Understanding the AS-level Path Disjointness Provided by Multi-homing

Vijay Vasudevan, David G. Andersen, Hui Zhang

July 2007
CMU-CS-07-141

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Many recent studies have shown that end-to-end path selection can increase availability using mechanisms based on overlay networks or end-host and edge multi-homing. In this paper, we address several unanswered questions about the path diversity that these mechanisms take advantage of: regardless of the path selection techniques used, *where* in the network is it necessary to provide path choice in order to achieve the best gains in availability, and *why* is this the case?
We present results of measurement simulations on inferred autonomous system (AS) topologies to discover the AS-level path disjointness properties of three multipath construction mechanisms. We find that a mechanism that selects *both* the egress edge leaving a source and the ingress edge entering the destination can often obtain as many disjoint paths as an optimal source routing mechanism. In contrast, choosing just the egress edge tends to favor paths that enter the same ingress edge at the destination, resulting in diminished failure resilience. These results are consistent across several different inferred AS topologies, and agree with the results observed using traceroute data between a subset of the Internet ASes.

Keywords: Internet architecture, network, interdomain routing, availability, AS topology

1 Introduction

Numerous studies over the past decade have shown that some form of end-to-end path selection can greatly improve availability [22, 3, 11, 4, 1, 2]. This prior work has typically attempted to answer two questions: how to provide access to multiple paths through the network, and how to select between those multiple paths. The mechanisms proposed have varied from end-host or edge multi-homing via NATs [1] or Web proxies [4], to overlay networks such as RON [3], SOSR [11], and Detour [22].

This path selection appears to enhance availability beyond that provided by IGP or EGP routing protocol fail-over for a number of reasons, the details of which are beyond the scope of this work. In some cases, the routing protocols may take too long to converge following a failure [15]; in other cases, failures may not be visible at the routing level, such as failures caused by misconfiguration [17], malice, or hardware and software bugs [21]. Regardless, the striking conclusion of these prior studies appears to be that *any* form of path selection can substantially improve availability, particularly if it can avoid failure-prone edge access links.

In this paper, we look beyond specific mechanisms for providing and selecting paths and strive to understand *where* in the network topology that it is most important to provide access to multiple paths, and *why* it is so. We believe that the answers to these questions are important for applications ranging from building more reliable networks to designing future network architectures that might natively or otherwise exploit multiple paths [24].

Our results agree with both intuition and prior studies that diversity near the edge or access link is often the bottleneck to availability [2]. We find, however, that being able to choose *both* the edge leaving the source *and* the edge entering the destination is most important. Indeed, choosing both the egress and ingress edges performs as well as being able to select arbitrary paths through the network for over 96% of autonomous system pairs measured. (Section 5). In contrast, the paths available to a system that only selects the egress edge from the source performs as well as arbitrary path construction for only 25% of pairs, showing a strong bias towards entering the *same* ingress edge at the destination, substantially reducing the failure resilience of such a scheme.

We answer our questions by analyzing several recent autonomous system (AS) topologies [7, 12]. We define three multipath path construction mechanisms and evaluate path disjointness properties for all pairs of small degree multihomed ASes in the topology. By analyzing a large traceroute dataset, we also find that similar path disjointness results appear in the real world.

2 Background and Related Work

Prior work has attempted to increase end-to-end availability and performance by providing *concurrent path choice*. While these designs describe different methods of path construction and selection, most share the common goal of providing users of the network with multiple paths to route around both wide-area and/or access-link failures. We believe that understanding the fundamental difference in their mechanism, the **location of path choice**, will help inform a design that provides higher availability.

Several proposals for source routing provide path choice [5, 25, 13, 27], but much of their benefits come at the cost of scalability and policy restrictions; we refer the reader to [24] for a detailed discussion of multipath routing architectures. Regardless, because of their enormous flexibility, we consider source routing models as the "gold standard" for obtaining high availability and seek to understand where path choice can most effectively provide as many disjoint paths as source routing designs.

Both MIRO and Routing Deflections [24, 26] propose new routing architectures to obtain alternate paths to a destination. MIRO obtains alternate paths through pairwise AS negotiation, while Routing Deflections obtains alternate paths by allowing routers to deflect packets off the default path. MIRO assumes that pairwise negotiation can provide enough path diversity to obtain disjoint paths. Routing Deflections uses a tag-based architecture as an alternative to explicit source routing. However, different tags are not guaranteed to provide disjoint paths to the destination – several different tags must often be tried before a successful path is found. Our work attempts to identify which ASes should negotiate to obtain disjoint paths that could be used in a future routing architecture providing failure disjoint paths to end-hosts.

Overlay-based techniques like RON, SOSR, and Detour [3, 11, 22] use overlay nodes to find failure-disjoint paths. RON discovers such paths by probing all one-hop indirect paths between overlay nodes, but suffers from scalability limitations thereby. SOSR instead sends packets through 4 randomly chosen overlay nodes, providing scalability but sacrificing the ability to optimize for latency. Our work identifies the important properties that these overlay routing techniques take advantage of; we believe that our results may help inform designs for future overlay networks by making it more clear where and why they should seek to provide disjoint paths.

Akella et al. note in their comparison of multi-homing and overlay routing that a lack of link diversity near the edges of the network are a major factor reducing the availability benefits from both multi-homing and overlays [2]. Their results particularly applied to multi-homed sources contacting single-homed destinations. Section 3.1 confirms this observation, and we extend this study to consider paths between multi-homed sources to multi-homed destinations.

Failover-path based routing mechanisms [6, 14] attempt to provide immediate failover to backup paths upon detected BGP failures. The authors of R-BGP [14] saw significant benefits to choosing backup paths that are most disjoint from the default path; prior work on best paths in peer-to-peer systems [8] also described the benefits of obtaining disjoint AS-level paths. Our work is complementary to these observations: they measured the benefits of disjointness, and this work hints at new, simpler ways to achieve such disjointness.

Finally, Teixeira et al. [23] found that a traceroute-inferred topology of the Sprint backbone under-counted the potential diversity in the network due to missing backup links. While our study is at the autonomous system level, their results provide a warning that similar un-exposed relationships may lead our AS study to also undercount diversity. We attempt to address this concern by examining the sensitivity of our results to differences in the input AS topology; our results hold across AS topologies with an order of magnitude difference in the number of peering links, lending us some confidence that our results are correct (Section 5.3).

3 AS-Level Path Disjointness

In this paper, we are primarily concerned with path disjointness at the autonomous system level; providing such disjointness is a high-level goal of many intelligent route control products such as Route-Science [20], routing research such as routing deflections [26], multi-homing studies [1], and interdomain routing proposals [14], among many others.[1]

This section first presents our metric for path disjointness and our formulation for the upper bound path disjointness value that an optimal source routing could achieve. It next discusses the three path constructions we compare in our measurement.

[1]While AS-level disjointness is a common goal, it is certainly not the only feasible approach. A harder question that we leave for future work is the issue of locating physically-disjoint paths that may share failures *below* the IP layer.

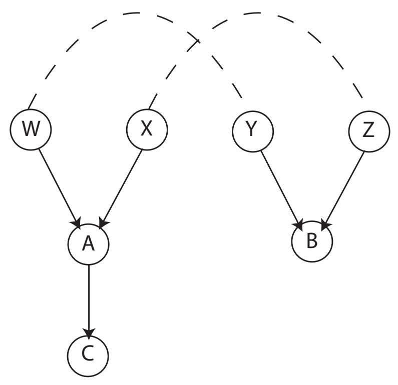


Figure 1: Arrows point in the direction of provider-to-customer relationship. Dashed lines represent peering links. C is not counted towards A's degree when calculating the optimal minimum vertex-cut: A cannot use C to reach any other destination.

3.1 A Metric for Path Disjointness

To understand path disjointness at the AS-level, we examine the minimum cut of the subgraph comprising paths between a source and destination. This metric counts the number of vertices or edges whose removal will disconnect the endpoints; we use this metric to approximate the likelihood of disconnection between two ASes. In this paper, we show results for AS disjointness calculated using min-vertex-cut.

First, we consider the upper-bound minimum vertex-cut, ignoring policy constraints.

Optimal policy-free minimum vertex-cut: we calculate the vertex cut as the *policy-independent* $s-t$ vertex-cut – that is, we assume packets can traverse any link in the system.

We initially calculate the optimal policy-free minimum vertex-cut using a min-cut algorithm between pairs of stub ASes in the entire AS graph. An examination of the value of the vertex-cut suggests that the vertex-cut is nearly always determined by the minimum number of incoming or outbound ASes from the source or the destination: we examined randomly selected pairs of ASes, finding that the cut is equal to $min(src\ degree, dst\ degree)$ for 99.93% of the pairs. Bottlenecks rarely exist more than one hop from the network edges—AS link diversity is plentiful in the core but scarce at the edge.

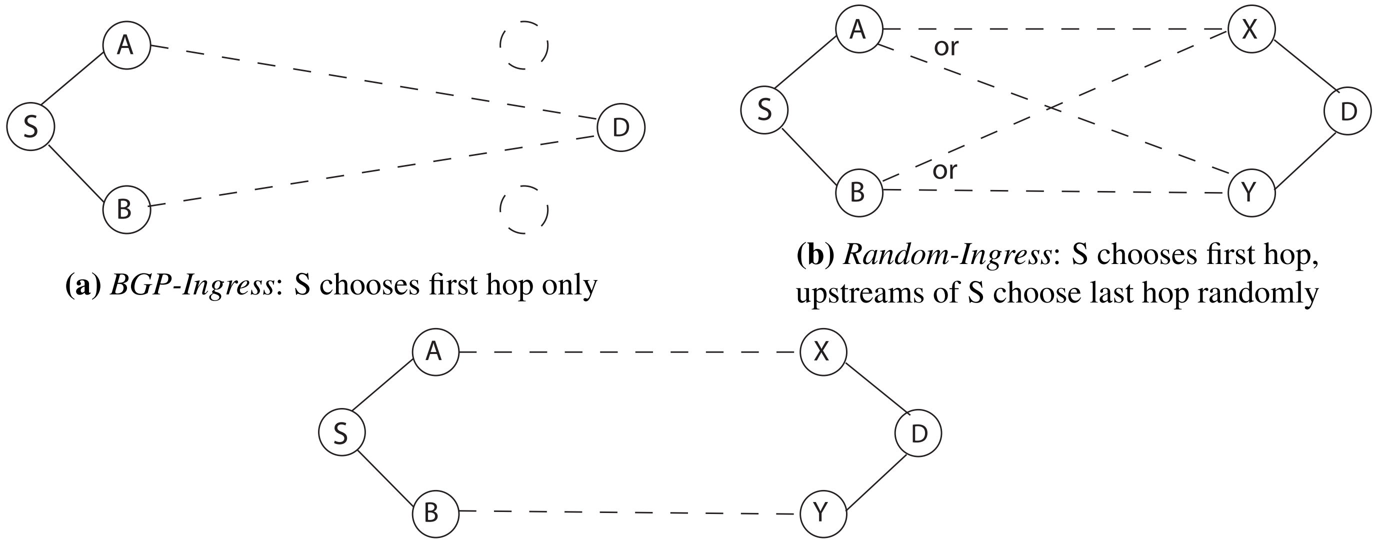
Based upon this result, in the rest of this paper we use $min(src\ degree, dst\ degree)$ as a close approximation of the optimal policy-free minimum vertex-cut. This approximation significantly speeds up the simulations we discuss in Section 4 without decreasing their fidelity.

Next, we consider the optimal minimum vertex-cut given valley-free policy constraints for path construction.

Optimal minimum vertex-cut: we calculate the optimal vertex cut given valley-free policy constraints on paths from the source to the destination. We assume packets can only traverse valley-free paths from the source to the destination.

From our approximation of the optimal policy-free vertex-cut, we apply an additional heuristic to obtain the optimal minimum vertex-cut between an AS pair (A, B). When A's neighbor is a customer of A, we count that link towards A.degree() *only if that neighbor can provide a valley-free path to B*. Consider the example topology in Figure 1. If A's customer neighbor, C, is a stub AS and is not the destination, then we only consider A to have a degree of 2, since C cannot be involved in any valley-free path to other destinations.

This modified approximation gives us the optimal minimum vertex-cut that an arbitrary, policy-compliant source routing scheme could achieve. For the rest of the paper, we use this value and "optimal



(a) *BGP-Ingress*: S chooses first hop only

(b) *Random-Ingress*: S chooses first hop, upstreams of S choose last hop randomly

(c) *Chosen-Ingress*: S chooses both first hop and last hop pair

Figure 2: Solid lines represent links chosen by source or source's neighbors. Dotted lines represent paths constructed using shortest-path-policy.

path disjointness" interchangeably.

3.2 Path Construction Mechanisms

In this section, we present mechanisms that construct a small subset of policy-compliant paths. For each mechanism, we calculate the *actual* minimum vertex-cut of the path set. Our goal is to understand how well these path construction mechanisms can produce path sets with the same minimum vertex-cut as the optimal minimum vertex-cut approximation, in order to discover where path choice can most effectively obtain path disjointness.

We only consider mechanisms with the capability to perform intelligent route control at the source; given that this technology is commercially available today [20], we consider this a reasonable baseline mechanism. This allows a source AS to choose through which upstream neighbor it wants to send packets in order to reach a destination, which any path construction mechanism attempting to obtain high availability would do. Throughout the paper, we use the term "egress AS" to denote the first-hop AS on a path, and "ingress AS" to denote the AS upstream from the destination AS. Below, we define three different methods for constructing paths.

- *BGP-Ingress*: The source chooses only through which egress AS to send a packet, using the shortest path from the source's neighbors to the destination.
- *Random-Ingress*: The source chooses the egress AS, but packets destined to the destination travel over the shortest path from the source upstream AS's to a randomly chosen upstream of the destination.
- *Chosen-Ingress*: The source explicitly specifies through which source egress and destination ingress AS to route, using the shortest path between the egress/ingress AS pairs.

BGP-Ingress is possible in today's BGP routing in the form of "Intelligent Multi-Homing" [1]. In this scheme, the source can choose which provider or peer it sends traffic to, letting the egress AS decide how it will send a packet to the destination. This method provides robustness to failures of any of the source's links, but does not control the path from the egress AS to the destination. Figure 2(a) shows an example where the source node only chooses which egress neighbor to send through, with shortest-path policy determining the path between the neighbors and the destination.

Random-Ingress exposes the choice to route through a destination's upstream neighbors. Although we do not propose this mechanism for real deployment, its behavior is useful to understand where in the AS topology path choice should be provided. Figure 2(b) shows how S's egress neighbors randomly choose one of the ingress neighbors to send traffic to, using shortest path policy to determine the path between egress and ingress neighbors.

Chosen-Ingress explicitly allows the source to pick both an egress AS and ingress AS to route through. Figure 2(c) shows an example where S chooses the paths defined by the pairs (A,X) and (B,Y). Notice that in this example, the source could also choose the pairs (A,Y) and (B,X), or if desired, (A,X) and (B,X).

To increase failure resilience by using multiple paths in parallel, the source should choose a matching between an egress AS and an ingress AS such that no two egress ASes match with the same ingress AS: the source should ensure that all of the ingress links at the destination are included in the set of paths. We therefore always consider such a matching when studying this model. While there are many matchings when dealing with large neighbor sets, we show in Section 5 that with high probability, any matching will yield the best possible minimum vertex-cut.

The aim of our measurement study is to see how many fully disjoint paths each path construction mechanism can obtain, and to compare this number with the optimal path disjointness value described in Section 3.1.

4 Method

We first discuss the data sources we use to construct the AS graph. Next, we describe our motivations for which ASes we study in our measurement. Finally, we describe our measurement algorithm in detail.

4.1 AS Topologies

We conduct our measurement study by performing simulations on three AS graphs. We generate different AS graphs based on annotated AS relationships from both CAIDA and He et. al ([7, 12]). We also use Gao's algorithm [10] to infer the relationships based on BGP routing data from RouteViews [19].

The AS graphs constructed from these datasets differ in several ways. Each graph is based on data from different dates, and certain graphs more accurately describe particular kinds of relationships. Most of the analysis in Section 5 uses the February 26, 2007 CAIDA dataset, though we also run the measurement on AS graphs generated from other datasets and find relatively little difference in our results. We discuss this difference in detail in Section 5.3.

4.2 Source and Destination ASes

We consider multi-homed ASes (those with more than 1 non-customer relationship) with degree between 2 and 5. We choose these ASes as they represent the organizations most likely to multi-home for relia-

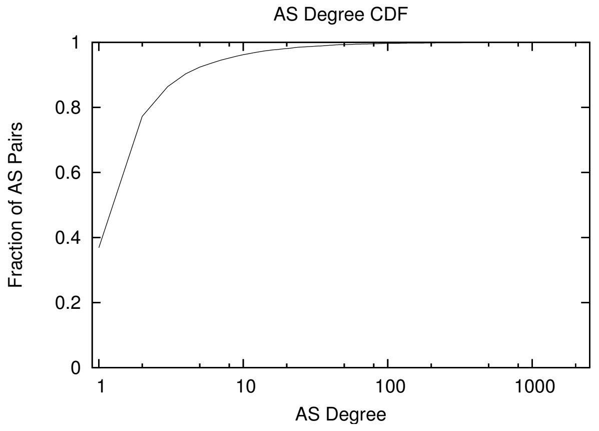


Figure 3: Distribution of AS degrees in Graph

Degree	Number of ASes
2	9431
3	2097
4	903
5	460

Table 1: Distribution of degree for ASes in our measurement

bility. Figure 3 shows the distribution of AS degrees for a February 2007 CAIDA annotation, showing that roughly 90% of ASes have degree of 5 or lower. In addition, 2 to 5-multi-homed ASes comprise about 84% of multi-homed ASes. ASes with degrees 6 or more are thus a small fraction of multi-homed ASes and are mostly larger providers whose path disjointness measurements are likely captured by the measurements between lower degree ASes. Table 1 shows the degree distribution of multi-homed ASes we consider in our measurement. Note that 2-multi-homed ASes will be involved in 92.8% of all pairs in our measurement.

To improve the accuracy of our results, we attempt to remove ASes that inaccurately appear to be multi-homed. In particular, some ASes are homed at geographically separated locations, but use only one AS number; AS X may advertise 128.2.0.0/16 in Pittsburgh but may also advertise 128.2.14.0/24 in Qatar. Including these ASes in our measurement would bias our results in favor of showing more path diversity than what really exists, because the "multi-homing" does not reach the same set of nodes.

Using the `netmap` database [9], which contains information about the geographic location of IP addresses, we identify multi-homed ASes in our study that advertise two different prefixes that are IP-geolocated over 100 miles apart, and remove them from consideration. Only 14 ASes in our original set of nodes appear to be geographically separated. Although the `netmap` database is based on measurements collected in 2005, we do not believe that there would be a significant increase in the number of geographically separated ASes if we were to use current information. Regardless, given 13000 ASes in our comparison set, these geographically separated ASes would not significantly affect our results.

4.3 Measurement Algorithm

For each path construction mechanism, we construct paths according to Algorithm 1 and run a minimum-cut algorithm to determine the number of disjoint paths obtained. We formally discuss each path construction below.

Algorithm 1 Path Construction Algorithms

```
procedure BGPIngress-PathConstruct(S,D)
    Initialize P to nil
    N = Neighbors(S).
    for all $N_i \in N$ do
        SP = ShortestPath($N_i$, $D$)
        P.append(S + SP)
    end for
    return P

procedure RandomIngress-PathConstruct(S,D)
    N = Neighbors(S)
    M = Neighbors(D)
    for all $N_i \in N$ do
        $M_R$ = randomly picked node from $M$
        SP = ShortestPath($N_i$, $M_R$)
        P.append(S + SP + D);
    end for
    return P

procedure ChosenIngress-PathConstruct(S,D)
    N = Neighbors(S)
    M = Neighbors(D)
    $\pi$ = Set of Maximal Matchings on Bipartite(N, M)
    for all x $\in \pi$ do
        Initialize P' to nil
        for all $x_i$ = ($N_a$, $M_b$) $\in$ x do
            SP = ShortestPath($N_a$, $M_b$)
            P'.append(SP);
        end for
        if minCut(P') > minCut(P)) then
            P = P'
        end if
    end for
    return P
```

In *BGP-Ingress*, paths are of the form $(S, N, O_1 ... O_k, D)$, where N is a neighbor of S and $O_1 ... O_k$ represents the shortest valley-free-customer-preferred (VFCP) route between N and D.

For *Random-Ingress*, paths are of the form $(S, N_i, O_1 ... O_k, M_R, D)$, where M_R is chosen randomly from the set of neighbors of D and $O_1 ... O_k$ is the shortest VFCP route between N_i and M_R.

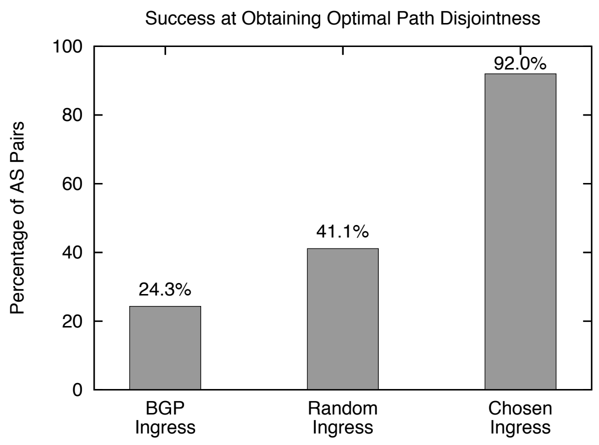


Figure 4: ***Chosen-Ingress*** **obtains optimal path disjointness for most AS pairs.**

Finally, *Chosen-Ingress* looks at all possible maximal matches of $N_i \in Neighbor(S)$ to $M_j \in Neighbor(D)$, and chooses the set of paths $(S, N_i, O_1 ... O_k, M_j, D)$ that obtains the maximum possible number of disjoint paths.

5 Results

5.1 Where Path Choice Matters

Our first results examine *where* in the network topology it is most important to provide access to multiple paths. We find that 1) being able to choose both the egress and ingress ASes can often achieve the optimal set of disjoint paths that an arbitrary source routing mechanism could achieve, 2) choosing only the egress AS does not effectively use all the ingress links at the destination, and 3) choosing a coordinated match that achieves optimal path disjointness under *Chosen-Ingress* is not difficult.

5.1.1 Choosing Both Egress and Ingress ASes Is Important

We compare the AS path disjointness values for each AS pair in our measurement under *BGP-Ingress, Random-Ingress, and Chosen-Ingress*. Figure 4 shows that 92% of AS pairs achieve optimal path disjointness under *Chosen-Ingress*. In contrast, *BGP-Ingress* obtains optimal path disjointness for fewer than 25% of AS pairs.

These results indicate that **to achieve disjoint paths through the Internet, it is important to express the AS upstream choice at both the source and the destination AS**. An architecture based on *Chosen-Ingress* path construction may provide as much path disjointness as choosing arbitrary paths through the Internet *without* the administrative drawbacks associated with source routing.

Random-Ingress obtains optimal path disjointness for 42% of the AS pairs measured. Given that most of the pairs we measure are both 2-multi-homed, this almost meets our expectation: two source egress neighbors will randomly choose the same ingress node 50% of the time, resulting in a minimum vertex-cut of 1 for half of the pairs, and a 90% chance of obtaining a minimum vertex-cut of 2 for the other half of the pairs given the results of *Chosen-Ingress*.

Additionally, *BGP-Ingress* performs worse than *Random-Ingress*, indicating that shortest paths between the source's upstreams and the destination do not randomly pick amongst the destination's up-

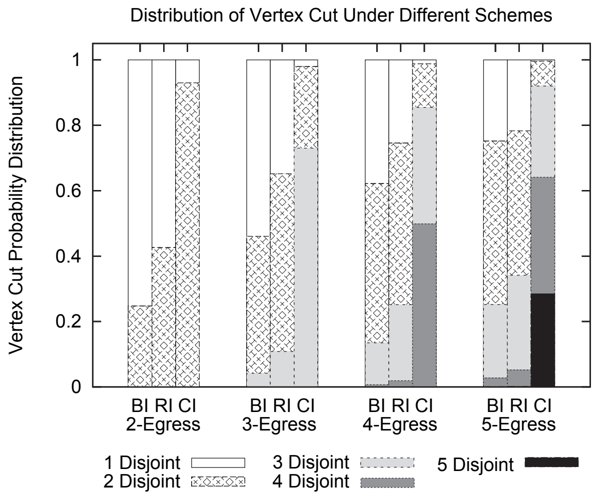


Figure 5: Path disjointness distribution split based on the number of upstream ASes at the source. BI = ***BGP-Ingress*****, RI =** ***Random-Ingress*****, CI =** ***Chosen-Ingress***

stream links – they usually favor one particular ingress link, yielding a suboptimal minimum vertex-cut.

5.1.2 Effectiveness Under Increased Multi-Homing

Figure 5 shows the distribution of path disjointness values when we split the results based on the number of upstream providers at the source AS. For example, the 3-egress cluster shows the probability distribution for minimum vertex-cut values under **-Ingress* when the source AS has three neighbors and the destination AS has *at least* three neighbors. This cluster shows that *Chosen-Ingress* obtains 3 disjoint paths for about 70% of 3-multi-homed AS pairs, whereas *BGP-Ingress* obtains 3 disjoint paths for fewer than 10% of these pairs.

These results show that *Chosen-Ingress* **makes effective use of increased multi-homing**. As we increase the number of egress links available to the source, *Chosen-Ingress* obtains the optimal minimum vertex-cut significantly more often than both *Random-Ingress* and *BGP-Ingress*. We find that about 60% of the 5-multi-homed AS pairs have at least 4 disjoint paths between them when using *Chosen-Ingress*.

In comparison, *BGP-Ingress* and *Random-Ingress* are not able to take advantage of increased multi-homing. When both the source and destination have more than two neighbors, *BGP-Ingress* achieves optimal path disjointness for fewer than 5% of pairs. In particular, almost none of the pairs achieve 4 or 5 disjoint paths when both ASes are 5-multi-homed. Similarly, *Random-Ingress* fails to obtain 5 disjoint paths for most 5-multi-homed pairs: the theoretical probability that each of the five source neighbors would independently choose an unmatched destination neighbor at random is about 4%. Thus, as the number of egress links increases, it becomes less likely that *Random-Ingress*, and hence *BGP-Ingress*, will obtain optimal path disjointness.

5.1.3 Choosing A Matching is Easy

For *Chosen-Ingress* path construction, there are many possible ways to match egress links to ingress links in an effective fashion, such that each ingress link is matched with a unique egress link. If there are k egress links and l ingress links with $k \leq l$, then there are $\frac{l!}{(l-k)!}$ possible matchings. If only a few

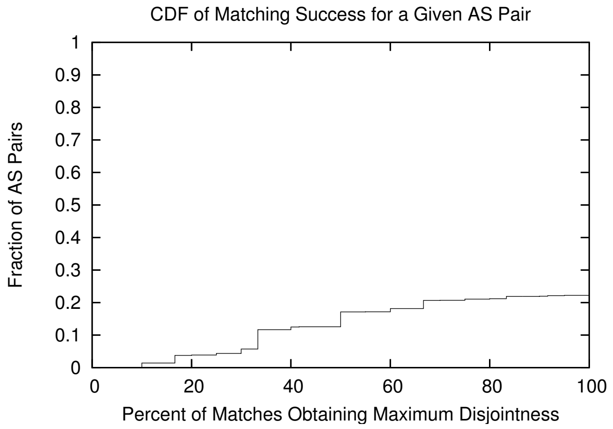


Figure 6: For a majority of AS pairs, all matches obtain maximum obtainable path disjointness characteristics.

of these matchings achieved the best possible number of disjoint paths for that pair, finding an optimal match would require testing many of these permutations.

For every pair of ASes in our study, we calculate the "matching success percentage," or the fraction of these $\frac{l!}{(l-k)!}$ matchings that obtained the maximum possible minimum vertex-cut metric for that pair. We plot the CDF of this success rate in Figure 6.

These results show that for almost 80% of AS pairs, *all* matchings obtain the maximum possible number of disjoint paths. Based on these results, it is unnecessary to try all different matchings: for most AS pairs, any match will obtain the highest number of disjoint paths possible.

5.2 Why Choosing the First and Last Hop is Important

Next, we try to analyze why *BGP-Ingress* performs poorly and *Chosen-Ingress* succeeds at obtaining optimal path disjointness. We find that 1) *BGP-Ingress* favors the ingress link closer to the core and 2) short path lengths tend to eliminate the possibility of AS collision in *Chosen-Ingress*.

5.2.1 Paths prefer the ingress AS closer to the core

Given the relatively poor performance of *BGP-Ingress*, we seek to understand why it fails to achieve optimal path disjointness for a majority of the AS pairs we study. To simplify our analysis, we consider the measurements between only 2-multi-homed ASes.

When a pair obtains only 1 disjoint path in *BGP-Ingress*, only one of the destination's ingress links is utilized. Since we use shortest-path policy to determine paths between the source's neighbors and the destination, we hypothesized that path lengths play a role in determining this preference. For each 2-multi-homed AS, we compute the lengths between each upstream neighbor and the closest Tier-1. Out of the approximately 9500 2-multi-homed ASes in our measurement, we find that half of them have upstream neighbors with different distances to the closest Tier-1.

When the destination's neighbors are different distances from the core, the neighbor closest to the Tier-1 will be the preferred ingress AS, since most sources will construct paths through the core to reach the destination. Thus, for half of the AS pairs, we expect one of the ingress links to be heavily preferred. When the distances are equal, we expect one of the neighbors to be chosen roughly at random: this case

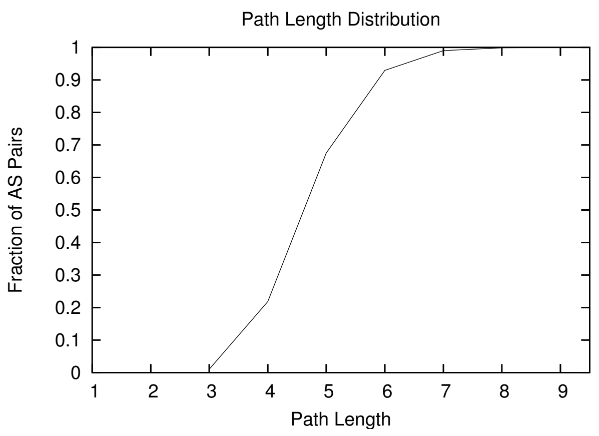


Figure 7: Most AS pair paths in our measurement are short: there is a low probability of colliding in the core under *Chosen-Ingress*.

boils down to *Random-Ingress* path construction, so we would expect to see results similar to *Random-Ingress*, which obtains optimal path disjointness half of the time.

If the closer neighbor is *always* chosen and that core-equidistant neighbors are chosen perfectly randomly, 75% of AS pairs would obtain a path disjointness of 1 and 25% would obtain a path disjointness of 2. These numbers match with our results in Figure 4, and provide some intuition for why *BGP-Ingress* obtains optimal path disjointness for only 25% of pairs.

This explanation appears mostly correct. The neighbor closer to the Tier-1's is not always taken, but we observe a strong bias. When a destination's neighbor is closer to the core, we find that a source will use the closer of the destination's neighbors in 87% of measurements[2].

Multi-homed ASes that do not choose their providers carefully will not see many of the benefits of multi-homing under *BGP-Ingress* path construction if shortest-path-policy is used. Some customers or ISPs may attempt to use path length padding or community string-based traffic engineering techniques to avoid this skew. Unfortunately, just the problem of load-balancing inbound traffic is difficult, and it is unclear if other traffic engineering demands would complement or conflict with an attempt to achieve random ingress selection.

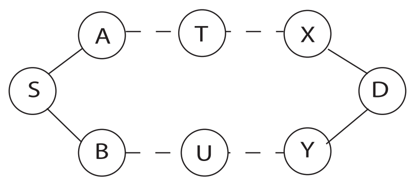
5.2.2 Collisions Unlikely Due to Short Path Lengths

To understand why *Chosen-Ingress* obtains disjoint paths through the core, we plot the distribution of path lengths for paths in our measurement in Figure 7. Over 70% of paths consist of five or fewer ASes (including the source and destination).

To understand why path length helps prevent collisions, consider a path of length 4, consisting of only the source AS, the source's upstream providers, the destination AS's upstream providers, and the destination AS. *Chosen-Ingress* would explicitly specify the *full* path in this scenario, so paths of length 4 *cannot* share any vertices. Similarly, paths of length 5 may share one common third hop AS, but given the diversity of core connectivity, we imagine that this worst case situation would not happen very often. (Figure 8(a)).

This intuition is complicated by the fact that default paths between neighbors may pass through

[2]Although this does not explain the remaining 13%, we believe our definition of "distance to the closest Tier-1" may introduce some unwanted asymmetries: we plan to address this in future work.



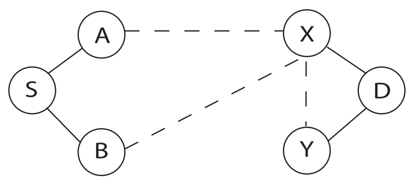


Figure 8: (a) When path lengths are short, vertex-disjoint paths are guaranteed. (b) The default path to a destination's neighbor traverses the other destination's neighbor in only 1.5% of AS pairs under *Chosen-Ingress*.

Dataset	# ASes	# Links	# Peer Links
CAIDA	24297	99166	8522
UCR	19936	119016	41966
Gao	23446	90000	6794

Table 2: AS Graph Characteristics for Different Datasets

another neighbor en route to the destination. This may happen, for example, if a destination uses one of its upstreams as a backup to reach the other. Figure 8(b) shows this case, where the default path from B to Y traverses D's other neighbor, X, likely colliding at the link entering X. This scenario is, however, rare, affecting only 1.5% of AS pairs.

As described in Section 6, most AS paths we measure using traceroutes are the same length as in our simulation, suggesting that short path lengths would also help achieve disjoint paths under *Chosen-Ingress* in the real world.

5.3 Sensitivity to AS Graph Inaccuracy

For economic and business reasons, AS relationships are not publicly available for researchers to study. In this section, we analyze how sensitive our results are to variance in the accuracy of the inferred AS graph.

We repeat our simulation measurement study using three inferred AS graphs: 1) An inferred graph using Gao's algorithm [10] on RouteViews data from October 2006, 2) an annotated AS relationship dataset from CAIDA [7] for February 2007, and 3) the annotated AS relationship dataset from UCRiverside [12] based on May 2006 data. Table 2 shows a few differences in the AS graph generated from these datasets. Note the significantly larger number of peering links inferred for the UCR dataset.

The fraction of AS pairs obtaining optimal path diversity did not change significantly when using these different AS graphs; the results are plotted in Figure 9. *Chosen-Ingress* achieved optimal path disjointness for 92% of AS pairs in a CAIDA-based dataset; for 87% of AS pairs using the UCR dataset; and for 94% in the Gao-inferred topology. It is unclear whether these differences are heavily influenced by the snapshot date of these topologies, though it appears that the accuracy of relationship inference has a larger influence on path disjointness.

Overall, we find that our results are not significantly changed by variability in the accuracy of the AS graph, suggesting that future improvements to AS graph inference would not invalidate our conclusions[3].

[3]We make no claim that our results would hold through structural changes to the Internet graph.

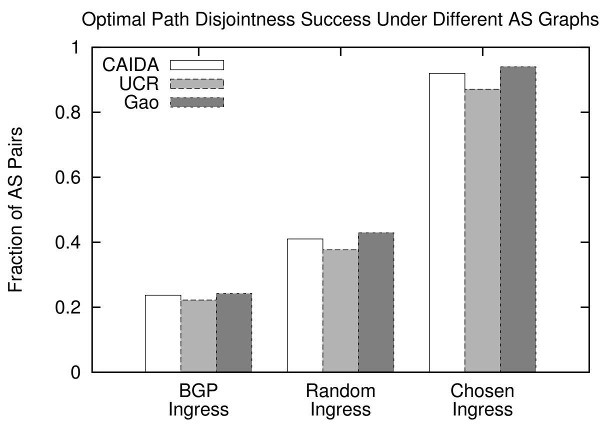
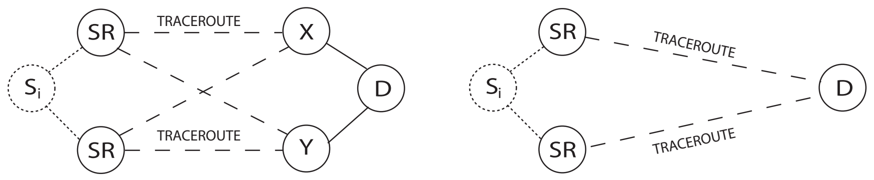


Figure 9: Different AS graphs produce only small differences in path disjointness values



(a) **(b)**

Figure 10: Traceroute Experiment Measurement for (a) ***Chosen-Ingress*** **and (b)** ***BGP-Ingress***

6 Traceroute Validation Study

We perform a verification study of our results in Section 5 using traceroute data provided through the iPlane project [16]. The iPlane project releases a daily set of traceroutes taken from about 175 PlanetLab servers and public traceroute servers to destinations representing BGP atoms, covering most advertised IP prefix destinations. The results shown below are based on a traceroute set from May 24, 2007.

We first identify the set of *emulatable ASes*; those 2-5 multi-homed ASes that have a PlanetLab/Public Traceroute server in each of their upstream providers. No ASes had more than 3 upstreams with PlanetLab/Public Traceroute servers in them, so we study only the two and three-multi-homed ASes. In total there were 163 emulatable ASes given the location of these *Upstream Traceroute Sources*.

We perform a measurement analagous to our simulation measurement study to understand how well *Chosen-Ingress* and *BGP-Ingress* path construction schemes can obtain optimal path disjointness in the real world. For each emulatable AS, we obtain the traceroute path from its upstream ASes to all other 2-5 multi-homed destinations using *Chosen-Ingress* and *BGP-Ingress* schemes.

In the iPlane dataset, some ASes have multiple upstream traceroute sources that are geographically separated. For example, the PlanetLab servers at Intel Research Pittsburgh and Berkeley are both in AS7018 according to the Routeviews table, yet are separated by about 3000 miles. If an emulated AS were to send packets through its upstream AS7018 to some destination AS D, the packets would traverse the particular AS path announced in the BGP announcement received from AS7018. However, as we do not have traceroutes originating in the emulated AS, the path from Intel Research Pittsburgh to AS D may be different than the path from Intel Research Berkeley.

When given a choice of multiple PlanetLab servers within the upstream AS, we always choose the server geographically closer to the emulated AS. The path chosen for packets originating in AS7018 may

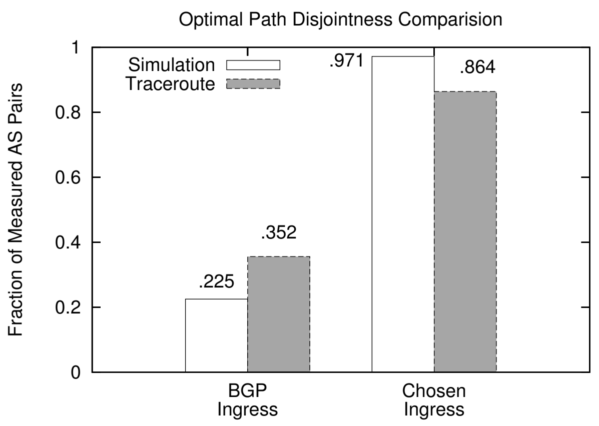


Figure 11: Path disjointness comparison between simulation and traceroute results for *Chosen-Ingress* and *BGP-Ingress*. Simulation results are computed on subset of AS pairs for which traceroute paths are available.

still be different from the path advertised to the emulated AS due to local decision processes; given the lack of traceroute data originating within the emulated AS, we are unable to verify how much impact this may have.

We reverse map the traceroute results to an AS path by performing a longest-prefix match in the Routeviews table. We employ some additional heuristics [18] to obtain a more accurate AS path when a traceroute contains router hops with private address space or other AS path anomalies. As a result, each traceroute gives us a series of AS path hops. We then perform the min-vertex-cut calculation on each set of AS paths and compare against the optimal vertex-cut as in our simulation.

6.1 Traceroute Results

From the available traceroutes, we obtain path disjointness calculations for 1,577,912 pairs for *Chosen-Ingress* and 538,219 pairs for *BGP-Ingress*. We obtain more results for *Chosen-Ingress* because it is more likely that iPlane traceroutes reach a destination's upstreams than the destination itself; destination ASes often do not respond to traceroute probes.

Based on the traceroute data, we find that 86.4% of AS pairs measured under *Chosen-Ingress* and 35.6% of pairs under *BGP-Ingress* obtain optimal path disjointness. To directly compare against simulation, we rerun our simulation on the subset of AS pairs for which we have traceroute results. As illustrated in Figure 11, traceroute-based paths show less disjointness using *Chosen-Ingress* but show more diversity using *BGP-Ingress*.

We split the traceroute results based on the egress-degree as before to understand how *Chosen-Ingress* and *BGP-Ingress* take advantage of more egress links. Again, we find that *Chosen-Ingress* produces 3 disjoint paths for around 70% of AS pairs with degree 3.

There are several interesting results that arise from this study. First, the near-optimal path disjointness properties provided by *Chosen-Ingress* appear to exist in actual real world measurements. Second, the increased path diversity for *BGP-Ingress* in traceroute compared to simulation suggests that either inferred AS topologies underestimate the diversity that exists in the real world, or that the shortest-path-policy may not be enough to simulate paths accurately; ASes may use traffic engineering tricks like BGP communities or path padding in BGP announcements to more effectively load-balance their upstream links. Finally, despite the higher diversity in traceroutes than simulation for *BGP-Ingress*, it still falls

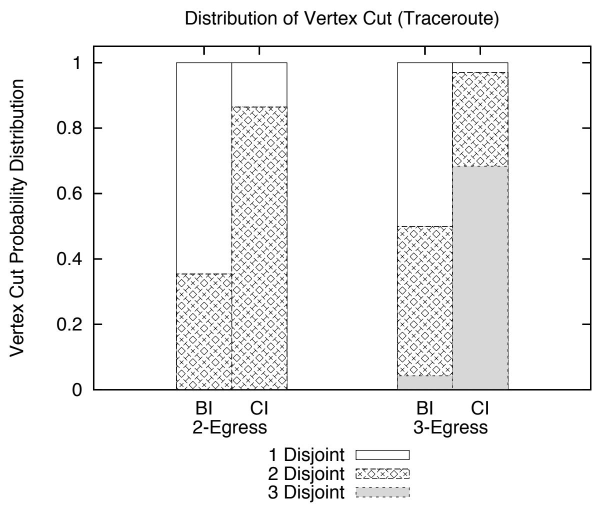


Figure 12: Path disjointness distribution split based on number of upstream ASes at the source; data generated from traceroute results.

well short of providing optimal path disjointness for the majority of AS pairs and may remain so given that *Random-Ingress* represents the best possible scenario for *BGP-Ingress*.

7 Conclusion

This paper examined the importance of egress and ingress path selection for effective multi-homing. We found that providing the ability to choose the ingress link to the destination is unexpectedly important because of path-length and policy-induced bias towards a single ingress link. With control allowing a choice of the egress AS and ingress AS, however, a source can achieve near-perfect AS path disjointness, without requiring the fine-grained AS path control provided by full source routing. These findings help explain the successes and limitations of several proposed multi-homing techniques, and we believe they hold insights that could be useful for the design of future network architectures.

References

[1] Aditya Akella, Bruce Maggs, Srinivasan Seshan, Anees Shaikh, and Ramesh Sitaraman. A measurement-based analysis of multihoming. In *Proc. ACM SIGCOMM*, Karlsruhe, Germany, August 2003.

[2] Aditya Akella, Jeff Pang, Bruce Maggs, Srinivasan Seshan, and Anees Shaikh. A comparison of overlay routing and multihoming route control. In *Proc. ACM SIGCOMM*, Portland, OR, August 2004.

[3] David G. Andersen, Hari Balakrishnan, M. Frans Kaashoek, and Robert Morris. Resilient Overlay Networks. In *Proc. 18th ACM Symposium on Operating Systems Principles (SOSP)*, pages 131–145, Banff, Canada, October 2001.

[4] David G. Andersen, Hari Balakrishnan, M. Frans Kaashoek, and Rohit Rao. Improving Web availability for clients with MONET. In *Proc. 2nd USENIX NSDI*, Boston, MA, May 2005.

[5] Katerina Argyraki and David R. Cheriton. Loose source routing as a mechanism for traffic policies. In *ACM SIGCOMM Workshop on Future Directions in Network Architecture*, Portland, OR, September 2004.

[6] O. Bonaventure, C. Filsfils., and P. Francois. Achieving sub-50ms recovery upon bgp peering link failures. In *Co-Next*, 2005.

[7] Xenofontas Dimitropoulos, Dmitri Krioukov, Marina Fomenkov, Bradley Huffaker, Young Hyun, kc claffy, and George Riley. AS relationships: Inference and validation. *ACM Computer Communications Review*, 37:29–40, 2007.

[8] Teng Fei, Shu Tao, Lixin Gao, and Roch Guerin. How to select a good alternate path in large peer-to-peer systems. In *Proceedings of INFOCOMM*, 2006.

[9] Michael Freedman, Mythili Vutukuru, Nick Feamster, and Hari Balakrishnan. Geographic locality of IP prefixes. In *Proc. ACM SIGCOMM Internet Measurement Conference*, New Orleans, LA, October 2005.

[10] Lixin Gao. On inferring automonous system relationships in the Internet. *IEEE/ACM Transactions on Networking*, 9(6):733–745, December 2001.

[11] Krishna P. Gummadi, Harsha V. Madhyastha, Steven D. Gribble, Henry M. Levy, and David Wetherall. Improving the reliability of Internet paths with one-hop source routing. In *Proc. 6th USENIX OSDI*, San Francisco, CA, December 2004.

[12] Y. He, G. Siganos, M. Faloutsos, and S. V. Krishnamurthy. A systematic framework for unearthing the missing links: Measurements and impact. In *Proc. 4th USENIX NSDI*, Cambridge, MA, April 2007.

[13] H. Tahilramani Kaur, S. Kalyanaraman, A. Weiss, S. Kanwar, and A. Gandhi. Bananas: an evolutionary framework for explicit and multipath routing in the internet. In *FDNA '03: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*, 2003.

[14] Nate Kushman, Srikanth Kandula, Dina Katabi, and Bruce M. Maggs. R-BGP: Staying connected in a connected world. In *Proc. 4th USENIX NSDI*, Cambridge, MA, April 2007.

[15] Craig Labovitz, Abha Ahuja, and F. Jahanian. Experimental study of Internet stability and wide-area network failures. In *Proc. FTCS*, Madison, WI, June 1999.

[16] Harsha V. Madhyastha, Tomas Isdal, Michael Piatek, Colin Dixon, Thomas E. Anderson, Arvind Krishnamurthy, and Arun Venkataramani. iPlane: An information plane for distributed services. In *Proc. 7th USENIX OSDI*, Seattle, WA, November 2006.

[17] Ratul Mahajan, David Wetherall, and Tom Anderson. Understanding BGP misconfiguration. In *Proc. ACM SIGCOMM*, pages 3–17, Pittsburgh, PA, August 2002.

[18] Zhuoqing Morley Mao, Jennifer Rexford, Jia Wang, and Randy Katz. Towards an Accurate AS-Level Traceroute Tool. In *Proc. ACM SIGCOMM*, Karlsruhe, Germany, August 2003.

[19] University of Oregon. RouteViews. `http://www.routeviews.org/`.

[20] RouteScience. Whitepaper available from `http://www.routescience.com/technology/tec_whitepaper.html`.

[21] Justin Ryburn. Re: possible l3 issues, February 2004. `http://www.merit.edu/mail.archives/nanog/2004-02/msg00814.html`, plus private communication from list members who wished to remain anonymous.

[22] Stefan Savage, Tom Anderson, et al. Detour: A Case for Informed Internet Routing and Transport. *"IEEE Micro"*, 19(1):50–59, January 1999.

[23] R. Teixeira, K. Marzullo, S. Savage, and G.M. Voelker. In Search of Path Diversity in ISP Networks. In *Proc. ACM SIGCOMM Internet Measurement Conference*, Miami, FL, October 2003.

[24] Wen Xu and Jennifer Rexford. MIRO: Multi-path Interdomain ROuting. In *Proc. ACM SIGCOMM*, Pisa, Italy, August 2006.

[25] Xiaowei Yang. NIRA: A New Internet Routing Architecture. In *ACM SIGCOMM Workshop on Future Directions in Network Architecture*, Karlsruhe, Germany, August 2003.

[26] Xiaowei Yang, David Wetherall, and Thomas Anderson. Source selectable path diversity via routing deflections. In *Proc. ACM SIGCOMM*, Pisa, Italy, August 2006.

[27] D. Zhu, M. Gritter, and D. Cheriton. Feedback based routing. In *Proc. 1st ACM Workshop on Hot Topics in Networks (Hotnets-I)*, Princeton, NJ, October 2002.