Title

BGP-ELF: Enhancing BGP To Eliminate Routing Loops and Oscillations without Using Path Vectors

Author

Garcia-Luna-Aceves, J.J.

Publication Date

2022-10-01

Data Availability

The data associated with this publication are within the manuscript.

Peer reviewed

BGP-ELF: Enhancing BGP To Eliminate Routing Loops and Oscillations without Path Vectors

J.J. Garica-Luna-Aceves
Computer Science and Engineering Department, University of California, Santa Cruz, CA, USA
jj@soe.ucsc.edu

***Abstract*—The policy mechanisms used in BGP are modified by substituting path vectors with path labels, which results in BGP-ELF (BGP Enhanced for Loop Freedom). BGP-ELF uses updates, queries, and replies based on path labels to attain multi-path loop-free and stable routing across autonomous systems without the need for path vectors. An autonomous system organized into clusters is required to elect a designated route reflector to ensure that no routing loops can be formed due to local preferences within clusters in the autonomous system. Well-known examples of systems in which EBGP and IBGP do not converge are used to illustrate the benefits of BGP-ELF.**

I. Introduction

The Internet is organized hierarchically into autonomous systems (AS) that consist of collections of routing prefixes under the control of a single administrative authority or domain. The topology of an AS consists of one or multiple computer networks connected with each other, and the same set of routing policies is used throughout the AS. There are many protocols for routing within ASes, but the Border Gateway Protocol (BGP) [15] is the only protocol being used for inter-AS routing.

The Exterior Gateway (EGP) [16] was the first protocol for routing across ASes but its reliance on an engineered Internet topology to operate without looping made it unacceptable as the Internet grew in size. To address the limitations of EGP, the design of BGP incorporated the use of path vectors to detect the occurrence of routing loops across ASes. In theory, the use of path vectors enables all routers executing the routing protocol to converge to valid routes to destinations; however, BGP speakers use local preferences rather than a global optimality criteria to select paths, which results in looping and non-termination problems.

Griffin et al. [9], [10] have shown that relying on programs that verify ahead of time that routing policies do not contain policy conflicts that could prevent BGP from converging to stable routes is not practical. Consequently, only approaches based on modifications to the policy mechanisms or signaling of BGP are viable in practice. We describe many of these approaches in in [7], [8]. Related to the approach we describe, van Beijnum et al. [17] presented an approach to support multi-path routing in BGP by requiring BGP routers to communicate the routes with the longest AS-paths among the routes locally available for each destination.

The looping and non-convergence problems in interior BGP (IBGP) has been addressed by many authors (e.g., [1], [13], [14], [18]), and as a result the BGP specification has been augmented [2] to account for the use of route reflectors and has added more path attributes in an attempt to avoid routing loops due to route reflectors. However, the proposed operation of BGP with route reflectors in [2] is still prone to route oscillations and loops.

Previous approaches attempting to solve the IBGP routing problems in large ASes that are not fully meshed have focused on either properly configuring ASes (e.g., [14]), or requiring BGP speakers to communicate much more path information [1], [13], which may induce excessive overhead.

Interestingly, all prior approaches focusing on solving the looping and convergence problems of BGP assume the use of path vectors. This is also the case for our own recent proposal to make BGP stable and loop-free by introducing total ordering in its policy-based mechanisms [6], [8]. By contrast, this paper presents the first proposed redesign of BGP that eliminates path vectors and provides stable, loop-free, multi-path routing across ASes.

Section II presents **BGP-ELF** (*BGP Enhanced for Loop Freedom*), which replaces the use of path vectors with path labels. A path label consists of the number of AS hops and the AS identifier of the first hop in the route to the destination. BGP-ELF allows route selection to be based on local preferences as in BGP, and routers from an AS can report a single route to neighbor routers in other ASes even when they use multiple local routes to destinations. Loop-freedom is guaranteed by requiring routers in an AS to accept a route to a destination from a neighbor AS only when the path label reported by that AS is lexicographically smaller than the path label currently held by the routers in the AS. ASes use a query-response process to search and find routes with path labels that are lexicographically smaller.

Section III discusses cases of route oscillation and non-deterministic convergence in BGP and how BGP-ELF eliminates these problems. Section IV summarizes our results.

II. BGP-ELF

A. Overview

BGP-ELF uses the same signaling and policy mechanisms defined for BGP. Accordingly, we only describe the changes needed to transform BGP into BGP-ELF.

We assume familiarity with BGP and the way in which IBGP and EBGP routers operate [2], [3], [11], [15], and describe the policy mechanisms for routing used in BGP as consisting of: (a) An import transformation with which routes

are accepted for consideration, (b) a preference function with which valid routes are compared and preferred routes are selected, and (c) an export transformation with which preferred routes are announced.

BGP-ELF makes four main changes to BGP signaling: (a) Path vectors are replaced with *labeled path lengths* specified in Definition 1; (b) ASes are classified into classes to allow route filtering based on them; (c) a large AS organized into clusters uses one of the route reflectors in the AS [3] as a *designated reflector* to order routes within the AS; and (d) BGP-ELF uses updates, queries, and replies that take advantage of labeled path lengths and AS classes.

Definition 1: **Labeled Path Length:** The labeled path length of $P_d^k(n)$ is denoted by $\ell_d^k(n)$, is assigned by the routers in AS k, and is defined to be the tuple $(k, h_d^k(n))$, where $h_d^k(n)$ is the number of AS hops in $P_d^k(n)$.

This definition transforms a path vector to its hop length and the identifier of the first node along the path. By definition, $\ell_o = (D, 0)$ is the initial labeled path length associated with a known reachable destination d, where D is the AS of destination d, and $\ell_\infty = (nil, \infty)$ is the labeled path length for an unreachable destination.

BGP-ELF uses updates, queries and replies to ensure that the following condition $\boldsymbol{L}$ is always satisfied between a BGP-ELF speaker and any one of its next hops to a destination d.

Definition 2: **Ordering on Labeled Path Lengths:** Node a is ordered along path $P_d^a(n)$ with respect to its next-hop node b along that path if

$$\boldsymbol{L}: \ \ell_d^b(m) \prec_\ell \ell_d^a(n) \ \equiv \tag{1}$$

$$[h_d^b(m) < h_d^a(n)] \vee [(h_d^b(m) = h_d^a(n)) \wedge (b < a)]$$

AS Classes: BGP allows BGP speakers to filter routes based on path information [3] to avoid cases in which an AS forwards traffic towards a destination that does not satisfy its local routing policies. BGP-ELF allows route filtering based on the type of ASes traversed in a route to a destination. For this purpose, BGP-ELF defines a number of AS classes, an AS may be part of one or multiple classes, and nodes add a class vector to each labeled path length to state all the AS classes of the ASes traversed in a route to a destination.

Designated Reflector: Each AS organized into clusters with route reflectors has a single route reflector that is either configured or elected to be the designated reflector for the AS. If the designated reflector is elected, the election can be made very simple by choosing, for example, the reflector with the smallest identifier or the smallest. cluster identifier as the designated reflector. This can be done very quickly given that reflectors should be fully meshed with one another.

BGP-ELF Signaling: Nodes simply send updates with their labeled path lengths as long as they have neighbor nodes that satisfy $\boldsymbol{L}$ for a given destination. Otherwise, a node sends a query stating its current labeled path length to a destination and a *requested label* that equals the value of its own labeled path length prior to the input event that prompted the query.

A node that receives a query sends a reply if its next hop along the path corresponding to its reported labeled path length satisfies $\boldsymbol{L}$ with the value of the requested label stated in the query. The reply from the node states its own labeled path length and the requested label in the query. Otherwise, the node propagates the query specifying its own labeled path length and the requested label in the query it received.

A query is propagated towards the destination along the path corresponding to the reported labeled path lengths.

A node that forwards a reply states its own labeled path length and the requested label in the response it receives. Updates and replies are sent to all neighbors, and queries may be sent to all neighbors or a single neighbor.

Like BGP, BGP-ELF consists of Exterior BGP-ELF (E-BGP-ELF) with which BGP-ELF speakers in different ASes share routing information, and Interior BGP-ELF (I-BGP-ELF) with which BGP-ELF speakers in the same AS share routing information. We describe more details about the operation of BGP-ELF based on this two components. We have proven that E-BGP-ELF is loop-free and that it converges deterministically to stable routes [7].

B. Exterior BGP-ELF [7]

BGP-ELF advertises one single route to any given destination d if it has at least one loop-free path to the destination, and sends the same routes to all or a subset of neighbor routers in other ASes.

The labeled path length for destination d reported by the routers in AS k is denoted by $\ell_d^k[r]$, and defined to be $\ell_d^k[r] = (k, h_d^k)$, where h_d^k is the number of AS hops in the path to d. For simplicity, $\ell_d^k[r]$ is called the **reported label** by node k for destination d.

Because each router in an AS can advertise at most one route to any destination, a router in AS k cannot have more than one route to destination d through a neighbor router in another AS q. We denote by ℓ_{dq}^k the reported label for destination d sent by a router in AS q and maintained at the routers in AS k.

Node node k maintains a Neighbor Table (NT^k) and a Routing Table (RT^k). NT^k stores the reported labels sent by each neighbor of node k. RT^k lists an entry for each destination d and states: The reported label ($\ell_d^k[r]$), a reference label (r_d^k), the set of next hops (S_d^k), and the next hop (s_d^k) along the path corresponding to $\ell_d^k[r]$. If there is no next hop to d, then $S_d^k = \emptyset$ and $s_d^k = 0$.

The value of the **reference label** r_d^k equals the value of $\ell_d^k[r]$ when node k has valid next hops to destination d, or the smallest value of a requested label stated in a query created or forwarded by the node. How a router in AS k uses the data in RT^k to populate its forwarding information base is outside the scope of this paper.

An update for destination d is denoted by $U(d, \ell_d^k[r])$; a query is denoted by $Q(d, \ell_d^k[r], \rho_d^k)$, where ρ_d^k is a labeled path length stated by the AS from which the the query originated; and a reply is denoted by $R(d, \ell_d^k[r], \rho_d^k)$, where ρ_d^k is copied

from the query being answered. For simplicity, we refer to ρ_d^k as a **requested label**.

1) BGP-ELF Import Transformation: BGP-ELF uses $\boldsymbol{L}$ rather than the simple loop-detection mechanism of BGP. Routers in an AS are allowed to accept routes for destinations in other ASes only if they are ordered according to $\boldsymbol{L}$, and also order the routes they store locally according to $\boldsymbol{L}$.

Route filtering based on AS classes is supported in BGP-ELF by adopting a system-wide approach to the classification of ASes into classes [7]. Each AS class is denoted by an integer value from 1 to $|C|$, where $|C|$ is the total number of AS classes defined in the system.

A class vector with a bit for each AS class is used to denote the fact that an AS belongs to one or multiple AS classes. The class vector of a given AS consists of the ordered sequence of bits $\{c_1, c_2, ..., c_{|C|}\}$, where $1 \leq i \leq |C|$ and $c_i = 1$ if the AS belongs to the ith AS class defined in the system.

Node k has a list of unwanted types of ASes for each destination d, which is denoted by the unwanted class vector u_d^k with $|C|$ bits. The ith bit of this vector is denoted by $u_d^k(i)$ and $u_d^k(i) = 1$ if node k does not want routes to destination d that contain ASes belonging to AS class i.

Node k reports all its routes to all its neighbors, and those neighbors filter out unwanted routes themselves based on their own preferences. This way, nodes sending updates do not have to keep track of the unwanted class vectors of all their neighbors. The reported label from node k to destination d, $\ell_d^k[r]$, is augmented with an associated unwanted class vector u_d^k and a class vector ν_d^k. An update, query or reply from node k regarding destination d includes only $\ell_d^k[r]$ and ν_d^k, because nodes need not know the unwanted class vectors of their neighbors.

Node k stores the reported label and the class vector for destination d reported by each neighbor node. The class vector stored at node k and reported by node q for destination d is by ν_{dq}^k, with $\nu_{dq}^k \leftarrow \nu_d^q$.

We use $u_d^k \cap \nu_{dq}^k = \bar{0}$ to denote the fact that $u_d^k(i) \cap \nu_{dq}^k(i) = 0$ for $1 \leq i \leq |C|$.

When a router in AS k receives an update, query or reply from a neighbor router in AS q with a reported label $\ell_d^q[r]$ for destination d, the import transformation of BGP-ELF consists of accepting $\ell_d^q[r]$ only if the reported label is totally ordered with respect to the current value of its own reported label $\ell_d^k[r]$ and does not correspond to a route with a class vector containing any AS class that belongs to any of the unwanted AS classes by node k. This can be stated as follows:

$$\boldsymbol{BE}_i : \left(\ell_d^q[r] \prec_\ell \ell_d^k[r]\right) \wedge \left(u_d^k \cap \nu_{dq}^k = \bar{0}\right) \qquad (2)$$

If $\boldsymbol{BE}_i$ is true, the reported route from AS q is accepted and $\ell_{dq}^k \leftarrow \ell_d^q[r]$. In addition, node k updates ν_d^k with the bitwise OR of its own class vector and the class vector of the new route, i.e., $\nu_d^k \leftarrow \nu_{dq}^k \cup \nu_d^k$. This way, the updates, queries and replies sent by node k for destination d contain the most recent class vector associated with the reported label for the destination. On the other hand, if $\boldsymbol{BE}_i$ is false, the reported route is not accepted. In this case, $\ell_{dq}^k \leftarrow \ell_\infty$.

Once node k updates NT^k, it updates RT^k and takes different steps depending on its routing state. The routing state of routers in AS k is determined by the following condition:

$$\mathcal{T} : \left(\exists\, q \in N^k \left[\ell_{dq}^k \prec_\ell r_d^k\right]\right) \vee \left(\forall q \in N^k \left[\ell_{dq}^k = \ell_\infty\right]\right) \qquad (3)$$

A node is said to be **passive** if $\mathcal{T}$ is true and is **active** otherwise. If node k is passive, then $r_d^k \leftarrow \ell_d^k[r]$. If it is active, then r_d^k is not updated and equals the last value of $\ell_d^k[r]$ when node k was passive. Node k sends an update, a query, or a reply depending on the the input event and whether it is passive or active after its routing table is updated.

Node k takes the following steps to process an update $U(d, \ell_d^q[r])$ or after detecting a change in the state of its link with neighbor q:
(i) Sends $U(d, \ell_d^k[r])$ if it remains or becomes passive.
(ii) Originates $Q(d, \ell_d^k[r], \rho_d^k = r_d^k)$ if it becomes active.
(iii) Sends $Q(d, \ell_d^k[r], \rho_d^k = r_d^k)$ if it remains active after the input event, at least one neighbor v has reported a finite distance, and h_d^k was updated.

Node k takes the following steps to process a reply $R(d, \ell_d^q[r], \rho_d^q)$ from neighbor q:
(i) Sends $R(d, \ell_d^k[r], \rho_d^k = \rho_d^q)$ if it either becomes passive and $\rho_d^q \leq r_d^k$, or it remains passive and either $\rho_d^q < r_d^k$ or the value of h_d^k was updated.
(ii) Originates $Q(d, \ell_d^k[r], \rho_d^k = r_d^k)$ if it becomes active as a result of the reply from q; however, if $q \in S_d^k$ before the reply made node k become active, it updates $S_d^k \leftarrow \{q\}$, $h_d^k \leftarrow h_d^q + 1$.
(iii) Stays silent if it was active before the reply from q and remains active.

Node k takes the following steps to process a query $Q(d, \ell_d^q[r], \rho_d^q)$ received from neighbor q:
(i) Sends $R(d, \ell_d^k[r], \rho_d^k = \rho_d^q)$ if it is passive and has a neighbor v such that $\ell_{dv}^k \leq \rho_d^q$.
(ii) Forwards $Q(d, \ell_d^k[r], \rho_d^k = \rho_d^q)$ to its next hop s_d^k if it remains passive and has no neighbor v such that $\ell_{dv}^k \leq \rho_d^q$.
(iii) Forwards query $Q(d, \ell_d^k[r], \rho_d^k = \rho_d^q)$ to all its neighbors if it becomes active or remains active and $\rho_d^q < r_d^k$, and sets $r_d^k \leftarrow Min\{r_d^k, \rho_d^q\}$.
(iv) Stays silent if it is active before the query from q is received and all its neighbors have sent ℓ_∞ for destination d.

2) Multi-Path Local-Preference Function: BGP-ELF allows routers to choose among accepted routes according to local preferences defined by the local preference function, which consists of the same steps as those taken during Phase 2 of the BGP Decision Process (Section 9.1.2.2 of RFC 4271).

Let W be the set of link weights in which each link weight describes performance or policy-based characteristics of the link. The weight of the link from router i to router j is denoted by $w(i, j)$, and we make the restriction that $w(i, j) \in \mathbb{R}$ and $w(i, j) > 0$.

BGP uses path attributes in sequence to select preferred paths as part of the Decision Process (Section 9.1 of RFC

4271). Accordingly, we define the weight of a path for BGP-ELF in terms of a sequence of attributes as stated below.

Definition 3: **Path Weight:** The weight $\omega_d^k(n)$ of path $P_d^k(n)$ is defined to be a tuple with a finite number of attribute values associated with the path.

The ordered sequence of the n attributes of a path weight is $A = \{a_1, a_2, ..., a_{|A|}\}$. The order followed in this sequence is given by the order in which the attributes are used to determine that a path has a smaller weight than another path, i.e., that a path is preferred over another path. The value of the jth attribute of path $P_d^a(n)$ is denoted by $a_j[P_d^a(n)]$.

The order relation $<$ defined for real numbers is valid for the values of any path attribute, because we can assume that attribute values can be expressed as integers or real numbers.

Definition 4: **Path-Weight Preference:** A path $P_d^b(m)$ is preferred over path $P_d^a(n)$ if the following path-preference condition is satisfied:

$$\omega_d^b(m) < \omega_d^a(n) \equiv \exists j \leq |A| \ \Big[\ \Big(\ a_j[P_d^b(m)] < a_j[P_d^a(n)] \ \Big) \ \wedge \ \Big(\forall i < j \ \Big[\ a_i[P_d^b(m)] = a_i[P_d^a(n)] \ \Big] \ \Big) \ \Big]$$

Because at most one path to each destination is shared across ASes and nodes maintain the set of locally-available routes for each destination, nodes must determine the route with the longest labeled path length among all valid routes available locally according to Definition 5.

The set of labeled path lengths corresponding to loop-free routes for destination d that are locally available at a router in AS k is denoted by $\mathcal{L}_d^k$, and the set of ASes directly connected to AS k is denoted by A^k. It follows that $\mathcal{L}_d^k = \{\ell_{dq}^k \mid q \in A^k\}$.

Definition 5: **Longest Labeled Path Length:** The longest labeled path length in $\mathcal{L}_d^k$ is denoted by ℓ_{dmax}^k and is such that

$$\forall \ \ell_{dq}^k \in \mathcal{L}_d^k - \{\ell_{dmax}^k\} \ \ (\ \ell_{dq}^k \prec_\ell \ell_{dmax}^k \ ) \tag{4}$$

A router in AS k takes the following steps for each destination d: **(i)** Maintain the set of labels $\mathcal{L}_d^k$ and update S_d^k (as next hops to d) to include those neighbors with labels in $\mathcal{L}_d^k$; and **(ii)** update ℓ_{dmax}^k to be the longest label in $\mathcal{L}_d^k$ each time an update is made to $\mathcal{L}_d^k$.

3) BGP-ELF Export Transformation: The route reported by a router in AS k for destination d must be the path corresponding to the maximum label among all the routes in $\mathcal{L}_d^k$. The approach of communicating the longest route in a policy-based routing protocol that supports multi-path routing was originally proposed by van Beijnum et al. [17].

In BGP-ELF, the constraint imposed by the export transformation for a router in AS k to inform **all or only some of its neighbor routers** of a new route for destination d (depending on whether they are in provider, consumer or peer ASes) is:

$$\boldsymbol{BE}_e : \ell_d^k[r] = (k, \ 1 + |\ell_{dmax}^k|) \tag{5}$$

where $|\ell_{dmax}^k|$ is the number of hops in ℓ_{dmax}^k.

The steps and signaling described for the import transformation of BGP-ELF are used together with the local use of multiple routes to destinations without incurring routing loops. This is the case because the test that $\boldsymbol{L}$ is satisfied at all times by any route used in any AS to reach any destination is done on the basis of the reported labels.

C. Interior BGP-ELF

We describe Internal BGP-ELF by stating the additions needed to the ordered import transformation, ordered export transformation, and multi-path local preference function introduced for External BGP-ELF.

1) Ordering of Internal Routes: This method modifies RFC 4456, Section 8. Each BGP-ELF speaker orders the valid routes it receives giving priority to the routes that include the designated reflector of its own AS.

All BGP-ELF speakers know the identifier of the designated reflector of their AS. The routes that are reflected across clusters in an AS are all based on the choices made by the designated reflector, rather than the local choices of clients or reflectors in different clusters, which have different local preferences and hence lead to conflicts.

In I-BGP-ELF, a BGP-ELF speaker reports a single route to each destination and there is a single designated reflector in an AS. A route reported in I-BGP-ELF is a tuple consisting of an internal component within the AS and an external component.

We denote by λ_d^a the internal component of the route reported by a BGP-ELF speaker a. It consists of the tuple $\lambda_d^a = (n, h_d^a)$, where h_d^a is the number of clusters traversed by the route in the AS and n is the identifier of the route reflector reporting the route. The external component of the route reported by router a in AS k is the same as the labeled path length discussed for E-BGP-ELF and is denoted by ℓ_d^k. The route to destination d reported by BGP-ELF speaker a in AS k is the tuple $\rho_d^a = [\ \lambda_d^a, \ \ell_d^k \ ]$.

We use δ_r to denote the identifier of the designated reflector of AS r. The methods defined for I-BGP-ELF pertain to the internal components of routes.

Definition 6: **Internal Label Ordering:** A BGP-ELF speaker in cluster a of AS r is ordered along internal route $I_d^a = aI_d^b$ with respect to its next hop in cluster b of the same AS r if the following condition is satisfied:

$$\boldsymbol{I}: \ \lambda_d^b \prec_\ell \lambda_d^a \ \equiv \tag{6}$$
$$\big[\big(\ \delta_r \notin \lambda_d^b \ \big) \wedge \big(\ \delta_r \notin \lambda_d^a \ \big) \wedge \big(\ h_d^b < h_d^a \ \big)\big] \vee$$
$$\big[\big(\ \delta_r \in \lambda_d^b \ \big) \wedge \big(\ (\delta_r \notin \lambda_d^a) \vee [\ (\delta_r \in \lambda_d^a) \wedge (h_d^b < h_d^a) \ ] \ \big)\big]$$

A router in cluster a of AS r can accept an internal route $\rho_d^b = [\lambda_d^b, \ell_d^r]$ reported by a router in cluster b in AS r if condition $\boldsymbol{L}$ is satisfied by ℓ_d^r and condition $\boldsymbol{I}$ in Eq. (6) is satisfied by λ_d^b.

If a router in cluster a accepts route ρ_d^b from cluster b and $\delta_r \in \lambda_d^b$, then router sets $\lambda_d^a = (\delta_r, h_d^a)$ with $h_d^a = h_d^b + 1$.

The reason for the ordering condition in Eq. (6) is that all routes traversing clusters should be based on what the designated reflector perceives as the best choice, rather than what individual BGP-ELF speakers perceive in their own clusters.

2) Multi-path Local-Preference Function: Once ordering condition ***I*** is used to accept or reject routes in I-BGP-ELF, the method used to implement preferences is the same as in the multi-path local-preference function discussed for E-BGP-ELF. Exemplary lists of of steps representing a valid preference function are stated in [15] for BGP, and an example for BGP-ELF is presented in [7].

III. Examples of BGP-ELF Operation

We illustrate the advantages of how BGP-ELF operates compared to BGP using a well-known example of looping and route-oscillation problems in BGP for routing across ASes.

A. BAD-GADGET System [10]

Figure 1 depicts the operation of BGP in the BAD-GADGET system introduced in [10] as an example of a system in which BGP cannot converge to a stable routing state. The lexicographic values of AS identifiers are such that $A < B < C < D < E$, which we use instead of the integers used in [10]. Destination d is assumed to be located at AS A.

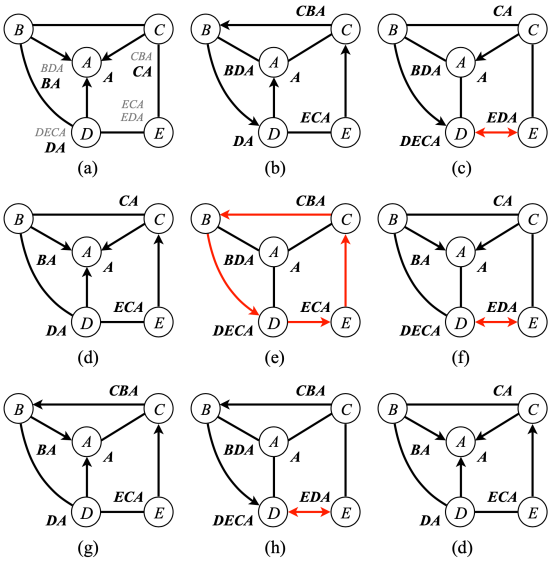

Figure 1: Operation of BGP in the BAD-GADGET system

Fig. 1(a) shows the path preferences for each AS listed in descending order next to each node representing an AS. The path announced by each AS is indicated in bold letters. For example, AS B prefers path BDA over the direct path BA. Arrowheads indicate next hops along paths to destination d. The sequence of steps starts with ASes B, C and D announcing their direct paths to destination d. Each subfigure shows one step of the routing state after ASes process the updates sent in the previous step. As Figs. 1(c), 1(e), 1(f), and 1(h) show, BGP cannot prevent temporary routing loops from occurring. Furthermore, BGP does not converge and ASes continue to cycle through the states shown in Figs. 1(d) to 1(h) without ever converging to stable routing state.

Figure 2 depicts the operation of BGP-ELF in the BAD-GADGET system of Figure 1 under the same assumptions. The reported labels that routers in one AS communicate to routers in neighboring ASes are indicated in the figure by tuples (F, h) next to the ASes, where F is the first AS along the route to destination d and h is the number of AS hops traversed in the route.The path corresponding to each reported label is also indicated. In this example, $\mathcal{T}$ is always satisfied and hence routers only exchange updates for destination d that state their reported labels. For simplicity, the updates sent between ASes are not shown.

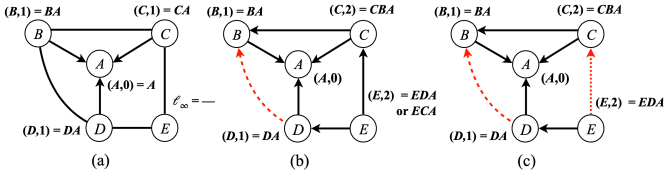

Figure 2: BGP-ELF operation in the BAD-GADGET system

BGP-ELF requires only two steps to converge deterministically to stable routing state after the initial updates from ASes B, C, and D. This is because $\mathcal{T}$ is used to constrain the local route preferences at each AS. AS Fig. 2(b) shows, AS B is not able to enact the local preference of using the route announced by AS D because $BA \equiv (B,1) \prec_\ell (D,1) \equiv DA$. By contrast, AS C can use the route announced by AS B because $BA \equiv (B,1) \prec_\ell (C,1) \equiv CA$; hence, AS C has two loop-free routes to destination d and announces the largest labeled path length. AS B is a possible next hop for AS D according to $\mathcal{T}$; however, this would be an additional local preference that was not considered in [10]. Accordingly, the route is shown with a dashed arrowhead and AS D is assumed to have only one route with label $(D,1)$. Similarly, Fig. 2(c) shows that AS E drops the route through AS C after it processes the update from AS C, and the corresponding route is indicated with dashed arrowheads to adhere to the preferences assumed in [10].

The route filtering assumed in Figs. 2(b) and 2(c) can be enacted in BGP-ELF by taking into account the class of AS that AS B is. The end result is that BGP-ELF *converges deterministically* to the final state shown in Figure 2(c) independently of how fast updates are propagated and without routing-table loops ever being created.

B. Link Failure in BAD-GADGET System

BGP-ELF does not suffer from any non-termination problems resulting from resource failures, because its use of total ordering among reported and stored routes leads to loop freedom at every instant and deterministic convergence. Figure 3 illustrates this point using a link failure in the same example of Figs. 1 and 2. Updates, queries, and replies are denoted without indicating destination d. Figure 3(a) shows the initial routing state of the system when the connection (B, A) fails. The example follows the same assumptions made in the example of Fig. 2.

Figure 3(b) shows that AS B must become active and send query $Q(d, \ell_\infty, (B,1))$ because $[(C,2) \not\prec_\ell (B,1)] \wedge [(D,1) \not\prec_\ell (B,1)]$. As Figure 3(c) shows, AS D and AS C can send

a response answering the query from AS B, because they both know that AS A has a reported label $(A,0) \prec_\ell (B,1)$. As Figure 3(d) shows, the reply from AS D allows AS B to become passive while adhering to the local preferences assumed in [10]. A second valid route indicated with a dashed arrowhead could be available to AS B through AS C depending on local route filtering (i.e., the type of ASes allowed in routes used by AS B). The figure also shows that the reply from AS C provides AS E with a second route through it.

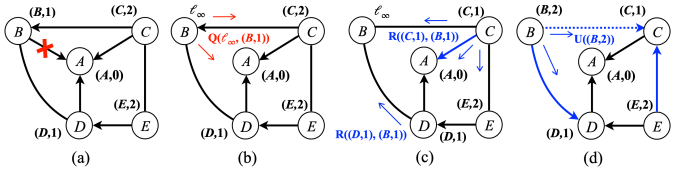

Figure 3: BGP-ELF convergence after failures

C. Eliminating Looping and Route Oscillations within ASes

Figure 4 shows a system in which IBGP with route reflection oscillates as described in Section 3 of [1]. Arrowheads indicate next hops along valid paths to destination d at BGP-ELF speakers in AS 0. Only the internal component of routes is shown, and it is assumed that stable routing is maintained within each cluster. Thick solid arrowheads indicate the next hops along the routes announced within. AS 0 to and from reflector A, which is the designated reflector for AS 0.

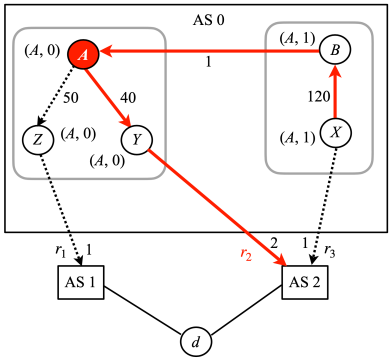

Figure 4: I-BGP-ELF convergences deterministically and is loop-free in ASes with route reflectors

As Figure 4 shows, I-BGP-ELF converges deterministically to one or multiple paths at each BGP-ELF speaker. The reason for this is that total ordering is maintained among routes reflected across clusters, and reflectors and clients of reflectors are required to adopt routes that include the routes chosen by the *designated reflector* A. This results in route reflectors establishing a directed tree towards the cluster of the designated reflector, which then points out to one or multiple paths to destinations in remote ASes. BGP-ELF speakers that are not border routers and are not in the same cluster of the designated reflector know only of paths to remote ASes that go to that cluster. Border routers may know local valid paths to destinations in remote ASes that do not involve the designated reflector but do not propagate them. An example of this case is router X in Fig. 4. The cluster in which the designated reflector resides may have multiple routers with routes to a destination (e.g., routers Y and Z in Fig. 4), but the designated reflector propagates a single route to other clusters

IV. Conclusions

BGP-ELF is the first protocol for inter-AS routing that is stable and loop-free at every instant without the use of path vectors or the need to engineer routing policies. It is based on four modifications of BGP. Eliminating loops without using path vectors in BGP-ELF required the introduction of queries and replies. However, the signaling in BGP-ELF is far more efficient than signaling based on diffusing computations [5] used in some intra-domain routing protocols like EIGRP, because an AS can become passive with the first reply it receives, rather than having to wait for all neighbor ASes to reply, and queries can be forwarded towards destinations.

AS classes were introduced as an alternative mechanism to enable route filtering based on the type of ASes along the paths used to reach destinations, rather than the specific ASes traversed in paths to destinations..

A detailed analysis of the performance of BGP-ELF within and across ASes is the subject of future work.

References

[1] A. Basu et al., "Route Oscillations in I-BGP with Route Reflection," *Proc. ACM SIGCOMM '02*, Aug. 2002.

[2] T. Bates, E. Chen, and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)," RFC 4456, , April 2006.

[3] J. Durand, I. Pepelnjak, and G. Doering, "BGP Operations and Security," RFC 7454, Feb. 2015.

[4] L. Gao and J. Rexford, "Stable Internet Routing without Global Coordination," *IEEE/ACM Trans. Networking*, 2001.

[5] J.J. Garcia-Luna-Aceves, "Loop-Free Routing Using Diffusing Computations," *IEEE/ACM Trans. on Networking,* Vol. 1, No. 1, 1993.

[6] J.J. Garcia-Luna-Aceves, "Stable, Loop-Free, Multi-Path Inter-Domain Routing Using BGP," *Proc. IEEE IEEE ICC '22 NGN*, May 2022.

[7] J.J. Garcia-Luna-Aceves, "Attaining Stable and Loop-Free Inter-Domain Routing without Path Vectors," *Proc. 1st ACM SIGCOMM Workshop on Future of Internet Routing & Addressing (FIRA 2022)*, Aug. 2022.

[8] J.J. Garcia-Luna-Aceves, "Eliminating Routing Loops and Oscillations in BGP Using Total Ordering," *Proc. IEEE LCN '22*, Sept. 2022.

[9] T.G. Grifin and G. Wilfong, "An Analysis of BGP Convergence Properties," *Proc. ACM SIGOMM '99*, Aug. 1999.

[10] T.G. Grifin, F.B. Shepherd, and G. Wilfong, "The Stable Paths Problem and Interdomain Routing," IEEE/ACM Trans. Networking, April 2002.

[11] J. Mauch, J. Snijders, and G. Hankins, "Default External BGP (EBGP) Route Propagation Behavior without Policies," RFC 4271, July 2017.

[12] D. McPherson, V. Gill, D. Walton, and A. Retana, "BGP Persistent Route Oscillation Condition," IETF Internet Draft, March 2001.

[13] R. Musunuri and J.A. Cobb, "A Complete Solution for iBGP Stability," *Proc. IEEE ICC '04*, June 2004.

[14] A. Rawat and M.A. Shayman, " Preventing Persistent Oscillations and Loops in IBGP Configuration with Route Reflection," *Computer Networks*, Dec. 2006.

[15] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)," RFC 4271, Jan. 2005.

[16] E. Rosen, "Exterior Gateway Protocol (EGP)," RFC 827, Oct. 1982.

[17] I. van Beijnum et al, "Loop-Freeness in Multipath BGP through Propagating the Longest Path," *Proc. IEEE ICC '09 Workshops*, 2009.

[18] D. Walton, D. Cook, A. Retana, and J. Scudder, "BGP Persistent Route Oscillation Solution," IETF Internet draft, May 2002.