# Making Routers Last Longer with ViAggre

Hitesh Ballani*, Paul Francis*, Tuan Cao* and Jia Wang†
*Cornell University †AT&T Labs – Research

***Abstract*— This paper presents ViAggre (Virtual Aggregation), a "configuration-only" approach to shrinking the routing table on routers. ViAggre does not require any changes to router software and routing protocols and can be deployed independently and autonomously by any ISP. ViAggre is effectively a scalability technique that allows an ISP to modify its internal routing such that individual routers in the ISP's network only maintain a part of the global routing table.**

**We evaluate the application of ViAggre to a few tier-1 and tier-2 ISPs and show that it can reduce the routing table on routers by an order of magnitude while imposing almost no traffic stretch and negligible load increase across the routers. We also deploy Virtual Aggregation on a testbed comprising of Cisco routers and benchmark this deployment. Finally, to understand and address concerns regarding the configuration overhead that our proposal entails, we implement a configuration tool that automates ViAggre configuration. While it remains to be seen whether most, if not all, of the management concerns can be eliminated through such automated tools, we believe that the simplicity of the proposal and its possible short-term impact on routing scalability suggest that it is an alternative worth considering.**

## I. Introduction

The Internet default-free zone (DFZ) routing table has been growing at a rapid rate for the past few years [21]. Looking ahead, there are concerns that as the IPv4 address space runs out, hierarchical aggregation of network prefixes will further deteriorate resulting in a substantial acceleration in the growth of the routing table [31]. A growing IPv6 deployment would worsen the situation even more [29].

The increase in the size of the DFZ routing table has several harmful implications for inter-domain routing.[1] [31] discusses these in detail. At a technical level, increasing routing table size may drive high-end router design into various engineering limits. For instance, while memory and processing speeds might just scale with a growing routing system, power and heat dissipation capabilities may not [30]. On the business side, the performance requirements for forwarding while being able to access a large routing table imply that the cost of forwarding packets increases and hence, networks become less cost-effective [27]. Further, it makes provisioning of networks harder since it is difficult to estimate the usable lifetime of routers, not to mention the cost of the actual upgrades. As a matter of fact, instead of upgrading their routers, a few ISPs have resorted to filtering out some small prefixes (mostly /24s) which implies that parts of the Internet may not have reachability to each other [20]. A recent proprietary conversation with a major Internet ISP revealed that in order to avoid router memory upgrades, the ISP is using a trick that reduces memory requirements but breaks BGP loop-detection and hence, would wreak havoc if adopted by other ISPs too. These anecdotes suggest that ISPs are willing to undergo some pain to avoid the cost of router upgrades.

Such concerns regarding FIB size growth, along with problems arising from a large RIB and the concomitant convergence issues, were part of the reasons that led a recent Internet Architecture Board workshop to conclude that scaling the routing system is one of the most critical challenges of near-term Internet design [30]. The severity of these problems has also prompted a slew of routing proposals [7,8,11,15,19,29,32,40]. All these proposals require changes in the routing and addressing architecture of the Internet. This, we believe, is the nature of the beast since some of the fundamental Internet design choices limit routing scalability; the overloading of IP addresses with "who" and "where" semantics represents a good example [30]. However, the very fact that they require architectural change has contributed to the non-deployment of these proposals.

This paper takes the position that a major architectural change is unlikely and it may be more pragmatic to approach the problem through a series of incremental, individually cost-effective upgrades. Guided by this and the aforementioned implications of a rapidly growing DFZ FIB, this paper proposes *Virtual Aggregation or ViAggre*, a scalability technique that focuses primarily on shrinking the FIB size on routers. ViAggre is a "configuration-only" solution that *applies to legacy routers*. Further, ViAggre can be *adopted independently and autonomously by any ISP* and hence the bar for its deployment is much lower. The key idea behind ViAggre is very simple: an ISP adopting ViAggre divides the responsibility for maintaining the global routing table amongst its routers such that individual routers only maintain a part of the routing table. Thus, this paper makes the following contributions:

- We discuss two deployment options through which an ISP can adopt ViAggre. The first one uses *FIB suppression* to shrink the FIB of all the ISP's routers while the second uses *route filtering* to shrink both the

[1]Hereon, we follow the terminology used in [39] and use the term "routing table" to refer to the Forwarding Information Base or FIB, commonly also known as the forwarding table. The Routing Information Base is explicitly referred to as the RIB.

FIB and RIB on all *data-path* routers.

- We analyze the application of ViAggre to an actual tier-1 ISP and several inferred (Rocketfuel [37]) ISP topologies. We find that ViAggre can reduce FIB size by more than an order of magnitude with negligible stretch on the ISP's traffic and very little increase in load across the ISP's routers. Based on predictions of future routing table growth, we estimate that ViAggre can be used to extend the life of already outdated routers by more than 10 years.
- We propose utilizing the notion of prefix popularity to reduce the impact of ViAggre on the ISP's traffic and use a two-month study of a tier-1 ISP's traffic to show the feasibility of such an approach.
- As a proof-of-concept, we configure test topologies comprising of Cisco routers (on WAIL [3]) according to the ViAggre proposal. We use the deployment to benchmark the control-plane processing overhead that ViAggre entails. For one of the presented designs, the overhead is minimal and hence, network properties such as convergence times are not affected. The other design has high overhead due to implementation issues and needs more experimentation.
- ViAggre involves the ISP reconfiguring its routers which can be a deterrent to adoption. We quantify this configuration overhead. We also implement a configuration tool that, given the ISPs existing configuration files, can automatically generate the configuration files needed for ViAggre deployment. We discuss the use of this tool on our testbed.

Overall, the incremental version of ViAggre that this paper presents can be seen as little more than a simple and structured hack that assimilates ideas from existing work including, but not limited to, VPN tunnels and CRIO [40]. We believe that its very simplicity makes ViAggre an attractive short-term solution that provides ISPs with an alternative to upgrading routers in order to cope with routing table growth till more fundamental, long-term architectural changes can be agreed upon and deployed in the Internet. However, the basic ViAggre idea can also be applied in a clean-slate fashion to address routing concerns beyond FIB growth. While we defer the design and the implications of such a non-incremental ViAggre architecture for future work, the notion that the same concept has potential both as an immediate alleviative and as the basis for a next-generation routing architecture seems interesting and worth exploring.

## II. ViAggre design

*ViAggre* allows individual ISPs in the Internet's DFZ to do away with the need for their routers to maintain routes for all prefixes in the global routing table. An ISP adopting ViAggre divides the global address space into a set of *virtual prefixes* such that the virtual prefixes are larger than any aggregatable (real) prefix in use today. So, for instance, an ISP could divide the IPv4 address space into 128 parts with a /7 virtual prefix representing each part (0.0.0.0/7 to 254.0.0.0/7). Note that such a naïve allocation would yield an uneven distribution of real prefixes across the virtual prefixes. However, the virtual prefixes need not be of the same length and hence, the ISP can choose them such that they contain a comparable number of real prefixes.

The virtual prefixes are not topologically valid aggregates, i.e. there is not a single point in the Internet topology that can hierarchically aggregate the encompassed prefixes. ViAggre makes the virtual prefixes aggregatable by organizing *virtual networks*, one for each virtual prefix. In other words, a virtual topology is configured that causes the virtual prefixes to be aggregatable, thus allowing for routing hierarchy that shrinks the routing table. To create such a virtual network, some of the ISP's routers are assigned to be within the virtual network. These routers maintain routes for all prefixes in the virtual prefix corresponding to the virtual network and hence, are said to be *aggregation points* for the virtual prefix. A router can be an aggregation point for multiple virtual prefixes and is required to only maintain routes for prefixes in the virtual prefixes it is aggregating.

Given this, a packet entering the ISP's network is routed to a close-by aggregation point for the virtual prefix encompassing the actual destination prefix. This aggregation point has a route for the destination prefix and forwards the packet out of the ISP's network in a tunnel. In figure 1 (figure details explained later), router C is an aggregation point for the virtual prefix encompassing the destination prefix and B $\rightarrow$ C $\rightarrow$ D is one such path through the ISP's network.

### A. Design Goals

The discussion above describes ViAggre at a conceptual level. While the design space for organizing an ISP's network into virtual networks has several dimensions, this paper aims for deployability and hence is guided by two major design goals:

1) *No changes to router software and routing protocols*: The ISP should not need to deploy new data-plane or control-plane mechanisms.
2) *Transparent to external networks*: An ISP's decision to adopt the ViAggre proposal should not impact its interaction with its neighbors (customers, peers and providers).

These goals, in turn, limit what can be achieved through the ViAggre designs presented here. Routers today have a Routing Information Base (RIB) generated by the routing protocols and a Forwarding Information Base (FIB) that is used for forwarding the packets.

Consequently, the FIB is optimized for looking up destination addresses and is maintained on fast(er) memory, generally on the line cards themselves [31]. All things being equal, it would be nice to shrink both the RIB and the FIB for all ISP devices, as well as make other improvements such as speed up convergence time.

While the basic ViAggre idea can be used to achieve these benefits (section VI), we have not been able to reconcile them with the aforementioned design goals. Instead, this paper is based on the hypothesis that given the performance and monetary implications of the FIB size for routers, an immediately deployable solution that reduces FIB size is useful. Actually, one of the presented designs also shrinks the RIB on routers; only components that are off the data path (i.e. route reflectors) need to maintain the full RIB.

### *B. Design-I: FIB Supression*

This section details one way an ISP can deploy virtual prefix based routing while satisfying the goals specified in the previous section. The discussion below applies to IPv4 (and BGPv4) although the techniques detailed here work equally well for IPv6. The key concept behind this design is to operate the ISP's internal distribution of BGP routes untouched and in particular, to populate the RIB on routers with the full routing table but to suppress most prefixes from being loaded in the FIB of routers. A standard feature on routers today is *FIB Suppression* which can be used to prevent routes for individual prefixes in the RIB from being loaded into the FIB. We have verified support for FIB suppression as part of our ViAggre deployment on Cisco 7300 and 12000 routers. Documentation for Juniper [44] and Foundry [43] routers specify this feature too. We use this as described below.

The ISP does not modify its routing setup – the ISP's routers participate in an intra-domain routing protocol that establishes internal routes through which the routers can reach each other while BGP is used for inter-domain routing just as today. For each virtual prefix, the ISP designates some number of routers to serve as aggregation points for the prefix and hence, form a virtual network. Each router is configured to only load prefixes belonging to the virtual prefixes it is aggregating into its FIB while suppressing all other prefixes.

Given this, the ISP needs to ensure that packets to any prefix can flow through the network in spite of the fact that only a few routers have a route to the prefix. This is achieved as follows:

– *Connecting Virtual Networks.* Aggregation points for a virtual prefix originate a route to the virtual prefix that is distributed throughout the ISP's network but not outside. Specifically, an aggregation point advertises the virtual prefix to its iBGP peers. A router that is not an aggregation point for the virtual prefix would choose the route advertised by the aggregation point closest to it and hence, forward packets destined to any prefix in the virtual prefix to this aggregation point.[2]

– *Sending packets to external routers.* When a router receives a packet destined to a prefix in a virtual prefix it is aggregating, it can look up its FIB to determine the route for the packet. However, such a packet cannot be forwarded in the normal hop-by-hop fashion since a router that is not an aggregation point for the virtual prefix in question might forward the packet back to the aggregation point, resulting in a loop. Hence, the packet must be tunneled from the aggregation point to the external router that was selected as the BGP NEXT_HOP. While the ISP can probably choose from many tunneling technologies, we use of MPLS Label Switched Paths (LSPs) for such tunnels. This choice was influenced by the fact that MPLS is widely supported in routers, is used by ISPs, and operates at wire speed. Further, protocols like LDP [1] automate the establishment of MPLS tunnels and hence, reduce the configuration overhead.

However, a LSP from the aggregation point to an external router would require cooperation from the neighboring ISP. To avoid this, every edge router of the ISP initiates a LSP for every external router it is connected to. Thus, all the ISP routers need to maintain LSP mappings equal to the number of external routers connected to the ISP, a number much smaller than the routes in the DFZ routing table. Note that even though the tunnel endpoint is the external router, the edge router can be configured to strip the MPLS label from the data packets before forwarding them onto the external router. This, in turn, has two implications. First, external routers don't need to be aware of the adoption of ViAggre by the ISP. Second, even the edge router does not need a FIB entry for the destination prefix, instead it chooses the external router to forward the packets to based on the MPLS label of the packet. The behavior of the edge router here is similar to the penultimate hop in a VPN scenario and is achieved through standard configuration.

We now use a concrete example to illustrate the flow of packets through an ISP network that is using ViAggre. Figure 1 shows the relevant routers. The ISP is using /7s as virtual prefixes and router C is an aggregation point for one such virtual prefix 4.0.0.0/7. Edge router D initiates a LSP to external router E with label *l* and hence, the ISP's routers can get to E through MPLS tunneling. The figure shows the path of a packet destined to prefix 4.0.0.0/24, which is encompassed by 4.0.0.0/7, through

[2]All other attributes for the routes to a virtual prefix are the same and hence, the decision is based on the IGP metric to the aggregation points. Hence, "closest" means closest in terms of IGP metric.

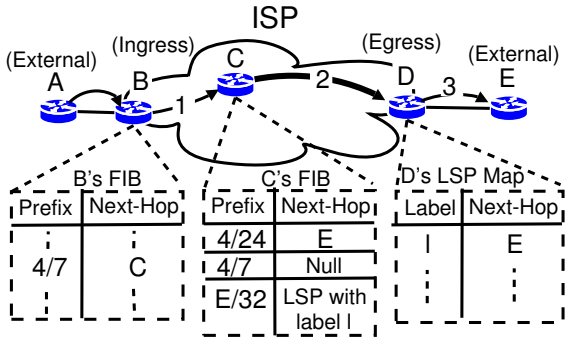



Fig. 1. Path of packets destined to prefix 4.0.0.0/24 (or, 4/24) between external routers A and E through an ISP with ViAggre. Router C is an aggregation point for virtual prefix 4.0.0.0/7 (or, 4/7).

the ISP's network. The path from the ingress router B to the external router E comprises of three segments:

1) VP-routed: Ingress router B is not an aggregation point for 4.0.0.0/7 and hence, forwards the packet to aggregation point C.
2) MPLS-LSP: Router C, being an aggregation point for 4.0.0.0/7, has a route for 4.0.0.0/24 with BGP NEXT_HOP set to E. Further, the path to router E involves tunneling the packet with MPLS label *l*.
3) Map-routed: On receiving the tunneled packet from router C, egress router D looks up its MPLS label map, strips the MPLS header and forwards the packet to external router E.

### C. Design-II: Route Reflectors

The second design offloads the task of maintaining the full RIB to devices that are off the data path. Many ISPs use route-reflectors for scalable internal distribution of BGP prefixes and we require only these route-reflectors to maintain the full RIB. For ease of exposition, we assume that the ISP is already using per-PoP route reflectors that are off the data path, a common deployment model for ISPs using route reflectors.

In the proposed design, the external routers connected to a PoP are made to peer with the PoP's route-reflector.[3] This is necessary since the external peer may be advertising the entire DFZ routing table and we don't want all these routes to reside on any given data-plane router. The route-reflector also has iBGP peerings with other route-reflectors and with the routers in its PoP. Egress filters are used on the route-reflector's peerings with the PoP's routers to ensure that a router only gets routes for the prefixes it is aggregating. This shrinks both the RIB and the FIB on the routers. The data-plane operation and hence, the path of packets through the ISP's network remains the same as with the previous design.

With this design, a PoP's route-reflector peers with all the external routers connected to the PoP. The RIB size on a BGP router depends on the number of peers it has and hence, the RIB for the route-reflectors can potentially be very large. If needed, the RIB requirements can be scaled by using multiple route-reflectors. Note that the RIB scaling properties here are better than in the status quo. Today, edge routers have no choice but to peer with the directly connected external routers and maintain the resulting RIB. Replicating these routers is prohibitive because of their cost but the same does not apply to off-path route-reflectors, which could even be BGP software routers.

[3]Note that these will be eBGP multihop peerings since the route-reflector is not directly connected to the external routers.

### D. Design Comparison

As far as the configuration is concerned, configuring suppression of routes on individual routers in design-I is comparable, at least in terms of complexity, to configuring egress filters on the route-reflectors. In both cases, the configuration can be achieved through BGP route-filtering mechanisms (access-lists, prefix-lists, etc.).

Design-II, apart from shrinking the RIB on the routers, does not require the route suppression feature on routers. Further, as we detail in section V-B, the specific filtering mechanism that we use for FIB suppression on the routers in our deployment leads to high CPU utilization at the peering establishment time and hence, requires more experimentation. However, design-II does require the ISP's eBGP peerings to be reconfigured which, while straightforward, violates our goal of not impacting neighboring ISPs.

### E. Network Robustness

ViAggre causes packets to be routed through an aggregation point which leads to robustness concerns. When an aggregation point for a virtual prefix fails, routers using that aggregation point are re-routed to another aggregation point through existing mechanisms without any explicit configuration by the ISP. In case of design-I, a router has routes to all aggregation points for a given virtual prefix in its RIB and hence, when the aggregation point being used fails, the router installs the second closest aggregation point into its FIB and packets are re-routed almost instantly. With design-II, it is the route-reflector that chooses the alternate aggregation point and advertises this to the routers in its PoP. Hence, as long as another aggregation point exists, failover happens automatically and at a fast rate.

### F. Routing popular prefixes natively

The use of aggregation points implies that packets in ViAggre may take paths that are longer than native paths. Apart from the increased path length, the packets incur queuing delay at all the extra hops. The extra hops also result in an increase in load on the ISP's routers and links and a modification in the distribution of traffic across them

Past studies have shown that a large majority of Internet traffic is destined to a very small fraction of prefixes [10,13,34,38]. The fact that routers today have no choice but to maintain the complete DFZ routing table implies that this observation wasn't very useful for routing configuration. However, with ViAggre, individual routers only need to maintain routes for a fraction of prefixes. The ISP can thus configure its ViAggre setup such that the small fraction of popular prefixes are in the FIB of every router and hence, are routed natively. For design-I, this involves configuring each router with a set of popular prefixes that should not be suppressed from the FIB. For design-II, a PoP's route-reflector can be configured to not filter advertisements for popular prefixes from the PoP's routers. Beyond this, the ISP may also choose to install customer prefixes into its routers such that they don't incur any stretch. The rest of the proposal involving virtual prefixes remains the same and ensures that individual routers only maintain routes for a fraction of the unpopular prefixes. In section IV-B.4, we analyze Netflow data from a tier-1 ISP network to show that not only such an approach is feasible, it also addresses all the concerns raised above.

## III. Allocating aggregation points

An ISP adopting ViAggre would obviously like to minimise the stretch imposed on its traffic. Ideally, an ISP would deploy an aggregation point for all virtual prefixes in each of its PoPs. This would ensure that for every virtual prefix, a router chooses the aggregation point in the same PoP and hence, the traffic stretch is minimal. However, this is often not possible in practice. This is because ISPs, including tier-1 ISPs, often have some small PoPs with just a few routers and therefore there may not be enough cumulative FIB space in the PoP to hold all the actual prefixes. Hence, the ISP needs to be smart about the way it designates routers to aggregate virtual prefixes and in this section we explore this choice.

### A. Problem Formulation

We first introduce the notation used in the rest of this section. Let $T$ represent the set of prefixes in the Internet routing table, $R$ be the set of ISP's routers and $X$ is the set of external routers directly connected to the ISP. For each $r \in R$, $P_r$ represents the set of popular prefixes for router $r$. $V$ is the set of virtual prefixes chosen by the ISP and for each $v \in V$, $n_v$ is the number of prefixes in $v$. We use two matrices, $D = (d_{i,j})$ that gives the distance between routers $i$ and $j$ and $W = (w_{i,j})$ that gives the IGP metric for the IGP-established path between routers $i$ and $j$. We also define two relations:

– "BelongsTo" relation $B$: $T \rightarrow V$ such that $B(p)=v$ if prefix $p$ belongs to or is encompassed by virtual prefix $v$.

– "Egress" relation $E$: $R$ x $T \rightarrow R$ such that $E(i, p)=j$ if traffic to prefix $p$ from router $i$ egresses at router $j$.

The mapping relation $A$: $R \rightarrow 2^V$ captures how the ISP assigns aggregation points; i.e. $A(r) = \{v_1 \dots v_n\}$ implies that router $r$ aggregates virtual prefixes $\{v_1 \dots v_n\}$. Given this assignment, we can determine the aggregation point any router uses for its traffic to each virtual prefix. This is captured by the "Use" relation $U$: $R$ x $V \rightarrow R$ where $U(i, v) = j$ or router $i$ uses aggregation point $j$ for virtual prefix $v$ if the following conditions are satisfied:

$$\begin{array}{lrcll} 1) & v & \in & A(j) & \\ 2) & w_{i,j} & \leq & w_{i,k} & \forall k \in R, v \in A(k) \end{array}$$

Here, condition 1) ensures that router $j$ is an aggregation point for virtual prefix $v$. Condition 2) captures the operation of BGP with design-I and ensures that a router chooses the aggregation point that is closest in terms of IGP metrics.[4]

Using this notation, we can express the FIB size on routers and the stretch imposed on traffic.

*1) Routing State:* In ViAggre, a router needs to maintain routes to the (real) prefixes in the virtual prefixes it is aggregating, routes to all the virtual prefixes themselves and routes to the popular prefixes. Further, the router needs to maintain LSP mappings for LSPs originated by the ISP's edge routers with one entry for each external router connected to the ISP. Hence, the "routing state" for the router $r$, hereon simply referred to as the FIB size ($F_r$), is given by:

$$F_r = \sum_{v \in A(r)} n_v + |V| + |P_r| + |X|$$

The **Worst FIB size** and the **Average FIB size** are defined as follows:

$$\begin{array}{rcl} \text{Worst FIB size} & = & \max_{r \in R}(F_r) \\ \text{Average FIB size} & = & \sum_{r \in R}(F_r)/|R| \end{array}$$

*2) Traffic Stretch:* If router $i$ uses router $k$ as an aggregation point for virtual prefix $v$, packets from router $i$ to a prefix $p$ belonging to $v$ are routed through router $k$. Hence, the stretch ($S$) imposed on traffic to prefix $p$ from router $i$ is given by:

$$\begin{array}{rcll} S_{i,p} & = & 0, & p \in P_i \\ & = & (d_{i,k} + d_{k,j} - d_{i,j}), & p \in (T - P_i),\ v = B(p) \\ & & & k = U(i, v)\ \&\ j = E(k, p) \end{array}$$

The **Worst Stretch** and **Average Stretch** are defined as follows:

$$\begin{array}{rcl} \text{Worst Stretch} & = & \max_{i \in R, p \in T}(S_{i,p}) \\ \text{Average Stretch} & = & \sum_{i \in R, p \in T}(S_{i,p})/(|R| * |T|) \end{array}$$

[4]With design-II, a router chooses the aggregation point closest to the router's route-reflector in terms of IGP metrics and so a similar formulation works for the second design too.

**Problem:** ViAggre shrinks the routing table on routers by ensuring that individual routers only maintain routes to a fraction of the prefixes and forward packets to an aggregation point for the rest. Thus, through the use of aggregation points, ViAggre trades off an increase in path length for a reduction in routing state. The ISP can use the assignment of aggregation points as a knob to tune this trade-off. Here we consider the simple goal of minimising the FIB Size on the ISP's routers while bounding the stretch. Specifically, the ISP needs to assign aggregation points by determining a mapping *A* that

$$\begin{array}{ll} \min & \text{Worst FIB Size} \\ \text{s.t.} & \text{Worst Stretch} \leq C \end{array}$$

where *C* is the specified constraint on Worst Stretch. Note that much more complex formulations are possible. Our focus on worst-case metrics is guided by practical concerns – the Worst FIB Size dictates how the ISP's routers need to be provisioned while the Worst Stretch characterizes the most unfavorable impact of the use of ViAggre. Specifically, bounding the Worst Stretch allows the ISP to ensure that its existing SLAs are not breached and applications sensitive to increase in latency (example, VOIP) are not adversely affected.

### *B. A Greedy Solution*

The problem of assigning aggregation points while satisfying the conditions above can be mapped to the MultiCommodity Facility Location (MCL) problem [33]. Using the MCL terminology, this involves routers representing facilities, virtual prefixes being commodities and each router's traffic to virtual prefixes serving as clients. MCL is NP-hard and [33] presents a logarithmic approximation algorithm for it. Here we discuss a greedy approximation solution to the problem.

The first solution step is to determine that if router $i$ were to aggregate virtual prefix $v$, which routers can it serve without violating the stretch constraint. This is the $can\_serve_{i,v}$ set and is defined as follows:

$$\begin{aligned} can\_serve_{i,v} \quad = \quad & \{j \mid j \in R, (\forall p \in T, B(p) = v, E(i,p) \\ & = k, (d_{j,i} + d_{i,k} - d_{j,k}) \leq C)\} \end{aligned}$$

Given this, the key idea behind the solution is that any assignment based on the $can\_serve$ relation will have Worst Stretch less than $C$. Hence, our algorithm designates routers to aggregate virtual prefixes in accordance with the $can\_serve$ relation while greedily trying to minimise the Worst FIB Size. The algorithm, shown below, stops when each router can be served by at least one aggregation point for each virtual prefix.

$Worst\_FIB\_Size$=0
**for all** r in R **do**
  **for all** v in V **do**
    Calculate $can\_serve_{r,v}$
  Sort V in decreasing order of $n_v$
**for all** v in V **do**
  Sort R in decreasing order of $|can\_serve_{r,v}|$
  **repeat**
    **for all** r in R **do**
      **if** $(F_r + n_v) \leq Worst\_FIB\_Size$ **then**
        A[r]=A[r] $\cup$ v  # Assign v to r
        $F_r = F_r + n_v$  # r's FIB size increases
        Mark all routers in $can\_serve_{r,v}$ as served
      **if** All routers are served for v **then**
        break
    **if** All routers are not served for v **then**
      # $Worst\_FIB\_Size$ needs to be raised
      **for all** r in R **do**
        **if** v $\notin$ A[r] **then**
          # r is not an aggregation point for v
          A[r]=A[r] $\cup$ v
          $F_r = F_r + n_v$
          $Worst\_FIB\_Size = F_r$
          break
  **until** All Routers are served for virtual prefix v

## IV. Evaluation

In this section we evaluate the application of ViAggre to a few Internet ISPs.

### *A. Metrics of Interest*

We defined (**Average** and **Worst**) **FIB Size** and **Stretch** metrics in section III-A. Here we define other metrics that we use for ViAggre evaluation.

*1) Impact on Traffic:* Apart from the stretch imposed, another aspect of ViAggre's impact is the amount of traffic affected. To account for this, we define **traffic impacted** as the fraction of the ISP's traffic that uses a different router-level path than the native path. Note that in many cases, a router will use an aggregation point for the destination virtual prefix in the same PoP and hence, the packets will follow the same PoP-level path as before. Thus, another metric of interest is the **traffic stretched**, the fraction of traffic that is forwarded along a different PoP-level path than before. In effect, this represents the change in the distribution of traffic across the ISP's inter-PoP links and hence, captures how ViAggre interferes with the ISP's inter-PoP traffic engineering.

*2) Impact on Router Load:* The extra hops traversed by traffic increases the traffic load on the ISP's routers. We define the **load increase** across a router as the extra traffic it needs to forward due to ViAggre, as a fraction of the traffic it forwards natively.

### *B. Tier-1 ISP Study*

We analysed the application of ViAggre to a large tier-1 ISP in the Internet. For our study, we obtained the ISP's router-level topology (to determine router set *R*) and the routing tables of routers (to determine prefix

set $T$ and the Egress $E$ and BelongsTo $B$ relations). We used information about the geographical locations of the routers to determine the Distance matrix $D$ such that $d_{i,j}$ is 0 if routers $i$ and $j$ belong to the same PoP (and hence, are in the same city) else $d_{i,j}$ is set to the propagation latency corresponding to the great circle distance between $i$ and $j$. Further, we did not have information about the ISP's link weights. However, guided by the fact that intra-domain traffic engineering is typically latency-driven [36], we use the Distance matrix $D$ as the Weight matrix $W$. We also obtained the ISP's traffic matrix; however, in order to characterise the impact of vanilla ViAggre, the first part of this section assumes that the ISP does not consider any prefixes as popular.

*1) Deployment decisions:* The ISP, in order to adopt ViAggre, needs to decide what virtual prefixes to use and which routers aggregate these virtual prefixes. We describe the approaches we evaluated.

– *Determining set V*. The most straightforward way to select virtual prefixes while satisfying the two conditions specified in section II is to choose large prefixes (/6s, /7s, etc.) as virtual prefixes. We assume that the ISP uses /7s as its virtual prefixes and refer to this as the "/7 allocation".

However, such selection of virtual prefixes could lead to a skewed distribution of (real) prefixes across them with some virtual prefixes containing a large number of prefixes. For instance, using /7s as virtual prefixes implies that the largest virtual prefix (202.0.0.0/7) contains 22,772 of the prefixes in today's BGP routing table or 8.9% of the routing table. Since at least one ISP router needs to aggregate each virtual prefix, such large virtual prefixes would inhibit the ISP's ability to reduce the Worst FIB size on its routers. However, as we mentioned earlier, the virtual prefixes need not be of the same length and so large virtual prefixes can be split to yield smaller virtual prefixes. To study the effectiveness of this approach, we started with /7s as virtual prefixes and split each of them such that the resulting virtual prefixes were still larger than any prefix in the Internet routing table. This yielded 1024 virtual prefixes with the largest containing 4,551 prefixes or 1.78% of the BGP routing table. We also use this virtual prefix allocation for our evaluation and refer to it as "Uniform Allocation".

– *Determining mapping A*. We implemented the algorithm described in section III-B and use it to designate routers to aggregate virtual prefixes.

*2) Router FIB:* We first look at the size and the composition of the FIB on the ISP's routers with a ViAggre deployment. Specifically, we focus on the router with the largest FIB for a deployment where the worst-case stretch ($C$) is constrained to 4ms. The first two bars in figure 2 show the FIB composition for a /7 and uniform allocation respectively. With a /7 allocation, the router's FIB contains 46,543 entries which represents 18.2% of the routing table today. This includes 22,772 prefixes, 128 virtual prefixes, 23,643 LSP mappings and 0 popular prefixes. As can be seen, in both cases, the LSP mappings for tunnels to the external routers contribute significantly to the FIB. This is because the ISP has a large number of customer routers that it has peerings with.

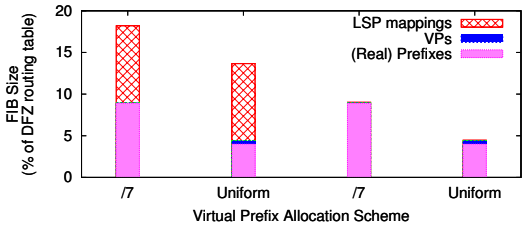
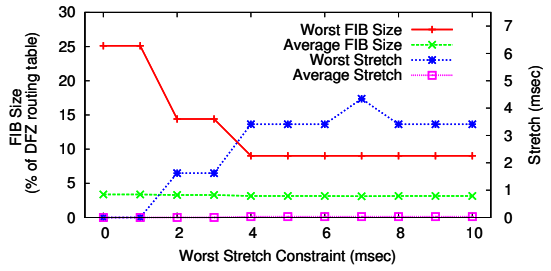


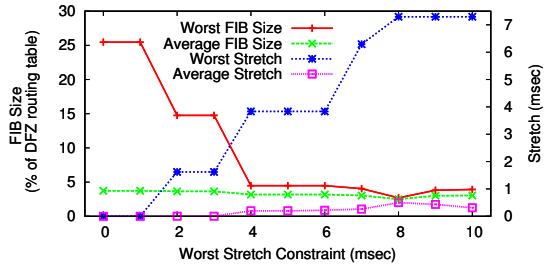

Fig. 2. FIB composition for the router with the largest FIB, $C$=4ms and no popular prefixes.

However, we also note that customer ISPs do not advertise the full routing table to their provider. Hence, edge routers of the ISP could maintain routes advertised by customer routers in their FIB, advertise these routes onwards with themselves as the BGP NEXT_HOP and only initiate LSP advertisements for themselves and for peer and provider routers connected to them. With such a scheme, the number of LSP mappings that the ISP's routers need to maintain and the MPLS overhead in general reduces significantly. The latter set of bars in figure 2 shows the FIB composition with such a deployment for the router with the largest FIB. For the /7 allocation, the Worst FIB size is 23,101 entries (9.02% of today's routing table) while for the Uniform allocation, it is 10,226 entries (4.47%). In the rest of this section, we assume this model of deployment.

*3) Stretch Vs. FIB Size:* We ran the assignment algorithm with Worst Stretch Constraint ($C$) ranging from 0 to 10 ms and determined the (Average and Worst) Stretch and FIB Size of the resulting ViAggre deployment. Figure 3(a) plots these metrics for the /7 allocation. The Worst FIB size, shown as a fraction of the DFZ routing table size today, expectedly reduces as the constraint on Worst Stretch is relaxed. However, beyond $C$=4ms, the Worst FIB Size remains constant. This is because the largest virtual prefix with a /7 allocation encompasses 8.9% of the DFZ routing table and the Worst FIB Size cannot be any less than 9.02% (0.12% overhead is due to virtual prefixes and LSP mappings). Figure 3(b) plots the same metrics for the Uniform allocation and shows that the FIB can be shrunk even more. The figure also shows that the Average FIB Size and the Average stretch are expectedly small throughout. The anomaly beyond $C$=8msec in figure 3(b) results from the fact that our assignment algorithm is an approximation that can yield non-optimal results.




(a) With /7 allocation




(b) With Uniform allocation

Fig. 3. Variation of FIB Size and Stretch with Worst Stretch constraint and no popular prefixes.

| | | Today | ViAggre | | | |
|---|---|---|---|---|---|---|
| | Worst Stretch (ms) | – | 0 | 2 | 4 | 8 |
| 239K FIB | Quad. Fit | Expired | 2015 | 2020 | 2039 | 2051 |
| | Expo. Fit | Expired | 2018 | 2022 | 2031 | 2035 |
| 1M FIB | Quad. Fit | 2015 | 2033 | 2044 | 2081 | 2106 |
| | Expo. Fit | 2018 | 2029 | 2033 | 2042 | 2046 |

TABLE I

ESTIMATES FOR ROUTER LIFE WITH VIAGGRE

Another way to quantify the benefits of ViAggre is to determine the extension in the life of a router with a specified memory due to the use of ViAggre. As proposed in [22], we used data for the DFZ routing table size from Jan'02 to Dec'07 [21] to fit a quadratic model to routing table growth. Further, it has been claimed that the DFZ routing table has seen exponential growth at the rate of 1.3x every two years for the past few years and will continue to do so [30]. We use these models to extrapolate future DFZ routing table size. We consider two router families: Cisco's Cat6500 series with a supervisor 720-3B forwarding engine that can hold upto 239K IPv4 FIB entries and hence, was supposed to be phased out by mid-2007 [6], though some ISPs still continue to use them. We also consider Cisco's current generation of routers with a supervisor 720-3BXL engine that can hold 1M IPv4 FIB entries. For each of these router families, we calculate the year to which they would be able to cope with the growth in the DFZ routing table with the existing setup and with ViAggre. Table I shows the results for the Uniform Allocation.

For ViAggre, relaxing the worst-case stretch constraints reduces FIB size and hence, extends the router life. The table shows that if the DFZ routing table were to grow at the aforementioned exponential rate, ViAggre can extend the life of the previous generation of routers

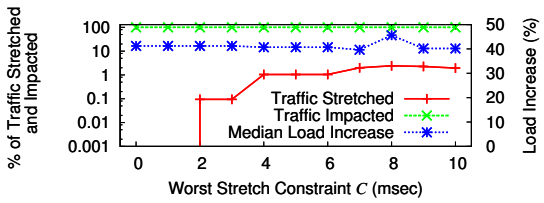



Fig. 4. Variation of the percentage of traffic stretched/impacted and load increase across routers with Worst Stretch Constraint (Uniform Allocation) and no popular prefixes.

to 2018 with no stretch at all. We realise that estimates beyond a few years are not very relevant since the ISP would need to upgrade its routers for other reasons such as newer technologies and higher data rates anyway. However, with ViAggre, at least the ISP is not forced to upgrade due to growth in the routing table.

Figure 4 plots the impact of ViAggre on the ISP's traffic and router load. The percentage of traffic stretched is small, less than 1% for $C \leq 6$ ms. This shows that almost all the traffic is routed through an aggregation point in the same PoP as the ingress. However, the fact that no prefixes are considered popular implies that almost all the traffic follows a different router-level path as compared to the status quo. This shows up in figure 4 since the traffic impacted is ≈100% throughout. This, in turn, results in a median increase in load across the routers by ≈39%. In the next section we discuss how an ISP can use the skewed distribution of traffic to address the load concern while maintaining a small FIB on its routers.

*4) Popular Prefixes:* Past studies of ISP traffic patterns from as early as 1999 have observed that a small fraction of Internet prefixes carry a large majority of ISP traffic [10,13,34,38]. We used Netflow records collected across the routers of the same tier-1 ISP as in the last section for a period of two months ($20^{th}$ Nov'07 to $20^{th}$ Jan'07) to generate per-prefix traffic statistics and observed that this pattern continues to the present day. The line labeled "Day-based, ISP-wide" in figure 5 plots the average fraction of the ISP's traffic destined to a given fraction of popular prefixes when the set of popular prefixes is calculated across the ISP on a daily basis. The figure shows that 1.5% of most popular prefixes carry 75.5% of the traffic while 5% of the prefixes carry 90.2% of the traffic.

ViAggre exploits the notion of prefix popularity to reduce its impact on the ISP's traffic. However, the ISP's routers need not consider the same set of prefixes as popular; instead the popular prefixes can be chosen per-PoP or even per-router. We calculated the fraction of traffic carried by popular prefixes, when popularity is calculated separately for each PoP on a daily basis. This is plotted in the figure as "Day-based, per-PoP" and the

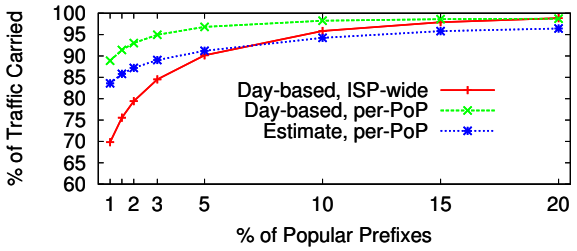
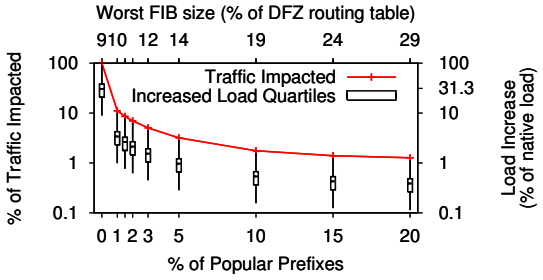
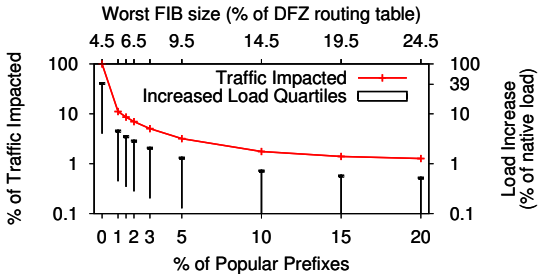



Fig. 5. Popular prefixes carry a large fraction of the ISP's traffic.




(a) With /7 allocation




(b) With Uniform allocation

Fig. 6. Variation of Traffic Impacted and Load Increase (0-25-50-75-100 percentile) with percentage of popular prefixes, $C$=4ms.

fractions are even higher.[5]

When using prefix popularity for router configuration, it would be preferable to be able to calculate the popular prefixes over a week, month, or even longer durations. The line labeled "Estimate, per-PoP" in the figure shows the amount of traffic carried to prefixes that are popular on a given day over the period of the next month, averaged over each day in the first month of our study. As can be seen, the estimate based on prefixes popular on any given day carries just a little less traffic as when the prefix popularity is calculated daily. This suggests that prefix popularity is stable enough for ViAggre configuration and the ISP can use the prefixes that are popular on a given day for a month or so. However, we admit that that these results are very preliminary and we need to study ISP traffic patterns over a longer period to substantiate the claims made above.

*5) Load Analysis:* We now consider the impact of a ViAggre deployment involving popular prefixes, i.e. the ISP populates the FIB on its routers with popular prefixes. Specifically, we focus on a deployment

[5]We did not have Netflow records for individual routers and hence, were unable to generate router-specific popular prefixes.

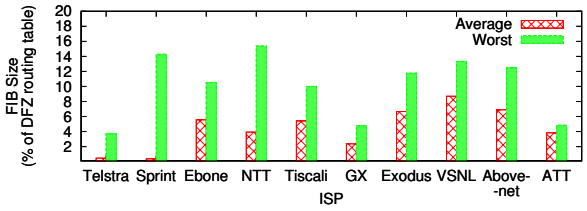



Fig. 7. FIB size for various ISPs using ViAggre.

wherein the aggregation points are assigned to constrain Worst Stretch to 4ms, i.e. $C$ = 4ms. Figure 6 shows how the traffic impacted and the quartiles for the load increase vary with the percentage of popular prefixes for both allocations. Note that using popular prefixes increases the router FIB size by the number of prefixes considered popular and thus, the upper X-axis in the figure shows the Worst FIB size. The large fraction of traffic carried by popular prefixes implies that both the traffic impacted and the load increase drops sharply even when a small fraction of prefixes is considered popular. For instance, with 2% popular prefixes in case of the uniform allocation (figure 6(b)), 7% of the traffic follows a different router-level path than before while the largest load increase is 3.1% of the original router load. With 5% popular prefixes, the largest load increase is 1.38%. Note that the more even distribution of prefixes across virtual prefixes in the uniform allocation results in a more even distribution of the excess traffic load across the ISP's routers – this shows up in the load quartiles being much smaller in figure 6(b) as compared to the ones in figure 6(a).

### C. Rocketfuel Study

We studied the topologies of 10 ISPs collected as part of the Rocketfuel project [37] to determine the FIB size savings that ViAggre would yield. Note that the fact we don't have traffic matrices for these ISPs implies that we cannot analyze the load increase across their routers. For each ISP, we used the assignment algorithm to determine the worst FIB size resulting from a ViAggre deployment where the worst stretch is limited to 5ms. Figure 7 shows that the worst FIB size is always less than 15% of the DFZ routing table. The FIB size is relatively higher for NTT and Sprint because they have a global footprint with a few small PoPs outside their main area of influence. For instance, Sprint has a few small PoPs in the Asia-Pacific region. The constraint on the worst stretch implies that in many cases, the traffic from these PoPs cannot be routed to an aggregation point in another PoP and so these PoPs must have aggregation points for all virtual prefixes. Consequently, the routers in these PoPs end up with a relatively large FIB. However, the Rocketfuel topologies are not complete and are missing routers. Hence, while the results presented here are encouraging, they should be treated as conservative estimates of the savings that ViAggre would yield for these ISPs.

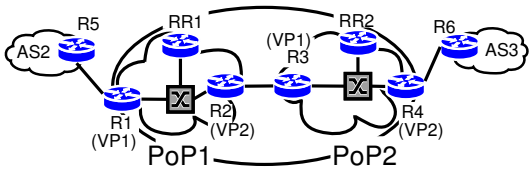



Fig. 8. WAIL topology used for our deployment. All routers in the figure are Cisco 7300s. RR1 and RR2 are route-reflectors and are not on the data path. Routers R1 and R3 aggregate virtual prefix VP1 while routers R2 and R4 aggregate VP2.

### D. Discussion

The analysis above shows that ViAggre can significantly reduce FIB size. Most of the ISPs we studied are large tier-1 and tier-2 ISPs. However, smaller tier-2 and tier-3 ISPs are also part of the Internet DFZ. Actually, it is probably more important for such ISPs to be able to operate without needing to upgrade to the latest generation of routers. The fact that these ISPs have small PoPs might suggest that ViAggre would not be very beneficial. However, given their small size, the PoPs of these ISPs are typically geographically close to each other. Hence, it is possible to use the cumulative FIB space across routers of close-by PoPs to shrink the FIB substantially. And the use of popular prefixes ensures that the load increase and the traffic impact is still small. For instance, we analyzed router topology and routing table data from a regional tier-2 ISP (AS2497) and found that a ViAggre deployment with worst stretch less than 5ms can shrink the Worst FIB size to 14.2% of the routing table today.

Further, the fact that such ISPs are not tier-1 ISPs implies they are a customer of at least one other ISP. Hence, in many cases, the ISP could substantially shrink the FIB size on its routers by applying ViAggre to the small number of prefixes advertised by their customers and peers while using default routes for the rest of the prefixes.

## V. Deployment

To verify the claim that ViAggre is a configuration-only solution, we deployed both ViAggre designs on a small network built on the WAIL testbed [3]. The test network is shown in figure 8 and represents an ISP with two PoPs. Each PoP has two Cisco 7301 routers and a route-reflector.[6] For the ViAggre deployment, we use two virtual prefixes: 0.0.0.0/1 (VP1) and 128.0.0.0/1 (VP2) with one router in each PoP serving as an aggregation point for each virtual prefix. Routers R1 and R4 have an external router connected to them and exchange routes using an eBGP peering. Specifically, router R5 advertises the entire DFZ routing table and this is, in turn, advertised through the ISP to router R6. We use OSPF for intra-domain routing. Beyond this, we configure the internal distribution of BGP routes according to the following three approaches:

1). **Status Quo.** The routers use a mesh of iBGP peerings to exchange the routes and hence, each router maintains the entire routing table.

2). **Design-I.** The routers still use a mesh of iBGP peerings to exchange routes. Beyond this, the routers are configured as follows:

– *Virtual Prefixes.* Routers advertise the virtual prefix they are aggregating to their iBGP peers.

– *FIB Suppression.* Each router only loads the routes that it is aggregating into its FIB. For instance, router R1 uses an `access-list` to specify that only routes belonging to VP1, the virtual prefix VP2 itself and any popular prefixes are loaded into the FIB. A snippet of this access-list is shown below.

```
! R5's IP address is 198.18.1.200
distance 255 198.18.1.200 0.0.0.0 1

! Don't mark anything inside 0.0.0.0/1
access-list 1 deny 0.0.0.0 128.255.255.255
! Don't mark virtual prefix 128.0.0.0/1
access-list 1 deny 0.0.0.0 128.0.0.0
! Don't mark popular prefix 122.1.1.0/24
access-list 1 deny 122.1.1.0 0.0.0.255
! ... other popular prefixes follow ...

! Mark the rest with admin distance 255

access-list 1 permit any
```

Here, the `distance` command sets the administrative distance of all prefixes that are accepted by `access-list 1` to "255" and these routes are not loaded by the router into its FIB.

– *LSPs to external routers.* We use MPLS for the tunnels between routers. To this effect, LDP [1] is enabled on the interfaces of all routers and establishes LSPs between the routers. Further, each edge router (R1 and R4) initiates a Downstream Unsolicited tunnel [1] for each external router connected to them to all their IGP neighbors using LDP. This ensures that packets to an external router are forwarded using MPLS to the edge router which strips the MPLS header before forwarding them onwards.

Given this setup and assuming no popular prefixes, routers R1 and R3 store 40.9% of today's routing table (107,943 prefixes that are in VP1) while R2 and R4 store 59.1%.

3). **Design-II.** The routers in a PoP peer with the route-reflector of the PoP and the route-reflectors peer with each other. External routers R1 and R6 are reconfigured to have eBGP peerings with RR1 and RR2 respectively. The advertisement of virtual prefixes and the MPLS configuration is the same as above. Beyond this, the route-reflectors are configured to ensure that they only advertise the prefixes being aggregated by a router to it. For instance, RR1 uses a `prefix-list` to ensure that only prefixes belonging to VP1, virtual prefix VP2 itself

[6]These are used only for the design-II deployment. We used both a Cisco 7301 and a Linux PC as a route-reflector.

and popular prefixes are advertised to router R1. The structure of this prefix-list is similar to the access-list shown above. Finally, route-reflectors use a route-map on their eBGP peerings to change the BGP NEXT_HOP of the advertised routes to the edge router that the external peer is connected too. This ensures that the packets don't actually flow through the route-reflectors.

### A. Configuration Overhead

A drawback of ViAggre being a "configuration-only" approach is the overhead that the extra configuration entails. The discussion above details the extra configuration that routers need to participate in ViAggre. Based on our deployment, the number of extra configuration lines needed for a router $r$ to be configured according to design-I is given by $(r_{int} + r_{ext} + 2|A(r)| + |P_r| + 6)$ where $r_{int}$ is the number of router interfaces, $r_{ext}$ is the number of external routers $r$ is peering with, $|A(r)|$ is the number of virtual prefixes $r$ is aggregating and $|P_r|$ is the number of popular prefixes in $r$. Given the size of the routing table today, considering even a small fraction of prefixes as popular would cause the expression to be dominated by $|P_r|$ and can represent a large number of configuration lines.

However, quantifying the extra configuration lines does not paint the complete picture since given a list of popular prefixes, it is trivial to generate an access or prefix-list that would allow them. To illustrate this, we developed a configuration tool as part of our deployment effort. The tool is 334 line python script which takes as input a router's existing configuration file, the list of virtual prefixes, the router's (or representative) Netflow records and the percentage of prefixes to be considered popular. The tool extracts relevant information, such as information about the router's interfaces and peerings, from the configuration file. It also uses the Netflow records to determine the list of prefixes to be considered popular. Based on these extracted details, the script generates a configuration file that allows the router to operate as a ViAggre router. We have been using this tool for experiments with our deployment and it is available at [45]. Further, we use *clogin* [42] to automatically load the generated ViAggre configuration file onto the router. Thus, we can reconfigure our testbed from status quo operation to ViAggre operation (design-I and design II) in an automated fashion. While our tool is specific to the router vendor and other technologies in our deployment, its simplicity and our experience with it lends evidence to the argument that ViAggre offers a good trade-off between the configuration overhead and increased routing scalability.

### B. Control-plane Overhead

Section IV evaluated the impact of ViAggre on the ISP's data plane. Beyond this, ViAggre uses control-plane mechanisms to divide the routing table amongst the ISP's routers – Design-I uses `access-lists` and Design-II uses `prefix-lists`. We quantify the performance overhead imposed by these mechanisms using our deployment. Specifically, we look at the impact of our designs on the propagation of routes through the ISP.

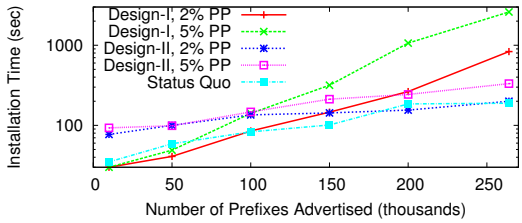
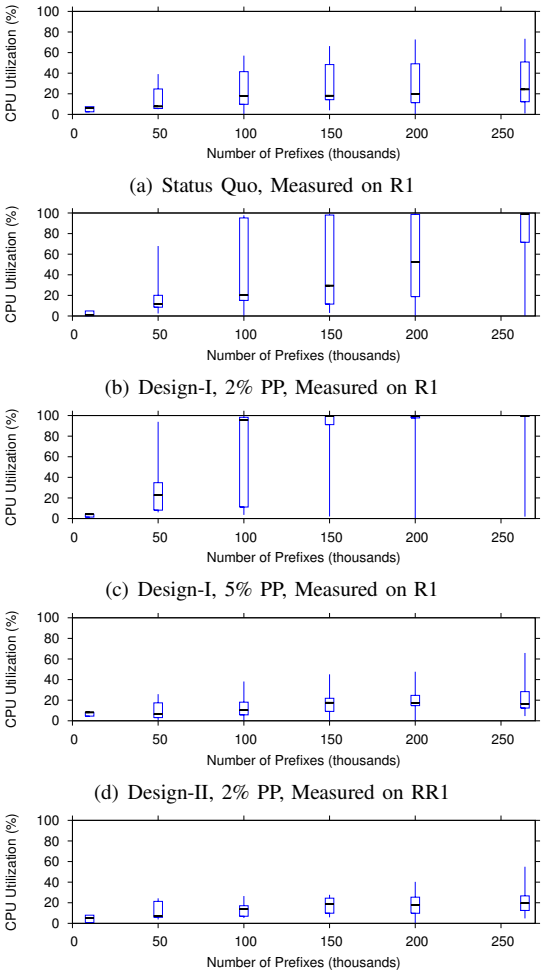



Fig. 9. Installation time with different approaches and varying fraction of Popular Prefixes (PP).




(a) Status Quo, Measured on R1

(b) Design-I, 2% PP, Measured on R1

(c) Design-I, 5% PP, Measured on R1

(d) Design-II, 2% PP, Measured on RR1

(e) Design-II, 5% PP, Measured on RR1

Fig. 10. CPU Utilization quartiles (0-25-50-75-100 percentile) for the three approaches and different fraction of Popular Prefixes (PP).

To this effect, we configured the internal distribution of BGP routes in our testbed according to the three approaches described above. External router R5 is configured to advertise a variable number of prefixes through

its eBGP peering. We restart this peering on router R5 and measure the time it takes for the routes to be installed into the FIB of the ISP's routers; hereon we refer to this as the *installation time*. During this time, we also measure the CPU utilization on the routers. We achieve this by using a clogin script to execute the "*show process cpu*" command on each router every 5 seconds. The command gives the average CPU utilization of individual processes on the router over the past 5 seconds and we extract the CPU utilization of the "BGP router" process.

We measured the installation time and the CPU utilization for the three approaches. For status quo and design-I, we focus on the measurements for router R1 while for design-II, we focus on the measurements for route-reflector RR1. We also varied the number of popular prefixes. Here we present results with 2% and 5% popular prefixes. Figures 9 and 10 plot the installation time and the quartiles for the CPU utilization respectively.

*Design-I Vs Status Quo*. Figure 9 shows that the installation time with design-I is much higher than that with status quo. For instance, with status quo, the complete routing table is transferred and installed on router R1 in 189 seconds while with design-I and 2% popular prefixes, it takes 834 seconds. Further, the design-I installation time increases significantly as the number of popular prefixes increases. Finally, figures 10(b) and 10(c) show that design-I results in very high CPU load during the transfer which increases as more prefixes are considered popular. This results from the fact that access-lists with a large number of rules are very inefficient and would obviously be unacceptable for an ISP deploying ViAggre. While we are currently exploring ways to achieve FIB suppression without the use of access-list, we note that the performance of access-lists has been improved on current generation Cisco routers (12000 and onwards) [41].

*Design-II Vs Status Quo*. Figure 9 shows that the time to transfer and install routes with design-II is not much higher than status quo, especially with 2% popular prefixes and a large number of advertised routes. For instance, design-II with 2% popular prefixes leads to an installation time of 200 seconds for the entire routing table as compared to 189 seconds for status quo. Figures 10(d) and 10(e) show that the CPU utilization is low with median utilization being less than 20%. We note that the increasing the number of prefixes being advertised increases the number of popular prefixes which, in turn, increases the size of the prefix-list being used. The CPU utilization increases as the number of prefixes advertised increases and then tapers off. Further, the trend is similar to status quo (figure 10(a)). Also note that the utilization shown for design-II was measured on route-reflector RR1 which has fewer peerings than router R1 in status quo. This explains the fact that the utilization with design-II is less than status quo.

### C. Failover

As detailed in section II-E, as long as alternate aggregation points exist, traffic in a ViAggre network is automatically re-routed upon failure of the aggregation point being used. We measured this failover time using our testbed. In the interest of space, we very briefly summarise the experiment here. We generated UDP traffic between PCs connected to routers R5 and R6 (figure 8) and then crashed the router being used as the aggregation point for the traffic. We measured the time it takes for traffic to be re-routed over 10 runs with each design. In both cases, the maximum observed failover time was 200 usecs. This shows that our designs ensure fast failover between aggregation points.

## VI. Discussion

**Pros.** ViAggre can be *incrementally deployed* by an ISP since it does not require the cooperation of other ISPs and router vendors. The ISP does not need to change the structure of its PoPs or its topology. What's more, an ISP could experiment with ViAggre on a limited scale (a few virtual prefixes or a limited number of PoPs) to gain experience and comfort before expanding its deployment. None of the attributes in the BGP routes advertised by the ISP to its neighbors are changed due to the adoption of ViAggre. Also, the use of ViAggre by the ISP does not restrict its routing policies and route selection. Further, at least for design-II, the control-plane overhead is minimal and hence, properties such as convergence times are similar. Finally, there is *incentive for deployment* since the ISP improves its own capability to deal with routing table growth.

**Management Overhead.** As detailed in section V-A, ViAggre requires extra configuration on the ISP's routers. Beyond this, the ISP needs to make a number of deployment decisions such as choosing the virtual prefixes to use, deciding where to keep aggregation points for each virtual prefix, and so on. Apart from such one-time or infrequent decisions, ViAggre may also influence very important aspects of the ISP's day-to-day operation such as maintenance, debugging, etc. All this leads to increased complexity and there is a cost associated with the extra management.

In section V-A we discussed a configuration tool that automates ViAggre configuration. We are also implementing a planning tool that takes as input high-level constraints specified by the human ISP manager such as constraints on the traffic stretch, router load, router memory used and the robustness of the resulting design. It then uses ILP to solve a multiple-constraint optimization problem to generate VA-specific deployment details such as the assignment of aggregation points. These two

tools combined would provide human ISP managers an automated means to adopt ViAggre without needing to delve into ViAggre and configuration-specific details.

It is difficult to speculate about actual costs and so we don't compare the increase in management costs against the cost of upgrading routers. While we hope that our tools will actually lead to cost savings for a ViAggre network, an ISP might just be inclined to adopt ViAggre because it breaks the dependency of various aspects of its operation on the size of the routing table. These aspects include its upgrade cycle, the per-byte forwarding cost, the per-byte forwarding power, etc.

**Other concerns.** An important concern arising out of the use of ViAggre is the tunneling overhead. However, the extensive use of tunnels (MPLS, GRE-IP, IPSec, VLAN tunneling) in ISP networks has meant that most routers are already equipped with interfaces that have extensive tunneling and detunneling capabilities at line rates [14].

As mentioned earlier, ViAggre represents a trade-off between FIB shrinkage on one hand and increased router load and traffic stretch on the other. The fact that Internet traffic follows a power-law distribution makes this a very beneficial trade-off. This power-law observation has held up in measurement studies from 1999 [10] to 2008 (in this paper) and hence, Internet traffic has followed this distribution for at least the past nine years in spite of the rise in popularity of P2P and video streaming. We believe that, more likely than not, future Internet traffic will be power-law distributed and hence, ViAggre will represent a good trade-off for ISPs.

**Other design points.** The ViAggre proposal presented in this paper represents one point in the design space that we focussed on for the sake of concreteness. Alternative approaches based on the same idea include

– *Adding routers.* We have presented a couple of techniques that ensure that only a subset of the routing table is loaded into the FIB. Given this, an ISP could install "slow-fat routers", low-end devices (or maybe even a stack of software routers [17]) in each PoP that are only responsible for routing traffic destined to unpopular prefixes. These devices forward a low-volume of traffic, so it would be easier and cheaper to hold the entire routing table. The popular prefixes are loaded into existing routers. This approach does away with a lot of deployment complexity. However, apart from the cost of the additional devices, this leads to concerns similar to the ones that ISPs have regarding routers that cache routes. For instance, attack traffic to unpopular prefixes could lead to a high relative increase in load across the low-end devices.

– *Router changes.* Routers can be changed to be ViAggre-aware and hence, make virtual prefixes first-class network objects. This would do away with a lot of the configuration complexity that ViAggre entails, ensure that ISPs get vendor support and hence, make it more palatable for ISPs. We, in cooperation with a router vendor, are exploring this option [16].

Routers today tend to have multiple blades with each blade maintaining its own copy of the entire routing table. Another approach involving vendor support is to split the routing table amongst router blades using ViAggre and hence, achieve FIB shrinkage with less burden on the ISP itself.

– *Clean-slate ViAggre.* The basic concept of virtual networks can be applied in an inter-domain fashion. The idea here is to use cooperation amongst ISPs to induce a routing hierarchy that is more aggregatable and hence, can accrue benefits beyond shrinking the router FIB. This involves virtual networks for individual virtual prefixes spanning domains such that even the RIB on a router only contains the prefixes it is responsible for. This would reduce both the router FIB and RIB and in general, improve routing scalability. We intend to study the merits and demerits of such an approach in future work.

## VII. Related Work

A number of efforts have tried to directly tackle the routing scalability problem through clean-slate designs. One set of approaches try to reduce routing table size by dividing edge networks and ISPs into separate address spaces [7,11,29,32,40]. Our work resembles some aspects of CRIO [40] which uses virtual prefixes and tunneling to decouple network topology from addressing. However, CRIO requires adoption by all provider networks and like [7,11,29,32], requires a new mapping service to determine tunnel endpoints. APT [23] presents such a mapping service. Alternatively, it is possible to encode location information into IP addresses [8,15,19] and hence, reduce routing table size. Finally, an interesting set of approaches that trade-off stretch for routing table size are *Compact Routing* algorithms; see [26] for a survey of the area.

The use of tunnels has long been proposed as a routing scaling mechanism. VPN technologies such as BGP-MPLS VPNs [9] use tunnels to ensure that only PE routers need to keep the VPN routes. As a matter of fact, ISPs can and probably do use tunneling protocols such as MPLS and RSVP-TE to engineer a BGP-free core [35]. However, edge routers still need to keep the full FIB. With ViAggre, none of the routers on the datapath need to maintain the full FIB. Router vendors, if willing, can use a number of techniques to reduce the FIB size, including FIB compression [35] and route caching [35]. Forgetful routing [24] selectively discards alternative routes to reduce RIB size. [2] sketches the basic ViAggre idea.

In recent work, Kim et. al. [25] use relaying, similar to ViAggre's use of aggregation points, to address the VPN

routing scalability problem. The VPN setting involves VPN-specific routing tables and the task of maintaining these can be split amongst PE routers; in our setting there is just the Internet routing table and we use the concept of virtual prefixes to make it divisible. We also have the additional challenge of dealing with networks other than customers since these networks might be advertising the full routing table, which is solved by not installing some routes in the FIB (design-I) or through the use filters on route-reflectors (design-II).

Over the years, several articles have documented the existing state of inter-domain routing and delineated requirements for the future [5,12,28]; see [12] for other routing related proposals. RCP [4] and 4D [18] argue for logical centralization of routing in ISPs to provide scalable internal route distribution and a simplified control plane respectively. We note that ViAggre fits well into these alternative routing models. As a matter of fact, the use of route-reflectors in design-II is similar in spirit to RCSs in [4] and DEs in [18].

## VIII. Summary

This paper presents ViAggre, a technique that can be used by an ISP to substantially shrink the FIB on its routers and hence, extend the lifetime of its installed router base. The ISP may have to upgrade the routers for other reasons but at least it is not driven by DFZ growth over which it has no control. While it remains to be seen whether the use of automated tools to configure and manage large ViAggre deployments can offset the complexity concerns, we believe that the simplicity of the proposal and its possible short-term impact on routing scalability suggest that is an alternative worth considering.

## References


[1] Andersson, L., Minei, I., and Thomas, B. RFC 5036 - LDP Specification, Jan 2006.

[2] Ballani, H., Francis, P., Cao, T., and Wang, J. ViAggre: Making Routers Last Longer! In *Proc. of Hotnets* (Oct 2008).

[3] Barford, P. Wisconsin Advanced Internet Laboratory (WAIL), Dec 2007. http://wail.cs.wisc.edu/.

[4] Caesar, M., Caldwell, D., Feamster, N., Rexford, J., Shaikh, A., and van der Merwe, J. Design and Implementation of a Routing Control Platform . In *Proc. of Symp. on Networked Systems Design and Implementation (NSDI)* (2005).

[5] Davies, E., and Doria, A. Analysis of Inter-Domain Routing Requirements and History. Internet Draft draft-irtf-routing-history-07.txt, Jan 2008.

[6] de Silva, S. 6500 FIB Forwarding Capacities. NANOG 39 meeting, 2007. http://www.nanog.org/mtg-0702/presentations/fib-desilva.pdf.

[7] Deering, S. The Map & Encap Scheme for scalable IPv4 routing with portable site prefixes, March 1996. http://www.cs.ucla.edu/~lixia/map-n-encap.pdf.

[8] Deering, S., and Hinden, R. IPv6 Metro Addressing. Internet Draft draft-deering-ipv6-metro-addr-00.txt, Mar 1996.

[9] E. Rosen and Y. Rekhter. RFC 2547 - BGP/MPLS VPNs, Mar 1999.

[10] Fang, W., and Peterson, L. Inter-As traffic patterns and their implications. In *Proc. of Global Internet* (1999).

[11] Farinacci, D., Fuller, V., Oran, D., and Meyer, D. Locator/ID Separation Protocol (LISP). Internet Draft draft-farinacci-lisp-02.txt, July 2007.

[12] Feamster, N., Balakrishnan, H., and Rexford, J. Some Foundational Problems in Interdomain Routing. In *Proc. of Workshop on Hot Topics in Networks (HotNets-III)* (2004).

[13] Feldmann, A., Greenberg, A., Lund, C., Reingold, N., Rexford, J., and True, F. Deriving traffic demands for operational IP networks: methodology and experience. *IEEE/ACM Trans. Netw. 9*, 3 (2001).

[14] Francios, P., and Bonaventure, O. An evaluation of IP-based Fast Reroute Techniques. In *Proc. of CoNEXT* (2005).

[15] Francis, P. Comparision of geographical and provider-rooted Internet addressing. *Computer Networks and ISDN Systems 27*, 3 (1994).

[16] Francis, P., Xu, X., and Ballani, H. FIB Suppression with Virtual Aggregation and Default Routes. Internet Draft draft-francis-idr-intra-va-01.txt, Sep 2008.

[17] Gillian, B. VYATTA: Linux IP Routers, Dec 2007. http://freedomhec.pbwiki.com/f/linux_ip_routers.pdf.

[18] Greenberg, A., Hjalmtysson, G., Maltz, D. A., Meyers, A., Rexford, J., Xie, G., Yan, H., Zhan, J., and Zhang, H. A clean slate 4D approach to network control and management. *ACM SIGCOMM Computer Communications Review* (October 2005).

[19] Hain, T. An IPv6 Provider-Independent Global Unicast Address Format. Internet Draft draft-hain-ipv6-PI-addr-02.txt, Sep 2002.

[20] Hughes, D., Dec 2004. PACNOG list posting http://mailman.apnic.net/mailing-lists/pacnog/archive/2004/12/msg00000.html.

[21] Huston, G. BGP Reports, Dec 2007. http://bgp.potaroo.net/.

[22] Huston, G., and Armitage, G. Projecting Future IPv4 Router Requirements from Trends in Dynamic BGP Behaviour. In *Proc. of ATNAC* (2006).

[23] Jen, D., Meisel, M., Massey, D., Wang, L., Zhang, B., and Zhang, L. APT: A Practical Transit Mapping Service. Internet Draft draft-jen-apt-01.txt, Nov 2007.

[24] Karpilovsky, E., and Rexford, J. Using forgetful routing to control BGP table size. In *Proc. of CoNext* (2006).

[25] Kim, C., Gerber, A., Lund, C., Pei, D., and Sen, S. Scalable VPN Routing via Relaying. In *Proc. of ACM SIGMETRICS* (2008).

[26] Krioukov, D., and kc claffy. Toward Compact Interdomain Routing, Aug 2005. http://arxiv.org/abs/cs/0508021.

[27] Li, T. Router Scalability and Moore's Law, Oct 2006. http://www.iab.org/about/workshops/routingandaddressing/Router_Scalability.pdf.

[28] Mao, Z. M. Routing Research Issues. In *Proc. of WIRED* (2003).

[29] Massey, D., Wang, L., Zhang, B., and Zhang, L. A Proposal for Scalable Internet Routing & Addressing. Internet Draft draft-wang-ietf-efit-00, Feb 2007.

[30] Meyer, D., Zhang, L., and Fall, K. Report from the IAB Workshop on Routing and Addressing. Internet Draft draft-iab-raws-report-02.txt, Apr 2007.

[31] Narten, T. Routing and Addressing Problem Statement. Internet Draft draft-narten-radir-problem-statement-02.txt, Apr 2008.

[32] O'Dell, M. GSE–An Alternate Addressing Architecture for IPv6. Internet Draft draft-ietf-ipngwg-gseaddr-00.txt, Feb 1997.

[33] Ravi, R., and Sinha, A. Multicommodity facility location. In *Proc. of ACM-SIAM SODA* (2004).

[34] Rexford, J., Wang, J., Xiao, Z., and Zhang, Y. BGP routing stability of popular destinations. In *Proc. of Internet Measurment Workshop* (2002).

[35] Scudder, J. Router Scaling Trends. APRICOT Meeting, 2007. http://submission.apricot.net/chatter07/slides/future_of_routing.

[36] Spring, N., Mahajan, R., and Anderson, T. Quantifying the Causes of Path Inflation. In *Proc. of ACM SIGCOMM* (2003).

[37] Spring, N., Mahajan, R., and Wetherall, D. Measuring ISP topologies with Rocketfuel. In *Proc. of ACM SIGCOMM* (2002).

[38] Taft, N., Bhattacharyya, S., Jetcheva, J., and Diot, C. Understanding traffic dynamics at a backbone PoP. In *Proc. of Scalability and Traffic Control and IP Networks SPIE ITCOM* (2001).

[39] Y. Rekhter and T. Li and S. Hares, Ed. RFC 4271 - A Border Gateway Protocol 4 (BGP-4), Jan 2006.

[40] Zhang, X., Francis, P., Wang, J., and Yoshida, K. Scaling Global IP Routing with the Core Router-Integrated Overlay. In *Proc. of ICNP* (2006).

[41] Access List Performance Improvements, Oct 2008. http://www.cisco.com/en/US/docs/ios/12_0s/feature/guide/hw_acl.html.

[42] clogin Manual Page, Oct 2008. http://www.shrubbery.net/rancid/man/elogin.1.html.

[43] Foundry Router Reference, Jul 2008. http://www.foundrynetworks.co.jp/services/documentation/srcli/BGP_cmds.html.

[44] JunOS Route Preferences, Jul 2008. http://www.juniper.net/techpubs/software/junos/junos60/swconfig60-routing/html/protocols-overview4.html.

[45] ViAggre Configuration Tool, Oct 2008. http://www.cs.cornell.edu/~hitesh/va-tools/va_conf_generator.py.