Computer Communications 67 (2015) 11–22

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Nexthop-Selectable FIB aggregation: An instant approach for internet routing scalability

Qing Li[a,*], Mingwei Xu[b], Dan Wang[c], Jun Li[d], Yong Jiang[a], Jiahai Yang[b]

[a] Graduate School at Shenzhen, Tsinghua University, Shenzhen, Guangdong, China
[b] Tsinghua University, Beijing, China
[c] Hong Kong Polytechnic University, Hong Kong
[d] University of Oregon, USA

ARTICLE INFO

Article history:
Received 1 August 2014
Revised 4 July 2015
Accepted 4 July 2015
Available online 11 July 2015

Keywords:
Internet routing
Routing scalability
FIB aggregation

ABSTRACT

Recently, the core net routing table is growing at an alarming speed which has become a major concern to Internet Service Providers. One effective solution is Forwarding Information Base (FIB) aggregation. All the previous studies assume every prefix has only one next hop. In this paper, we argue that a packet can be delivered to its destination by multiple selectable next hops. Based on this observation, we propose Nexthop-Selectable FIB aggregation. Prefixes, including those which originally have different next hops, are aggregated if they share one common next hop.

We provide a systematic study on this Nexthop-Selectable FIB aggregation problem. We present several practical choices to build selectable next hops for prefixes. We propose a non-trivial $O(N)$ algorithm to optimally solve the problem. We then study a generalized problem where we assign weights for different next hops to bound path stretch. We further develop an optimal incremental updating algorithm with constant running time. We evaluate our algorithms through a comprehensive set of simulations with BRITE and real world topologies. Our evaluation shows that the aggregated FIB is one order of magnitude smaller than the original one.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

The global Internet has experienced tremendous growth over the past decade. The sheer growth of user population, as well as factors including multi-homing, traffic engineering, policy routing, have driven the growth of Default Free Zone (DFZ) routing table size at an alarming rate [1,2]. The Internet Service Providers (ISPs) are forced to upgrade their routers in an unanticipated pace, which leads to sharp increase in the cost of packet forwarding. Even the large ISPs cannot afford to upgrade all their routers [3,4]. A few ISPs have even resorted to filtering out some small prefixes (mostly /24), which implies that parts of the Internet may not have reachability to each other [5]. 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 make a tradeoff between the path stretch and the routing table size by designing a new fully scalable distributed addressing & routing protocol [6–8]. Another set of proposals is to protect the core net router tables from the edge network addresses by Identifier/Locator separation [9–17]. Although some of these proposals emphasize incremental deployment, none of them can work on one single router without upgrading other ones.

A more immediate solution is Forwarding Information Base (FIB) aggregation. FIB aggregation shrinks the FIB with only local router upgrade and requires no protocol change. It is compatible with other architecture solutions as well. In FIB aggregation, multiple IP prefixes can be aggregated into one prefix if two conditions are satisfied: (1) the prefixes are numerically aggregatable and (2) their next hops are the same. FIB aggregation is not new. Many techniques [18–21] are proposed in academia.

All these previous studies focus on the first condition of FIB aggregation, which is how to find the numerical aggregatable prefixes. Very commonly, these algorithms assume that every IP prefix has only one next hop in the FIB. In contrast, we make a key observation that there can be multiple selectable next hops for each IP prefix, and through any one of these next hops, the packets can be delivered to the destination. As a matter of fact, such schemes including equal-cost multipath routing (ECMP) and many multi-path routing (routing protection) schemes [22–24] naturally exist, making a selection of multiple

[*] Corresponding author. Tel.: +8618038153239.
E-mail address: li.qing@sz.tsinghua.edu.cn (Q. Li).

http://dx.doi.org/10.1016/j.comcom.2015.07.003
0140-3664/© 2015 Elsevier B.V. All rights reserved.

next hops possible. Accordingly, we propose Nexthop-Selectable FIB (NS-FIB) aggregation, through which multiple IP prefixes can be aggregated into one prefix if (1) they are numerically aggregatable; (2) they have at least one common next hop.

Example: given two FIB entries, ⟨158.0.0.0/8, a⟩, ⟨158.128.0.0/9, b⟩ where the first element is the prefix and the second element is the next hop. Though these two prefixes are numerically aggregatable, 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 of 158.0.0.0/8 to the destination; and both b and c can deliver the packets of 158.128.0.0/9 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, ⟨158.0.0/8, $\{a, b\}$⟩, ⟨158.128.0.0/9, $\{b, c\}$⟩. Thus, they can be aggregated into one entry ⟨158.0.0/8, b⟩.

To fully explore the gain of NS-FIB aggregation, many difficulties need to be addressed. First, we need to provide the approach for a router to construct the NS-FIB, where every IP prefix has a set of selectable next hops. Through these next hops, the corresponding packets can be delivered to the destinations. Second, we need to design an effective algorithm to aggregate the NS-FIB at maximum. Third, selection of a sub-optimal next hop may result in a longer path. Thus, it is necessary to control the path stretch. Fourth, an efficient incremental algorithm is required to handle the dynamical updates.

In this paper, we for the first time provide a systematic study on the aforementioned problems. Inspired by LFA [22], we introduce two principles to construct the selectable next hops in practice. Through any of these next hops, the corresponding packets are guaranteed to be delivered to the destinations. We next formulate NS-FIB aggregation as an optimization problem. We show that it can be solved by dynamic programming. As a straightforward application of dynamic programming requires exponential running time, we develop a novel algorithm with complexity of $O(N)$. We then assign weights to different next hops and develop an algorithm that bounds the path stretch. We develop an optimal incremental updating algorithm, with constant complexity, to handle dynamical routing updates. We show that our scheme is orthogonal and can coexist with the existing FIB aggregation techniques.

We evaluate our algorithms by comprehensive simulations in China Education and Research Network (CERNET) [25]. We also evaluate our algorithms through BRITE-generated [26] and real world topologies, with the routing tables from RouteViews [27]. Our evaluation shows that NS-FIB aggregation achieves more than an order of the FIB size reduction, reducing the FIB size to that of 1998. We believe our scheme, locally and incrementally deployable, can reserve sufficient time for the agreement of advanced infrastructure changes of the Internet.

2. Background and problem formulation

2.1. Background of FIB aggregation

The growth of the Internet has made it a huge concern of the industry whether the capability of the router can match the increasing demand in Internet scalability. Even if the emerging advanced routers can match the demand, the ISPs cannot afford to upgrade all their routers, some of which are more than ten yeas old [3,4]. Among multiple solutions, a shrinking of the routing table is an immediate approach. In each router, there are two types of global routing tables, the Routing Information Base (RIB) and the Forwarding Information Base (FIB). The RIB stores the route information, including the path parameters and other attributes. When there are routing updates from BGP, the RIB will be updated. The RIB and the intra-domain routing table generate the FIB, which is stored in line cards. In the FIB, generally, every IP prefix has only one next hop. When a corresponding packet arrives, it will be forwarded to this next hop. With such a specific task (packet forwarding), the FIB uses the memory in high performance yet at a high price. This makes the FIB the core bottleneck of the Internet routing scalability.

Table 1
The notation list.

Notation	Definition
$\mathcal{F}$	The default Nexthop-Selectable FIB
$\mathcal{P}$	The set of prefixes in $\mathcal{F}$
N	The number of entries in $\mathcal{F}$, i.e., $\lvert\mathcal{P}\rvert$
$\mathcal{F}_{aggr}$	An NS-FIB aggregation for $\mathcal{F}$
$\mathcal{T}(\mathcal{V}, \mathcal{E})$	The NS-FIB tree of $\mathcal{F}$, $\mathcal{V} = \mathcal{P}$
T	A certain subtree (or aggregation cell) in $\mathcal{T}$
R_T	The root of the tree T
$\mathcal{C}_T$	The set of children of T in $\mathcal{T}$
p	A certain prefix of $\mathcal{F}$
x	A certain next hop for some prefix
$\mathbf{G}(T)$	T's optimal aggregation size
$\mathbf{G}_x(T)$	T's opt. aggr. size with x selected by R_T
$\mathcal{S}_p$	p-rooted branch of $\mathcal{T}$
$\mathcal{U}_x(p)$	p-rooted x-selectable aggr. cell set in $\mathcal{T}$
T_x^*	The opt. R_T-rooted x-selectable aggr. cell

Although FIB aggregation cannot shrink the RIB in RAM, it can reduce the entries in the TCAM, which is quite expensive and generally the bottleneck of old routers. Besides, FIB aggregation requires a pure local upgrade, with no change to routing protocols or router hardware. In contrast to a replacement, FIB aggregation can co-exist with other architectural solutions as well. All these make FIB aggregation attractive to industry and academia alike.

Nexthop-Selectable FIB (NS-FIB) aggregation is fundamentally different from previous FIB aggregation schemes. There are two levels of next hops for each IP prefix. For intra-domain routing, there can be multiple selectable next router hops towards the egress router of the current AS. For inter-domain routing, there can be multiple selectable next AS hops towards the destination (although involving policy problems). The mathematical foundation of our scheme (i.e., the aggregation algorithm) is applicable to both. However, further study is required before applying our algorithm in the inter-domain case, which would change routing protocols and the AS-level routing behavior (causing commercial issues). Therefore, we concentrate on applying our scheme in the intra-domain case and keep the inter-domain case as a possible future work.

We also delay the detailed discussion on constructing the NS-FIB in Section 4. We would like to comment that our work does not depend on specific approaches of constructing the set of selectable next hops. In what follows, we focus on how to aggregate the NS-FIB with a set of selectable next hops for each prefix.

2.2. The Nexthop-Selectable FIB aggregation problem

Although NS-FIB is compatible with some existing complex FIB aggregation algorithms (see Section 5), we first mainly focus on computing the minimized aggregated FIB with the restriction that no new prefix is generated.

Let a *Nexthop-Selectable FIB (NS-FIB)* be a set $\mathcal{F} = \{\langle p, \mathcal{A}_p\rangle | p \in \mathcal{P}\}$, where $\mathcal{P}$ is the prefix set ($|\mathcal{P}| = N$) and $\mathcal{A}_p$ is the set of selectable next hops for p. A feasible aggregation for $\mathcal{F}$ is a set $\mathcal{F}_{aggr} = \{\langle p, a_p\rangle | p \in \mathcal{P}', \mathcal{P}' \subseteq \mathcal{P}$ and $a_p \in \mathcal{A}_p\}$ where: for any IP address and its longest matching entries $\langle p, \mathcal{A}_p\rangle$ in $\mathcal{F}$ and $\langle p', a_{p'}\rangle$ in $\mathcal{F}_{aggr}$, we have $a_{p'} \in \mathcal{A}_p$. The **Nexthop-Selectable FIB aggregation problem (NS-FIB aggregation)** is: given a NS-FIB $\mathcal{F}$, find a feasible aggregation $\mathcal{F}_{aggr}$ for $\mathcal{F}$ with the minimized $|\mathcal{F}_{aggr}|$. The notations used in the paper are summarized in Table 1.

A NS-FIB is shown in Fig. 1. Two corresponding feasible aggregations are shown in Fig. 2, and only *Aggr. 1* is an optimal aggregation for the NS-FIB of Fig. 1.

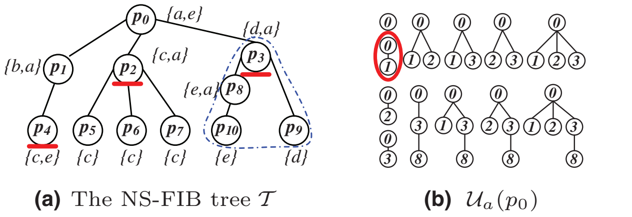
$$\left\{\begin{array}{lll} <p_0=/0,\ \{a, e\}> & <p_5=01100/5,\ \{c\}> & <p_1=001/3,\ \{b, a\}> \\ <p_6=01101/5,\ \{c\}> & <p_2=011/3,\ \{c, a\}> & <p_7=01110/5,\ \{c\}> \\ <p_3=101/3,\ \{d, a\}> & <p_8=1010/4,\ \{e, a\}> & <p_4=0011/4,\{c,e\}> \\ <p_9=10111/5,\ \{d\}> & <p_{10}=101011/6,\{e\}> & \end{array}\right\}$$

Fig. 1. An example of Nexthop-Selectable FIB.

$$Aggr.\ 1\ \left\{<p_0, a> <p_2, c> <p_3, d> <p_4, c> <p_8, e>\right\}$$

$$Aggr.\ 2\ \left\{\begin{array}{c} <p_0, a>\quad <p_4, c>\quad <p_5, c>\quad <p_{10}, e> \\ <p_6, c>\quad <p_7, c>\quad <p_9, d> \end{array}\right\}$$

Fig. 2. Two feasible aggregations for NS-FIB in Fig. 1.



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

Fig. 3. (a) Corresponds to the NS-FIB in Fig. 1. (b) $\mathcal{U}_a(p_0)$, the set of p_0-rooted a-selectable cells of $\mathcal{T}$.

3. Algorithm design

In this section, we first solve the above formulated problem. The construction of NS-FIB and the compatibility with other FIB aggregation algorithms (including ORTC [18]) will be discussed in Sections 4 and 5.

3.1. A dynamic programming solution

The routing table is generally organized as a radix tree. We follow this convention but compress the radix tree by removing nonexist prefixes. For the prefix set $\mathcal{P}$ of $\mathcal{F}$, a corresponding NS-FIB tree $\mathcal{T}(\mathcal{V}, \mathcal{E})$ can be constructed ($\mathcal{V}$ is the prefix set of $\mathcal{F}$). By abusing notations, we use p to denote a node of $\mathcal{T}$; $\forall p, p' \in \mathcal{V}$, 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. 3(a) shows the tree corresponding to the NS-FIB in Fig. 1.

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

We first design an optimal sub-structure for our problem. 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 and $\mathbf{G}_\mathbf{x}(T)$ 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 $\mathbf{G}(T) \le \mathbf{G}_\mathbf{x}(T)$ and

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

We will prove that $\mathbf{G}_\mathbf{x}(T)$ 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. 3). Let $\mathcal{C}_{T'}$ denote the set of the immediate children of a subtree T' in $\mathcal{T}$ (See $\mathcal{C}_{T'}$ in Fig. 3). Let $\mathcal{U}_x(p)$ denote the set of p-rooted and x-selectable cells in $\mathcal{T}$ where $x \in \mathcal{A}_p$ (See $\mathcal{U}_a(p_0)$ in Fig. 3).

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

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

Intuitively, we would like to divide an 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, $\mathbf{G}_\mathbf{x}(T)$ takes the minimum of all different forms of T'. For example, in Fig. 3, 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 aggregation, based on the first division, is four (*Aggr.* 1 in Fig. 2), and the size of the aggregation, based on the second division, is six (*Aggr.* 2 in Fig. 2). 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 $\mathbf{G}_\mathbf{x}(T)$ for special cases:

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

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

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

A dynamic programming algorithm can be derived from the above optimal sub-structure. While the optimal solution can be obtained, as Lemma 1 suggests, $|\mathcal{U}_x(R_T)|$ has an exponential relation with the number of nodes $\mathcal{U}_x(R_T)$ involves (i.e., $|\mathcal{U}_a(p_0)| = 12$ in Fig. 3). Thus, if a straightforward exhaustive search is applied, the complexity of the algorithm will be exponential, which is unacceptable.

Lemma 1. *The number of R_T-rooted x-selectable cells ($|\mathcal{U}_x(R_T)|$) has an exponential relation with the number of nodes $\mathcal{U}_x(R_T)$ involves.*

Proof. Let $\hat{u}$ be the max cell in $\mathcal{U}_x(R_T)$. We can prove the Lemma by proving that $|\mathcal{U}_x(R_T)|$ has an exponential relation with the size of $\hat{u}$.

We assume that $\hat{u}$ is a *perfect binary tree* with l ($l \ge 2$) layers and m ($2^l - 1$) nodes. Let Q_i be the number of subtrees rooted at R_t in an i-layer *perfect binary treet*. The recurrent relation can be described as $Q_i = (Q_{i-1} + 1)^2, i \ge 2$ with $Q_1 = 1$. Now we prove the lemma by segment amplification and minification.

$$\begin{aligned} Q_i &= (Q_{i-1} + 1)^2 = ((Q_{i-2} + 1)^2 + 1)^2 \\ &> (Q_{i-2} + 1)^{2^2} = ((Q_{i-3} + 1)^2 + 1)^{2^2} \\ &> \cdots \\ &> (Q_1 + 1)^{2^{i-1}} = 2^{2^{i-1}}, \quad i \ge 2 \end{aligned}$$

We can see that $|\mathcal{U}_x(R_T)| > 2^{2^{l-1}} = 2^{\frac{m+1}{2}}$. Thus, $|\mathcal{U}_x(R_T)|$ has an exponential relation with the size of $\hat{u}$ and the lemma is proved. □

3.2. 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 $\mathbf{G}_\mathbf{x}(T) - 1$). A crucial challenge of calculating $\mathbf{G}_\mathbf{x}(T)$ 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 $\mathbf{G}_\mathbf{x}(T)$. OptimalCell() will become a building block of our main algorithm. In OptimalCell(), 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 children number of the branch root.

Algorithm 1 OptimalCell(x, T)

```
$T' \Leftarrow \{R_T\}$,   $\mathbf{G}' \Leftarrow 1$;
for all $p \in$ children of $R_T$ do
    if $x \in A_p$ and $\mathbf{G_x}(\mathcal{S}_p) = \mathbf{G}(\mathcal{S}_p)$ then
        $\mathbf{G}' \Leftarrow \mathbf{G}' + \mathbf{G}(\mathcal{S}_p) - 1$
        $T' \Leftarrow T' \cup (\mathcal{S}_p)^*_x$
    else   $\mathbf{G}' \Leftarrow \mathbf{G}' + \mathbf{G}(\mathcal{S}_p)$
    end if
end for
$T^*_x \Leftarrow T'$,   $\mathbf{G_x}(T) \Leftarrow \mathbf{G}'$
```

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 of 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 into T^*_x computed by OptimalCell().

Lemma 2. *The branches of an optimal x-selectable cell are also optimal x-selectable cells.*

We first explain an example in Fig. 3. $T_a = (p_0, p_1, p_3, p_8)$ is an optimal p_0-rooted a-selectable cell. As a branch of T_a, (p_3, p_8) is also an optimal p_3-rooted a-selectable cell. We now formally prove Lemma 2.

Proof. We prove this lemma by contradiction. Let T_x be an optimal x-selectable cell in $\mathcal{T}$. Let p be any prefix in T_x and T_1 be the p-rooted branch of T_x. Assume that T_1 is not an optimal x-selectable cell in $\mathcal{T}$. As such, there exists an optimal p-rooted x-selectable cell T_2 and

$$\sum_{p\in\mathcal{C}_{T_1}} \mathbf{G}(\mathcal{S}_p) > \sum_{p\in\mathcal{C}_{T_2}} \mathbf{G}(\mathcal{S}_p)$$

Let $T'_x = (T_x \setminus T_1) \cup T_2$. Then

$$\sum_{p\in\mathcal{C}_{T_x}} \mathbf{G}(\mathcal{S}_p) > \sum_{p\in\mathcal{C}_{T'_x}} \mathbf{G}(\mathcal{S}_p)$$

contradicting to the fact that T_x is optimal. □

Lemma 3. *A branch T_1 of an optimal x-selectable cell T_x can be exchanged with any other optimal R_{T_1}-rooted x-selectable cell T_2, without changing the optimality of T_x.*

Before proving this lemma, we discuss an example in Fig. 3 first. $T_a = (p_0, p_1, p_3)$ is an optimal p_0-rooted a-selectable cell. (p_3, p_8) is an optimal p_3-rooted a-selectable cell. By exchanging the branch (p_3) of T_a with (p_3, p_8), T_a is transformed into (p_0, p_1, p_3, p_8) and remains to be an optimal a-selectable cell according to the definition of optimality. We now formally prove Lemma 3.

Proof. Let $T'_x = (T_x \setminus T_1) \cup T_2$, $C_x = \mathcal{C}_{T_x}$, $C'_x = \mathcal{C}_{T'_x}$, $C_1 = \mathcal{C}_{T_1}$ and $C_2 = \mathcal{C}_{T_2}$. We can see that $C'_x = (C_x \setminus C_1) \cup C_2$. Now we prove that T'_x is an optimal x-selectable cell.

T_1 is an optimal x-selectable cell by Lemma 2.

$$\sum_{p\in C_1} \mathbf{G}(\mathcal{S}_p) = \sum_{p\in C_2} \mathbf{G}(\mathcal{S}_p) = \mathbf{G_x}(\mathcal{S}_{R_{T_1}}) - 1$$

$$\begin{aligned}\sum_{p\in C_x} \mathbf{G}(\mathcal{S}_p) &= \sum_{p\in C_x\setminus C_1} \mathbf{G}(\mathcal{S}_p) + \sum_{p\in C_1} \mathbf{G}(\mathcal{S}_p)\\ &= \sum_{p\in C_x\setminus C_1} \mathbf{G}(\mathcal{S}_p) + \sum_{p\in C_2} \mathbf{G}(\mathcal{S}_p)\\ &= \sum_{p\in (C_x\setminus C_1)\cup C_2} \mathbf{G}(\mathcal{S}_p)\\ &= \sum_{p\in C'_x} \mathbf{G}(\mathcal{S}_p)\end{aligned}$$

As T_x is an optimal cell, T'_x is optimal as well. □

Theorem 4. *OptimalCell() computes an optimal R_T-rooted x-selectable cell and $\mathbf{G_x}(T)$.*

Proof. Based on Lemmas 2 and 3, we prove that the cell T^*_x computed by OptimalCell() is optimal by transforming any optimal R_T-rooted x-selectable cell T_x into T^*_x.

Let C_1 be the children set of R_T in T^*_x. Let C_2 be the children set of R_T in T_x. $C_1 \subseteq C_2$, which can be proved by contradiction as follows.

Assume that $\exists \bar{p} \in C_1$ and $\bar{p} \notin C_2$. According to OptimalCell(), $\mathbf{G}_x(\mathcal{S}_{\bar{p}}) = \mathbf{G}(\mathcal{S}_{\bar{p}})$. Let T' be any optimal $\bar{p}$-rooted x-selectable cell. Let $T'_x = T_x \cup T'$.

$$\begin{aligned}\sum_{p\in\mathcal{C}_{T_x}} \mathbf{G}(\mathcal{S}_p) &= \sum_{p\in\mathcal{C}_{T_x}\setminus\{\bar{p}\}} \mathbf{G}(\mathcal{S}_p) + \mathbf{G}(\mathcal{S}_{\bar{p}})\\ &= \sum_{p\in\mathcal{C}_{T_x}\setminus\{\bar{p}\}} \mathbf{G}(\mathcal{S}_p) + \mathbf{G}_x(\mathcal{S}_{\bar{p}})\\ &= \sum_{p\in\mathcal{C}_{T_x}\setminus\{\bar{p}\}} \mathbf{G}(\mathcal{S}_p) + 1 + \sum_{p'\in\mathcal{C}_{T'}} \mathbf{G}(\mathcal{S}_{p'})\\ &= 1 + \sum_{p\in\mathcal{C}_{T'_x}} \mathbf{G}(\mathcal{S}_p)\end{aligned}$$

which contradicts the fact that T_x is optimal x-selectable cell. Thus, $C_1 \subseteq C_2$.

$\forall p \in C_2$, two cases exist: if $p \in C_1$, the p-rooted branch of T_x can be replaced by the p-rooted branch of T^*_x without changing the optimality of T_x; if $p \notin C_1$, the p-rooted branch of T_x can be removed without changing the optimality of T_x. Below are the details of these two cases.

Case One: $p \in C_2$ and $p \in C_1$. Let T_1 be the p-rooted branch of T^*_x. According to OptimalCell(), T_1 is an optimal p-rooted x-selectable cell. Let T_2 be the p-rooted branch of T_x. By Lemma 3, T_2 can be replaced by T_1 without changing the optimality of T_x.

Case Two: $p \in C_2$ but $p \notin C_1$. According to OptimalCell(), $\mathbf{G_x}(\mathcal{S}_p) \neq \mathbf{G}(\mathcal{S}_p)$, thus $\mathbf{G_x}(\mathcal{S}_p) > \mathbf{G}(\mathcal{S}_p)$. T_2 is an optimal p-rooted x-selectable cell by Lemma 2, thus

$$\sum_{p'\in\mathcal{C}_{T_2}} \mathbf{G}(\mathcal{S}_{p'}) = \mathbf{G_x}(\mathcal{S}_p) - 1 \ge \mathbf{G}(\mathcal{S}_p), \quad \text{and}$$

$$\begin{aligned}\sum_{p'\in\mathcal{C}_{T_x}} \mathbf{G}(\mathcal{S}_{p'}) &= \sum_{p'\in\mathcal{C}_{T_x\setminus T_2}} \mathbf{G}(\mathcal{S}_{p'}) - \mathbf{G}(\mathcal{S}_p) + \sum_{p'\in\mathcal{C}_{T_2}} \mathbf{G}(\mathcal{S}_{p'})\\ &\ge \sum_{p'\in\mathcal{C}_{T_x\setminus T_2}} \mathbf{G}(\mathcal{S}_{p'})\end{aligned}$$

Consequently, T_2 can be removed without changing the optimality of T_x.

Therefore, any optimal R_T-rooted x-selectable cell T_x can be transformed into T^*_x without changing the optimality, which means T^*_x is an optimal x-selectable cell. As OptimalCell() computes the value of $\mathbf{G_x}(T)$ according to T^*_x, the result can be guaranteed to be correct. □

Algorithm 2 NS-FIB-aggregation(T)

```
1:  for all $p \in V_T$ (post order) do
2:      $\mathbf{G}(\mathcal{S}_p) \Leftarrow \infty$
3:      for all $x \in A_p$ do
4:          Compute $(\mathcal{S}_p)^*_x$, $\mathbf{G_x}(\mathcal{S}_p)$ 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
```

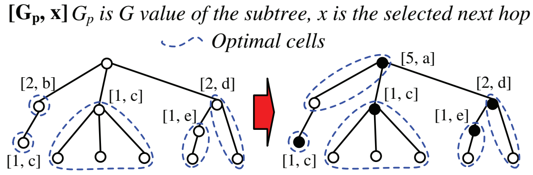


Fig. 4. The last round of NS-FIB-aggregation() to generate an optimal aggregation for the NS-FIB in Fig. 3.

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 formulas (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.

We use the NS-FIB $\mathcal{T}$ in Fig. 3 as the input to illustrate the algorithm of NS-FIB-aggregation(). In Fig. 4, $\forall p \in \mathcal{T}$, 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. 2.

Theorem 5. *NS-FIB-aggregation() computes an optimal (minimized) aggregation for the input NS-FIB.*

Proof. This can be recursively proved according to formula (1) and Theorem 4. □

Theorem 6. *Complexity of NS-FIB-aggregation is $O(N)$.*

Proof. Let C_1 be the running time of operations of *line 2* in NS-FIB-aggregation(). Let C_2 be the running time of the **if** block from *line 5* to *line 9*. C_1 and C_2 are both constants. The time complexity is

$$\begin{aligned}
&\sum_{p\in V_T}\{C_1 + O(m)[\Theta(|C_T(p)|) + C_2]\}\\
&\quad = \sum_{p\in V_T}[O(m)\Theta(|C_T(p)|) + O(m)]\\
&\quad = O(m)\Theta(|V_T|) + O(m)\Theta(|V_T| - 1)\\
&\quad = O(m|V_T|) = O(mN) = O(N) \qquad \square
\end{aligned}$$

3.3. The incremental updating algorithm

In practice, the routing table changes dynamically. A BGP route change may trigger route withdrawal, update or insertion in the NS-FIB. We use UPDATE as an example. The operations of handling a BGP update include: (1) update entry in the RIB, (2) if the optimal BGP route changes, compute the new next hop(s) for this prefix (i.e., generate a new FIB entry), and (3) update the FIB change in the line card. The FIB aggregation algorithms fall between (2) and (3). The bottleneck of the above operations is (1), which has complexity of $O(\log N)$.

Algorithm 3 NS-FIB-Update(T, $\langle p, A_{new}\rangle$)

```
A ⇐ $(A_{new} \cup A_{old}) \setminus (A_{new} \cap A_{old})$
for all $x \in A$ do
    Compute $(\mathcal{S}_p)^*_x$ and $\mathbf{G_x}(\mathcal{S}_p)$ by OptimalCell($x$, $\mathcal{S}_p$)
    Update $\mathbf{G}(\mathcal{S}_p)$, the selected next hop and Aggr. children
end for
while $p$ is not the root of $T$ do
    $p' \Leftarrow p$,   $p \Leftarrow$ the father of $p$
    for all $x \in A \cap A_p$ do
        Update $\mathbf{G_x}(\mathcal{S}_p)$ according to $\mathbf{G_x}(\mathcal{S}_{p'})$ and $\mathbf{G_x}(\mathcal{S}_{p'})$
        Update $\mathbf{G}(\mathcal{S}_p)$, the selected next hop, etc.
    end for
end while
```

We develop Algorithm NS-FIB-Update(). Given an update $\langle p, A_{new}\rangle$ where p is a prefix and A_{new} is the new set of next hops of p, NS-FIB-Update() recalculates $\mathbf{G}$ values of the node p and all the upstream nodes.

Theorem 7. *If the original aggregation is optimal, NS-FIB-Update() computes an optimal aggregation for the Nexthop-Selectable FIB after the update.*

Proof. Given an optimal aggregation and the corresponding $\mathbf{G}$, $\mathbf{G_x}$ are stored in the tree before the update. According to Theorem 4, OptimalCell() computes an optimal p-rooted x-selectable cell and $\mathbf{G_x}(\mathcal{S}_p)$ for every changed next hop x of p. Thus, NS-FIB-Update() computes $\mathbf{G}(\mathcal{S}_p)$. For every ancestor p' of p in $\mathcal{T}$, $\mathbf{G}(\mathcal{S}_{p'})$ and $\mathbf{G}(\mathcal{S}_{p'})$ are updated according to formula 1. Thus, NS-FIB-Update() is the same as NS-FIB-aggregation() on the updated NS-FIB; and NS-FIB-Update() computes an optimal aggregation for the Nexthop-Selectable FIB after the update. □

Assume that the probability of updating each prefix is the same, next we prove the average complexity of NS-FIB-Update() is constant. This is practically significant as it shows that NS-FIB-Update() is negligible compared with other operations in a BGP update (e.g., a search of the RIB entry is $O(\log N)$). There are two dominated steps in NS-FIB-Update(). The first is recalculation of $\mathbf{G}$, $\mathbf{G_x}$ for the node of the updated prefix. This is determined by the number of the node's children and, on average, such an operation is constant. The second is the re-computation from p upstream to the root of the tree. Notice that the depth of the NS-FIB tree is 32 (IPv4) at maximum. Thus, this operation is also constant. We formally prove the average complexity as follows.

Lemma 8. *Given any tree T, the average number of a node's children in T is $(N-1)/N$, which is less than one.*

Proof. Let N be the node number in T and T has $N-1$ edges. The average degree of the tree T is $2 \times (N-1)/N$, thus average children number is $(N-1)/N\ (\leq 1)$. □

Theorem 9. *Assume that the probability of updating each prefix is the same, the complexity of NS-FIB-Update() is, in expectation, constant.*

Proof. The average complexity of OptimalCell() is $O(1)$ according to Lemma 8. Thus, the complexity of the first **for** loop is $O(m)$, as the largest number of the changed next hops of p ($|A|$ in the algorithm) is no more than m. The complexity of the second **for** loop is also $O(m)$. The rounds of the **while** loop are no more than the largest layer number of $\mathcal{T}$, which can be regarded as a constant. Thus, the complexity of the **while** loop is $O(m)$. Since m is a constant, the complexity of NS-FIB-Update() is $O(1)$. □

4. Construction of Nexthop-Selectable FIB

In the current Internet routing system, the FIB of a router is generally generated by three steps: (1) compute the intra-domain routing table; (2) select the optimal BGP route for each prefix. Every BGP route has one single BGP next hop, which is attached to the egress router of the given AS and visible to the intra-domain routing system. Therefore, every prefix corresponds to one single egress router (or an optimal BGP next hop) in the given AS; (3) allocate the next hop to the prefix according to the corresponding egress router (BGP next hop) of the selected optimal BGP route. Although we can select multiple BGP routes for each prefix, this inter-domain method changes the routing protocol and AS-level behavior, involving commercial issues. Therefore, we select the intra-domain approach to construct the NS-FIB: change the first step by computing multiple selectable next hops to each intra-domain router. A set of selectable next hops can be generated for each prefix according to its egress router (or the optimal BGP next hop). Our NS-FIB construction approach only changes intra-domain routing and has no impact on inter-domain routing, as the packet will still be delivered to the original egress router after NS-FIB aggregation.

If any multi-path or backup path protocols are used, it is straightforward to use the multiple next hops built by these protocols. Here we present schemes that require only local information and no modification to the existing Internet infrastructure. Our idea is inspired by loop-free alternates (LFA) [22]. We use *loop-free condition (LFC)* and *downstream condition (DSC)* to construct NS-FIB.

LFC Nexthop-Selectable FIB construction: Let Dt be the intra-domain destination for the prefix p (i.e., the egress router of the optimal BGP route). For a router Rt, a neighbor NG_i of Rt meets *LFC Condition* for Dt iff Rt is not on the optimal route(s) from NG_i to Dt. The optimal next hop(s) always meet(s) LFC condition. As one single optimal BGP route is selected for each prefix, each prefix has one corresponding egress router, to which Rt has multiple selectable next hops according to LFC. Therefore, 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 uses information of the local router only.

Lemma 10. *Let Rt construct the set of selectable next hops according to LFC condition. Assuming 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 want to selectively deploy our scheme for their most aged routers, LFC is recommended.

DSC Nexthop-Selectable FIB construction: Given a router Rt and a destination prefix Dt, a neighbor NG_i of Rt meets *DSC Condition* iff 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. As one single optimal BGP route is selected for each prefix, each prefix has one corresponding egress router, to which Rt has multiple selectable next hops according to DSC. Therefore, 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 by DSC condition.

Lemma 11. *If every router constructs the selectable next hops by DSC condition, no forwarding loop exists.*

The rigorous proof of the Lemmas 10 and 11 can be found in [22]. Fig. 5 shows an example of LFC and DSC. Assume that all the other routers except R select the optimal routes. If R constructs the NS-FIB according to LFC, N_1 (the optimal), N_2 and N_3 are all selectable next hops for the prefix 111/3. Because N_2 and N_3 will forward the packet to 111/3 by N_1, thus no loop exists. For DSC Condition, only N_1 and N_2 are selectable next hops for the destination 111/3. The monotonicity that DSC Condition contains can guarantee that no loop exists even all the routers select multiple next hops according to DSC.

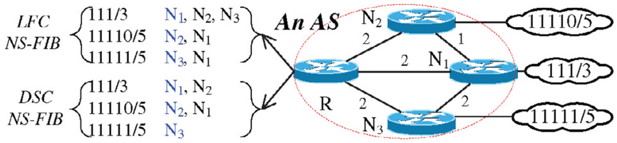
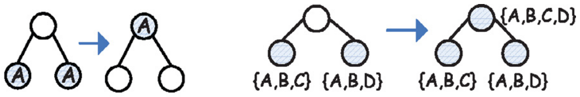


Fig. 5. An example of LFC/DSC NS-FIB.



(a) General form. **(b)** NS-FIB form.

Fig. 6. Level 2 compatibility with NS-FIB aggr.

Both the LFC and DSC NSFIB construction methods strictly follow the current Internet intra-domain routing protocol OSPF and can be implemented directly.

5. Compatibility with the existing FIB aggregation approaches

There are many FIB aggregation schemes to date. For example, Zhao et al. proposed several aggregation techniques by packing new entries into the FIB [19]. In this section, we take *Level 2 aggregation* of ref. [19] as an example to show how the *packing* technique can be compatible with our Nexthop-Selectable FIB aggregation. The choice of Level 2 aggregation is not special. We emphasize again that intrinsically, our scheme allocates multiple next hops for each IP prefix and is thus orthogonal to all the previous schemes.

The basic idea of Level 2 aggregation is illustrated in Fig. 6(a). The sibling prefixes with the same next-hop are combined into a packed parent prefix (originally nonexistent). Level 2 aggregation in NS-FIB is illustrated in Fig. 6(b). Given two sibling prefixes with a common next hop(s), the parent prefix is packed into the NS-FIB and its selectable next-hops include all the selectable next-hops of the two sibling prefixes. The two sibling prefixes are temporarily not removed. In the aggregation algorithm we proposed, they will be aggregated in an optimal way. The aggregation result of the new NS-FIB is better than the original one without the packed prefix.

Besides, NS-FIB not only allows Level-3 and Level-4 aggregations [19], but also increases the aggregation probability. Because the probability that two prefixes share a common selectable next hop is higher than the probability that two prefixes have the same optimal next hop. Therefore, Level-3 and Level-4 aggregations are more possible to occur in NS-FIB aggregation. By setting the union of the next hop sets of the two real prefixes sharing a common selectable next hop as the next hop set of their grandfather node ([19]), Level-3 and Level-4 are introduced. Other packing FIB aggregation techniques, including ORTC [18] and SMALTA[20], can be introduced as well. Besides, the method of parallelization, employed by MMS [28] (local deployment) to accelerate ORTC, can also be used to accelerate the computation of NS-FIB aggregation. Locality-aware FIB aggregation [29] aims at accelerating the updating process of FIB aggregation algorithms.[1] It aggregates stable parts of the FIB while keeping the less stable ones untouched. This approach can also be used to accelerate the updating process of NS-FIB aggregation. However, it inevitably sacrifices the compression performance.

[1] Although Locality-aware FIB aggregation is abbreviated as LFA, it has no relation with Loop-Free Alternates (also abbreviated as LFA), which is used in our scheme to generate multiple selectable next hops (please refer to Section 4). In the whole paper, LFA always refers to Loop-Free Alternates.

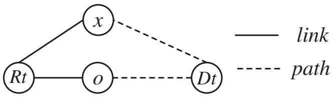


Fig. 7. The weight of a next hop x.

6. Impact on intra-domain traffic

6.1. Path stretch control for NS-FIB aggregation

With NS-FIB aggregation, a packet may be forwarded along a non-optimal path (note that the packet will be delivered to the destination). This will cause intra-domain path stretch, resulting in higher bandwidth consumption and traffic delay. Even though the current Internet is bandwidth redundant, we would like to bind the path stretch for each individual packet.

The global path stretch is related to all the intra-domain forwarding steps. However, FIB aggregation is achieved by the router individually. In order to maintain the router-level incremental deployability, we need to find a method for the router to calculate the path stretch locally.

Formally, we assign a weight $w_x(p)$ to each next hop $x \in \mathcal{A}_p$ of a prefix p in the router Rt. Let Dt be the destination (egress router) of p. Let $plen(z_1, z_2)$ be the optimal path length from z_1 to z_2. Let $linklen(z_1, z_2)$ be the length of $link(z_1, z_2)$. Let the optimal next hop from Rt to Dt be o. In the FIB of Rt,

$$w_x(p) = \frac{linklen(Rt, x) + plen(x, Dt) - plen(Rt, Dt)}{linklen(Rt, o)}$$

where the numerator is the *one-step path stretch* caused by Rt's forwarding action. We define *path stretch* of prefix p as $PathStr(p) = w_x(p)$ where x is the selected next hop after the aggregation. An example can be found in Fig. 7. We define the *network path stretch* after NS-FIB aggregation as the maximum path stretch of all prefixes, i.e., $\max_{\forall p \in \mathcal{F}_{aggr}} PathStr(p)$, which actually reflects the worst case. Our **Weighted Nexthop-Selectable FIB aggregation (WNS-FIB aggregation)** problem is to minimize $|\mathcal{F}_{aggr}|$ while the network path stretch is bounded by a user-required threshold $\mathcal{D}$.

WNS-FIB aggregation emphasizes on locally bounding the path stretch of individual packets. We admit that there are many possible definitions of network path stretch. For example, the network path stretch can be a summation of the path stretches of all the prefixes. We put a comprehensive study on various WNS-FIB aggregation into our future work.

To solve the WNS-FIB aggregation problem, we simply eliminate the next hop with weight greater than the $\mathcal{D}$ for each prefix. Then we apply NS-FIB-aggregation( ) to the remaining NS-FIB.

6.2. Traffic unpredictability

In NS-FIB aggregation, a router is not aware of other routers' aggregation results, which leads to traffic unpredictability. The traffic unpredictability might be an obstacle for some commercial demands, e.g., traffic engineering (TE). Multi Protocol Label Switching (MPLS) is widely deployed in the current ISP networks [30] for TE.

Steps: MPLS-TE is employed when an ISP wants to assign a specific path for some packets. In order to achieve this objective, the following steps are required:

- Setting up the MPLS tunnel along a specific path. NS-FIB aggregation does not affect the process.
- Making the decision of routing: MPLS or FIB. The router might make the decision according to the packet's upstream AS, type of service (ToS), quintuple (source IP/port, destination IP/port and the protocol type), etc. These cases are not affected by NS-FIB aggregation. However, there is **one case** that NS-FIB makes some differences. That is, the network provider might want to shift part of the traffic between R_s and R_d to an MPLS tunnel if it detects congestion. In this case, the traffic unpredictability might become an obstacle. The shifting might fail because the shifted traffic between R_s and R_d may not follow the optimal path (with NS-FIB deployed).
- Forwarding the packet by label switching. FIB will not be used for these packets. Therefore, NS-FIB aggregation does not affect the process either.

Table 2
Parameters of BRITE topologies.

Mode	Model	HS	LS	# Nodes
Router only	Waxman	1000	100	100–1400
links/node	α / β	NP	Growth type	
2–15	0.15 / 0.2	Random	Incremental	

Impact: The case that NS-FIB affects TE is acceptable. Because with NS-FIB deployed, the traffic will be dispersed among all the possible paths (not only the optimal ones), which decreases the possibility of congestion on a certain optimal path.

Solution: To further solve this problem, we provide the approach based on popular prefixes: controlling most traffic by fewer popular prefixes and aggregating most prefixes carrying less traffic. In ref. [31], Gadkari et al., measured the traffic of two Tier-1 ISPs. The results show that 0.6% (1851/292851) of the prefixes carry 80% of the traffic. Accordingly, we divide the aggregation procedure into two steps. First, we select a certain number (5% of the total) of popular prefixes. Among these popular prefixes, only the optimal next hops can be selected. Second, we only apply NS-FIB aggregation to the remaining prefixes. Therefore, the major traffic (>80%) will not be affected by NS-FIB aggregation, which controls the traffic unpredictability to an acceptable level. In the following section, Fig. 12 shows the simulation results.

7. Simulation

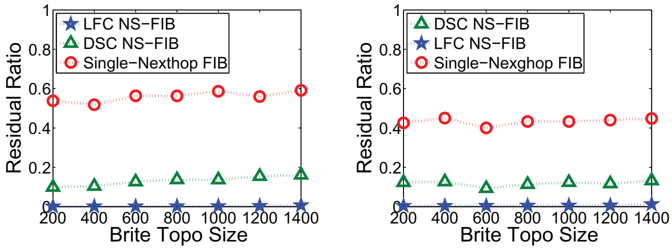
7.1. Simulation setup

We evaluate our algorithms by comprehensive simulations. To construct the NS-FIB, both the RIBs and topologies are required. We first generate diverse topologies by BRITE [26]. The node number ranges from 100 to 1400. The parameter m (links/new node) range from 2 to 15. Note that in BRITE, m indicates the average degree of the generated topology is $2m$. We set the propagation delay as the link cost. Other parameters are listed in Table 2. The default topology size is 100 nodes and the default average degree is 10 (i.e., $m = 5$). We use the RIB on May 16th, 2010 from RouteViews [27], 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 BRITE topology nodes. The next router hops can thus be computed for each prefix by using these egress routers as the destinations.

We use LFC and DSC to construct the set of selectable next hops for a prefix in the RIB. We compared our algorithms with a state-of-the-art Single-Nexthop FIB aggregation scheme (*Level 1* in [19]), i.e., every prefix has only one optimal next hop. We show the performance of the combination of our scheme and the previous aggregation techniques as well.

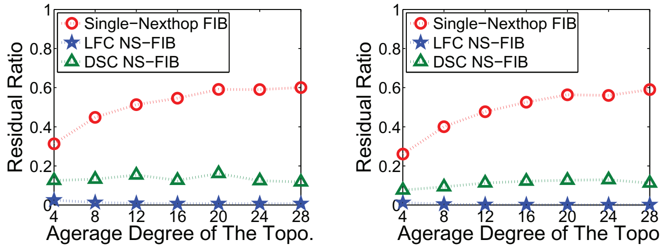
7.2. Simulation results

We use the residual ratio as the main evaluation criterion. The residual ratio is the ratio between the aggregated FIB size and the original FIB size. Note that any prefix might be aggregated as long as



(a) Average degree: 20. **(b)** Average degree: 8.

Fig. 8. Residual ratio as a function of topology size.



(a) BRITE Topo: 200 nodes. **(b)** BRITE Topo: 1000 nodes.

Fig. 9. Residual ratio as a function of average degree.

it has one common next hop with the immediate parent prefix in the trie. As the set of the selectable next hops of each prefix depends on the topology, we first show the performance of NS-FIB aggregation in diverse topologies.

7.2.1. Offline results

We first show the residual ratio with different topology sizes. In Fig. 8(a), Single-Nexthop FIB aggregation can reduce the FIB sizes to 60%. However, with NS-FIB aggregation scheme (by LFC), the aggregated FIB size is only 0.12–0.69% of the original FIB size. Even if the selectable next hops are constructed by DSC, our scheme can still achieve a residual ratio of less than 17% (9.89–16.10%), which is a four-fold improvement to that of the current state-of-the-art FIB aggregation scheme.

In Fig. 8(b), we set the average degree of the topologies to be eight. We find 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 a router has, the more prefixes having the same next hop.

Next we study the effect of the topology's average degree on the residual ratio in detail. Fig. 9 shows that when the average degree (indicating the average number of neighbors) increases, the impact of Single-Nexthop FIB aggregation becomes less significant. Again, this conforms to the intuition that the more neighbors a router has, the less number of prefixes having 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 degree. The reason is that more neighbors means more selectable next hops. Such property makes our scheme especially attractive.

In Fig. 10, we set an upper limit (*FixNum*) of the selectable next hop number for each prefix in NS-FIB. We find that if there are more selectable next hops, the FIB enjoys higher reduction. But the biggest jump comes from the change of a single next hop to two selectable next hops. The change from two to three next hops also contributes a notable reduction. This can be explained as follows: (1) the prefixes with at least i selectable next hops are more than those with at least $i+1$ prefixes, which means changing *FixNum* from i to $i+1$ involves more prefixes than changing *FixNum* from $i+1$ to $i+2$; (2) the topology degree is 10 (default value), the expected probabilities of two prefix with one/two/three next hops sharing at least one common next hop are respectively 0.1/0.44/0.79. Therefore, changing *FixNum* from i to $i+1$ when $i \geq 3$ does not make a notable difference.

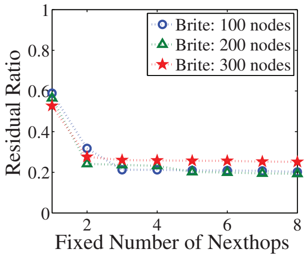


Fig. 10. Residual ratio as a function of next hop number.

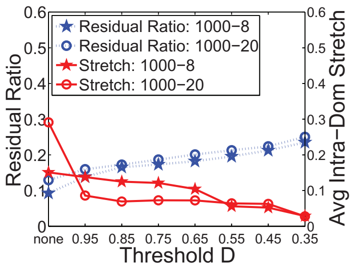


Fig. 11. Residual ratio and intra-domain stretch as a function of $\mathcal{D}$.

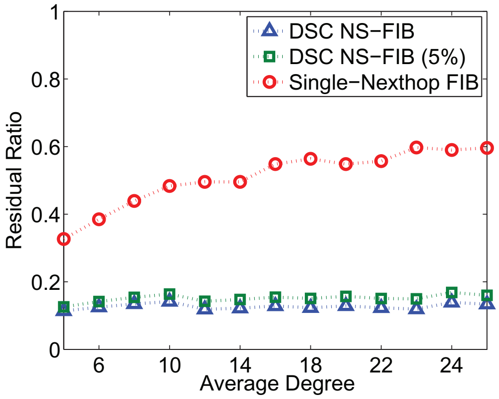


Fig. 12. NS-FIB aggregation with top 5% popular prefixes using the optimal next hops. (BRITE - 400 nodes.).

We now study the effect of path stretch. We evaluate the performance of WNS-FIB aggregation in topologies with 1000 nodes (average degree: 8 or 20), where a threshold $\mathcal{D}$ is used to bound the maximum (not the average) path stretch. We can see that when there is no bound on path stretch, the residual ratio of FIB is around 12%. When we set $\mathcal{D} = 0.35$, the residual ratio is 24%. Note that $\mathcal{D} = 0.35$ indicates that the one-step path stretch for each packet is at most 35%. Fig. 11 also shows the real average intra-domain path stretch, which is only about 3% when $\mathcal{D} = 0.35$. Clearly, WNS-FIB aggregation successfully controls the path stretch of NS-FIB aggregation.

We have also done some simulation on the efficiency of NS-FIB aggregation under the condition where the top 5% popular prefixes can only select the optimal next hops, which controls the traffic unpredictability caused by NS-FIB aggregation to an acceptable level (refer to Section 6.2). As Fig. 12 shows, the residual ratios of NS-FIB aggregation, with or without popular prefixes, stay at the same order of magnitude, although a minor decline occurs with popular prefixes. This further proves the feasibility of our scheme in the real Internet environment.

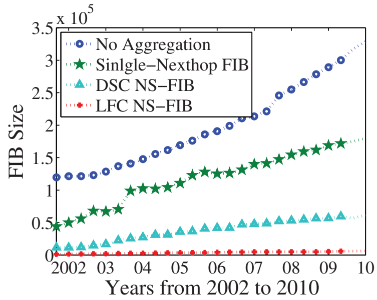
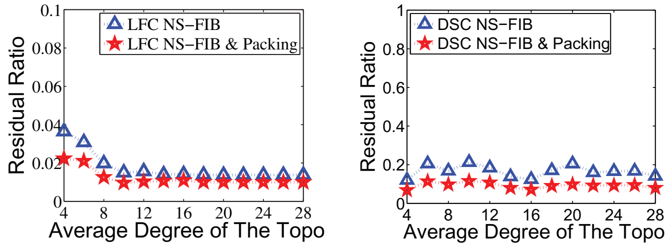


Fig. 13. (Aggregated) FIB size as a function of time.



(a) LFC: BRITE - 100 nodes **(b)** DSC: BRITE - 100 nodes

Fig. 14. The residual ratios of NS-FIB aggregation and packing NS-FIB aggregation on diverse BRITE topologies.

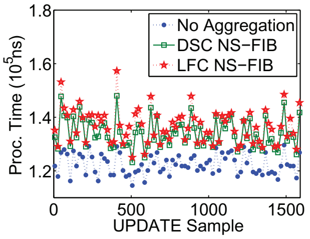


Fig. 15. Processing time of BGP UPDATEs selected by sampling.

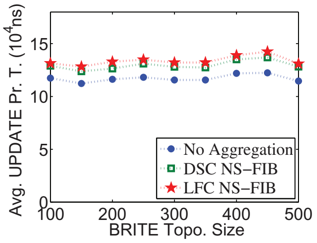


Fig. 16. Avg. processing time of UPDATES on BRITE topologies.

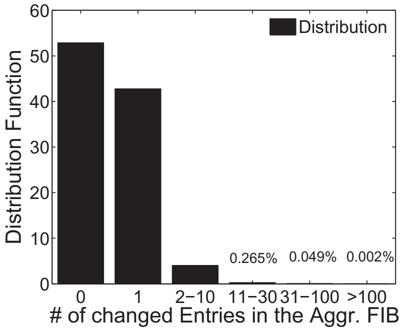


Fig. 17. Distribution of #(changed entries) caused by each FIB update.

Table 3
The characteristics of rocketfuel AS topologies.

AS Number	Name	#Routers	#Links
1221	Telstra (au)	104	151
1239	Sprint (us)	315	972
1755	Ebone (eu)	87	161
3257	Tiscali (eu)	161	328
3967	Exodus (us)	79	147
6461	Abovenet (us)	128	372

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. In Fig. 13, we find that if the Single-Nexthop FIB aggregation is used, the routing table size can be reduced to that in 2006. NS-FIB aggregation can reduce the routing table size to 1998. We therefore, believe that our scheme can reserve sufficient time for the agreement and the transition of more architecture-oriented solutions which may intrinsically address the routing scalability problem.

Finally, we study the performance of combining our scheme and the existing packing FIB aggregation. As Fig. 14 shows, packing FIB aggregation reduces the residual ratio of NS-FIB aggregation by another 25%.

7.2.2. Updating results

To evaluate the update algorithm, we use the BGP update data from May 16th to May 18th, 2010, collected from RouteViews. There are 40,306,741 route items in total during the three days, and 478,431 of them trigger NS-FIB change (these UPDATEs also change the general single-nexthop FIB). We evaluate the effectiveness of NS-FIB-Update() with these 478,431 route items.

As NS-FIB-Update() is optimal, the main concern is whether it brings computational burden to the router. We develop a program to simulate the entire BGP update process. We run our program on an Intel Core Duo CPU of 2.00 GHz with RAM 2.0 GB. In Fig. 15, we plot the processing time of BGP update with and without NS-FIB aggregation. We find that the time to handle a BGP update by LFC and DSC NS-FIB aggregation is 10–15% and 13–20% greater than the normal BGP update, respectively. In Fig. 16, we compare the expected times to handle a BGP update with different BRITE topologies. We observe a 10–12% computational overhead.

Next we show the stability of NS-FIB aggregation. In NS-FIB aggregation, one single FIB update may change multiple entries (or no entry) in the aggregated FIB. The dynamics of the aggregated FIB may cause traffic flap. Fig. 17 illustrates the stability of NS-FIB aggregation. Although some FIB updates trigger a change of more than 100 entries in the aggregated FIB, most of the FIB updates trigger no (>50%) or one-entry change in the aggregated FIB. Averagely, every FIB update triggers only 0.64 entry changes in the aggregated FIB. This indicates that the aggregated FIB of our scheme is even more stable than the original single-nexthop FIB without aggregation.

8. Verification in real-world scenarios

8.1. Scenario description

In order to further confirm the effectiveness of our scheme, we collect two sets of real-world data from Rocketfuel [32] and China Education and Research Network (CERNET) [25].

(1) We obtain six ISP topologies from Rocketfuel Project [32]. The details of these topologies are shown in Table 3. We also use the RIB from RouteViews [27], which has 328,076 entries and 37 next AS hops.
(2) CERNET is China’s first and largest national academic Internet backbone, and currently the second largest network backbone in China. The network infrastructure mainly serves the universities,

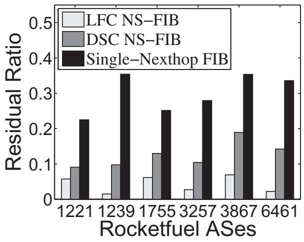


Fig. 18. Residual ratios of Single-Nexthop FIB aggr. and NS-FIB aggr. on Rocketfuel topologies.

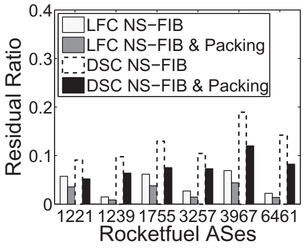


Fig. 19. The Residual ratios of packing NS-FIB aggregation on rocketfuel topologies.

institutes, colleges and schools in China. There are about 1500 universities and institutions connected and about 20 million end users. By the end of 2005, backbone bandwidth has been up to multiple 10 Gbps and regional bandwidth up to multiple 2.5 Gbps, reached 200+ cities in 31 provinces China. CERNET also has several global connection links to North America, Europe, Asia and Pacific. There are 53,012 entries in the RIB of CERNET.

8.2. Results in rocketfuel topologies

We then study NS-FIB aggregation on real world topologies from Rocketfuel Project. Fig. 18 illustrates the residual ratios of NS-FIB aggregation on different ASes. We find that NS-FIB aggregation by LFC achieves 0.92–6.1% residual ratios; NS-FIB aggregation by DSC achieves 9.1–18.2% residual ratios; and Single-Nexthop FIB aggregation achieves 22.9–34.7% residual ratios. These results conform to the results in BRITE topologies.

We again study the performance of combining our scheme and the existing packing FIB aggregation. We find in Fig. 19 the packing NS-FIB aggregation has an additional 30% gain.

8.3. Results in CERNET

We simulate our NS-FIB aggregation and Single-Nexthop FIB aggregation in CERNET. Fig. 20 shows the simulation results of ten randomly selected nodes of CERNET. We can see that the results of LFC and DSC NS-FIB aggregation are consistent with the results in Brite and Rocketfuel topologies. While Single-Nexthop FIB aggregation also achieves a notable reduction in FIB size, which is because that the average degree of CERNET topology is only about 4.33.

We also compare the results of Single-Nexthop FIB aggregation, DSC NS-FIB aggregation, packing DSC NS-FIB aggregation, LFC NS-FIB aggregation and packing LFC NS-FIB aggregation for CERNET in Fig. 21. The average residual ratios are 28.28, 15.44, 10.23, 4.66 and 2.98%, respectively. These results verify the effectiveness of our scheme.

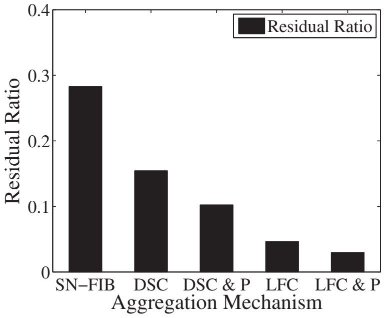


Fig. 21. The residual ratios of all the five different aggregation mechanisms on the CERNET topology.

9. Routing scalability: solutions and comparison

The essence of the routing scalability is not that the growth of the routing table might exceed the growth of hardware (Moore's Law) [3,33]; as many people believe this may not happen. To control the cost, ISPs (even the larger ones) will not replace their tens of thousands of routers every time that new hardware is in store. It has been reported that legacy routers purchased before 2000 are still in work in the production network [3,4]. Thus, the essence of the Internet routing scalability problem is the contradiction between the growth of the routing table size and the budget limit of the operation cost of the ISPs.

To handle the Internet routing scalability problem, many solutions have been proposed. From the perspective of deployment, these solutions can be classified into long-term solutions and short-term solutions. Long-term solutions aim to design fresh new architectures which are more supportive to Internet routing scalability. Nevertheless, these long-term solutions require significant transition time, as we have learned from the previous experiences of IPv6. Short-term solutions aim to shrink the routing tables based on the current Internet architecture. They are incrementally deployable and can take instant effect.

9.1. Long-term solutions

There are two broad approaches to solve the IP overloading problems, i.e., host-based ID/Loc elimination [9–12] and network-based ID/Loc separation [13–17]. These two directions both solve the routing scalability of the Internet by blocking the edge network addresses flowing into the core network.

In host-based elimination methods [9–12], the applications use an ID that is independent from the locator of the host. As such, multiple locators can be used in multi-homing and traffic engineering, etc., and no re-numbering is needed at the application layer.

In network-based ID/Loc separation [13–17], edge network addresses (IDs) are separated from the core network by encapsulating or translating the packets by core network addresses (Locs). Core network addresses are globally routable in the core network and edge

Different Aggr. Methods	Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7	Node 8	Node 9	Node 10
LFC NS-FIB Aggr.	8.35%	8.19%	10.82%	11.93%	0.51%	0.59%	4.10%	1.23%	4.22%	6.07%
DSC NS-FIB Aggr.	8.35%	8.19%	26.11%	22.46%	12.37%	16.11%	8.34%	22.77%	24.08%	22.77%
Single Nexthop FIB Aggr.	8.35%	8.19%	48.93%	30.22%	34.23%	16.55%	14.57%	30.31%	43.22%	27.85%

Fig. 20. Residual ratios of different aggregation methods (Single-Nexthop FIB and NS-FIB) on CERNET topology with 10 randomly sampled nodes.

network addresses are only locally routable in the edge network. The hosts use edge network addresses to communicate with each other and are unaware of core network addresses.

Other relevant scalable routing approaches include compact routing [6–8], geographic routing [34,35], etc. Nevertheless, these routing solutions do not directly target on the Internet.

9.2. Short-term solutions

Although these long-term proposals can solve the routing scalability problem, they require a long transition time, as we have learned from the deployment of IPv6. Thus, in [4], Khare et al. claimed that the most efficient way to address the Internet scalability problem is through an evolutionary path. The solution should be incremental at the AS level or even at the router level and can benefit the first mover without cooperation from others. The first step on this path is FIB shrinking, which aims to solve the most urgent part of the routing scalability problem and thus save enough time for the long-term solutions.

Draves et al. proposed *Optimal Routing Table Constructor* (ORTC) [18]. ORTC produces the minimized FIB that preserves the equivalent forwarding behavior by the following three steps. **Step 1**: *"Normalize"* the trie of the FIB so that each inner node has two children. The normalized FIB is equal with the original one in forwarding behavior. **Step 2**: Calculate the most prevalent next hops at every level of the trie in post order. **Step 3**: Aggregate the prefixes sharing common next hops with their immediate ancestor prefixes from top to down and generate the aggregated FIB.

Systematic work on four levels of FIB aggregation techniques can be found in [19]. *Level 1 aggregation* 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 aggregation* remove prefixes with the same next hop by packing a special prefix that can cover them all. As *Level 2, 3 and 4* all achieve extra aggregation by packing new prefixes in the FIB, we call them packing FIB aggregation technique in this paper. FIFA [36] further accelerates the updating speed of these algorithms. Besides, to accelerate the computation of ORTC, SMALTA [20], MMS [28], FIFA [36] and Locality-aware FIB aggregation [29] are proposed. Other significant works of FIB aggregation include [37] and [38]. Ref. [37] provides a merging approach for multiple FIB aggregation in virtual routing platform. Ref. [38] includes a formal study of the tradeoff between the aggregated FIB size and the update churn.

Our Nexthop-Selectable FIB aggregation is orthogonal to all these studies as we allow a set of selectable next hops for each prefix. We emphasize again that after the execution of our algorithm, every prefix is still mapped to one next hop; and packets will be delivered in single path to the destination. The construction of our set of selectable next hops does not need infrastructure or protocol change of the Internet.

In addition to FIB aggregation, Virtual Aggregation (ViAggr) [39] can shrink the FIB by configuration only. ViAggr divides the global address space into a set of virtual prefix blocks (i.e., 00/2, 01/2, 10/2 and 11/2). Aggregation Point Routers (APR) are responsible for specific virtual prefix blocks. APRs manage all the routes of prefixes covered by the corresponding virtual prefix block. Other routers only manage the intra-domain routes to all the APRs and forward packets to the corresponding APRs. The advantage of ViAggr is that it is incrementally deployable at the ISP level. The configuration overhead of ViAggr is non-trivial. ViAggr also involves twice of tunneling, which may slow down the forwarding speed.

Besides, analogous to our NS-FIB aggregation, MMS [28] also uses multiple acceptable routes to improve the compression. In the intra-domain case, MMS selects sets of acceptable BGP routes. By providing the compression algorithms with the flexibility to choose amongst this set, additional compression can be achieved. In the intra-domain case, MMS select sets of BGP routes that are acceptable for use. By providing the compression algorithms with the flexibility to choose amongst this set, additional compression can be achieved. However, route coalescing of Level 1–6 changes the BGP route selection process and has some influence on inter-AS traffic. To prevent the routing loop caused by this changing, tunnel must be used to forward the affected packets, which complicates the network management. Besides, the aggregating algorithm might change the selected BGP routes for multiple destination prefixes even if only one BGP update occurs, which causes inter-domain routing oscillations. Compared with MMS, NS-FIB aggregation does not change the process of BGP route selection, thus has no influence on inter-AS routing. Multiple selectable next hops to the optimal egress router (of the optimal BGP route) are generated to improve the compression. Therefore, NS-FIB aggregation avoids the potential routing loops or routing oscillations caused by MMS.

As for the AS-wide deployment of MMS, a small set of MMS servers are deployed to run portion of BGP functions on behalf of the AS. MMS servers can select "better" BGP routes for the intra-domain routers to improve FIB compression. In this way, routing inconsistency (and the caused loops) can be avoided. However, as "AS-wide deployment" indicates, this method does not support router-level incremental deployment, which we believe is a significant drawback compared with NS-FIB aggregation and the intra-domain case of MMS.

9.3. Comparison of short-term schemes

As our paper focuses on instant approaches to the Internet routing scalability problem, we make a comparison on the short-term solutions. We compare them in the following six metrics: (1) the FIB size; (2) their capability of being incrementally deployed; (3) the forwarding approaches, to be explained shortly; (4) the dependence of the FIB size on the degree the nodes in the network; (5) path stretch control; (6) management overhead.

We now formally classify the forwarding operation. Let s represent the source and d the destination. Let $c(v_i, v_j)$ denote the cost of link (v_i, v_j), and $dist(v_i, v_k)$ the cost of the shortest path from v_i to v_k. We classify the forwarding operation into the following three forwarding approaches.

Definition 1 (Optimal Forwarding)**.** Let s' denote a neighboring node of s. Then, forwarding $s \to s'$ is an optimal forwarding iff $dist(s, d) = c(s, s') + dist(s', d)$.

Definition 2 (Closing Forwarding)**.** Let s' denote a neighboring node of s. Then, forwarding $s \to s'$ is a closing forwarding iff $dist(s', d) < dist(s, d)$.

Definition 3 (Loose Forwarding)**.** Let s' denote a neighboring node of s. Then, forwarding $s \to s'$ is a loose forwarding iff ss' is part of a path from s to d.

Intuitively, optimal forwarding $s \to s'$ guarantees that the packet is always on the shortest path from s to d. Closing forwarding $s \to s'$ moves the packet closer to the destination. That is to say, s' has a shorter distance to d than s. Closing forwarding guarantees loop avoidance. Loose forwarding may introduce loops and some other methods are needed to avoid forwarding loops. For example, in ViAggr [39] encapsulation is used to divide the forwarding path into two parts.

Lemma 12. *An optimal forwarding is a closing forwarding; a closing forwarding is a loose forwarding.*

Proof. As $c(s, s')$ is positive, $dist(s, d) = c(s, s') + dist(s', d) \Rightarrow dist(s', d) < dist(s, d)$. Thus, an optimal forwarding must be a closing forwarding. It is straightforward that a closing forwarding must be a loose forwarding. □

Here we compare four well-known groups of schemes: (1) General FIB aggregation schemes, mainly based on [19], where several FIB aggregation schemes are classified; (2) NS-FIB aggregation; (3) ViAggr [39], which has been implemented by some vendors; and (4) LISP [14] and APT [16], where the edge addresses are separated from the core network addresses. Note that schemes in the fourth group should actually be considered as long-term solutions. We put them here for completeness.

LISP and APT involve new protocols, i.e., new mapping systems that can fundamentally solve the Internet Scalability problem. Whether or when they will be finally implemented is yet to know. ViAggr [39] shrinks the FIB size to the constant level (the number of virtual aggregation points). However, ViAggr does not support the outer-level incremental deployment. Besides, it leads to high configuration and management overhead. The general FIB aggregation scheme makes the minimum changes to the current Internet infrastructure. Zhao et al. [19] discussed four different maneuvers, each of which can squeeze some drops out of the FIB size. These schemes are degree-dependent. That is, as the degree of the routers increases, (the network is denser), the aggregated FIB size will increase.

There is no almighty scheme solving all the problems: high compression ratio, incremental deployment, low path stretch, fast update, etc. NS-FIB aggregation achieves a higher compression ratio, supports router-level incremental deployment, and enables faster updating, the sacrifice of path stretch is worthwhile. Besides, NS-FIB aggregation avoids the compression performance degrading of single-nexthop aggregation with the increasing of topology density. We believe NS-FIB aggregation provides another choice for ISPs to extend the router survival time.

Acknowledgment

The research is supported by the National Basic Research Program of China (973 program) under grant 2009CB320502 and 2012CB315803, the National Natural Science Foundation of China (61073166), the National High-Tech Research and Development Program of China (863 program) under grants 2011AA01A101 and 2014ZX03002004, and Shenzhen Key Laboratory of Software Defined Network.

References

[1] BGP Reports, (http://bgp.potaroo.net) (Accessed 16.05.2010).
[2] D. Meyer, L. Zhang, K. Fall, Report from the IAB Workshop on Routing and Addressing, 2007 (Internet RFC 4984).
[3] X. Zhao, D.J. Pacella, J. Schiller, Routing scalability: an operator's view, IEEE J. Sel. Areas Commun. 28 (8) (2010) 1262–1270.
[4] V. Khare, D. Jen, X. Zhao, Y. Liu, D. Massey, L. Wang, B. Zhang, L. Zhang, Evolution towards global routing scalability, IEEE J. Sel. Areas Commun. 28 (8) (2010) 1363–1375.
[5] Hubble: monitoring internet reachability in real-time, (http://hubble.cs.washington.edu) (Accessed 20.04.2010).
[6] D. Krioukov, K. Fall, Compact routing on internet-like graphs, in: Proc. IEEE INFOCOM, Hong Kong, 2004.
[7] G. Konjevod, A.W. Richa, D. Xia, H. Yu, Compact routing with slack in low doubling dimension, in: Proc. PODC, Portland, OR, 2007.
[8] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, ROFL: routing on flat labels, in: Proc. Sigcomm, Pisa, Italy, 2006.
[9] R. Moskowitz, P. Nikander, Host Identity Protocol (HIP) Architecture, 2006, (Internet RFC 4423).
[10] E. Nordmark, M. Bagnulo, Shim6: Level 3 Multihoming Shim Protocol for IPv6, 2009, (Internet RFC 5533).
[11] W. Wong, F.L. Verdi, M.F. Magalhães, A next generation internet architecture for mobility and multi-homing support, in: Proc. CoNEXT, New York, NY, 2007.
[12] R. Atkinson, S. Bhatti, S. Hailes, ILNP: mobility, multi-homing, localised addressing and security through naming, Telecommun. Syst. 42 (3) (2009) 273–291.
[13] R. Hinden, New scheme for internet routing and addressing (ENCAPS) for IPNG, 1996, (Internet RFC 1955).
[14] D. Farinacci, V. Fuller, D. Meyer, D. Lewis, Locator/ID Separation Protocol (LISP), 2010, (Internet Draft, draft-ietf-lisp-07).
[15] C. Vogt, Six/one router: a scalable and backwards compatible solution for provider-independent addressing, in: Proc. MobiArch, Seattle, WA, 2008.
[16] D. Jen, M. Meisel, H. Yan, D. Massey, L. Wang, B. Zhang, L. Zhang, Towards a new internet routing architecture: arguments for separating edges from transit core, in: Proc. HotNets, Calgary, Alberta, 2008.
[17] J. Pan, R. Jain, S. Paul, C. So-in, MILSA: a new evolutionary architecture for scalability, mobility, and multihoming in the future internet, IEEE J. Sel. Areas Commun. 28 (8) (2010) 1344–1362.
[18] R. Draves, C. King, S. Venkatachary, B. Zill, Constructing optimal IP routing tables, in: Proc. IEEE INFOCOM, New York, NY, 1999.
[19] X. Zhao, Y. Liu, L. Wang, B. Zhang, On the aggregatability of router forwarding tables, in: Proc. IEEE INFOCOM, San Diego, CA, 2010.
[20] Z. Uzmi, A. Tariq, P. Francis, FIB aggregation with SMALTA, 2010, (Internet Draft, draft-uzmi-smalta-00).
[21] G. Rtvri, J. Tapolcai, A. Kolrosi, A. Majdn, Z. Heszberger, Compressing IP forwarding tables: towards entropy bounds and beyond, in: Proc. ACM Sigcomm, Hong Kong, 2013.
[22] A. Atlas, A. Zinin, basic specification for IP fast-reroute: loop-free alternates, 2008, (Internet RFC 5286).
[23] K. Kwong, L. Gao, R. Guérin, Z. Zhang, On the feasibility and efficacy of protection routing in IP networks, in: Proc. IEEE INFOCOM, San Diego, CA, 2010.
[24] M. Kodialam, T.V. Lakshman, Dynamic routing of restorable bandwidth-guaranteed tunnels using aggregated network resource usage information, IEEE/ACM Trans. Netw. 11 (3) (2003) 399–410.
[25] The China Education and Research Network (CERNET), (http://www.edu.cn/english) (Accessed 10.06.2010).
[26] BRITE: Boston University Representative Internet Topology Generator, (http://www.cs.bu.edu/brite) (Accessed 12.05.2010).
[27] The Route Views Project, (http://www.routeviews.org) (Accessed 16.05.2010).
[28] E. Karpilovsky, M. Caesar, J. Rexford, A. Shaikh, J. van der Merwe, A trade-off between space and efficiency for routing tables, IEEE Trans. Netw. Serv. Manage. 9 (4) (2012) 446–458.
[29] N. Sarrar, R. Wuttke, S. Schmid, M. Bienkowski, S. Uhlig, Leveraging locality for FIB aggregation, in: Proc. IEEE GLOBECOM, 2014.
[30] U. Lakshman, L. Lobo, Mpls traffic engineering, in: MPLS Configuration on Cisco IOS Software, Cisco Press, 2005.
[31] K. Gadkari, D. Massey, C. Papadopoulos, Dynamics of prefix usage at an edge router, in: Proceedings of the 12th International Conference on Passive and Active Measurement, Atlanta, Georgia, USA, 2011.
[32] Rocketfuel: an ISP topology mapping engine, (http://www.cs.washington.edu/research/networking/rocketfuel) (Accessed 17.06.2010).
[33] D.G. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, S. Shenker, Accountable Internet Protocol (AIP), in: Proc. ACM SIGCOMM, ACM, Seattle, WA, 2008.
[34] R. Oliveira, M. Lad, B. Zhang, L. Zhang, Geographically informed inter-domain routing, in: Proc. ICNP, Beijing, China, 2007.
[35] T. Li, Y. Zhu, K. Xu, M. Chen, Performance model and evaluation on geographic-based routing, Comput. Commun. 32 (2) (2009) 343–348.
[36] Y. Liu, B. Zhang, L. Wang, FIFA: fast incremental fib aggregation, in: Proc. IEEE INFOCOM, Turin, Italy, 2013.
[37] L. Luo, G. Xie, K. Salamatian, S. Uhlig, L. Mathy, Y. Xie, A trie merging approach with incremental updates for virtual routers, in: Proc. IEEE INFOCOM, Turin, Italy, 2013.
[38] M. Bienkowski, N. Sarrar, S. Schmid, S. Uhlig, Competitive FIB aggregation without update churn, in: Proc. IEEE ICDCS, Madrid, Spain, 2014.
[39] H. Ballani, P. Francis, T. Cao, J. Wang, Making routers last longer with viaggre, in: Proc. USENIX NSDI, Boston, Massachusetts, 2009.