# Inter-AS Inbound Traffic Engineering via ASPP

Jessie Hui Wang, Dah Ming Chiu, John C. S. Lui, and Rocky K. C. Chang

***Abstract*— AS Path Prepending (ASPP) is a popular method for the inter-AS inbound traffic engineering, which is known to be more difficult than the outbound traffic engineering. Although the ASPP approach has been extensively practised by many ASes, it is surprising that there still lacks a systematic study of this approach and the basic understanding of its effectiveness. In this paper, we introduce the concept, applicability and potential instability problem of the ASPP approach. Some guidelines are given as the first step to study the method to avoid instability problem. Finally, we study the dynamic prepending behavior of ISPs and show a real-world pathologic case of prepending instability based on our measurement study of RouteViews data.**

***Index Terms*— ASPP, BGP, instability.**

## I. INTRODUCTION

ONE can view the global Internet as an interconnection of autonomous systems (ASes). In general, there are two types of AS, namely, *transit AS* and *stub AS*. A transit AS provides Internet connectivity to other ASes by forwarding all types of traffic across its network. A stub AS, on the other hand, does not provide transit service for other ASes and only sends or receives its own traffic. The interconnection of ASes can also be described by a *business relationship*. Major business relationships include the *provider-to-customer* relationship and the *peer-to-peer* relationship. These business relationships play a crucial role in shaping the structure of the Internet and the end-to-end performance characteristics [1]. From the viewpoint of AS relationship, stub ASes are those which have no customer (or client AS), while transit ASes are those with customers. Transit ASes without provider are called "tier-1" ASes.

ASes that have more than one provider are called *multihomed ASes*. Motivated by the need to improve network resilience and performance, there is an increasing number of enterprise and campus networks connecting to the Internet via multiple providers. These multihomed ASes, therefore, must undertake the task of engineering the traffic flowing in and out of the network through these multiple links. Using different inter-AS traffic engineering approaches, ASes can distribute traffic to satisfy their performance or cost constraints [2] [3] [4]. The focus of this paper is on the *inter-AS inbound traffic engineering* problem, which is known to be more difficult than the outbound traffic engineering problem because an AS generally cannot control the routing path for the inbound traffic. Moreover, we restrict our attention to the *AS Path Prepending* (ASPP) approach based on the Border Gateway Protocol (BGP) [5].

In [6], we did a measurement study based on BGP routing tables from RouteViews Project [7]. RouteViews operates a number of BGP data collection points which peer with BGP routers at various ISPs. It captures snapshot every four hours from November 8, 1997, containing more than 7,000,000 routes for more than 160,000 prefixes in each snapshot. This large database makes it possible to study the ASPP behavior of ISPs.

According to our measurement study, at least 12% of the routes have some amount of ASPP today and this indicates that ASPP has a significant impact on the current Internet routing structure [6]. However, it is surprising that there still lacks a systematic study of this approach and the basic understanding of its effectiveness.

The purpose of this paper is to introduce the concept, applicability and potential instability problem of the ASPP approach. The rest of the paper is organized as follows. We introduce the background information about BGP in Section II. In Section III, we explain the concept and principle of the ASPP approach. In particular, we summarize the justifications of performing ASPP in real world based on offline discussions with NANOG [8] subscribers. Section IV extends and improves [6]. We present the algorithm on how to perform ASPP systematically and illustrate that the interaction of ASPP by various ASes can cause routing instability. We also present some simple guidelines for ASes to perform prepending properly. Finally, we introduce our study on dynamic ASPP behavior of ISPs and show a real-world case of ASPP instability based on the analysis of RouteViews data in Section V. Related work is given in Section VI and Section VII concludes.

## II. BORDER GATEWAY PROTOCOL

It is customary to use a connected graph $G = (V,E,B)$ to represent the interdomain network topology and business relationship. Each node $v \in V$ represents an entire AS; and each edge $e \in E$ denotes a logical link connecting two ASes (or ISPs). For each edge $e$, $B(e)$ defines the business relationship between the two ASes for which e connects. Fig. 1 illustrates a network of seven ASes with edges that convey different relationships. In particular, a provider-customer relationship is represented by a directed edge wherein the pointed node represents the customer, whereas a peering relationship between two ASes is represented by a undirected edge.

BGP is the inter-domain routing protocol for the current Internet. The purpose of BGP is to allow two different ASes

Manuscript received December 23, 2005; revised June 5, 2006. The associate editor coordinating the review of this letter and approving it for publication was J. Schoenwaelder.

J. H. Wang and D. M. Chiu are with the Department of Information Engineering, The Chinese University of Hong Kong (email: {hwang3, dmchiu}@ie.cuhk.edu.hk).

John C. S. Lui is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong (email: cslui@cse.cuhk.edu.hk).

Rocky K. C. Chang is with the Department of Computing, The Hong Kong Polytechnic University (email: csrchang@comp.polyu.edu.hk).

Digital Object Identifier 10.1109/TNSM.2007.030106.

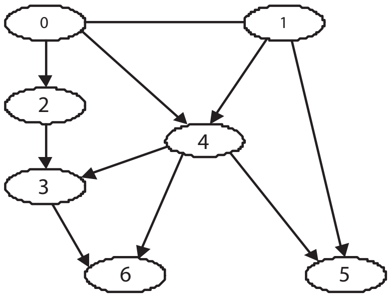

Fig. 1. An example of network with different relationships.

to exchange routing information so that data traffic can be forwarded across the AS border. BGP is based on the distance vector algorithm and it uses TCP as its transport protocol. After a BGP session is setup, each end advertises a route for every prefix that it wants the other end to know. After these initial messages have been exchanged, a node needs to inform the node at the other end about any route changes[5].

When a BGP router advertises a prefix to one of its BGP neighbors, a number of attributes are associated with that announcement. Attributes are used heavily in BGP to carry a wide range of information. For instance, *AS-PATH*, which is an important attribute of BGP, contains the ASes through which the announcement for the prefix has passed. As an announcement is passed between ASes, each AS adds its AS number (ASN) to the AS-PATH attribute. This, by itself, is useful for the operators of the ASes to learn all the information of this route. However, it also provides the critical feature of detecting and preventing looping announcement.

Different ASes have different business concerns, so there are different business agreements between ASes. BGP provides a mechanism to enforce business agreements made between two or more parties. This can be illustrated by the following example. In Fig. 1, $AS_3$ and $AS_4$ are providers of $AS_6$, which implies $AS_6$ pays for the traffic going through the link $AS_3 - AS_6$ and the link $AS_4 - AS_6$. Imagine $AS_3$ wants to send traffic to $AS_4$. $AS_6$, being a customer to both $AS_3$ and $AS_4$, obviously does not want to provide transit service for its providers. To achieve this goal, $AS_6$ will not announce the reachability information of $AS_3$ ($AS_4$) to $AS_4$ ($AS_3$). In short, it is the role of an ISP's routing policy to enforce these kinds of business agreements.

BGP has two kinds of routing policies: *import routing policy* and *export routing policy* (also referred to as *import filtering* and *export filtering*). Import policy determines which routes should be accepted from a neighbor and the preference with which those routes should be treated, while export policy determines which routes should be advertised to a neighbor. If an AS accepts a route from a neighbor, it means this AS agrees to provide transit service for the traffic destined to the prefix of this route. If an AS advertises a route to one of its neighbors, it means this AS would like to accept traffic destined to the prefix of this route from this neighbor. Thus this kind of *routes filtering* is important and necessary for BGP to control how an ISP network is used by its neighbors.

BGP is also a *policy-based* path vector routing protocol. In [9], the authors illustrate the popular policies adopted by ASes in the Internet are: (a) the *typical local preference import policy* and (b) the *selective announcement export policy*. Under the typical local preference policy, an AS prefers to use a customer link than a peering link to forward a packet, and it prefers to use a peering link than a provider link to forward a packet, provided that these links can reach the destination AS. This is natural since an AS does not need to pay for the traffic going through its customer link, while it must pay for the traffic going through its provider link. Under the selective announcement export policy, an AS would not announce the routes learned from its providers or peers to other providers and peers, thus an AS does not provide transit service between its providers or its peers. To illustrate, let us assume all ASes in Fig. 1 obey "local preference" and "selective announcement" policies. Then routes with AS path $(AS_5,AS_4,AS_6)$ or $(AS_5,AS_1,AS_4,AS_3,AS_6)$ are considered legal or valid routes, while routes with AS path $(AS_1,AS_0,AS_4,AS_6)$ would not appear in this network because $AS_1$ would select $AS_4$, instead of AS0 as the next hop to reach $AS_6$ according to the typical local preference. Also, route with AS path $(AS_1,AS_4,AS_0)$ would not appear since $AS_4$ would not announce AS path $(AS_4,AS_0)$ to $AS_1$ according to the "*selective export policy*".

## III. AS Path Prepending and Inbound Traffic Engineering

A BGP router's process to select the best routes from all accepted routes is complicated. A BGP router picks the route with the shorter AS Path among two equivalent routes after the comparison of their "local preference". Thus a possible way to influence the selection of the best routes by a distant AS is to *artificially* increase the length of the AS path by including multiple of its own AS number. This method, which is called *AS Path Prepending* (ASPP), is a popular BGP-based inbound traffic engineering method. In other words, a prepended AS path is an AS path that has some duplicated AS numbers that appear consecutively.

Through ASPP, an AS could affect the distribution of traffic flowing into it. The usage of ASPP for inbound traffic engineering can be illustrated by the following example. Consider the traffic from $AS_1$ to $AS_5$ in Fig. 1. In this network, $AS_1$ receives two routes for prefixes in $AS_5$: ($AS_4$,$AS5$) and ($AS_5$). These two routes have the same local preference because both of them are announced by $AS_1$'s customer neighbors ($AS_5$ and $AS_4$), then the router in $AS_1$ selects the second route as the preferred route for prefixes in $AS_5$ since it has a shorter AS path. If $AS_5$ wishes that traffic from $AS_1$ goes through the link $AS_4 - AS_5$, it can use ASPP and announce AS path $(AS_5,AS_5,AS_5)$ to $AS_1$. Now $AS_1$ receives two routes with AS path $(AS_4,AS_5)$ and $(AS_5,AS_5,AS_5)$. Therefore, the router in AS1 would choose the first route and its decision is changed.

An AS can also ask other ASes to do prependings for it through the *community* attribute. Simply speaking, the capability offered by the community attribute is the ability to associate an identifier with a route. After ASes make agreements on the meaning of some special identifier values, community attribute can be used by an AS to affect the routing policy of other ASes. For example, $AS_6$ sends an announcement to $AS_4$ with a pre-defined community value 400:002. After $AS_4$ receives this route and extracts this community value, it knows $AS_6$ wants it to prepend this route by 2 when it advertises this route to $AS_0$. Therefore, $AS_4$'s routing policy is affected by $AS_6$, and $AS_0$'s best route selection could be affected by $AS_4$'s prepending action. An application of the BGP community attribute in multihome routing is described in [10].

ASPP is used by ASes to tune their inbound traffic distribution. But different ASes have different traffic distribution targets due to their own technical and business requirements. So that it is difficult to discern ASes' concrete goals of their ASPP behaviors. From discussions with some network operators, we summarize some justifications of performing ASPP in the next subsection.

### A. Justifications of using ASPP

- *Load Balancing*:
  The prepending is performed because ASes want to balance the inbound load to meet the capacity requirement. For example, assume $AS_a$ has a Gigabit Ethernet (GigE) link to one provider, and an OC3 to another. Then $AS_a$ is likely going to prepend the OC3. Likewise, if inbound traffic is pushing the AS over its minimum commitment bandwidth on one provider, but is well under its minimum on another, the AS may prepend to help balance the traffic levels.
- *Cost Minimization*:
  In order to minimize the transit cost, a multihomed AS may want to achieve a particular traffic distribution. For example, $AS_c$ has two providers, $AS_a$ and $AS_b$. The total inbound bandwidth of $AS_c$ is 15MBps. The cost for the traffic going through $AS_a$ is $10/MBps, while the cost for the traffic going through $AS_b$ is $8/MBps when the bandwidth is below 10MBps, $12 when the bandwidth is over 10MBps. Thus $AS_a$ wants to tune the inbound traffic to achieve this traffic distribution: 10MBps on the link to $AS_b$ and 5MBps on the link to $AS_a$. It can be implemented through the ASPP approach.
- *Performance Optimization*:
  In general, the length of AS path is not a good metric to measure the performance of a path, *e.g.*, although a route via $AS_a$ has the shortest AS path, the performance of this path may not be the best [11]. Then one AS might prepend this kind of paths to achieve a better performance.
- *Creating Backup Route*:
  Some links only serve as backup paths. One AS may want to prepend a link to make this path a backup choice for failover purposes. In this case, the AS would increase the prepending length on the link until no traffic can be shifted.

### B. Comparison with Other Inbound Traffic Engineering Methods

Note that ASPP is not the only method to do the inbound traffic engineering [12][13]. The other method is to rely on selective advertisements and announce different route advertisements on different links. This method suffers from an important drawback: if a link fails, the prefixes that were announced only on the failed link will not be reachable anymore. A variant of the selective advertisements is the advertisement of more specific prefixes. This technique relies on the fact that an IP router will always select in its forwarding table the most specific route for each packet (*i.e.* the matching route with the longest prefix). For example, if a forwarding table contains both a route toward 16.0.0.0/8 and a route toward 16.1.2.0/24, then a packet whose destination is 16.1.2.200 would be forwarded along the second route. This fact can be used to control the incoming traffic by advertising a large aggregate on all links for fault-tolerance reasons and specific prefixes on some links. This solution solves the problem of selective advertisement, but it may increase the size of the BGP routing tables. Many large providers have implemented filters that reject advertisements for too long prefixes.

The ASPP approach does not introduce longer prefixes, and at the same time takes the advantage of resilience protection from multihomed connections. However, the ASPP approach is often performed in a trial-and-error basis, and many operators believe the route metric is much more accurate and less prone to surprise changes.

## IV. ASPP Practice and Routing Instability

Based on the RouteViews data, we presented some measurement results to show the growth of ASPP in [6]. The result shows the number of the multihomed stub ASes that use ASPP for inbound traffic control is around 33% and the share of such transit ASes from the total number of transit ASes is around 40% in 2004. It also shows that the share of prepended routes has been increased to more than 12%.

We need to point out that the result only shows a conservative view of prepended routes in the Internet. As we know, a BGP router only advertises to its neighbors the routes which are selected as its best routes. So routes with prepending, especially with higher number of prependings, were most likely not selected by the transit ASes and therefore filtered out. So the fact that we observe so many prepended routes in Route View's routing tables also implies ASPP does not always produce the intended results for those ASes that are included as prepended ASes in the AS paths. Otherwise, such paths would not be present as the best paths in the routing tables.

In fact, although the ASPP mechanism is widely used in ASes' traffic engineering for all kinds of goals, there is little prescription for a systematic way to implement it. ASPP is purely a heuristic method. Currently, ISPs do it by trial-and-error, which may take some time to converge to a desirable ASPP configuration and in the meantime make real customer traffic try out different routes. In the next subsection, we propose a Greedy ASPP Search Algorithm for ISPs to practise ASPP systematically.

### A. A Greedy Search Algorithm

In [14], the authors proposed a systematic procedure to predict the changes in traffic distribution for a given new ASPP configuration. We will refer to it as an *ASPP Impact Estimator*. Let $AS_v$ have $m$ provider links, and the current (incoming) traffic distribution be $R(p) = (r_1, r_2, \ldots, r_m)$ where $p$ represents the current ASPP configuration and $r_i$ denotes the traffic intensity on the *ith* provider link. The procedure in [14] would then predict the new traffic distribution $R(p')$, where $p'$ is a new ASPP configuration. Briefly speaking, the ASPP Impact Estimator works as follows:

1) use passive traffic monitoring (such as netflow analysis) to identify a few top (heaviest) traffic flows;
2) announce BGP routes for an unused IP prefix $a$ in $AS_v$ with the new ASPP configuration $p'$;
3) after the new BGP announcements take effect, ping typical source addresses (representing the top senders identified above) from $a$, and watch for any change in the routes for these top flows;
4) based on the change in the routes for the top flows, estimate the change in the volume of traffic, $R(p')$, by assuming the ratio of route change for other flows is the same as that for the top flows.

In this algorithm, $a$ is sometimes referred to as a BGP beacon [15]. Since the beacon is in the local AS, this procedure only works for estimating shifts in traffic destined for the local AS. Therefore, it is most suitable for a stub AS.

If an AS has a well-defined traffic engineering goal (*viz* $R^* = (r_1^*, r_2^*, \ldots, r_m^*)$), then it is theoretically possible to use the ASPP Impact Estimator to search for the best ASPP configuration $p^*$ that best meets the target traffic distribution. We propose that ISPs can deploy the following improved search algorithm, to be referred to as the *Greedy ASPP Search Algorithm*.

To describe the algorithm more formally, let $R^*$ denote the desired (optimal) traffic distribution. Let $f(R(p))$ be a measure of the goodness of a given traffic pattern $R$, resulting from $p$. By definition, $f(R(p)) \leq f(R^*)$. Given $p$ in which link $e$ is prepended, let $p - e$ denote the prepending configuration with one prepending on $e$ removed; similarly let $p + e$ denote $p$ with one additional prepending on link $e$.

**Greedy ASPP Search Algorithm:**
ASes execute this algorithm to search for the best prepending configuration

```
while ( TRUE ) {

    compute $f(R(p))$;

    let $e$ be the link with most room to add traffic
      according to the desired distribution;
    if (the prepending length on $e > 0$) {
      $p' = p - e$;
      if ($(f(R(p')) > f(R(p)))$) {
        $p = p - e$;
        continue;
      }
    }

    let $e$ be the link with most room to reduce
      traffic according to the desired distribution;
    $p'' = p + e$;
    if ($(f(R(p'')) > f(R(p)))$) {
      $p = p + e$;
        continue;
    }

    break;
}
```

The algorithm first tries to reduce the prepending length on the lightest-loaded link, then tries to increase the prepending length on the heaviest-load link. Similar as other greedy search algorithms, the basic idea of this algorithm is to search in the most likely helpful direction in each step, until the desired traffic distribution is reached or there is no helpful prepending action.

We have the following observations on this greedy algorithm:

**Observation 1:** *The greedy ASPP search algorithm stops after a finite number of iterations for any single AS.*

Consider $AS_v$ with $m$ providers, and use a vector $(l_1, l_2, \ldots l_i \ldots l_m)$ to represent one of its prepending configuration, where $l_i$ is the prepending length on its $i$th provider link in this prepending configuration.

Let $n$ be the diameter of the network which is the length of the longest AS path among all possible paths within this network. Obviously, prepending a link with a length of $n$ should be enough to shift all traffic whose routing can be affected by ASPP on this link to other links, and prepending a link with a length of more than $n$ should have the same effect with a prepending with a length of $n$. It means the maximum useful prepending length is $n$. Note that $n$ is the upper bound of the length of a useful prepending.

So we can assume $0 \leq l_i \leq n (i = 1 \ldots m)$. For $AS_v$, there are $(n+1)^m$ possible prepending configurations. Let us sort all these prepending configuration as $(p_1, p_2, \ldots, p_i, \ldots, p_{(n+1)^m})$, where $f(R(p_i)) \leq f(R(p_j))$ when $i < j$.

During the execution, the greedy algorithm would generate a series of prependings configurations $P = (p_{a_1}, p_{a_2}, \ldots, p_{a_i} \ldots p_{a_k})$ where $p_{a_i}$ is the prepending configuration after $i$th iteration. $f(R(p_{a_i})) < f(R(p_{a_j}))$ must hold for any $i < j$ because the algorithm should improve the traffic distribution in each iteration. So we have $a_1 < a_2 < \ldots < a_k \leq (n+1)^m$. We can see that $k$ should be a finite integer which means the greedy ASPP search algorithm stops after a finite number of iterations for any single AS.

**Observation 2:** *The result of the greedy ASPP search algorithm is not guaranteed to be optimal, but the performance is good and the algorithm converges quickly.*

The result of the greedy algorithm is not guaranteed to be optimal because the AS only tries prepending changes on two links in each iteration, while other links are ignored. However, our simulation shows that the performance of this algorithm is really acceptable and it converges quickly.

We do the simulation as follows. We focus on one stub AS, say $AS_v$. The goal of $AS_v$ is to balance its incoming traffic using ASPP approach. Let $E(v)$ be the set of $AS_v$'s provider links and re be the traffic volume on the link $e \in E(v)$. We assume all links in the network have the same bandwidth in order to simplify the simulation. $AS_v$ measures the degree of load balance on its provider links by the following equation:

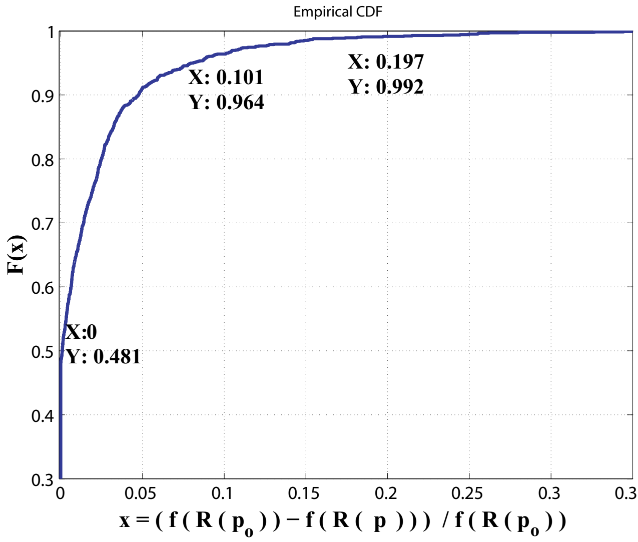

Fig. 2. Comparison of the result of the greedy algorithm with optimal result.

$$f(R(p)) \quad = \quad \frac{\left(\sum_{e\in E(v)} r_e\right)^2}{|E(v)|\sum_{e\in E(v)} r_e^2}. \qquad (1)$$

This index was first proposed for measuring the fairness of bandwidth allocation [16], but it also serves the purpose to measure the degree of the load balance.

In each simulation run, we generate a set of random flows destined to this AS. Each flow can reach $AS_v$ via multiple paths with randomly generated path lengths. We apply the greedy algorithm to search for the best prepending configuration. We then compute the optimal configuration by exhaustive search and compare the results. After 2000 runs, we summarize the results in terms of cumulative distribution functions of different outcomes, as shown in Figure 2 and Figure 3.

In Figure 2, we compare the result of the greedy algorithm with the optimal configuration (derived by exhaustive search). Let $p_o$ denote the optimal prepending configuration, and $p_g$ denote the prepending configuration resulting from the greedy algorithm. We plot the cumulative distribution function of $x = \dfrac{f(R(p_o)) - f(R(p_g))}{f(R(p_o))}$ to evaluate the performance of our greedy algorithm.

We can see that the greedy algorithm gives the optimal result in about 48.1% (960) simulations, $x \leq 0.101$ in 96.4% simulations, and $x > 0.197$ only in 0.8% simulations. It shows the performance of our algorithm is acceptable.

In Figure 3, we plot the cumulative distribution function of the iteration numbers in 2000 simulation runs. We can see that 68% simulations stop in four iterations, 90% simulations stop in six iterations, and only less than 3% simulations take more than 8 iterations to converge. This result shows our algorithm converges quickly.

An exhaustive search algorithm can find the optimal prepending configuration. However, it is infeasible in practice, especially when the ISP has many provider links. There are at least two reasons for this infeasibility. First, there are many prepending configurations so that the AS cannot try all of them. Second, the procedure to evaluate the impact of each ASPP configuration can be very tedious and time consuming.

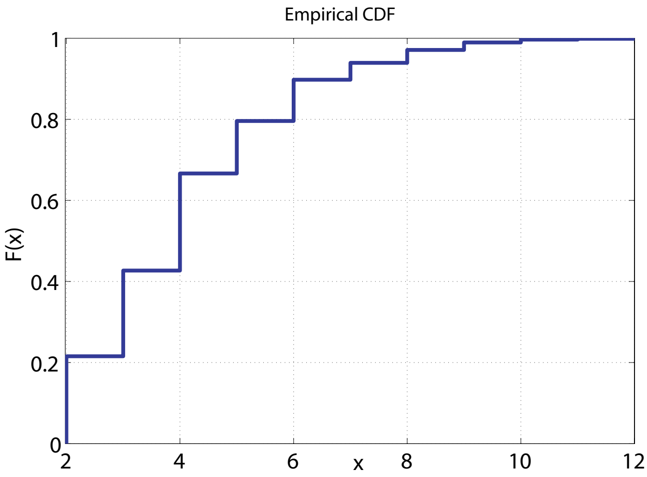

Fig. 3. The distribution of iteration numbers.

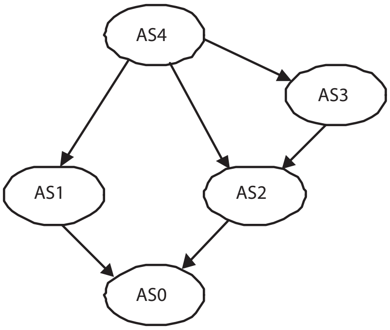

Fig. 4. A network to show the interaction of $AS_0$ and $AS_2$.

Currently ISPs always perform ASPP by trial-and-error, and the process may produce a lot of route update messages. Our proposed algorithm is a feasible approach for ISPs to perform ASPP systematically.

### *B. Routing Instability Caused by Multiple ASes Performing ASPP*

BGP enables ASes to independently define their routing policies based on local objectives and local information with little or no global coordination, thus BGP is not safe in the sense that routing policies can conflict and result in persistent routing oscillations [17]. Various kinds of BGP route oscillation problems have been studied in the past. Varadhan *et al* [18] studied persistent route oscillation in general whenever ASes do independent route selection under local policies. Route oscillation problems with using the MED attribute have been studied in [19]. But none of the previous work considered the interaction of ASPP policies of different ASes.

Fig. 4 gives a simple example of the interaction. In this network, both $AS_0$ and $AS_2$ are multihomed. For $AS_0$, $AS_1$ is more expensive than $AS_2$, therefore $AS_0$ would like to make the link $(AS_1 \rightarrow AS_0)$ a backup link using the ASPP approach. Similarly, the link $(AS_4 \rightarrow AS_2)$ is used as a backup link by $AS_2$.

In this case, $AS_0$'s prepending policy is to "increase the prepending length on the link $(AS_1 \rightarrow AS_0)$ until no traffic

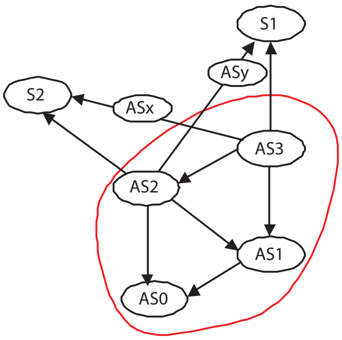

Fig. 5. Interference of prepending actions by $AS_0$ and $AS_1$.

TABLE I
TRAFFIC MATRIX

| Traffic Intensity | $AS_0$ | $AS_1$ |
|---|---|---|
| $S_1$ | 20 | 30 |
| $S_2$ | 10 | 80 |

can be shifted to the other link", while $AS_2$'s prepending policy is to "increase the prepending length on the link $(AS_4 \rightarrow AS_2)$ until no traffic can be shifted to the other link". One can find these two local policies can be satisfied at the same time. The solution is that the traffic from $AS_4$ to $AS_0$ goes through $(AS_4,AS_3,AS_2,AS_0)$. So after $AS_0$ and $AS_2$ find the needed prepending configurations and perform their prepending actions, the network becomes stable. Here, the local policies of these ASes in the network are not conflicting.

However, the distributed prepending actions under conflicting policies of different ASes may interfere with each other and make the routing unstable since there lacks global coordination. Consider the network in Fig. 5. Let us assume that $AS_0$ and $AS_1$ are doing prepending for their inbound load balance on their provider links using the Greedy ASPP Search Algorithm. The degree of load balance is also measured by Equation 1. $S_1$ and $S_2$ are top senders of these two ASes and the traffic demand can be represented by matrix $T$, specifying the traffic intensity from $S_1$ to $AS_0$ and $AS_1$, and $S_2$ to $AS_0$ and $AS_1$ respectively.

When there is no prepending in the network, the traffic from $S_1$ to $AS_0$ goes through three paths $(S_1,AS_y,AS_2,AS_0)$, $(S_1,AS_3,AS_2,AS_0)$ and $(S_1,AS_3,AS_1,AS_0)$. The traffic from $S_2$ to $AS_0$ goes through $(S_2,AS_2,AS_0)$. Thus $r_{AS_2,AS_0} = 2/3*20+10 = 23.33$ [1]. Similarly, we can calculate the traffic on other links and predict what will happen in this network based on the above $f(R(p))$ and Greedy ASPP Search Algorithm.

Table II shows the detailed information about the interference of prepending actions by $AS_0$ and $AS_1$. The first column shows the prepending *change* because of the last execution of Greedy ASPP Search Algorithm. For example, $(2,1)^1$ denotes that $AS_1$ decides to increase the prepending length on the link $AS_2 \rightarrow AS_1$ by 1, and $(2,1)^{-1}$ denotes that $AS_1$ decides to decrease the prepending length on the link $AS_2 \rightarrow AS_1$ by 1.

TABLE II
INTERFERENCE OF ASPP

| P. C. | $AS_0$ | | | $AS_1$ | | | $AS_2$ |
|---|---|---|---|---|---|---|---|
| | $r_{(1,0)}$ | $r_{(2,0)}$ | $f$ | $r_{(2,1)}$ | $r_{(3,1)}$ | $f$ | $r_{(3,2)}$ |
| ∅ | 6.67 | 23.33 | 0.76 | 80.00 | 36.67 | 0.88 | 6.67 |
| $AS_0$ finds nothing to do. $AS1$ finds $(2,1)^1$ can improve its local metric. | | | | | | | |
| $(2,1)^1$ | 6.67 | 23.33 | 0.76 | 40.00 | 76.67 | 0.91 | 6.67 |
| $AS_0$ finds $(2,0)^1$ can improve its local metric. $AS_1$ finds nothing to do. | | | | | | | |
| $(2,0)^1$ | 20.00 | 10.00 | 0.90 | 40.00 | 90.00 | 0.87 | 0.00 |
| $AS_1$ finds $(2,1)^{-1}$ can improve its local metric. $AS_0$ finds nothing to do. | | | | | | | |
| $(2,1)^{-1}$ | 25.00 | 5.00 | 0.69 | 85.00 | 50.00 | 0.93 | 0.00 |
| $AS_0$ finds $(2,0)^{-1}$ can improve its local metric. $AS_1$ finds nothing to do. | | | | | | | |
| $(2,0)^{-1}$ | 6.67 | 23.33 | 0.76 | 80.00 | 36.67 | 0.88 | 6.67 |
| All prependings are cancelled. The network goes back to the initial state. | | | | | | | |

One can find that the prepending actions of $AS_1$ and $AS_0$ are interfering with each other from Table II. The reason for this instability is that there is no solution for both ASes to balance their load at the same time, which means these ASes have conflicting prepending requirements. From the game theory point of view, we can say there is no *Nash Equilibrium* for this game played by $AS_0$ and $AS_1$. If there is no other mechanism to stop it, neither of them would give up and then the best routes for $AS_0$ and $AS_1$ involve an oscillation. In fact, in this example, the prepending policy is dependent on the traffic distribution and *vice versa*, thus the oscillation is similar with the oscillation caused by load-dependent routing.

### C. Guidelines to Avoid Routing Instability

In this subsection, we propose some guidelines to avoid the routing instability caused by prepending actions.

**Guideline 1:** *If only stub ASes are performing prepending actions to balance the traffic on their local provider links, then these prepending actions will not result in routing instability.*

**Guideline 2:** *If no AS performs prepending except on the routes originated by itself, then these prepending actions will not result in routing instability.*

The detailed proof for these two guidelines can be found in [6]. Basically, we propose that ASes should not do prepending on the transit traffic. In practice, transit ASes may lose business (*i.e.* transit traffic) due to their prepending actions. Since they would like to induce more transit traffic to make more money, ASPP is not a suitable approach for them to do traffic engineering to some extent. However, the measurement study in [6] shows many transit ASes are performing ASPP, hence we present the following relaxed guideline:

**Guideline 3:** *If every prefix in the network has only one owner, and only the owner can do prepending on the prefix, the prepending actions will not result in routing instability.*

We suggest that one AS must announce its ownership of the prefix before it uses the traffic to this prefix for traffic engineering, and the AS can announce its ownership only when the prefix does not have an owner. In this way, only one AS may prepend this route. Therefore, there is no policy conflict and routing instability is avoided. Under this guideline, ASes can do prepending on their transit traffic if downstream ASes do not announce ownership for the prefixes. Clearly,

[1] If there are multiple shortest paths with the same length, we assume the traffic from s to d is evenly divided on these paths. This assumption tends to balance traffic automatically. We are interested in studying how load balancing works even under this more favorable assumption.

ASes near the origin AS have a higher priority than ASes far away from the origin AS to be the owner of the prefix. It is reasonable that the provider ISPs should respect their customer ISPs.

Here, we assume transit ASes are able to do *prefix-based prepending* [6], while the first guideline focuses on *link-based prepending*. Our measurement study in [6] shows more than 65% multihomed transit ASes are deploying prefix-based prepending currently. This makes the third guideline feasible in the Internet.

In order to implement it, we should define some special "community" attribute values for global coordination in ASPP practice. An AS needs to signal its use of certain routes for traffic engineering by associating them with the designated community attribute value. This then prevents other ASes from using the same route for traffic engineering.

## V. Inferring ASPP Instability from RouteViews

In the last section, we hypothesized a model for rational ISP behavior, in which each ISP tries to balance its inbound traffic with a well-defined local objective. Based on such a model, we studied the interaction of local ASPP strategies. One observation is that these locally load-balancing ISPs may have conflicting requirements which result in *repetitive* adjustment of ASPP actions.

As we have stated, ASPP is one traffic engineering method achieved by explicitly announcing routes with inflated AS paths to influence other ISPs, thus it is possible to analyze its use based on publicly available routing tables. In this section, we analyze the routing information from the real-life Internet to see if the oscillatory behavior do occur in practice. Based on RouteViews database, we analyze how the prepending configurations change over time using a total number of 388 snapshots, from 8pm February 24th, 2004 to 8pm April 30th, 2004.

Since there are too many prefixes in the database, the first thing we need to do is to find the prefixes which are highly likely to have conflicting prepending requirements. Based on a random picked snapshot, we extract 229 prefixes for further analysis [2].

For each extracted prefix, we analyze its routes over a period of several months to see how different ISPs change their ASPP actions. We also look into the corresponding routing tables to find the best route and see how the prepending action changes affect the best route selection.

Our preliminary study reveals some prependings appear only briefly. However, some other prependings *repeatedly change* during the whole period, which indicates likely interference between ASPP actions by different ISPs.

Table III shows detailed information for one pathological example, where the prefix $\mathscr{P}$ is 80.96.218.0/24, including the prepending actions and the best route. The first column shows the snapshot date (empty means the same date as the previous entry); the second column indicates the AS that performed the ASPP action; the second and the third column together give the link involved in the prepending; and the 4th and the 5th column show the change of the prepending length (from and to). The last column is the best route to prefix $\mathscr{P}$ at that time. For example, the 1st row means that $AS_3$ changed the prepending length from 3 to 0 when it announced routes to $AS_5$ during the period between 8am and 12am on Mar. 15th 2004. [3]

TABLE III
Prepending Changes of Prefix $\mathscr{P}$

| Date | $AS_a$ | $AS_b$ | $l_{before}$ | $l_{now}$ | Best Route |
|---|---|---|---|---|---|
| 03-15-12 | $AS_3$ | $AS_5$ | 3 | 0 | $S\ AS_5\ AS_3\ AS_1$ |
| | $AS_1$ | $AS_2$ | 2 | 3 | |
| 03-17-00 | $AS_3$ | $AS_5$ | 0 | 3 | $S\ AS_4\ AS_2\ AS_1\ AS_1\ AS_1$ |
| | $AS_1$ | $AS_2$ | 3 | 2 | |
| 03-26-08 | $AS_3$ | $AS_5$ | 3 | 0 | $S\ AS_5\ AS_3\ AS_1$ |
| | $AS_1$ | $AS_2$ | 2 | 3 | |
| 03-26-12 | $AS_3$ | $AS_5$ | 0 | 3 | $S\ AS_4\ AS_2\ AS_1\ AS_1\ AS_1$ |
| | $AS_1$ | $AS_2$ | 3 | 2 | |
| 03-29-12 | $AS_3$ | $AS_5$ | 3 | 0 | $S\ AS_5\ AS_3\ AS_1$ |
| | $AS_1$ | $AS_2$ | 2 | 3 | |

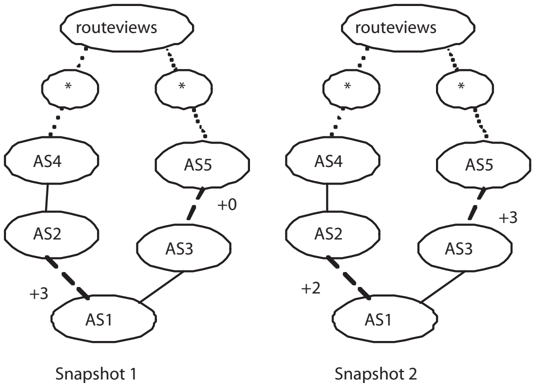

Fig. 6. Routes to prefix $\mathscr{P}$ from RouteViews.

From Table III, one can find the prepending length on the links ($AS5 \rightarrow AS3$) and ($AS2 \rightarrow AS1$) alternatingly changes during the period between 12pm March 15th, 2004 and 4am March 28th, 2004. We observe $AS_3$'s prepending length changes from 0 to 3 and back many times, and $AS_1$'s prepending length changes from 2 to 3 back and forth in a similar way as the example shown in the last section.

Moreover, each snapshot of RouteViews's routing table contains over forty routes for prefix 80.96.218.0/24. In order to clarify the situation, we infer a simplified topology graph, showing all routes from RouteViews to the prefix $\mathscr{P}$ for each snapshot. The first two snapshots are shown in Fig. 6. Because of the route oscillation, the 3rd, 5th and 7th snapshots are the same as the first one, and the 4th, 6th and 8th snapshots are the same as the second one. We see that there are only two groups of routes from RouteViews to this prefix: one group goes through the link $(AS_2 \rightarrow AS_1)$, and the other group goes through the link ($AS5 \rightarrow AS3$). The best route, from the RouteViews vantage point, actually *oscillates* between the two subpaths $(\ldots AS_4AS_2AS_1)$ and $(\ldots AS_5AS_3AS_1)$, depending on the relative amount of prepending applied by $AS_1$ on

[2] For the detailed information on how to extract these prefixes, please refer to our technical report [20].

[3] Note, we have replaced the real AS number with shorthand here.

$(AS_2 \rightarrow AS_1)$ and by $AS_3$ on $(AS_5 \rightarrow AS_3)$.

A possible explanation of what is going on is as follows. At the left snapshot, the best route goes through $AS_5$, thus $AS_3$ wants to increase the prepending length (from 0 to 3) to reduce the traffic, while $AS_1$ wants to decrease the prepending length to induce more traffic on the link from $AS_2$. Because of these two prepending changes, the best route becomes the path through $AS_2$ at the second snapshot. Since there is too much traffic shifted from the right path to the left path, now $AS_3$ wants to increase the traffic on $(AS_5 \rightarrow AS3)$, while $AS_1$ wants to decrease the traffic on $(AS_2 \rightarrow AS_1)$. Then both of them decide to revert their earlier changes, and the oscillation occurs.

Note that there are some limitations with using RouteViews for our analysis. RouteViews receives only the best routes for each prefix from all its neighbors. Therefore, we can only catch the situation when the route with changing prepending is also the best route. Clearly this only gives us a small subset of all the potential cases.

Changes in prependings may also be caused by other reasons, *i.e.*, simply due to the non-stationarity of traffic. We have already carried out some analysis of prepending changes for random prefixes found in the RouteViews database. Indeed they change at a much lower rate. We are also embarking on a bigger task of converting the routing information into a form so that we can analyze *link-based* prepending policies [6]. As we assumed in [6], it is quite likely that ISPs prepend all routes on specific incoming links. If prepending policies are link-based, one can get a better picture if we use a link-based algorithm for the analysis.

## VI. Related Work

Swinnen *et al* used computer simulation to evaluate the ASPP method [12]. In the simulation model, each stub AS was connected to two different transit ASes. When each stub AS prepended one AS to one of the route announcements, their simulation results showed that the distribution of the interdomain paths changed for almost all stub ASes. Moreover, the impact of the ASPP was different for each stub AS. With a prepending length of 2, almost all the inter-domain paths were shifted to the nonprepending link. Beijnum studied the impact of ASPP on a doubly homed stub AS under two different scenarios [21]. The first one was when the stub AS was doubly homed to similar ISPs in the sense that the ISPs directly peered with each other via the same network access point. The second case was when the stub AS was doubly homed to dissimilar ISPs that did not directly peer with each other. He used a simple example to show that applying the ASPP to the second case had a more gradual effect on the change of the incoming traffic distribution.

Lo *et al* conducted an active measurement in RIPE NCC network to study the route-level effects of prepending [22]. They injected beacon prefixes and changed the AS path prepending length of those beacon prefixes every 2 hours for 26 hours. The results reveal a number of hidden processes in the course of propagating prepended routes, which is useful for explaining the method's efficacy and for systemizing the often ad hoc prepending procedure.

Motivated by a lack of systematic procedure to tune the ASPP, Chang and Lo proposed a procedure to predict the traffic change before effecting it. They implemented and tested the procedure in an operational, doubly homed AS which was connected to two regional ISPs [14]. The measurement results showed that the prediction algorithm was fairly accurate. Moreover, the traffic shift peaked when the prepending length was changed from 2 to 3, and almost 60% of the routes were affected.

In [23], the authors proposed a polynomial-time algorithm that determines the optimal prepending length vector for an advertised route at each ingress link of the target network. Specifically, given a set of elephant source networks and some maximum load constraints on the ingress links of the target AS, their algorithm determines the minimum prepending length at each ingress link so that the load constraints are met, when it is feasible to do so. Their algorithm requires as input an AS-Path length estimate from each source network to each ingress link. To deal with unavoidable inaccuracies in the ASPath length estimates, and also to compensate for the generally unknown BGP tie-breaking process in upstream networks, they also developed a robust variation (RPV) of that algorithm.

In [24], Gao and Rexford proposed a set of guidelines for an AS to follow in setting its local policies to avoid route oscillations. But it only focused on local preference setting.

## VII. Discussions and Conclusion

As the ASPP approach continues to be applied by more and more ASes in the Internet, its effectiveness and problems should be studied in detail. In this article, we introduce the basic concept, applications, algorithms and the instability problem of using the ASPP approach. We point out the distributed prepending actions by different ASes may cause routing instability. In our measurement study based on RouteViews data, we observe the pathologic case really happens although the reason is not clear. We also present some guidelines for ISPs to perform ASPP properly. We believe this will help AS operators to apply ASPP systematically and to avoid possible pitfalls.

## Acknowledgments

The work described in this paper was partially supported by Hong Kong RGC #411505.

## References

[1] L. Gao, "On inferring autonomous system relationships in the internet," in *Proc. IEEE Global Internet Symposium*, Nov. 2000.

[2] O. Bonaventure, P. Trimintzios, G. Pavlou, B. Quoitin, A. Azcorra, M. Bagnulo, P. Flegkas, A. García-Martínez, P. Georgatsos, L. Georgiadis, C. Jacquenet, L. Swinnen, S. Tandel, and S. Uhlig, "Internet traffic engineering." in *QofIS Final Report*, 2003, pp. 118–179.

[3] B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, and O. Bonaventure, "Interdomain traffic engineering with bgp," in *IEEE Commun. Mag.*, 2003.

[4] D. K. Goldenberg, L. Qiu, H. Xie, Y. R. Yang, and Y. Zhang, "Optimizing cost and performance for multihoming," in *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY: ACM Press, 2004, pp. 79–92.

[5] John W. Stewart, *BGP4: Inter-Domain Touting in the Internet*. Addison Wesley, 1999.

[6] H. Wang, R. Change, D. M. Chiu, J. C. S. Lui, "Characterizing the performance and stability issues of the AS path prepending method: taxonomy, measurement study and analysis," in *Proc. ACM SIGCOMM Asia Workshop*, April 2005.

[7] Route Views Project, http://www.antc.uoregon.edu/route-views/.

[8] NANOG, http://www.nanog.org/.

[9] F. Wang and L. Gao, "On inferring and characterizing Internet routing policies," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, Oct. 2003.

[10] E. Chen and T. Bates, "Rfc 1998: An application of the bgp community attribute in multi-home routing," ftp://ftp.rfc-editor.org/innotes/rfc1998.txt, Jun. 2003.

[11] B. Huffaker, M. Fomenkov, D. Plummer, D. Moore, and K. Claffy, "Distance metrics in the internet," in *Proc. IEEE International Telecommunications Symposium*, Sept. 2002.

[12] L. Swinnen, S. Tandel, S. Uhlig, B. Quoitin and O. Bonaventure, "An evaluation of bgp-based traffic engineering techniques," www.info.ucl.ac.be/people/ OBO/papers/cost263-chapter.pdf.

[13] B. Quoitin, C. Pelsser, O. Bonaventure, and S. Uhlig, "A performance evaluation of bgp-based traffic engineering," *Int'l. J. Network Management*, vol. 15, no. 3, pp. 177–191, 2005.

[14] R. Chang and M. Lo, "Inbound traffic engineering for multihomed ases using as path prepending," *IEEE Network*, vol. 19, no. 2, March 2005.

[15] Z. M. Mao, "Bgp beacon info," http://www.psg.com/zmao/ BGPBeacon.html.

[16] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," Digital Equipment Corporation, Maynard, MA, USA, DEC Research Report TR-301, Sept. 1984. [Online]. Available: ftp://ftp.netlab.ohio-state.edu/pub/jain/papers/fairness.htm

[17] T. G. Griffin, F. B. Shepherd, and G. Wilfong, "The stable paths problem and interdomain routing," *IEEE/ACM Trans. Networking*, vol. 10, no. 2, pp. 232–243, 2002.

[18] K. Varadhan, R. Govindan, D. Estrin, "Persistent route oscillations in inter-domain routing," *Computer Networks*, vol. 32, no. 1, pp. 1–16, 2000.

[19] T. Griffin and G. Wilfong, "Analysis of the MED oscillation problem in BGP," in *Proc. 10th IEEE International Conference on Network Protocols (ICNP'02)*, Paris, France, Nov. 2002.

[20] J. H. Wang, D. M. Chiu, R. K. Chang, and J. C. Lui, "Inferring stability of decentralized as path prepending policies from routing tables," http://home.ie.cuhk.edu.hk/hwang3/file/infer.pdf, Tech. Rep.

[21] I. Beijnum, *BGP*. O'Reilly, 2002.

[22] S. Lo, R. K. C. Chang, and L. Colitti, "RIPE RRC14 beacon AS paths with AS prepending (collection)," http://imdc.datcat.org/collection/1-01J7-G=RIPE+RRC14+beacon+AS+paths+with+AS+prepending (accessed on 2007-03-20).

[23] R. Gao, C. Dovrolis, and E. W. Zegura, "Interdomain ingress traffic engineering through optimized as-path prepending." in *NETWORKING*, 2005, pp. 647–658.

[24] L. Gao and J. Rexford, "Stable internet routing without global coordination," *IEEE/ACM Trans. Networking*, vol. 9, no. 6, pp. 681–692, 2001.

**Jessie Hui Wang** is now a Ph.D. candidate in the Department of Information Engineering at the Chinese University of Hong Kong. She received her Bachelor degree and Master degree from Tsinghua University in 1999 and 2001. Her research interests include traffic engineering, routing protocols and economic analysis of data networks.

**Dah Ming Chiu** received the B.Sc degree from Imperial Colleage London and the Ph.D. degree from Harvard University in 1975 and 1980. After twenty years in industry (Bell Labs, DEC and Sun), he is now a professor in the Department of Information Engineering in CUHK. His research interest includes Internet, wireless networks and P2P networking. He is an editor for *IEEE/ACM Transactions on Networking*, and TPC member for various IEEE conferences including Infocom, ICNP and IWQoS.

**John C. S. Lui** received his Ph.D. in Computer Science from UCLA. He later joined the Department of Computer Science and Engineering at the Chinese University of Hong Kong. His research interests include theoretic/applied topics in data networks, distributed multimedia systems, network security and mathematical optimization and performance evaluation theory. John was the TPC co-chair of ACM Sigmetrics 2005 and the general co-chair of the International Conference on Network Protocols 2006. His personal interests include films and general reading.

**Rocky K. C. Chang** received his Ph.D. in Rensselaer Polytechnic Institute and was with Computer Science Department at IBM Thomas J. Watson Research Center during 1990-1993. He is currently leading an Internet Infrastructure and Security Group working on Internet security problems, network measurement problems, and the theory of queue stability. His research is supported by the Areas of Excellence Scheme in Information Technology, the Research Grant Council of Hong Kong, Cisco Inc., and The Hong Kong Polytechnic University.