# On the Scalability of Router Forwarding Tables: Nexthop-Selectable FIB Aggregation

Qing Li*, Dan Wang†, Mingwei Xu* and Jiahai Yang‡
* Dept. of Comp. Sci. & Tech., Tsinghua Univ., Tsinghua National Laboratory for Information Science and Technology
† Dept. of Computing, The Hong Kong Polytechnic Univ. ‡ Network Research Center, Tsinghua Univ.

***Abstract*—In recent years, the core-net routing table, e.g., Forwarding Information Base (FIB), is growing at an alarming speed and this has become a major concern for Internet Service Providers. One effective solution for this routing scalability problem, which requires only upgrades on individual routers, is FIB aggregation. Intrinsically, IP prefixes with numerical prefix matching and the same next hop can be aggregated. Very commonly, all previous studies assume that each IP prefix has one corresponding next hop, i.e., towards $one$ optimal path. In this paper, we argue that a packet can be delivered to its destination through a path other than the $one$ optimal path. Based on this observation, we for the first time propose Nexthop-Selectable FIB Aggregation that is fundamentally different from all previous aggregation schemes. IP prefixes are aggregated if they have numerical prefix matching and share one common next hop. Consequently, IP prefixes that cannot be aggregated, due to lack of the same next hop, are aggregated; and we achieve a substantially higher aggregation ratio.**

**In this paper, we provide a systematic study on this Nexthop-Selectable FIB Aggregation problem. We present several practical choices to build the sets of selectable next hops for the IP prefixes. To maximize the aggregation, we formulate the problem as an optimization problem. We show that the problem can be solved by dynamic programming. While the straightforward application of dynamic programming has exponential complexity, we propose a novel algorithm that is $O(N)$. We then develop an optimal online algorithm with constant running time. We evaluate our algorithms through a comprehensive set of simulations with BRITE with RIBs collected from RouteViews. Our evaluation shows that we can reduce more than an order of the FIB size.**

## I. Introduction

The global Internet has experienced tremendous growth over the last decade. The sheer growth of user population, as well as such factors as multi-homing, traffic engineering, policy routing, have driven the growth of Default Free Zone (DFZ) routing table size at an alarming rate [1–3] (see the routing table size from 1994 to 2010 in Fig. 1). The Internet Service Providers (ISPs) are forced to upgrade routers in an unanticipated pace. 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. This suggests that ISPs are undergoing some pain to avoid the cost of router upgrades.

To handle this severe Internet routing scalability problem, many solutions are proposed. One set of proposals is to get rid of the IP-oriented infrastructure and design a fully scalable addressing system. Example protocols and frameworks include [4–6]. Another set of proposals is to protect the core-net router tables by address encapsulation or translation. These proposals [7–9] require a significant transition time, as what we have learned from the deployment of IPv6.

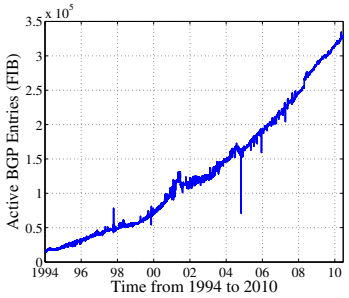



Fig. 1. The historical FIB size. Data obtained from bgp.potaroo.net [1].

A more immediate solution is Forwarding Information Base (FIB) aggregation. FIB aggregation can reduce FIB size with only local router upgrade and requires no protocol change. In FIB aggregation, multiple prefixes can be aggregated into one if two conditions hold: 1) the prefixes are numerically aggregatable; 2) their next hops are the same. Some vendors have already implemented simple FIB aggregation schemes and many techniques [10, 11] are proposed in academia. In [10], systematic analysis of FIB aggregation is presented.

All these previous studies focus on the first condition of FIB aggregation, that is, how to find or construct the numerical aggregatable prefixes. Very commonly, these algorithms consider that each IP prefix has one corresponding next hop. As a sharp contrast, in this paper, we make a key observation that it is practically possible that there are multiple selectable next hops for each IP prefix; and through any one of these next hops, the packets can be delivered to the destination. In practice, such schemes as equal-cost multi-path routing (ECMP), loop-free alternative (LFA) [12] naturally exist. There are also many multi-path routing (path protection) schemes proposed in literature, making a selection of multiple next hops possible. Consequently, we propose a fundamentally different FIB aggregation approach, where multiple IP prefixes can be aggregated into one if 1) they are numerically aggregatable, and 2) one of their selectable next hops is the same. We illustrate our idea with a simple example:

**An Example**: Consider two prefix entries, $< 158/8, a >$, $< 158.128/9, b >$. They cannot be aggregated in any previous FIB aggregation schemes, as they have different next hops. Assume that both next hops $a$ and $b$ can deliver the packets

The works of Q. Li, M. Xu and J. Yang are supported by grants of NSFC 61073166 (only for Q. Li and M. Xu), 973 Pro. 2009CB320502, 863 Pro. 2009AA01Z251, National Sci. & Tech. Pillar Pro. 2008BAH37B05 (only for J. Yang). D. Wang's work is supported by grants Hong Kong PolyU/G-YG78, A-PJ19, 1-ZV5W, and RGC/GRF PolyU 5305/08E.

of $158/8$ to the destination; and both $b$ and $c$ can deliver the packets of $158.128/9$ to the destination. Instead of allocating a single next hop for each prefix, we propose to allocate selectable next hops and the two prefix entries are as follows, $< 158/8, \{a, b\} >$, $< 158.128/9, \{b, c\} >$. Thus, these two entries can be aggregated into $< 158/8, b >$.

From this example, we can see that our assumption is that for each prefix, there is a set of *selectable* next hops. Note, however, that after aggregation, each IP prefix is still mapped with one single next hop and no multi-path routing is needed.

In this paper, we for the first time provide a systematic study on the aforementioned problems. Inspired by LFA, we introduce two principles to construct the set of selectable next hops. Through any of these next hops, the packets are guaranteed to be delivered to the destination. We formulate our Nexthop-Selectable FIB Aggregation problem as an optimization problem and show that it can be solved by dynamic programming. As a straightforward dynamic programming has exponential complexity, we develop a novel algorithm of $O(N)$. We then develop an optimal online algorithm, with constant complexity, to handle online routing updates.

We evaluate our algorithms through BRITE generated and real world topologies, with the routing tables from RouteViews. Our evaluation shows that we can achieve more than an order of the FIB size reduction, reducing the FIB size to that of 1998. As such, we believe our scheme, locally and incrementally deployable, can reserve sufficient time for the agreement of advanced infrastructure changes of the Internet.

## II. NextHop-Selectable FIB Aggregation: The Problem and Algorithms

To make our presentation focused, we delay the detailed discussion on constructing the set of selectable next hops in section III and focus on how to maximally aggregate the FIB given a set of selectable next hops allocated to each prefix. We comment that our work does not depends on specific approaches of constructing the set of selectable next hops.

### A. The Nexthop-Selectable FIB Aggregation Problem

Let a *Nexthop-Selectable FIB (NS-FIB)* be a set $\mathcal{F} = \{< p_i, \mathcal{A}_{p_i} > | p_i \in \mathcal{P}, 0 \leq i \leq N\}$ where $\mathcal{P} = \{p_i | 0 \leq i \leq N\}$ is the set of IP prefixes and $\mathcal{A}_{p_i}$ is the set of selectable next hops for $p_i$. A aggregation for $\mathcal{F}$ is a set $\mathcal{F}_{aggr} = \{< p, a_p > | p \in \mathcal{P}'$ and $a_p \in \mathcal{A}_p\}$ where $\mathcal{P}' \subseteq \mathcal{P}$. $\mathcal{F}_{aggr}$ is feasible for $\mathcal{F}$, if for any IP address and its longest matching entries $< p, \mathcal{A}_p > \in \mathcal{F}$ and $< p', a_{p'} > \in \mathcal{F}_{aggr}$, we have $a_{p'} \in \mathcal{A}_p$. The **Nexthop-Selectable FIB Aggregation Problem (NS-FIB Aggregation)** is: given a Nexthop-Selectable FIB $\mathcal{F}$, find a feasible aggregation $\mathcal{F}_{aggr}$ for $\mathcal{F}$ with a minimized $|\mathcal{F}_{aggr}|$.

We illustrate our definitions with an example. A Nexthop-Selectable FIB is shown in Fig. 2. Two corresponding feasible aggregations are shown in Fig. 3, and *Aggr. 1* is an optimal aggregation for the NS-FIB.

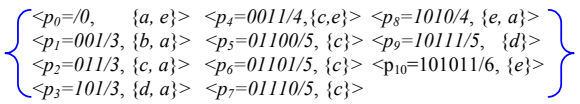



Fig. 2. An example of Nexthop-Selectable FIB.

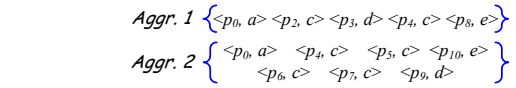
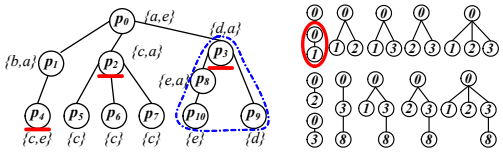



Fig. 3. Two feasible aggregations for the NS-FIB in Fig. 2.

### B. A Dynamic Programming Solution

The routing table is generally organized as a radix tree. We follow this convention. For the prefix set $\mathcal{P}$ of $\mathcal{F}$, a corresponding NS-FIB tree $\mathcal{T}(\mathcal{V}, \mathcal{E})$ can be constructed as, $\mathcal{V}$ corresponds to all the prefixes of $\mathcal{F}$. By abusing notations, we use $p$ to denote a node of $\mathcal{T}$; and $\forall p, p' \in \mathcal{V}$, $p$ is the immediate parent of $p'$ if and only if $p$ is the longest matching prefix of $p'$ (other than $p'$) in $\mathcal{T}$. For example, Fig. 4(a) shows the tree corresponding to the NS-FIB in Fig. 2.

A *subtree* $T$ of $\mathcal{T}$ is a tree that is a connected part or a single node in $\mathcal{T}$. Let $R_T$ denote the root of $T$. The indication of a subtree is that the prefixes it represents are numerically aggregatable. Thus, if all nodes of $T$ have a common next hop, they can be aggregated into $R_T$. We define *aggregation cell* (or *cell* for short) as a subtree in $\mathcal{T}$ and all the nodes in the subtree have a common next hop. Intuitively, a cell corresponds to an aggregated entry and our algorithm is to find a set of disjoint cells to cover $\mathcal{T}$. We define an $x$-selectable cell to be a cell with the common selectable next hop $x$.




(a) The NS-FIB tree $\mathcal{T}$ (b) $\mathcal{U}_a(p_0)$

Fig. 4. (a) Tree $\mathcal{T}(\mathcal{V}, \mathcal{E})$ corresponding to the NS-FIB in Fig. 2. The $p_3$-rooted branch is $\mathcal{S}_{p_3} = (p_3, p_8, p_9, p_{10})$. $\mathcal{C}_{T'} = (p_4, p_2, p_3)$ for $T' = (p_0, p_1)$. (b) $\mathcal{U}_a(p_0)$, the set of $p_0$-rooted $a$-selectable cells of $\mathcal{T}$.

We first show that our problem has an optimal sub-structure. As such, it can be solved by dynamic programming.

Let $\mathbf{G}(T)$ denote the size of an optimal aggregation of the NS-FIB $T$. Let $\mathbf{G_x}(T)$ denote the size of an optimal aggregation for the NS-FIB $T$ where the root $R_T$ selects $x \in \mathcal{A}_{R_T}$ as its next hop. Clearly $\mathbf{G}(T) \leq \mathbf{G_x}(T)$ and

$$\mathbf{G}(T) = \min_{x \in \mathcal{A}_{R_T}} \mathbf{G_x}(T) \tag{1}$$

We will show that $\mathbf{G_x}(T)$ can be linked to an optimal sub-structure. We present a few more definitions.

A *branch* $T$ of a tree $\mathcal{T}$ is a subtree consisting of a node and all of its descendants in $\mathcal{T}$. Let $\mathcal{S}_p$ be the *p-rooted branch* of $\mathcal{T}$ (See $\mathcal{S}_{p_3}$ in Fig. 4). Let $\mathcal{C}_{T'}$ denote the set of the immediate children of a subtree $T'$ in $\mathcal{T}$ (See $\mathcal{C}_{T'}$ in Fig. 4). Let $\mathcal{U}_x(p)$ denote the set of $p$-rooted $x$-selectable cells in $\mathcal{T}$ where $x \in \mathcal{A}_p$ (See $\mathcal{U}_a(p_0)$ in Fig. 4).

The optimal sub-structure of $\mathbf{G_x}(T)$ can be written as

$$\mathbf{G_x}(T) = 1 + \min_{T' \in \mathcal{U}_x(R_T)} \sum_{p \in \mathcal{C}_{T'}} \mathbf{G}(\mathcal{S}_p) \tag{2}$$

The intuition is that we want to divide a NS-FIB tree $T$ into one $R_T$-rooted $x$-selectable cell $T'$, and a set of branches rooted at the children of $T'$ ($\mathcal{C}_{T'}$). As $T'$ can be aggregated into one entry, the size of the optimal aggregation based on

this division is $1 + \sum_{p \in \mathcal{C}_{T'}} \mathbf{G}(\mathcal{S}_p)$. Note that $T'$ can be of any form, as long as it is $x$-selectable and $R_T$-rooted. Therefore, $\mathbf{G_x}(T)$ takes the minimum of all different forms of $T'$. For example, in Fig. 4, supposing $x = a$, one possible division is $T' = (p_0, p_1), \mathcal{S}_{p_4} = (p_4)$, $\mathcal{S}_{p_2} = (p_2, p_5, p_6, p_7)$ and $\mathcal{S}_{p_3} = (p_3, p_8, p_9, p_{10})$ ($\mathbf{G}(\mathcal{S}_{p_3}) = 2$). Another possible division is $T' = (p_0, p_1, p_2, p_3, p_8)$, $\mathcal{S}_{p_4} = (p_4)$, $\mathcal{S}_{p_5} = (p_5)$, $\mathcal{S}_{p_6} = (p_6)$, $\mathcal{S}_{p_7} = (p_7)$, $\mathcal{S}_{p_9} = (p_9)$ and $\mathcal{S}_{p_{10}} = (p_{10})$. We can see that the size of the optimal aggregation, based on the first division, is four (*Aggr. 1* in Fig. 3); and the size of the optimal aggregation, based on the second division, is six (*Aggr. 2* in Fig. 3). In fact, the first division achieves an optimal aggregation with $a$ selected as the next hop for $R_T$.

We initialize $\mathbf{G}(T)$ and $\mathbf{G_x}(T)$ for special cases:

$$\mathbf{G}(T) = 1, \quad \text{if} \quad |T| = 1 \tag{3}$$

$$\mathbf{G_x}(T) = \infty, \quad \text{if} \quad x \notin \mathcal{A}_{R_T} \tag{4}$$

Condition (3) says that an optimal aggregation of a single node tree is one and condition (4) says that selecting an unavailable next hop for the root of a tree is not allowed.

A dynamic programming algorithm can be derived based on the above optimal sub-structure. $|\mathcal{U}_x(R_T)|$ is exponential (proved in [13]), making the complexity of the algorithm unacceptable. In what follows, we develop a bottom-up algorithm of polynomial time.

### C. The Polynomial Time Algorithm

For a branch $T$ of $\mathcal{T}$, let $T_x^*$ be an optimal $R_T$-rooted $x$-selectable cell with the *optimality* of minimizing $\sum_{p \in \mathcal{C}_{T_x^*}} \mathbf{G}(\mathcal{S}_p)$ (thus equal to $\mathbf{G}_x(T) - 1$). The crucial challenge for calculating $\mathbf{G_x}(T)$ and $\mathbf{G}(T)$ is to efficiently find $T_x^* \in \mathcal{U}_x(R_T)$. We propose Algorithm OptimalCell() to compute $T_x^*$ and $\mathbf{G_x}(T)$. OptimalCell() will become a building block of our main algorithm. The intuition of OptimalCell() is that instead of searching for the entire $\mathcal{U}_x(R_T)$, it is enough to only evaluate the children of $R_T$, and combine the optimal cells rooted at the children of $R_T$ if necessary. Therefore, the complexity of OptimalCell() is $\Theta(|\mathcal{C}_{(R_T)}|)$, which is only related to the number of the children of the branch root.

We now prove that OptimalCell() finds an optimal $R_T$-rooted $x$-selectable cell $T_x^*$ and computes $\mathbf{G_x}(T)$. We first prove two lemmas. The first lemma says that the branches of an optimal $x$-selectable cell are also optimal $x$-selectable cells. This lemma is the foundation for the correctness of dynamic programming. The second lemma says that any branch of an optimal cell $T_x$ can be exchanged with another optimal cell without changing the optimality of $T_x$. This lemma is critical as we will show that any optimal $R_T$-rooted $x$-selectable cell can be transformed to $T_x^*$ computed by OptimalCell().

**Lemma 1.** *The branches of an optimal $x$-selectable cell are also optimal $x$-selectable cells.*[1]

**Lemma 2.** *A branch ($T'$) of an optimal $x$-selectable cell $T_x$ can be exchanged with any other optimal $R_{T'}$-rooted $x$-selectable cell ($T''$), without changing the optimality of $T_x$.*

[1]The proofs for all the lemmas and theories can be found in [13].

**Theorem 3.** *OptimalCell() computes an optimal $R_T$-rooted $x$-selectable cell and $\mathbf{G_x}(T)$.*

Based on Algorithm OptimalCell(), we propose Algorithm NS-FIB-Aggregation() to calculate $\mathbf{G}(T)$ iteratively by dynamic programming. Given an NS-FIB tree $T = \mathcal{T}$ as the input, NS-FIB-Aggregation() computes $\mathbf{G}(T)$ by computing the $\mathbf{G}$ value for all the branches of $T$ according to Formula (1) and (2). An optimal aggregation $\mathcal{F}_{aggr}$ for $\mathcal{F}$ is generated in the process of computing $\mathbf{G}(T)$. $\mathcal{F}_{aggr}$ is stored in the original tree structure. The selected next hops for $\mathcal{F}_{aggr}$ are also set in the algorithm.

**Algorithm 1** OptimalCell($x$, $T$)

**Require:** $x \in \mathcal{A}_{R_T}$
1: $T' \Leftarrow \{R_T\}$, $G' \Leftarrow 1$;
2: **for all** $p \in$ children of $R_T$ **do**
3: **if** $x \in A_p$ and $\mathbf{G_x}(\mathcal{S}_p) = \mathbf{G}(\mathcal{S}_p)$ **then**
4: $\mathbf{G}' \Leftarrow \mathbf{G}' + \mathbf{G}(\mathcal{S}_p) - 1$
5: $T' \Leftarrow T' \cup (\mathcal{S}_p)_x^*$ $\{(\mathcal{S}_p)_x^*$ is an optimal cell rooted at $p\}$
6: **else** $\mathbf{G}' \Leftarrow \mathbf{G}' + \mathbf{G}(\mathcal{S}_p)$
7: **end if**
8: **end for**
9: $T_x^* \Leftarrow T'$, $\mathbf{G_x}(T) \Leftarrow \mathbf{G}'$

**Algorithm 2** NS-FIB-Aggregation($T$)

1: **for all** $p \in V_T$ **do** {postorder}
2: $\mathbf{G}(\mathcal{S}_p) \Leftarrow \infty$
3: **for all** $x \in A_p$ **do**
4: Compute $(\mathcal{S}_p)_x^*$ and $\mathbf{G_x}(\mathcal{S}_p)$ by OptimalCell($x$, $\mathcal{S}_p$)
5: **if** $\mathbf{G}(\mathcal{S}_p) > \mathbf{G_x}(\mathcal{S}_p)$ **then**
6: Set $\mathbf{G}(\mathcal{S}_p)$ as $\mathbf{G_x}(\mathcal{S}_p)$
7: Set the selected next hop of $R_{\mathcal{S}_p}$ as $x$
8: Set the aggr. children of $p$ as $\mathcal{C}_{(\mathcal{S}_p)_x^*}$
9: **end if**
10: **end for**
11: **end for**

We show an example of Algorithm NS-FIB-Aggregation() with the NS-FIB $\mathcal{T}$ in Fig. 4 as the input. In Fig. 5, $\forall p \in \mathcal{T}$, a tuple $[\mathbf{G_p}, \mathbf{x}]$ is associated with $p$. $\mathbf{G}_p$ corresponds to $\mathbf{G}(\mathcal{S}_p)$ and $x$ corresponds to the selected next hop for $p$ by NS-FIB-Aggregation(). A subtree with a dash circle is an optimal cell. The figure shows the last round of NS-FIB-Aggregation() to compute $\mathbf{G}(\mathcal{T})$ and generate an optimal aggregation. The black nodes and their selected next hops form an optimal aggregation for $\mathcal{T}$, which is the same as *Aggr. 1* in Fig. 3.

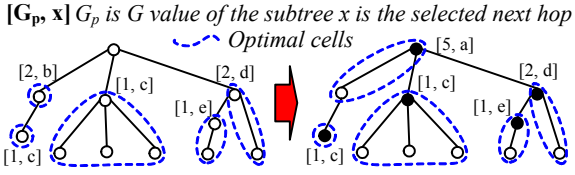



Fig. 5. The last round of NS-FIB-Aggregation() to generate an optimal aggregation for the NS-FIB in Fig. 4.

**Theorem 4.** *NS-FIB-Aggregation() computes an optimal solution for the Nexthop-Selectable FIB Aggregation Problem.*

**Theorem 5.** *The complexity of NS-FIB-Aggregation() is $O(mN)$ where $m$ is the maximum number of next hops for any prefix. Since $m$ is a constant, the complexity is $O(N)$.*

### D. The Online Algorithm

In practice, the routing changes are online. The BGP route changes may trigger route withdrawal, route update or route insertion in the NS-FIB. We use UPDATE as an example. The operations of handling a BGP update is 1) update the BGP route in the RIB. Generally speaking, this operation involves a search of the RIB entry for the updated prefix, 2) if the optimal BGP route changes (e.g., the egress router is changed), find the new next hop(s) for this prefix according to the intra-domain routing table (i.e., generate a new FIB entry), and 3) update the FIB change in the line card. The FIB aggregation algorithms will fall between 2) and 3). The bottleneck of the above operations is 1), which has complexity of $O(\log N)$.

We develop Algorithm NS-FIB-Online(). The idea is that given an update $< p, A_{new} >$ where $p$ is a prefix and $A_{new}$ is the new set of next hops of $p$, NS-FIB-Online() recalculates $\mathbf{G}$, $\mathbf{G_x}$ values of the node $p$ and all the upstream nodes of $p$ in $\mathcal{T}$. We give the detailed description of the algorithm and the analysis of constant complexity in [13].

## III. Construction of Nexthop-Selectable FIB

In this section, we propose two practical methods to construct the set of selectable next hops for each prefix. Our schemes require only local information and no modification on existing Internet infrastructure. The idea is inspired by loop-free alternates (LFA) [12]. We use *loop-free condition (LFC)* and *downstream condition (DSC)* to select next hops for each prefix. In Fig. 6, we show an example of LFC and DSC.

**LFC Nexthop-Selectable FIB Construction:** Let $Dt$ be the destination for $p$ in the AS (i.e., the egress router for $p$). For a router $Rt$, a neighbor $NG_i$ of $Rt$ meets *LFC Condition* for $Dt$ if and only if $Rt$ is not on the optimal route(s) from $NG_i$ to $Dt$. The optimal next hop(s) always meet(s) LFC condition. Thus, given the RIB of $Rt$ and the topology of the AS that $Rt$ belongs to, we can construct the set of selectable next hops for a prefix according to LFC. Such computation only uses information of the local router.

**Lemma 6.** *Let $Rt$ construct the set of selectable next hops according to LFC condition. Assume all the other routers in the AS still choose the optimal next hop for each prefix, then no routing loop exists.*

Note that LFC Nexthop-Selectable FIB construction is useful if it is not widely deployed. For the ISPs that wants to selectively deploy our scheme for their most aged routers, LFC is suggested. We next present another construction scheme which guarantees loop avoidance in all circumstances.

**DSC Nexthop-Selectable FIB Construction:** Given a router $Rt$ and a destination prefix $Dt$, a neighbor $NG_i$ of $Rt$ meets *DSC Condition* if and only if the optimal path from $NG_i$ to $Dt$ is shorter than the optimal path from $Rt$ to $Dt$. If $NG_i$ meets DSC condition, it also meets LFC condition for $Dt$. Given the RIB of $Rt$ and the topology of the AS $Rt$ belongs to, we can construct the set of selectable next hops for a prefix according to DSC condition.

**Lemma 7.** *If any $Rt$ in the AS construct the selectable next hops according to DSC condition, no forwarding loop exists.*

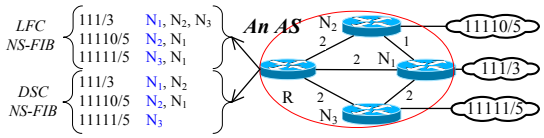



Fig. 6. An example of LFC and DSC Nexthop-Selectable FIBs. The costs of the links are specified on the lines. For router $R$, $N_3$ is choosen as an LFC selectable next hop for *111/3* since $R$ is not on the optimal route from $N_3$ to $N_1$. Similarly, N2, N3 are also LFC selectable next hops for *111/3*.

## IV. Simulation

To construct the NS-FIB, both the RIB and AS topology are required. We generate topologies by BRITE. We use the propagation delay as the cost of a link. The parameters for BRITE are in Table I. We use the RIB of 16th May 2010 from RouteViews, which has 328,076 entries and 37 next AS hops. Note that this RIB only has next AS hops, but no next router hops. We randomly select 37 routers from each AS as egress routers and map the 37 BGP next AS hops to these routers. The next router hops can thus be computed. This method has been proven to be a good approximation.

TABLE I
Parameters of BRITE Topologies

| Mode | Model | HS | LS | Nodes Num |
|---|---|---|---|---|
| Router Only | Waxman | 1000 | 100 | 50-500 |
| links/new node | $\alpha$ / $\beta$ | NP | Growth Type | |
| 2-15 | 0.15 / 0.2 | Random | Incremental | |

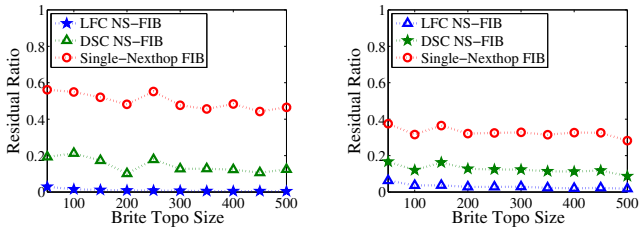
We use residual ratio as the main evaluation criterium. The residual ratio is the ratio between the aggregated FIB size and the original FIB size.

We first show the residual ratio with different topology sizes. In Fig. 7(a), we see that Single-Nexthop FIB aggregation can reduce the FIB sizes to 60%. Using NS-FIB Aggregation by LFC, the aggregated FIB size is only 0.45%-2.88% of the original FIB size. Even by DSC, our scheme still can achieve a residual ratio of less than 20% (12.43% to 19.28%).

In Fig. 7(b), we set the average degree of the topologies to four. Our schemes still achieve almost the same residual ratio. Notice that the residual ratio is smaller for Single-Nexthop FIB aggregation. This is not surprising as the fewer neighbors, the more prefixes with the same next hop.

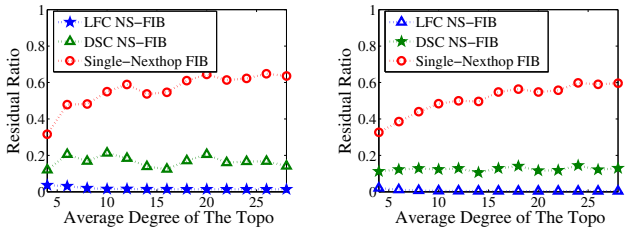
Another interesting observation is that when the topology size increases, the residual ratio slightly improves. This is because when the topology size increases, the neighbors used as next hops as the routers mapped as egress routers lay disproportionably.

We next study the impact of the average degree of the topology on the residual ratio in detail. We see in Fig. 8 that when the average degree increases, the impact of Single-Nexthop FIB aggregation becomes less significant. Again, this conforms to the intuition that the more neighbors, the less prefixes with the same next hop. The residual ratios of Single-Nexthop FIB aggregation increase above 60%. However, for our aggregation schemes, they are not affected by the average




(a) Average degree: 10. (b) Average degree: 4.

Fig. 7. Residual ratio as a function of the size of the topologies.




(a) BRITE Topo: 100 nodes. (b) BRITE Topo: 400 nodes.

Fig. 8. Residual ratio as a function of average degree.

degree. This is because if there are more neighbors, the number of selectable next hops also increases. Such property makes our scheme especially attractive.

In Fig. 9, we fix the number of selectable next hops for each prefix in DSC NS-FIB to be equal or less than $FixNum$. We see that if there are more next hops selectable, the FIB enjoys higher reduction, but the biggest jump comes from the change of a single next hop to two selectable next hops.

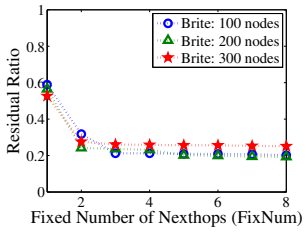



Fig. 9. Residual ratio as a function of the number of next hops.

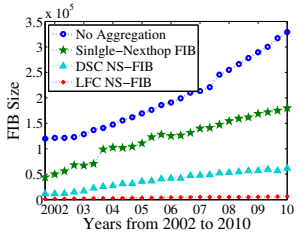



Fig. 10. The sizes of the un-aggregated and aggregated FIBs

To see the impact of FIB aggregation, we apply multiple FIB aggregation schemes to the historical routing tables of RouteViews from November 2001 to May 2010. The results are in Fig. 10. We see that if the Single-Nexthop FIB aggregation is used, the routing table size can be reduced to that of 2006. Ours, even considering the DSC implementation, can reduce the routing table size to that of 1998 (see Fig. 1).

## V. Related Work

In [14], Zhang, et. al, claimed that the most efficient way to address the Internet scalability problem is through an evolutionary path. The first step on this path is FIB shrinking.

A systematic work on four levels of FIB aggregation techniques can be found in [10]. *Level 1* removes the prefix that has the same next hop with the immediate parent in the radix tree of the FIB, which is Single-Nexthop FIB aggregation mentioned in this paper; *Level 2, 3* and *4* remove prefixes with the same next hop by packing a special prefix that can cover all of them. Several additional packing and splitting techniques are proposed in [11].

Another notable scheme is Virtual Aggregation (VA) [15], which can shrink the FIB by configuration only. VA divides the global address space into a set of virtual prefix blocks. Some Aggregation Point Routers (APR) are responsible for specific virtual prefix blocks. Other routers forward packets to the corresponding APRs. VA is incrementally deployable at ISP level. However, the configuration overhead is non-trivial and VA does not support router-level incremental deployment.

## VI. Conclusion and Future Work

In this paper, we for the first time proposed Nexthop-Selectable FIB Aggregation, which is fundamentally different from all previous FIB aggregation schemes. Most notably, each prefix has a set of selectable next hops, and prefixes with one common next hop can be aggregated. We developed the mathematical foundation of the Nexthop-Selectable FIB Aggregation problem, including problem formulation, the efficient optimal algorithms for both offline and online scenarios. Our evaluation demonstrated that our scheme achieves more than an order reduction of the FIB size.

As a first study on Nexthop-Selectable FIB Aggregation, we admit that there can be many future works. An interesting one is a more comprehensive study on path stretch control in NS-FIB aggregation. We also believe there are additional ways to construct the selectable next hops for both intra-domain and inter-domain routing.

## References

[1] BGP Routing Table Analysis Reports, http://bgp.potaroo.net.
[2] D. Meyer, L. Zhang, and K. Fall, "Report from the IAB Workshop on Routing and Addressing," RFC 4984, Sep. 2007.
[3] B. Zhang, V. Kambhampati, D. Massey, R. Oliveira, D. Pei, L. Wang, and L. Zhang, "A Secure and Scalable Internet Routing Architecture," in *Proc. ACM SIGCOMM'06 Poster*, Pisa, Italy, Sep. 2006.
[4] R. Moskowitz and P. Nikander, "Host Identity Protocol (HIP) Architecture," RFC 4423, May 2006.
[5] E. Nordmark and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6," RFC 5533, Jun. 2009.
[6] W. Wong, F. L. Verdi, and M. F. Magalhães, "A Next Generation Internet Architecture for Mobility and Multi-homing Support," in *Proc. CoNEXT*, New York, NY, Dec. 2007.
[7] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis, "Locator/ID Separation Protocol (LISP)," Internet Draft, draft-ietf-lisp-07, Apr. 2010.
[8] D. Jen, M. Meisel, H. Yan, D. Massey, L. Wang, B. Zhang, and L. Zhang, "Towards A New Internet Routing Architecture: Arguments for Separating Edges from Transit Core," in *Proc. HotNets*, Calgary, Alberta, Oct. 2008.
[9] C. Vogt, "Six/One Router: A Scalable and Backwards Compatible Solution for Provider-independent Addressing," in *Proc. MobiArch*, Seattle, WA, Aug. 2008.
[10] X. Zhao, Y. Liu, L. Wang, and B. Zhang, "On the Aggregatability of Router Forwarding Tables," in *Proc. IEEE INFOCOM*, San Diego, CA, Mar. 2010.
[11] Z. Uzmi, A. Tariq, and P. Francis, "FIB Aggregation with SMALTA," IETF Draft, draft-uzmi-smalta-00, Jul. 2010.
[12] A. Atlas and A. Zinin, "Basic Specification for IP Fast-Reroute: Loop-free Alternates," RFC 5286, Sep. 2008.
[13] Q. Li, D. Wang, M. Xu, and J. Yang, "On the Scalability of Router Forwarding Tables: Nexthop-Selectable FIB Aggregation," Dept. of Comp Sci. & Tech., Tsinghua University, Tech. Rep., Jul. 2010.
[14] B. Zhang, L. Zhang, and L. Wang, "Evolution Towards Global Routing Scalability," Internet Draft, draft-zhang-zhang-evolution-02, Oct. 2009.
[15] H. Ballani, P. Francis, T. Cao, and J. Wang, "Making Routers Last Longer with ViAggre," in *Proc. USENIX NSDI*, Boston, MA, Jun. 2009.