2p, p2c)$) for AS relationship pairs.

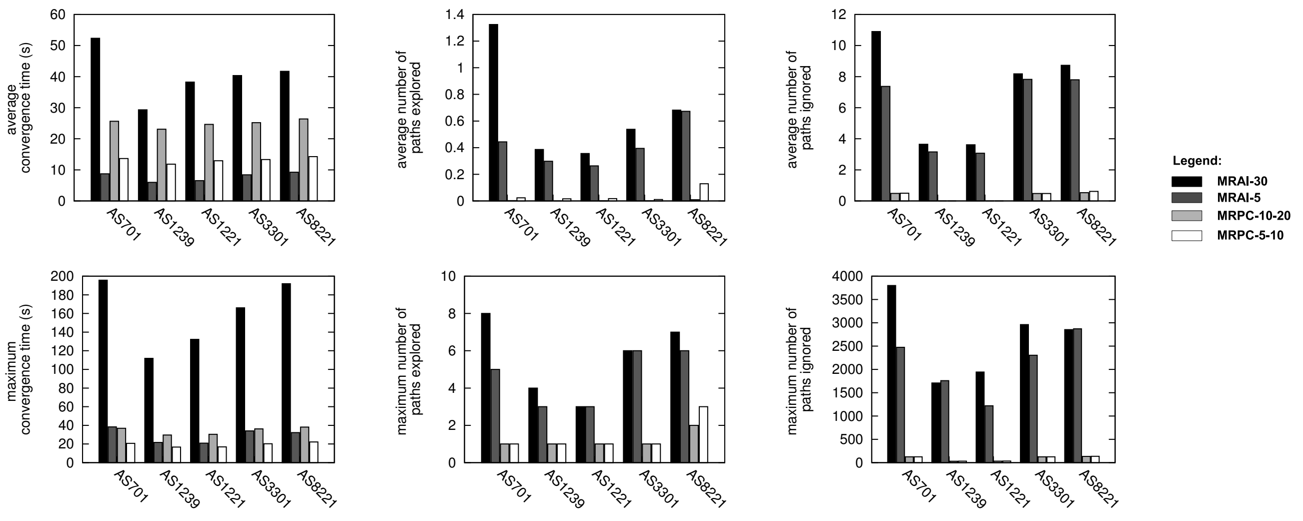


Figure 5: Comparison of convergence properties of MRAI and MRPC timers.

We focus on three aspects of convergence: network convergence time, number of explored paths, and number of ignored paths. The network convergence time is the time it takes for all routers in the AS-level topology to converge. The number of explored paths refers to the number of transient paths chosen as best by the routers and thus propagated by routers. The number of ignored paths refers to the number of paths a router has learned but never selected as best, before it converges towards its final best. We run our simulations by originating a prefix in several ASs, including tier-1 (such as AS701 and AS1239), tier-2 (such as AS1221 and AS3301), and stub ASs (such as AS8221), and measuring convergence time, explored paths, and ignored paths in the whole AS-level topology.

Figure 5 compares the four sets of simulation, by showing results for prefixes originated in the previously mentioned ASs and measuring the three convergence-related metrics across the whole AS-level topology. The AS numbers of the x-axis of Figure 5 represent the AS from which we originate a prefix. Let us first focus on results for MRAI. The *MRAI-5* setting leads to far smaller convergence times (average and maximum) than *MRAI-30*. We can appreciate the particularly bad convergence times of the *MRAI-30* timers, consistent with observations in the Internet [21]. *MRAI-5* leads to pretty fast network convergence, even smaller than MRPC timers for our settings on average (but not in the worst case). The drawback of MRAI, whatever the value of the timers, can be seen in the number of paths explored as well as the number of paths ignored. MRAI tries to prevent transient paths by waiting before sending messages. The global effect is actually to slow down all messages, both those that will not be selected as best at the end of convergence and those that will be selected as best at the end of convergence. This is a very ineffective strategy, as only the paths that will not be selected as best at the end of convergence should be delayed. With MRAI, the number of paths a router learns before learning its best final path is large, especially in the worst-case (lower right graph of Figure 5).

If we turn to MRPC timers, we observe that they perform systematically better than MRAI in terms of path exploration and ignored paths (middle and right graphs of Figure 5), as they should. On average, a router receives as first message the best route he will ever learn, i.e. without incurring path exploration. Even in the worst-case, most routers will explore only a very small number of transient paths. However, path exploration occurring with MRPC timers does not have a cascading effect as with MRAI, as such transient paths have almost no chance to propagate very far unless these paths are good enough with respect to the path metrics and the routing policies of the routers that receive them. The number of ignored paths illustrates best this point: with MRAI, exploration triggers the propagation of a large number of paths (sometimes thousands) that will never be selected as best by some routers (right graphs of Figure 5). These paths are typically replaced by better paths within a short period of time. Note that the large number of ignored paths is related to the routing diversity available in the Internet [27, 34, 36].

With respect to convergence time, the performance of MRPC timers depends mostly on the choice of the delays set per AS hop and per policy. The convergence of *MRPC-5-10* is exactly half the convergence time of *MRPC-10-20*. This illustrates the impact of scaling MRPC timers. When scaling MRPC timers to too low values, some path exploration may happen due to the stochastic nature of BGP pass-through times [11]. However, the path exploration allowed is still very limited and cannot escalate as it does with MRAI.

6. DEALING WITH ARBITRARY TOPOLOGICAL DYNAMICS

6.1 Incremental protocols

To make the concept of MRPC timers easier to understand, we have so far only focused on the case of a destination announced to the network. We now explain how arbitrary failures can be handled with our mechanism. The injection of a shorter path in the network (link or router restoration, metric decrease) is equivalent to the previous case. It corresponds to the spread of a new valid path from one router and can only lead to some routers electing this new path as shortest. Therefore routers that select new paths that contain this new path are still updated in the right order. As a consequence, in this situation we do not have to modify our mechanism. On the other hand, the deletion of a shortest path from the network (link or router failure, metric increase) must be handled somehow differently. In that case, obsolete routes need to be purged from the network, so that we are back to the basic case.

Let us consider the network topology from Figure 6. Each router has several routes to reach destination router a, and uses as best the route labeled with '>'. The other routes will be stored as backup paths and will be used only if the best route fails.

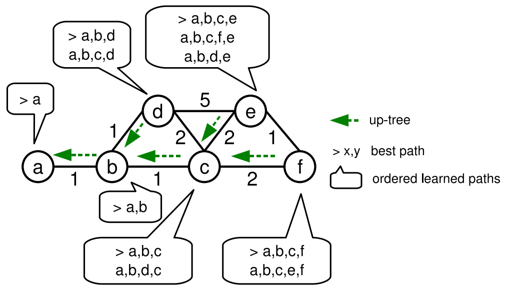


Figure 6: Shortest paths to router a.

Assume now that a link failure happens in the network, for example link (b, c) as represented on Figure 7. Given the way they are impacted by this event, routers can be divided into three categories:

- *Safe routers:* The routers which are not impacted by the topological event. These routers are shown in white on Figure 7.
- *Originator routers:* The impacted routers which are still able to compute a valid path on their own. These are the routers which are going to originate the spread of backup paths into the impacted topology. They are represented in grey on Figure 7.
- *Empty routers:* The impacted routers which have to wait for new routing information to compute a new valid path. They are represented in black on Figure 7.

To avoid path exploration in such a situation two main issues must be overcome.

Flushing obsolete paths.

Obsolete paths must be purged from the network. In path vector protocols, some obsolete paths might be present in the routers Routing Information Bases[6]. This is achieved by (i) letting routers announce explicit withdraw messages to each other without delaying them and (ii) scaling MRPC timers so that they do not expire before all the explicit withdraws have been propagated in the network. We can reasonably assume that the time it takes for the explicit withdraws to propagate will be very small compared to the convergence time. Even if this is not the case in some parts of the network, the only drawback will be a small amount of exploration.

Synchronizing originators.

Originators have to be "synchronized" as if the backup paths propagation had been initiated by the destination. This is achieved by letting routers wait for an extra delay, called *synchronization delay*. Routers compute it by applying their MRPC timer on the metric of their new path to the sink. However, this requires that routers are able to detect that they have switched to a worse path. To do so, we propose that before installing a path to a destination in its RIB[7], a router compares the path to be installed with the one already in the RIB. If the new path to be installed is worse, for example in the case of a shortest path deletion somewhere in the network, then the new path is not immediately installed. Instead the router waits for an amount of time equal to the *synchronization delay* before installing the new shortest path in its RIB. However, because obsolete paths are flushed, *empty* routers have no paths for some destinations until they receive new ones from $originators$. During this time *empty* nodes may therefore undergo some traffic loss. To prevent packet loss we need a default value of *synchronization delay* to be defined for the case of a router having no path to the destination. $Empty$ routers would keep on forwarding packets on their old best path to the sink s until the expiration of the *synchronization delay*. The key point to understand why traffic loss can be reduced by doing so is that by construction our method ensures that during re-convergence a router in the network can only be in two states: either (1) it is using its old shortest path or (2) it has a new valid path (meaning it has been updated). Therefore, the packets forwarded by an *empty* node u on its old shortest path will either only cross *empty* routers and therefore traffic might be lost in case of link failure on this path for instance. Otherwise there is a router v on the old shortest path of u which has been updated or is at least an $originator$ and therefore the traffic will be correctly forwarded from v to the sink.

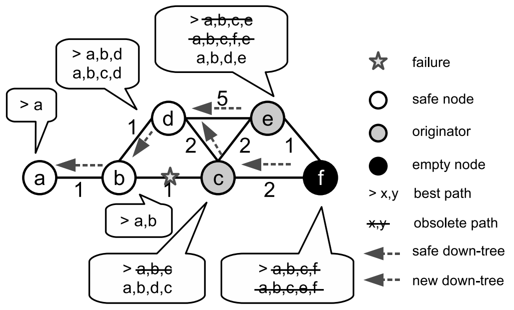


Figure 7: Routers c and e originate the backup paths propagation.

This behaviour is illustrated on Figure 7. As previously, we consider a path vector routing protocol using the usual shortest path algebra, where routers only forward their best path to their neighbours. After the failure of link (b, c), explicit withdraws are propagated across the network without delays. Once this is done, routers e and c are *originator nodes* while router f is an *empty node*.

Router e compares its new shortest path to router a, (a, b, d, e), whose weight is 7, with the previous shortest path, (a, b, c, e), whose weight is 4. The new path is worse than the previous one, hence router e waits for $7 \times k$ (the *synchronization delay* computed on the metric associated to the path (a, b, d, e)) before installing this new path in its RIB, then it waits $1 \times k$ time units (as the value of its MRPC timers as the weight on the arc (e, f) is 1) before announcing the path to f. The resulting delay induced by router e is $8 \times k$ time units. Router c does the same and obtains a delay of $4 \times k$ time units. As a consequence, router f learns its new best path (a, b, d, c, f) first. What is more as far as c is up-to-date, then the traffic of f indeed reaches a even if f has not been updated yet.

With this add-on to the mechanism, dynamics is handled in the same way as in the nominal case. Once obsolete paths have been flushed and if timers are large enough compared to the propagation time of withdraws, no path exploration nor loops appear. Traffic loss may also be reduced.

[6]For example, BGP maintains special tables (Adj-RIB-in's) which contain the paths announced by neighbours. A router then elects for each destination the best path and inserts it in its routing table. In the case its best path is withdrawn, it will recompute a new best path based on the routes in its Adj-RIB-in's, which may be obsolete.

[7]A RIB is the set of best routes chosen by the protocol to each each destination.

6.2 Flooding protocols

Flooding protocols do not suffer from path exploration but are known to be impacted by forwarding loops during convergence. To avoid them, routers must be updated in the right order for any destination. This means that for each shortest path tree, routers must be updated from the sink to the leaves.

With a flooding protocol, routing messages contain information about link states and are not related to some particular destination. However, it is possible to reproduce in flooding protocols a behaviour similar to the one of incremental protocols. Instead of delaying routing messages, we delay the time when a route is installed in the forwarding information bases[8] (FIB). The routing messages are then forwarded as is usually the case in flooding protocols.

When a path is changed, its installation in the FIB is done according to the MRPC timer related to the total cost of the path from the router to the sink. Its installation in the FIB is independent from the path followed by the routing message. If a router u_p received a routing message that requires it to use a new shortest path to reach u_1, denoted by $\mu = (u_1, u_2, \ldots, u_p)$, then u_p installs this path in its FIB after an amount of time equal to

$$t_\mu = \sum_{i=1}^{p-1} (\Delta_t(|u_i, u_i + 1|)).$$

By nature of flooding protocols, during re-convergence a router in the network can only be in two states: either (1) it is using its old shortest path or (2) it has a new valid path (meaning it has been updated). What we enforce with MRPC timers is that routers are updated for any destination from the sink to the leaves. The traffic can therefore only either be dropped or correctly forwarded to the destination just like in section 6.1, but no loops should appear.

7. RELATED WORK

Many works investigate improvements to BGP convergence. Most have focused on obsolete path detection in order to avoid their propagation in the network [3, 8, 30, 31, 38]. This is most of the time achieved using heuristics on the AS path attribute [3, 30, 31]. To overcome lack of information about convergence, many authors also propose modifications to the protocol by adding extra information in BGP updates [8, 30, 31, 38]. For instance, [8] adds information to the AS path attribute to identify dependencies between paths to distinguish between valid and invalid paths. [38] on the contrary tends to prevent convergence from malign instability by rejecting obsolete paths through fault information carried in BGP updates.

Other works have tackled the issue of transient paths exploration. For instance, [4] proposes to delay BGP route propagation according to the carried AS path, but the authors only consider the AS path length decision step. [23] presents a dynamic adjustment of the MRAI timers based on the activity (number of messages received) related to this destination. In [35], the authors suggest to forward firstly routing messages along the *current* covering shortest path tree. They propose a heuristic computed by each router in order to infer which neighbours cross it to reach the destination. As routing messages are diffused on the obsolete shortest path tree, path exploration cannot be avoided, especially in the case when a new link or a new router appears.

[8]A Forwarding Information Base (FIB) in a router is a table that keeps mappings between destinations and an output interfaces where the packet has to be sent.

8. CONCLUSION

Internet routing convergence has been shown to be problematic since almost a decade [18, 21]. In this paper, we explained how to *improve the convergence* of routing protocols through timers that enforce a proper ordering of the routing messages, therefore limiting path exploration to a minimum. We designed such timers, called MRPC, for *metrics and routing policy compliant*. MRPC timers depend only on the path metrics of the routes received by a router and its routing policies. No knowledge about routing policies from other ASs is assumed in our solution.

We explained under which conditions MRPC timers can be computed in routing protocols that rely on single and multiple metrics. We studied the expected gain of MRPC timers in terms of network convergence properties against the current mechanism used in BGP, MRAI. We showed that MRPC timers significantly reduce the number of exchanged messages. Furthermore, they can be scaled down to keep convergence time low, while also keeping low the number of exchanged messages. Finally, and most important, our approach is generic, in that it applies to a large class of routing protocols, including existing link state and path vector protocols. Our solution works on the ordering of the of the routing messages exchanged in incremental protocols, while in flooding protocols FIB updates are updated in a specific order.

So far, routing algebras have been largely focused on the comparison and concatenation mechanisms. In this paper, we worked on the scheduling mechanism. The fourth mechanism, diffusion, has still to be studied. Current routing protocols rely on manually configured diffusion topologies to propagate the routes across the Internet. Such diffusion topologies are hard to design as many different considerations are relevant, like correctness, optimality, and scalability. Further work towards establishing a firm basis for diffusion mechanisms that adapt to topological changes and retain certain properties, especially scalability, should lead to routing protocols that have far better properties than the current ones.

Acknowledgments

Part of this work has been funded by the G-lab project, a research project of the DFG in Germany (support code 01 BK 0805).

9. REFERENCES

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

[2] BATTISTA, G. D., ERLEBACH, T., HALL, A., PATRIGNANI, M., PIZZONIA, M., AND SCHANK, T. Computing the types of the relationships between autonomous systems. *IEEE/ACM Trans. Netw. 15*, 2 (2007).

[3] BREMLER-BARR, A., AFEK, Y., AND SCHWARZ, S. Improved BGP convergence via ghost flushing. In *Proc. of IEEE INFOCOM* (2003).

[4] BREMLER-BARR, A., CHEN, N., KANGASHARJU, J., MOKRYN, O., AND SHAVITT, Y. Bringing order to BGP: decreasing time and message complexity. In *Proc. of ACM PODC* (2007).

[5] BUSH, R., GRIFFIN, T., AND MAO, Z. Route flap damping: harmful? NANOG presentation, `www.nanog.org/mtg-0210/ppt/flap.pdf`, October 2002.

[6] CAESAR, M., AND REXFORD, J. BGP Routing Policies in ISP Networks. *IEEE Network Magazine* (2005).

[7] CHANDRA, R., TRAINA, P., AND LI, T. BGP Communities Attribute. IETF RFC1997, August 1996.

[8] CHANDRASHEKAR, J., DUAN, Z., ZHANG, Z., AND KRASKY, J. Limiting path exploration in BGP. In *Proc. of IEEE INFOCOM* (2005).

[9] DESHPANDE, S., AND SIKDAR, B. On the impact of route processing and MRAI timers on BGP convergence times. In *Proc. of IEEE Global Telecommunications Conference (GLOBECOM)* (December 2004).

[10] DONNET, B., AND BONAVENTURE, O. On BGP communities. *SIGCOMM Comput. Commun. Rev. 38*, 2 (2008).

[11] FELDMAN, A., KONG, H., MAENNEL, O., AND TUDOR, A. Measuring BGP pass-through times. In *Proc. of Passive and Active Measurement conference* (2004).

[12] FRANCOIS, P., AND BONAVENTURE, O. Avoiding transient loops during the convergence of link-state routing protocols. *IEEE/ACM Trans. Netw. 15*, 6 (2007).

[13] GAO, L. On inferring Autonomous System relationships in the Internet. *IEEE/ACM Trans. Netw. 9*, 6 (2001), 733–745.

[14] GONDRAN, M., AND MINOUX, M. *Graphs, Dioids and Semirings: New Models and Algorithms*. Springer-Verlag, 2008.

[15] GRIFFIN, T., AND PREMORE, B. An experimental analysis of BGP convergence time. In *Proc. of IEEE ICNP* (November 2001).

[16] GRIFFIN, T., SHEPHERD, F., AND WILFONG, G. The stable paths problem and interdomain routing.

[17] GRIFFIN, T., AND SOBRINHO, J. Metarouting. In *Proc. of ACM SIGCOMM* (2005).

[18] GRIFFIN, T., AND WILFONG, G. An analysis of BGP convergence properties. In *Proc. of ACM SIGCOMM* (September 1999).

[19] GURNEY, A., AND GRIFFIN, T. Lexicographic products in metarouting. In *Proc. of IEEE ICNP* (2007).

[20] JAKMA, P. Revised default values for the BGP 'Minimum Route Advertisement Interval'. Internet draft, draft-jakma-mrai-02.txt, work in progress, November 2008.

[21] LABOVITZ, C., AHUJA, A., BOSE, A., AND JAHANIAN, F. An experimental study of Internet routing convergence. In *Proc. of ACM SIGCOMM* (August 2000).

[22] LAMBERT, A., BUOB, M., AND UHLIG, S. Proofs for *Improving Internet-wide routing protocols convergence with MRPC timers*. `http://sites.google.com/site/mrpctimers/`.

[23] LASKOVIC, N., AND TRAJKOVIC, L. BGP with an adaptive minimal route advertisement interval. In *Proc. of IEEE IPCCC* (2006).

[24] LEE, S., YINZHE, Y., NELAKUDITI, S., ZHANG, Z.-L., AND CHUAH, C. Proactive vs reactive approaches to failure resilient routing. In *Proc. of IEEE INFOCOM* (March 2004).

[25] MAENNEL, O., TUDOR, A., FELDMANN, A., AND BÜRKLE, S. Observed properties of bgp convergence. RIPE45 presentation, `www.ripe.net/ripe/meetings/ripe-43/presentations/ripe43-routing-flap.pdf`, May 2003.

[26] MAO, Z., GOVINDAN, R., VARGHESE, G., AND KATZ, R. Route flap damping exacerbates Internet routing convergence. In *Proc. of ACM SIGCOMM* (2002).

[27] MÜHLBAUER, W., FELDMANN, A., MAENNEL, O., ROUGHAN, M., AND UHLIG, S. Building an AS-Topology Model that Captures Route Diversity. In *Proc. of ACM SIGCOMM* (2006).

[28] MÜHLBAUER, W., UHLIG, S., FU, B., MEULLE, M., AND MAENNEL, O. In Search for an Appropriate Granularity to Model Routing Policies. In *Proc. of ACM SIGCOMM* (2007).

[29] NELAKUDITI, S., ZHONG, Z., WANG, J., KERALAPURA, R., AND CHUAH, C.-N. Mitigating transient loops through interface-specific forwarding. *Compututer Networks 52*, 3 (2008), 593–609.

[30] PEI, D., AZUMA, M., MASSEY, D., AND ZHANG, L. BGP-RCN: improving BGP convergence through root cause notification. *Computer Networks 48*, 2 (2005), 175–194.

[31] PEI, D., ZHAO, X., WANG, L., MASSEY, D., MANKIN, A., WU, S., AND ZHANG, L. Improving BGP convergence through consistency assertions. In *Proc. of IEEE INFOCOM* (2002).

[32] RAY, S., GUÉRIN, R., AND SOFIA, R. Distributed path computation without transient loops: An intermediate variables approach. In *Proc. of International Teletraffic Congress* (June 2007).

[33] REKHTER, Y., AND LI, T. A border gateway protocol 4 (BGP-4). IETF RFC4271, January 2006.

[34] SAVAGE, S., COLLINS, A., HOFFMAN, E., SNELL, J., AND ANDERSON, T. The End-to-End Effects of Internet Path Selection. In *Proc. of ACM SIGCOMM* (1999).

[35] SUN, W., MAO, Z., AND SHIN, K. Differentiated BGP update processing for improved routing convergence. In *Proc. of IEEE ICNP* (2006).

[36] TEIXEIRA, R., MARZULLO, K., SAVAGE, S., AND VOELKER, G. In search of path diversity in ISP networks. In *Proc. ACM IMC* (2003).

[37] VILLAMIZAR, C., CHANDRA, R., AND GOVINDAN, R. BGP Route Flap Damping. IETF RFC2439, November 1998.

[38] ZHANG, H., ARORA, A., AND LIU, Z. G-BGP: Stable and fast convergence in the Border Gateway Protocol. Tech. rep., June 2003. `http://www.cse.ohiostate.edu/~zhangho/publications/G-BGP-TR.pdf`.