Deconstructing Internet Paths: An Approach for AS-Level Detour Route Discovery

Sing Wang Ho * Thom Haddow * Jonathan Ledlie † Moez Draief * Peter Pietzuch *
* Imperial College London † Nokia Research

Abstract

Detour paths provide overlay networks with improved performance and resilience. Finding good detour routes with methods that scale to millions of nodes is a challenging problem. We propose a novel approach for decentralised discovery of detour paths based on the observation that Internet paths that traverse overlapping sets of autonomous systems may benefit from the same detour nodes. We show how nodes can learn about overlap between Internet paths at the level of autonomous systems and demonstrate how they can exploit detours that other nodes have already found. Our approach is to cluster paths based on the extent to which the autonomous systems traversed overlap and gossip potential detours among nodes. We find that our centralised path clustering algorithm correctly classified over 90% of potential latency detours in a 176-node dataset drawn from PlanetLab. In our decentralised version, we detected 60% of potentially available detours with each node sampling data from only 10% of other nodes.

1 Introduction

Internet routing is not well suited to satisfy the diverse quality-of-service (QoS) requirements of modern distributed Internet applications. For example, a content distribution system may prefer Internet paths that have high available bandwidth, whereas an audio conferencing application may rather use paths with low latency. Internet routing decisions made by interior and exterior gateway protocols are shared across all applications with no scope for individual requirements.

In contrast to traditional client/server systems, modern distributed applications can construct overlay networks that give them the flexibility to use multi-hop routing paths that are different from direct Internet paths. Overlay networks abstract away the direct IP connections between the nodes that constitute these applications. Instead of sending data directly to a target IP address, such as a server in the traditional model, current applications often explicitly measure the network and guide packets according to their own internal requirements. This enables them to choose custom paths with desirable QoS properties. In previous work, this *detour routing* [10] approach has been shown to improve the reliability of Internet paths [1, 3] and to reduce their latency [7], among other desirable properties. Such improvements are achieved by exploiting potential path diversity beyond the default Internet path when suboptimal performance is caused by hops other than the first or last on the default path [3].

An open issue is how applications can discover detour paths that improve a given QoS measure in a scalable and efficient manner. A common approach is to perform end-to-end measurements of a set of paths to reveal improvements caused by combining paths [10]. However, without either careful selection of which end-to-end measurements to take, or measuring almost all network paths, it is difficult to capture the full extent of what detour paths may be available. Fundamentally, discovering detour paths at the application layer remains a hard problem because applications have only limited insight into routing decisions made by lower Internet layers. The cost of a large number of end-to-end measurements may outweigh any potential benefit of detour routing.

Previous work on overlay detour routing often treated the network itself as a black box. Measurements of QoS parameters were between individual nodes, and the underlying physical and contractual location of these nodes was ignored. Participants in these types of wide-area distributed applications are, however, part of autonomous systems. Considering nodes as part of their larger environment — their autonomous systems and the routes among them — is more natural when considering network-level behaviour, such as detour paths.

This paper proposes a new approach for discovering detour paths that works at the level of autonomous system (AS) paths. By analysing detour paths at the AS level, we exploit that detours are frequently a consequence of the intrinsic behaviour of BGP routing, for example, caused by the fact that certain peering relationships are not advertised externally, or because BGP does not react effectively to link failure or congestion. We obtain Internet AS paths through *traceroute* measurements of direct paths and translate the obtained IP addresses to AS numbers. We then decompose the AS paths into pairwise *AS links* and use these links to construct *fingerprints* of clusters of direct Internet paths that share similar detour nodes. These fingerprints are a representation of direct

paths with detours and are disseminated throughout the system. This enables us to reduce the number of detour nodes that need to be stored and to discover previously unknown detours by establishing their similarity to known paths.

The main contributions of this paper are two-fold. First, we show how a hierarchical clustering algorithm can be used to construct fingerprints of sets of direct Internet paths with and without detours. Our experiments indicate that our approach correctly classifies 94.3% of paths with detours and 83.1% of paths without detours on a 176-node PlanetLab dataset. 85.3% of paths classified as having detours experience a reduction in latency by using the suggested detour node.

Second, we show how this process can be decentralised to construct an overlay network that an application can use to discover better detour paths on demand. In the overlay network, nodes perform decentralised clustering of AS paths and gossip fingerprint information among themselves to reduce measurement overheads. The decentralised execution of the clustering algorithm finds most of the potentially available detours (60%), while only requiring each node to collect a small number of measurements from 10% of other nodes.

The rest of the paper is organised as follows. In the next section, we discuss work on detour routing. In Section 3, we present the centralised version of our clustering algorithm for fingerprint generation of Internet paths, which is followed by a description of an overlay network for detour routing in Section 4. Section 5 evaluates our approach in terms of detection accuracy and detour path improvement and we conclude in Section 7.

2 Related Work

Our basic model of an overlay network is derived from research on resilient overlay networks (RON) [1]. We picture each node as part of a distributed system, performing ongoing measurements with a small set of other nodes.

Savage *et al.* [10] presented the main idea behind detour routing: that sets of specially-tuned routers could tunnel traffic in more intelligent ways than standard IP. Instead of routers themselves becoming more attuned to application traffic, distributed applications themselves have taken on the role of determining packet flows; Savage's "routers" are overlay nodes. In Andersen's RON [1], overlay nodes actively measured network characteristics to each other so that they could rapidly circumvent network failures. RON does not scale due to its n^2 inter-node measurements, but it confirms the benefits of detour routing.

Subsequent research further explored *why* direct IP routing was often poorer than alternative routes. Gummadi *et al.* [3] found that, if a detour could improve the reliability of a route at all, detouring via only a single random hop was sufficient in almost all cases. Madhyastha *et al.* [8] built iPlane, which combined continuous network measurements from many vantage points to return information about network characteristics. Like the method we propose, their approach builds a view of the network topology.

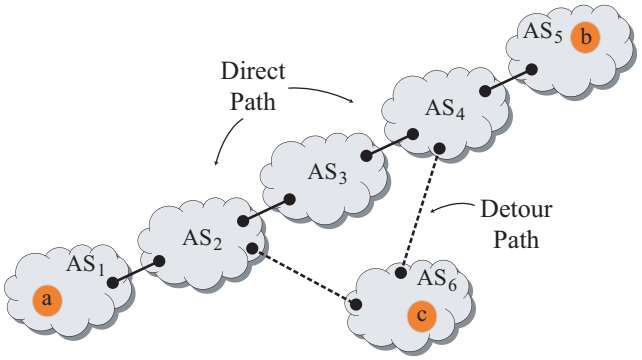

Figure 1: **Detour routing:** direct Internet path (a, b) from host a to b traversing autonomous systems $\mathrm{AS}_1, \ldots, \mathrm{AS}_5$ shown with a detour path via host c, in which AS_3 is replaced by AS_6.

Other research most similar to ours has examined *how* to select good detour routes. Su *et al.* [11] minimise the overhead of detour discovery by taking advantage of on-going measurments of commercial content distribution networks. The work by Lee *et al.* [5] uses heuristics to perform strategic measurements of detour paths that may yield higher bandwidth. The PeerWise system [7, 6] uses network coordinates to discover latency detours. In contrast to our approach, all of these techniques make assumptions about the properties of path costs. For example, PeerWise assumes that the cost measure (latency) is embedable in a virtual coordinate system and therefore cannot support non-embedable measures, such as bandwidth.

3 Autonomous System Path Deconstruction

We let the *direct path* (a, b) between two hosts a and b be the routing path across multiple autonomous systems $\mathrm{AS}_1, \ldots, \mathrm{AS}_n$ given by regular Internet routing. Figure 1 illustrates a direct path from host a to b. We let this path have a cost $\mathcal{C}(a, b)$, which could be, for example, the one-way communication latency, bandwidth or loss experienced by packets.

Detour routing exploits the fact that most measures of Internet path costs, including latency, do not form pure metric spaces and contain *triangle inequality violations* (TIVs) [13]. Therefore a *detour path* (a, c, b) that relays traffic via another node c may exhibit a lower overall cost. The inequality differs based on the cost metric:
for latency,

$$\mathcal{C}(a, c) + \mathcal{C}(c, b) < \mathcal{C}(a, b) \tag{1}$$

for bandwidth,

$$\mathcal{C}(a, b) < \min\left(\mathcal{C}(a, c), \mathcal{C}(c, b)\right) \tag{2}$$

and for loss,

$$1 - \mathcal{C}(a, b) < \left(1 - \mathcal{C}(a, c)\right)\left(1 - \mathcal{C}(c, b)\right) \tag{3}$$

In this work, we do not make any further assumptions about the cost but for simplicity of evaluation in terms of available datasets we focus on communication latency.

Previous work has shown that the bulk of improvement is obtained by relaying through a single detour node, with diminishing returns when additional detour nodes are included [7]. Therefore we focus on single-hop detours only. In what follows, we will refer to a path as a *TIV path* if there is a detour node such that the corresponding detour path exhibits a lower cost than the direct path, and a *no-TIV path* if no such detour exists.

In many cases, TIVs are the result of routing policies chosen by network operators of ASs. For example, an AS that is a customer of a transit AS will not advertise routes to the transit AS externally to avoid attracting chargeable third-party traffic. Traffic flowing through a detour node in the customer AS will be treated as local traffic and hence benefit from the transit arrangement. Another example is a transit AS, such as Internet2 in the US, that has a policy preventing it from carrying certain classes of traffic. Here a detour node can enable the external traffic of an application to be treated in the same way as local traffic, therefore leading to better end-to-end performance.

We propose an approach to discover detours at the AS-level graph. We believe this to be valid as the majority of detour routes exist at the AS level. For example, in the all-pairs traceroute dataset between 176 PlanetLab nodes described in Section 5, 96.1% of all existing detour paths include at least one different AS from the original direct path. By working with AS paths instead of IP paths, we drastically reduce the size of the data. We also benefit from the fact that AS paths are relatively static compared to IP paths. However, we assume that nodes within the same AS behave similarly in terms of externally visible network measurements.

Path clustering. Our method is based on the rationale that end-to-end paths traversing similar AS links will benefit from the same sets of detour nodes. To share potentially useful detours, we group paths that are similar into *clusters* and share potential detours within each cluster. To group sets of similar paths, we examine their *shared links*: a *link* is a particular pair of ASs along a path and a *shared link* is a link that is shared by two or more paths, as illustrated in Figure 2. We use the number of shared links between paths to determine whether they should be combined into a cluster, and the number of shared links between clusters to determine whether they should be merged into a larger cluster. Because knowing that certain paths do not have good detours is also useful, we form both clusters of TIV paths, called *TIV clusters* and of no-TIV paths, called *no-TIV clusters*.

We now give a formal description of our algorithm. Each pair of nodes a and b is represented by the set of AS links in the direct path (a, b). In Figure 2, the path (a, b) is associated with the set $\{(AS_1, AS_5), (AS_5, AS_6), (AS_6, AS_2)\}$. Let (a, b) and (c, d) be two pairs of nodes with respective AS paths $P^{a,b} = \{p_1^{a,b}, \ldots, p_l^{a,b}\}$ and $P^{c,d} = \{p_1^{c,d}, \ldots, p_m^{c,d}\}$, where $p_i^{a,b}$, $i = 1, \ldots l$ and $p_j^{c,d}$, $j = 1, \ldots, m$, correspond to AS links. We define the *similarity ratio* $d((a, b), (c, d))$ between the two AS paths by

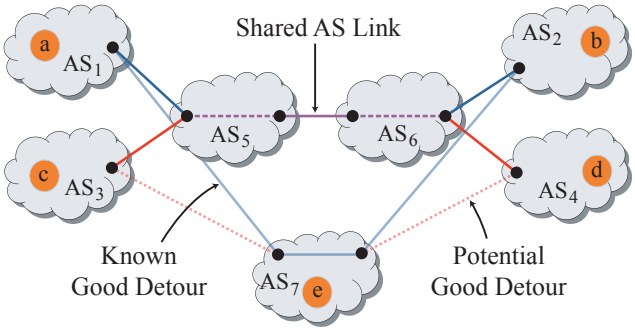

Figure 2: **Shared links:** two AS paths (a, b) and (c, d) have a shared link. Path (c, d) may benefit from using (a, b)'s detour node e. Shared links are clustered so that paths can discover each other's detours.

$$d((a,b),(c,d)) = \frac{1}{\max(m,l)} \sum_{i=1}^{l} \sum_{j=1}^{m} \mathbf{1}_{\{p_i^{a,b} = p_j^{c,d}\}} \quad (4)$$

where $\mathbf{1}$ is the indicator function. It corresponds to the proportion of shared AS links between the two paths. In the example of Figure 2, the similarity ratio between the paths (a, b) and (c, d) is $1/3$. The similarity ratio provides a measure of the closeness of two given paths with respect to their sequence of AS links and is indicative of the existence or non-existence of detour paths.

The main idea behind our algorithm consists of finding clusters of resemblant paths based on their similarity ratio. To this end, we introduce a similarity threshold $\tau \in [0, 1]$ and we let two paths (a, b) and (c, d) be similar if $d((a, b), (c, d)) \geq \tau$. In this case, these two paths are merged to form a cluster with *fingerprint* $\{P^{a,b}, P^{c,d}\}$. As shown in Figure 2, for $\tau = 0.2$, the paths (a, b) and (c, d) have a similarity ratio larger that 0.2 and are thus merged into one cluster $\{\{(AS_1, AS_5), (AS_5, AS_6), (AS_6, AS_2)\}, \{(AS_3, AS_5), (AS_5, AS_6), (AS_6, AS_4)\}\}$. In addition, as node e is known to be a detour node for path (a, b) it will be included in the list of detours for the resulting TIV cluster. In practice, shared links in clusters will only be stored once to obtain a more space-efficient representation.

To perform the clustering of AS path data, we use a variant of bottom-up hierarchical clustering. Each individual AS path is initially assigned its own cluster. Then, we iteratively merge clusters that have a high similarity ratio. At each step, we define the similarity ratio between two clusters $C_1 = \{P_1, \ldots, P_L\}$ and $C_2 = \{P'_1, \ldots, P'_M\}$, where P_i, $i = 1, \ldots, L$ and P'_j, $j = 1, \ldots, M$ are AS paths, by

$$d_c(C_1, C_2) = \frac{1}{LM} \sum_{i=1}^{L} \sum_{j=1}^{M} d(P_i, P'_j) \quad (5)$$

where $d(P_i, P'_j)$ is the similarity ratio between paths P_i and P'_j as defined in (4). Any given cluster C_1 is merged with the cluster C_2 having the highest similarity ratio, as defined by

(5), if $d_c(C_1, C_2) \geq \tau$. Note that TIV and no-TIV clusters are only merged with other TIV and no-TIV clusters, respectively.

When TIV clusters are merged, we also merge their lists of detour nodes, which are then sorted in decreasing order of their usage frequency in detours. We may envisage other schemes for maintaining the lists of detour nodes to ensure load balancing between detours as discussed in Section 6.

The similarity threshold accounts for the trade-off between having many small clusters and a small number of very large clusters. When $\tau = 1$, each cluster corresponds to an individual AS path; when $\tau = 0$ all the AS paths belong to the same cluster.

To discover a detour path for a given direct path (a, b), we first find the cluster that is most similar to it: if it is a TIV cluster, we test the two most frequently used detour nodes in the corresponding detour list, say c and d, and return the one that yields the lower cost among the detour paths (a, c, b) and (a, d, b). Otherwise, if it is a no-TIV cluster, no detour can be provided. We observed that considering more than two candidate detours for a given TIV cluster resulted in a marginal improvement but the increase from one to two was substantial.

4 Decentralised Detour Discovery

In the decentralised version of our path clustering approach, a set of n nodes cooperate to form an overlay network that is used to discover detour paths between any two nodes. Each node performs measurements between a small subset of all nodes in the network and attempts to discover detour paths within this set of nodes. Measured paths are clustered according to the approach described above. However, only TIV paths are clustered — paths for which no detours were found may still have detours outside of the set of considered nodes. This means that such paths cannot be classified reliability as no-TIV paths. TIV clusters are then exchanged between nodes using a gossiping algorithm. This spreads information about them and means that nodes can benefit from each others' measurements. In more detail, the nodes operate in four phases:

1. Measurement phase. Each node picks a set of k randomly-distributed destination nodes. It then measures its cost to these destination nodes directly and requests that they supply cost measurements between themselves. Therefore the nodes cooperate to allow each to form an all-to-all view of the costs between a small subset of nodes. This dataset is analysed to identify any TIV paths between nodes in the subset, and then the AS-level paths for these paths are recorded along with their associated detour nodes. Note that the decentralised version relies on considerably fewer measurements for detour discovery than all measurements.

2. TIV clustering phase. Each node clusters its measured TIV paths into TIV clusters with associated detours, as described above in the centralised case.

	Estimate	
	Correct	Incorrect
Have Detours (TIV paths)	94.3%	5.7%
No Detours (No-TIV paths)	83.1%	16.9%
Overall	90.6%	9.4%

Table 1: **Classification accuracy:** Percentage of paths with and without detours that were classified correctly and incorrectly.

3. TIV cluster exchange phase. After constructing its TIV clusters, a node begins exchanging them with other nodes using gossiping. In each gossip round, a node receives a set of TIV clusters with detours from another node and merges them with its own TIV clusters. TIV clusters are merged greedily by combining the most similar clusters up to the similarity threshold τ. In order to limit the communication overhead, at the cost of convergence speed, a node only gossips its original set of clusters derived in the TIV clustering phase and not the merged cluster set.

4. Detour discovery phase. To discover a detour path for a given direct path, a node finds the TIV cluster in the merged cluster set that is most similar to the given path and returns the ordered set of candidate detour nodes associated with this TIV cluster. For a direct path that does not have a detour (no-TIV path), none of the returned candidate detour nodes will provide a cost improvement and the application should use the direct path instead.

In the above, we assume that detours are static, implying that nodes do not need to re-measure paths. In an actual deployment, nodes could periodically re-execute the above four phases to update their knowledge about detours as routing paths and costs change over time. We leave an investigation of such dynamic behaviour for future work.

5 Evaluation

Our evaluation has two aims. First, we want to investigate the accuracy and quality of discovered detours using our clustering approach with complete knowledge of all measurements. Second, we want to examine how the detection efficiency decreases in a decentralised setting, when nodes do not possess complete measurements but only measure a subset of nodes and exchange information about clusters through gossipping.

Collected dataset. To carry out our experiments, we collected a dataset of all-pairs *traceroute* measurements between 176 PlanetLab nodes on Dec 10th, 2008. Each traceroute invocation sent 100 probes, spaced at one second intervals, to each host on the routing path up to the destination host. We also recorded the round-trip times to destination hosts, halving them to obtain estimates of one-way latencies. Measurements were taken concurrently over a period of 24 hours.

To obtain AS paths from traceroute measurements, we followed Mao *et al.*'s method [9] with the exception of using Team Cymru's service for AS lookups [12]. Team Cymru actively collects AS numbers from BGP tables at more than 50 vantage points. If an IP address could not be resolved by

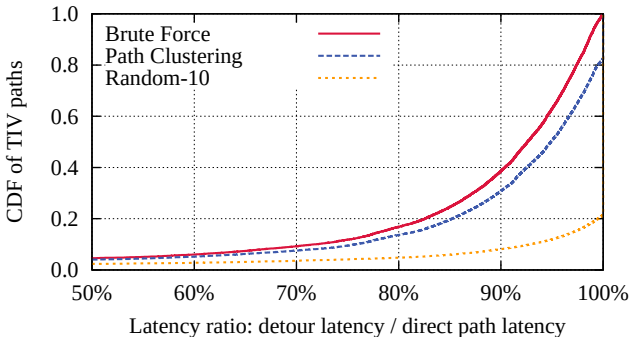

Figure 3: **Detour latency reduction (centralised):** Latencies of detours selected through PATH CLUSTERING are almost as low as latencies of detours found through brute force search.

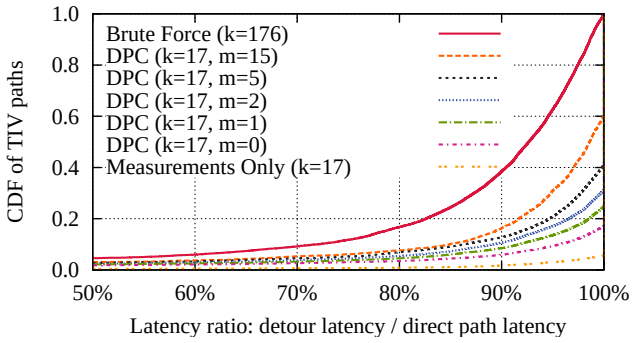

Figure 4: **Detour latency reduction (decentralised):** Even with limited initial measurements (low k value), latencies of detours in decentralised path clustering (DPC) improve as more cluster exchange takes place (higher m value).

this service, we performed a *whois* lookup. Out of 9939 encountered IP addresses, we resolved 99.58% using the Team Cymru service, 0.26% using *whois* lookups and 0.16% were reserved addresses that are unroutable on the public Internet.

We collected a total of $20,614$ paths that successfully reached their destination host, which is 66.2% of all possible paths. 69.2% of the collected paths can be improved through a detour path and 68.8% of the paths can be improved by at least 100 ms. Due to the homogeneous nature of the academic PlanetLab network [2], we believe that the number of detours and the scope for latency improvement on detour paths is lower than for a more representative subset of the Internet.

Centralised detour discovery. In our first experiment, we applied our centralised clustering algorithm to our dataset and investigated the accuracy of detour discovery and the quality of discovered detours.

We perform centralised clustering using all available paths. The best detour node is associated with each TIV path in a TIV cluster. We varied the similarity threshold τ (described in Section 3) from 0 to 1 at 0.10 increments and empirically determined the best detection accuracy to be at 0.4. With this similarity threshold, we obtained 5357 clusters in total, with 47.6% of them being no-TIV clusters. The higher proportion of no-TIV clusters compared to the proportion of no-TIV paths (33.8%) can be explained by the fact that no-TIV paths are more heterogeneous and thus less amenable to clustering.

Table 1 shows the correctness of identifying paths with and without detours. With knowledge of all measurements, our clustering approach manages to correctly classify 94.3% of all TIV paths by matching them to TIV clusters and 83.1% by matching them to no-TIV clusters. The accuracy of misclassification of no-TIV paths is higher because, as mentioned before, no-TIV clusters are more diverse and therefore easier to misclassify.

In Figure 3, we plot the distribution of detour improvements as the fraction of detour path latency over direct path latency. We compare a brute force search for the best detour (BRUTE FORCE), with our clustering approach (PATH CLUSTERING) and a random strategy that picks the best detour node out of 10 random choices (RANDOM-10). In our clustering approach, we used the two most frequently featured detour nodes per TIV cluster and picked the better one.

As shown in the graph, PATH CLUSTERING performs significantly better than the RANDOM strategy. The quality of discovered detours remains high — our approach reduces the latency of 85.3% of TIV paths as discovered by the BRUTE FORCE strategy, while having to store fewer detour nodes overall: BRUTE FORCE associates a detour node per TIV path $(20,614)$, whereas the centralised clustering approach stores two detour nodes per cluster (5357 in total).

Decentralised detour discovery. In our second experiment, we simulated the decentralised execution of our path clustering approach, as described in Section 4. Each node took a set of measurements between k other nodes, clustered the measurements and then TIV clusters were exchanged using m gossip rounds. Finding the balance between k and m is beyond the scope of this paper and we plan to investigate it in future work. Here we chose $k = 17$ as to provide reasonable coverage (10%) of the 176-node dataset, while keeping the number of path measurements gathered by each node sufficiently small. In the measurement phase, each node finds, on average, 59 TIV paths across its measurement subset, resulting in 32 clusters.

Figure 4 shows the latency reduction when using our decentralised path clustering (DPC) approach for $k = 17$ and different numbers of rounds (m) of cluster exchange. This is compared to the best-possible latency reduction as discovered by BRUTE FORCE searching the global dataset ($k = 176$), and the detours found by the original latency measurements between k nodes (MEASUREMENTS ONLY). The intersection of each curve with the right hand y-axis represents the proportion of potentially detourable routes found. With the original measurement set and no clustering, we find around 3% of possible detours. Applying clustering to this dataset (DPC, $m = 0$) increases our detour detection rate to around 8%. Each successive round of cluster exchange ($m = \{1, 2, 5, 15\}$ shown) improves detour detection, but with diminishing returns for each round. All of the curves show a fairly similar shape, suggesting the quality of detours found is consistent with those found by BRUTE FORCE.

6 Future Work

Our current implementation relies on clusters' fingerprints that incorporate complete information on the AS links of paths belonging to clusters. To reduce storage overhead, we plan to interpret the similarity between paths in graph theoretical terms. Starting from all AS paths, we can construct a *similarity graph* that consists of the set of nodes corresponding to the AS paths discovered. The existence of a link between two paths P_1 and P_2 depends on their similarity ratio $d(P_1, P_2)$. This perspective opens up various avenues for future work to refine our approach both for cluster identification and compact fingerprint construction [4].

We will also implement more elaborate schemes for detour selection that ensure load balancing between detour paths. To this end, we plan to construct detour lists that provide some diversity, as measured by our similarity ratio given by Eq. (4), in the detour paths suggested by our scheme in order to avoid overwhelming detour nodes and detour paths.

We are currently working on an implementation of our approach for detour discovery as an overlay network on PlanetLab. We want to investigate the benefits of detour routing when considering actual communication patterns used between nodes in a distributed application and study the impact of churn of detour nodes on performance. A PlanetLab deployment will also enable us to compare our approach in terms of detouring benefits and incurred costs to other systems, such as PeerWise [6] and iPlane [8].

7 Conclusions

We have presented a novel technique for discovering detour nodes to improve the end-to-end performance provided by standard Internet routing. Unlike previous work where the network is treated as a black box, we explore the AS path description of routes to enable collaborative discovery of detour nodes. To this end, we use a variant of a hierarchical clustering algorithm to partition paths into TIV and no-TIV clusters and exploit this partitioning to associate paths with detour nodes based on the similarity of their AS links. We evaluate the latency reduction when using discovered detours. Although our evaluation focused on latency, we believe that our detour mechanism can be applied to other measures, such as bandwidth and packet loss.

References

[1] ANDERSEN, D. G., BALAKRISHNAN, H., KAASHOEK, M. F., AND MORRIS, R. Resilient Overlay Networks. In *SOSP* (Chateau Lake Louise, Banff, Canada, Oct. 2001).

[2] BANERJEE, S., GRIFFIN, T. G., AND PIAS, M. The Interdomain Connectivity of PlanetLab Nodes. In *PAM* (Antibes Juan-les-Pins, France, Apr. 2004).

[3] GUMMADI, K. P., MADHYASTHA, H. V., GRIBBLE, S. D., LEVY, H. M., AND WETHERALL, D. Improving the Reliability of Internet Paths with One-hop Source Routing. In *OSDI* (2004).

[4] HARTUV, E., SCHMITT, A., LANGE, J., MEIER-EWERT, S., LEHRACH, H., AND SHAMIR, R. An Algorithm for Clustering cDNAs for Gene Expression Analysis Using Short Oligonucleotide Fingerprints. *Genomics 66*, 3 (2000), 249–256.

[5] LEE, S.-J., BANERJEE, S., SHARMA, P., YALAGANDULA, P., AND BASU, S. Bandwidth-Aware Routing in Overlay Networks. In *INFOCOM* (Phoenix, AZ, 2008).

[6] LUMEZANU, C., BADEN, R., LEVIN, D., SPRING, N., AND BHATTACHARJEE, B. Symbiotic Relationships in Internet Routing Overlays. In *Proc. of Symposium on Networked Systems Design and Implementation (NSDI)* (Boston, MA, USA, Apr. 2009).

[7] LUMEZANU, C., LEVIN, D., AND SPRING, N. PeerWise Discovery and Negotiation of Faster Paths. In *HotNets* (Atlanta, GA, Nov. 2007).

[8] MADHYASTHA, H. V., ISDAL, T., PIATEK, M., DIXON, C., ANDERSON, T. E., KRISHNAMURTHY, A., AND VENKATARAMANI, A. iPlane: An Information Plane for Distributed Services. In *OSDI* (Seattle, WA, Nov. 2006).

[9] MAO, Z. M., REXFORD, J., WANG, J., AND KATZ, R. H. Towards an Accurate AS-Level Traceroute Tool. In *SIGCOMM* (New York, NY, 2003).

[10] SAVAGE, S., ANDERSON, T., AGGARWAL, A., BECKER, D., CARDWELL, N., COLLINS, A., HOFFMAN, E., SNELL, J., VAHDAT, A., VOELKER, G., AND ZAHORJAN, J. Detour: Informed Internet Routing and Transport. *IEEE Micro 19*, 1 (January 1999), 50–59.

[11] SU, A.-J., CHOFFNES, D. R., KUZMANOVIC, A., AND BUSTAMANTE, F. E. Drafting behind Akamai (Travelocity-based Detouring). In *SIGCOMM* (Sept. 2006).

[12] Team Cymru. `http://www.team-cymru.org`.

[13] ZHENG, H., LUA, E. K., PIAS, M., AND GRIFFIN, T. G. Internet Routing Policies and Round-Trip-Times. In *PAM* (Boston, MA, Mar. 2005).