On the Predictive Power of Shortest-Path Weight Inference

Andrew Coyle
University of Adelaide
SA, Australia
andrew.coyle@adelaide.edu.au

Miro Kraetzl
Dept. of Defence
Canberra, Australia
miro.kraetzl@defence.gov.au

Olaf Maennel
Tech. Universität Berlin
Deutsche Telekom Labs
olaf@maennel.net

Matthew Roughan
University of Adelaide
SA, Australia
matthew.roughan@adelaide.edu.au

ABSTRACT

Reverse engineering of the Internet is a valuable activity. Apart from providing scientific insight, the resulting datasets are invaluable in providing realistic network scenarios for other researchers. The Rocketfuel project attempted this process, but it is surprising how little effort has been made to validate its results. This paper concentrates on validating a particular inference methodology used to obtain link weights on a network. There is a basic difficulty in assessing the accuracy of such inferences in that a non-unique set of link-weights may produce the same routing, and so simple measurements of accuracy (even where ground truth data are available) do not capture the usefulness of a set of inferred weights. We propose a methodology based on *predictive power* to assess the quality of the weight inference. We used this to test Rocketfuel's algorithm, and our tests suggest that it is reasonably good particularly on certain topologies, though it has limitations when its underlying assumptions are incorrect.

Categories and Subject Descriptors

C.2.3 [**Computer-Communications Networks**]: Network Operations—*network monitoring*

General Terms

Measurement

Keywords

Topology, Inference, Routing

1. INTRODUCTION

Internet providers are (quite reasonably) unwilling to share information about their networks. The information (such as topology, routing policies and so on) would be very useful to network researchers, but also of great interest to their competitors. Nevertheless, some example networks are required for much of network research. Such data also holds great interest for scientists interested in properties of large networks (or graphs).

The Rocketfuel project [1, 2] suggested techniques for "reverse engineering" the Internet. Spring *et al.* [1] performed extensive traceroute studies to determine the router- or Point-of-Presence (PoP) level topologies of ISPs (Internet Service Providers), including estimates of Interior Gateway Protocol (IGP) weights [2]. Such weights are useful for a number of reasons, for instance as a basis for comparison when studying weight optimization [3–5].

The Rocketfuel topologies and weights have been of such great use that they have become the basis for many recent networking papers, so it may seem somewhat surprising then that very little validation has been performed. Some work [6, 7] has questioned the accuracy of the network graphs provided by Rocketfuel. However, little work has considered systematically the question of weight inference. The lack of validation is problematic because the inference problem is underconstrained: the possible solutions are non-unique, and identified only through the use of additional side-assumptions about the behaviour of "typical" weights. The side-assumptions are known to be wrong in some instances, but until now there has been no understanding of the impact this may have on the inferred weights.

The problem is complicated by the fact that direct comparisons of two sets of links weights tells us little about the usefulness of a set of inferences. For instance, one could scale links weights by a constant factor and obtain a completely equivalent set of weights. The fact that the inferred links weights can thus be made arbitrarily "inaccurate" has little impact on their usefulness for network experiments. More subtle cases are common: the change of a weight can have no consequence or may cause dramatic rerouting. So even if one knew the true value of the link weights, we still require a new way to assess the value of results. A naive approach would be to assess inferred weights on how closely the paths chosen by the resulting algorithm match those of the real routing. However, inference techniques can be designed to ensure this condition, but still be wildly inaccurate.

We take the approach of considering the *predictive power* of the inferred weights. That is, we use the inferred weights to predict network behaviour after a change in the network occurs, and compare the prediction with real behaviour. The aim is to find whether the weights are useful, rather than "accurate". A *useful* set of weights can be used to make predictions about the behaviour of a network. We concentrate on the ability to predict routes that we didn't observe in our measurements, or to predict rerouting after link failure.

The problem is actually more general than just the inference of IGP link weights. In general, we are trying to infer routing policies. IGP weights are a simple example, but the inference of policies can extend to estimation of inter-domain routing policies. Inter-domain policy inference has been considered in a number of papers [8–11]. For instance, Mühlbauer *et al.* [11] propose to use the inferred routing policies to answer "what if" questions to aide operators in making decision about inter-domain routing configuration. Answers to such questions could be very useful to network operators, and are hard to obtain without inference.

Routing weight inference lies in the area of *network tomography*. Tomography problems in general are formulated in such a way as to infer desired information from a set of indirect mea-

IMC'08, October 20–22, 2008, Vouliagmeni, Greece.

surements. The canonical examples of network tomography are traffic matrix inference [12, 13], link performance estimation [14], and topology inference [15]. Routing policy inference displays distinct similarities to these, in particular, to traffic matrix estimation. The problems are both linear inverse problems, and they are both highly underconstrained. In link-weight inference we write a set of linear constraints that express the fact that the observed routes are shortest-paths. The constraints represent the measurable quantities (in traffic matrix estimation these would be the link load measurements, whereas here these are path measurements). As in traffic matrix inference, there are (typically) an infinite number of possible feasible solutions to these constraints. We choose one solution by selecting a single "best" set of weights by solving an optimization problem, where the objective function expresses our *prior* beliefs about IGP weights. Medina *et al.* [12] showed that in the traffic matrix inference the quality of the solution was critically dependent on the quality of the prior. However, it is known [16, 17] that at least one network operator systematically varies its weights away from the priors used by Rocketfuel. So we need, at the very least, to understand how robust route-weight inference is to errors in these priors.

We test our approach on a variety of networks: the Rocketfuel networks themselves plus two real networks for which we have precise routing information (Abilene and GEANT). We test the distance proportionality prior against a number of different weight scenarios varying from the real (or hypothetical weights in the case of Rocketfuel), to sets of weights generated by a variety of synthetic techniques intended to test the range of possible responses.

2. WEIGHT INFERENCE

A good deal of past research has considered topology inference. In this paper our focus is router- or PoP-level network structure. There are several approaches to topology measurement, but for the purpose of assessing route weight inference we shall assume that this has already been done, and we know the topology, and the routes used across a significant part of this topology.

Many modern networks use SPF (Shortest-Path First) routing internally. SPF routing should not be confused with shortest-geographic-distance routing — in most SPFs the notion of "path-length" has been generalized to allow links to have an arbitrary *link weight* that might have little to do with geographic distances.

Let us describe the problem formally: the network is a graph $G = (N, E)$ where the set N denotes the nodes, and E the edges. Then we assign link weights $w_e \geq 0$ to the edges of the network. A SPF calculation seeks to find the paths which minimize the sum of link weights along the paths, i.e., it finds $\hat{\mu}_{ij} = \underset{\mu \in P_{ij}}{\operatorname{argmin}} \sum_{e \in \mu} w_e$, where P_{ij} is the set of all loop-free paths between i and j.

Weight inference refers to the problem of inferring the network weights w_e from a set of indirect measurements. An observation of a route implies that it is shorter than alternative routes. Therefore the sum of the weights of this specific path is known to be no more than the weights of all alternate paths. The observed routes therefore imply a set of constraints on the link weights from which we must select a feasible set of weights. More formally, given that an edge from node i to j (with latency d_{ij}) has weight w_{ij} we could easily determine the shortest-paths. We could also determine the complete set of paths P_{ij}, and for each such path we can write an inequality

$$\sum_{e \in \hat{\mu}_{ij}} w_e \leq \sum_{e \in \mu} w_e, \quad \forall \mu \in P_{ij},$$

where $\hat{\mu}_{ij}$ is the observed path from i to j. The number of paths between a source and a destination grows exponentially with network size, and hence so does the number of constraints, but can reduce their number by pruning [2] redundant constraints.

The actual weights of a network must be a feasible solution to the above set of constraints. However, there are multiple feasible solutions to these constraints. For example, all weights from a solution could be scaled up without affecting the routing in the network. There are a number of approaches to choose a single solution from this set of feasible solutions. In Rocketfuel, the most recent approach [2] was to choose the set of weights closest to the link latencies (which are in turn proportional to the link's length). This "distance proportionality" represents a prior model for the weights, and in effect this paper presents a test of how important this prior is in the inference process, because we know that it is not always used, e.g., Cisco's default setting for IGP weights is *inverse-capacity*.

The distance proportionality prior can be incorporated in our inference by performing an optimization: minimize the absolute value of the difference between the distances and the link weights under the constraints imposed by our shortest-path route measurements. We can write the problem as an Linear Program (LP) as follows:

$$\begin{aligned}
&\text{minimize} \quad f = \sum_{e \in E} \varepsilon_e, \\
&\text{subject to} \\
&\quad w_e - \varepsilon_e \leq d_e, \quad \forall e \in E, \\
&\quad w_e + \varepsilon_e \geq d_e, \quad \forall e \in E, \\
&\quad \sum_{e \in \hat{\mu}_{ij}} w_e \leq \sum_{e \in \mu} w_e, \quad \forall i, j \in N, \text{ and } \forall \mu \in P_{ij}, \\
&\quad w_e, \varepsilon_e \geq 0, \quad \forall e \in E,
\end{aligned}$$

where d_e is the link latency in milliseconds, ε_e is the absolute difference between the link weight and the latency, $\hat{\mu}_{ij}$ is the observed path from i to j, and P_{ij} is the set of all loop-free paths between i and j. In effect the optimization attempts to minimize the sum of the absolute differences $\varepsilon_e = |w_e - d_e|$.

In some networks, weights may be configured such that there are multiple equal-cost paths between a source and destination. Such paths complicate both measurement and inference, though note that our constraints explicitly allow for equal cost paths.

3. PREDICTIVE POWER

Two sets of weights may be quite different, and yet still result in the same routing. For instance consider the network shown in Figure 1, in which we see two quite different sets of possible weights (shown as bold, and italic text), however, they produce the same routing, both under normal conditions, and under single link failures. If one set of weights was real, and the other inferred, we might legitimately criticise the *accuracy* of the inference. However, the difference is unimportant for all practical purposes. We say that the real and observed weights are *predictively equivalent* in the sense that both predict exactly the same routing.

It should be noted that weight inference will automatically result in a set of weights that are consistent with observed routing, and so predictively equivalent for all *observed* routes. Hence predictive equivalence is only interesting when defined with respect to unobserved paths or abnormal conditions. In this paper we will consider two cases: *predictive equivalence* of

1. unobserved paths (where not all paths are measured),
2. route changes under single link failures.

In many cases two sets of weights will not be exactly equivalent, and we compare them by measuring the proportion of routes that are correctly inferred, i.e., $S = |C|/|U|$, where $C \subset U$ is the subset of correctly inferred paths out of the set of paths U that we seek to predict. We call this the *predictive power* of the weights. Where we measure S with respect to the proportion of measured paths r we will write $S(r)$, and we will perform multiple experiments removing data in random orders to obtain averages $\bar{S}(r)$.

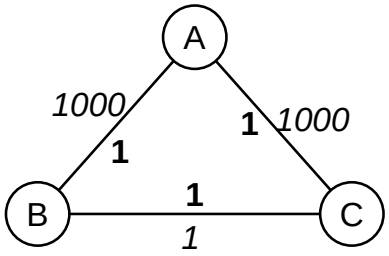


Figure 1: Two sets of equivalent weights. The bold and italic weights give the same routing under normal conditions, and all single link failures. These weights are predictively equivalent.

When considering single link failures we consider the failure of a particular link e and denote the predictive power by Q_e. Under normal routing conditions a set of routes R_e will use link e. When link e fails, these will be rerouted on alternative paths (assuming such exist). Then $Q_e = |C_e|/|R_e|$, where $C_e \subset R_e$ is the set of alternative paths that are correctly predicted by the inferred weights (the notation Q and R are used instead of S and U in order to keep the two sets of results clearly separated). We define the average value by $Q = \frac{1}{|E|} \sum_e Q_e$.

Predictive power is more complicated where Equal-Cost Multiple Paths (ECMP) exists. In this case, we present two metrics. The first requires that the sets of paths observed, and predicted should be identical. The second (which we denote by S' and Q') tests whether the set of predicted paths is a subset of the true ECMP set of solutions. This metric is perhaps more useful for practical characterisations of the results, though it is sometimes interesting to see where the two differ.

4. DATA

In order to test predictive equivalence we need networks with known topology and weights. We have access to two such networks: Abilene and GEANT. However, both networks are academic networks, and relatively small in scale (in terms of numbers of routers and links). In addition, the weight choice in Abilene is very simple (and easy to infer), and so this network shows us only that the route weight inference performs well on a network where the prior is true.

In order to have a larger selection of networks to test we use Rocketfuel data. We work with the PoP-level topologies. We considered the large group of Rocketfuel networks shown in Table 3 (indexed by the AS number, though note that we do not assert that these are truly accurate renditions of these particular networks), but in order to see more detail we will focus on two networks.

The Rocketfuel data have the problem that the real weights are not publically known. We also wish to test the impact of incorrect priors on the inference process. Hence we need to generate sets of simulated weights that have some realistic properties of weights, and yet are not necessarily correlated with distances. Our first approach is to start with a network topology (in the examples we use Rocketfuel's inferred topology for AS 1). We then use a simple gravity model to generate a random traffic matrix for the network. This approach was validated in [18], which showed that such a matrix could have some properties in common with real traffic matrices. We make no attempt to match the traffic matrix to the topology, as in [16], because we need a traffic matrix that is not fitted exactly to the network in question so that our traffic engineering step (where we determine the weights) will have a greater impact. We determine the weights by solving an optimization problem: namely, we solve the optimal weights traffic engineering problem, where the set of weights that balance the load best are chosen. There are several good approaches to solving such problems (e.g., see [3–5]), and we use a genetic algorithm based approach similar to that of [4]. The resulting set of weights has some aspects of randomness due to the randomness of the input traffic matrices. However, we know that many operators will be performing a similar operation in determining their weights, even though in many cases it may be performed in an ad hoc manner by humans attempting to balance traffic by hand. Hence these weights give us a reasonable way of generating multiple simulation scenarios.

The resulting weights appear to be quite random. For instance, Figure 2 shows the mean value of the weight for each link, along with 95th % Confidence Intervals (CIs) for the mean estimates, and minimum and maximum values for the weight on each link (over 100 simulations) vs distances of the links. Note that we have constrained the weights to lie in the range $[1, 255]$ in the above optimization, and that the mean values largely fall around half way through this range (shown as a dashed line), with a few exceptions where they appear to systematically vary from the average. Also, the weights appear to vary over almost the entire range of possible values as shown by the minimum and maximum values shown in the figure. Most importantly, the weights have little correlation to the geographic distances in the network. The average correlation coefficient over the 100 simulations conducted above is 0.05 indicating only insignificant correlation with distance, which is visually confirmed in Figure 2.

The next step away from the distance proportionality prior is to take weights which are intrinsically uncorrelated with distance, for instance, unit weights. Given every link has a weight equal to one, shortest-path routing reverts to minimum hop routing. However, strict unit-weights results in many equal-cost shortest paths, and so we also consider a case with unit weights with a small random jitter added to change the weights slightly away from one and separate the equal cost paths.

A more extreme approach is to generate a network in which the routing is essentially a tree. In this approach, we generate weights to be either 1 or 100 and choose the links with $w_e = 1$ to form a spanning tree of the network. We will refer to this approach as a *backbone* network because it resembles what might happen if a network provider were to build a tree-like high-capacity backbone and then create additional low capacity links for redundancy. In such a network, the ISP might wish the low-capacity links to be used only in the event of a link failure, and so would give them much higher weights. This is a rather extreme variation from the distance related weights because the resulting network will generally shunt traffic over this backbone regardless of geographic distance. This approach also has the advantage that we can draw some idea of the behaviour of inter-domain routing, which has been modelled using minimum-hop with filtering rules [11]. The filtering rules might be modelled by infinite weights (that restrict the use of certain routes), while otherwise minimum hop-length paths are used because weights are equal elsewhere. We keep our large weight finite because otherwise our test networks can become disconnected under some failure scenarios, but we choose a value (100) large enough that these links would not normally be used.

We generate a set of backbone networks by taking unit weight networks, adding jitter to the weights, and then finding the minimum spanning trees. The links on the minimum spanning tree are given weight one, and the other links are given weight 100. We do not contend that these networks are actually realistic network designs (though they do reflect some of the lessons learnt in [16, 17] regarding the use of large weights to create backup links). The networks are primarily intended to generate a network with weights as far from the Rocketfuel prior as is possible.

5. RESULTS

Network	weights				
	given[1]	unit	unit + jitter	synthetic	backbone
Rocketfuel AS 1	92.0%, 97.3%	86.9%, 95.3%	83.7%, 95.5%	82.7%, 92.9%	69.9%, 78.3%
Rocketfuel AS 1239	86.0%, 96.6%	90.7%, 96.4%	88.8%, 96.6%	67.9%, 92.9%	65.9%, 74.2%
GEANT	83.9%, 91.5%	86.1%, 95.4%	86.6%, 94.4%	81.2%, 90.3%	63.0%, 67.8%
number of simulations	30	30	100	100	100

Table 1: Predictive equivalence ($\bar{S}$ and $\bar{S}'$ values shown for each case) for the various networks with incomplete data (5 routes are missing in the dataset). (1) 'given' refers to either the hypothetical weights in the Rocketfuel data, or the real weights in GEANT.

Network	weights				
	given[1]	unit	unit + jitter	synthetic	backbone
Rocketfuel AS 1	94.2%, 94.4%	98.8%, 99.9%	98.7%, 99.2%	90.8%, 90.9%	68.1%, 69.5%
Rocketfuel AS 1239	86.1%, 89.9%	83.7%, 100.0%	93.3%, 94.1%	59.3%, 59.8%	23.5%, 27.3%
GEANT	82.5%, 87.8%	95.1%, 99.7%	93.2%, 94.2%	74.6%, 74.7%	31.0%, 35.5%
number of simulations	$\|E\|$	$\|E\|$	$10 \times \|E\|$	$10 \times \|E\|$	$10 \times \|E\|$

Table 2: Predictive equivalence ($\bar{Q}$ and $\bar{Q}'$ values shown for each case) for the various networks under single link failures. (1) 'given' refers to the weights given either in the Rocketfuel data, or the real weights in GEANT.

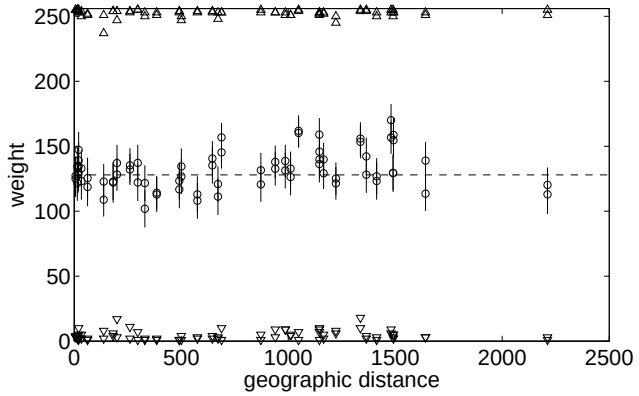


Figure 2: Relationship between weights and distance showing the mean (o), 95th % CIs (lines), max. (△) and min. (▽).

We first test our algorithm on the Abilene network where the weights are known. This is a diagnostic test, because the Abilene network (at the time of measurement) used weights which were proportional to geographic distances. The algorithm performs correctly, predicting these weights correctly (up to a scale factor). We do not need to test power equivalence on this network as the two weights are equivalent, and we will omit Abilene results from the remainder of the paper.

We will investigate three networks (GEANT and Rocketfuel ASs 1 and 1239) in more detail. Tables 1 and 2 summarize the results.

Table 1 shows the average predictive power $\bar{S}$ for unobserved routes for each network and each weight scenario, where routes associated with five links are missing (the choice of the value five is explained later). The table also shows the number of simulations upon which these results are based. There are two results per network/weight scenario. The first of these shows the average proportion of exactly inferred sets of paths $\bar{S}$, and the second shows the average proportion where the inferred set of ECMPs is a subset of the actual set, i.e., $\bar{S}'$.

The results show that for many cases the route weight inference is highly effective. $\bar{S}'$ values above 90% are common, and $\bar{S}$ values are typically above 80% (requiring exact matches for all ECMPs seems to degrade the results by 5-10% in most cases). The exception where the results are noticeably worse is with the backbone weight scenarios. As noted, it is in this case that the network weights most clearly violate the prior assumption of the inference technique, and so it is not surprising to see these results. What is perhaps surprising is that the results are not terrible. We still obtain $\bar{S}'$ values around 70%, which is not fantastic, but may still be at a level that is useful. More importantly, even in cases such as unit weights (which have no distance term), or synthetic weights (which we showed were not correlated with distances), we get quite good predictive power. In fact, in some cases the predictive power is better for unit weights than it is for the real weights (e.g., see GEANT — note that although these GEANT weights are not directly proportional to distance, there is a correlation between distance and the real weights). These results are very promising.

Table 2 shows the average predictive powers $\bar{Q}$ and $\bar{Q}'$ for single link failures. In these cases we created the same number of networks as for prediction of unobserved routes, but run one test case considering the failure of each link in each network, so the total number of results being averaged in the value reported is dependent on $|E|$, the number of edges in a network (see Table 3 to find the number of edges.). The results in this table support the previous results in that they show that quite good predictive power can be obtained over a range of networks and scenarios. However, the results are far worse for the backbone network weights. In particular, for AS 1239 and GEANT the predictive power is poor. We will further discuss reasons for this below where we consider the detailed results.

5.1 Detailed results

5.1.1 Given weights

In our first comparison, we consider a wide selection of the Rocketfuel networks using the hypothetical weights from Rocketfuel. Our estimation algorithm is similar to Rocketfuel, and so we expect to get good estimates. We do not know real weights, so this test simply compares the differences between our approach, and the Rocketfuel's. Table 3 shows the values of $\bar{Q}$ and $\bar{Q}'$. As expected, the results are good, though there are small errors as we use implicit routing information rather than actual traceroutes. These results also reveal some of the basic properties of this approach. Typical values are above 80%, but many are greater than 90%, with an average of 95%. When we consider $\bar{Q}'$ we note that in many cases $\bar{Q} \simeq \bar{Q}'$ as ECMP has little impact on the network, but in a few cases, $\bar{Q}'$ is somewhat larger. We considered a number of factors that could influence the quality of the results. We considered the correlation coefficients of the errors $1 - \bar{Q}$ with the number of nodes $|N|$, edges $|E|$, and the average node degree $|E|/|N|$, with values 0.32, 0.29 and 0.50, respectively. The average node degree has the largest impact on the quality of the results. The results are

not conclusive. There are clearly other factors at work, but the fact that there is any reduction in quality appears counter-intuitive. A higher node degree leads to more paths and more constraints, which we might assume improves the quality of estimates. However, in fact, given more possible alternative paths it is harder to correctly predict the rerouting after a failure.

The computation times (on a 1.8 Ghz Intel PC) for the optimization problems (after pruning) are also shown in Table 3 with respect to the number of edges in the network. The CPU time is well approximated as cubic $O(|E|^3)$.

ASN	$\|N\|$	$\|E\|$	$\|E\|/\|N\|$	Q (%)	Q' (%)	CPU (ms)
1	24	74	3.1	94.2	94.4	20
174	22	112	5.1	86.6	92.3	30
702	38	162	4.3	79.4	92.4	60
852	15	38	2.5	95.0	95.0	10
1239	33	130	3.9	86.1	89.9	60
1299	17	60	3.5	84.6	94.9	10
2686	17	54	3.2	98.1	98.1	10
3300	21	68	3.2	95.2	99.4	10
3561	59	592	10.0	88.0	99.6	6369
3701	3	6	2.0	100.0	100.0	0
4323	40	300	7.5	73.8	87.6	460
5511	26	80	3.1	95.0	99.3	20
5669	10	22	2.2	1.00	1.00	0
6453	22	70	3.2	71.9	93.9	20
7018	36	136	3.8	95.5	98.1	40
7170	18	120	6.7	66.8	88.2	70
8220	23	124	5.4	93.5	96.7	50
average				88.3	95.2	

Table 3: Predictive power $\bar{Q}$ and $\bar{Q}'$ for Rocketfuel weights, along with CPU times, and the number of edges and nodes in each network (after removing degree one nodes).

5.1.2 *Synthetic weights*

The above results are favourably biased by the nature of the experiment, so we also investigate our synthetic weights, which are not correlated with distance. We simulate 10 weight scenarios for each of the three networks investigated in detail. Tables 1 and 2 present a summary of the results.

We first consider the quality of predictions under single link failures. Figure 3 shows Cumulative Distribution Functions (CDFs) for $1 - Q_e^{(i)}$ for each of the three networks. Note that the plots for ${Q'}_e^{(i)}$ look almost identical though slightly shifted. We can see that the average Q values 90.8%, 59.3%, 74.6% (for AS 1, 1239 and GEANT respectively), are reflected in the CDFs.

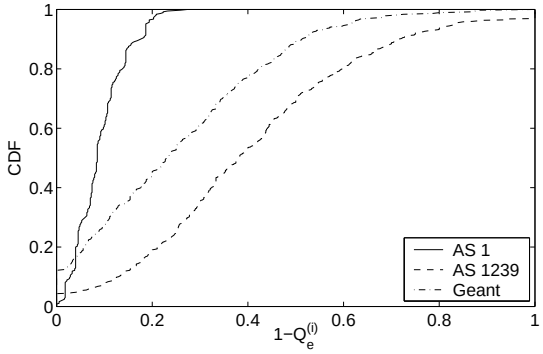
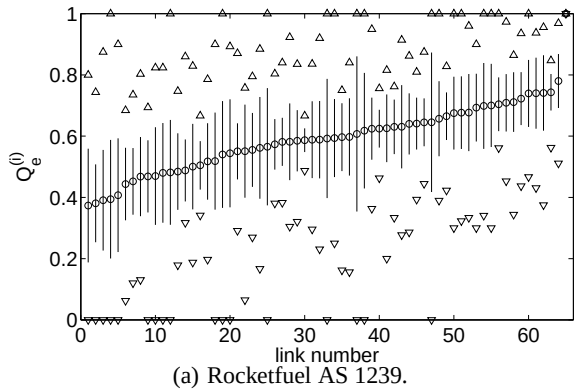
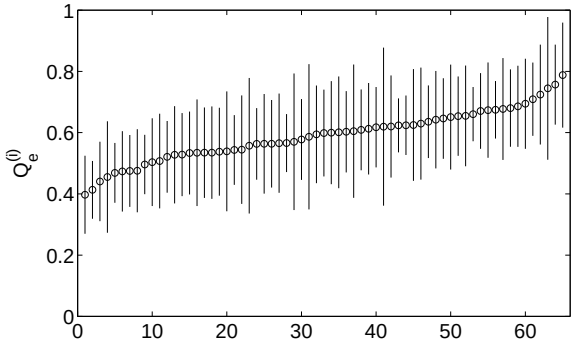


Figure 3: A CDF of the values of $1 - Q_e^{(i)}$ for the single link failure prediction cases for synthetic weights.

The most notable thing about these CDFs is that they spread the errors out across quite a range of values (e.g., the full range from 0 to 1 for AS 1239). Given this range of values, it is natural to ask whether there is some pattern in the values. Does a particular link in the network topology have more or less impact results when it fails? Figure 4 (a) shows a plot of the values of $\bar{Q}_e$ for each individual link failure (averaged over the different simulations, but also showing 95th percentile confidence intervals for $\bar{Q}_e$) for Rocketfuel AS 1239. The links are sorted in order of increasing $\bar{Q}_e$ to make the plot clearer. This leads to an apparent trend in the data, but we argue here that the variations shown in this trend are at best only marginally statistically significant. Figure 4 (b) shows a similar plot based on a Independent, Identically-Distributed (IID) Gaussian data (with similar parameters to the data shown in Figure 4 (a)). We can see that when sorted in this fashion, this data also appears to present a similar pattern despite the IID nature of the data. In comparison, similar plots (not shown here) for GEANT show a little more link based variability, and those for Rocketfuel AS 1 show less. Furthermore, we can see that the range of variations resulting from different weight scenarios (shown by the maximum and minimums on the graphs) is generally wider than the variations due to the link's position in the network topology. Hence, we argue that the particular weights of a network are more important than the link's position in the network topology.



(a) Rocketfuel AS 1239.



(b) Random data.

Figure 4: $\bar{Q}_e$ for each link e for synthetic weights including 95th percentile CIs (lines), and the maximum ($\triangle$) and minimum ($\triangledown$).

We also consider the quality of predictions of unobserved routes, i.e., $\bar{S}$. Figure 5 (a) shows the value of $\bar{S}(r)$ as we successively removed random links from the set of measurements. The results summarize 10 random order choices for link removal for each of the 10 synthetically chosen weights (100 simulations in all). Also shown (as short vertical bars) are 95th percentile confidence intervals for the estimates of S. The plots start (for $r = 0$) at $\bar{S}(0) = 1$ because in this case there is no missing data and so the weights obtained will be consistent with all routing. The value of $\bar{S}$ then decreases sharply. A very significant proportion of the loss of fidelity occurs in the first five steps (hence our choice of five in Table 1.) The most interesting feature of these graphs is the fact that the *per-*

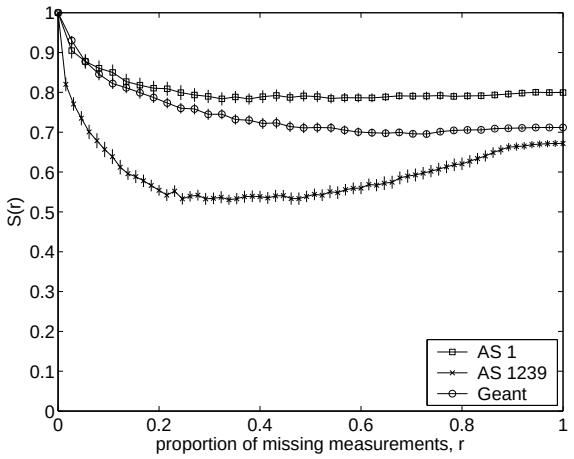


Figure 5: $\bar{S}(r)$ and 95th percentile confidence intervals.

formance can improve when we remove data and worsen when we add data. This seems counter-intuitive and requires future investigation. Also noteworthy is the fact that S does not decrease to zero. When there is zero information (in the form of route measurements), the Rocketfuel prior will result in the inferred weights being exactly proportional to the distances. The fact that $S(1) \neq 0$ indicates that even the (known to be incorrect) prior can make a reasonable proportion of valid predictions.

This is both a negative and positive result. It is negative because it means that a partial traceroute survey of a network might not improve route weight estimates dramatically (at least in terms of inferring unobserved paths). It could even result in worse performance. However, the result is positive in the sense that a complete, or almost complete traceroute survey can be quite useful even if the real link weights aren't directly correlated with distance.

5.1.3 Unit weights

We next consider the predictive power when the network uses unit weights, or unit weights with jitter. These mirror the results for synthetic weights (qualitatively, though there are some quantitative differences), and so we do not reproduce them here. The major thing that we learn from these scenarios concerns ECMP. The CDFs for the case of pure unit weights differ profoundly between $1 - Q_e^{(i)}$ and $1 - {Q'}_e^{(i)}$. However, for the case where the ECMPs have already been removed by adding some jitter, there is very little change between the CDFs for $1 - Q_e^{(i)}$ and $1 - {Q'}_e^{(i)}$.

5.1.4 Backbone weights

We repeat the results shown for the previous scenarios for the backbone weights. The prediction of routes after failures is quite poor. Apart from poor average results consideration of the CDF (not shown) shows there are now many cases where $Q_e^{(i)} = 0$. Also, as we remove data, the average performance now degrades monotonically to a much smaller value of $S(1)$ than before. Although the average values of $S(r)$ is now monotonic individual sample paths still show non-monotonic behaviour, but the net result is a much poorer set of estimated routes.

6. CONCLUSION

We have introduced, in this paper, a new concept: predictive power, and we have used this to assess the accuracy of shortest-path weight inference. In our tests, the accuracy of the prior model used in the inference process is not as important as it is for traffic matrix estimation. Significant departures from the prior resulted in only a few percent change in the predictive power. The exception is where we use the backbone weight scenario, which results in worse performance particularly when trying to predict the routes used after a link failure. Also, interesting is the fact that in some cases additional information is not helpful. We observed cases where predictive performance decreases as more measurements were added.

The results are quite positive, but they also leave scope for improving such techniques, for instance in choosing between multiple equivalent solutions to the LP. There are several directions for future work: e.g., investigating the cause of information reversal.

Acknowledgement

This work was supported by the Defence Science and Technology Organisation of Australia under contract number 4500474549. Olaf Maennel was supported by ARC grants DP0557066, and DP0665427 at the University of Adelaide.

7. REFERENCES

[1] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," in *ACM SIGCOMM*, (Pittsburg, USA), August 2002.

[2] N. Spring, R. Mahajan, and T. Anderson, "Quantifying the causes of path inflation," in *ACM SIGCOMM 2003*, (Karlsruhe, Germany), 2003.

[3] B. Fortz and M. Thorup, "Robust optimization of OSPF/IS-IS weights," in *Proc. INOC 2003* (W. Ben-Ameur and A. Petrowski, eds.), pp. 225–230, October 2003.

[4] L. S. Buriol, M. G. C. Resende, C. C. Ribeiro, and M. Thorup, "A memetic algorithm for OSPF routing," in *Proc. 6th INFORMS Telecom*, pp. 187–188, 2002.

[5] M. Roughan, M. Thorup, and Y. Zhang, "Traffic engineering with estimated traffic matrices," in *ACM SIGCOMM Internet Measurement Conference (IMC)*, (Miami Beach, FL, USA), pp. 248–258, 2003.

[6] R. Teixeira, K. Marzullo, S. Savage, and G. Voelker, "In search of path diversity in ISP networks," in *ACM Internet Measurement Conference*, Oct. 2003.

[7] B. Augustin, T. Friedman, M. Curie, and R. Teixeira, "Measuring load-balanced paths in the Internet," in *ACM SIGCOMM Internet Measurement Conference*, (San Diego, CA, USA), October 2007.

[8] L. Subramanian, S. Agarwal, J. Rexford, and R. Katz, "Characterizing the Internet hierarchy from multiple vantage points," in *Infocom*, 2002.

[9] F. Wang and L. Gao, "On inferring and characterizing Internet routing policies," in *ACM SIGCOMM/USENIX Internet Measurement Conference*, (Miami, Florida, USA), October 2003.

[10] J. Xia and L. Gao, "On the evaluation of AS relationship inferences," in *Globecom*, 2004.

[11] W. Mühlbauer, A. Feldmann, M. R. O. Maennel, and S. Uhlig, "Building an AS-topology model that captures route diversity," in *ACM SIGCOMM*, (Pisa, Italy), 2006.

[12] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic matrix estimation: Existing techniques and new directions," in *ACM SIGCOMM*, (Pittsburg, USA), August 2002.

[13] Y. Zhang, M. Roughan, C. Lund, and D. Donoho, "An information-theoretic approach to traffic matrix estimation," in *ACM SIGCOMM*, (Karlsruhe, Germany), pp. 301–312, August 2003.

[14] R. Cáceres, N. Duffield, J. Horowitz, and D. Towsley, "Multicast-based inference of network-internal loss characteristics," *IEEE Trans. in Information Theory*, vol. 45, pp. 2462–2480, 1999.

[15] N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley, "Multicast topology inference from measured end-to-end loss," *IEEE Transactions in Information Theory*, vol. 48, no. 1, pp. 26–45, 2002.

[16] A. Nucci, A. Sridharan, and N. Taft, "The problem of synthetically generating IP traffic matrices: Initial recommendations," *ACM Computer Communication Review*, vol. 35, no. 3, 2005.

[17] A. Nucci, S. Bhattacharyya, N. Taft, and C. Diot, "IGP link weight assignment for operational Tier-1 backbones," *Networking, IEEE/ACM Transactions on*, vol. 15, no. 4, pp. 789–802, Aug. 2007.

[18] M. Roughan, "Simplifying the synthesis of Internet traffic matrices," *SIGCOMM Comput. Commun. Rev.*, vol. 35, pp. 93–96, October 2005.