Stable, Loop-Free, Multi-Path Inter-Domain Routing Using BGP

J.J. Garcia-Luna-Aceves
Computer Science and Engineering Department
University of California, Santa Cruz, CA 95064
jj@soe.ucsc.edu

***Abstract*—We introduce the concept of a routing etiquette as the set consisting of the information maintained by routers, local operations that can be carried out on such information, and the information that routers share with one another to reach stable routes using local private policies for path selection. OPERA is presented as one such etiquette. Sufficient conditions for stable and loop-free routing based on OPERA are proven. Based on these results, BGP is proven to be inherently unstable and prone to looping, and the policy mechanisms of BGP are modified slightly to attain OPERA-based BGP (OBGP). OBGP is the first loop-free inter-domain multi-path routing solution based on BGP. OBGP is proven to be stable and loop-free. Well-known examples of systems in which BGP does not converge are used to show the benefits of OBGP.**

I. Introduction

Routing in the Internet is hierarchical and supported by routing protocols that operate within an autonomous system (AS), and routing protocols that operate across ASes. Each AS is a collection of routing prefixes under the control of network operators working on behalf of a single administrative authority or domain, such that a the same well-defined set of policies for routing is used throughout the AS. Each AS includes one or multiple computer networks connected with each other. Today, BGP-4 [12] is the protocol used for routing among ASes in the Internet, and we refer to it simply as BGP.

Routing across ASes using BGP is intended to: (a) allow the use of routing policies that take into account local preferences within each AS that need not be known by other ASes, and (b) use signaling among ASes that results in all routers computing stable paths to destinations in different ASes. We call this type of signaling a **routing etiquette**, which we define in this paper as *"the code of polite behavior adopted and followed by all routers of a group."* A routing etiquette does not require routers to select routes based on any system-wide optimality criteria, state their local preferences publicly, or make all routers use the same routing preferences.

BGP is designed to provide a single path from an AS to a destination and a BGP update carries information about the AS path traversed by a routing update from an originating AS to a destination address range. Unfortunately, BGP is well-known to have non-termination and route oscillation problems. As Section II describes, considerable work has been carried out to address the limitations of BGP; however, there has not been a verifiable approach to make BGP stable and loop-free for routing across ASes.

This paper introduces an algebraic framework for routing etiquettes, and the first approach based on this framework that modifies only the policy mechanisms of BGP to attain stable, loop-free, multi-path routing across ASes in the Internet, without requiring changes in the signaling defined for BGP.

Section III presents the *Ordered Path Etiquette for Routing Algebra* (**OPERA**), which is the first algebra that formalizes routing etiquettes, and Section IV introduces and proves sufficient conditions for stable and loop-free operation of routing protocols based on OPERA. For simplicity, the rest of this paper assumes that routing within an AS (i.e., Internal BGP (IBGP) [12]) operates without causing loops by using a fully-meshed IBGP or other means (e.g., [11]), and focuses only on eliminating looping in External BGP (EBGP) [12].

Section V summarizes how BGP routers select routes and describes why BGP is inherently unstable and prone to temporary routing-table loops across ASes. Following this, Section VI presents OPERA-based modifications of the routing policies used in BGP that result in OPERA-based BGP (**OBGP**), which is proven to be stable and loop-free.

Section VII discusses well-known cases of route oscillation and non-deterministic convergence in BGP to illustrating that OBGP is stable and free of temporary routing-table loops. Section VIII summarizes our results.

II. Prior Work

The prior work addressing the convergence problems of BGP, and more specifically EBGP, can be classified into BGP analysis and extensions, alternatives to BGP, and routing algebras intended to formalize the description of BGP and other routing protocols.

Several studies have examined the dynamic behavior of inter-AS routing in BGP and path-vector routing protocols in general (e.g., [7], [9]). In particular, the seminal paper by Griffin and Wilfong [7] shows that a static-analysis approach to the BGP convergence problem is not practical, because the complexity of statically checking routing policies is either NP-complete or NP-hard. Hence, only dynamic approaches that focus on BGP signaling and how information is processed in BGP are practical.

A number of dynamic schemes have been proposed (e.g., [8], [9], [10]), and some provide guidelines for the setting of routing policies [6].

Although extensive work exists on protocols for loop-free multipath routing within ASes (e.g., [17]) there is very limited work on multipath loop-free routing across ASes. Recently, however, van Beijnum et al. [15] presented an approach to support multipath routing in BGP by requiring BGP routers to communicate the routes with the longest AS-paths among the routes locally available for each destination. While these techniques may help improve the convergence speed of BGP in some cases, none can guarantee convergence or avoid the occurrence of routing-table loops.

Few alternatives to BGP have been proposed for inter-AS routing. The Inter-Domain Policy Routing (IDPR) [4] architecture adopted a "link state" approach for the support of inter-AS routing. It did not receive much support because of its complexity and the need to modify the data plane. The Inter-Domain Routing Protocol (IDRP) [1] was a protocol for inter-AS routing proposed as an international standard that includes BGP as a proper subset. MIRO [16] is a multi-path approach to inter-AS routing in which routers learn default routes through the existing BGP protocol, and arbitrary pairs of ASes negotiate the use of additional paths that are bound to tunnels in the data plane.

Many routing algebras have been presented to formalize the way in which routes are computed in path-vector protocols. In particular, Sobrinho [14] has shown that a path-vector protocol can be made to converge to stable routes if and only if the routing algebra on which it is based is *monotone*, which means that the weight of a path, which is obtained based on some metrics defined in the algebra, cannot decrease when it is extended. However, Sobrinho offers only preliminary ideas on how to apply the algebra to BGP.

Chau et al. [3] present a unified treatment of previous work routing algebras based on total ordering [2], [14] to present *sufficient conditions* for the existence of stable paths when routing protocols that operate on the basis of local policies for path selection are used. In contrast to this work, OPERA can be used to address necessary and sufficient conditions for stable and loop-free operation of routing protocols.

III. OPERA

OPERA is motivated by the proposal in [15] on the advertising of the longest preferred paths. OPERA enables the design of stable and loop-free routing protocols based on a routing etiquette that accommodates local preferences, rather than on the use of a routing metric that renders system-wide optimal values. Following prior results on monotonicity [14] or strict boundedness [13] of routing algebras, OPERA is designed to be monotone in order to allow the design of protocols based on it to converge.

We use the following terminology to define OPERA: (a) $\mathbb{Z}^+$ and $\mathbb{R}$ is the set of positive integers and real numbers, respectively; (b) N is a set of nodes; (c) E is the set of edges with each edge connecting two nodes; (d) a node in N is denoted by a lower-case letter; (e) a link between nodes n and m in N is denoted by (n, m), and nodes n and m are said to be immediate neighbors of each other; (f) the set of nodes that are immediate neighbors of node k is denoted by N^k; and (g) the nth path from node k to destination node d is denoted by $P_d^k(n)$.

Path $P_d^k(j)$ can be viewed as the sequence of links along the path or the sequence of nodes along the path. Such a path can be denoted as the augmentation of a path $P_d^q(i)$ with link (k, q) to node q; therefore, $P_d^k(j) = (k, q)P_d^q(i) = kP_d^q(i)$.

The next hop along path $P_d^k(n)$ from node k to destination d is denoted by $s_d^k(n)$. Hence, path $P_d^k(n)$ consists of the concatenation of the link $(k, s_d^k(n))$ with a path $P_d^{s_d^k(n)}(m)$ offered by $s_d^k(n)$ to k. Therefore,

$$P_d^k(n) = (k, s_d^k(n))P_d^{s_d^k(n)}(m) = kP_d^{s_d^k(n)}(m)$$

Definition 1: The **Ordered Path Etiquette for Routing Algebra (OPERA)** is defined to be the elements of the tuple $\mathcal{R} = (\mathbb{I}, \mathcal{W}, \Pi, \Omega, \mathcal{M}, \mu_o, \mu_\infty, \varrho_\omega, \varrho_\pi, \leq, \prec_\pi, \varrho_\epsilon)$ where:

$\mathbb{I}$ is a set of node identifiers in which each such identifier is uniquely assigned to a node. $\mathbb{I}$ can be a subset of the set of alphanumeric strings, or a subset of $\mathbb{Z}^+$. In this paper, it is assumed that $\mathbb{I} \subset \mathbb{Z}^+$, and the identifier of a node is denoted by a lower-case letter.

$\mathcal{W}$ is a set of link weights in which each link weight describes performance- or policy-based characteristics of the link. The weight of the link from node i to node j is denoted by $w(i, j)$.

Π is the set of path labels in which each path label is assigned uniquely to a path from an origin node to a destination node. The path label of $P_d^k(n)$ is denoted by $\pi_d^k(n)$ and consists of the ordered sequence of node identifiers corresponding to the nodes along the path starting with k and ending with d.

Ω is the set of path weights, where each such path weight describes performance- or policy-based characteristics of a path based on link weights. The weight of path $P_d^k(n)$ is denoted by $\omega_d^k(n)$.

$\mathcal{M}$ is the set of routing-metric values, where the routing-metric value of path $P_d^k(n)$ is denoted by $\mu_d^k(n)$ and is defined by the tuple $\mu_d^k(n) = [\ \omega_d^k(n),\ \pi_d^k(n)\ ]$.

μ_o is the initial path metric assigned to a known destination for which a path can be found. By definition, $\mu_o = [\ \omega_o,\ \pi_o\ ]$, where ω_o and π_o are the initial path weight and path label associated with a known reachable destination, respectively.

μ_∞ is the routing-metric value assumed for an unreachable or unknown destination. By definition, $\mu_\infty = [\ \omega_\infty,\ \pi_\infty\ ]$, where ω_∞ and π_∞ are the path weight and path label associated with an unknown or unreachable destination, respectively.

ϱ_ω is a **path-weight function** (PWF) that takes as inputs the link weight $w(k, q)$ of the link from a node k to a node q and a path weight $\omega_d^q(j)$ associated with path $P_d^q(j)$ from q to d and returns a path weight $\omega_d^k(n)$ associated with the extended path $P_d^k(n) = kP_d^q(j)$.

ϱ_π is a **path-label function** (PLF) that takes as inputs a node identifier k and a path label $\pi_d^q(j)$ associated with path $P_d^q(j)$ from q to d and returns a path label $\pi_d^k(n)$ associated with the extended path $P_d^k(n) = kP_d^q(j)$.

$\leq$ is a **weight-induced order relation** defined for any three values $\omega_d^a(i)$, $\omega_d^b(j)$, and $\omega_d^c(k)$ such that the following properties are satisfied:
(1) *Reflexivity:* $\omega_d^a(i) \leq \omega_d^a(i)$
(2) *Transitivity:* For any three nodes $a \neq b \neq c$,
$[(\omega_d^a(i) \leq \omega_d^b(j)) \wedge (\omega_d^b(j) \leq \omega_d^c(k))] \rightarrow (\omega_d^a(i) \leq \omega_d^c(k))$
(3) *Antisymmetry:*
$(\omega_d^a(i) \leq \omega_d^b(j)) \wedge (\omega_d^b(j) \leq \omega_d^a(i)) \rightarrow (\omega_d^a(i) = \omega_d^b(j))$
(4) *Totality:* $(\omega_d^a(i) \leq \omega_d^b(j)) \vee (\omega_d^b(j) \leq \omega_d^a(i))$

$\prec_\pi$ is a **label-induced order relation** defined for any values $\pi_d^a(i)$, $\pi_d^b(j)$, and $\pi_d^c(k)$, with a, b, and c being three different nodes, such that the following properties are satisfied:
(1) *Irreflexivity:* $\pi_d^a(i) \not\prec_\pi \pi_d^a(i)$
(2) *Transitivity:*
$[(\pi_d^a(i) \prec_\pi \pi_d^b(j)) \wedge (\pi_d^b(j) \prec_\pi \pi_d^c(k))] \rightarrow (\pi_d^a(i) \prec_\pi \pi_d^c(k))$
(3) *Totality:* $(\pi_d^a(i) \prec_\pi \pi_d^b(j)) \vee (\pi_d^b(j) \prec_\pi \pi_d^a(i))$

ϱ_ϵ is an **etiquette function**. Its inputs are the identifier of a destination d, the set of locally-selected paths to that destination, and the identifier of a neighbor node n. Its output is either μ_∞ or the routing-metric value associated with one of the locally-selected paths for destination d.

OPERA does not mandate how nodes (routers or groups of routers) should select paths locally, PWF and PLF can be used in any order and in combination, and path selection functions need not be the same in every node. The monotonicity of OPERA follows directly from the properties of $\leq$ and $\prec_\pi$.

IV. Convergence without Optimality

A typical routing protocol operates on the notion that routers always attempt to select an optimum path to a destination based on choices available and a total ordering of routing-metric values. A different definition of convergence is needed for routing etiquettes in which routers select paths based on local preferences that may be private and the routes selected need not be optimum according to a system-wide metric.

This section formalizes the notion of convergence without optimality by ensuring the polite behavior of routers.

Polite behavior among routers (i.e., a routing etiquette) can be established in the context of OPERA in many different ways based on the ordering of paths established by means of the weight-induced order relation and label-induced order relation. For simplicity, in the rest of this paper we focus on using only a label-induced order relation to establish ordering of routes, and define this relation as follows.

$$(\pi_d^b(i) \prec_\pi \pi_d^a(j)) \equiv \left(a \notin \pi_d^b(j) \right) \wedge \tag{1}$$
$$\left([|\pi_d^b(i)| < |\pi_d^a(j)|] \vee [(|\pi_d^b(i)| = |\pi_d^a(j)|) \wedge (b < a)]\right)$$

where $|\pi_d^u(n))|$ denotes the number of hops along path $P_d^u(n)$.

We observe that the required properties of irreflexivity, transitivity, and totality of $\prec_\pi$ follow from the properties of the order relation $\leq$ defined over $\mathbb{Z}^+$, plus the facts that node identifiers are assigned uniquely to nodes and $(\pi_d^b(i) \prec_\pi \pi_d^a(j))$ implies that either $[|\pi_d^b(i)| < |\pi_d^a(j)|]$ or $[|\pi_d^b(i)| = |\pi_d^a(j)|] \wedge [b < a]$, with both the size of path labels and node identifiers being positive integers.

Definition 2: **Feasible Path:** A path to destination d is said to be feasible if it does not contain any node more than once, i.e., it dos not involve a loop.

Definition 3: **Stability** (Convergence to Feasible Routes): A routing protocol is said to converge to feasible routes for a given destination d after topology changes stop occurring at time T if: **(1)** For any destination d that a node k can reach, node k obtains at least one path $P_d^k(n)$ within a finite time after T, such that $\pi_d^k(n) < \pi_\infty$ and does not include any node identifier more than once; **(2)** for any unreachable destination d for node k, node k sets $\pi_d^k(1) = \pi_\infty$ within a finite time after T; and **(3)** node k does not change the value of any $\pi_d^k(n)$ within a finite time after T. □

Definition 4: **Loop Freedom:** A routing protocol is loop-free if all the paths to a given destination d implied by the routing information maintained by nodes define feasible paths at every instant. □

If node k uses node q as its next hop along a feasible path $P_d^k(n)$ to destination d and $P_d^k(n) = kP_d^q(m)$, then $P_d^q(m)$ is better than $P_d^k(n)$. The following definition formalizes this intuition.

Definition 5: **Label-Based Ordering along Loop-Free Paths:** Node k is ordered along path $P_d^k(n) = kP_d^q(m)$ with respect to its next hop q based on path labels if

$$\boldsymbol{L} : \pi_d^q(m) \prec_\pi \pi_d^k(n) \tag{2}$$

We note that $\boldsymbol{L}$ is trivially true for a path that does not exist, because a node k with no path to destination d is assumed to have π_∞ as its label for that destination, and π_∞ is larger than any actual path label.

Theorem 1: A routing protocol based on OPERA is guaranteed to be loop-free if the ordering condition $\boldsymbol{L}$ is satisfied at every instant by every node for any destination d.

Proof: Assume that $\boldsymbol{L}$ is true but the routing protocol is not loop-free and a loop L of h hops is created at some point in time with $L = \{n(1) \rightarrow n(2) \rightarrow ... \rightarrow n(h-1) \rightarrow n(1)\}$.

Without loss of generality, assume that each node has a single path to d. This implies that $\pi_d^{n(1)} \prec_\pi \pi_d^{n(h-1)}$ and $\pi_d^{n(i)} \prec_\pi \pi_d^{n(i-1)}$ for $1 < i \leq h-1$ because $\boldsymbol{L}$ is true. This is a contradiction, because it implies that $\pi_d^{n(i)} \prec_\pi \pi_d^{n(i)}$ for $1 \leq i \leq h-1$, which cannot be true because $|\pi_d^{n(i)}| \not< |\pi_d^{n(i)}|$ and $n(i) \not< n(i)$. Therefore, the theorem is true. ■

Theorem 2: If a routing protocol based on OPERA ensures convergence to feasible routes for each destination d, the ordering condition $\boldsymbol{L}$ must be satisfied by every node within a finite time after topology changes stop occurring.

Proof: The proof is by contradiction. Assume that a routing protocol based on OPERA has converged to feasible routes at time T but $\boldsymbol{L}$ is not satisfied. From the definition of convergence to feasible routes, no node can change the path label of any path after time T and no node can transmit a signaling message to update a path label. Hence, node k cannot change the label $\pi_d^k(n)$ of path $P_d^k(n)$ after time T. Let q be the next hop along path $P_d^k(n)$. node k must have used the path label reported by q to select q as its next hop along $P_d^k(n)$, and that path label corresponds to a path $P_d^q(m)$ from q to d. Furthermore, $\pi_d^q(m)$ cannot change after time T.

Because $\boldsymbol{L}$ is not satisfied at time T, node k can use q as its next hop along path $P_d^k(n) = kP_d^q(m)$ at time T while node q uses node n as its next hop along path $P_d^q(m) = qP_d^k(n)$ at time T. This is a contradiction, because then $P_d^k(n)$ and $P_d^q(m)$ cannot be feasible paths. ■

Theorem 3: If the ordering condition $\boldsymbol{L}$ is satisfied by every node for any destination d within a finite time after topology changes stop occurring, then a routing protocol based on OPERA ensures convergence to feasible routes.

Proof: The proof is by contradiction. Let T_s be the time when topology changes stop occurring. Because $\boldsymbol{L}$ must be satisfied within a finite time $T_o \geq T_s$, it must be true that Eq. (2) is satisfied at time T_o by each node k and its next hop along any path to any destination d that is reachable. From Theorem 1, it follows that the routes to d at each node are feasible. On the other hand, because each node computes routes based on OPERA, no node needs to update any route to destination d after time T_o with each route being feasible, which is a contradiction to the assumption that some node is unable to converge to a feasible route to d. ■

Theorem 4: A routing protocol based on OPERA in which the ordering $\boldsymbol{L}$ is satisfied at every instant by every node for any destination d is guaranteed to be stable.

Proof: The proof follows directly from the proofs of the previous theorems. ■

V. BGP AND ITS INHERENT LIMITATIONS

We assume that the reader is familiar with the neighbor acquisition, neighbor reachability, and network reachability procedures of BGP; as well as the way in which IBGP and EBGP routers operate. Conceptually, BGP can be viewed as implementing three policy mechanisms for routing across ASes: an import transformation used to accept valid routes, a preference function used to choose among valid routes, and an export transformation used to report selected routes.

A. Import Transformation in BGP

This transformation is applied to new routes learned at an AS from another AS. A local policy is applied to determine if a new route is accepted, and if so a transformation is applied as defined by the policy. It enforces the rule that a router never accepts a route if the AS of the router is already listed in the AS-path of a route reported by a neighbor router.

Using the terminology we introduced for OPERA, the AS-path included in the route record of an update is the path label defined in OPERA, and the constraint imposed by the BGP import transformation for a router in AS k to accept a route from a neighbor router in AS q for destination d can be stated as: $\boldsymbol{B}_i : k \notin \pi_d^q(1)$, where $\pi_d^q(1)$ is used to denote the fact that BGP is defined to support single-path policy-based routing.

B. Local-Preference Function in BGP

A preference function is used to select the best route for a given destination among the accepted routes reported by neighbor routers, and local preferences are not passed from one AS to another.

The local-preference function is a sequence of path-selection steps, where each step must be taken prior to the next. The specification of BGP [12] assumes single-path routing; however, many implementations (e.g., [5]) allow the use of multiple paths locally with some restrictions intended to prevent long-term routing loops. The list of steps presented in Algorithm 1 is based on the BGP local-preference function stated in [5], which augments the original function in [12]. We indicate whether a step of the preference function is part of a path-weight function (PWF) or a path-label function (PLF) as defined in OPERA.

Algorithm 1 BGP Local Preference Functions

```
Step 1 (PWF): Prefer the route with the highest weight (vendor-specific
parameter [5]), which is local and private to the router.
Step 2 (PWF): If multiple routes have the same weight, prefer the route
with the highest local preference. The local preference is the same for all
routers in the same AS.
Step 3 (PWF): If multiple routes have the same local preference, prefer the
route that was originated by the local router.
Step 4 (PLF): If none of the routes were originated by the local router,
prefer the route with the shortest AS-path, which contains the ASes an
update has traversed.
Step 5 (PLF): If the AS-path length is the same, prefer the lowest origin
type (IGP ¡ EGP ¡ incomplete).
Step 6 (PWF): If all origin codes are the same, prefer the path with the
lowest multi-exist discriminator (MED).
Step 7 (PWF): If the routes have the same MED, prefer external paths
(EBGP) over internal paths (IBGP).
Step 8 (PWF): Prefer the path with the lowest IGP metric to the BGP next
hop.
Step 9 (PWF): For EBGP paths, select the oldest route to minimize the
effect of routes going up and down (flapping).
Step 10 (PLF): Prefer the route with the lowest neighbor BGP router ID
value.
Step 11 (PLF): Prefer shortest cluster-list
Step 12 (PLF): If the BGP router IDs are the same, prefer the route with
the lowest neighbor IP address.
```

The PWF and PLF defined in OPERA can be used in any order and combination to select one or multiple paths to destinations, as long as a router can locally compare any two paths based on their weights and labels, and determine which path is better locally based on the order relation $\prec_\pi$. Accordingly, the local-preference function of BGP [12] and its commercial implementations fall within the scope of the PWF and PLF defined in OPERA, and can be viewed as the combined use of both functions.

C. Export Transformation in BGP

This transformation is applied to new routes communicated from an AS to another AS. A local policy is applied to determine if the route should be exported, and if so a transformation is applied as defined by the policy. A router sends updates to its neighbors to withdraw routes or add new routes to destinations. The attributes of a route advertised in an update must include: the origin type (IGP, EGP or incomplete); the AS path, which includes the set or sequence of ASes traversed by the update; and a next hop, which is a unicast IP address of the router that should be used as next hop to the destination. A destination of a BGP path is an IP address range.

This transformation enforces the rule that a route record must contain the AS-path stating the ASes that the update has traversed. The constraint imposed by the BGP export transformation for a router in AS k to **inform all or only some** of its neighbor routers (depending on whether they are in provider, consumer or peer ASes) of a new route for destination d can be stated as: $\boldsymbol{B}_e : \pi_d^k(1) = k\ \pi_d^{s_d^k(1)}(1)$.

According to $\boldsymbol{B}_e$, a router cannot communicate different routes for the same destination to different neighbor routers. A router can only stop the export of routes to a neighbor router depending on the type of AS of its neighbor router [6].

D. BGP Inherent Limitations

The loop-detection mechanism in the import transformation of BGP eliminates routing-table loops once all routing tables are consistent, but cannot prevent transient loops due to inconsistent routing tables. It follows from Theorem 1 that BGP is not loop-free because the combination of its import transformation and local-preference function does not guarantee that the order relation $\boldsymbol{L}$ is satisfied at every instant. Furthermore, it follows from Theorems 2 and 3 that BGP is inherently unstable (i.e., it cannot guarantee convergence within a finite time in all policy cases) because the combination of its import and export transformations and local-preference function does not guarantee that the order relation $\boldsymbol{L}$ is satisfied within a finite time after the topology of the system becomes stable.

Theorems 1 to 4 imply that the only practical way to make BGP stable and loop-free is by using $\boldsymbol{L}$ in its policy mechanisms for routing, and this is exactly what OPERA-based BGP does.

VI. OPERA-BASED BGP (OBGP)

OPERA-based BGP (**OBGP**) consists of introducing ordering along loop-free paths. This ordering can be enacted based on the weight- and label-induced order relations of OPERA. However, in this paper we focus on label-based ordering as stated in Eq. (2) as part of the import and export transformations of BGP. The local-preference function of BGP is augmented slightly to allow routers to use multiple loop-free routes to destinations without requiring the selected routes to be of equal weight or length.

A detailed description of how the weight-induced and label-induced order relations of OPERA can be used in OBGP is the subject of future work. In addition, the way in which routers can take advantage of the loop-free multi-path nature of OBGP is beyond the scope of this paper and is the subject of future work.

Due to space limitations we present only the changes needed in the import and export transformations and local-preference function of BGP for routing across ASes.

Each router advertises one 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 neighbor routers in other ASes.

The route advertised by a router in AS k to destination d is denoted by $P_d^k[r]$ and its label is denoted by $\pi_d^k[r]$.

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 P_{dq}^k the route to destination d stored at a router in AS k and reported by a router in AS q, and the corresponding path label is denoted by π_{dq}^k.

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

The maximum path label in Π_d^k is denoted by π_{dmax}^k and is such that

$$\forall\ \pi_{dq}^k \in \Pi_d^k - \{\pi_{dmax}^k\}\ \left(\ \pi_{dq}^k \prec_\pi \pi_{dmax}^k\ \right) \tag{3}$$

The path label of a non-existent path is π_∞ and its size is defined to be $|\pi_\infty| = \infty$.

Given that path labels state the AS routes advertised by routers, it is possible to determine whether a path label is a subset of another label. We denote the case in which a label value $\pi_d^q[r]$ is contained in a label π_{dy}^k stored locally at a router in AS k by $\pi_d^q[r] \in \pi_{dy}^k$.

A. Ordered Import Transformation

OBGP constrains the import transformation of BGP to accept routes only if they are ordered according to $\boldsymbol{L}$, and to order those routes stored locally according to $\boldsymbol{L}$.

When a router in AS k receives an update with a route with path label $\pi_d^q[r]$ from a neighbor router in AS q for destination d, the ordered import transformation of OBGP consists of accepting $\pi_d^q[r]$ only if the reported label is better than $\pi_d^k[r]$ and to eliminate local routes for which the previous value of $\pi_d^q[r]$ was a subset of the corresponding path labels.

The constraint used in the import transformation can be stated in terms of $\boldsymbol{L}$ as:

$$\boldsymbol{OB}_i : \pi_d^q[r] \prec_\pi \pi_d^k[r]. \tag{4}$$

If $\boldsymbol{OB}_i$ is satisfied, then the reported route from AS q is accepted and $\pi_{dq}^k \leftarrow \pi_d^q[r]$. On the other hand, if $\boldsymbol{OB}_i$ is not satisfied, the reported route is not accepted. In this case, $\pi_{dq}^k \leftarrow \pi_\infty$. In addition, once a route must be reset to π_∞ locally or as a result of an update stating that value, a router in AS k must reset the labels of those routes locally stored that contained the invalidated route.

Let $\pi_{dy}^k[old]$ and $\pi_{dy}^k[new]$ denote the previous and updated value of the label for the path P_{dy}^k from AS k to destination d through AS y. A router in AS k sets $\pi_{dv}^k[new] \leftarrow \pi_\infty$ if $(\pi_d^q[r] = \pi_\infty) \wedge \left(\pi_{dq}^k[old] \in \pi_{dv}^k[old]\right)$. This is done to cope with failures of sessions between ASes more efficiently.

B. Multi-Path Local-Preference Function

The same preference functions defined for BGP or implemented to date can be adopted in OBGP. The functionality of the function needs to add the maintenance of the set of locally-available routes for each destination, and determining the route that has the maximum path label as defined previously. Hence, in addition to the steps carried out by the BGP preference function, a router in AS k must take two steps for each destination d: (a) Maintain the set of labels Π_d^k; and (b) update π_{dmax}^k to be the maximum label in Π_d^k each time an update is made to Π_d^k. From Eq. (3) and the definition of $\boldsymbol{OB}_e$, it follows that $\forall\ \pi_d^k(i) \neq \pi_d^k[r]\ (\ \pi_d^k(i) \prec_\pi \pi_d^k[r]\ )$.

C. Ordered Export Transformation

The ordered export transformation enables the use of multiple routes to destinations, without requiring that the routes have the same weights or AS-path lengths. This is accomplished by requiring that 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 Π_d^k.

The constraint imposed by the ordered export transformation for a router in AS k to inform **all or only some of its neighbor routers** of a new route $P_d^k[r]$ for destination d (depending on whether they are in provider, consumer or peer ASes) is:

$$\boldsymbol{OB}_e : \pi_d^k[r] = k\ \pi_{dmax}^k. \tag{5}$$

A router in AS k sends an update message with a new route record for destination d if the value of $\pi_d^k[r]$ changes. Furthermore, if $(\pi_{dq}^k = \pi_\infty)\ \forall \pi_{dq}^k \in \Pi_d^k$ at a router in AS k, then $\pi_d^k[r] \leftarrow \pi_\infty$ and the router must send an update message with a route withdrawal for destination d, because the router does not have a route to d guaranteed to be loop-free.

D. Correctness of OBGP

Given Theorems 1 to 4, the following theorem implies that OBGP is loop-free and stable, i.e., that it must converge to feasible routes to destinations if they exist, without ever creating a loop.

Theorem 5: Ordering along loop-free paths ($\boldsymbol{L}$) is satisfied at every instant if OBGP is executed correctly.

Proof: The proof is by contradiction, i.e., by showing that having both OBGP executed correctly and $\boldsymbol{L}$ not being satisfied by a router in an AS k for a given destination d at some point in time T is a contradiction.

According to the correct operation of OBGP that is assumed, a router in AS k either has no route to a destination d and $\pi_d^k[r] = \pi_\infty$, or it has a route with $\pi_d^k[r] \prec_\pi \pi_\infty$. A router cannot negate the ordering constraint $\boldsymbol{L}$ in the first case, because it does not have any path to d. Therefore the rest of the proof can focus on the second case.

Assume that a router y in AS k computes a finite route $P_d^k(n)$ to destination d at time T executing OBGP correctly but $\boldsymbol{L}$ is false. Because OBGP is executed correctly, it follows from the execution of the local-preference function at router y that $\pi_d^k(n) = q\pi_{dq}^k$ with $q \in A^k$ and $\pi_{dq}^k \in \Pi_d^k$. Because router y stores route π_{dq}^k, it follows from the execution of the ordered import transformation (Eq. (4)) that $\pi_{dq}^k = \pi_d^q[r] \prec_\pi \pi_d^k[r]$ when router y accepts the route with label $\pi_d^q[r]$.

If router y updates π_{dmax}^qas a result of the new route it accepts with label $\pi_d^q[r]$, it follows from the correct execution of the ordered export transformation (Eq. (5)) that either $\pi_d^q[r] \prec_\pi \pi_{dmax}^k \prec_\pi \pi_d^k[r]$ or $\pi_d^q[r] = \pi_{dmax}^k \prec_\pi \pi_d^k[r]$.

The previous three facts constitute a contradiction to the assumption that ordering along loop-free paths given by Eq. (2)) is not true at some point in time when router y computes a new finite route $P_d^k(n)$. Therefore, the theorem is true. ■

VII. OBGP Examples

We use a few well-known cases of route oscillation and non-deterministic convergence in BGP to illustrate the stable and loop-free operation of OBGP.

A. BAD-GADGET System

BAD GADGET [7] is a well-known example of an unsolvable BGP system, with no execution of BGP being capable of arriving to a stable routing state. Figure1 illustrates BAD GADGET using circles to represent ASes and capital letters to denote the AS identifiers, such that $A < B < C < D$ to correspond to the original example in [7]. An intended destination d is located at AS A. In the BAD-GADET system, each AS has a local preference for the counter-clockwise route of length 2 over all other routes to AS A. Hence, absent any ordering constraints, AS D would prefer route DCA, AS C would prefer route CBA, and AS B would prefer route BDA.

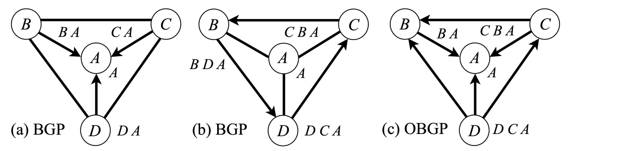

Fig. 1. OBGP converges in the BAD-GADGET system

The initial updates communicated among routers are shown in Fig. 1(a), which shows routers in ASes B, C and D announcing routes of one AS hop to AS A. The route announced by each AS is shown next to the circle representing the AS. Without any ordering and with routes in ASes reporting the one route they adopt, this leads to non-convergence and temporary routing-table loops. Figure 1(b) shows one of many possible intermediate states assuming that the routers in an AS process the initial updates from the adjacent AS counter-clockwise. Routers in AS D announce route DCA, routers in AS C announce route CBA, and routers in AS B announce route BDA, all of which induces a temporary routing-table loop. The loop is broken when routers process the new updates, at which point they reverse back to their original routes shown in Fig. 1(a). However, routers keep engaging in route oscillations involving the states shown in Figures 1(a) and 1(b) without termination after processing new updates from routers in other ASes.

By contrast, in OBGP, routers in AS B are unable to enact the local preference of using the route initially announced by AS D because $BA = \pi_d^B \prec_\pi \pi_d^D = DA$. However, routers in AS D can use routes announced by routers in AS C because $CA \prec_\pi DA$, and can also use routes announced by routers in B if local preferences allow because $BA \prec_\pi DA$. Similarly, routers in AS C can use the route announced by routers in AS B because $BA \prec_\pi CA$. As a result, the system *converges deterministically* to one or multiple routes to the final state shown in Fig. 1(c) independently of how fast updates are propagated and without routing-table loops ever being created.

B. DISAGREE System

DISAGREE [7] is a well-known example of a BGP system that can have more than one stable routing states starting with the same initial routing state. Figure 2 illustrates DISAGREE showing destination d located at AS A, with $A < B < C$.

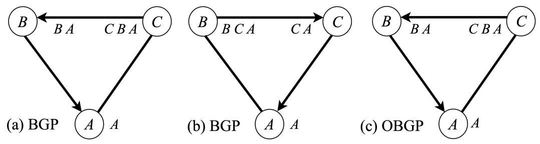

Fig. 2. OBGP convergences deterministically in the DISAGREE system

As it is described in [7], the final routing states shown in Fig. 2(a) and 2(b) are possible and their occurrence depends on how updates propagate in the system. The reason for two final routing states being possible in BGP is that there is no ordering between the route announced by AS C and the routes locally available at AS B. By contrast, the ordering established in OBGP among routes provides *deterministic convergence*, which results in the final routing state shown in Fig. 2(c).

The reader is referred to [7] for a description of the PRECARIOUS system, which is a combination of the BAD-GADGET and DISAGREE systems. Because OBGP enforces topology-independent deterministic convergence, it provides deterministic convergence in the PRECARIOUS system.

C. SURPRISE System

Link or router failures may result in non-convergent systems in BGP. The SURPRISE system is an example of this case presented in [7], and Figure 3 shows how OBGP converges deterministically in this system.

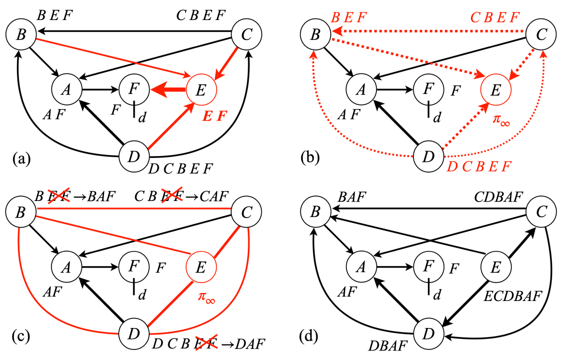

Fig. 3. OBGP convergences after failures in the SURPRISE system

Figure 3(a) shows the routing state at the routers in all the ASes when the session between ASes E and F fails. The routes at ASes E, D, B, and C are impacted by this event. Figure 3(b) illustrates the fact that routers in AS E do not have any loop-free route to d because none of the local choices satisfies $\boldsymbol{L}$. Accordingly, such routers must send updates with $\pi_d^E = \pi_\infty$. The figure shows in dashed lines the AS link corresponding to routes affected by the necessary deletion of the AS route EF. Figure 3(c) shows that routers in ASes C, B and D determine that their reported paths to d must be updated because they contain route EF as part of their own reported routes; however, the routers find alternate routes with labels that satisfy $\boldsymbol{L}$ and send the corresponding updates stating the routes with the maximum labels among those locally available. As Figure 3(d) shows, routers in all ASes, including AS E, quickly find one or multiple loop-free routes to d and report the maximum label among all those labels that satisfy $\boldsymbol{L}$.

VIII. Conclusions

OPERA was introduced for the design of routing protocols that are stable, loop-free, and select routes based on local policies rather than global optimality criteria. The necessary and sufficient conditions for any routing protocol based on OPERA to be stable and loop-free while allowing the use of private local policies for path selection were proven. These results were used to show why BGP, and more specifically EBGP, is inherently unstable and loop-prone.

OPERA-based BGP (OBGP) was introduced to provide stable, loop-free multi-path routing across ASes by means of small modifications of BGP's policy mechanisms in a way that ordering is always maintained among paths to destinations.

An attractive aspect of OBGP is that it can be deployed incrementally, because it does not change any of the signaling of BGP. Using OPERA in EBGP together with a loop-free IBGP makes BGP stable, loop-free, and capable of supporting multiple paths.

Our future work addresses the use of both label-induced and weight-induced order relations allowed in OPERA in the definition of policy mechanisms used in EBGP and IBGP.

References

[1] ANSI, "Intermediate System to Intermediate System Inter-Domain Routing Information Exchange Protocol," ANSI Doc. X3S3.3/90-132, 1990.

[2] B. Carre, *Graphs and Networks*, Clarendon Press, 1979.

[3] C.K. Chau et al., "Towards a Unified Theory of Policy-Based Routing," *Proc. IEEE Infocom 2006*, April 2006.

[4] D. Estrin et al., "A Protocol for Route Establishment and Packet Forwarding across Multidomain Internets," *IEEE/ACM Trans. on Networking*, Feb. 1993.

[5] Cisco Systems, "BGP Best Path Selection Algorithm," Document ID 13753, Sept. 2016.

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

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

[8] T. G. Griffin and G. Wilfong, "A Safe Path Vector Protocol," *Proc. IEEE INFOCOM 2000*, March 2000.

[9] C. Labovitz et al., ,"Delayed Internet Routing Convergence," *Proc. ACM SIGCOMM 2000*, Aug. 2000.

[10] Z. Mao et al., "Route Flap Damping Exacerbates Internet Routing Convergence," *Proc. ACM SIGCOMM 2002*, Aug. 2002.

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

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

[13] B.R. Smith and J. Samson, "Herding packets: Properties Needed of Metrics for Loop-Free & Best Forwarding Paths," *Proc. IEEE ICNC '17*, 2017.

[14] J. L. Sobrinho, "Network Routing with Path Vector Protocols: Theory and Applications," *Proc. ACM SIGCOMM '03*, Aug. 2003.

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

[16] W. Xu and J. Rexford, "MIRO: Multi-path Interdomain Routing," *Proc. ACM SIGCOMM '06*, 2006.

[17] W.T. Zaumen and J.J. Garcia-Luna-Aceves, "System for Maintaining Multiple Loop-Free Paths between Source Node and Destination Node in Computer Network," U.S. Patent 5,881,243, March 9, 1999.