Interdomain Traffic Engineering with minimal BGP configurations

Steve Uhlig[a], Olivier Bonaventure[a], Bruno Quoitin[b]

[a]Computer Science and Engineering Department, Université Catholique de Louvain.
E-mail:{suh,Bonaventure}@info.ucl.ac.be

[b]Computer Science Institute, Facultés Universitaires Notre-Dame de la Paix à Namur.
E-mail: bqu@infonet.fundp.ac.be

We propose a method based on multi-objective combinatorial optimization to perform interdomain traffic engineering with minimal BGP configurations. Our method relies on an evolutionary algorithm that tries to minimize an objective function by finding the successive BGP filters to be applied on the BGP routes. We study the impact of the number of providers, the traffic aggregation in the AS-level topology and the objective function on the behavior of our algorithm.

1. Introduction

The Internet nowadays is becoming more and more complex. Its size as well as its technical and economical complexity both contribute to this phenomenon. As a consequence, the task of today's network managers is becoming more and more difficult. While within the boundaries of a single organization some control of the traffic is possible thanks to the good knowledge of the internal network topology, beyond the Internet access points things become a little more intricate. The Internet routing system today is divided into two conceptual views: intradomain and interdomain. At the intradomain level, the topology is known and the control of the routing information can be assumed. Engineering the flow of the traffic within an organization is thus possible.

At the interdomain level on the other hand, the approach used today in the Internet is the one of a distributed control of the routing information. The current interdomain routing protocol, BGP [7], knows only a limited part of the topology of the Internet. The global Internet is divided into Autonomous Systems (AS), each running an intradomain routing protocol within its boundaries and using BGP to exchange routing information with other ASes. The objective of BGP is to distribute information about reachability of IP prefixes to other ASes. With BGP, an AS advertises to each neighbor AS all the networks it can reach. A key feature of BGP is that it allows each network operator to define its routing policies. Those policies are implemented by preferring some routes over others and also by filters. Those filters allow an operator to advertise all its routes or a fraction of its routes to neighboring ASes on a per-AS basis. Traffic engineering at the interdomain level is hence particularly difficult [5] since the network manager does not have a detailed information about the topology between its AS and the sources and destinations of its interdomain traffic (the traffic that comes from or goes to other ASes).

The ASes we target in this paper are stub ASes that constitute the majority of the ASes in the Internet [8] (at the date of July 29th 2002 more than 80 % of the ASes could be considered as stub-ASes). A stub-AS is an AS that does not provide transit service for other ASes. It can host content or provide access to dialup or broadband users. More specifically, our focus is on stub ASes having several access links to the Internet. The problem we are trying to tackle is the one of an AS that wants to distribute its traffic among several access links in order to optimize an objective function defined on the total traffic exchanged over each link. Almost all ASes have to pay large

ISPs [8] to get a global connectivity on the Internet. The cost of the traffic on these links tends to become quite large while the bandwidth prices tend to vary much between ISPs (see `http://www.telegeography.com`) so some significant gain could be achieved by redistributing some traffic from more expensive links towards less expensive ones.

The remainder of this paper is structured as follows. Section 2 introduces the main features of a simplified version of our problem. Section 3 explains the combinatorial nature of interdomain traffic engineering. Section 4 presents the multi-objective combinatorial optimization problem. Section 5 describes the evolutionary algorithm aimed at solving the multi-objective combinatorial optimization problem. Section 6 describes the traffic demands on which we rely to evaluate the performance of our method. Section 7 provides detailed simulation results aimed at evaluating our evolutionary approach.

2. The "simple" combinatorial optimization problem

To introduce the problem we intend to solve, we first discuss in this section a simplified version of the problem. Assume a stub AS having several interdomain links with different providers. Our stub AS receives a full BGP routing table from each of its providers. To control the flow of its interdomain traffic, our ISP can rely on the information found in these BGP routing tables. BGP is a path vector routing protocol. Each BGP router sends BGP advertisements to its peers. A route advertisement sent by an AS through BGP means that the AS that advertises the route agrees to forward IP packets to the destinations corresponding to this route. In addition, the AS PATH attribute [7] contained in this route advertisement also tells through which ASes the IP packets will transit to attain their destination. This information allows our ISP to reconstruct the AS-level topology for outgoing taffic by relying on these BGP routing tables.

To give an example of the AS-level topology our stub AS could obtain through the BGP routes for three different destination ASes, let us have a look at figure 1. This figure shows an AS with three providers. Each provider advertises one route towards all the three destinations that we

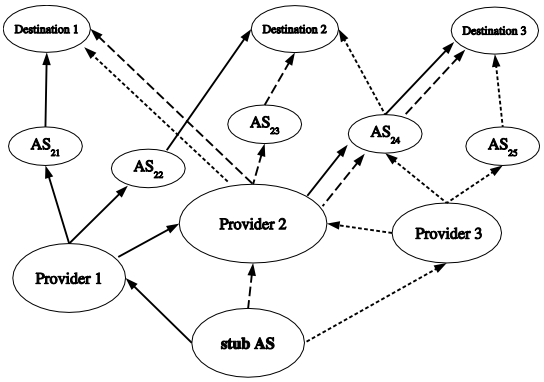

Figure 1. Example AS-level topology.

consider. In this figure, an arrow from AS X to AS Y indicates that the pattern "X Y" appears in the AS path for some destination. In addition, we use three different line styles to identify each provider's routes, so that if part of the AS path of some routes that our ISP received through several providers are identical, then we shall have as many arrows as there are routes from these different providers. On figure 1, the links between two ASes learned through routes advertised by

provider 1 are represented by continuous arrows, while the ones from provider 2 are represented with medium dashed arrows and those from provider 3 with fine dashed arrows.

With the BGP routes received from each provider, our stub knows through which providers it can reach a particular destination. For the example provided on figure 1, all three destinations can be reached through any of the three providers. Assume that each of these three destinations represent one third of the outgoing traffic for our ISP. Our ISP could choose to use one different provider for each destination so that its outgoing traffic is well balanced.

Upon receiving a BGP route, an AS may modify the value of some of the BGP attributes of this route through BGP filters. It is possible to apply a BGP filter that changes the value of the `local-pref` attribute of some route so that one route for a given destination received through one particular BGP peer be preferred over other routes for the same destination received through other BGP peers.

To achieve a good balance of its outgoing traffic, our ISP can rely on the `local-pref` attribute to prefer some routes over others. Suppose that the restricted BGP decision process with only `local-pref` and the AS path length is considered. If the routes towards the three destinations have a default `local-pref` value of 100, then given the fact that the shortest AS path route would be chosen by BGP to reach a destination, the traffic towards destination 1 will be forwarded through provider 2 (AS path length of 2), the traffic towards destination 2 will be forwarded through providers 2 or 3 (AS path length of 3), while the traffic towards destination 3 through providers 2 or 3 (AS path length of 3). This would mean that the routes for destinations 2 and 3 are non-deterministic for this restricted BGP decision process since there are several "best routes" among which BGP must choose. In practice, the BGP decision process ensures that only one route is used but this would not automatically lead to a good balance of the traffic. So to get back to our problem a solution is to put a higher value of the `local-pref` attribute in one of the routes learned from destinations 2 and 3 to ensure that only one best route remains and that the traffic balance be good among the three providers. For that purpose, our stub can attach a `local-pref` value of 110 to the route towards destination 2 learned from provider 1 and also attach a `local-pref` value of 110 to the route towards destination 3 learned from provider 3. This configuration would ensure that each provider gets one third of the outgoing traffic.

3. Interdomain TE as an Assignment Problem

Interdomain traffic engineering as presented in the previous section can be seen as an assignment problem in combinatorial optimization. This problem consists in assigning the traffic t_i towards each destination i through one and only one provider j in order to minimize the total cost of the assignment $\sum_{j=1}^{P} C_j(\sum_{i=1}^{N} x_{ji}t_i)$, where C_j represents the total cost of sending traffic through provider j and x_{ji} tells through which provider j the traffic towards destination i is sent. The assignment problem can be efficiently solved. However, the problem we are interested at solving and presented in the next section can be seen as an instance of the generalized assignment problem which is known to be NP-hard [6]. This problem is difficult to solve, and particularly in our case where the two objectives, the cost of sending traffic and the number of BGP filters, are noncommensurate.

Note that we consider throughout this paper the two concepts of "provider" and "interdomain link" as equivalent although this does not need to be the case in practice since an AS may have several physical links with one of its providers.

4. The multi-objective combinatorial optimization problem

Up to now, we have dealt with what we called the "simple" problem. The reason why we called this problem "simple" is that it was purely combinatorial. It does not make any mention

to the fact that the number of `local-pref` changes we need to configure on the BGP routers is potentially as large as the number of destinations for which we need to override the choice made by BGP in order to distribute the traffic as we would like it among the providers. Our aim however would be to utilize as few BGP filters as possible in order to achieve the "best" (according to the objective function) distribution of the traffic. This problem is thus a multi-objective combinatorial optimization problem, since we want to optimize the objective function while minimizing the number of BGP filters at the same time. So in fact we are not interesting in the pure combinatorial problem since the optimal solution to the assignment problem could involve too many BGP filters. On the other hand, we do not know the cost of a BGP filter, this objective is not measurable. Hence our two objectives, cost of traffic and number of BGP filters, are *noncommensurate*.

Given that we want to find the optimal distribution of the interdomain traffic for a given number n of BGP filters to be configured, our objective is to find those among the best n filters for the m possible destinations and the p available providers, but starting from the default BGP configuration. So the actual problem we have to solve is not an unrestricted search by trying to swap the destinations over the providers but we need to know how to tweak BGP and find the successive filters that allow to reduce the objective function starting from the default BGP routes.

Now a crucial step is to get a rough idea of the size of the search space. Assuming there are about 1000 ASes with whom the local AS exchanges a significant amount of traffic each day among the 14000 ASes that composed the Internet in late 2002, and given that we have a worst case of the p providers having each one route towards each destination, finding the best 10 BGP filters to apply to the routes would require to search for the best 10 BGP filters among 1000 times p possible ones. For the case $p = 2$ this amounts to $\frac{2000!}{10!1990!}$ possibilities only for the first 10 BGP filters, a number with 32 decimal digits! Since we do not know how many BGP filters will be required to find an acceptable solution to the multi-objective problem, we need to reduce the size of the search space in some way.

In order to reduce this number of filters, we have to aggregate the interdomain destinations into coarser objects. The AS-level topology obtained from the BGP routing tables constitutes a sound basis for that, by aggregating the destinations into "AS sink trees". So let us define some concepts that will be useful in the remainder of this paper. A "terminal AS" (destination) is *an AS that appears as the last AS in the AS path of a BGP route*. A "sub-ASx path" of an AS path is *the suffix of that AS path whose first AS is the first occurrence of ASx*. Note that we cannot remove the "first occurrence" part of the previous definition because of AS path prepending [7], although loops do not occur in the paths of a path vector protocol like BGP, repetitions of the same AS are allowed in that case. The "ASx sink tree" is an instance of an "AS sink tree", defined by *the set of all ASes that appear in all sub-ASx paths in a given BGP table*. A practical, but important reason to utilize these definitions is that all existing BGP routers can be configured to attach `local-pref` values to learned routes whose AS path matches such a regular expression.

Getting back to the problem of the size of the search space, let us evaluate what gain there is at considering AS sink trees instead of interdomain destinations. In the simulations proposed in section 7 with a little less than 1000 terminal ASes, we have about more than 1100 AS sink trees while when taking into account only the AS sink trees for ASes located two AS hops away (which we call "level 2" AS sink trees) from the local ISP, this number reduces to 150 for 2 providers. This means that considering such "level 2" AS sink trees would result in a search space of $\frac{300!}{10!290!}$ points for 2 providers, a number with 21 decimal digits instead of 32. This is still a large search space but the gain is significant. It must be clear that we cannot perform neither an exhaustive search nor a random search in such a large space. An exhaustive search would take too long while a random one would not guarantee that we find a good solution. Biasing the search by trying first the most important destinations in terms of their traffic provides good solutions for a few BGP filters only and does not provide any advantage for a few tens of filters. Using an integer program

to solve this problem is quite difficult, mainly because our problem depends on the behavior of BGP when a filter is applied. An integer program would require that we be able to explicit the behavior of the BGP decision process which is by no means simple. Our algorithm simulates a subset of the real BGP decision process and takes as input real BGP routing tables, hence we do not rely on any theoretical assumption concerning the way BGP works in our algorithm, which we could hardly do with an integer program. So the "evolutionary" approach provides a simple way to search for good BGP filters in a way that is close semantically to the problem we are trying to solve.

5. The evolutionary algorithm

Given that our problem is largely combinatorial, we rely on an evolutionary algorithm that will perform a stochastic search using a population of solutions to find among the BGP filters the most interesting ones. We encode an individual as an integer vector v: each value v_i of the vector v represents the provider to be used to attain a particular destination. An individual is hence equivalent to the output of the BGP decision process. Our algorithm works as follows. We initialize the search by running the BGP decision process once and all individuals are then set according to the results of the BGP decision process. At each iteration and for each individual, we choose randomly (and uniformly) an AS sink tree and compute through which provider we should sent the traffic for all the destinations of this AS sink tree. For each provider, we apply a `local-pref` value higher than the default one to all the routes of this provider for these destinations and run the BGP decision process to know the new distribution of the traffic among the providers. If this change leads to improvement of the objective function we actually change the `local-pref` value of all the routes of this AS sink tree. The subtlety is that we cannot know in advance how the distribution of the traffic among providers will change due to both the BGP decision process and the overlap of the destinations among different AS sink trees. If it happens that some BGP filters have been applied for AS sink trees that have common routes with the AS sink tree we would like to change during the current generation, then the new distribution of the traffic among the providers is not known until we run the BGP decision process with this new BGP filter. If some destinations have the same value of `local-pref` due to different BGP filters, it will be up to the BGP decision process to decide about the final traffic distribution. Among these individuals not all lead to an improvement in the objective function. We do not remove the individuals which did not lead to an improvement during the current iteration because they could allow us to improve a lot during the next iterations and it is up to the selection procedure of the evolutionary algorithm to decide which individual must survive.

6. Traffic demands

To evaluate our algorithm, we use in this paper the data available on the web from Abilene [1]. Abilene is a high-speed backbone network that interconnects universities and research labs in the US. Abilene collects Netflow [2] statistics every day from its POPs and allows these statistics to be queried from the web. For this study, we used one month of data starting from August 22^{nd} to September 21^{st} 2002. We used the whole "source-destination AS" traffic matrix that provides with total bytes counts for each AS. To give an idea of the variability of the traffic demands over this one month period, figure 2 presents the amount of traffic in bytes for the largest destination ASes of Abilene.

We have ordered the destination ASes seen during the one month of the trace (x-axis) and we plotted the amount of bytes (in logarithmic scale) sent towards these ASes (only the 1000 with the largest amount of traffic) for each day of the trace (y-axis). Most ASes received a daily amount of bytes that varied by a factor of 10 or more. This indicates that although ASes that receive a lot of

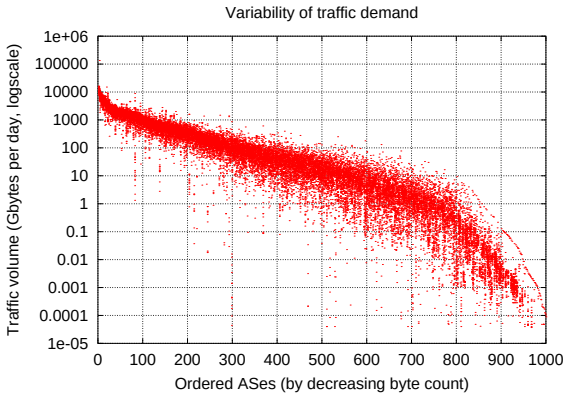

Figure 2. Traffic demands of Abilene destination ASes.

traffic often do so over large time periods, the absolute amount of traffic can vary a lot from day to day. Even if the most important ASes in terms of the total traffic volume seem to have a smaller variability of their daily traffic volume on a logscale plot, the changes in the daily traffic volume for all ASes are significant.

7. Simulations

We insist on the fact that the benefits of interdomain traffic engineering depend on the particular context, and different contexts require different types of solutions. Not all ISPs need to optimize their traffic distribution. The purpose of this section is to study the behavior of our algorithm to get an idea of the impact of some networking related parameters. The parameters we are interested in are the number of providers, the objective function and the AS sink tree aggregation we allow during the search.

From the discussion of section 4, we can see that increasing the number of providers increases dramatically the size of the search space, so this should reduce the performance of the search. The type of objective function can also make the life harder for our algorithm, for instance if the improvements in the objective are a complex non-linear function of the amount of traffic that are switched among providers, then the algorithm will take longer to find a way for improvements. Finally, we would also like to find what the "ideal" AS sink tree aggregation is to perform interdomain traffic engineering. Allowing all possible AS sink trees to be tried (any AS in the AS-level topology) is due to provide the largest flexibility for the search since the algorithm could find the minimal number of BGP filters (i.e. AS sink trees) to obtain a given value of the objective function. The issue is that allowing more AS sink trees means to increase the size of the search space, hence slowing down the search.

In our simulations we use two different objective functions. The first objective function is the maximum amount of the traffic exchanged on any of the interdomain links. This objective function achieves traffic balance for the traffic exchanged with all providers. The second objective function is a total cost defined on the traffic exchanged with all providers, similar to the monetary cost of the interdomain traffic under the assumption that each provider bills the ISP on the traffic volume. Our evolutionary algorithm could be adapted to support other objective functions.

The scenario of our simulations consists of a stub ISP connected to two to four large Tier-1 providers. Measurements in the Internet topology show that many stub ISPs have a two or more providers and that many stub ISPs are directly connected to Tier-1 providers. The traffic demand of

this would-be ISP was gathered on the web from the Abilene Netflow statistics page [1]. We have used an aggregate traffic demand of one month, from August 22^{nd} to September 21^{st} 2002. This traffic demand concerned 977 different destination ASes. We gathered four BGP routing tables from Oregon Route views [4] from four major Tier-1 providers from October 3^{rd} 2002: AT&T, Sprint, Level3 and Verio.

7.1. Traffic balance

Figure 3 presents the results of the optimization with the traffic balance objective with 2 and 4 providers. For the case with two providers, we confront AT&T and Sprint. For three providers, we confront AT&T, Sprint and Level3. The x-axis shows the number of generations required to obtain the smallest value of the objective function. The y-axis gives the value of the objective function *max(traffic(i))*, $i = 1, .., 4$ where *traffic(i)* denotes the amount of traffic exchanged with provider i. The total traffic of the one month traffic demand corresponds to 640.393 units, one unit being one billion of bytes with the 1/1000 packet sampling performed by Netflow. So the unit corresponds roughly to 1 Terabytes of traffic.

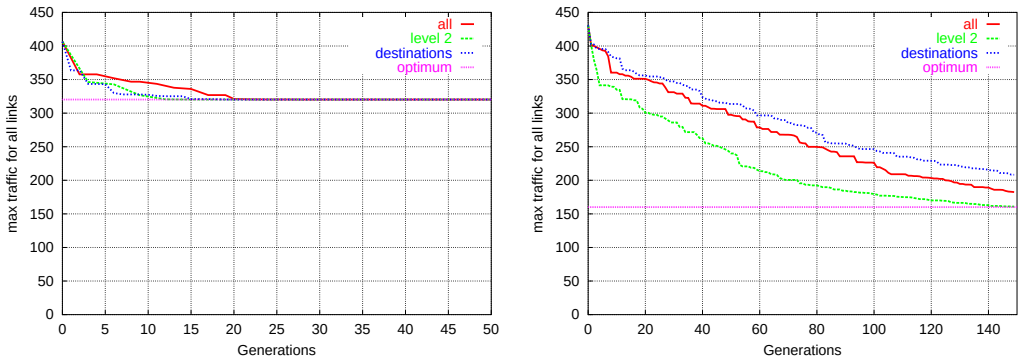

Figure 3. Simulations for traffic balance: 2 providers (left) and 4 providers (right).

Figure 3 illustrates the effect of two parameters: the number of providers and the AS sink tree granularity. First the effect of the number of providers. The solution found by the normal BGP decision process appears for generation 0 (the default BGP solution). As the number of providers increases, BGP does not provide a good traffic balance among all the available providers when the default values of the route attributes are used because the distance between the optimum and the BGP solution does not decrease when the number of providers increases. Increasing the number of providers a stub would have thus does not ensure that the traffic balance among these providers shall be good. Furthermore, the more providers a stub has, the bigger this stub is generally speaking, so its needs for traffic engineering are due to be more critical. The reason for this behavior of BGP is that a large fraction of the routes that are common among Tier-1 providers are non-deterministic [9]. This means that as the number of providers increases, the traffic balance among these providers will depend a lot on the tie-breaking rules of the BGP decision process. Indeed, a closer look at the reason for this inability of BGP to correctly balance the interdomain traffic is the tie-breaking rules in the BGP decision process after the shortest AS path selection rule. Figure 3 shows that a larger number of generations (hence of BGP filters) is required for the algorithm to converge towards the optimum: 20 generations for 2 providers, a little less than 150 generations for 3 providers (not shown) and more than 150 generations for 4 providers. Since the number of generations provides an upper bound on the number of BGP filters, we see that a good

traffic balance for a small number of BGP filters can be obtained with a few BGP filters only in the case of 2 providers.

The other parameter studied in figure 3 is the AS sink tree granularity. Each graph of figure 3 compares three different ways to leverage the aggregation of the traffic destinations with the AS sink trees. The first way consists of allowing all ASes seen in the AS paths for all traffic destinations to be an AS sink tree. This means that we allow AS sink trees to be anywhere in the AS-level topology. This corresponds to the curves labeled "all" on the graphs of figure 3. There are then respectively 1075, 1112, 1118 and 1125 different AS sink trees for the simulations with one, two, three and four providers. The second case corresponds to destination ASes only. This means that one restricts the AS sink trees to the leaves of the AS-level graph only. There are 977 AS sink trees in that case, this curve is labeled "destinations". The last situation we study is a heuristic based on our knowledge of the AS-level topology. The coarsest aggregation for AS sink trees is the four providers themselves, but this is not due to provide a sufficiently fine-grained granularity at the traffic-level to allow optimal load balancing. Advancing one AS hop farther on the other hand still provides a good aggregation of the traffic destinations together with a good granularity of the traffic for these AS sink trees. Because most of the traffic destinations for interdomain traffic are located a few AS hops away from a stub ISP [3,11], aggregating traffic destinations just behind the providers is due to give a good aggregation while still a fine traffic granularity.

Figure 3 confirms this intuition by showing the fastest decrease of the objective function for a given number of BGP filters to be applied. There are respectively 80, 155, 185 and 225 AS sink trees at a distance of two AS hops (curve labeled "level 2"). This search space is therefore also the smallest among the three aggregation levels. All simulations of figure 3 start the search with the traffic distribution obtained by using the default BGP routing, the traffic distribution when no BGP filter is applied to the routes. What we see when comparing the three curves of figure 3 is that the AS sink trees located at 2 AS hops provide the smallest value of the objective function for few BGP filters, confirming our intuition that these AS sink trees constitute a good balance between the number of AS sink trees and their traffic granularity. The important thing to bear in mind when reading this figure is that the main difference between the three curves comes from the size of the search space in which the algorithm works in each case. The disadvantage of the "all" and the "destinations" cases is that the number of possible AS sink trees becomes too large. These two search spaces would need a larger population in order to be able to find good solutions for a few BGP filters, but this would take too much time when the number of filters is larger than 2 or 3. The main issue with the multi-objective problem is that using a heuristic based on the amount of traffic corresponding to each destination provides good results for a very limited number of BGP filters, but once the largest gains have been achieved these heuristic do not work well for a large number of BGP filters no matter the cleverness of the heuristic. The largest gains in the objective function come from tuning the tie-breaking rules of the BGP decision process, to approach the unconstrained optimal value requires influencing many destinations [10].

7.2. Cost of traffic distribution

Now we complexify the task for our algorithm by providing a different cost function for the traffic exchanged through each provider, each traffic unit exchanged through a given provider having a fixed cost. An advantage of an evolutionary algorithm compared to mathematical programming methods, is that we are not restricted by analytical constraints on the objective function. Such a cost function is more complex for our search algorithm because improvements depend not only on the amount of traffic that is switched from some provider to another but also on the relative cost of the traffic that changes from one provider to another. For our simulations, we use volume-based billing (X USD per Y bytes) because it is quite common today in the Internet.

The simulation with two providers (left of figure 4) shows the value of the best solution found

at each generation, with two values for the relative cost of the traffic for the two providers. The cost of each provider is provided on the legend of the graphs. The two values of the relative cost illustrate the important things about the convergence of the algorithm. First, the difference between the cost of the traffic for the two providers must be sufficiently important in order to achieve a non-negligible gain (defined as the percentage in cost reduction compared to the BGP solution) with a small number of BGP filters. For instance, with a cost of 2 for the first provider and 1 for the second provider (denoted by (2,1) on figure 4), the total gain after 150 generations is about 23 % for all AS sink trees while 17.5 % for the "level2" aggregation. For the (4,1) cost, the gains are a little under 40 % for all AS sink trees and a little under 30 % for the "level2" aggregation.

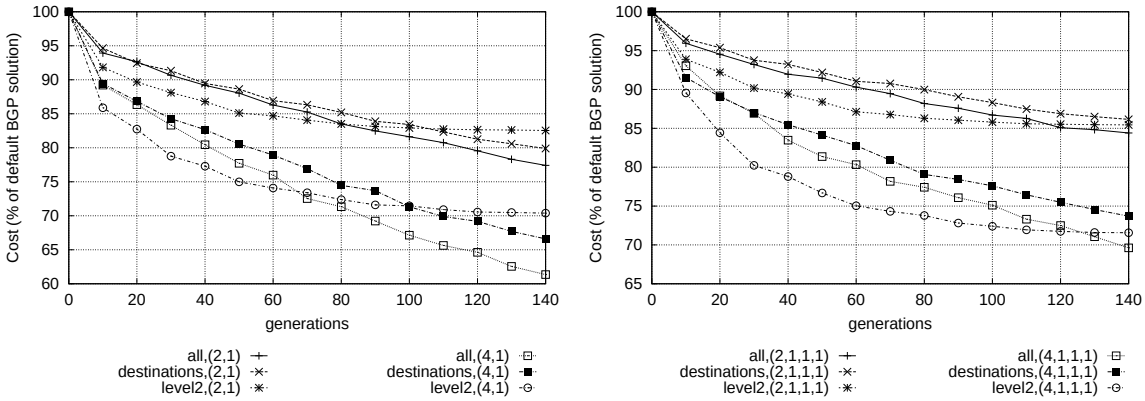

Figure 4. Simulation for cost function: 2 providers (left) and 4 providers (right).

The second thing we can notice concerns the quality of the solutions found by the three AS sink tree aggregations. After a large number of generations, we see a decreasing efficiency of the three aggregation levels in the following order: all, destinations and level2. This phenomenon should be no surprise because it merely reflects the fact that the search space of these three aggregation levels is smaller and smaller, hence reduces the possibility for finding good solutions. This is however true for a large number of BGP filters only, for few BGP filters the "level2" aggregation finds a lower cost compared to the other two aggregation levels since the limited number of "level2" AS sink trees allows the algorithm to spend more time on trying to improve the cost in its smaller search space. We see that for up to 60 generations the "level2" aggregation is systematically better than the other two aggregations while for more than 100 generations the aggregation allowing all AS sink trees finds the smallest cost in their larger search space that allows to leverage both good topological aggregation and a fine traffic granularity.

The simulations with four providers give gains similar to the two providers case. As the simulations with four provider show, when one provider has a cost significantly larger than all others, the algorithm can find improvement as in the two providers case. When the cost setting is more complex, moving the traffic on a "level2" AS sink tree has a non-trivial impact on the improvement in the cost function because several destinations are involved. When one provider has a larger cost than all others, switching a "level2" AS sink tree towards a smaller cost provider is certain to reduce the total cost of the operation since some traffic goes from the expensive provider to less costly ones while the other destinations whose traffic change between equal cost providers does not increase the cost. When the cost setting is more complex however, moving some traffic to a less costly provider may also move some other traffic from a still less costly provider to a more

costly one, making the net result of the change difficult to predict.

8. Conclusion

This paper proposes a solution to interdomain traffic engineering for outgoing traffic. Our method takes as input the previous traffic demand and the corresponding BGP routing tables and could be used on existing routers. We present an evolutionary algorithm that tries to minimize an objective function by finding the successive BGP filters to be applied on the BGP routes. We evaluate this algorithm on two objective functions and a real interdomain traffic demand and real BGP routing tables. We study the behavior of our algorithm for three parameters: the number of providers, the AS sink tree aggregation for the interdomain traffic and the type of objective function.

Our simulations show that the default way BGP performs traffic balancing between providers is poor, and it does not improve as the number of providers increases due to the tie-breaking in the BGP decision process. For more general objective functions, our simulations show that the gain of the optimization depends heavily on the particular objective function used. However, we show that reducing the cost of the interdomain traffic by a significant percentage can be achieved with a limited number of BGP filters.

Acknowledgements

We are highly indebted to all the people that allowed the Abilene Netflow data to be available on-line, and especially Mark Fullmer for his help. We are also grateful for the BGP routing tables available on Oregon Route-views. This work was partially supported by the European Commission within the IST ATRIUM project.

REFERENCES

1. Abilene NetFlow Statistics. Available at `http://www.itec.oar.net/abilene-netflow/`.
2. Cisco. NetFlow services and applications. White paper, available from `http://www.cisco.com/warp/public/732/netflow`, 1999.
3. W. Fang and L. Peterson. Inter-AS traffic patterns and their implications. In *IEEE Global Internet Symposium*, December 1999.
4. D. Meyer. University of Oregon Route Views Project. Available at `http://antc.uoregon.edu/route-views/`.
5. B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, and O. Bonaventure. Interdomain traffic engineering with BGP. *IEEE Communications Magazine, Internet Technology Series*, May 2003.
6. S. Sahni and T. Gonzalez. P-complete approximation problems. *J. ACM*, 23:555–565, 1976.
7. J. Stewart. *BGP4 : interdomain routing in the Internet*. Addison Wesley, 1999.
8. L. Subramanian, S. Agarwal, J. Rexford, and R. Katz. Characterizing the internet hierarchy from multiple vantage points. In *Proc. of INFOCOM'02*, June 2002.
9. L. Swinnen, S. Tandel, S. Uhlig, B. Quoitin, and O. Bonaventure. An Evaluation of BGP-based Traffic Engineering Techniques. Technical Report Infonet-2002-10. Available at `http://www.infonet.fundp.ac.be/doc/tr`, December 2002.
10. S. Uhlig. Interdomain Traffic Engineering: an Evolutionary Approach. Université Catholique de Louvain, Research Report RR 2002-10 INFO, December 2002.
11. S. Uhlig and O. Bonaventure. Implications of interdomain traffic characteristics on traffic engineering. *European Transactions on Telecommunications*, January 2002.