MIFO: Multi-Path Interdomain Forwarding

Ming Zhu*†, Dan Li*†, Ying Liu*†, Dan Pei*†, K.K. Ramakrishnan‡, Lili Liu*†, Jianping Wu*†

*Tsinghua National Laboratory for Information Science and Technology
†Department of Computer Science and Technology, Tsinghua University ‡University of California, Riverside, USA

***Abstract*—Today's interdomain routing is traffic agnostic when determining the single, best forwarding path. Naturally, as it does not adapt to congestion, the path chosen is not always optimal. In this paper, we focus on designing a multi-path interdomain forwarding (MIFO) mechanism, where AS border routers adaptively forward outbound traffic from a congested default path to an alternative path, without touching the interdomain routing protocols. Different from previous efforts which enable multi-path on control plane, MIFO achieves multi-path on data plane. The multiple alternative forwarding paths are obtained by exploring local BGP RIB.**

Multi-path forwarding on data plane can create a loop even within a stable network. MIFO solves this problem with a simple and practical approach. Several other challenges are also addressed including preventing cycling packet between iBGP peers and choosing the best alternative path from among multiple candidates. Our evaluations show that MIFO significantly improves the end-to-end throughput at the AS level, compared to traditional BGP and MIRO. For example, with only 50% of the ASes being MIFO capable, a significant percentage of the flows (about 40%) can use at least 50% of the inter-AS link capacity. In contrast, BGP and MIRO routing make less effective use of the inter-AS links, with only 7% and 17% of the flows can be so. Finally, we have developed a prototype implementation of MIFO on Linux with the forwarding engine in the kernel, with the routing daemon developed on XORP platform. The experiments on a testbed built with prototypes show that MIFO can improves the aggregate throughput by 81% compared with BGP routing.

I. Introduction

A significant challenge facing today's Internet is the rapid growth of interdomain traffic volume [1]. Daily traffic in large Internet Exchange Points (IXP) is often in tens of petabytes (PB) [2]. Meanwhile, the hierarchical structure of the Internet is becoming flatter [3], which leads to abundance of paths existing between every pair of ASes. However, BGP, as the dominant interdomain protocol, fails to take advantage of the availability of a multitude of paths to increase the end-to-end network throughput. Instead, today's BGP routers only use a single, default path, which when congested leads to even poorer performance. With BGP being traffic agnostic, it fails to react to congestion and burstiness in the traffic [4]. The most recent industrial report [5] confirms that congestion on the default path between popular ISPs (*e.g.,* Verizon and Level3) is increasingly common.

There are two fundamentally causes for this. First is the conflict between the rapid growth of traffic load and BGP's rigid, *static*, single-path routing principle. When there is higher load, the lack of ability to use all available paths means that the one path used is congested because all the traffic is forwarded on that path. Secondly, the mismatch between *fast* dynamics of traffic and *slow* route convergence results in poor end-to-end network performance. More specifically, *dynamic* traffic load leads to congestion happening on data plane, while changing forwarding path is realized in control plane, such as reconfiguring the weight in routing protocols (*e.g.,* local preference in BGP) or statically switching into an alternative path. The decoupling between control plane and data plane makes it difficult to adapt.

In this paper we design **M**ulti-Path **I**nterdomain **FO**rwarding (MIFO), which enables AS border routers to adaptively offload outbound traffic from a congested default path to an alternate path with only data plane modification, rather than having the load information communicated to the control plane and using it to recognize alternate paths. Specifically, MIFO explores multiple paths learnt by the control plane and uses load-aware forwarding at the data plane over different outgoing AS paths, so as to keep forwarding in line speed.

AS border routers in MIFO directly utilize existing multiple paths obtained from the local BGP RIB, without requiring any additional protocol exchange in the control plane. In the BGP RIB, different paths are learned from different AS neighbors. It is therefore better to achieve multi-path forwarding on data plane in order to reduce the overhead of looking up in the RIB.

MIFO is an entirely distributed system and is compatible with legacy routing system. However, several challenges must be addressed in designing a practical solution. The *first* and the greatest challenge is, *in a distributed routing system, multi-path forwarding can always create a loop even within a stable network.*Within present interdomain

This work is supported by the National Key Basic Research Program of China (973 program) under Grant 2014CB347800, the National Natural Science Foundation of China under Grant No.61170291, No.61432002, the National High-tech R&D Program of China (863 program) under Grant 2013AA013303, and Tsinghua University Initiative Scientific Research Program.

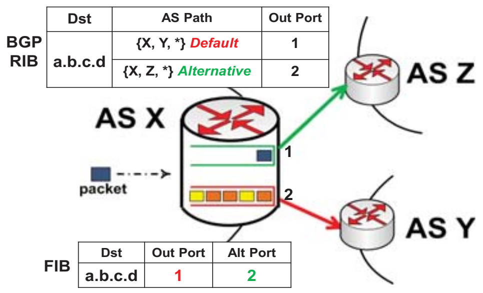


Figure 1: MIFO Overview.

routing protocols (*e.g.,* BGP), whenever a router forwards packets along with an alternative path instead of the default path, a loop can be generated. We illustrate an example in Section II-C to show that such loop can exist even under a simple topology (*e.g.,* only 3 nodes). The reason is the inconsistency between control plane and data plane. It is "loop-free" on control plane since packet is considered to be always forwarded along the default path. However, on data plane, the packet is possibly sent to an alternative path to bypass a default congested outgoing link. In this case, the loop may be generated in data plane, but is invisible to control plane. The *second* challenge is, a cycle may be produced once packets are deflected from the default egress router to another egress router directing to an alternative path, either in MIFO or in BGP. The reason is the inconsistency between iBGP peers: except for the default egress router itself, all the other routers are unaware of the congestion on the default path and thus sending packets back to the default egress router. The *third* challenge is, once the default path is congested, it is important to make the border router select the best alternative path for subsequent packets in line speed, in a fully distributed manner.

MIFO addresses the first challenge by leveraging a "valley-free" rule into the data plane. More precisely, we add a simple and practical constraint in the data plane to let routers forward packets following a *loop-free* path. We not only prove the correctness of this idea in theory, but also propose an effective implementation by using just one bit in the packets, which are supposed to be forwarded along with an alternative path. In particular, MIFO consumes one bit in packet header to denote the "valley-free" rule on data plane. IP-in-IP encapsulation between iBGP routers is applied to address the second challenge. By checking the outer IP header, an iBGP router can easily decide whether it can forward the packet to the iBGP peer on the default path or not. Finally, to address the third challenge, border routers measure the path quality by selective probing. In the current design, we also come up with a greedy solution to let border routers select the best alternative path depending on local link capacity.

Extensive simulations based on traced AS-level Internet topology and reasonable generated traffic show that MIFO can significantly improve the end-to-end network throughput at the AS level, compared to traditional BGP and MIRO. For example, with only 50% of the ASes being MIFO capable, a significant percentage of the flows (about 40%) can use at least 50% of the inter-AS link capacity, *i.e.,* the average throughput of these flows is 500Mbps if the link capacity is 1Gbps. In contrast, BGP and MIRO routing make less effective use of the inter-AS links, with only 7% and 17% of the flows attain 50% of the link capacity. We have also developed a prototype implementation of MIFO on Linux with the forwarding engine in the kernel, with the routing daemon developed on XORP platform. The experiment on a testbed built with prototypes show that MIFO can improves the aggregate throughput by 81% compared with BGP routing.

The rest of the paper is organized as follows. The following section presents the overview of MIFO. Section III analyzes the root cause of loops in multi-path forwarding and prove the correctness of our solution, as well as the design details. Section IV evaluates MIFO's performance. Section V describes our prototype implementation of MIFO. Section VI discusses related work. Finally, Section VII concludes the paper.

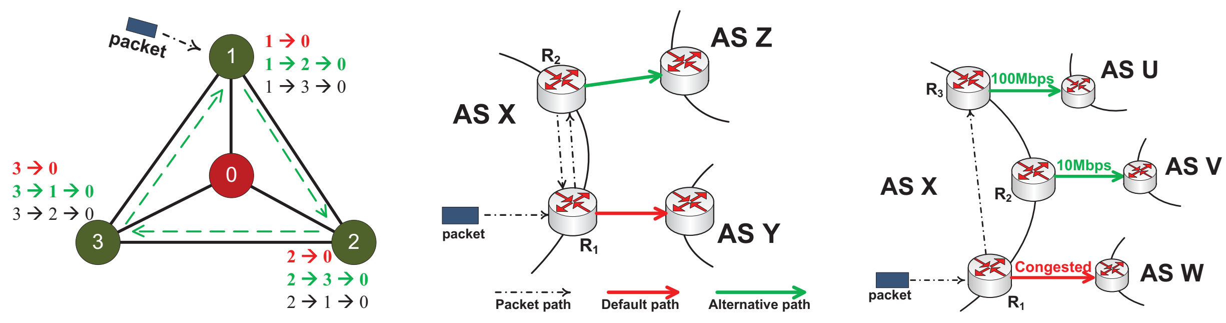
II. Overview and Technical Challenges

In this section we present the basic idea and technical challenges in designing MIFO.

A. Basic Idea

We now illustrate the basic idea of MIFO. As shown in Figure 1, AS X learns two AS paths towards destination $a.b.c.d$ (for convenience we ignore the length of prefix in our notation). Assume that the path learned from AS Y is selected as the default path (red arrow), while the path learned from AS Z is selected as the alternative one (green arrow). We slightly modify the FIB by adding *alt port* filed, which corresponds to the alternative path, as shown in Fig. 1. As the traffic volume towards $a.b.c.d$ through AS X grows rapidly, congestion may occur on the default path. More specifically, on the border router, more packets accumulate in the *tx* queue of the default output port, *i.e.,* port 1. With MIFO, the router could adaptively deliver subsequent packets for $a.b.c.d$ to port 2, which corresponds to an underutilized alternative path $\{X, Z, *\}$. Note that MIFO does not specify how to identify the congestion on border routers. It is an open to different congestion definitions. Throughout this paper, we simply denote the queuing ratio of output ports as the congestion signal.

To keep forwarding at line speed, MIFO implements path selection in the data plane. In particular, once congestion occurs, the border router directly forwards the packet to the alternate port denoted in the FIB, rather than looking up the BGP RIB in the control plane. Such a design significantly



(a) Loop on data plane.
(b) Cycling between iBGP peers.
(c) Alternative path selection.

Figure 2: Three challenges.

reduces the overhead caused by switching between the two planes, considering that congestion occurs frequently on interdomain border routers [6]. The alternate port in the FIB is updated by a MIFO daemon.

To avoid packet reordering issues, forwarding is deterministic at the flow level. As Fig. 1 shows, packets with same color belong to the same flow. The eventual path for subsequent packet is determined by hashing[1]. MIFO prefers the alternative path that has the higher available bandwidth. More details of path selection is introduced in Section III-C.

B. Design Rationale

Although several studies [7–10] have explored multi-path interdomain routing, the additional overhead in obtaining alternative paths extremely limits them for practical use. For example, to make every AS advertise alternative paths, either a new message or an extra BGP attribute has to be introduced. Moreover, routers have to maintain another routing table to track the alternate paths in fast memory. In our design, we aim at obtaining multiple paths with *zero overhead*.

To achieve this, we just utilize existing paths in BGP RIB, where different paths are learned from different neighboring ASes. Given that paths are valid in BGP RIB (*e.g.,* loop-free), no further examination is required. The degree of path diversity gained by an AS is therefore dependent on how many neighbors it has. By examining the BGP RIB provided by Routeview [11], we found that most of ASes are able to benefit from multi-neighbor forwarding.

C. Challenges

Traffic-sensitive forwarding via MIFO can be easily deployed on border routers while being compatible with legacy routers. The key idea is to leverage alternative paths in forwarding packets while avoiding congestion on default paths. However, several challenges must be addressed in designing a practical solution.

[1] For example, a flow is identified by a five-tuple: source address, destination address, source port, destination port, protocol type.

Loops on the Data Plane. We assume that the network is in the stable state, meaning traditional loops caused by inconsistent route views in a transient state are not considered. The standard route selection rule is also adopted, meaning that customer routes are preferred over peer routes, which in turn are preferred over provider routes. In single path routing, the full AS path vector in a BGP announcement guarantees that each AS can avoid loops by filtering the paths which contain its own number. However, for multi-path routing, loops can still exists. More specifically, we state that, *within present inter-domain routing protocols (*e.g., *BGP), whenever a router forwards packets along an alternative path instead of the default path, a loop can be generated.*

Fig. 2(a) illustrates an example. ASes 1, 2 and 3 are peering with each other, while AS 0 is their customer. By following the route selection criteria, each AS uses the direct link to AS 0 as its default path and routes via its neighbors as its alternative path. Once congestion occurs on the default path, rather than dropping the packets, each AS would leverage the alternative paths in forwarding packets. Considering the worst case where every default path (red route) is congested, each AS may forward the packet along with an alternative path (green route). As the dash lines show, a loop is generated if each AS chooses the alternative path in a clockwise direction. The root cause is the lack of a constraint in forwarding packets on data plane. Recognizing this, we demonstrate a simple and practical approach to solve this problem in section III-A2.

Cycling between iBGP peers. The multiple paths at the AS-level can be translated to different AS border routers. Take Fig. 2(b) as an example, in contrast to Fig. 1. The alternative path is derived from another border router compared to the default path. Once packets are deflected from the default path to an alternative path, a cycle may be produced between iBGP peers. As Fig. 2(b) shows, only two paths are available towards the destination, where $XY*$ and $XZ*$ are selected as the default path and alternative

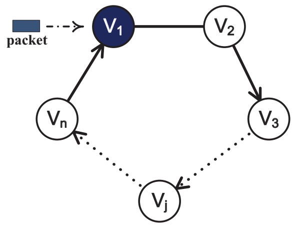


Figure 3: Formalized Loop.

path respectively. Assume that the default path is congested. Rather than sending packets to a eBGP peer, R_1 prefers to forward packets to an iBGP peer, R_2, so as to deliver packets through the alternative path. However, R_2 will send packets back to R_1, which produces a cycle. The reason is that the iBGP peers are out of sync in the data plane. In particular, R_2 is unaware of congestion on the default path through R_1 and still delivers packets to R_1 to use the default path. Section III-B presents an easy but elegant solution to solve this problem.

Selecting the Best Alternative Path in a Distributed Manner. Given that MIFO works in an entirely distributed way, once congestion occurs on the default path, the border routers of every AS have to independently choose the best alternative path for each subsequent packet. Take Fig. 2(c) as an example. As soon as congestion is observed on the default path, the default egress router (R_1) has to deliver subsequent packets to an alternative path with the most available bandwidth. Therefore, R_1 prefers the alternative path through R_3 since it has a larger available bandwidth (100Mbps) than the path through R_2 (10Mbps). We propose a simple, but practical, greedy method to address this challenge and is detailed in Section III-C.

III. MIFO Design

In this section we design MIFO in details, in a particulary notice on how to address the challenges elaborated in the previous section.

A. Breaking the Loop on Data Plane

1) The Root Cause of Loop: In multi-path forwarding, when an AS decides to forward a packet, the full path is indeterminate. As Fig. 2(a) shown, when AS 1 forwards packets to AS 2, it supposes that packets will follow the path described in control plane, $1 \rightarrow 2 \rightarrow 0$. However, the real path in data plane is $1 \rightarrow 2 \rightarrow 3 \rightarrow 1$. This is different from existing single-path routing, where data plane strictly follows the default path selected by control plane. *The inconsistency between control plane and data plane is the root cause of generating loop in multi-path forwarding.* Therefore, to prevent the loop: *i)* we can force the data plane definitely follows control plane, or *ii)* examine the exact packet path before forwarding, or *iii)* add some constraints in data plane to drop the packet automatically.

The first approach falls back to single path routing by only using default path, *i.e.,* AS 1 either forwards packets to AS 0 or drops them. The second approach means each AS knows instantaneous global information, *i.e.,* AS 1 will drop the packet right away as soon as it founds that AS 3 and AS 2 are both congested on default path. To achieve this, the link utilization between each pair of ASes has to be endlessly propagated on Internet, which is unscalable and impractical. Therefore, we adopt the third approach to break the loop: *add a simple constraint on data plane*.

2) Main Idea of Loop Breaking: Currently, "valley-free" [12] policies are only used on control plane to match economic incentives, whereby routes through peers and providers are exported only to customers, and customer routes are exported to everyone. However, we prove that the loop issues can be easily resolved by building a similar regulation into data plane. Specifically, before forwarding packets to an alternative path, current AS will examine the business relationship with upstream neighbor (***UN***) and downstream neighbor (***DN***). Here upstream and downstream denote the direction of packets, incoming and outgoing respectively. The procedure of path verification is described as following:

- If UN is a customer: forward the packet to this path.
- If UN is a provider or a peer: forward the packet to this path if and only if DN is a customer.
- Otherwise, drop the packet.

Take Fig. 2(a) for instance, when AS 2 receives packets (towards AS 0) from AS 1, by learning that default path ($2 \rightarrow 0$) is congested, it will choose an alternative path ($2 \rightarrow 3 \rightarrow 0$) to forward packets. However, other than sending packets to AS 3, AS 2 should directly drop packets since both upstream neighbor (AS 1) and downstream neighbor (AS 3) are AS 2's peers, which violates the rule of path verification. Consequently, the potential loop ($1 \rightarrow 2 \rightarrow 3 \rightarrow 1$) in data plane is immediately cut off.

3) Correctness of the Idea: In our definition, a packet is forwarded in loop-free if the packet reaches the destination without loop, or the packet is dropped before forming a loop. We then draw the conclusion and prove the correctness as follow:

Theorem. *Within multi-path interdomain forwarding, by adding valley-free regulation in data plane, each packet is forwarded in loop-free.*

Proof: We first declare that the necessary condition to generate a loop is that, at least one AS adopts the alternative path other than the default path. This is an obvious fact since BGP already ensures loop-free on default path through full AS-path vector.

We then focus on the graph, $G(V, E)$, where vertices $\{v \in V\}$ denote ASes and edges $\{e \in E\}$ denote inter-AS links. The business relationship between adjacent ASes can be defined as algebraic logic:

$$\begin{aligned} v_i &< v_{i+1}, \quad \text{if } (v_i, v_{i+1}) \text{ is (customer, provider)} \\ v_i &= v_{i+1}, \quad \text{if } (v_i, v_{i+1}) \text{ is (peer, peer)} \\ v_i &> v_{i+1}, \quad \text{if } (v_i, v_{i+1}) \text{ is (provider, customer)} \end{aligned}$$

Note that transitivity is only valid between customer and provider, saying that:

$$\text{if } v_{i-1} > v_i \text{ and } v_i > v_{i+1}, \text{ then } v_{i-1} > v_{i+1} \quad (1)$$

$$\text{if } v_{i-1} < v_i \text{ and } v_i < v_{i+1}, \text{ then } v_{i-1} < v_{i+1} \quad (2)$$

However, it is not suitable in peering: if $v_{i-1} = v_i$ and $v_i = v_{i+1}$, v_{i-1} and v_{i+1} can have any kinds of relationship. Accordingly, the procedure of path verification can be formalized as: v_i is allowed to transit packets from v_{i-1} to v_{i+1} if and only if:

$$v_{i-1} < v_i \quad \text{or} \quad v_i > v_{i+1} \quad (3)$$

We next complete the proof by contradiction. As Fig. 3 shown, we denote v_1 to represent the node who takes alternative path other than default path, which results in a loop. We assume the loop is clockwise (counterclockwise is the same), means packets are transferred through $v_1 \rightarrow v_2 \rightarrow ...v_n \rightarrow v_1 \rightarrow v_2...$ where $n > 2$ and v_i also represents v_{n+i}.

If $\boldsymbol{v_1 > v_2}$, with Eq. 3, v_2 can forward packets to v_3 if and only if $v_2 > v_3$. In a similar fashion along the loop, we have $v_1 > v_2 > v_3... > v_n$. According to Eq. 1, it leads to $v_1 > v_n$. However, to construct the loop, v_n should also forward packet to v_1. Given that $v_{n-1} > v_n$, by following Eq. 3, it leads to $v_n > v_1$. Contradiction is generated.

If $\boldsymbol{v_1 = v_2}$, with Eq. 3, it turns to $v_2 > v_3$. Applying Eq. 3 along the loop, we have $v_2 > v_3 > ... > v_n > v_1$. However, if $v_n > v_1$, by Eq. 3, v_1 can forward packet to v_2 if and only if $v_1 > v_2$, which contradicts with $v_1 = v_2$.

If $\boldsymbol{v_1 < v_2}$, to form a loop, v_n should forward the packet to v_1. With Eq. 3, it turns to $v_n < v_1$. Applying Eq. 3 in a counterclockwise direction, we have $v_2 < ... < v_{n-1} < v_n < v_1$. According to Eq. 2, it leads to $v_2 < v_1$, which contradicts with $v_1 < v_2$.

Therefore, by adding valley-free regulation on data plane, it is loop-free in multi-path interdomain routing. ■

4) One More Bit is Enough: In AS level, the business relationship with neighbors is predefined in control plane. However, in our system, such information is also requested in data plane to achieve loop-free packet forwarding. Fig. 4 illustrates the forwarding process through three ASes, $V_{i-1} \rightarrow V_i \rightarrow V_{i+1}$. On one hand, as the packet entering point in AS V_i, R_1 is the only device who knows the upstream neighbor is V_{i-1} because of the direct connection. On the other hand, as the packet exit point in AS V_i, R_2 is the only device who knows the downstream neighbor is V_{i+1} (on alternative path) other than V^*_{i+1} (on default path).

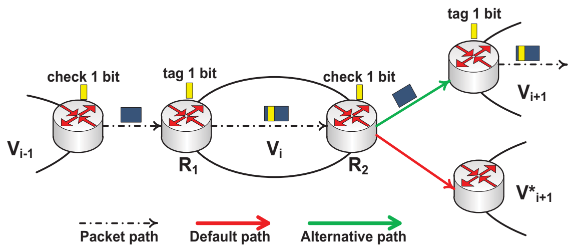


Figure 4: Tag-Check Strategy.

Therefore, to obtain a forwarding path in loop-free: R_1 should tell R_2 that the upstream neighbor is V_{i-1}, and R_2 has to examine the relationships of (V_{i-1}, V_i) and (V_i, V_{i+1}) in terms of Eq. 3. *The key is to pass the relationship with upstream neighbor from the packet entering point to the packet exit point.* To achieve this, we show that "one bit is enough".

Take Fig. 4 as an example. To transit the packet from V_{i-1} to V_{i+1}, border router R_2 on V_i is required to verify that either $V_{i-1} < V_i$ or $V_i > V_{i+1}$. Given that the relationship between V_i and V_{i+1} is known due to the direct connection, R_2 only needs to determine whether $V_{i-1} < V_i$ or not. Such information can be easily expressed by one bit: 1 denotes $V_{i-1} < V_i$ and 0 denotes $V_{i-1} \geq V_i$. Therefore, a simple "Tag-Check" approach is used to fulfill the path verification on data plane:

- When the border router (*e.g.,* R_1) receives a packet from an eBGP peer, it tags one bit to identify the relationship with corresponding upstream neighbor.
- When the border router (*e.g.,* R_2) intends to deliver a packet to an eBGP peer, it checks that bit in terms of Eq. 3.

It is easy in deploying "Tag-Check" strategy in practise. Multi-Protocol Label Switching (MPLS) is widely deployed in ASes [13], where a label is inserted on each incoming packet at entering point and removed at the exit point. This is just right for "Tag-Check" strategy by consuming an unused bit in the label. Even for the ASes without using MPLS, it could be accomplished by taking one reserved bit in IP header or allocate one bit in IP option filed.

B. Cycle Avoidance via IP-in-IP Encapsulation

The primary cause of cycling packets in Fig. 2(b) is the asynchronism between iBGP peers. More specifically, R_2 is unaware of the congestion through R_1, and thus sends packets back to R_1 to use the default path. Therefore, a notification mechanism is requested to keep iBGP peers have a consistent view on data plane.

We adopt the means of IP-in-IP encapsulation for several promising reasons. First, it is practical given that IP-in-IP function is widely supported in current commercial routers. Second, the reliability is guaranteed based on the existing

Algorithm 1 Pseudocode of forwarding process

```
Input:
    p: Incoming packet;    I_in: input port;
1:  if isIPinIP($p$)                                    ▷ IP-in-IP packet
2:      $s \leftarrow$ GetPacketSender($p$)             ▷ Sender is iBGP peer
3:      StripOuterIPHeader($p$);
4:  $I_{out}, I_{alt} \leftarrow$ FIBLookup($p$);
5:  if isConnectToEBGP($I_{in}$)                        ▷ Entering Point
6:      $V_{up} \leftarrow$ GetNeibor($I_{in}$);        ▷ Get Upstream Neighbor
7:      if isCustomer($V_{up}$)                         ▷ Tag
8:          SetOneBit($p$);
9:      else
10:         ClearOneBit($p$);
11: if isCongest($I_{out}$) or $s =$ GetNextHop($I_{alt}$)
12:     if isConnectToIBGP($I_{alt}$)                   ▷ Alt path on iBGP peer
13:         Encap($p$);                                 ▷ Encapsulate by IP-in-IP
14:         SendTo($p$, $I_{alt}$);                     ▷ Send to iBGP peer
15:         return
16:     $V_{down} \leftarrow$ GetNeibor($I_{alt}$);
17:     if isCustomer($V_{down}$) or isSetBit($p$)      ▷ Check
18:         SendTo($p$, $I_{alt}$);                     ▷ Alt path is Valley Free
19:     else
20:         Drop($p$);
21:     return
22: SendTo($p$, $I_{out}$);
23: return
```

connection between each pair of iBGP peers. Finally, this is a lightweight in-band approach, without any additional messages.

The border router relies on the encapsulated IP header to determine the forwarding path: default one or alternative one. We take Fig. 2(b) as an instance to describe our non-trivial approach. R_1 is the final hop to exit local AS towards the destination. Once R_1 sees congestion on default outgoing link (red arrow), it encapsulates each subsequent packet with an outer IP header and forwards it to an iBGP peer, R_2, to leverage the alternative path (green arrow). The source address and destination address in outer IP header are R_1 and R_2 respectively. Once R_2 receives the packet, it identifies the sender (R_1) by extracting the source address and obtains the origin packet by decapsulating the outer IP header. To forward the origin packet, R_2 primarily compares the corresponding nexthop with the sender. If the nexthop equals to sender (means R_1 is the nexthop as default), it indicates the packet is "deflected" from the default path because of the congestion. R_2 thereupon sends the packet to an alternative path (green arrow) instead. Otherwise, R_2 forwards packet to the nexthop (R_1) directly.

C. Selecting the Best Alternative Path

The priority of an alternative path is identified by the available bandwidth. Within an AS, the border routers periodically measure the available bandwidth to other ASes and communicate the measurement results with each other. As shown in Fig. 1, the alt port in FIB is corresponded to the alternative path with most available bandwidth.

Given that each pair of border routers are iBGP peer, exchanging measurement results is easily implemented by reusing the TCP session between them. Therefore, the key step is to efficiently realize the measurement on border routers. Plenty of measurement tools are developed in past years [14]. However, none of these is suitable in our system. On one hand, packets are fast forwarded while end-to-end bandwidth estimation is extremely slow in comparison. On the other hand, it is impractical to constantly measure such many destinations, given that current BGP table has 500K entries [15]. Even in AS level, there are 50K [15] potential targets, which is still unacceptable.

In our design, we adopt a greedy method by simply turning "path" measurement into "link" monitoring. Specifically, the border router takes remaining capacity on direct connected inter-AS link as the path available bandwidth. Take Fig. 2(c) as example, other than probing the path bandwidth [14], R_2 and R_3 merely monitor the spare capacity on link XV and link XU respectively. Such greedy scheme provides several advantages:

- It is considerably easier in exploring local link than sending probe packets with complicated estimation algorithm.
- Comparing with deferred path measuring, local link monitoring achieves real-time results.
- As for a router, the amount of links is far less than the number of ASes to be monitored.

D. MIFO Forwarding Engine

The MIFO forwarding engine on data plane is described in pseudocode 1. When the border router receives a packet, it examines the packet type at first. If the packet is encapsulated by IP-in-IP (Line 1), it extracts the packet sender (Line 2) and strips the outer IP header to obtain the origin packet (Line 3). The default output port and alternative output port are acquired by looking up FIB (Line 4). If the packet comes from an eBGP peer (Line 5), it means current router is the packet entering point. Accordingly, it should check the relationship with upstream neighbors (Line 6-7) and set the bit to one if upstream neighbor is a customer (Line 8), or clear the bit to zero in otherwise (Line 10). Finally, it sends the packet to intradomain (Line 22). As described in Section III-B, the router can detect congestion by either seeing it on output port or receiving an encapsulated packet from the default path (Line 11). Therefore, the packet should be forwarded through the alternative path. If the alternative path is on an iBGP peer (Line 12), the router encapsulates

Date	# of Nodes	# of Links	P/C Links	Peering Links
11/2014	44,340	109,360	75,046	34,314

Table I: Attributes of Data-set.

the packet (Line 13) and sends it to that iBGP peer (Line 14). If the alternative path is on an eBGP peer, it forwards the packet to the alternative path (Line 18) if path is verified (Line 17), or drops the packet in otherwise (Line 20).

IV. Performance Evaluation

In this section, we study the performance of MIFO and compare it with conventional BGP and MIRO [7] by using an NS-3 simulation. MIRO implements multi-path interdomain routing by letting ASes advertise alternative routes to neighbors. To achieve scalability, MIRO strictly limits the number of routes that each AS can advertise. In our simulation, all the link capacities are set to 1Gbps. The MIFO forwarding engine is implemented on each routing node. The evaluation is built on an actual AS-level topology [16] trace. In addition to every AS representing a node, we expand several tier-1 ASes to capture all of their internal topologies at the router level; in doing so, we assume all the border routers (iBGP peers) within a tier-1 AS are connected in a full mesh topology.

We first show that MIFO provides sufficient path diversity, and then evaluate the throughput improvement. Given that a complete view of interdomain traffic matrix is difficult to obtain because of proprietary restrictions, we instead generate a synthetic traffic matrix in two ways. One is to randomly choose a pair of ASes as the source and destination, so as to analyze MIFO in a generic manner. The second is to think of popular content providers as traffic producers, such as Google (AS 15169) and Facebook (AS 32934), and take stub ASes as traffic consumers. We assume that the traffic follows a power-law distribution, in that the higher a content provider ranks (by the number of providers and peers), more of its traffic is consumed. We believe that this helps to have a more realistic evaluation of typical real-world workloads. The start time for each flow follows a Poisson process such that the average number of flows initiated in one second is set to 100. The flow size is chosen to be 10MB and packet size is set to 1KB.

A. Path Diversity

Given that both MIFO and MIRO exploit multiple paths, we first study the path diversity with the AS-level topology [16]. Table I summarizes the key features of the AS topology. [16] measured global AS-links and inferred the business relationship on each link, indicating that 69% are provider-customer (P/C) and 31% are mutual peering relationships.

To build the BGP RIB on each node, we use the standard "valley-free" [12] export policies and route selection criteria,

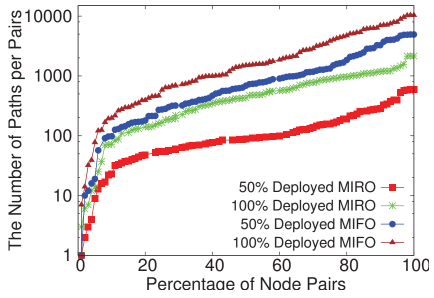


Figure 7: Available Paths Comparison.

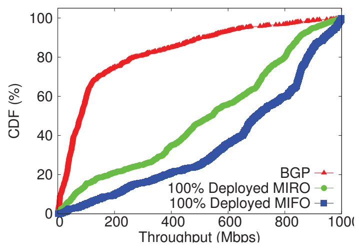
with customer routes preferred over peer routes, which in turn are preferred over provider routes. If multiple routes fall in the same category, the first tie breaker is the length of AS paths, and the second is the lowest next-hop AS identifier. As for MIRO, we adopt the strict policy described in [7], which indicates that each AS only announces the alternative paths with the same local preference as the default path.

Fig. 7 illustrates the available paths between each AS pair in MIFO and MIRO. In particular, we present the results both for fully deployment and partial deployment. It is notable that with even half of the ASes being MIFO capable, more paths are available than when MIRO is fully deployed. The reason is that the AS with MIRO only announces a fraction of the alternative routes for the sake of scalability. In contrast, the ASes with MIFO are able to leverages all the available paths. In addition, as for the 100% deployment of MIFO, 90% of the AS pairs have at least a hundred alternative paths, and nearly half of the AS pairs have thousands of available paths.

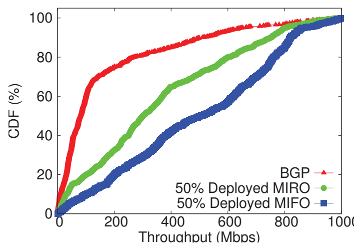
B. Throughput Improvement

Given an AS-level traffic matrix, we are interested in the end-to-end throughput of flows compared to conventional BGP routing (single path) and MIRO. We first take an uniformly distributed traffic matrix with *one million* flows. For each traffic flow, the source, destination are determined by randomly choosing a pair of ASes. Fig. 5 illustrates the cumulative distribution of flow throughput for varying ratios of the deployment of MIFO and MIRO. As expected, both MIFO and MIRO significantly dominate BGP since there is insufficient network capacity when using single path routing. When MIFO and MIRO are fully deployed, MIFO outperforms MIRO significantly. Only half of the flows get more than 500 Mbps end-to-end throughput in MIRO, while nearly 80% flows can achieve more than 500 Mbps under MIFO. Within 50% deployment of MIFO, it still enables half of the flows to achieve 500 Mbps throughput, while only 35% flows reach this in MIRO. Even under 10% deployed ratio, MIFO outperforms MIRO as well.

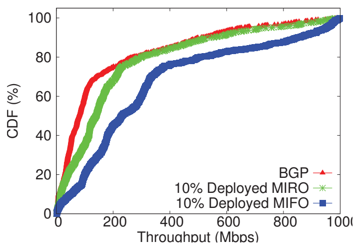
Next, we turn to a traffic matrix distributed according to the power-law distribution, which is closer to a real-world



(a) 100% Deployment

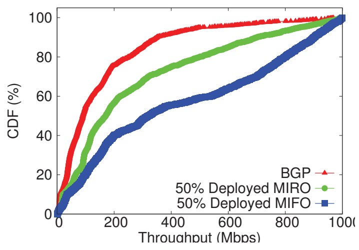


(b) 50% Deployment.

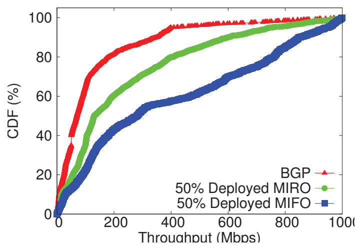


(c) 10%. Deployment

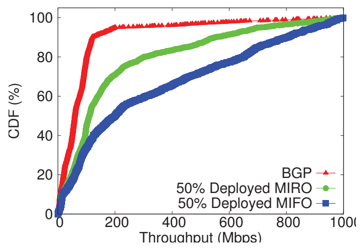
Figure 5: Throughput Comparison against Different Deployment Ratio.



(a) $\alpha = 0.8$



(b) $\alpha = 1.0$



(c) $\alpha = 1.2$

Figure 6: Throughput Comparison against Power-Law Traffic Generation.

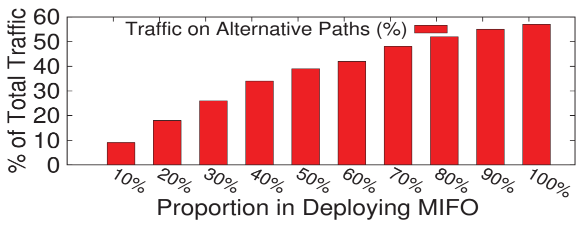


Figure 8: Traffic Offload on Alternative Paths.

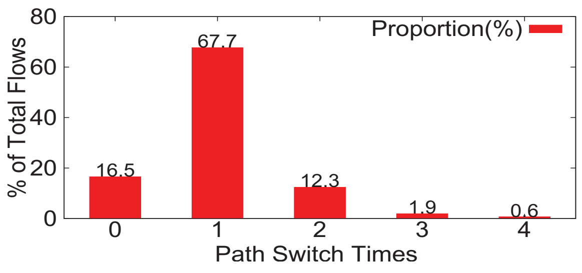


Figure 9: Path Switch Distribution.

workload [17]. More specifically, the probability of consuming traffic from the i-th content provider is $F(i) = a \times i^{-\alpha}$, where $a = 1/\sum_{i=1}^{N} i^{-a}$ and N is the number of content providers, which is set to one million. We fix the deployment ratio to 50% and vary α to understand the performance under varying degrees of traffic skewness. The performance of BGP routing degrades as the skewness grows, which can be explained by the fact that serious congestion is caused by the increasing traffic flowing through a limited number of paths. In contrast, MIFO achieves much higher throughput thanks to multi-path forwarding. MIFO outperforms MIRO as well. Take $\alpha = 1.0$ for instance. With MIFO, 40% of flows get 500 Mbps bandwidth during the transfer, while only 17% and 7% of the flows can achieve this in MIRO and BGP routing, respectively.

C. Load Balancing

It is important to understand how much congestion on inter-AS paths is relieved with the help of MIFO. Fig. 8 illustrates the proportion of traffic offloaded to alternative paths for varying degrees of MIFO deployment. We collect the number of flows transferred on alternative paths and divide it by the total number of flows. With 100% deployment of MIFO, half of the flows are delivered over alternative paths, rather than competing for bandwidth on default paths. It indicates that MIFO utilizes more paths in order to provide more network capacity. Even with a small deployment of MIFO (*e.g.,* 10%), a non-trivial amount of traffic (*e.g.,* 9%) is offloaded from the default paths.

D. Stability

We also evaluated the stability of MIFO. A path switch happens if the border router deflects the traffic from the default path to an alternative path, or resumes using the default path after using an alternative path. It would be unstable if most of the flows have a large number of

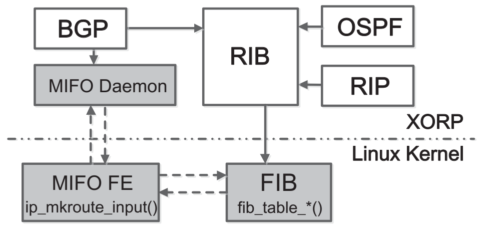


Figure 10: Main modules in Prototype Implementation

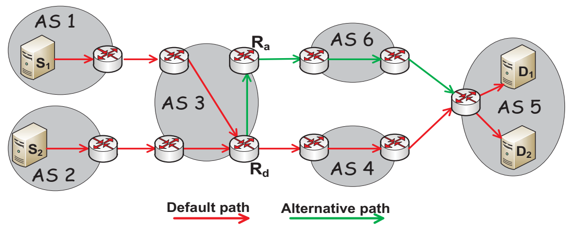
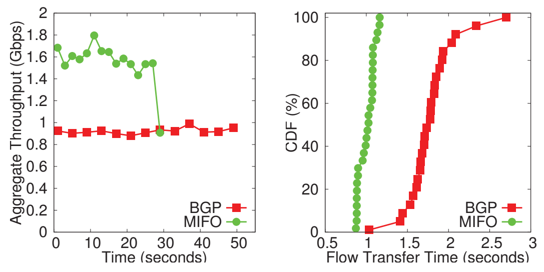


Figure 11: Experiment Topology



(a) Aggregate Throughput (b) Flow Completion Time

Figure 12: Testbed Experimental Result.

switches, meaning traffic all shifts over first and then shifts back. However, as Fig. 9 shown, more than half of flows (67.7%) had a path switch only once and most flows (97.5%) switch paths no more than twice. This indicates that MIFO associated with most of the traffic is reasonably stable.

V. Implementation based Experiments

In this section we report on experiments based on a prototype implementation of MIFO. The experiments were carried out on a testbed, so as to characterize the performance of MIFO in a real system.

A. Prototype Implementation

We have implemented the MIFO Daemon as a module in the eXtensible Open Router Platform (XORP) [18] for the control plane, and developed the MIFO Forwarding Engine (MIFO FE) in Linux kernel space for the data plane. The implementation includes 1500 lines of code in C++ (XORP) and C (kernel).

Fig. 10 shows the implementation architecture. We developed a new module in XORP to implement the MIFO Daemon. It interacts with the BGP module to acquire available alternative paths. Further, it constantly collects available link capacity from the data plane (through MIFO FE) and updates the 'alt' port in the FIB to direct packets to the best alternative path. The MIFO Forwarding Engine is realized as a kernel module. In addition, we re-implement the FIB lookup function, *ip_mkroute_input()*, by adding two callback functions: one is used to monitor the utilization of each link, the other to interact with theMIFO Daemon and update FIB entries. We also add 'alt port' attribute into struct *fib_table* and modify related manipulation functions, such as *fib_table_insert() etc.*

B. Experiment Setup

The testbed comprises 15 desktop machines. 4 of them are end hosts to send and receive flows while the other 11 machines are equipped with the MIFO implementation and function as routers. 6 ASes are constructed by these machines as shown in Fig. 11. All the machines are connected by Gigabit Ethernet links.

30 TCP flows are produced from S_1 to D_1 one after another, as well as from S_2 to D_2. Each flow is 100MB in size and the data packet is set to 1KB. S_1 and S_2 start transferring data at the same time. With BGP routing, (S_1, D_1) and (S_2, D_2) take $1 \rightarrow 3 \rightarrow 4 \rightarrow 5$ and $2 \rightarrow 3 \rightarrow 4 \rightarrow 5$ as default AS path, respectively. Consequently, the link between AS 3 and AS 4 is the bottleneck, with contention between the flows (S_1, D_1) and (S_2, D_2). On the other side, with MIFO, the border router R_d is able to relieve congestion on the default path by leveraging the alternative path $3 \rightarrow 6 \rightarrow 5$ through the iBGP peer, R_a. We run BGP routing as well as MIFO respectively, and observe the completion time for each flow as well as the aggregate throughput of the network.

C. Experimental Results

From Fig. 12(a), we see that MIFO significantly outperforms BGP routing by achieving much higher aggregate throughput. With BGP routing, each flow gets limited link capacity as a result of congestion through R_d. In contrast, by delivering traffic through alternative paths through R_a, a higher total end-end capacity is achievable with MIFO. The aggregate throughput with MIFO is around 1.7Gbps, while it is 0.94Gbps with BGP routing. Thus, MIFO improves the aggregate throughput by 81% compared with BGP routing. Fig. 12(b) illustrates the CDF of the flow completion time. All the flows complete within 1.1s in MIFO, while 80% transferred flows take more than 1.6s with BGP routing. Moreover, completing all the flows take only 30s in MIFO , but takes 51s in BGP routing.

VI. Related Work

Multi-path Interdomain Routing. Several works have been proposed to exploit path diversity by advertising multiple paths in achieving a better interdomain routing.

MIRO [7] enables each pair of ASes to negotiate an alternative path (*e.g.,* bypass an AS with bad reputation). PDAR [8] extends BGP to advertise a most disjoint alternative path along with the best path. However, above solutions either require dedicated communication channel (*e.g.,* MIRO) or produce extra BGP UPDATE messages (*e.g.,* PDAR), which causes additional communication overhead. In our design, MIFO obtains multiple paths with zero overhead by learning alternative paths in local BGP RIB. Moreover, no further verification is needed since every path in BGP RIB is validate. Therefore, the magnitude of path diversity gained in MIFO is relied on the node degree in AS-level Internet topology.

Internet Topological Features. Previous measurements show that the hierarchy of Internet is becoming flatter [3]. The inferred AS-level Internet topology in Nov. 2014 has a large average node degree but a small diameter [16], while the real-time information about backbone routers demonstrates the path richness in BGP RIB [11]. In addition, popular content providers such as Google and Facebook have enormous amounts of peers [19]. All these evidence indicates that numerous paths are existed between each pair of ASes, which benefits MIFO by allowing border routers to forward packets among multiple paths.

Load Adaptive Forwarding. The fundamental goal of MIFO is to optimize the network performance through load adaptive forwarding. Most of past works accomplish this by applying traffic engineering (TE) protocols [20–22] on top of an existing MPLS infrastructure. Performance improvement is achieved by adjusting the distribution of traffic among the paths with the same ingress/egress nodes (*e.g.,* TeXCP [21]). However, these schemes only work on intradomain, with the help of centralized control in local AS. Instead, MIFO works on interdomain level in a fully distributed manner. MIFO enables AS border routers to adaptively forward packets to different paths.

VII. Conclusion

In this paper, we proposed a Multi-path inter-domain forwarding mechanism (MIFO) that enables border routers of each AS to adaptively forward packets on alternative paths instead of being limited to a default path that may be congested. MIFO allows the border routers to efficiently obtain multiple paths from the local BGP RIB, whereby different paths are learned from different neighbors with zero overhead comparing with previous methodologies. The key contribution of MIFO is to solve the challenges faced in realizing multi-path forwarding in practice. First, MIFO avoids loops on the data plane by subtly adopting "valley-free" rule in the forwarding process. In addition to the proof of correctness, we show that one more bit in the forwarding policy is enough to realize MIFO in a real system. Second, the issue of cycling between iBGP peers is addressed by means of IP-in-IP encapsulation. Finally, MIFO provides an efficient greedy method in selecting the best alternative path. Our evaluation shows that MIFO can significantly improve end-to-end throughput compared to conventional BGP routing and other multi-path routing methods (*e.g.,* MIRO). We also implemented the prototype of MIFO on Linux with the XORP router platform. The experimental results also show that MIFO notably improves end-to-end throughput.

References

[1] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, "Internet inter-domain traffic," in *SIGCOMM*, 2010.

[2] "Amsterdam Internet IXP." https://www.ams-ix.net/technical/statistics, 2014.

[3] P. Gill, M. Schapira, and S. Goldberg, "Let the market drive deployment: A strategy for transitioning to BGP security," in *SIGCOMM*, 2011.

[4] H. Jiang and C. Dovrolis, "Why is the Internet traffic bursty in short time scales?," in *SIGMETRICS*, pp. 241–252, 2005.

[5] "INTERNET INTERDOMAIN CONGESTION." http://www.caida.org/publications/presentations/2014/bitag-congestion/bitag-congestion.pdf.

[6] D. Clark, S. Bauer, k. claffy, A. Dhamdhere, B. Huffaker, W. Lehr, and M. Luckie, "Measurement and Analysis of Internet Interconnection and Congestion," in *Telecommunications Policy Research Conference (TPRC)*, 2014.

[7] W. Xu and J. Rexford, "MIRO: multi-path interdomain routing," in *SIGCOMM*, 2006.

[8] F. Wang and L. Gao, "Path diversity aware interdomain routing.," in *INFOCOM*, IEEE, 2009.

[9] Y. Liao and L. Gao, etc., "Reliable interdomain routing through multiple complementary routing processes.," in *CoNEXT*, 2008.

[10] N. Kushman, S. Kandula, D. Katabi, and B. Maggs, "R-BGP: Staying connected in a connected world," in *NSDI*, 2007.

[11] University of Oregon, "Routeviews." http://www.routeviews.org/.

[12] L. Gao and J. Rexford, "Stable Internet routing without global coordination," *IEEE/ACM Trans. Network.*, 2001.

[13] X. Xiao, A. Hannan, and B. Bailey, "Traffic engineering with mpls in the internet," *IEEE Network Magazine*, 2000.

[14] E. Goldoni and M. Schivi, "End-to-end available bandwidth estimation tools, an experimental comparison.," in *PAM*, 2010.

[15] "BGP Table Size in 2014." http://bgp.potaroo.net/, 2014.

[16] "AS Topology." http://irl.cs.ucla.edu/topology/, 2014.

[17] L. A. Adamic and B. A. Huberman, "Zipf's law and the internet," *Glottometrics*, vol. 3, pp. 143–150, 2002.

[18] "The eXtensible Open Router Platform." http://www.xorp.org/.

[19] Y. Shavitt and U. Weinsberg, "Topological trends of Internet content providers," *CoRR*, 2012.

[20] A. Elwalid, C. Jin, S. H. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering.," in *INFOCOM*, 2001.

[21] S. Kandula, D. Katabi, B. S. Davie, and A. Charny, "Walking the tightrope: responsive yet stable traffic engineering.," in *SIGCOMM*, 2005.

[22] "Traffic engineering extensions to ospf version 3." http://tools.ietf.org/html/rfc5329, 2008.