Inferring AS Relationships: Dead End or Lively Beginning?

Xenofontas Dimitropoulos[1,2], Dmitri Krioukov[2], Bradley Huffaker[2], kc claffy[2], and George Riley[1]

[1] School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
fontas@ece.gatech.edu,riley@ece.gatech.edu
[2] Cooperative Association for Internet Data Analysis (CAIDA)
La Jolla, California 92093–0505
dima@caida.org,brad@caida.org,kc@caida.org

Abstract. Recent techniques for inferring business relationships between ASs [1, 2] have yielded maps that have extremely few *invalid* BGP paths in the terminology of Gao [3]. However, some relationships inferred by these newer algorithms are incorrect, leading to the deduction of unrealistic AS hierarchies. We investigate this problem and discover what causes it. Having obtained such insight, we generalize the problem of AS relationship inference as a multiobjective optimization problem with node-degree-based corrections to the original objective function of minimizing the number of invalid paths. We solve the generalized version of the problem using the semidefinite programming relaxation of the MAX2SAT problem. Keeping the number of invalid paths small, we obtain a more veracious solution than that yielded by recent heuristics.

1 Introduction

As packets flow in the Internet, money also flows, not necessarily in the same direction. Business relationships between ASs reflect both flows, indicating a direction of money transfer as well as a set of constraints to the flow of traffic. Knowing AS business relationships is therefore of critical importance to providers, vendors, researchers, and policy makers, since such knowledge sheds more light on the relative "importance" of ASs.

The problem is also of multidimensional interest to the research community. Indeed, the Internet AS-level topology and its evolutionary dynamics result from business decisions among Internet players. Knowledge of AS relationships in the Internet provides a valuable validation framework for economy-based Internet topology evolution modeling, which in turn promotes deeper understanding of the fundamental laws driving the evolution of the Internet topology and its hierarchy.

Unfortunately, the work on inferring AS relationships from BGP data has recently encountered difficulties. We briefly describe this situation in its historical context.

Gao introduces the AS relationship inference problem in her pioneering paper [3]. This work approximates reality by assuming that any AS-link is of one of the following three types: customer-provider, peering, or sibling. If all ASs strictly adhere to import and export policies described in [3], then every BGP path must comply with the following hierarchical pattern: an uphill segment of zero or more customer-to-provider or sibling-to-sibling links, followed by zero or one peer-to-peer links, followed by a downhill segment of zero or more provider-to-customer or sibling-to-sibling links. Paths with the described hierarchical structure are deemed *valid*. After introducing insight about valid paths, Gao proposes an inference heuristic that identifies top providers and peering links based on AS degrees and valid paths.

In [4], Subramanian *et al.* (SARK) slightly relax the problem by not inferring sibling links, and introduce a more consistent and elegant mathematical formulation. The authors render the problem into a combinatorial optimization problem: given an undirected graph G derived from a set of BGP paths P, assign the edge type (customer-provider or peering) to every edge in G such that the total number of valid paths in P is maximized. The authors call the problem the type-of-relationship (ToR) problem, conjecture that it is NP-complete, and provide a simple heuristic approximation.

Di Battista *et al.* (DPP) in [1] and independently Erlebach *et al.* (EHS) in [2] prove that the ToR problem is indeed NP-complete. EHS prove also that it is even harder, specifically APX-complete.[3] More importantly for practical purposes, both DPP and EHS make the straightforward observation that peering edges *cannot* be inferred in the ToR problem formulation. Indeed, as the validation data presented by Xia *et al.* in [5] indicates, only 24.63% of the validated SARK peering links are correct.

Even more problematic is the following dilemma. DPP (and EHS) come up with heuristics that outperform the SARK algorithm in terms of producing smaller numbers of invalid paths [1, 2]. Although these results seem to be a positive sign, closer examination of the AS relationships produced by the DPP algorithm [6] reveals that the DPP inferences are further from reality than the SARK inferences. In the next section we show that improved solutions to the ToR problem do not yield practically correct answers and contain obviously misidentified edges, e.g. well-known global providers appear as customers of small ASs. As a consequence, we claim that improved solutions to the unmodified ToR problem do not produce realistic results.

An alternative approach to AS relationship inference is to disregard BGP paths and switch attention to other data sources (e.g. WHOIS) [7, 8], but nothing suggests that we have exhausted all possibilities of extracting relevant information from BGP data. Indeed, in this study we seek to answer the following question: can we adjust the original (ToR) problem formulation, so that an algo-

[3] There exists no polynomial-time algorithm approximating an APX-complete problem above a certain *inapproximability* limit (ratio) dependent on the particular problem.

rithmic solution to the modified problem would yield a better answer from the practical perspective?

The main contribution of this paper is that we positively answer this question. We describe our approach and preliminary results in the subsequent two sections, and conclude by describing future directions of this work.

2 Methodology

2.1 Inspiration behind our approach

The main idea behind our approach is to formalize our knowledge regarding why improved solutions to the ToR problem fail to yield practically right answers. To this end we reformulate the ToR problem as a multiobjective optimization problem introducing certain corrections to the original objective function. We seek a modification of the original objective function, such that the minimum of the new objective function reflects an AS relationship mapping that is closer to reality.

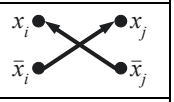
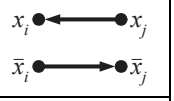
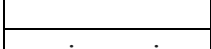
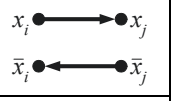
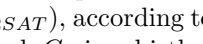
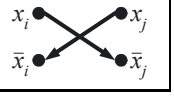
2.2 Mapping to 2SAT

To achieve this purpose, we start with the DPP and EHS results [1, 2] that deliver the fewest invalid paths. Suppose we have a set of BGP paths P from which we can extract the undirected AS-level graph $G(V, E)$. We introduce *direction* to every edge in E *from* the customer AS *to* the provider AS. Directing edges in E induces direction of edges in P. A path in P is valid if it does not contain the following invalid pattern: a provider-to-customer edge followed by a customer-to-provider edge. The ToR problem is to assign direction to edges in E minimizing the number of paths in P containing the invalid pattern.

The problem of identifying the directions of all edges in E making *all* paths in P valid—assuming such edge orientation exists—can be reduced to the 2SAT problem.[4] Initially, we arbitrarily direct all edges in E and introduce a boolean variable x_i for every edge i, $i = 1 \ldots \left|E\right|$. If the algorithms described below assign the value *true* to x_i, then edge i keeps its original direction, while assignment of *false* to x_i reverses the direction of i. We then split each path in P into pairs of adjacent edges involving triplets of ASs (all 1-link paths are always valid) and perform mapping between the obtained pairs and 2-variable clauses as shown in Table 1. The mapping is such that only clauses corresponding to the invalid path pattern yield the *false* value when both variables are *true*. If there exists an assignment of values to all the variables such that all clauses are satisfied, then this assignment makes all paths valid.

[4] 2SAT is a variation of the satisfiability problem: given a set of clauses with two boolean variables per clause $l_i \vee l_j$, find an assignment of values to variables satisfying all the clauses. MAX2SAT is a related problem: find the assignment maximizing the number of simultaneously satisfied clauses.

Table 1. Mapping between pairs of adjacent edges in P, 2SAT clauses, and edges in G_{2SAT}. The invalid path pattern is in the last row.

Edges in P	2SAT clause	Edges in G_{2SAT}
	$x_i \vee x_j$	
	$x_i \vee \bar{x}_j$	
	$\bar{x}_i \vee x_j$	
	$\bar{x}_i \vee \bar{x}_j$	

To solve the 2SAT problem, we construct a dual graph, the 2SAT graph $G_{2SAT}(V_{2SAT}, E_{2SAT})$, according to the rules shown in Table 1: every edge $i \in E$ in the original graph G gives birth to two vertices x_i and $\bar{x}_i$ in V_{2SAT}, and every pair of adjacent links $l_i \vee l_j$ in P, where literal l_i (l_j) is either x_i (x_j) or $\bar{x}_i$ ($\bar{x}_j$), gives birth to two directed edges in E_{2SAT}: from vertex $\bar{l}_i$ to vertex l_j and from vertex $\bar{l}_j$ to vertex l_i. As shown in [9], there exists an assignment satisfying all the clauses if there is no edge i such that both of its corresponding vertices in the 2SAT graph, $x_i, \bar{x}_i \in V_{2SAT}$, belong to the same strongly connected component[5] (SCC) in G_{2SAT}.

If an assignment satisfying all the clauses exists we can easily find it. We perform topological sorting[6] t on nodes in V_{2SAT} and assign *true* or *false* to a variable x_i depending on if $t(\bar{x}_i) < t(x_i)$ or $t(x_i) < t(\bar{x}_i)$ respectively. All operations described so far can be done in linear time.

2.3 MAX2SAT: DPP vs. EHS

As soon as a set of BGP paths P is "rich enough," there is no assignment satisfying all clauses and making all paths valid. Furthermore, the ToR problem of maximizing the number of valid paths can be reduced to the MAX2SAT [1, 2] problem of maximizing the number of satisfied clauses. Making this observation, DPP propose a heuristic to find the maximal subset of paths $P_S \subset P$ such that all paths in P_S are valid.

EHS use a different approach. They first direct the edges $i \in E$ that can be directed without causing conflicts. Such edges correspond to vertices $x_i, \bar{x}_i \in V_{2SAT}$

[5] An SCC is a set of nodes in a directed graph s. t. there exists a directed path between every ordered pair of nodes.

[6] Given a directed graph $G(V, E)$, function $t : V \mapsto \mathbb{R}$ is topological sorting if $t(i) \leq t(j)$ for every ordered pair of nodes $i, j \in V$ s. t. there exists a directed path from i to j.

that have indegree or outdegree zero. Then EHS iteratively remove edges directed as described above and strip P, G, and G_{2SAT} accordingly. This procedure significantly shortens the average path length in P, which improves the approximation of ToR by MAX2SAT. Finally, they approximate MAX2SAT to find a solution to the ToR problem.

2.4 Solving MAX2SAT with SDP

The MAX2SAT problem is NP- and APX-complete [10], but Goemans and Williamson (GW) [11] construct a famous approximation algorithm that uses semidefinite programming (SDP) and delivers an approximation ratio of 0.878. The best approximation ratio currently known is 0.940, due to improvements to GW by Lewin, Livnat, and Zwick (LLZ) in [12]. Note that this approximation ratio is pretty close to the MAX2SAT inapproximability limit of $\frac{21}{22} \sim 0.954$ [13].

To cast a MAX2SAT problem with m_2 clauses involving m_1 literals (variables x_i and their negations $\bar{x}_i$, $i = 1 \dots m_1$) to a semidefinite program, we first get rid of negated variables by introducing m_1 variables $x_{m_1+i} = \bar{x}_i$. Then we establish mapping between boolean variables x_k, $k = 1 \dots 2m_1$, and $2m_1 + 1$ auxiliary variables $y_0, y_k \in \{-1, 1\}$, $y_{m_1+i} = -y_i$, using formula $x_k = (1 + y_0 y_k)/2$. This mapping guarantees that $x_k = true \Leftrightarrow y_k = y_0$ and $x_k = false \Leftrightarrow y_k = -y_0$. Given the described construction, we call y_0 the *truth* variable. After trivial algebra, the MAX2SAT problem becomes the maximization problem for the sum $1/4 \sum_{k,l=1}^{2m_1} w_{kl}(3 + y_0 y_k + y_0 y_l - y_k y_l)$, where weights w_{kl} are either 1 if clause $x_k \vee x_l$ is present in the original MAX2SAT instance or 0 otherwise. Hereafter we fix the notations for indices $i, j = 1 \dots m_1$ and $k, l = 1 \dots 2m_1$.

The final transformation to make the problem solvable by SDP is relaxation. Relaxation involves mapping variables y_0, y_k to $2m_1+1$ unit vectors $v_0, v_k \in \mathbb{R}^{m_1+1}$ fixed at the same origin—all vector ends lie on the unit sphere S_{m_1}. The problem is to maximize the sum composed of vector scalar products:

$$\begin{aligned} \max \ & \frac{1}{4} \sum_{k,l=1}^{2m_1} w_{kl}(3 + v_0 \cdot v_k + v_0 \cdot v_l - v_k \cdot v_l) \\ \text{s.t.} \ & v_0 \cdot v_0 = v_k \cdot v_k = 1, \ v_i \cdot v_{m_1+i} = -1, \\ & k = 1 \dots 2m_1, \ i = 1 \dots m_1. \end{aligned} \tag{1}$$

Interestingly, this problem, solvable by SDP, is equivalent to the following minimum energy problem in physics. Vectors v_0, v_k point to the locations of particles p_0, p_k freely moving on the sphere S_{m_1} except that particles p_i and p_{m_1+i} are constrained to lie opposite on the sphere. For every MAX2SAT clause $x_k \vee x_l$, we introduce three constant forces of equal strength (see Fig. 1): one repulsive force between particles p_k and p_l, and two attractive forces: between p_k and p_0, and between p_l and p_0—the *truth* particle p_0 attracts all other particles p_k with the forces proportional to the number of clauses containing x_k. The goal is to find the location of particles on the sphere minimizing the potential energy of the system. If we built such a mission-specific computer in the lab, it would solve this problem in constant time. SDP solves it in polynomial time.

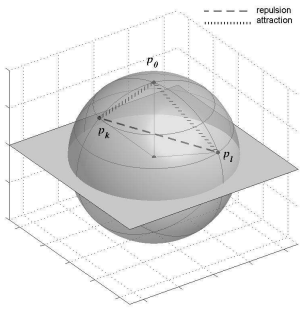


Fig. 1. The semidefinite programming relaxation to the MAX2SAT problem. Point p_0 (corresponding to vector v_0 from the text) is the *truth* point. It attracts both points p_k and p_l representing the boolean variables from the clause $x_k \vee x_l$. Points p_k and p_l repel each other. The problem is to identify the locations of all points on the sphere that minimize the potential energy of the system. Given an orientation by SDP, we cut the system by a random hyperplane and assign value *true* to the variables corresponding to points lying on the same side of the hyperplane as the truth p_0.

To extract the solution for the MAX2SAT problem from the solution obtained by SDP for the relaxed problem, we perform rounding. Rounding involves cutting the sphere by a randomly oriented hyperplane containing the sphere center. We assign value *true* (*false*) to variables x_k corresponding to vectors v_k lying on the same (opposite) side of the hyperplane as the *truth* vector v_0. GW prove that the solution to the MAX2SAT problem obtained this way delivers the approximation ratio of 0.878 [11]. We can also rotate the vector output obtained by SDP before rounding and skew the distribution of the hyperplane orientation to slightly prefer the orientation perpendicular to v_0. These two techniques explored to their greatest depths by LLZ improve the approximation ratio up to 0.9401 [12].

2.5 Analysis of the unperturbed solution

We now have the solution to the original ToR problem and are ready to analyze it. While the number of invalid paths is small [2], the solution is not perfect—some inferred AS relationships are not in fact accurate. What causes these misclassifications?

First, some edges may be directed either way resulting in exactly the same number of invalid paths—such edges are directed randomly. To exemplify, consider path $p \in P$, $p = \{i_1 i_2 \dots i_{|p|-1} j\}$, $i_1, i_2, \dots, i_{|p|-1}, j \in E$, and suppose that the last edge j appears only in one path (that is, p) and that it is from some large provider (like UUNET) to a small customer. Suppose that other edges $i_1, i_2, \dots, i_{|p|-1}$ appear in several other paths and that they are correctly inferred as customer-to-provider. In this scenario both orientations of edge j (i.e. correct and incorrect: provider-to-customer and customer-to-provider) render path p valid. Thus, edge j is directed randomly, increasing the likelihood of an incorrect inference. We can find many incorrect inferences of this type in our experiments in the next section and in [6], e.g. well-known large providers like UUNET, AT&T, Sprintlink, Level3, are inferred as customers of smaller ASs like AS1 (AS degree 67), AS2685 (2), AS8043 (1), AS13649 (7), respectively.

Second, not all edges are customer-to-provider or provider-to-customer. In particular, trying to direct sibling edges leads to proliferation of error. Indeed, when the only objective is to maximize the number of valid paths, directing a sibling edge brings the risk of misdirecting the dependent edges sharing a clause with the sibling edge. To clarify, consider path $p \in P$, $p = \{ij\}$, $i, j \in E$, and suppose that in reality i is a sibling edge that appears in multiple paths and that j is a customer-to-provider edge that appears only in one path p. The algorithm can classify edge i either as customer-to-provider or provider-to-customer depending on the structure of the paths in which it appears. If this structure results in directing i as provider-to-customer, then the algorithm erroneously directs edge j also as provider-to-customer to make path p valid. In other words, the outcome is that we maximize the number of valid paths at the cost of inferring edge j incorrectly.

We can conclude that the maximum number of valid paths does not correspond to a correct answer because, as illustrated in the above two examples, it can result in miss-inferred links. Specifically, in the presence of multiple solutions there is nothing in the objective function to require the algorithm to prefer the proper orientation for edge j. Our next key question is: Can we adjust the objective function to infer the edge direction correctly?

2.6 Our new generalized objective function

A rigorous way to pursue the above question is to add to the objective function some small modifier selecting the correct edge direction for links unresolved by the unperturbed objective function. Ideally this modifier should be a function of "AS importance," such as the relative size of the customer tree of an AS. Unfortunately, defined this way the modifier is a function of the end result, edge orientation, which makes the problem intractable (i.e. we cannot solve it until we solve it).

The simplest correcting function that does not depend on the edge direction and is still related to perceived "AS importance," is the AS degree "gradient" in the original undirected graph G—the difference between node degrees of adjacent ASs. In the examples from the previous subsection, the algorithm that is trying not only to minimize the number of invalid paths but also to direct edges from adjacent nodes of lower degrees to nodes of higher degrees will effectively have an incentive to correctly infer the last edge $j \in p$.

More formally, we modify the objective function as follows. In the original problem formulation, weights w_{kl} for 2-link clauses $x_k \vee x_l$ (pairs of adjacent links in P) are either 0 or 1. We first alter them to be either 0, if pair $\{kl\} \notin P$, or $w_{kl}(\alpha) = c_2\alpha$ otherwise. The normalization coefficient c_2 is determined from the condition $\sum_{k \neq l} w_{kl}(\alpha) = \alpha \Rightarrow c_2 = 1/m_2$ (recall that m_2 is the number of 2-link clauses), and α is an external parameter, $0 \leq \alpha \leq 1$, whose meaning we explain below.

In addition, for every edge $i \in E$, we introduce a 1-link clause weighted by a function of the node degree gradient. More specifically, we initially orient every edge $i \in E$ along the node degree gradient: if d_i^- and d_i^+, $d_i^- \leq d_i^+$, are degrees of

nodes adjacent to edge i, we direct i from the d_i^--degree node to the d_i^+-degree node, for use as input to our algorithm.[7]

Then, we add 1-link clauses $x_i \vee x_i$, $\forall i \in E$, to our MAX2SAT instance, and we weight them by $w_{ii}(\alpha) = c_1(1-\alpha)f(d_i^-, d_i^+)$. The normalization coefficient c_1 is determined from the condition $\sum_i w_{ii}(\alpha) = 1-\alpha$, and the function f should satisfy the following two conditions: 1) it should "roughly depend" on the *relative* node degree gradient $(d_i^+ - d_i^-)/d_i^+$; and 2) it should provide higher values for node pairs with the same relative degree gradient but higher absolute degree values. The first condition is transparent: we expect that an AS with node degree 5, for example, is more likely a customer of an AS with node degree 10 than a 995-degree AS is a customer of a 1000-degree AS. The second condition is due to the fact that we do not know the true AS degrees: we approximate them by degrees of nodes in our BGP-derived graph G. The graphs derived from BGP data have a tendency to underestimate the node degree of small ASs, while they yield more accurate degrees for larger ASs [14]. Because of the larger error associated with small ASs, an AS with node degree 5, for example, is less likely a customer of an AS with node degree 10 than a 500-degree AS is a customer of a 1000-degree AS.

We select the following function satisfying the two criteria described above:

$$f(d_i^-, d_i^+) = \frac{d_i^+ - d_i^-}{d_i^+ + d_i^-} \log(d_i^+ + d_i^-). \tag{2}$$

In summary, our new objective function looks exactly as the one in (1), but with different weights on clauses:

$$w_{kl}(\alpha) = \begin{cases} c_2\alpha & \text{if} \quad \{kl\} \in P, \\ c_1(1-\alpha)f(d_k^-, d_k^+) & \text{if} \quad k = l \le m_1, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Now we can explain the role of the parameter α. Since $\sum_{k\neq l} w_{kl}(\alpha) = \alpha$ and $\sum_{k=l} w_{kl}(\alpha) = 1-\alpha$, parameter α measures the relative importance of sums of all 2- and 1-link clauses. If $\alpha = 1$, then the problem is equivalent to the original unperturbed ToR problem—only the number of invalid paths matters. If $\alpha = 0$, then, similar to Gao, only node degrees matter. Note that in the terminology of multiobjective optimization, we consider the simplest scalar method of weighted sums.

In our analogy with physics in Fig. 1, we have weakened the repulsive forces among particles other than the *truth* particle p_0, and we have strengthened the forces between p_0 and other particles. When $\alpha = 0$, there are no repulsive forces, the *truth* particle p_0 attracts all other particles to itself, and all the vectors become collinear with v_0. Cut by any hyperplane, they all lie on the same side as v_0, which means that all variables x_i are assigned value *true* and all links i remain directed along the node degree gradient in the output of our algorithm.

[7] An initial direction along the node degree gradient does not affect the solution since any initial direction is possible. We select the node degree gradient direction to simplify stripping of non-conflict edges in the next section.

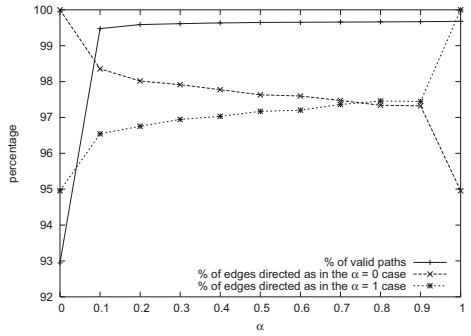


Fig. 2. Percentage of valid paths, of edges directed as in the $\alpha = 0$ case and of edges directed as in the $\alpha = 1$ case for different values of α.

3 Results

In our experiments, the BGP path set P is a union of BGP tables from RouteViews [15] and 18 BGP route servers from [16] collected on May 13, 2004. Paths of length 1 are removed since they are always valid. The total number of paths is 1,025,775 containing 17,557 ASs, 37,021 links, and 382,917 unique pairs of adjacent links.

We first pre-process the data by discovering sibling links. For this purpose, we use a union of WHOIS databases from ARIN, RIPE, APNIC, and LACNIC collected on June 10, 2004. We say that two ASs belong to the same organization if, in the WHOIS database, they have exactly the same organization names, or names different only in the last digits, e.g. "ATT-37" and "ATT-38,"or very similar names, e.g. "UUNET South Africa" and "UUNET Germany." We infer links in P between adjacent ASs belonging to the same organization as sibling. We find 211 sibling links in our dataset, which we ignore in subsequent steps. More precisely, we do not assign boolean variables to them.

We then direct the remaining links in the original graph G along the node degree gradient, assign boolean variables to them, and construct the dual G_{2SAT} graph. After directing edge i along the node degree gradient, we check whether this direction satisfies all clauses containing l_i (x_i or $\bar{x}_i$). If so, we then remove the edge and strip P, G, and G_{2SAT} accordingly. In this case we say that edge i causes no conflicts because the value of the corresponding literal l_i satisfies all the clauses in which l_i appears, independent of the values of all other literals sharing the clauses with l_i. A non-conflict edge has two corresponding vertices in the G_{2SAT} graph, x_i and $\bar{x}_i$. It follows from the construction of the G_{2SAT} graph that x_i has an outdegree of zero and $\bar{x}_i$ has an indegree of zero. We repeat the described procedure until we cannot remove any more edges. The stripped graph G has 1,590 vertices (9% of the original $\left|V\right|$) and 4,249 edges (11% of the original $\left|E\right|$). The stripped G_{2SAT} graph has 8,498 vertices and 46,920 edges. In summary, we have 4,249 (m_1) 1-link clauses and 23,460 (m_2) 2-link clauses. We feed this data into a publicly available SDP solver DSDP v4.7 [17], reusing parts of the code from [2] and utilizing the LEDA v4.5 software library [18]. We

Table 2. Hierarchical ranking of ASs. The position *depth* (the number of AS at the levels above) and *width* (the number of ASs at the same level) of the top five ASs in the $\alpha = 0$ and $\alpha = 1$ cases are shown for different values of α. The customer leaf ASs are marked with asterisks.

			$\alpha = 0.0$		$\alpha = 0.2$		$\alpha = 0.5$		$\alpha = 0.8$		$\alpha = 1.0$	
AS #	name	degree	dep.	wid.	dep.	wid.	dep.	wid.	dep.	wid.	dep.	wid.
701	UUNET	2373	0	1	0	173	1	232	1	252	17	476
1239	Sprint	1787	1	1	0	173	1	232	1	252	17	476
7018	AT&T	1723	2	1	0	173	1	232	1	252	17	476
3356	Level 3	1085	3	1	0	173	1	232	1	252	17	476
209	Qwest	1072	4	1	0	173	1	232	1	252	17	476
3643	Sprint Austr.	17	194	1	222	1	250	1	268	1	0	4
6721	Nextra Czech Net	3	1742	941	833	88	868	90	884	89	0	4
11551	Pressroom Ser.	2	1742	941	1419	398	1445	390	1457	386	0	4
1243	Army Systems	2	2683	14725*	2753	14655*	1445	390	1457	386	0	4
6712	France Transpac	2	2683	14725*	2753	14655*	292	3	1	252	4	13

incorporate the pre-rounding rotation and skewed distribution of hyperplane orientation by LLZ [12].

Fig. 2 shows results of edge orientations we derive for different values of α in (3). Specifically, the figure shows the percentage of valid paths, edges directed as in the $\alpha = 0$ case, and edges directed as in the $\alpha = 1$ case. In the particular extreme case of $\alpha = 1$, the problem reduces to the original ToR problem considered by DPP and EHS, and its solution yields the highest percentage of valid paths, 99.67%. By decreasing α, we increase preference to directing edges along the node degree gradient, and at the other extreme of $\alpha = 0$, all edges become directed along the node gradient, but the number of valid paths is 92.95%.

Note that changing α from 0 to 0.1 redirects 1.64% of edges, which leads to a significant 6.53% increase in the number of valid paths. We also observe that the tweak of α from 1 to 0.9 redirects 2.56% of edges without causing any significant decrease (only 0.008%) in the number of valid paths. We find that most of these edges are directed randomly in the $\alpha = 1$ case because oriented either way they yield the same number of valid paths. In other words, the AS relationships represented by these edges cannot be inferred by minimizing the number of invalid paths.

We also rank ASs by means of our inference results with different α values. To this end we split all ASs into hierarchical levels as follows. We first order all ASs by their *reachability*—that is, the number of ASs that a given AS can reach "for free" traversing only provider-to-customer edges. We then group ASs with the same reachability into levels. ASs at the highest level can reach all other ASs "for free." ASs at the lowest level have the smallest reachability (fewest "free" destinations). Then we define the position *depth* of AS X as the number of ASs at the levels above the level of AS X. The position *width* of AS X is the number of ASs at the same level as AS X.

Table 2 shows the results of our AS ranking. For different values of α, we track the positions of the top five ASs in the $\alpha = 0$ and $\alpha = 1$ cases. In the former case, well-known large ISPs are at the top, but the number of invalid paths is relatively large, cf. Fig. 2. In the latter case delivering the solution to the unperturbed

ToR problem, ASs with small degrees occupy the top positions in the hierarchy. These ASs appear in much lower positions when $\alpha \neq 1$. Counter to reality, the large ISPs are not even near the top of the hierarchy. We observe that the depth[8] of these large ASs increases as α approaches 1, indicating an increasingly stronger deviation from reality. The deviation is maximized when $\alpha = 1$. This observation pronounces the limitation of the ToR problem formulation based solely on maximization of the number of valid paths.

4 Conclusion and future work

Using a standard multiobjective optimization method, we have constructed a natural generalization of the known AS relationship inference heuristics. We have extended the combinatorial optimization approach based on minimization of invalid paths, by incorporating AS-degree-based information into the problem formulation. Utilizing this technique, we have obtained first results that are more realistic than the inferences produced by the recent state-of-the-art heuristics [1, 2]. We conclude that our approach opens a promising path toward increasingly veracious inferences of business relationships between ASs.

The list of open issues that we plan to address in our future work includes: 1) modifications to the algorithm to infer *peering*; 2) careful analysis of the *trade-off surface* [19] of the problem, required for selecting the value of the external parameters (e.g. α) corresponding to the right answer; 3) detailed examination of the *structure* of the AS graph directed by inferred AS relationships; 4) *validation* considered as a set of constraints narrowing the range of feasible values of external parameters; and 5) investigation of other *AS-ranking mechanisms* responsible for the structure of the inferred AS hierarchy.

5 Acknowledgements

We thank Thomas Erlebach, Alexander Hall and Thomas Schank for sharing their code with us.

Support for this work was provided by the Cisco University Research Program, by DARPA N66002-00-1-893, and by NSF ANI-0221172, ANI-9977544, ANI-0136969 and CNS-0434996, with support from DHS/NCS.

References

1. Battista, G.D., Patrignani, M., Pizzonia, M.: Computing the types of the relationships between Autonomous Systems. In: IEEE INFOCOM. (2003)

[8] Note that the large ISPs are at the same depth as soon as $\alpha \neq 0$, which is expected since they form "almost a clique" [4] and are likely to belong to the same SCC. All nodes in the same SCC have the same reachability. The converse is not necessarily true.

2. Erlebach, T., Hall, A., Schank, T.: Classifying customer-provider relationships in the Internet. In: Proceedings of the IASTED International Conference on Communications and Computer Networks (CCN). (2002)
3. Gao, L.: On inferring Autonomous System relationships in the Internet. In: IEEE/ACM Transactions on Networking. (2001)
4. Subramanian, L., Agarwal, S., Rexford, J., Katz, R.H.: Characterizing the Internet hierarchy from multiple vantage points. In: IEEE INFOCOM. (2002)
5. Xia, J., Gao, L.: On the evaluation of AS relationship inferences. In: IEEE GLOBECOM. (2004)
6. Rimondini, M.: Statistics and comparisons about two solutions for computing the types of relationships between Autonomous Systems (2002) `http://www.dia.uniroma3.it/~compunet/files/ToR-solutions-comparison.pdf%`.
7. Siganos, G., Faloutsos, M.: Analyzing BGP policies: Methodology and tool. In: IEEE INFOCOM. (2004)
8. Huber, B., Leinen, S., O'Dell, R., Wattenhofer, R.: Inferring AS relationships beyond counting edges. Technical Report TR 446, ETH Zürich (2004)
9. Aspvall, B., Plass, M.F., Tarjan, R.E.: A linear time algorithm for testing the truth of certain quantified boolean formulae. Information Processing Letters **8** (1979) 121–123
10. Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., Protasi, M.: Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties. Springer Verlag, Berlin (1999)
11. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of the ACM **42** (1995) 1115–1145
12. Lewin, M., Livnat, D., Zwick, U.: Improved rounding techniques for the MAX 2-SAT and MAX DI-CUT problems. In: Proceedings of the 9^{th} International IPCO Conference on Integer Programming and Combinatorial Optimization. (2002)
13. Håstad, J.: Some optimal inapproximability results. In: Proceedings of the 29^{th} Annual ACM Symposium on Theory of Computing. (1997)
14. Chang, H., Govindan, R., Jamin, S., Shenker, S.J., Willinger, W.: Towards capturing representative AS-level Internet topologies. Computer Networks Journal **44** (2004) 737–755
15. Meyer, D.: University of Oregon Route Views Project (2004)
16. : A traceroute server list. http://www.traceroute.org (2004)
17. Benson, S., Ye, Y., Zhang, X.: A dual-scaling algorithm for semidefinite programming (2004) `http://www-unix.mcs.anl.gov/~benson/dsdp/`.
18. GmbH, A.S.S.: L E D A library (2004) `http://www.algorithmic-solutions.com/enleda.htm`.
19. Collette, Y., Siarry, P.: Multiobjective Optimization: Principles and Case Studies. Springer-Verlag, Berlin (2003)