

Classifying Customer-Provider Relationships in the Internet

Thomas Erlebach, Alexander Hall¹
Computer Engineering and Networks Lab
ETH Zürich, CH-8092 Zürich
{erlebach|hall}@tik.ee.ethz.ch

Thomas Schank
Department of Computer & Information Science
Universität Konstanz, D-78457 Konstanz
schank@fmi.uni-konstanz.de

Abstract

The problem of inferring customer-provider relationships in the autonomous system topology of the Internet leads to the following optimization problem: given an undirected graph G and a set P of paths in G , orient the edges of G such that as many paths as possible are valid, meaning that they do not contain an internal node with both incident edges on the path directed away from that node. The complexity of this problem was left open by Subramanian et al. (“Characterizing the Internet hierarchy from multiple vantage points,” INFOCOM 2002). We show that finding an orientation that makes all paths valid (if such an orientation exists) can be done in linear time and that the maximization version of the problem is \mathcal{NP} -hard and cannot be approximated within $1/n^{1-\epsilon}$ for n paths unless $\mathcal{NP}=\text{co-}\mathcal{RP}$. We present constant-factor approximation algorithms for the case where the paths have bounded length and prove that the problem remains \mathcal{APX} -hard in this case. Finally, we report experimental results demonstrating that the approximation algorithm yields very good solutions on real data sets.

KEY WORDS

autonomous system, internet topology, approximation

1 Introduction

The Internet is a huge, complex network whose present state is the outcome of a distributed growth process without centralized control. Because of the importance of the Internet as our basic communication infrastructure, significant research efforts have recently been devoted to the discovery and analysis of its topology. The Internet topology can be considered either on the level of individual routers and hosts or on the level of *autonomous systems* (subnetworks under separate administrative control). In this paper, we focus on the autonomous system (AS) topology of the Internet. The AS topology can be represented as an undirected graph: each vertex corresponds to an AS, and two vertices are joined by an edge if there is at least one physical link between the corresponding ASs. The AS topology has been investigated by a number of authors, see e.g. [6, 15, 4, 17].

¹Supported by the joint Berlin/Zurich graduate program Combinatorics, Geometry, and Computation (CGC), financed by ETH Zurich and the German Science Foundation (DFG).

Different ASs exchange routing information using the Border Gateway Protocol (BGP). If an AS X is connected to another AS Y , it *announces* the routes to a certain set $A(X, Y)$ of destination addresses to Y . This implies that when Y has a packet with a destination in the set $A(X, Y)$, it can forward the packet to the AS X , because X knows a route to that destination. An AS may choose not to announce all routes that it knows to all of its neighbors; this decision is determined by the routing policy it employs.

While the AS topology provides information about the connections between ASs, it has been pointed out in [9] that it is also important to know the economic relationships between ASs, because these relationships determine the routing policies and thus the paths that packets can potentially take in the network. If an AS X is connected to an AS Y , then Y can be a customer, provider, or peer of X . If Y is a customer of X , then X announces all its routes to Y . If Y is a provider or a peer of X , then X announces only the routes to destinations in its own AS and to destinations announced by its own customers; it does (usually) not announce routes to destinations that it can reach only through another provider or through peers. By knowing customer-provider relationships between ASs, one could have a sound basis for investigations of routing issues in the Internet.

Since data about these relationships is not easy to obtain directly, however, a natural idea is to infer AS relationships from the routing paths observed in the network (these paths can be determined from BGP information). Subramanian et al. [16] pursue this approach and propose a formulation of this inference problem as a combinatorial optimization problem, called the Type-of-Relationship problem: given a graph and a set of paths in the graph, classify the edges of the graph into customer-provider relationships and peer-peer relationships such that as many of the paths as possible are valid (consistent with this classification). Here, a path is valid if it consists of zero or more customer-provider edges, followed by at most one peer-peer edge, followed by zero or more provider-customer edges. Subramanian et al. write that they suspect the problem to be \mathcal{NP} -hard but have been unable to prove this, and they propose a heuristic algorithm.

Inference of AS relationships is e.g. applied by Mahajan et al. [14]. The authors investigate certain misconfigurations of BGP routers. To this aim they utilize a heuristic inference algorithm presented by Gao [9].

1.1 Why investigate approximability and in-approximability

Many interesting “real world” optimization problems turn out to be \mathcal{NP} -hard, i.e. the possibility of finding a polynomial time algorithm for such a problem is widely considered as very unlikely. As a natural consequence, heuristic approaches yielding good results quickly, i.e. in polynomial time, are sought. Often these heuristics are quite hard to analyze and no guarantee on their worst-case performance can be given. This might be satisfactory in many cases, but a provably good heuristic, also called *approximation algorithm*, has obvious advantages and in some cases might even be a necessity. With “provably good” we mean that a ratio can be given such that the quality of the solution computed by the algorithm is not worse than this ratio times the optimum for any input. Formally in the case of maximization problems for all possible inputs it must hold that $A \geq c \cdot Opt$, where A , Opt are the objective values of the approximate respectively optimal solution and $c < 1$ is the *approximation ratio*. Such an algorithm is also called a c -approximation algorithm. If c is a constant the problem is said to be in the complexity class \mathcal{APX} .

Conversely when an approximation algorithm with a certain ratio c was found, it is obviously of interest to know whether this c can be improved or not. For many problems certain *inapproximability* results can be obtained. I.e. for a certain c_0 it can be shown that there is no c approximation with $c \geq c_0$, unless $\mathcal{P} = \mathcal{NP}$. To give a concrete example: if the problem MAX2SAT could be approximated within a ratio of 0.955 or better this would imply that polynomial time algorithms exist for all problems in \mathcal{NP} . As mentioned this is considered very unlikely. The ratio c_0 gives an upper bound of how good an approximation algorithm can be. Of course it is desirable to have an approximation algorithm whose ratio is close to c_0 .

For a complete introduction to approximation algorithms see for instance [2]. The book also contains a comprehensive list of approximability and inapproximability results.

1.2 Our results

In this paper, we consider two versions of the Type-of-Relationship problem: the problem of computing a classification such that *all* paths are valid (if such a classification exists), denoted ALLTOR, and the problem of computing a classification that maximizes the number of valid paths, denoted MAXTOR. We prove that ALLTOR can be solved in linear time by reducing it to 2SAT and we show that MAXTOR is \mathcal{NP} -hard, thus settling the complexity of the Type-of-Relationship problem left open in [16]. Our hardness proof implies that MAXTOR cannot be approximated within $1/n^{1-\varepsilon}$ (for any $\varepsilon > 0$) for general instances with n paths unless $\mathcal{NP}=\text{co-}\mathcal{RP}$. Motivated by the characteristics of instances arising in practice, we then consider the case where the given paths are short. First, we show that MAX-

TOR can be approximated within a factor of $(k+1)/2^k$ if all paths have length at most k . Then we give approximation algorithms achieving a better approximation ratio for $k = 2, 3, 4$. We also prove that MAXTOR is \mathcal{APX} -complete for paths of bounded length. \mathcal{APX} -completeness (see e.g. [2]) implies that MAXTOR cannot be approximated within a certain constant factor unless $\mathcal{P} = \mathcal{NP}$, even for instances with short paths. Finally, we have implemented our approximation algorithm and obtained very encouraging results on real data sets.

Independently of our work, Di Battista, Patrignani and Pizzonia [5] have recently also obtained the result that ALLTOR can be solved in linear time and that MAXTOR is \mathcal{NP} -hard. Interestingly, they also use 2SAT to solve the ALLTOR problem and present an \mathcal{NP} -hardness proof by reducing MAX2SAT. Their reduction is based on the same idea with which we are able to prove \mathcal{APX} -hardness. Di Battista et al. do not consider the question of approximability or inapproximability of MAXTOR. Instead, they give an \mathcal{NP} -hardness result for the problem of maximizing peer-peer relationships and present a new heuristic algorithm for MAXTOR. They do not give approximation bounds for the algorithm, but they show that it performs well on real data sets.

The remainder of the paper is structured as follows. Definitions and preliminaries are given in Section 2. Section 3 deals with the ALLTOR problem. The hardness results for general MAXTOR are given in Section 4. In Section 5, the approximability of MAXTOR instances with paths of bounded length is studied. Section 6 describes our experimental results. We conclude in Section 7.

2 Preliminaries

We are given a simple, undirected graph $G = (V, E)$ whose nodes correspond to the autonomous systems of the Internet and whose edges correspond to physical connections between those autonomous systems. Furthermore, we are given a set P of simple, undirected paths in G . (We allow that P contains a path several times, i.e., P can actually be a multi-set.) The length of a path is defined as the number of edges on the path. All nodes on a path except its two endpoints are called *internal* nodes of the path. These paths P are possible data routes in the Internet; they are usually obtained from BGP routing tables. Informally, the goal is to classify the edges of G as customer-provider or peer-peer relationships in a “correct” way by using only the information obtained from the set of paths P . Here, the basic assumption is that the routing policies of the ASs depend on these relationships in the way described in Section 1, thus motivating the following definition (given in [16]).

Definition 1 *For a classification of the edges of G into customer-provider and peer-peer relationships, a path $p \in P$ is valid if: it starts with zero or more customer-provider edges; followed by zero or one peer-peer edge; followed by zero or more provider-customer edges.*

The problem of classifying the edges into peer-peer or customer-provider relationship such that a maximum number of paths are valid was posed as the Type-of-Relationship (ToR) problem in [16]. We consider two versions of this problem: In the first variant, denoted ALLTOR, the goal is to decide whether there is an edge classification such that all paths in P are valid, and to compute such a classification if it exists. In the second variant, denoted MAXTOR, the goal is to compute an edge classification such that a maximum number of paths in P are valid.

For convenience we represent a customer-provider edge as a directed edge from customer to provider and a peer-peer edge as a bidirected edge between the two peers as shown in Figure 1.



Figure 1. Representation of customer-provider and peer-peer relationship.

Figure 2 gives some examples of valid and invalid paths. Using the directed and bidirected edge formulation for customer-provider and peer-peer relationships, we can rephrase the validity of a path as follows, where we take a bidirected edge to be pointing away from both of its endpoints.

Lemma 2 For a given edge classification (represented as directed and bidirected edges), a path p is valid if and only if it does not contain a node from which two edges of p are pointing away.

Such a situation, where two edges point away from an internal node of some path, is also called an *anomaly*.

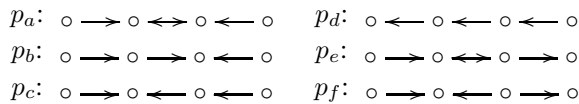


Figure 2. Paths p_a, \dots, p_d are valid, p_e and p_f are not valid.

Finally, we argue that peer-peer edges can be completely disregarded in this formulation of the problem. To see this, assume that for a graph G and a set of paths P , the relationships between the ASs have been classified as directed or bidirected edges. Now replace every bidirected edge by a directed edge in arbitrary direction. Using Lemma 2, we see immediately that all the valid paths remain valid after this operation. Figure 2 illustrates this transition for both directions from path p_a to path p_b and from path p_a to p_c , respectively. Therefore, we can assume without loss of generality that the solutions for MAXTOR and ALLTOR computed by an algorithm do not contain any bidirected edges.¹

¹From a practical view, it might be interesting to find an alternative formulation of the problem where peer-peer edges are actually necessary.

An assignment of directions to the edges of an undirected graph is called an *orientation* of the graph. We can now state the ALLTOR and MAXTOR problems simply as follows:

ALLTOR: Given a graph G and a set of paths P , decide whether there is an orientation of G such that all paths in P are valid, and compute such an orientation if it exists.

MAXTOR: Given a graph G and a set of paths P , compute an orientation of G such that a maximum number of paths in P are valid.

In the case of MAXTOR, we will be interested in approximation algorithms. Here, an algorithm for MAXTOR is a ρ -approximation algorithm if it runs in polynomial time and always computes an orientation of G such that the number of valid paths is at least ρ times the number of valid paths in the optimal orientation. Note that $\rho \leq 1$.

3 The ALLTOR problem

In this section we show that the ALLTOR problem is solvable in linear time. To this end we show that an instance of ALLTOR can be reduced to a 2SAT problem.

Lemma 3 A path $p \in P$ of length k can be split up into $k - 1$ paths p_i , $1 \leq i \leq k - 1$, each of length 2, such that for any orientation of G , p is valid if and only if all p_i are valid.

Proof: The construction of the $k - 1$ paths works as follows: path p_1 consists of the first two edges of p . Path p_i consist of edge i and edge $i + 1$ of path p . In this way, path p_i overlaps with path p_{i+1} by one edge. Figure 3 shows how a path of length 3 is split up into two paths of length 2. The correctness of the lemma follows directly from Lemma 2. \square

Decompose p :	directed p_1	valid	clause
$\circ \xrightarrow{e_1} \circ \xleftrightarrow{e_2} \circ \xrightarrow{e_3} \circ$	$\circ \xrightarrow{e_1} \circ \xleftrightarrow{e_2} \circ$	yes	$e_1 \vee e_2$
into p_1, p_2 :	$\circ \rightarrow \circ \rightarrow \circ$	yes	$e_1 \vee \bar{e}_2$
$\circ \xrightarrow{e_1} \circ \xleftrightarrow{e_2} \circ$	$\circ \leftarrow \circ \leftarrow \circ$	yes	$\bar{e}_1 \vee e_2$
$\circ \xrightarrow{e_2} \circ \xleftrightarrow{e_3} \circ$	$\circ \leftarrow \circ \rightarrow \circ$	no	$\bar{e}_1 \vee \bar{e}_2$

Figure 3. Decomposition of paths (left), truth table for a length 2 path (right).

By Lemma 3, it suffices to consider the ALLTOR problem for instances with paths of length 2. (Paths of length 1 are always valid.) The truth table on the right-hand side of Figure 3 shows a one-to-one correspondence between the validity of a path of length 2 and the logical or of two literals. This suggests the use of 2SAT² to solve the ALLTOR problem, resulting in the following algorithm:

²A 2SAT instance consists of a set of clauses with only two literals, e.g. $x_1 \vee x_2, \bar{x}_3 \vee x_2, \dots$, the question being: is there a truth-assignment to x_1, x_2, \dots , so that all clauses are *satisfied*, i.e. evaluate to true?

1. Orient the edges of $G = (V, E)$ arbitrarily.
2. Split all paths $p \in P$ into $\sum_{p \in P} (\ell(p) - 1)$ paths of length 2 as shown above, where $\ell(p)$ is the length of path p .
3. Construct a 2SAT instance where each edge $e_i \in E$ corresponds to a variable and each path p (of length 2) corresponds to a clause in the following way: like in the truth table of Figure 3, e_i appears negated if the corresponding edge is pointing away from the internal node v of p and not negated if it is pointing towards v .
4. Solve the resulting 2SAT instance.
5. If the 2SAT instance is not satisfiable, the ALLTOR instance is not solvable. Otherwise, flip the directions of all edges whose corresponding variable has been assigned *false*. This gives an orientation where all paths are valid.

Correctness of the algorithm can be verified easily by looking at the four possible configurations for input paths of length two and the assignment of the variables in the corresponding clause. Steps 1, 2, 3 and 5 can obviously be performed in time linear in the size of the input. 2SAT is well known to be solvable in linear time, so the running-time for Step 4 is linear as well.

Theorem 1 ALLTOR can be solved in linear time.

4 The general MAXTOR problem

In this section we show that MAXTOR is \mathcal{NP} -hard and even cannot be approximated well. To achieve this we will give an approximation-preserving polynomial reduction from the well known \mathcal{NP} -hard maximum independent set problem (denoted MAXIS, problem definition: given a graph, find a maximum set of nodes such that no two nodes are connected by an edge) to MAXTOR. First, consider a graph G with two paths as shown in Figure 4. It can be verified that in this graph there is no orientation of the edges such that both paths are valid: Each of the three shared edges of both paths would require to flip the direction in one of the paths to make them valid. But since a change of direction of the edges in a valid path can happen only once, one of the two paths must be invalid.

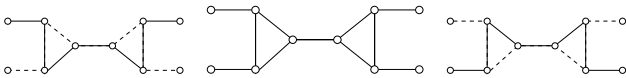


Figure 4. A graph (middle) with two paths (left and right) that cannot both be valid in any orientation.

Lemma 4 There exists a graph G with two paths such that only one of the two paths can be valid.

We will use this construction as a gadget to obtain a reduction from MAXIS to MAXTOR. Let an instance of MAXIS be given by an undirected graph $H = (V_H, E_H)$. We create an instance (G, P) of MAXTOR by mapping every node in H to a path in P such that any two paths in P are edge-disjoint if there is no edge between them in H respectively cannot be valid simultaneously in any orientation if there is an edge between them in H .

Obviously, such an instance (G, P) can be constructed in polynomial time using the gadget of Figure 4.

Now observe that there is a one-to-one correspondence between orientations of G with t valid paths and independent sets in H with cardinality t . In particular, if we could solve MAXTOR in polynomial time or approximate it in polynomial time with ratio ρ , we could also solve MAXIS in polynomial time or approximate it with ratio ρ , respectively. Since MAXIS is known to be \mathcal{NP} -hard and not even approximable with ratio $1/n^{1-\epsilon}$ on graphs with n nodes for any $\epsilon > 0$ unless $\mathcal{NP} = \text{co-}\mathcal{RP}$ [11], we obtain the following hardness result for MAXTOR.

Theorem 2 MAXTOR is \mathcal{NP} -hard and cannot be approximated within a factor of $1/n^{1-\epsilon}$ on instances with n paths for any $\epsilon > 0$ unless $\mathcal{NP} = \text{co-}\mathcal{RP}$.

Thus it is not feasible to get a good approximation ratio for the general MAXTOR problem. We might hope, however, to be able to exploit the structure of real AS graphs or the observed routing paths in order to give an algorithm with good approximation ratio for a restricted case of MAXTOR.

5 Approximating MAXTOR instances with bounded path length

Motivated by the negative result concerning the approximability of the general MAXTOR problem, we now consider a restricted version of the problem. Looking at real world data [16, 1], we noticed that most of the paths are actually quite short (see also Section 6). This leads us to consider instances of MAXTOR where the maximum length of the paths is bounded by a constant. In the following, we first present a very simple randomized approach achieving constant approximation ratio for this case. Then we show how to use an approximation algorithm for MAX2SAT to achieve significantly better approximation ratios for instances containing only paths of length at most k for $k = 2, 3, 4$. Finally, we prove \mathcal{APX} -completeness for instances where the path length is bounded by an arbitrary constant, by an approximation preserving reduction from the MAX2SAT problem, which is well known to be \mathcal{APX} -complete [2].

5.1 A simple constant factor approximation algorithm

For paths of constant length there is a very easy randomized approximation algorithm: just select the directions of the edges independently at random. If each edge is oriented in one of the two possible ways with probability $1/2$, a path of length k is valid with probability

$$p_k = \frac{k+1}{2^k}.$$

To see this, note that in a valid path the direction of the edges can change only once at one of the $k-1$ internal nodes or not at all. There are $k-1$ possibilities for the direction of all edges in the path in the former case and 2 possibilities in the latter case. Altogether there are 2^k possibilities to orient the edges. If all paths have length at most k , we get by linearity of expectation $\mathbb{E}(A_{rand}) \geq n \cdot (k+1)/2^k \geq Opt \cdot (k+1)/2^k$, where A_{rand} is the value of the approximate solution and Opt the value of the optimum. The algorithm can easily be derandomized in the standard way (method of conditional probabilities), giving the following theorem.

Theorem 3 *The MAXTOR problem with paths no longer than k edges can be approximated within a factor of $\frac{k+1}{2^k}$ of the optimum in polynomial time.*

For example, this gives ratio 0.75 for paths of length at most 2, 0.5 for paths of length at most 3, and $5/16 = 0.3125$ for paths of length at most 4. We can also state the following corollary.

Corollary 5 *The MAXTOR problem with average path length bounded by k can be approximated within a factor of $\frac{k+2}{(k+1) \cdot 2^{k+1}}$ of the optimum in polynomial time.*

Proof: Consider an instance of MAXTOR with n paths. If the average path length is k , there are at least $n/(k+1)$ paths with length at most $k+1$. Applying the simple randomized algorithm described above to these paths, we obtain a solution with at least $\frac{n}{k+1} \cdot \frac{k+2}{2^{k+1}}$ valid paths. \square

5.2 Better ratios for paths of short length

To obtain better ratios for paths of length at most 4 or less we use the approximation algorithm for MAX2SAT due to Lewin, Livnat and Zwick [13]. They achieve a ratio of $r = 0.940$ by enhancing the semidefinite programming (SDP) based approach first presented in the seminal paper by Goemans and Williamson [10]. Given an instance (G, P) of MAXTOR, we construct an instance of MAX2SAT from the paths as described in Section 3 (by splitting every path into paths of length two and adding a clause for each length 2 path) and apply the MAX2SAT approximation algorithm to the resulting instance. Then we orient the edges of G according to the assignment returned by the MAX2SAT algorithm. It may seem surprising that this approach gives a good ratio because there is

not necessarily a one-to-one correspondence between paths and clauses.

First, a bit of notation: Let Opt denote the optimum value of the considered MAXTOR instance and A_k the value of our approximate solution for an instance with path length bounded by k . Let g_k be the number of paths in P that have length exactly k . Note that paths of length 1 can be ignored, since they are always valid. Hence, we can assume $g_1 = 0$. Furthermore, note that for deriving lower bounds on the approximation ratio A_k/Opt , it suffices to consider only instances in which all paths have length exactly k : If an instance contains a shorter path, this path can easily be lengthened by adding an appropriate number of extra nodes and edges to the path at one of its ends. At most $k \cdot n$ edges and nodes are added, and a solution to this modified instance clearly gives a solution to the original instance with at least the same number of valid paths.

In the simplest case, when all paths have length 2, we can directly transfer the ratio $r = 0.940$ from MAX2SAT to MAXTOR. This is because each path is represented by exactly one clause which is satisfied if and only if the path is valid.

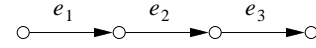


Figure 5. A path of length 3 which leads to the two clauses $e_1 \vee \bar{e}_2$ and $e_2 \vee \bar{e}_3$. See Section 3 for details about the definition of the clauses given a graph and paths.

Now consider a path of length 3. It is represented by two 2SAT clauses. The variable corresponding to the edge in the middle appears in both clauses, once negated and once not negated (see Figure 5 for an example). Therefore, one of the two clauses is always satisfied. Clearly, both clauses are satisfied if and only if the path is valid. This gap of either one or two clauses being satisfied can be used to derive a bound on the approximation ratio. Consider a MAXTOR instance with $n = g_3$ paths of length 3 with optimal value Opt . Note that an optimal solution of MAX2SAT has the value $Opt + g_3$ and directly gives an optimal solution to the MAXTOR problem. The MAX2SAT approximation algorithm satisfies $A_3 + g_3$ clauses with

$$\frac{A_3 + g_3}{Opt + g_3} \geq r. \quad (1)$$

Because there is always a solution to MAX2SAT such that at least $3/4$ of the clauses are satisfied and there are $2 \cdot g_3$ clauses, we have $A_3 + g_3 \geq 3/2 \cdot g_3$ or $g_3 \leq 2 \cdot A_3$. Applying this to (1) leads to

$$\begin{aligned} A_3 &\geq r \cdot (Opt + g_3) - g_3 \geq r \cdot Opt + 2(r-1) \cdot A_3 \\ A_3 &\geq \frac{r}{3-2r} Opt \end{aligned}$$

giving approximation ratio $r/(3-2r) \geq 0.839$. Note that this is a considerable improvement over the ratio $1/2$ obtained for paths of length three by Theorem 3.

In a MAX2SAT instance derived from paths of length 4 there will be three clauses for each of the paths. We refer to the three clauses of a path as a *triple*. With the same argumentation as for length 3 paths, we have that for each triple at least one clause is always satisfied and all three are satisfied if and only if the corresponding path is valid. This shows that any solution of this instance satisfies

$$x + y + g_4$$

clauses, where g_4 , y and x are the number of triples with at least one, at least two and exactly three satisfied clauses, respectively. Note that x yields the number of valid paths and $x \leq y \leq g_4$.

We now compare a solution $S = A_4 + y + g_4$ computed by the MAX2SAT approximation algorithm with a solution $S' = Opt + y' + g_4$ derived from an optimal solution of the corresponding MAXTOR instance. By [13] we know that $S/S' \geq r$. From this we can bound the approximation ratio A_4/Opt . In the worst case $y = g_4$ and $y' = Opt$. This gives

$$\frac{A_4 + 2 \cdot g_4}{2 \cdot Opt + g_4} \geq r$$

$$A_4 \geq 2r \cdot Opt + (r - 2) \cdot g_4 \geq 2r \cdot Opt + 4(r - 2)A_4,$$

because there are $3 \cdot g_4$ clauses of which at least $3/4$ are satisfied in the approximate solution, i.e. $A_4 + 2 \cdot g_4 \geq 9/4 \cdot g_4$ or $g_4 \leq 4 \cdot A_4$. Solving for A_4 we get

$$A_4 \geq \frac{2r}{9 - 4r} \cdot Opt.$$

This gives an approximation ratio of $2r/(9 - 4r) \geq 0.358$, which is a slight improvement compared to $5/16 = 0.3125$.

Note that this approach cannot be carried over to paths of length greater than 4. It uses the fact that at least $3/4$ of the clauses can be satisfied, which does not help if each path is represented by 4 or more clauses and the path is valid if and only if all of them are satisfied.

Summarizing the results discussed above, we get the following theorem.

Theorem 4 *The MAXTOR problem with paths no longer than k edges can be approximated within a factor c_k of the optimum in polynomial time, where c_k has the following form: $c_2 := 0.940$, $c_3 := 0.839$, $c_4 := 0.358$, and $c_k := \frac{k+1}{2^k}$ for $k > 4$.*

5.3 \mathcal{APX} -hardness

So far we have given constant-factor approximation algorithms for instances of MAXTOR with paths of bounded length. Now we show that even instances containing only paths of length 2 cannot be approximated better than some constant unless $\mathcal{P} = \mathcal{NP}$. Both results together give that MAXTOR with constant maximal path length is \mathcal{APX} -complete.

To this aim, we reduce MAX2SAT and apply an \mathcal{APX} -hardness result by Håstad [12]. We start by reducing a MAX2SAT instance to a MAXTOR instance (G, P) with paths of length 3. Then we explain how this instance can be modified such that it contains only paths of length 2.

Assume that we are given a MAX2SAT instance with variables $x_i, i \in \{1 \dots n'\}$ and clauses $c_j, j \in \{1 \dots m'\}$. For each variable x_i we add two nodes $\overline{x_i}, x_i$ to G and an edge $e_i = \{\overline{x_i}, x_i\}$ between them. If in a solution to MAXTOR this edge is directed towards x_i this corresponds to $x_i = \text{true}$. Otherwise, if it is directed towards $\overline{x_i}$, this means $x_i = \text{false}$. For each clause $c_k = l_i \vee l_j$, with literals $l_i \in \{x_i, \overline{x_i}\}$ and $l_j \in \{x_j, \overline{x_j}\}$, one edge $\{l_i, l_j\}$ is added. Additionally a path of length 3 is added to P along the nodes $\overline{l_i}, l_i, l_j, \overline{l_j}$. Figure 6 shows a simple example with one clause, and the graph on the left-hand side of Figure 7 shows an example with two clauses.

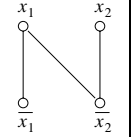
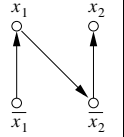
clause	path	graph	directed	truth val
$x_1 \vee \overline{x_2}$	$\overline{x_1}, x_1,$ $\overline{x_2}, x_2$			$x_1 = 1,$ $x_2 = 1$

Figure 6. Example of how an instance of MAXTOR is constructed from a MAX2SAT instance. On the right-hand side, a possible assignment of directions to the edges and the corresponding truth values are shown.

The paths and edges are defined such that a path is valid in a solution to the MAXTOR instance if and only if the corresponding clause is satisfied. This is clear because a path is valid if and only if either e_i is directed towards l_i or e_j towards l_j , which corresponds to a truth assignment where $l_i \vee l_j$ is satisfied. In particular, note that given a satisfied clause the edge $\{l_i, l_j\}$ can always be directed such that the whole path is valid. Conversely, for an unsatisfied clause no such direction of $\{l_i, l_j\}$ exists.

Thus, maximizing the number of valid paths also maximizes the number of satisfied clauses. With [12] we get that MAXTOR is not approximable within ratio $q = 0.955$ for instances with paths of length at most k for any constant $k \geq 3$.

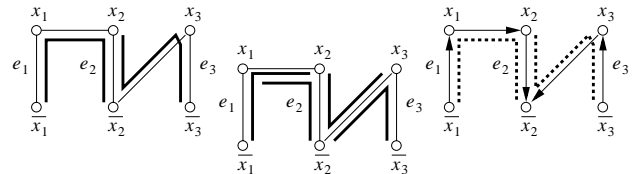


Figure 7. Network and paths resulting from the two clauses $x_1 \vee x_2, \overline{x_2} \vee x_3$: Constructed instance with length 3 paths (left), modified instance with length 2 paths (middle), and possible solution where both clauses are satisfied (right).

To obtain a similar result for paths of length 2, we modify the instance as follows: each path $\bar{l}_i, l_i, l_j, \bar{l}_j$ is replaced by two overlapping paths \bar{l}_i, l_i, l_j and l_i, l_j, \bar{l}_j . See the graph in the middle of Figure 7 for an example. Clearly, in a MAXTOR solution one of the two paths is always valid. The corresponding clause is satisfied if and only if both paths are valid. So an optimal solution to the MAXTOR instance with Opt valid paths gives an optimal assignment to the variables such that $Opt - m'$ clauses are satisfied. An approximate solution to MAXTOR giving A_2 valid paths leads to $A_2 - m'$ satisfied clauses. With [12] we know that

$$\frac{A_2 - m'}{Opt - m'} \leq q$$

for at least one instance of MAX2SAT. With $Opt - m' \geq 3/4 \cdot m'$ (at least 3/4 of the clauses of any MAX2SAT instance can be satisfied) this yields

$$A_2 \leq \frac{4 + 3q}{7} \cdot Opt,$$

which gives the following theorem.

Theorem 5 *Unless $\mathcal{P} = \mathcal{NP}$, there is no approximation algorithm for MAXTOR with paths of length at most k that achieves ratio at least $\frac{4+3q}{7} \leq 0.981$ if $k = 2$ and ratio at least 0.955 if $k \geq 3$.*

In particular, with Theorem 4 we have that MAXTOR is \mathcal{APX} -complete for paths with constant maximum length.

6 Experimental results

In this section we discuss some experiments conducted with the approximation algorithm for MAXTOR presented in Section 5.2: the algorithm constructs a MAX2SAT instance from the given paths and uses an SDP-based algorithm to obtain an approximate solution. For our experiments we used the data that has been accumulated by Subramanian et al. [16] and is available on the WWW [1]. For four different dates (18 April 2001, 4 February 2002, 6 April 2002, and 9 July 2002) routing paths from 10, 9, 14, respectively 12 autonomous systems were collected. From these 1.4 to 6.4 million paths in a network of about 10,000 nodes and 25,000 edges, the edge directions were deduced. It is notable that the path lengths are relatively short (see Table 1), i.e. the assumption in Section 5.2 that the path lengths are bounded by a small constant is quite realistic.

In our implementation the paths are first preprocessed by directing edges that can be directed without conflicts. This shortens the paths considerably (cf. Table 1). Then it is checked with the ALLTOR algorithm described in Section 3 whether the graph can be oriented such that all paths are valid. This was not the case for any of the four dates. Finally, an approximate solution is calculated as described in Section 5.2: MAX2SAT is relaxed as in [10] and the rounding is done as in [7]. In [7] the improved approximation ratio could be achieved by adding new constraints

Table 1. Size of the network and number of paths at the three different dates. The maximal and average path lengths are given before (m_1 and a_1) and after (m_2 and a_2) the preprocessing described in the text.

Date	nodes	paths	m_1 / m_2	a_1 / a_2
04/18/01	10923	3423422	12 / 10	3.5 / 1.7
02/04/02	12788	4988100	11 / 9	3.5 / 1.3
04/06/02	13164	6356435	11 / 8	3.5 / 1.4
07/09/02	13409	1406465	11 / 8	3.1 / 1.3

and a modified rounding strategy. We could not add the constraints because the instance would have become too large. The new rounding strategy was adopted though. The freely available solver DSDP 4.5 [3] was used to solve the semidefinite programs. The overall results are given in Table 2.

Table 2. Experimental results achieved with the approximation algorithm presented in Section 5.2. The two rightmost columns give the corresponding results of Subramanian et al.'s algorithm [16] (solutions can be downloaded at [1]) and Gao's algorithm [9] (using an implementation available at [8]).

Date	valid paths	[16]	[9]
04/18/01	3360926 (98.17%)	70.57%	70.10%
02/04/02	4919310 (98.62%)	66.60%	72.36%
04/06/02	6204849 (97.62%)	68.00%	74.43%
07/09/02	1400565 (99.58%)	70.26%	89.12%

The computed orientations make roughly 98% of the given paths valid. This should be contrasted with the edge classifications computed by the heuristic algorithm of Subramanian et al. [16] that are also available at [1] respectively the algorithm of Gao [9] for which an implementation can be downloaded at [8]. In these edge classifications, only about 70% respectively 70%–90% of the paths are valid. This demonstrates that it pays off to use SDP-based approximation algorithms for MAXTOR in practice. Note: In the input data a single path can occur several times. In the solutions obtained with Gao's and our approach we consider such multiple occurrences. The classifications given at [1] though are optimized for the case where each path is only considered once. Therefore the values in the third column of Table 2 refer to this case. The output of Gao's algorithm also contains so called *sibling* edges between closely related ASs. For the evaluation we treat these edges as peer-peer relationships.

Recall that an anomaly is a situation where two edges point away from an internal node of some path. Thus each anomaly in a solution invalidates at least one of the given paths. An interesting question is how the invalid paths are distributed over the anomalies. In other words: How many

anomalies are responsible for invalidating the majority of the invalid paths? It turns out that few anomalies invalidate most of these paths, e.g. in the solution for the 18th of July 2001 only 16.5% of the anomalies are responsible for 80% of the invalid paths. Details can be seen in Table 3.

Table 3. The number of anomalies that are responsible for invalidating 20%, 40% ... 100% of the invalid paths. Paths are counted separately for each anomaly they contain. Note that less than 0.1% of the invalid paths contain more than one anomaly.

Date	20%	40%	60%	80%	100%
04/18/01	0.8%	2.6%	6.9%	16.5%	100%
02/04/02	0.8%	2.3%	5.7%	14.4%	100%
04/06/02	0.4%	1.4%	4.0%	11.6%	100%
07/09/02	1.2%	3.0%	7.1%	15.4%	100%

7 Conclusion

In this paper we settle the complexity of the Type-of-Relationship problem, which was left open in [16]. In addition to proving that the maximization version is \mathcal{NP} -hard we are even able to show that it is very hard to find a good approximate solution in the general case. Thus we restrict ourselves to instances with only relatively short paths. This is a natural assumption in the TCP/IP-context because packets aren't routed along paths which contain more than a certain number of hops. With that restriction we can give an approximation algorithm with good constant ratio which depends on the longest path length. For the cases that all paths have length at most 2 respectively 3 the ratio is quite close to an upper bound we are able to prove. Checking the algorithm's performance on actual BGP routing-data yields that it also works very well in practice.

An interesting open question remaining is how to nicely integrate the notion of peer-peer relationships into the maximization setting. In the model which was proposed in the literature, these edges are of no benefit and therefore don't have to be taken into account so far.

References

[1] S. Agarwal. Data used in [16]. <http://www.cs.berkeley.edu/~sagarwal/research/BGP-hierarchy/>, 2002.

[2] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation. Combinatorial Optimization Problems and their Approximability Properties*. Springer, Berlin, 1999.

[3] S. Benson. Semidefinite programming solver DSDP 4.5. <http://www-unix.mcs.anl.gov/~benson/>, 2002.

[4] T. Bu and D. Towsley. On distinguishing between Internet power law topology generators. In *Proceedings of INFOCOM'02*, June 2002.

[5] G. Di Battista, M. Patrignani, and M. Pizzonia. Computing the types of the relationships between autonomous systems. Technical Report RT-DIA-73-2002, Dipartimento di Informatica e Automazione, Università di Roma Tre, 2002.

[6] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the Internet topology. In *Proceedings of ACM SIGCOMM'99*, 1999.

[7] U. Feige and M. Goemans. Approximating the value of two proper proof systems, with applications to MAX 2SAT and MAX DICUT. In *Proceedings of the Third Israel Symposium on Theory of Computing and Systems*, pages 182–189, 1995.

[8] D. R. Figueiredo, Z. Ge, and S. Jaiswal. Implementation of algorithm given in [9]. <http://www-net.cs.umass.edu/~ratton/AS/>, 2002.

[9] L. Gao. On inferring autonomous system relationships in the internet. *IEEE/ACM Transactions on Networking*, 9:733–745, 2000.

[10] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995.

[11] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182:105–142, 1999.

[12] J. Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.

[13] M. Lewin, D. Livnat, and U. Zwick. Improved rounding techniques for the MAX 2-SAT and MAX DICUT problems. In *Integer Programming and Combinatorial Optimization (IPCO)*, LNCS 2337, pages 67–82, 2002.

[14] R. Mahajan, D. Wetherall, and T. Anderson. Understanding BGP misconfiguration. In *Proceedings of ACM SIGCOMM'02*, 2002.

[15] A. Medina, I. Matta, and J. Byers. On the origin of power laws in Internet topologies. *ACM Computer Communication Review*, 30(2):18–28, April 2000.

[16] L. Subramanian, S. Agarwal, J. Rexford, and R. Katz. Characterizing the Internet hierarchy from multiple vantage points. In *Proceedings of INFOCOM'02*, June 2002.

[17] D. Vukadinović, P. Huang, and T. Erlebach. On the spectrum and structure of Internet topology graphs. In *Innovative Internet Computing Systems*, LNCS 2346, pages 83–95, June 2002.