This paper was presented as part of the 14th IEEE Global Internet Symposium (GI) 2011 at IEEE INFOCOM 2011

Compact Routing on the Internet AS-Graph

Stephen D. Strowes
School of Computing Science
University of Glasgow
Email: sds@dcs.gla.ac.uk

Graham Mooney
Cisco Systems Ltd
Edinburgh, UK
Email: grmooney@cisco.com

Colin Perkins
School of Computing Science
University of Glasgow
Email: csp@csperkins.org

***Abstract*—Compact routing algorithms have been presented as candidates for scalable routing in the future Internet, achieving near-shortest path routing with considerably less forwarding state than the Border Gateway Protocol. Prior analyses have shown strong performance on power-law random graphs, but to better understand the applicability of compact routing algorithms in the context of the Internet, they must be evaluated against real-world data. To this end, we present the first systematic analysis of the behaviour of the Thorup-Zwick (TZ) and Brady-Cowen (BC) compact routing algorithms on snapshots of the Internet Autonomous System graph spanning a 14 year period. Both algorithms are shown to offer consistently strong performance on the AS graph, producing small forwarding tables with low stretch for all snapshots tested. We find that the average stretch for the TZ algorithm increases slightly as the AS graph has grown, while previous results on synthetic data suggested the opposite would be true. We also present new results to show which features of the algorithms contribute to their strong performance on these graphs.**

I. Introduction

Internet growth will ultimately lead to scalability problems for the Border Gateway Protocol (BGP) in the default-free zone of the network [1]. The fragmentation of the remaining IPv4 address space and the increasing use of IPv6 suggest future massive growth in the routing and forwarding state required to sustain Internet routing. To achieve long-term scalability of the network in the face of this growth will either require introducing further layers of indirection (such as various edge/core split schemes, e.g., LISP [2]), or investigating new routing paradigms such as compact routing.

Compact routing algorithms abandon the goal that traffic traverses shortest paths by trading off routing state for *path stretch*, where path stretch is generally a multiplicative measure of deviation from shortest path. Two widely studied compact routing algorithms are the Thorup-Zwick (TZ) algorithm [3] and the Brady-Cowen (BC) algorithm [4]. TZ defines an upper bound of multiplicative stretch-3; BC, however, defines an upper bound of additive stretch d, where d is a configurable parameter (discussed in Section III-B). Results from prior work using power-law random graphs have shown that these algorithms can achieve very low average multiplicative path stretch of around 1.1 [5]–[7]. This suggests that the actual performance of these algorithms on Internet graphs, thought to exhibit a power-law degree distribution [8], may also be considerably better than the theoretical upper bounds.

In this paper, we investigate the behaviour of the TZ and BC algorithms by systematically evaluating their performance on snapshots of the Autonomous System (AS) graph derived from RouteViews data over the 14 year period from 8 November 1997 to 8 November 2010, inclusive. We use two snapshots per year, each 6 months apart and totalling 27 snapshots of the AS graph, to provide a representative coverage of the Internet as it has evolved.

Our contributions are as follows: 1) we present the first systematic study of the performance of the TZ and BC algorithms on the AS graph, in terms of forwarding table sizes and the distribution of path stretch values, to demonstrate that these algorithms can perform as well on the AS graph as suggested by previous studies on power-law random graphs; 2) we present an analysis of the TZ landmark selection algorithm, to better understand *why* this algorithm performs so well on the AS graph, according to Thorup and Zwick's metrics, and to highlight why these metrics might not be appropriate for practical Internet routing; and 3) we present an analysis of the contribution of various components of the BC algorithm to its performance.

The remainder of this paper is structured as follows. Section II covers background material and related work; Section III outlines the TZ and BC compact routing algorithms. Section IV presents our analysis of forwarding table sizes, path stretches, and the behaviour of the TZ landmark set algorithm. Section V covers some related work on Internet measurement, and Section VI concludes the paper.

II. Background & Related Work

Typically, shortest path routing protocols are used in networks to minimise hop counts between nodes. BGP is the common protocol used to exchange and propagate the routing advertisements that allow inter-domain routing on the Internet. BGP allows local policy enforcement, but it is otherwise a path-vector protocol used to compute the shortest policy-compliant AS path length to all destinations.

Shortest-path routing requires the retention of considerable state: it has been proven that any universal routing algorithm which guarantees multiplicative path stretch of less than 3 has a lower bound on the state required at each node of $\Omega(n^2)$ [9] where n is the number of nodes in the graph. However, algorithms with an upper bound on multiplicative path stretch of 3 exist that permit sublinear forwarding state growth at all nodes. The TZ algorithm achieves worst-case multiplicative stretch 3 with a per-node upper bound on space of $O(n^{1/2} \log^{1/2} n)$.

978-1-4244-9920-5/11/$26.00 ©2011 IEEE

Previous work [5] has evaluated compact routing algorithms on power-law random graphs with node degree distributions obeying a power-law, such as the classical Barabási-Albert model of preferential attachment [10]. Degree distribution alone does not fully describe the Internet graph, however, and ignores topological features such as the clustering evident in AS topologies; it is therefore desirable to test the TZ and BC algorithms on real Internet graphs. Both algorithms were tested on two Internet topology snapshots derived from different sources in [7], but these solitary snapshots do not capture the growth or variation in complexity of the AS graph over the years.

To understand the behaviour of compact routing on the Internet graph, we evaluate these algorithms against a series of AS graph topologies derived from full BGP routing tables from the same source spanning 14 years. These tables contain the AS paths that BGP advertisements traversed en-route to the collector. Using these paths, we construct a set of adjacencies between ASes, creating a connected graph. Our aim here is to evaluate these algorithms on real-world graphs spanning many years. We further discuss our data in Section V.

III. Compact Routing Algorithms

In this section, we describe the two compact routing algorithms we evaluate in this paper. Both algorithms as described require total knowledge of the graph to label nodes appropriately before routing can take place, but that is not to say they could not be decentralised in future.

A. Routing using the Thorup-Zwick Algorithm

Given a graph $G = (V, E)$, the TZ algorithm [3] selects a set of nodes $A \subseteq V$ to act as *landmarks*, and restricts forwarding tables at all nodes to contain only the set A and a partial view of the rest of the network. A node u is retained in the forwarding table of node v where $u, v \in V$ if u is in the *cluster* C_v as specified in Equation 1, where $d(u, v)$ is the shortest-path distance between u and v, and $D(u, A)$ is the shortest-path distance between u and the closest node in the landmark set A.

$$C_v = \{u \in V \,|\, d(v, u) < D(u, A)\} \tag{1}$$

The TZ landmark selection algorithm operates as follows. The landmark set A is populated with an initial set of nodes drawn from V with uniform probability $2\sqrt{(n \log n)}/n$, where $n = |V|$. Next, the algorithm uses Equation 1 to construct a cluster for each node, and then derives a set of nodes $W \subseteq V$ whose clusters are larger than the upper bound of $4n \log^{1/2} n/\sqrt{n}$, specified in [3]. The algorithm then sets A to be the union of A and W, and repeats until the cluster for each node is smaller than the limit. The forwarding table of each node v holds entries for the union of C_v and A.

Packet forwarding under the TZ algorithm requires three pieces of information: the identifier of the destination node, the identifier of the destination's landmark node, and the next hop from the landmark to the destination. The forwarding algorithm at each node attempts to match on the destination, and forwards the packet toward the destination's landmark if the destination is not found. If the packet arrives at the landmark, then the landmark node uses the next hop information in the packet header to forward the packet toward the destination. The landmark is used by the forwarding algorithm at intermediate nodes only when the destination is not known.

It is important to understand that paths traversed by packets do not necessarily include the landmark. Indeed, paths that use the landmark imply that the *longest* possible TZ path has been taken. It is possible, and expected, that packets arrive at an intermediate node which has, through the clustering process, retained a reference to the packet's destination prior to reaching the landmark. Shortest paths are always used from the source toward the landmark until a route to the destination is found, at which point the shortest path is used from that node to the destination itself. Thus, stretch is only inflicted if the path toward a destination's landmark takes the packet off the shortest path from source to destination.

Building clusters according to Equation 1 means that for each node v, any node u adjacent to v will not be included in v's forwarding table if u is itself adjacent to a landmark. Previous analysis modifies the basic algorithm to insert entries to all adjacent nodes into forwarding tables after the landmark selection and clustering phase [6]. In this paper we look primarily at performance with all direct neighbour links in use, but we also evaluate the performance gains of this modification to the algorithm in Section IV-A.

B. Routing using the Brady-Cowen Algorithm

The BC algorithm [4] aims to improve on the worst-case performance of the TZ algorithm, specifically in power-law graphs. It provides an upper bound on *additive* path stretch of d hops, *i.e.*, maximally d hops longer than the shortest path. Using the highest-degree node as the root, d specifies the diameter of a *core* of nodes at most $d/2$ hops from this root; nodes outside the core constitute the *fringe*. The algorithm then operates as follows. First, a primary spanning tree is constructed on the full graph from the root. One spanning tree is then constructed on each connected region contained within the fringe. We refer to BC using the union of these two sets of trees in Section IV-A as the simplest BC variant, *bcn*.

Next, the algorithm identifies the minimal set of edges within the fringe that must be removed to make the fringe acyclic, and constructs for each edge e a full spanning tree that includes e itself. We refer to the union of the resulting set of trees and *bcn* as a second variant, *bce*. In the evaluation of the BC algorithm in [4], if fewer than five edges are identified, then edges in the fringe are chosen randomly from which additional spanning trees are constructed to create five trees. The number five is not explained in [4]. We refer to this third variant of the algorithm as *bch*.

Increasing the d parameter reduces the size of the set of edges identified for *bce*, but increases stretch. In this paper we set $d = 6$ as the largest value that identifies additional edges for the *bce* variant on our snapshots. We explore the properties of d on AS graphs further in [11].

Once all trees have been constructed, each node in each tree is given a Peleg distance label [12] and a TZ tree label (described in [3], and different to that described in Section III-A). Forwarding from a node u to a node v requires the following: u compares its own Peleg distance label against v's distance label to determine the tree with the fewest hops to v, then forwards packets using v's TZ tree label. All intermediate nodes use only the tree labelling.

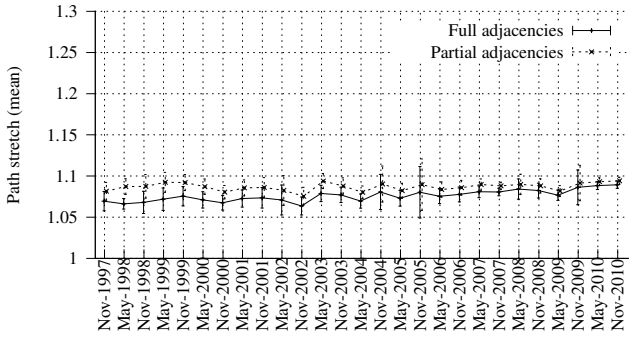
IV. Experimental Analysis

Both algorithms pre-process a full graph according to the outlines described above to generate the appropriate node labelling to permit compact routing. After pre-processing the AS graph snapshots with both algorithms, we can determine all forwarding table sizes, and all BC path lengths. However, TZ labelling contains no distance information and the path stretch incurred by TZ is dependent on the forwarding state held at intermediate nodes between source and destination. Computation of distances under the TZ algorithm is not so trivial as to compute for two nodes u and v the sum of $d(u, l_v)$ and $d(l_v, v)$, where l_v indicates the correct landmark to use to reach v. Accordingly, we simulate the forwarding algorithm to determine the number of hops between source and destination pairs. Further, paths under the TZ algorithm are not necessarily symmetrical. That is, $d(u, l_v)$ then $d(l_v, v)$ may be different in length to $d(v, l_u)$ followed by $d(l_u, u)$, and so with the TZ algorithm we test both directions.

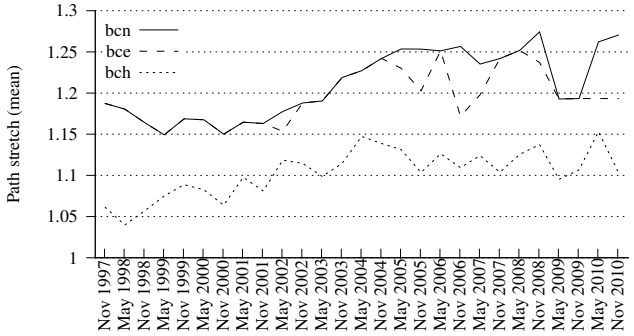
The complexity of comparing routes between all possible pairs of nodes grows as $O(n^2)$, where n is the number of nodes, and it becomes computationally infeasible to simulate the TZ forwarding algorithm on the larger graphs. Rather than exhaustively test routing between all pairs of nodes, we base our TZ results on representative subsets of the paths in each snapshot. We restrict the size of our sample sets by selecting, uniformly randomly, 1% of all nodes in each snapshot, and determine the stretch from each of those nodes to *all* other nodes in the graph. By selecting uniformly randomly, there is no bias toward high-degree nodes or low-degree nodes, and landmark nodes can also be chosen. These sample sizes are large enough that the range of observed stretches is representative of the full graph, and also large enough that we are likely to have observed a representative number of maximum-stretch paths. To help confirm that our results are consistent, we determine the path stretches for *all pairs* of nodes for the smallest of our graphs, 8 November 1997.

As the TZ algorithm builds landmark sets from an initial random selection of nodes, the algorithm is *not* deterministic. Stretch results may vary given different landmark sets. To accommodate this, we generate a representative sample of 50 landmark sets for each snapshot, and simulate forwarding on 5 of these. The 5 sets are chosen according to the size of the 50 landmark sets, so that we evaluate with varying landmark set sizes from smallest to largest.

Routing under the BC algorithm is deterministic, so path stretch values can be directly determined without simulation. We use this information to compute all-pairs BC distances.



(a) TZ means of means: Path stretches are consistently low, on average, and show little variation across landmark sets.



(b) BC means of means: Path stretches are consistently low, on average, but are more variable than TZ, and indicate an upward trend.

Fig. 1. Effect of variations in each algorithm on multiplicative path stretch.

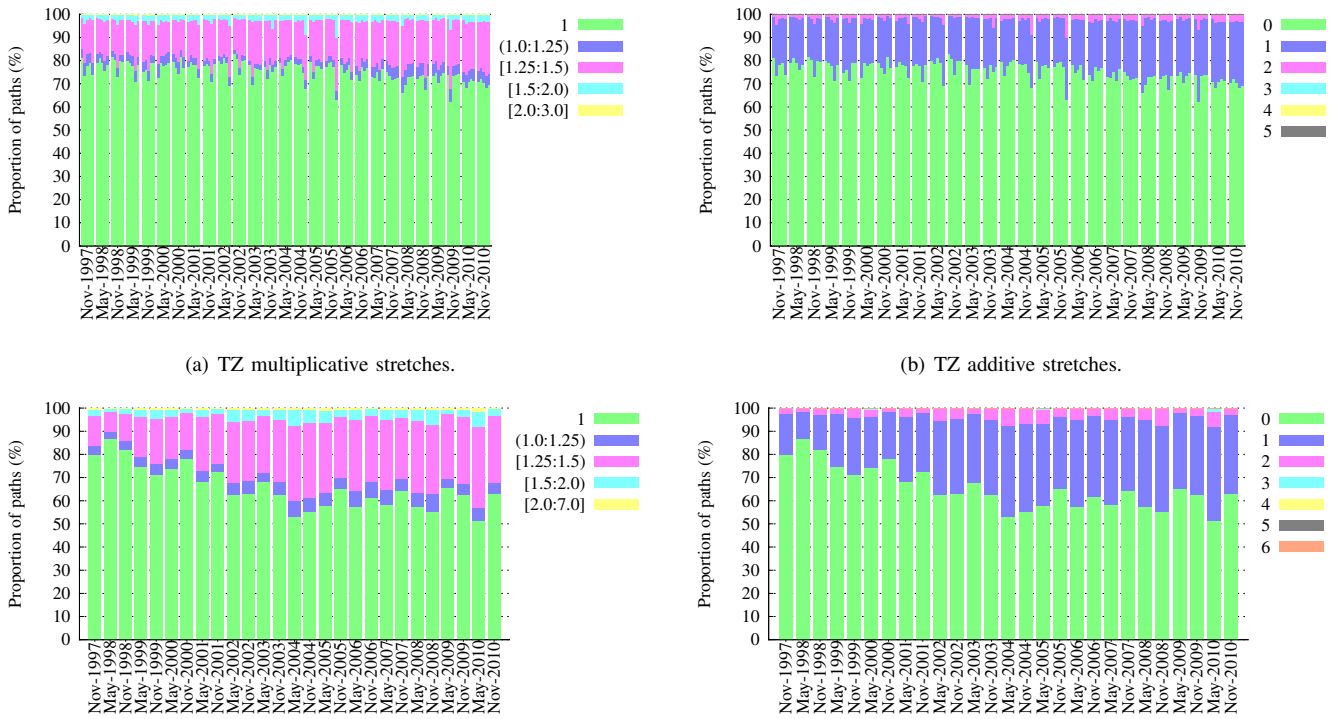
A. Path Stretch

In this section, we discuss the stretch behaviour of the different algorithms, to confirm that they perform well on these graphs. In particular, we show first the behaviour of the minor variants of the two algorithms already identified in Section III.

Figure 1(a) shows average TZ multiplicative path stretches, with and without full adjacencies. Adding missing adjacencies into forwarding tables leads to a minor improvement in performance, though this gain has narrowed over time. This, we believe, is due to the sublinear growth of the landmark set relative to the full network, and the corresponding reduction in nodes directly adjacent to a landmark. Given that full adjacencies always offer best performance, for the remainder of this paper we use this variant of the algorithm. The average multiplicative stretch in November 2010 is around 1.09.

Figure 1(b) shows the average BC multiplicative path stretches for the three variants of the algorithm from Section III. There is a clear reduction in stretch when extra trees are added in *bch* in addition to the primary tree set, *bcn*, but we can see with $d = 6$ that *bce* often offers no improvement. The performance gain observed here is simple to understand: additional spanning trees use more of the underlying links in the network, offering multiple paths to all destinations. Despite the extra links, BC performance declines slightly over time and is more variable than TZ, we believe because BC is still confined to forwarding within a small set of spanning trees.

We show actual multiplicative stretch distributions for each



(a) TZ multiplicative stretches.

(b) TZ additive stretches.

(c) BC multiplicative stretches.

(d) BC additive stretches.

Fig. 2. The multiplicative and additive stretch behaviour of TZ and BC across all snapshots.

of our TZ tests in Figure 2(a). We show the results from multiple tests on each snapshot. We see that the vast majority of paths do not exhibit any stretch at all (*i.e.*, are shortest-path); across all tests on the November 2010 snapshot, 70.2% of paths are shortest paths. 4.95% exhibit multiplicative stretch of less than 1.25; 26.54% exhibit multplicative stretch of less than 1.5, and a further 3.17% exhibit multplicative stretch between this and 2.0. Fewer than 0.01% of paths exhibit stretch higher than 2.0. It is important to realise that, given the upper bound of multiplicative stretch-3, this performance is exceptional. We see in Figure 2(b) that the path stretch is generally only one additional hop on the shortest path: 26.6% of paths in November 2010 have one additional hop; 3.14% have two additional hops.

It is claimed in [5] that the average stretch with TZ decreases as the network size increases, for power law random graphs. Our results suggest the opposite is true on Internet graphs, that average stretch has increased slightly. Most of this increase appears to come from single additional hops. This result emphasises the importance of testing these algorithms against real-world data.

In Figure 2(c), we show the multiplicative stretch for our BC tests. The results here are more variable, as expected from Figure 1(b), but in all tests the majority of paths are stretch-1 (63.21% for November 2010), and a very small proportion are above multiplictive stretch-2.0 (0.37% for November 2010). Correspondingly, Figure 2(d) shows that most stretch is incurred by one single hop.

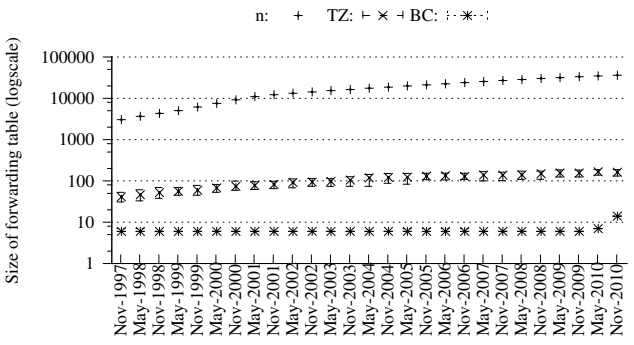
By setting the parameter $d = 6$, BC guarantees a maximum additive stretch of 6 hops. We observe additive stretch of 6 between only 381 pairs of nodes (out of over 650 million). This additive bound means 6 additional hops which, for adjacent nodes, means the maximum *multiplicative* stretch is 7. In November 2010, We observe 5,260 paths with multiplicative stretch higher than three. The majority of these would be eliminated if we were to deviate from BC's tree-based labelling algorithm and use a different form of labelling for direct neighbours.

On all of the TZ tests we ran, only two tests featured paths with five additional hops. In all other cases, four additional hops was the maximum additive strech.

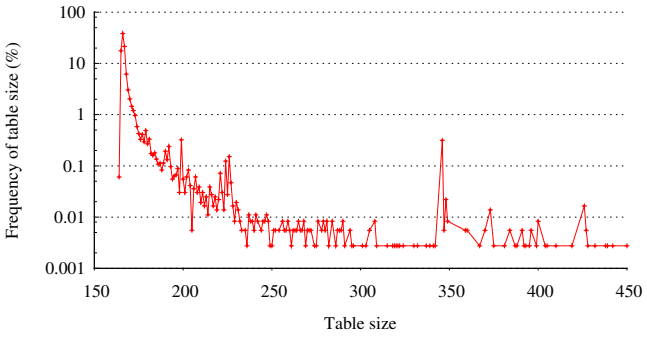
B. Forwarding Table Sizes

Figure 3(a) shows average forwarding table sizes at all nodes for each of the TZ (with full adjacenceis) and BC (*bch*) experiments we ran. The upper line on this graph is n, the number of participating ASes in the Internet for each snapshot. This has grown from 3030 in November 1997, to 36255 in November 2010.

In this time, the average size of TZ forwarding tables on these snapshots has risen from 41.13 to 162.59 entries, the latter equivalent to 0.45% visibility of the full graph. This, given the proportion of shortest paths discovered by the TZ algorithm, is extremely small. We show a representative distribution of forwarding table sizes for one of the landmark sets generated for the November 2010 snapshot in Figure 3(b).



(a) Mean routing table growth. TZ and BC values indicate means, with 1^{st} and 99^{th} percentiles. Note that all nodes here retain *full* forwarding tables according to their routing algorithms, and no default routes have been used.



(b) Distribution of TZ routing table sizes.

Fig. 3. Forwarding table growth.

As these forwarding tables retain all adjacencies, the largest of our forwarding tables are dictated by node degree, and the upper bound on the forwarding table size of a node $w \in V$ becomes, maximally, $d_w + |A| + |C_w|$, where d_w is the degree of w. Given the degree distribution of the network [8], it is not surprising that over all of the tests we ran on November 2010, only 0.0332% of nodes had forwarding tables containing more than 1,000 entries, and 91.19% have forwarding tables containing 180 entries or fewer.

The BC forwarding table sizes shown in Figure 3(a) are even smaller, as they are based on the number of trees each node is contained within. Given that each node is a member of the primary spanning tree, any other spanning trees generated across the full graph, and possibly one smaller tree connecting nodes in the fringe, the forwarding tables generated by BC show little variance. These forwarding tables are constant in size until 2010 owing to the use of $d = 6$ and the algorithm's reliance on a minimum of five spanning trees in addition to the primary tree. The 2010 snapshots make use of additional trees, indicating network growth outwith the core region that the algorithm has made use of. These forwarding tables for November 2010 contain 14 entries in over 99% of nodes, and 15 entries in the remainder.

C. Remarks on the Choice of TZ Landmarks

The performance of the TZ algorithm relies on the behaviour of the landmark selection process, in particular how it behaves on its second or later iterations.

We show in Figure 4 the frequency with which a node was selected in the 50 landmark sets generated for each snapshot. We overlay each snapshot's recurrence rate onto the same plot. We observe that the likelihood of the landmark selection algorithm selecting a node is heavily influenced by node degree, to the extent that some nodes were selected in every one of our tests. This is an unintuitive result, as the first iteration of this algorithm randomly selects nodes from the graph irrespective of node degree.

However, Equation 1 states that the cluster for each node u contains all nodes nearer to u than to any members of the landmark set. Since high-degree nodes are intrinsically 'near' to much of the network, they are disproportionately likely to have clusters large enough to break the constraints specified in the equation, so they are therefore often selected for the second round of the landmark selection algorithm. The node degree explains the frequency of selection.

We believe this aids the stretch performance of the algorithm. As stated in Section III-A, if the path toward a destination's landmark does not deviate from the shortest path to that destination, no stretch will be inflicted. As high degree nodes are connected to a large proportion of the network, many paths already naturally use these nodes for transit as the shortest path toward a destination. This is beneficial: this level of connectedness increases the likelihood that this landmark is already on a shortest-path to many destinations.

As we treat these snapshots as undirected graphs, this landmark selection works very well (as is also exploited by [13]). However, the Internet is not an undirected graph, and ASes implement various policies over certain links implying that some ASes may not form part of a valid path between certain sources and destinations. That is, node degree alone is not a useful mechanism to select landmarks for the Internet, and the particular landmarks selected here may not be willing to operate as such in the real-world.

These landmark sets are small, as shown in Figure 5, suggesting that a similarly-small, policy-compliant set of transit networks may be discoverable. This set has grown with the network, as we may expect, and the variation in landmark set size has grown linearly with the size of the network.

V. Accuracy of Work

Our basis for using AS numbers rather than address prefixes in this paper is that, after the prefix length (and local, often manually configured, policy decisions), the AS path length is the primary routing metric in BGP.

There are some limitations in the data that affects all work of this nature. The set of passive BGP collectors utilised by routeviews provide an important source of data, but the data may not be complete [14], [15]. Links missing from the data are primarily "peering" links, generally settlement-free (*i.e.*, "no-cost") links between networks with a shared provider to avoid the costs of using the provider network. We should note that, as in the TZ work presented here, nothing is lost by

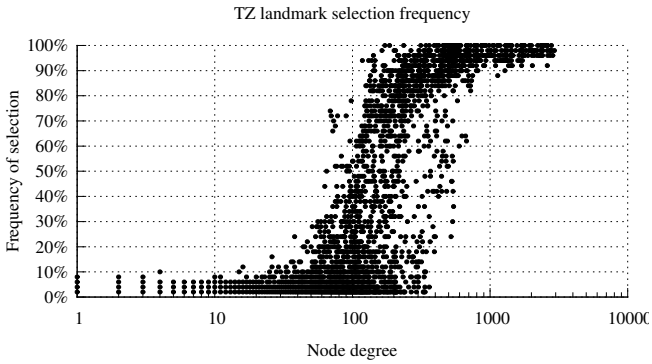


Fig. 4. TZ landmark selection frequency. This plot overlays the selection frequency for each snapshot tested (*i.e.*, how often a landmark appears in each of the fifty landmark sets.)

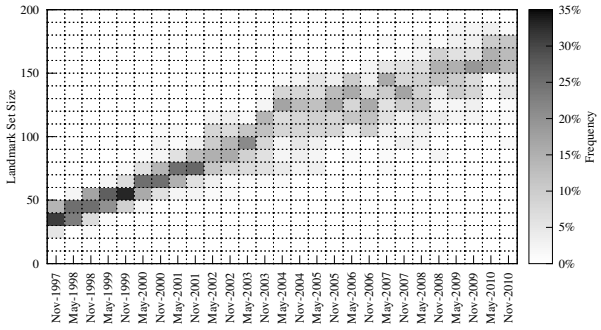


Fig. 5. Distribution of landmark set sizes.

retaining forwarding state to neighbouring nodes and a partial view of the rest of the network.

Revealing more of the underlying AS-level structure of the Internet is an active research area. There is scope for aggregating more paths from all other repositories, such as the BGP data RIPE offers access to. There is also other work on performing active measurement, using traceroute to reveal paths not visible in the BGP data [16]. Evaluating compact routing algorithms on augmented graphs such as these is an important next step.

VI. Conclusions and Future Work

We have shown that these compact routing algorithms do perform well on the Internet graph, and have continued to perform well through the massive growth of the network. This is the first systematic analysis of the TZ and BC algorithms on the AS graph. As highlighted in Section V, there is further scope for evaluating the algorithms with more accurate models of the AS graph, should they become available.

The deployability of these algorithms as network protocols requires additional work. For example, the TZ algorithm relies on a global view of the network and grows its landmark set from an initial random selection of nodes. In the Internet, a stable set of centrally located transit networks acting as landmarks would be more beneficial than a set of arbitrarily-chosen networks. Definition of such a set, and a routing protocol that uses this set, is left as future work.

Policy-based routing, and the ability to make local decisions on link utilisation, or to use preferred links (for cost purposes, etc), is considered an important aspect of network management. The algorithms we have evaluated do not make allowances for policy, though there is still scope for affecting policy on routes within clusters, and routes toward landmarks. We also cannot easily evaluate against the path inflation introduced by policy-based routing in BGP, as such information is generally considered proprietary. It may be possible, however, to begin evaluating this using publicly available network maps annotated with inferred AS relationships [17], [18].

We believe that these algorithms offer scope for further work. If either algorithm can be decentralised for use in a dynamic network, the potential reduction in forwarding state is massive. Consider that a full BGP table for November 2010 contains 338,988 IP prefixes and that the TZ algorithm requires that, on average, nodes retain forwarding entries to 0.45% of the rest of the network, then a TZ-based protocol would provide forwarding tables of 1,521 entries on average. This is a rather simplistic mapping, but the savings are profound. Of the two algorithms shown here, TZ is the most stable and also potentially most easily decentralisable. We believe it offers a genuine future direction for further research.

Acknowledgements

This work was supported in part by the UK EPSRC.

References

[1] D. Meyer, L. Zhang, and K. Fall, "Report from the IAB Workshop on Routing and Addressing," RFC 4984, September 2007.
[2] D. Meyer, "The Locator Identifier Separation Protocol (LISP)," *Internet Protocol Journal*, vol. 11, no. 1, March 2008.
[3] M. Thorup and U. Zwick, "Compact routing schemes," in *SPAA*, 2001.
[4] A. Brady and L. Cowen, "Compact Routing on Power Law Graphs with Additive Stretch," in *ALENEX*, 2006, pp. 119 – 128.
[5] D. Krioukov, K. Fall, and X. Yang, "Compact Routing on Internet-Like Graphs," in *INFOCOM*, vol. 1, March 2004, pp. 219–229.
[6] D. Krioukov and kc klaffy, "Toward Compact Interdomain Routing," *CoRR*, vol. abs/cs/0508021, 2005.
[7] D. Krioukov, kc klaffy, K. Fall, and A. Brady, "On Compact Routing for the Internet," in *ACM SIGCOMM CCR*, vol. 37, no. 3, July 2007.
[8] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law Relationships of the Internet Topology," in *ACM SIGCOMM*, 1999, pp. 251–262.
[9] C. Gavoille and M. Gengler, "Space-Efficiency for Routing Schemes of Stretch Factor Three," *J. Par. Distrib. Computing*, vol. 61, no. 5, 2001.
[10] A.-L. Barabási and R. Albert, "Emergence of Scaling in Random Networks," *Science*, vol. 286, no. 5439, pp. 509–512, Oct 1999.
[11] G. Mooney, "Evaluating Compact Routing Algorithms on Real-World Networks," Master's thesis, Dept. Comp. Sci., Univ. Glasgow, 2010.
[12] D. Peleg, "Proximity-Preserving Labeling Schemes and Their Applications," in *LNCS*, 1999, vol. 1665.
[13] W. Chen, C. Sommer, S.-H. Teng, and Y. Wang, "Compact routing in power-law graphs," in *23rd Intl. Symp. Distributed Computing*, 2009.
[14] R. Oliveira *et al.*, "The (in)Completeness of the Observed Internet AS-level Structure," *IEEE/ACM Trans. Networking*, vol. 18, no. 1, Feb. 2010.
[15] M. Roughan *et al.*, "Bigfoot, Sasquatch, the Yeti and Other Missing Links: What We Don't Know About the AS Graph," in *IMC*, 2008.
[16] K. Chen *et al.*, "Where the Sidewalk Ends: Extending the Internet AS Graph Using Traceroutes From P2P Users," in *CoNEXT '09*, 2009.
[17] "AS relationships," http://www.caida.org/data/active/as-relationships/.
[18] X. Dimitropoulos *et al.*, "AS Relationships: Inference and Validation," *ACM SIGCOMM CCR*, vol. 37, no. 1, pp. 29 – 40, 2007.