

ASSEMBLER: A BGP-compatible Multipath Inter-Domain Routing Protocol

José M. Camacho, Alberto García-Martínez, Marcelo Bagnulo, and Francisco Valera

Abstract—The amount of redundant paths among ASes has dramatically increased throughout the Internet. Unfortunately, the unipath nature of BGP constrains border routers to course traffic across a single path at a time. Although, multipath inter-domain routing is able to provide richer routing configurations, the lack of incentives to replace BGP as inter-domain routing protocol implies that multipath solutions must be backwards compatible with BGP. Existing multipath solutions are limited by those requirements or their deployment requires coordination among ASes that are likely to have different stakes. This paper presents ASSEMBLER, the first BGP-compatible multipath protocol that can be progressively deployed by each AS individually and provide border routers with reachability information to forward traffic through multiple paths simultaneously.

Index Terms—IEEEtran, journal, LATEX, paper, template.

I. INTRODUCTION

The provision of multiple paths between two nodes has been envisioned for many years as a natural way to enhance communication networks. Once multiple paths are in place, nodes can divert traffic from failed links or split load among them, achieving fast recovery [1] and load balancing [2], [3] respectively. Those techniques should improve the reliability and the performance of the network.

Recent contributions [4], [5], [6] point out that the usage of multipath routing can be advantageous in inter-domain scenarios. Since most of the ASes through the Internet already have redundant connections with their neighbors [7], by embracing multipath inter-domain routing they could benefit from a more flexible use of their resources [8]. The reachability information advertised through these redundant connections should provide ASes with multiple alternatives to route the traffic towards a destination. Those alternative paths could be used simultaneously and enable the aforementioned recovery and balance techniques. Unfortunately, in most cases the unipath nature of the *Border Gateway Protocol* [9] impedes the use of those multiple paths concurrently.

Most ASes have no choice but to rely on techniques such as prefix deaggregation [9] or load sharing [10] to relax the constrains of BGP. Nevertheless, those techniques present their own limitations. For instance, per flow load balancing among paths [11], [12], [3] cannot be achieved using load sharing, since every time the traffic is switched to a different path a new BGP advertisement is generated and the network needs to converge. In practice, only stub ASes exploit their multi-homing connections to perform load balancing among different egress ASes, provided that they do not have to re-advertise BGP information.

The previous example shows that ASes are keen on more flexible routing configurations. However, in spite of the potential benefits that using multipath routing can bring about, so far, the lack of economic incentives to replace BGP has hindered Internet-wide deployments. Moreover, the latter imposes that any approach to deploy multipath inter-domain must be BGP-compatible.

Aimed at hastening large-scale deployments, some backwards compatible solutions have appeared in the literature in the latest years. BGP extensions such as [14], [15] provide multipath capabilities by taking advantage of the multiple interconnections between two ASes. Those paths have the same BGP attributes, such that every selected path can be advertised by the same BGP update. Whereas that ensures backwards compatibility with BGP, the set of multiple paths (i.e. hereafter multipath set) yielded by these solutions is rather limited, e.g. traffic cannot be forwarded across different egress ASes simultaneously even though available paths exist.

An alternative to exploit richer multipath sets is to forwarding packets among all available paths and advertise only one. That would require additional mechanisms to detect traffic loops [1], [5] or advertise paths that may be less attractive to legacy routers (e.g. longer paths [31]). Other solutions rely on a separate protocol to incrementally request or advertise additional paths [4], [16]. Those solutions can provide very flexible multipath configurations. Yet, they require that at least two neighbor ASes must coordinate to deploy that type of solutions, which represents a main drawback for those approaches.

In this work, a novel protocol for multipath inter-domain routing, ASSEMBLER, is presented. ASSEMBLER stands for *AS-Set-based Multipath BLeNDing Routing* since the protocol operation resembles a mixing of paths. It is the first inter-domain routing protocol that features both, flexible multipath routing and backwards compatibility with BGP, without any kind of coordination between ASes or additional protocols.

Furthermore, not only is ASSEMBLER backwards compatible with BGP, but also it adheres to its philosophy. Current routing policies, path import and export rules, and traffic engineering techniques are supported and in some cases extended. It is able to support and map to routing policies the existing business relationships among ASes. ASSEMBLER advertisements do not incur in any penalization when compared to BGP thanks to its *path assembling* technique. In addition, its selection process (so-called *K-Best Routing Optimizer*) can be locally tuned to cover a myriad of multipath configurations ranging from unequal AS path length multipath through different egress ASes to a *fallback* configuration that

mimics exactly the BGP behaviour.

This work is structured as follows, Section II introduces the requirements that are aimed for the protocol design. The protocol itself is presented in Section III along with an example to show the flexibility supported by ASSEMBLER in its configurations. A group of important deployment considerations are detailed in Section IV. The stability of the protocol is proven and configuration guidelines to guarantee stability are given in Section V. The work is completed with a comparison between ASSEMBLER and the existing multipath inter-domain proposals in the related work in Section VI. The conclusions are due to Section VII.

II. PROTOCOL REQUIREMENTS

The main goal of ASSEMBLER is to provide ASes with a backwards compatible solution for inter-domain routing that enables multipath routing. In this section, the defining requirements for ASSEMBLER are introduced prior to describing the relevant parts of the protocol in depth. Those requirements motivate the design choices that are presented in the next section and provide a clear view of the main features of the protocol. The discussion addresses the issues of target multipath configurations, backwards compatible updates, stability and data plane growth.

A. Flexible Multipath Routing

ASSEMBLER must flexibly let administrators choose the characteristics and the amount of paths yielded by the routing system. The multipath protocol must feature enough flexibility to concurrently select paths that, (1) have different next AS, (2) have different AS path length and (3) have different internal cost. Moreover, the protocol must be able to select a subset of the paths matching the previous conditions using a deterministic tie-break. ASSEMBLER must empower administrators with the tools to implement such a broad range of routing policies. For example, in Fig.1 for the prefix 150.0/16, R6 is able to get up to 3 paths, 2 from AS6 and another from AS7. In some cases, the administrator would like to provide R6 with the whole set of paths and in other cases, administrators may prefer keeping only those with certain attributes, (e.g. shortest AS path length).

The first requirement that we impose on ASSEMBLER is that regardless how the selection process of multiple paths is tuned, the BGP *winner* path is always included in the multipath set. In addition to the BGP winner, according to criterium (1), a router can either include paths through different egress ASes or limit the multipath set to go through the same next AS. Referring to R6, it can choose the paths from AS6 and AS7 or only from one of them. The criterium (2) defines if *equal AS length multipath* (i.e. ELMP) is enabled or additional paths, longer than the shortest one, can be selected. If ELMP is enabled, then R6 can select only the path from AS7, which is the shortest. Criterium (3) implies that internal *equal cost multipath routing* (i.e. ECMP) is supported and additionally, the administrator can tune how much the internal paths deviates from the hot-potato routing behaviour [18]. Assuming R4 has a lower cost to R7 than to R6 and it performs

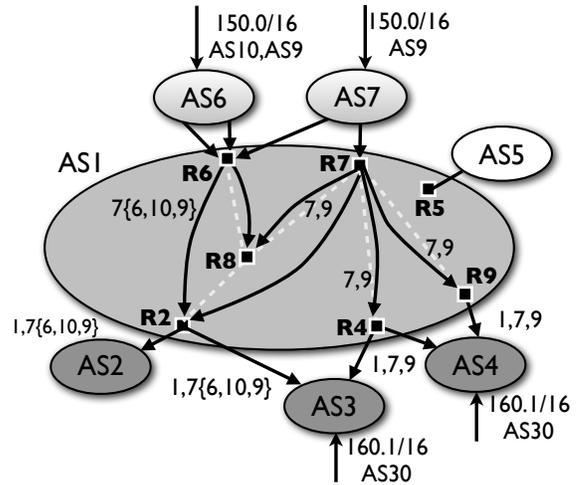


Fig. 1: Model of a transit AS with Path ASSEMBLER Routers

ECMP routing, thus it only selects the path through R7. A strong requirement is that ASSEMBLER must allow every AS to select the type of multipath that they need independently from other ASes, i.e. without any type of coordination.

B. BGP-Compatible Advertising Scheme

ASes advertise each other reachability information by means of BGP updates. Whereas processing regular BGP updates should not present any shortcoming for multipath routers, advertising multiple paths per network prefix in a single BGP update is not a trifle. Multipath routers should respect the structure and the semantic of the attributes included in the updates, such that legacy routers can keep on processing them. Concatenating multiple paths to the BGP update is not enough, provided that legacy routers sequentially process all of them and each path replace the state created by the previous one.

Therefore, ASSEMBLER must carry out some additional processing to merge information from multiple paths and accommodate them into regular BGP updates. To that extent, path *assembling* (Section III-B), a particular case of prefix aggregation [9] seems an outstanding candidate. It is a crucial requirement that generated advertisements must be representative of the aggregated paths, such that a router (legacy or not) can perform any regular BGP processing over the advertisement, as if the paths were announced separately. For instance, when a router receives an announcement containing an aggregate of paths, it must be able to derive the local preference for the aggregate or apply MED values comparison consistently. The protocol must identify those cases in which the advertisements are not representative and do not perform aggregation. Even for BGP, RFC4271 [9] identifies different situations where it is not consistent to aggregate multiple prefixes due to conflicting attributes. Thus, the protocol must avoid those situations in which inconsistent network advertisements may be created as a consequence of the aggregation process.

C. Controlled Routing Table Growth

The design must address the well known problem of the inter-domain routing table growth. Whereas routers feature more processing and memory capacity at the control plane, the situation at the data-forwarding plane is completely different. The hardware that forwards packets at wire-speed is expensive and its storage space constrained. The adoption of multipath does nothing but worsening the problem as multiple next-hops are stored per prefix. A growth in the amount of paths selected can potentially rise issues with the limitations of the data plane.

Therefore, the protocol must be aware of those constraints and limit the amount of paths relayed to the data plane. The requirement for the protocol is to be able to select a subset of *k-best* paths per prefix, such that the size of each routing table entry is limited. Every path in the subset must be compliant with the routing policy and the *k-best tie-break* must be deterministic. In our example, if R6 can only store two next-hops in its routing table for the same prefix, it has to discard one of its paths towards the prefix 150.0/16.

D. Stable under Common Configurations

BGP has been proven to be unstable under conflicting routing policies. The existing relation among those routing policies that cannot be fulfilled simultaneously is called *dispute-wheel* [19]. In presence of dispute-wheels BGP is not guaranteed to converge and the network may end up in a permanent oscillation. Permanent oscillations do not happen often in practice since routing policies are typically overruled by the business relationships among ASes. It can be proven that when routing policies align with those relationships dispute-wheels cannot be created.

The work in [20] presents a more abstract framework for the analysis of the stability in policy-based routing protocols. The framework extends the concept of dispute-wheels to *reflexive* policy relations. The concept of reflexive relations is more powerful in the sense that it covers the BGP dispute-wheels and allows to extent stability results to multipath policy-based routing protocols. ASSEMBLER must be able to converge in absence of reflexive relations among policies. Relying on the abstract framework in [20], the protocol must be proven stable in those situations in which conflicting policies do not exist, specially in those that align with business relations among ASes.

III. PATH ASSEMBLER

ASSEMBLER is a novel multipath inter-domain routing protocol inspired in the BGP prefix aggregation to compact the multipath information. ASSEMBLER stands for AS-Set-based Multipath BLEnding Routing, since the protocol *blends* the additional AS_PATHs and stores the result in AS_SETs. ASSEMBLER keeps backwards compatibility and allows for a progressive deployment of multipath-capable routers. The specification of ASSEMBLER relies on two main cornerstones: a multipath selection process and a BGP-compatible multipath advertising scheme.

Fig.2 shows the block diagram of an ASSEMBLER process running in the control plane of a router. There are some differences between Fig.2 and a BGP process diagram. The import policy (*ingress filtering* in Fig.2) is applied first, like in BGP. The BGP decision process has been replaced by the multipath selection algorithm K-BESTRO (pronounced *cabestro*) that stands for *K-Best Routes Optimizer*. K-BESTRO is presented in detail in Section III-A. The output of the K-BESTRO block is a set of *K* paths instead of a single winner path. K-BESTRO features three parameters to tune the characteristics of the multipath set. The parameter ELMP defines the maximum difference in AS path length between the shortest and the longest AS path. ECMP defines the difference in internal cost among selected paths. Finally, the KBEST parameter limits the maximum size of the multipath set, which should be set depending on the capacity of the routing table to store prefixes with multiple next-hops.

The paths in the multipath set are passed to the RIB in order to be installed in the data plane (through the FIB). Afterwards, they undergo the export policy. The export policy (*egress filtering*) generates the same advertisement for all the peering sessions that the router maintains. Therefore, a neighbor is either advertised or the export policy discards the whole multipath set as soon as one path matches a filter. Otherwise, different paths could be discarded for each peering session, generating different advertisements. Adding *neighbor-specific* announcements [21] is out of the scope of this paper.

Next to the egress filtering block, there is the new block called *Assembling*, which is responsible for generating the advertisements. The assembling algorithm ensures backwards compatibility, creating special BGP announcements that can be processed by legacy routers, do not incur in penalization when competing with regular unipath BGP announcements in the selection process and allow multipath capable ASes to use several paths concurrently. The algorithm takes its name from the way of constructing those announcements that resembles an assembling of pieces (e.g. AS_NUMBERS in this case). The announcement is an aggregated version of the multipath set that cannot be distinguished from the outcome of a regular prefix aggregation. See Section III-B for details about the assembling procedure for external (i.e. eASSM) and internal (i.e. iASSM) ASSEMBLER peering sessions (eASSM/iASSM are also used to refer BGP peering sessions unless stated otherwise). Finally, the advertisement containing the *assembled* path is propagated to the neighbor routers.

A. Decision Process: The K-BESTRO Algorithm

The decision process of ASSEMBLER is carried out over the set of advertisements for a given prefix that are not discarded by the import policy. The decision rules resemble those for BGP except for some subtleties. Meanwhile the BGP decision process clearly has a tie-breaking character and paths are *trimmed* from the set of candidates on the look out for the most suitable path. The requirements regarding flexibility of K-BESTRO completely redefines its philosophy and it is inspired in the decision process of Morpheus [22]. In the design of Morpheus, the decision process creates a *ranking*

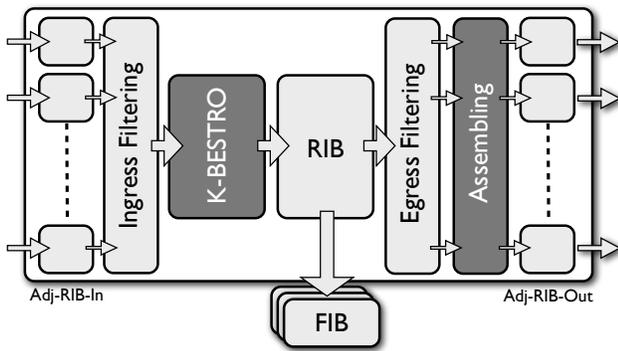


Fig. 2: Path ASSEMBLER Process Architecture

of the candidate paths according to some configurable criteria rather than discarding them. Afterwards, the set of best paths is selected, possibly according to different criteria, this time applied over the paths already sorted (e.g. select the first k paths in the ranking with MED value equal to 10). Therefore, K-BESTRO can be seen as a particular instance of a Morpheus ranking with the criteria presented in the next paragraphs. Each of them is mapped into a phase of the algorithm depicted in Table I,

a) The ranking criteria of K-BESTRO respect the semantic of the BGP attributes: As a consequence, K-BESTRO ranks the BGP winner in first position. Rules 1 to 5 discard paths like a regular BGP decision process. The BGP winner is never discarded by those rules and rules 6.a-d give higher ranks following the order used by BGP to tie-break the paths. Therefore the BGP winner is always ranked first. Generally speaking, the algorithm must always advertise the winner and propagate other aggregatable paths whenever possible.

In addition, in order to keep the semantic of BGP attributes and make the paths sortable, some of them must be discarded before the remaining paths are sorted in a rank. Otherwise, inconsistent multipath decisions can be made with respect to the semantic of the attributes. For instance, it is not consistent that two paths with different LOCAL_PREF appear in the final ranking or in the selected multipath set. It is not sounded either that two paths coming from the same AS and with different MED values are simultaneously used, since the customer is explicitly stating that it prefers one path to the other to receive the traffic.

The first phase starts with rules 1-2, which keep just the paths with highest local preference and with lowest ORIGIN value. The next step in BGP is to keeping paths with shortest AS path length. Instead, K-BESTRO considers paths that satisfy the relation $AS_PATH_LENGTH \leq shortest-l + ELMP$, being $shortest-l$ the shortest AS path length value found in the candidate set and ELMP is the parameter introduced earlier. The latter is implemented in rule 3.

b) The ranking criteria of K-BESTRO keep the order of the BGP rules: For example, the latter implies that the BGP AS path length rule must not be overridden by the MED rule, i.e. a path with lower MED but longer AS path must not cause any path with shorter AS path length to be overlooked by the algorithm. K-BESTRO ensures that every path taken into

account in the ranking honours the highest LOCAL_PREF, highest ORIGIN, lowest MED per AS and session TYPE (i.e. eASSM or iASSM) criteria exactly as BGP does.

The second phase applies these criteria (rules 4-5). For each AS advertising a path for the prefix, the algorithm looks for the paths of shortest length through that AS and the lowest MED value of that path. Every path from that AS and different MED value is removed at rule 4.c. The phase is completed by leaving paths only from one session TYPE. If there is a path from an external session, i.e. eASSM, paths with session TYPE iASSM are removed (rule 5). This is needed to avoid that two border routers try to course traffic through the external path of the other (see [9] for further details).

c) The final ranking of paths must be performed over monotonically increasing and bounded attribute values:

Applying rules 1-5 leads to consistent results regarding the selected multipath set. Once the considered paths are compliant with those rules, the ranking can be performed upon the remaining attributes without violating the specified routing policy and the order of BGP rules. For example, two paths with equal preference, origin and coming from different ASes, can be ranked according to their AS path length without creating any inconsistency.

The algorithm executes the third phase (rule 6) and ranks the paths according to the criterium of shortest AS_PATH_LENGTH first. If a several paths in a subset draw in AS path length, it sorts the subset from lower internal cost to higher. Within the subset, if the first ranked path has a cost of $lowest-c$ it removes the paths with internal cost higher than $lowest-c + ECMP$, where ECMP is a parameter that can be tuned by the administrator. While either tunnelling or IGP equal cost multipath are used inside the AS, ECMP should be equal to 0. If at this point some paths have the same AS path length and interior cost towards the next-hop, the paths with lower BGP_ID are ranked first. If several paths have the same BGP_ID attribute, then the ones advertised from the interface with the lowest address are ranked first.

d) The K-BESTRO algorithm selects paths in order of appearance in the ranking and selected paths are aggregatable: As described at the beginning of the section, the selected multipath set ends up aggregated into a single BGP advertisement. Therefore, the selected paths must be aggregatable, otherwise the generated announcement is not representative of the multipath set and that may lead to routing inconsistencies. RFC4271 [9] defines that two path with different MED values should not be aggregated. This restriction only applies to paths advertised through iBGP sessions. Similarly, only paths with the same NO_EXPORT:X Community (i.e. do not export this path to a specific peer X) can be aggregated, since as mentioned above, neighbor-specific configurations are not supported by ASSEMBLER. If the first ranked path does not include the NO_EXPORT:X community for any peer, the algorithm should overlook other paths in the ranking including any NO_EXPORT:X community when selecting the multipath set.

This last criteria is implemented in the fourth phase (rules 7-10) is executed. The parameter KBEST is defined by the administrator and limits the maximum size of the multipath

TABLE I: K-BESTRO Algorithm

1.-	Keep paths with highest LOCAL_PREF value
2.-	Keep paths with lowest ORIGIN value
3.-	Look for the shortest AS path, store the length in <i>shortest-l</i> and keep paths with $AS_PATH_LENGTH \leq shortest-l + ELMP$.
4.-	For each advertising AS,
4.a.-	Look for the subset of paths with lowest AS path length
4.b.-	Select the lowest MED value in that subset
4.c.-	Delete the paths from that AS with different MED value
5.-	If there is a remaining path with session TYPE eASSM, delete paths with TYPE iASSM
6.-	Rank the paths according to,
6.a.-	Paths with shortest AS_PATH_LENGTH go first
6.b.-	If a subset paths have the same length, paths with lowest internal cost goes first Discard paths within the subset with internal cost > lowest-cost + ECMP (Default ECMP=0)
6.c.-	If equal cost, lowest BGP_ID goes first
6.d.-	If same BGP_ID, lowest peer address goes first
7.-	If the first ranked path has the NO_EXPORT:X Community,
7.a.-	Then select only the first KBEST ranked path with the same NO_EXPORT:X Community
7.b.-	Else, delete paths with any NO_EXPORT:X Community
8.-	If the ranked paths have session TYPE iASSM, select the first <i>KBEST</i> paths
9.-	Else if the ranked paths have session TYPE eASSM, select the first <i>KBEST</i> paths from the same AS and MED value as the first ranked path
10.-	Return the selection. K-BESTRO ENDS

set. The fourth phase takes care of selecting a maximum of KBEST paths that can be aggregated and advertised together. According to the previous paragraph, if the first ranked path is tagged with the BGP Community NO_EXPORT:X, then the multipath set contains only the first KBEST ranked paths with the same community. Otherwise, paths with any NO_EXPORT:X community are deleted from the rank to avoid aggregation conflicts. Thereafter, if the ranked paths come from an iASSM session, then select the first KBEST paths in the ranking. Else if they come from eASSM sessions, select the first ranked KBEST paths coming from the same AS and with the same MED value as the first ranked path, as stated in RFC4271.

The algorithm finishes relaying the set of KBEST paths to the egress filtering block that implements the export policy.

B. Route Dissemination: Path Assembling

The decision process constructs a multipath set compliant with the preferences of the administrator. The multipath set must be advertised to every neighboring AS with an established peering session. Advertising an array of paths for

each network prefix is not supported by BGP. The algorithm presented in this section is applied to the set of paths to embed the multipath information into a single BGP advertisement.

The algorithm follows the philosophy used in prefix aggregation to compact the multipath information. Prefix aggregation defined in [9] defines how the attributes of two advertisements can be combined under some conditions, such that two contiguous prefixes propagated within each advertisement can be combined into a larger prefix and advertised in a single BGP update message. The attributes of the new message are the result of aggregating those in the two advertisements. Our *path assembling* can be understood as the aggregation of several advertisements carrying the same prefix.

Besides other attributes like the NEXT-HOP or the ORIGIN, of special interest is the AS_PATH attribute aggregation. When two contiguous prefixes are aggregated, it is necessary to keep the AS_PATH information to maintain path loop-freeness. In [9] the minimum requirements for the path aggregation algorithm are specified. Any algorithm compliant with those minimum specifications can safely combine the AS_PATH information from several announcements. The algorithm used by ASSEMBLER meets the minimum requirements of [9] and creates an AS_PATH following the most commonly found format in current routing tables. Data sets collected at some Internet vantage points [23], [7] brings out that recorded aggregations construct always an AS_PATH with an AS_SEQUENCE followed by an AS_SET. For example, the aggregate of paths $P = 1, 2, 3, 5$ and $Q = 2, 3, 4, 5$ should look like $A = 2, 3, \{1, 4, 5\}$.

In addition, the algorithm tries to be consistent in the assembling and keep meaningful information. For example, it creates AS_PATHs whose length is equal to the path length BGP would advertise. Thus, using assembling neither represents a penalty nor an advantage to multipath nodes, what we believe is fair. Moreover, the algorithm also preserves the last AS added to the AS_PATH as it is consider meaningful in some policies (e.g. neighboring AS filtering). The algorithm does not preserve the position of the origin AS, given that typically ASes rely on RIRs to check the origin and ASes included in an advertisement [24].

The algorithm is displayed in Table II. The AS_SEQUENCE is constructed with the AS_NUMBERS common to all the paths in the multipath set as suggested in RFC4271. The order between AS_NUMBERS is kept. If two AS_NUMBERS X and Y appear always one after the other in every path (even though some AS_NUMBERS may appear in the middle), they are said to be in order (rule 4.a). If two ASes, common to all paths, are not in order, then the second in appearance within the shortest path is put in the AS_SET (rule 4.b). The *assembled* AS_PATH is the result of concatenating the AS_SET at the end of the AS_SEQUENCE (rule 5). The remaining AS_NUMBERS not common to every path are added to the AS_SET (rule 6). Afterwards, rules 7.a-b check that the AS_PATH_LENGTH of the resulting AS_PATH is exactly the same as the shortest path in the multipath set. Rules 8.a-b deal with the fact that the local AS_NUMBER is not added to the advertisements until it is propagated to a peering router outside the AS.

TABLE II: Assembling Algorithm

1.-	Create an empty AS_SEQUENCE and AS_SET.
2.-	Pick up the shortest path from the multipath set and initialize <i>shortest</i> to its AS_PATH_LENGTH.
3.-	Copy the most to the left AS_NUMBER of the shortest path into the AS_SEQUENCE.
4.-	Keep parsing the AS_NUMBERS in the shortest path (if repeated, process it only once): Check its presence in other paths in the set.
4.a.-	If present in all paths and after AS_NUMBERS already in the AS_SEQUENCE, concatenate it to the AS_SEQUENCE.
4.b.-	Else add it to the AS_SET
5.-	Append the AS_SET at the end of the AS_SEQUENCE to create the AS_PATH.
6.-	For each remaining path, parse every AS_NUMBER. If the number has not been previously added to the AS_PATH, add it to the AS_SET.
7.-	Compare the length of the resulting path, if longer than <i>shortest</i> run 7.a, otherwise run 7.b,
7.a.-	Starting from the most to the right AS_NUMBER in the AS_SEQUENCE, move as many AS_NUMBERS into the AS_SET until the path length is equal to <i>shortest</i> .
7.b.-	Append at the beginning of the AS_SEQUENCE the most to the left AS_NUMBER as many times as needed until the path length is equal to <i>shortest</i> .
8.-	If the assembled path is advertised through an iASSM sessions run 8.a, run 8.b otherwise,
8.a.-	Return the resulting AS_PATH
8.b.-	Append the local AS_NUMBER at the beginning of the path and return the resulting AS_PATH

C. Example: An ASSEMBLER-Capable Autonomous System

This example refers always to the AS depicted in Fig.1. The figure represents a transit AS (AS1) with three customers (AS2,AS3,AS4), one peer (AS5) and two providers (AS6 and AS7) connected to AS1 by means of ASSEMBLER-capable routers. Router (R5,R7,R9) set (ELMP=0,ECMP=0,KBEST=1) and (R2,R4,R6,R8) aggregate the maximum number. Routers establish a full-mesh of intra-AS peering sessions to redistribute routing information. The example present two cases, a prefix propagated downstream from providers to customers and another prefix propagated upstream from the customers.

1) *Downstream Advertisement*: In the first case, two paths are advertised to AS6 and AS7 towards 150.0/16. The paths are propagated further and four paths reach AS1 and all of them are assigned with the same local preference. The egress routers for the paths in AS1 are R6 and R7. The router R6 can aggregate up to three paths depending on the ELMP parameter. If ELMP=0, then only the path from AS7 is selected according to rule 3 in Table I. Otherwise, if ELMP=1 or higher the three paths can be aggregated as depicted in the figure. The aggregated path from R6 is constructed using the algorithm in Table II. The first AS_NUMBER of the shortest path, AS7 leads the AS_SEQUENCE. Only AS9 is common to all paths and it is aggregated to the AS_SEQUENCE, as well. The remaining ASes are added to the AS_SET, AS6

and AS10. The length of the aggregated path is checked and it is 3 where it should be 2, therefore AS9 is moved into the AS_SET to compensate the length. Finally, the update is re-advertised through iASSM. Router R7 is unipath and selects only the path through AS7. Routers R2 and R8 receives both announcements from R6 and R7. The announcements have equal AS path length and equal IGP cost, therefore they are aggregated by the routers, although in this case the resulting aggregated paths are the same as for R6. Routers R4 and R9 are advertised as well. Both paths from R6 and R7 have the same AS_PATH_LENGTH therefore the ranking is done according to the IGP cost. Routers R4 and R9 has its ECMP parameter equal to 0 and consider only paths with lowest internal cost. Both, R4 and R9 select in this case the path through R7. Routers R2,R4 and R9 propagate the paths towards AS1 clients adding the AS_NUMBER 1 to the advertised AS_PATH.

2) *Upstream Advertisement*: In this second case, a couple of customers of AS1, AS3 and AS4 advertise a path towards their customer AS30. The effect of the MED values on the ranking function is shown in this second case. AS3 advertises two paths towards AS30, one to R2 with MED=20 and another one to R4 with MED=10. AS4 does not use MED values. Every path is assigned with the same local preference. Hence, three routers end up with a path from eBGP sessions and redistribute them through iASSM. The router R4 discards the internal path through R2-AS3 with highest MED and selects the eASSM path from AS3 and from AS4. Nevertheless, the two paths cannot be aggregated since the path from AS3 has MED value, therefore only the path through AS3, which has a lower BGP_ID, is ranked first and selected. R2 discards its own eASSM path and selects the internal path through R4 with the lowest MED for AS3. Assuming there is no restriction for R2 on the IGP cost to R9, it can aggregate the internal path through R9 as well, since the aggregated path is advertised only through eASSM sessions and the MED values are not taken into account.

IV. DEPLOYMENT CONSIDERATIONS

When it comes to the deployment of ASSEMBLER in a real AS there are some considerations to be taken into account beforehand. This section outlines them and points out how the main shortcomings that may appear during the deployment can be solved using the appropriated settings. As mentioned during the introduction and shown in the previous section, ASSEMBLER does not required of any kind of coordination between ASes to take advantage of multiple paths with high flexibility. Therefore, the deployment issues may rise while deploying it inside an AS. The issues are collected in three categories, problems related to mixed configurations, inconsistent multipath routing policies and traffic engineering techniques.

A. Deployments with Legacy Routers

Two different types of intra-AS techniques to forward the traffic are considered: internal redistribution and tunnelling. Internal redistribution implies that every router inside the AS

understands reachability information and fills in the routing tables accordingly. An internal protocol such as iBGP/iASSM is required to perform the redistribution. Tunnelling techniques rely on the encapsulation of packets. Only the ingress and the egress routers need to be aware of the paths advertised to the AS. In this discussion we consider IP over IP tunnelling and MPLS tunnelling [25] as representative techniques.

If the AS performs a full deployment, such that every legacy router is replaced inside the AS, both internal redistribution and tunnelling can be used without any kind of limitation. The only potential advantage of tunnelling is the addition of eBGP configurations [15] but that topic is out of the scope of this work. On the other hand, if the AS is not planning a full upgrade of the network, ASSEMBLER can still be deployed progressively. The difference in this case is that any router in the network can be randomly replaced if tunnelling is used, whereas special attention must be paid for internal redistribution. Using redistribution and legacy routers, ASSEMBLER routers must not aggregate paths received from internal peering sessions. The reason is that that internal aggregation may lead to routing inconsistencies. For instance in Fig.1, assuming that R2,R6 and R7 are unipath routers, if R2 receives the paths from R6 and R7 through the iASSM sessions and chooses the one from R6. R8 is multipath and aggregates the paths through R6 and R7, however it does not advertise the aggregate through iASSM to R2 to avoid internal loops RFC4271. If the IGP path between R2 and R6 passes through R8. The router R2 announces to the AS2 and AS3 its choice through AS6, however when packets get to R8, the multipath router can forward some of them towards R3 which is inconsistent with the network view that R2 is advertising.

B. Multipath Routing Policies

In addition to the considerations regarding the deployment of ASSEMBLER along with legacy routers, some other issues may arise related to the policy configurations of the ASSEMBLER routers. Defining simultaneously import and export policies for several paths that match a given criteria is supported nowadays in regular BGP routers. For instance, Cisco IOS uses the *route-maps* to define policies for several paths at once. ASSEMBLER-capable routers should support the same definition of policies. A BGP router can be transparently replaced and provide the same functionality configuring the same *route-maps* and setting K-BESTRO with the combination (ELMP=0,ECMP=0,KBEST=1).

However, the combination of policies for multiple paths with more lax K-BESTRO configurations may lead to inconsistent states regarding the export of paths. In contrast to BGP, the ASSEMBLER decision process yields a set of KBEST paths instead of a *winner* path. If several paths are assigned with the same LOCAL_PREF, and the import policy is not design appropriately, a path coming from a provider may end up in the selected multipath set with a path coming from a customer, which cannot be exported together. No ASSEMBLER advertisement is generated towards the providers whereas BGP would advertise the path from the customer. Therefore, LOCAL_PREF assigned by the import policy of

a router should be the same only for paths coming from the same type of neighbor AS, or at least the export policy should be modified accordingly. E.g., even though an AS ends up carrying traffic from one provider to another.

C. Enhanced Traffic Engineering

Once several paths are selected and advertised by an ASSEMBLER router they can be used simultaneously to forward packets. Outgoing traffic engineering (hereafter TE) is usually based on those local preference settings in the import policy. Nevertheless, in BGP only one path at a time is selected and the most flexible outgoing TE techniques are load-sharing [10] and tuning of IGP costs. Multipath BGP defined in [15], [14] allows for load-balancing among multiple parallel connections between two ASes but they only support load sharing to split the traffic among multiple ASes. ASSEMBLER allows to perform load-balancing across multiple ASes and in contrast to the proposals in [15], [14], the traffic can be switched from one egress AS to another without re-advertising additional routing information. In addition, how much traffic balance over each path becomes a new TE parameter.

Regarding TE using IGP costs, currently ASes send the traffic to the closest egress point in the network, which can be used a form of performing TE. ASSEMBLER extends this TE technique and administrators can modify the ECMP parameter of K-BESTRO to define how close to this hot-potato routing they want to sticks to.

On the other hand, the most widespread incoming TE techniques according to [26], [27] are prefix deaggregation, path prepending and TE with BGP Communities. Prefix deaggregation and BGP Communities for TE are supported. Yet, in order to respect the TE performed by neighbor ASes using path prepending, a maximum value must be setup for the ELMP parameter. That maximum value does not have to be public since in practice downstream ASes tune the amount of AS numbers to prepend on a trial and error basis [27].

V. STABILITY ANALYSIS

The stability analysis presented in this section is aimed at showing under which conditions ASSEMBLER converges. The analysis begins with a high-level discussion about the effect on the stability of the network of allowing nodes to choose multiple paths towards a given prefix. In order to give the discussion a formal formulation, the framework presented in [20] is used. The results focus on the consequences of using different *k-best flavours* at each node (e.g. unipath, multipath or mixed scenarios).

After the analytical model of the protocol is introduced, the stability analysis continues with the proof of the *synchronous* convergence of the protocol. If synchronous convergence is not achieved, then it is shown that the conflicting relation among policies exists. The stability proof is completed with results that prove the convergence of the protocol in asynchronous executions.

Provided that ASes choose their policies independently, the absence of conflicting policy relation in multipath cannot be guaranteed unless some policy guidelines, such as those

suggested for current unipath scenarios [17], are followed. The proposed multipath guidelines and the economic incentives of ISPs to follow them are presented in Section V-E after the stability analysis. In addition, it is shown that if nodes follow those multipath guidelines, it is not possible to have conflicting policies and the protocol should be able to find a stable state in finite time.

A. On Dispute Wheels in Unipath and Multipath Scenarios

The goal of this section is to introduce a discussion about the impact of multipath in the stability of policy-based routing protocols. Some notation is introduced before the discussion. The notation comes from the framework defined in [20].

In policy-based scenarios where a path vector protocol is running, paths propagate from one node to another as they are chosen in the ranking procedure as most preferred and announced to the routers with peering sessions. When a path P is preferred over a path Q according to the preferences of a node, that relationship of *preference* can be denoted like,

$$P \succsim Q \quad (1)$$

If a path P is announced and it is chosen as most preferred by an arbitrary number of nodes v_{i+n}, \dots, v_{i+1} in the network, the assigned path to v_{i+n} is a propagated version of P , i.e. $P' = v_{i+n}, \dots, v_{i+1}, v_i, \dots, v_0 = (v_{i+n}, \dots, v_{i+1})P$ and its relationship of *composition* with P can be expressed like,

$$P \succcurlyeq P' \quad (2)$$

Using these two simple concepts different relations among the policies of the different nodes and the paths announced by them can be denoted. A particular type of relations between paths caused by routing policies are the *reflexive* relations. Hereby, just a fair definition is introduced for the shake of clarity. For a formal and rigorous definition of *anti-reflexive* and *reflexive* relations see [20]. If there is an alternating sequence of preference (\succsim) and composition (\succcurlyeq) relations created among a set of paths $P_1, P_2, \dots, P_n, Q_1, Q_2, \dots, Q_n$, it is said that the relation is a reflexive relation if there is a path for which the sequence is cyclic, e.g.

$$P_1 \succcurlyeq Q_n \succsim P_n \succcurlyeq \dots \succcurlyeq \dots \succcurlyeq Q_2 \succsim P_2 \succcurlyeq Q_1 \succsim P_1 \quad (3)$$

Reflexive relations cannot be fulfilled simultaneously. For instance the path Q_n composed by an arbitrary path and P_1 is more preferred than P_1 , which is a contradiction since when Q_n selected P_1 is not selected and Q_n is feasible only if P_1 is selected. Hence, the protocol cannot find a solution and it is likely to oscillate.

The first reflexive relations depicted in the literature for policy-based routing are the BGP *dispute wheels* shown in [19]. Although the analysis in [19] shows the stability results using an abstraction of the ranking function, which covers all the different rankings that can be configured for ASSEMBLER, the analysis does not cover the case in which multiple paths are ranked as most preferred instead of just one. The framework in [20] considers multiple equally preferred paths and provided that our ultimate goal is to analyze the stability in mixed environments in which multipath nodes coexist with

unipath nodes, it is interesting to study the stability for the different flavours of the protocol under that framework.

The condition used in the framework to guarantee existence of multipath solution is that the preferences assigned by the ranking functions do not create a reflexive relation among the paths propagated throughout the network [20]. Since the generation of a dispute wheel involves a specific relation among the ranking functions, by changing the ranking functions and announcing additional paths, it is expected that the relation among them changes as well. The stable state of the initial scenario is no longer likely to be reachable, a different solution may exist or there may not be solution anymore. Before addressing the formal analysis of the problem, two examples are provided to show that the propagation of additional paths can provide a network running ASSEMBLER with an stable solution in unstable unipath scenarios, whereas in stable cases it can activate a dispute wheel.

Example 1: In Fig.3 a network is depicted in which every node is running the *1-best* flavour of ASSEMBLER (e.g. ELMP=0,ECMP=0,KBEST=1) which is equivalent to BGP. The path ranked in first position by each node is displayed with a solid arrow. Paths with lower rank are displayed in dashed arrows. The combination of each node preferences and the 1-best ranking creates the relation of paths shown in the table at Fig.3d. In Fig.3a the node v_1 receives three paths through v_2, v_5 and v_6 respectively. The three paths are of the same AS length and v_1 chooses the path $v_2v_7 \dots v_0$ with the lowest BGP_ID. The node v_2 has three paths of equal path length through v_7, v_8 and v_3 . It is not aware yet of the path $v_3v_9 \dots v_0$ and chooses to go through v_7 using the lowest BGP_ID criteria. Node v_3 is configured in a similar way, it chooses v_9 as next-hop since it is not aware of the path $v_4v_{11} \dots v_0$. The node v_4 receives a path through v_{11} but it is not aware of the path $v_1v_5 \dots v_0$, therefore it chooses v_{11} even though its highest preference is to use $v_1v_5 \dots v_0$. In addition, v_4 filters any AS path containing v_2 .

In Fig.3b, v_2 becomes aware of the path through v_3 and changes its assignment, forcing v_1 to change its path as well. Node v_1 does not select the new path of v_2 because is longer than the path through v_5 . However, v_3 becomes aware of the path through v_4 and changes as well. The path assignment is again modified in Fig.3c. Node v_2 loses its path $v_2v_3v_9 \dots v_0$ as it is longer than $v_2v_7 \dots v_0$ and v_4 prefers the path announced by v_1 . In the next step, the nodes go back to the initial assignment shown in Fig.3a completing a cycle in the oscillation. The reflexive relation can be expressed in this case as follows,

$$\begin{aligned} & v_1v_5 \dots v_0 \succcurlyeq v_4v_1v_5 \dots v_0 \succcurlyeq v_4v_{11} \dots v_0 \succcurlyeq \\ & \succcurlyeq v_3v_4v_{11} \dots v_0 \succcurlyeq v_3v_9 \dots v_0 \succcurlyeq v_2v_3v_9 \dots v_0 \succcurlyeq \\ & \succcurlyeq v_2 \dots v_0 \succcurlyeq v_1v_2v_7 \dots v_0 \succcurlyeq v_1v_5 \dots v_0 \end{aligned} \quad (3)$$

If the K-BESTRO selection algorithm is tuned to (ELMP=0,ECMP=0,KBEST=2), the scenario becomes stable. Every node chooses the path through one neighbor node on the dashed circle and a path through an outer neighbor except for v_4 , . For instance, v_1 selects the paths through v_5 and v_2 .

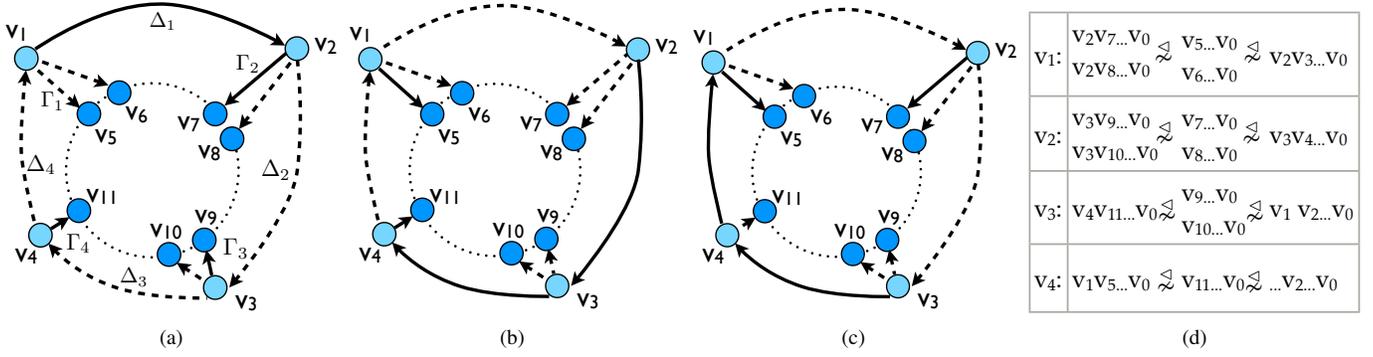


Fig. 3: Dispute wheel for the 1-best ASSEMBLER Flavour

Fig.4a shows the final path assignment. Node v_4 ranks only the path through v_{11} since it filter the advertisement from v_1 that contains v_2 .

Finally, if ASSEMBLER does not constrain the path length or the amount of paths, the stable path assignment displayed in Fig.4b can be achieved.

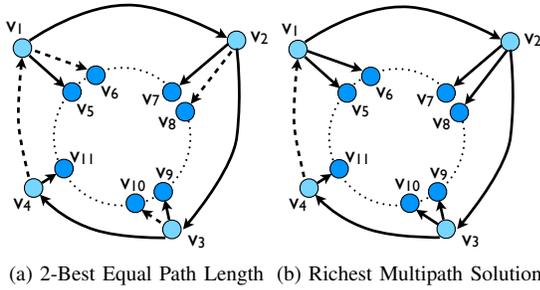


Fig. 4: Examples of stable multipath networks

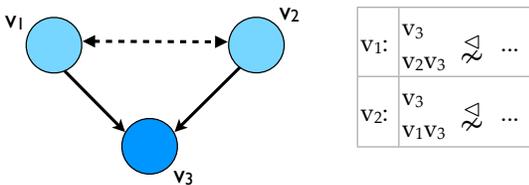


Fig. 5: Scenario with unipath but not multipath solution.

Example 2: In this second example we want to show that a particular configuration for which there is unipath solution but no multipath. Nodes v_1 and v_2 are running ASSEMBLER with (ELMP=1,ECMP=0,KBEST=2). Assume that for traffic engineering reasons, v_1 and v_2 give the same preference to their peering connection than to the connection to the customer that is multihomed to them. The policies in that case can be expressed as in the table on the right side of Fig.5. The reflexive relation is in this case,

$$v_1v_0 \succsim v_2\{v_1, v_0\} \succsim v_2v_0 \succsim v_1\{v_2, v_0\} \succsim v_1v_0 \quad (4)$$

The reflexive relation is created due to the fact that both nodes prefer the aggregated of the direct and indirect paths, to only

the direct path. Using ELMP=0 there is solution, in which ASSEMBLER assign only direct paths to each node.

Those two examples lead to the conclusion that the the propagation of additional paths can either stabilize a network running ASSEMBLER or activate a dispute wheel and turn unstable a previously stable configuration. Therefore, it is interesting to analyze the stability of ASSEMBLER under common configuration that align with the business incentives of the ASes, as those studied for BGP in [17]. After introducing the abstract modelling and stability analysis of ASSEMBLER, an extension of the guidelines proposed in [17] and the safety of ASSEMBLER under those practical are discussed in Section V.

B. Synchronous Model of Path ASSEMBLER

Let $G = \langle V, E \rangle$ be a topology graph, where V is the set of vertex and E is the set of edges of the graph and $v_0 \in V$ denotes the origin. Let $\mathcal{P}(v_i, v_0)$ be the set of *reachable* paths between v_i and v_0 in G , i.e. any path that can be physically constructed from v_i to v_0 . Now $\mathbb{P}(\mathcal{P}(v_i, v_0))$ is defined as the super-set of the subsets in $\mathcal{P}(v_i, v_0)$, so to speak, every element $\Phi_{v_i} \in \mathbb{P}(\mathcal{P}(v_i, v_0))$ is an arbitrary collection of elements in $\mathcal{P}(v_i, v_0)$. A multipath assignment over G is defined as,

$$\Phi = \{\Phi_{v_i} \in \mathbb{P}(\mathcal{P}(v_i, v_0)), \forall v_i \in V\} \quad (5)$$

Additionally, a partial multipath assignment is defined as,

$$\Phi = \{\Phi_{v_i} \in \mathbb{P}(\mathcal{P}(v_i, v_0) \cup \emptyset), \forall v_i \in V\} \quad (6)$$

I.e. for some nodes the assignment may be empty. The protocol is modelled as a fixed-point iteration of a distributed synchronous Bellman-Ford mapping $\mathcal{F}(\Phi)$ over the multipath assignment Φ . The mapping can be also defined as the set of local operations at each vertex during the k^{th} iteration like,

$$\mathcal{F}(\Phi_{[k]}) = \{\mathcal{F}_{v_i}(\Phi_{v_i[k]}), v_i \in V\} \quad (7)$$

The protocol starts growing from the initial iteration, in which the origin v_0 announces the path containing itself to its neighbors, and it increases the path assignment until reaching an assignment Φ that verifies the fixed-point equation,

$$\Phi = \mathcal{F}(\Phi) \quad (8)$$

In our synchronous Bellman-Ford mapping, nodes exchange announcements constructed as depicted in Section III-B. The most recent advertisement received by node v_i from node v_j at iteration k is denoted as

$$adv(v_j \rightarrow v_i)_{[k]} \quad (9)$$

Hereafter, we use indistinctly the terms path and aggregated path, even though a path is a particular case of an aggregate constructed upon a single route. The set of available paths for a node v_i at iteration k is the set,

$$\Omega_{v_i[k]} = \{adv(v_j \rightarrow v_i)_{[k]}, \forall v_j \in peers(v_i)\} \quad (10)$$

The Bellman-Ford mapping is defined locally at node v_i as the operation of selecting the KBEST first ranked paths in $\Omega_{v_i[k]} \in \mathbb{P}(\mathcal{P}(v_i, v_0))$ according to the K-BESTRO algorithm (Section III-A). The ranking procedure applied over a set of paths is denoted like $\lambda(\Omega_{v_i[k]})$ and the rank value of a path or a set of paths like $\lambda(\theta)$ and $\lambda(\Theta)$ respectively. Also, it is possible to define the value $\lambda_{max}(\Omega_{v_i[k]})$ over a set of announcements $\Omega_{v_i[k]}$ as the highest rank value assigned to any announcement within $\Omega_{v_i[k]}$.

The analysis assumes that each node defines a different ranking procedure and the following notation is used to differentiate among ranking functions, so that $\lambda_{v_i}(\Omega_{v_i[k]})$ denotes the ranking function at node v_i .

Given an advertisement and the ranking procedure at v_i , λ_{v_i} (which establishes ranking values λ , its maximum for a given set of announcements λ_{max}), the set of most preferred paths at iteration k can be defined as,

$$\beta_{v_i[k]} = \mathcal{F}_{v_i}(\Omega_{v_i[k]}) \quad (11)$$

where,

$$\beta_{v_i[k]} = \{\theta \in \Omega_{v_i[k]} / \lambda(\theta) = \lambda_{max}(\Omega_{v_i[k]})\} \quad (12)$$

Notice that although two announcements share most of the paths upon which they are constructed, a difference in one of the aggregated paths may cause the rank to vary.

As the synchronous execution of the mapping goes on, the paths in $\beta_{v_i[k]}$ change dynamically. Before expressing those dynamic changes and the evolution of the mapping in a formal way, first we have to define the concepts of *feasible* and *stabilized* set of paths.

The concept of feasible multipath assignment is rather intuitive if we define the set of all possible multipath assignments as the following Cartesian product (including partial assignments),

$$\chi = \prod_{v_i \in V} \mathbb{P}(\mathcal{P}(v_i, v_0) \cup \emptyset) \quad (13)$$

Therefore any multipath assignment $\Phi = (\Phi_{v_0}, \Phi_{v_1}, \dots, \Phi_{v_n}) \in \chi$ establishes at each vertex v_i a set of paths Φ_{v_i} among the reachable paths of that vertex. In addition, for each vertex in V , we define the following,

Definition 1.-: Given two vertex $v_i, v_j \in V$ such that $(v_i v_j) \in E$, the assignment Φ_{v_i} is said to be *consistent* with Φ_{v_j} if $\forall \rho \in \Phi_{v_i}$ of the shape $\rho = (v_i v_j)\theta$, it holds that $\theta \in \Phi_{v_j}$ and $v_i \notin \theta$ (i.e. to ensure loop-freeness).

It seems clear from the definition of χ that not all the components of $\Phi \in \chi$ are necessarily consistent with each other. Since our protocol handles only local information, the paths that it is able to construct must be consistent for all the vertex in the path. Hence, the definition of a feasible multipath assignment for our protocol can be expressed like,

Definition 2.-: A multipath assignment $\Phi \in \chi$ is said to be *feasible* if $\forall v_i, v_j \in V$ and $(v_i v_j) \in E$ then Φ_{v_i} is consistent with Φ_{v_j} .

Before defining the concept of *stabilized* multipath assignment, the following relations between multipath feasible assignments must be defined,

Definition 3.-: Let $\Phi, \Phi' \in \chi$ be two feasible partial multipath assignments then, Φ' contains Φ , i.e. $\Phi \subseteq \Phi'$, if $\Phi_{v_i} \subseteq \Phi'_{v_i} \quad \forall v_i \in V$ and $\Phi \subsetneq \Phi'$, if $\Phi \subseteq \Phi'$ and $\Phi_{v_i} \subsetneq \Phi'_{v_i}$ for some $v_i \in V$.

Definition 4.-: Given a partial feasible multipath assignment Φ , the set $\Psi(\Phi)$ defined as,

$$\Psi(\Phi) = \{\Phi' \in \chi / \Phi \subseteq \Phi'\} \quad (14)$$

is the set of feasible assignments which contain the path assignment Φ .

Definition 5.-: An assignment $\Theta_{[k]}$ is said to be *stabilized* if for all the sets of feasible sets containing $\Theta_{[k]}$, i.e. $\forall \Phi \in \Psi(\Theta_{[k]})$, it holds that

$$\Phi \supseteq \Theta_{[k]} \quad \text{implies} \quad \mathcal{F}(\Phi) \supseteq \Theta_{[k]} \quad (15)$$

The latter means that for any feasible assignment Φ containing $\Theta_{[k]}$, an iteration of the mapping over that larger assignment does not remove any path in $\Theta_{[k]}$ from the network. Therefore, any path $\theta \in \Theta_{[k]}$ is part of the fixed-point solution of Eq. 8 for the function \mathcal{F} and its ranking value verifies that

$$\lambda(\theta) = \lambda_{max}(\Psi(\Theta_{[m]})) \quad \forall m \geq k \quad (16)$$

Definition 6.-: Let $C_{[k]}$ be the set of *converged* nodes at the k^{th} iteration. Any node $v_i \in C_{[k]}$ verifies that $\beta_{v_i[k]}$ is a stabilized set at iteration k , what means that v_i converged at iteration k or before. Being $m \leq k$ the iteration at which v_i converged, then $\forall n > m$, $\beta_{v_i[n]} = \beta_{v_i[m]}$, therefore $adv(v_i \rightarrow w)_{[n]} = adv(v_i \rightarrow w)_{[m]}$.

Definition 7.-: Let $D_{[k]} \subseteq V - C_{[k]}$ be the set of nodes which are *direct peers* of converged nodes, then, $\forall v \in D_{[k]}$, $\exists u \in C_{[k]}$ and $e = (v u) \in E$.

C. Path ASSEMBLER Convergence

In this section, the anti-reflexive property of routing policies introduced in Section V-A, the notation presented in Eqs.1 and 2 and the protocol model depicted in the previous section are combined to assess the stability of ASSEMBLER. The analysis begins with the progress condition of the protocol. We show that if the progress condition does not hold for some iteration of the protocol, then there is a reflexive relation among the

policies. Afterwards, we prove that the progress condition implies that at each iteration the protocol gets closer to the fixed-point solution. Finally, the safety of the protocol is shown at the final theorem stated in this section. Notice that thanks to the definition of the ranking functions, the stability results apply to any flavour of the protocol running at each node.

Lemma 1.-: (Progress Condition) Let S be the set of routing policies of nodes in G . If any relation among policies in S is anti-reflexive and the current overall stabilized assignment $\Theta_{[k]}$ is not a fixed-point of the mapping, then there is an assignment $\Theta_{[k+1]}$ such that,

- 1) $\Theta_{[k+1]} \supseteq \Theta_{[k]}$
- 2) $\Theta_{[k+1]}$ is also stabilized
- 3) $\forall \Phi \in \Psi(\Theta_{[k]})$, then $\mathcal{F}(\Phi) \supseteq \Theta_{[k+1]}$

Proof: If $\Theta_{[k]}$ is stabilized it means that $\mathcal{F}(\Theta_{[k]}) \supseteq \Theta_{[k]}$ and $\mathcal{F}_{v_0}(\Theta_{[k]}) = \{v_0\}$, $k \geq 0$. In order to increase the multipath stabilized assignment there must be at least one node $v \in D_{[k]}$ peer of $u \in C_{[k]}$ such that,

- 1) By definition 6 u has a stabilized set $\beta_{u[k]}$, then $\forall \theta \in \beta_{u[k]}$ it holds $\theta \in \Theta_{[k]}$
- 2) Given $\rho = (v u) \in E$, $\alpha = adv(u \rightarrow v)_{[k]}$, it holds that

$$\lambda(\rho\alpha) = \lambda_{max}(\Psi(\Theta_{v[m]})) \quad \forall m \geq k \quad (17)$$

hence $\rho\alpha \in \Theta_{v_i[k+1]}$ (i.e. $\rho\alpha$ is stabilized since no path with higher rank will replace it in later iterations).

- 3) At iteration k , $\rho\alpha \in \Theta_{[k+1]} = \mathcal{F}(\Theta_{[k]})$ and $\rho\alpha \notin \Theta_{[k]}$

If such a node v exists then the proof of the lemma is completed since by construction $\mathcal{F}(\Theta_{[k]}) \supseteq \Theta_{[k]}$. By definition 5 and Eq.17, $\rho\alpha$ will not be removed in further iterations, therefore $\mathcal{F}(\Theta_{[k]})$ is stabilized.

Now we show that if that node $v \in D_{[k]}$ does not exist then the anti-reflexivity property does not hold over the policies in S . If v does not exist then no node $v_1 \in D_{[k]}$ is able to find a *direct* path Γ_1 constructed like above $\Gamma_1 = \rho\alpha$ for any peer node $u \in C_{[k]}$ and being $\lambda(\Gamma_1) = \lambda_{max}(\Psi(\Theta_{v[m]})) \quad \forall m \geq k$. Therefore, v_1 prefers more a path Δ_1 that is not through a converged peer. Using the *preference* and *composition* relations defined in Eqs.1 and 2 respectively, we can express the policy relation between Γ_1 and Δ_1 like,

$$\Delta_1 \succsim \Gamma_1 \quad (18)$$

Then if Δ_1 is not at one hop to a converged vertex, then Δ_1 must come from a propagated version of a direct path of some node $v_2 \in D_{[m]}$, therefore it can be constructed like $\Delta_1 = \Pi_2\Gamma_2$. Π_2 is an arbitrary path through nodes in $V - C_{[m]}$ and $\Gamma_2 = \rho'\alpha'$, with $\rho' = (v_2 u') \in E$ and $\alpha' = adv(u' \rightarrow v_2)_{[m]}$, is a direct path of v_2 . In terms of policy relations the latter can be expressed like,

$$\Gamma_2 \succ \Delta_1 \succ \Gamma_1 \quad (19)$$

Using the same reasoning, v_2 is not choosing any direct path Γ_2 , otherwise the path Δ_1 would become stabilized and the stabilized paths assignment would grow. Therefore the set $\beta_{v_2[m]}$ is formed by at least one path Δ_2 which is not direct and goes through a direct path announced by some node $v_3 \in$

$D_{[m]}$. The same procedure repeats for v_3 and we get to the relation,

$$\Gamma_3 \succ \Delta_2 \succ \Gamma_2 \succ \Delta_1 \succ \Gamma_1 \quad (20)$$

The relation keeps repeating for every element in $D_{[m]}$ until it hits v_1 again, producing a circular relationship of policies that cannot be fulfilled simultaneously,

$$\Gamma_1 \succ \Delta_n \succ \dots \succ \Gamma_3 \succ \Delta_2 \succ \Gamma_2 \succ \Delta_1 \succ \Gamma_1 \quad (21)$$

The latter relation implies that Γ_1 is less preferred than a path which is composed by an arbitrary path and Γ_1 , which is a contradiction. The latter completes the proof by showing that if the protocol gets stuck, then a dispute wheel exists among the policy relations.

Lemma 2.-: If a path $\theta = v_i v_{i-1} \dots v_0$ does not appear infinitely often in the multipath set β_{v_i} of v_i , then there is an iteration k after which any path of the form $\rho\theta$ disappears from the network.

Proof: Given a vertex v_i , $\theta = \rho'\theta'$ with $\rho' = v_i v_{i-1} \dots v_{j+1} v_j$ and $\theta' = v_j v_{j-1} \dots v_1 v_0$, if $\theta = \rho'\theta'$ does not appear in $\beta_{v_i[m]} \quad \forall m \geq k$ it means that there is at least one node $v_j \quad 0 \leq j \leq i$, for which there is a path $\theta'' \in \mathcal{P}(v_j, v_0)$ such that $\lambda(\theta'') > \lambda(\theta')$, therefore θ' is not part of $adv(v_j \rightarrow v_{j+1})_{[k]}$ after iteration k . At iteration $k+1$ the nodes $w \in peers(v_j)$ cannot use the path θ' any longer. The process repeats at each iteration along the next-hop in the path ρ' until $\rho'\theta'$ disappears. Thus, v_i cannot announce θ any longer and eventually $\rho\theta$ also disappears.

Lemma 3.-: The successive iterations of the Bellman-Ford mapping $\mathcal{F}(\Theta_{[0]})$, $\mathcal{F}(\Theta_{[1]})$, \dots , $\mathcal{F}(\Theta_{[k]})$, over the stabilized partial assignments reduce at each step the set of feasible path assignments Ψ , i.e. $\Psi(\Theta_{[0]}) \supseteq \Psi(\Theta_{[1]}) \supseteq \dots \supseteq \Psi(\Theta_{[k]})$.

Proof: Since we are using a synchronous model, we can assume that changes made by the mapping at v_i are propagated to the peers of v_i in the next iteration. At iteration zero, the set of feasible paths is equal to the super-set $\Psi(\Theta_{[0]})$ whose elements are any feasible set Φ defined by Eq. 14. Since the mapping evolves by repeatedly applying Lemma 1, then all those paths with lower rank than stabilized paths in the current iteration are not announced anymore. Then, by Lemma 2 lower ranked paths and those constructed upon them eventually disappear. In other words, following iterations of the mapping will not propagate them throughout the network and they are not feasible paths anymore. Those paths are removed from the set of feasible sets at that iteration, proving Lemma 3.

Theorem 1.-: (Safety) Given a network graph $G = \langle V, E \rangle$, given the set of policies S defined by each vertex in V , a synchronous distributed Bellman-Ford mapping $\mathcal{F}(\Theta_{[k]})$ iterating over the path assignment $\Theta_{[k]}$ which is initially defined as,

$$\Theta_{v_i[0]} = \begin{cases} \{v_0\}, & i = 0 \\ \emptyset, & i \neq 0 \end{cases} \quad (22)$$

If every policy relation over S is *anti-reflexive* then the mapping is able to grow the path assignment at each iteration

until the fixed-point of the following equation is reached at some iteration m ,

$$\Theta_{[m]} = \mathcal{F}(\Theta_{[m]}) \quad (23)$$

Thus, it can be stated that in absence of reflexive policies the protocol is able to synchronously converge.

Proof: By applying Lemma 1 at each iteration, in absence of conflicting policy relations, the mapping is always able to increase the path assignment with at least one path such that the new assignment $\mathcal{F}(\Theta_{[k]}) \supseteq \Theta_{[k]}$ is also stabilized. By Lemma 3, as the mapping is consolidating stabilized paths at each vertex, the set of feasible paths Ψ is decreasing, until the highest ranked paths feasible at each node are announced. Hence, there is one iteration k at which the only feasible set of paths at a certain node v_i is the set $\Phi_{v_i[k]} \in \mathbb{P}(\mathcal{P}(v_i, v_0))$ formed by elements that verify the equation $\lambda(\theta) = \lambda_{max}(\Psi(\Phi_{v_i[m]}))$, $\forall \theta \in \Phi_{v_i[k]}$ and $\forall m \geq k$. Since the mapping does not remove paths from a stabilized assignment and it cannot find higher ranked paths at any node, the next iteration $k + 1$ will have as outcome the same path assignment. Therefore, we can say that the fixed-point has been hit at iteration k .

D. Asynchronous Convergence

Despite using a reliable transport protocol, which guarantees ordered message delivery among nodes, the execution of our protocol is not free from communication delays since data synchronization is not enforced between peers. Therefore it may happen that the set of paths used by a node v_i to compute its multipath set at time t (i.e. $\Omega_{v_i[t]}$),

$$\beta_{v_i[t+1]} = \mathcal{F}_{v_i}(\Omega_{v_i})_{[t]} \quad (24)$$

it is a distorted version due to delay in the propagation of some of the announcements coming from peers. A distortion due to a delay of $t - \tau_{v_i, v_j}(t)$ between peers v_i and v_j , with $0 \leq \tau_{v_i, v_j}(t) \leq t$, makes v_i perceive,

$$\beta_{v_i[t+1]} = \mathcal{F}_{v_i}(\Omega_{v_i})_{[\tau_{v_i, v_j}(t)]} \quad (25)$$

Since the iteration over the fixed point is now distorted by $t - \tau_{v_i, v_j}(t)$ we cannot ensure convergence of the fixed-point iteration anymore. According to the general results in [20], it is possible to ensure convergence for a totally asynchronous distributed fixed-point iteration if,

- 1) The propagation of information happens infinitely often. In other words, it can be assumed that after a certain time $t' > t$ all the announcements $adv(v_j \rightarrow v_i)_{[t]}$ have been propagated and renewed at peer nodes.
- 2) *Synchronous Condition:* The protocol creates at each iteration $k = 0, 1, \dots, m$, a sequence of sets,

$$X_{[0]} \supseteq X_{[1]} \supseteq \dots \supseteq X_{[n-1]} \supseteq X_{[n]} \supseteq \dots \quad (26)$$

and it holds

$$\forall x \in X_{[k]} \text{ that } \mathcal{F}(x) \in X_{[k+1]} \quad (27)$$

- 3) *Box Condition:* For each iteration $k = 0, 1, \dots, m$ and each node v_i , $i = 0, 1, \dots, n$, there exist sets of

elements $X_{v_i[k]}$ such that the set of elements $X_{[k]}$ can be expressed as the Cartesian product,

$$X_{[k]} = \prod_i X_{v_i[k]} \quad (28)$$

The box condition implies that different elements in $X_{v_i[k]}$ can be exchanged without affecting the final result of the iteration, so to speak, the order in which the Bellman-Ford makes decision does not affect to the evolution of the mapping.

If those three condition hold, the following theorem can be proven,

Theorem 2.-: (Asynchronous Convergence) Given a network graph $G = \langle V, E \rangle$ with n nodes, the set of policies S defined by each node and a distributed Bellman-Ford mapping $\mathcal{F}(\Theta_{[k]})$ iterating over the path assignment $\Theta_{[k]}$, if every policy relation over S is *anti-reflexive* then the mapping is able to asynchronously converge to a multipath assignment of paths over G .

Proof: The condition (1) is guaranteed since ASSEMBLER uses a reliable transport protocol to exchange the information and every node advertises its neighbors with every change in the selected multipath set. Condition (2) is guaranteed by Theorem 1. In addition, replacing $X_{[k]}$ by $\Psi(\Theta_{[k]})$, the set of feasible sets containing $\Theta_{[k]}$, both Eq.26 and 27 can be rewritten as follows. Lemma 3 proves that the protocol creates the sequence $\Psi(\Theta_{[0]}) \supseteq \Psi(\Theta_{[1]}) \supseteq \dots \supseteq \Psi(\Theta_{[k]}) \dots$, which can be easily identified with the sequence in Eq.26. Moreover, it can be stated by definition 4,

$$\forall \Phi \in \Psi(\Theta_{[k]}) \text{ then } \Theta_{[k]} \subseteq \Phi \quad (29)$$

and by definition 5, applying an iteration of the algorithm on both sides, if $\Theta_{[k]}$ is stabilized then,

$$\mathcal{F}(\Theta_{[k]}) \subseteq \mathcal{F}(\Phi) \Rightarrow \Theta_{[k+1]} \subseteq \mathcal{F}(\Phi) \quad (30)$$

again, by definition 4, $\mathcal{F}(\Phi) \in \Psi(\Theta_{[k+1]})$, so that Eq.27 can be rewritten as well.

Finally, in order to complete the proof of Theorem 2, the box condition must be verified. Again, $X_{[k]}$ is identified with $\Psi(\Theta_{[k]})$. Every element $\Phi_{[k]}^{(p)} \in \Psi(\Theta_{[k]})$ can expressed in terms of its individual multipath assignment at each node like $\Phi_{[k]}^{(p)} = (\Phi_{v_0[k]}^{(p)}, \Phi_{v_1[k]}^{(p)}, \dots, \Phi_{v_n[k]}^{(p)})$. The stabilized assignment can be also expressed like $\Theta_{[k]} = (\Theta_{v_0[k]}, \Theta_{v_1[k]}, \dots, \Theta_{v_n[k]})$. By Definitions 3 and 4, each different $\Phi_{[k]}^{(p)}$ verifies that at least one node increases the stabilized path assignment, i.e. $\Theta_{v_i[k]} \subsetneq \Phi_{v_i[k]}^{(p)}$ for some $0 \leq i \leq n$. The superset $\Psi_{v_i[k]}$ is defined for all the possible assignments that v_i can receive containing $\Theta_{v_i[k]}$, i.e. $\Psi_{v_i[k]} = \{\Phi_{v_i[k]}^{(p)}, \forall p\}$. Therefore, the set of feasible sets containing $\Theta_{[k]}$, can be rewritten as the following cartesian product,

$$\Psi(\Theta_{[k]}) = \Psi_{v_0[k]} \times \Psi_{v_1[k]} \times \dots \times \Psi_{v_n[k]} \quad (31)$$

which can be arranged to resemble Eq.28 as follows,

$$\Psi(\Theta_{[k]}) = \prod_{0 \leq i \leq n} \Psi_{v_i[k]} \quad (32)$$

and the box condition is proven. Provided that the three conditions are verified for the protocol, by the *General Asynchronous Convergence Theorem* (Proposition 2.1 in [28]) it can be stated that the protocol is able to converge asynchronously.

E. Stable Multipath Policy Guidelines

The analytical results obtained in the previous section show that if ASes select their policies independently, stability cannot be always guaranteed. Therefore, it is interesting to derive a set of multipath policy guidelines compliant with the ASes business relationships and analyze whether ASSEMBLER is stable under those conditions. Although, our guidelines are not unique and they are not necessarily the only one stable, for profitability reasons, we expect the most common configurations embraced by ASes to follow them in practice. Similar guidelines were presented for BGP in [17]. In this section we discuss which are the additional constraints that must be imposed over the BGP guidelines in order to adapt them to multipath scenarios. Prior to presenting the multipath guidelines, the following two assumptions are stated,

Assumption 1.-: The export policies presented in [17] are followed by multipath nodes as well. An ASes advertises paths coming from its provider only to its customers. Paths coming from peers only to its customers and finally, paths coming from its customers to other customers, peers and providers.

Assumption 2.-: A customer AS cannot be an indirect provider of one of its direct providers. For instance in Fig.3a, if v_4 is a provider of v_1 , then v_4 cannot be a customer/peer of v_3 and v_3 a customer/peer of v_2 which is in turn a customer/peer of v_1 . The assumption is broken, otherwise.

Now, we present the guidelines. The first guideline is an extension of the guideline 5.1 in [17] and the second practical corollary of Theorems 1 and 2. The corresponding stability proof is presented after the statement of each guideline.

Guideline 1.-: If every AS assigns a higher local preference to the paths received from its customers than to paths received from its peers, and it assigns higher preference to paths coming from its peers than to paths coming from its providers, if Assumptions 1 and 2 are respected, then a BGP/ASSEMBLER network is stable.

Proof: The proof tries to construct a reflexive relation. Without loss of generality the proof refer to the scenario in Fig.3a. First we assume that Γ_1 comes from a customer of v_1 , then according to Guideline 1, Δ_1 must come from another customer, otherwise it cannot have higher preference. Then Δ_1 is of the form $\Delta_1 = \Pi_2\Gamma_2$ (in Fig.3a Π_2 is just a link between v_1 and v_2 but in general is an arbitrary path). Since v_1 is a provider of v_2 , according to Assumption 1, the latter can only advertise paths from its customers to v_1 (notice that if intermediate nodes between v_1 and v_2 exists the situation is the same). If Δ_2 has higher preference than Γ_2 and v_2 is following the Guideline 1, then it must come from another customer of v_2 . The same reasoning apply for v_3 . Now, at v_4 , the path Δ_4 through v_1 should be preferred over the path Γ_4 . That can only happen if Δ_4 comes from a customer of v_4 , however if v_1 is a customer of v_4 and v_4 is in the chain of customers from v_1 it means that Assumption 2 is broken.

In the second case, we assume that Γ_1 comes from a peer of v_1 . Therefore, Δ_1 must come from either a peer or a customer of v_1 (Assumption 1). In both cases, it means that Γ_2 and Δ_2 come from customers of v_2 , otherwise they cannot be advertised to a peer or a provider. The chain of customers continue until v_4 . A reflexive relation would be constructed if v_1 is a customer of v_4 . If v_2 is a peer of v_1 , then Δ_1 cannot be advertise to v_4 as it is a v_1 provider. If v_2 is a customer of v_1 we are in the previous case.

In the last case, v_1 learns Γ_1 from a provider, then v_2 can be by a customer, peer or provider of v_1 . If v_2 is a provider of v_1 and Γ_2 and Δ_2 are advertised to v_1 since they come from customers or peers of v_2 , the chain continues like in the two previous cases. Otherwise, if Γ_2 and Δ_2 come from providers of v_2 , then the chain of providers continue to v_4 . If Γ_4 comes from a customer of v_4 then according to Guideline 1 Δ_4 must come from a customer, however v_1 does not advertise its provider v_4 with paths from other providers, therefore Δ_4 is not announced and the reflexive relation is broken. If Γ_4 comes from a peer the situation is the same. Only if Γ_4 comes from a provider, Δ_4 can be more preferred, therefore if v_1 is the provider of v_4 then Δ_4 is advertised, however in that case v_4 becomes the indirect provider of one of its direct providers (through v_3 and v_2) and Assumption 2 is broken.

Guideline 2.-: (multipath relaxation) In addition to following Guideline 1, ASes using ASSEMBLER can aggregate paths coming from the same type of neighbor AS (i.e. customer, peers or providers), no matter ELMPECMP,KBEST setting of ASSEMBLER the system is still stable.

Proof: The proof comes from the stability proof of ASSEMBLER. Revisiting the results pointed out in the previous section, the stability analysis shows that the condition to guarantee stability is the absence of reflexive relations among policies. Those results are derived for *assembled* paths, i.e. advertisements representing a set of paths. Hence, a foregone conclusion is that if the aggregate of paths is equally preferred than the individual paths by the neighbor ASes, the aggregation does not affect the stability, which proves that Guideline 2 is stable.

A counter case of Guideline 2 is the adoption of the Guideline 5.2 in [17], in which path coming from peers can be mixed with paths coming from customers. The latter can lead to different preferences for the individual paths than for the aggregate. The example in Fig.5 in which the aggregate of paths is less preferred by both nodes (filtered) than the individual direct paths, shows how Guideline 5.2 is unstable in ASSEMBLER.

VI. RELATED WORK

This section compares the most relevant BGP-compatible multipath inter-domain routing proposals with ASSEMBLER. Alternative protocols that require a global upgrade of the network (see for instance [29]) are not considered in this section. The first set of solutions comparable to ASSEMBLER achieve backwards compatibility using BGP to exchange the primary path (ensuring backwards compatibility) and they use a parallel protocol or BGP extension to advertise additional

paths. This is the case of R-BGP, which advertise failover paths [30] to achieve fast recovery. BGP Add-Paths [16] is another solution in which routers add a new BGP capability to incrementally advertise extra paths. Finally, MIRO [4] relies also on an additional negotiation of paths. In this case the new path is advertised to the neighbor AS as a tunnel. Although, they are compatible with BGP, these solutions require that two or more neighbor ASes coordinate to deploy multipath border routers. ASSEMBLER does not require of such an incremental/additional negotiation of path and a coordinated deployment between neighbor ASes is not required either.

Another set of multipath inter-domain protocols compatible with BGP do not modify the BGP protocol at all and no additional/incremental advertisement of paths is performed. The first solution is the Multipath-BGP proposed by the manufacturers Cisco and Juniper [15], [14], in which all the considered paths must share the same attributes except for the BGP_ID and interface address of the announcing border router. This type of multipath is intended for a set of particular settings in which several physical connections exists between two ASes. Hence, the multipath set yielded is constrained to have the same AS_PATH attribute in all the routes. As shown in Section III-A, the K-BESTRO path selection algorithm and the assembling technique used by ASSEMBLER does not constrain the path diversity and any set of aggregatable paths can be used concurrently, even if they have different egress ASes.

Some other BGP-compatible protocols propose to advertise one path and use the different alternatives received through BGP to forward the traffic without advertise them. For instance, the inter-domain flavours of Routing Deflections [1] and Path Splicing [5] forward traffic among the available alternative BGP paths according to a *tag* in the packet header. Thus, since BGP advertises only one of those paths, the loop-freeness of the multiple BGP routes cannot be guaranteed, since routes that are not propagated to further ASes are used in practice to forwards the packets. Therefore the control plane information in neighbor ASes is inconsistent with how the traffic is forwarded. The authors in [5] argue that if the common routing policies presented in [17] are followed, no routing loops are possible. In addition, they propose some additional mechanisms to overcome that limitation like deflection counters or include the AS number of the ASes crossed before. An alternative that solves the loop-freeness problem is to propagate the longest available path like in LP-BGP [31], however longer paths are likely to suffer a penalization when compared with other paths at a legacy router.

Thanks to the advertisement scheme presented in Section III-B, ASSEMBLER is able to advertise information that is consistent with the forwarding of traffic currently used. Using ASSEMBLER, the forwarding mechanisms of Routing Deflections and Path Splicing can be used while preserving loop-freeness and without propagating paths that are likely to be considered worse by legacy routers like in [31], since the AS path length attribute in the advertisement is equal to the shortest path within the multipath set.

VII. CONCLUSIONS

In this work, ASSEMBLER a novel protocol for multipath inter-domain routing has been presented. It is the first inter-domain routing protocol that features both, flexible multipath routing and backwards compatibility with BGP, without limiting the path diversity or using another protocol in parallel. Thanks to its design, ASes can benefit from multipath capabilities upgrading progressively their network equipment inside the AS and no coordination or global upgrade is required to take advantage of multiple inter-domain paths.

The characteristics of the multipath set provided by ASSEMBLER can be flexibly tuned using a few parameters to fully exploit the available path diversity or constrain the amount of paths installed in the data-plane (avoiding an exponential growth of the routing tables).

The ASSEMBLER announcements are regular BGP updates generated with an special algorithm such that advertised updates gather information from multiple paths in one message. Those updates can be processed by legacy routers, they are not penalized when compared to regular BGP paths and loop-freeness is maintained. The deployment in a real AS can be carried out progressively and current routing policies and traffic engineering techniques are supported by ASSEMBLER. It can be combined with multipath forwarding techniques to split the traffic amount those installed paths.

The stability of the protocol has been proven in absence of conflicting configurations. Whenever the ASes can simultaneously fulfil their preferences with the available paths, the protocol is able to converge. Two guidelines have been introduced in order to guarantee global stability for every possible configuration of ASSEMBLER.

The adoption of ASSEMBLER as multipath inter-domain routing should provide ISPs with more flexible routing configurations, simplified and dynamic traffic engineering techniques and decrease inter-domain churn.

REFERENCES

- [1] X. Yang and D. Wetherall, "Source selectable path diversity via routing deflections," in *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2006, pp. 159–170.
- [2] S. Kandula, D. Katabi, S. Sinha, and A. Berger, "Dynamic load balancing without packet reordering," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 2, pp. 51–62, 2007.
- [3] J. He and J. Rexford, "Toward internet-wide multipath routing," *Network, IEEE*, vol. 22, no. 2, pp. 16–21, 2008.
- [4] W. Xu and J. Rexford, "MIRO: multi-path interdomain routing," in *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2006, pp. 171–182.
- [5] M. Motiwala, N. Feamster, and S. Vempala, "Better interdomain path diversity with BGP path splicing," 2007.
- [6] B. Ramamurthy, G. Rouskas, and K. Sivalingam, *Next-Generation Internet: Architectures and Protocols*. Cambridge Univ Pr, 2011.
- [7] D. Meyer, "University of oregon route views archive project," at <http://archive.routeviews.org>.
- [8] D. Wischik, M. Handley, and M. Braun, "The resource pooling principle," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 5, pp. 47–52, 2008.
- [9] Y. Rekhter, T. Li, and S. Hares, "RFC 4271: a Border Gateway Protocol 4 (BGP-4)," *Internet Engineering Task Force, Tech. Rep.* 2006.
- [10] Cisco, "Load sharing with bgp in single and multihomed environments: Sample configurations," http://www.cisco.com/en/US/tech/tk365/technologies_configuration_example09186a00800945bf.shtml.

- [11] —, “How does load balancing work?” http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a0080094820.shtml.
- [12] F. Guo, J. Chen, W. Li, and T. Chiueh, “Experiences in building a multihoming load balancing system,” in *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2. IEEE, 2004, pp. 1241–1251.
- [13] C. Hopps, “Analysis of an Equal-Cost Multi-Path Algorithm, IETF,” Internet RFC 2992, Novembre 2000, Tech. Rep.
- [14] Juniper, “Configure bgp to select multiple bgp paths,” <http://www.juniper.net/techpubs/software/junos/junos53/swconfig53-ipv6/html/ipv6-bgp-config29.html>.
- [15] Cisco, “Bgp best path selection algorithm,” http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a0080094431.shtml.
- [16] V. Van den Schrieck, P. Francois, and O. Bonaventure, “BGP add-paths: the scaling/performance tradeoffs,” *Selected Areas in Communications, IEEE Journal on*, vol. 28, no. 8, pp. 1299–1307, 2010.
- [17] L. Gao and J. Rexford, “Stable Internet routing without global coordination,” *IEEE/ACM Transactions on Networking (TON)*, vol. 9, no. 6, pp. 681–692, 2001.
- [18] R. Teixeira, A. Shaikh, T. Griffin, and G. Voelker, “Network sensitivity to hot-potato disruptions,” in *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4. ACM, 2004, pp. 231–244.
- [19] T. Griffin, F. Shepherd, and G. Wilfong, “The stable paths problem and interdomain routing,” *IEEE/ACM Transactions on Networking (TON)*, vol. 10, no. 2, pp. 232–243, 2002.
- [20] C. Chau, “Policy-based routing with non-strict preferences,” in *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 2006, pp. 387–398.
- [21] Y. Wang, M. Schapira, and J. Rexford, “Neighbor-specific BGP: more flexible routing policies while improving global stability,” in *Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems*. ACM, 2009, pp. 217–228.
- [22] Y. Wang, I. Avramopoulos, and J. Rexford, “Design for configurability: rethinking interdomain routing policies from the ground up,” *Selected Areas in Communications, IEEE Journal on*, vol. 27, no. 3, pp. 336–348, 2009.
- [23] T. McGregor, S. Alcock, and D. Karrenberg, “The RIPE NCC internet measurement data repository,” in *Passive and Active Measurement*. Springer, 2010, pp. 111–120.
- [24] T. S. A. Labs, “Best practices for network interconnection,” NANOG 43.
- [25] E. Rosen, Y. Rekhter *et al.*, “Bgp/mps vpns,” 1999.
- [26] M. Caesar and J. Rexford, “BGP routing policies in ISP networks,” *Network, IEEE*, vol. 19, no. 6, pp. 5–11, 2005.
- [27] B. Quoitin, C. Pelsser, L. Swinnen, O. Bonaventure, and S. Uhlig, “Interdomain traffic engineering with BGP,” *Communications Magazine, IEEE*, vol. 41, no. 5, pp. 122–128, 2003.
- [28] D. Bertsekas and J. Tsitsiklis, “Parallel and distributed computation,” 1989.
- [29] P. Godfrey, I. Ganichev, S. Shenker, and I. Stoica, “Pathlet routing,” in *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*. ACM, 2009, pp. 111–122.
- [30] N. Kushman, S. Kandula, D. Katabi, and B. Maggs, “R-BGP: Staying connected in a connected world,” in *Proc. NSDI, 2007*, pp. 341–354.
- [31] I. van Beijnum, J. Crowcroft, F. Valera, and M. Bagnulo, “Loop-freeness in multipath BGP through propagating the longest path,” in *Communications Workshops, 2009. ICC Workshops 2009. IEEE International Conference on*. IEEE, 2009, pp. 1–6.