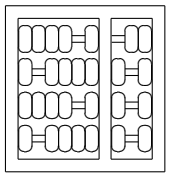

INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Optimal Mapping of Virtual Networks

G. P. Alkmim *D. M. Batista*
N. L. S. da Fonseca

Technical Report - IC-13-28 - Relatório Técnico

October - 2013 - Outubro

Optimal Mapping of Virtual Networks

Gustavo P. Alkmim * Daniel M. Batista † Nelson L. S. da Fonseca‡

10/29/2013

Abstract

Network virtualization is a promising technique for building the Internet of the future since it enables the low cost introduction of new features into network elements. An open issue in virtualization is how to search for an efficient mapping of virtual network elements onto those of the existing physical network, also called the substrate network. Mapping is an NP-hard problem and existing solutions ignore various real network characteristics in order to solve problem in a reasonable time frame. This paper introduces two new algorithms for the solution of the mapping problem, both based on 0–1 integer programming, for the solution of the mapping problem which consider a whole new set of network parameters not taken into account by previous proposals. Simulation experiments confirm the efficiency of the proposed algorithms.

1 Introduction

The minimalism approach and the independence of the technology of specific network has enabled the global spread of the Internet. The core of this network was designed to be simple, with the TCP/IP stack operational over different types of technologies. However, as a consequence of this simplicity, various attempts have been made to provide missing features in its original design. However, the impossibility of including new features in the core of the Internet has prevented the development of many possible applications and services, this has often been labeled the "ossification of the Internet".

To overcome these limitations, various new architectures and mechanisms have been proposed to promote the evolution of the Internet [2] [3] [4] [5] [6]. Several of these are based on network virtualization which allows the definition of virtual networks composed of

*Institute of Computing, University of Campinas, Campinas, SP, Brazil, 13089-971.

†Institute of Mathematics and Statistics, University of São Paulo, São Paulo, Brazil

‡Institute of Computing, University of Campinas, Campinas, SP, Brazil, 13089-971. This research was partially sponsored by Fapesp, process number 2008/07753-6, and CNPq.

This paper was published in G. Alkmim, D. Batista, and N. da Fonseca, "Optimal mapping of virtual networks,"in 2011 IEEE Global Telecommunications Conference (GLOBECOM 2011), 2011, pp.1–6. [1]

virtual routers and links; these are then hosted by routers and links of the real network called "substrate network". Network virtualization permits the coexistence of different protocol stacks and architectures on the same substrate, without the need to modify the actual physical network. Moreover, it imposes no restrictions on these protocols and architectures.

One of the main issues in network virtualization is the efficient mapping of virtual networks onto the substrate network [4] [5]. This mapping determines the allocation of routers and links of a virtual network onto the routers and links of the substrate network. However, the search for the optimal mapping of virtual networks is an NP-hard problem [6].

Various solutions have been proposed for the problem of mapping virtual networks [5] [4] [7] [8]. However, most of them assume restrictions to make the problem tractable, such as the consideration that requests for virtual network establishment be previously known [7] [8] or that the substrate capacity be infinite [8] [9] and network topology restricted [7].

This paper proposes solution to the mapping problem which helps to overcome such limitations by imposing fewer restrictions than previous proposals. The proposed solution considers more realistic scenarios and, hence, handles a large number of parameters that impacts the complexity of the solution. However, as will be shown, the proposed algorithms are efficient and can provide solutions in a reasonable time frame.

The proposed algorithms also consider the presence of repositories of images which contain software and protocols required by the virtual networks. These software are used to instantiate the virtual routers on real routers. Since all images must be transferred from the repository to the real router previous to the operation of the virtual network, an adequate mapping algorithm needs to select the image and the path the image transfer should take.

The two mapping algorithms proposed here are based on a 0–1 integer linear programming (ILP). The objective of this formulation is the minimization of the total amount of real bandwidth allocated to each virtual network. By doing so the resources available for the incoming requests are maximized. One of the algorithms provides exact solutions, while the other is a heuristic designed to decrease the execution time.

Results confirm that execution time of the proposed algorithms is acceptable for various scenarios involving different virtual networks requirements. The algorithms also produced a reasonable probability of blocking requests for virtual network establishment. Moreover, the cost of the mapping provided by the heuristic algorithm is usually very close to that provided by the exact algorithm.

The paper is organized as follows: Section 2 summarizes related work. Section 3 introduces the two proposed algorithms. Section 4 presents the performance evaluation of the algorithms and Section 5 presents the conclusions and suggestions for future work.

2 Related Work

In most of the existing proposals [5] [4] [6], the resources considered are bandwidth and routers processing capacity. Some papers [10] suggested the inclusion of other characteristics as a topic for future work; nonetheless, no solution has so far been proposed. Our proposal has been able to makes several realistic assumptions on resource availability, for example,

the available memory, the number of processing elements of routers, and the time required to instantiate a virtual router.

The network in [5] supports path splitting and path migration, thus allowing the solution of a problem in polynomial time. In path splitting, a single virtual link can be mapped onto more than one physical path of the substrate. Path migration allows a virtual link to be remapped offline to adapt a solution in the face of changes in resource availability. Although the algorithm in [5] runs in a very short time, as do the other existing proposals, it does not consider many of the realistic parameters involved.

In several studies [5] [6], the mapping of virtual links is separated from the mapping of virtual routers and two algorithms were proposed in [4] to integrate the two steps called Deterministic Embedding VN (D-Vine) and Randomized Embedding VN (R-Vine). In the algorithms presented in this paper, routers and links can be mapped simultaneously.

In [6], a distributed algorithm to map virtual networks was introduced to balance the load among all routers in the substrate. Experiments show that this distributed algorithm can generate a large number of control messages but such messages can cause long delay and overhead for network operation. The algorithms presented here do not overload the network with such messages.

By comparing the characteristics of the algorithms proposed in this paper with characteristics of existing algorithms summarized in this section, it is possible to note that the proposed algorithms represent an important contribution to the topic of mapping virtual networks onto substrate networks, due to its realistic assessment of real networks.

3 Proposed Algorithms

Two algorithms based on a 0–1 integer programming have been proposed here to map virtual networks onto substrate networks. These algorithms consider various parameters, as well as the existence of images stored in repositories on the substrate network. The following notation is used for the formulation of the problem:

- $N \in \mathbb{Z}$ is the set of physical routers;
- $F \in \mathbb{Z}$ is the set of physical links, with the physical link (n_1, n_2) connecting two physical routers n_1 and $n_2 \in N$;
- $M \in \mathbb{Z}$ is the set of virtual routers;
- $V \in \mathbb{Z}$ is the set of virtual links with the virtual link (m_1, m_2) connecting two virtual routers m_1 and $m_2 \in M$;
- $I \in \mathbb{Z}$ is the set of images stored in the repository. Each image corresponds to a file with an operating system and a specific set of software ready to be instantiated in a physical router;
- $A \in \mathbb{Z}$: the number of available cores in the physical routers;
- $A(n)$, $n \in N$, gives the number of cores of router n;

- $P \in \mathbb{Z}$ is the set of the number of cores requested by the virtual routers;
- $P(m)$, $m \in M$, gives the number of cores required by the virtual router m to be instantiated;
- $C \in \mathbb{R}$ is the set of values of the available bandwidth in the physical links;
- $C(f)$, $f \in F$, gives the available bandwidth in the link f;
- $Q \in \mathbb{R}$ is the set of bandwidth values requested by the virtual links;
- $Q(v)$, $v \in V$, gives the bandwidth required by the virtual link v;
- $D \in \mathbb{R}$ is the set of values of delay in the physical links;
- $D(f)$, $f \in F$, gives the delay in link f;
- $K \in \mathbb{R}$ is the set of values of maximum delay allowed on a virtual link;
- $K(v)$, $v \in V$, represents the maximum delay allowed on the virtual link v;
- $L_{n,m} \in \{0, 1\}$: binary values that gives location restrictions. If the virtual router m can be mapped onto the physical router n, the value of the variable is 1. Otherwise, it assumes 0. This variable is useful for imposing policy restrictions;
- $R_{n,i} \in \{0, 1\}$: binary values that provides details about the location where images can be stored. If the image i is located in a repository with a direct link dedicated to the physical router n, the value of the variable is 1. Otherwise, it is 0;
- $E_{m,i} \in \{0, 1\}$: binary values related to software restrictions. If the image i contains all the software requirements required by the virtual router m (operating system, protocol stacks, kernel modules and others), the value of the variable is 1. Otherwise, it is 0;
- $B \in \mathbb{R}$ is the set of values that furnish the available memory in physical routers;
- $B(n)$, $n \in N$, represents the memory available in the router n;
- $G \in \mathbb{R}$ is the set of image sizes;
- $G(i)$,$i \in I$, represents the size of the image i;
- $S \in \mathbb{R}$: time threshold for instantiation of the virtual network;
- $T_{n,i} \in \mathbb{R}$: represents the time the physical router n takes to boot the image i;

Consideration of the inputs D, K, $L_{n,m}$, $E_{m,i}$, B, G, S, $T_{n,i}$, I and $R_{n,i}$, all in the same formulation makes this work unique since many of them have never before been taken into consideration. These inputs make the formulation more realistic than that ones previously proposed in the literature. The requests must specify the maximum delay allowed

in the network links (D, K), because this information affects the performance of network applications. The specific image to each virtual router must be defined because different configurations can exist (I, $E_{m,i}$). The content of each repository must be known ($R_{n,i}$) because this affects the decision about from where the image will be copied. The size of the images should be considered because the routers have limited storage capacity (B, G). Also, It is important to consider that clients have particular issues that prevent the utilization of some physical routers ($L_{n,m}$). Moreover, the maximum time acceptable to the instantiation of the virtual network must be considered (S, D, K, $T_{n,i}$).

The solution of the problem is given by the binary variables:

- $X_{n,m,i}$: if the virtual router m is mapped onto the physical router n using the image i this value is 1; otherwise, its value is 0;
- $Y_{u,v,w}$: if the physical path used by the virtual link w includes the physical link (u, v), this value is 1; otherwise, it is 0;
- $Z_{u,v,n,m,i}$: if the physical link (u, v) is used to transfer the image i, needed by the virtual router m, to the physical router n this value is 1; otherwise, it is 0.

The problem is formulated as follows:

$$\text{Minimize } \sum_{u\in N}\sum_{v\in N}\sum_{w\in V} Y_{u,v,w} \times Q(w)$$

subject to

$$\sum_{n\in N}\sum_{i\in I} X_{n,m,i} = 1 \qquad (C1)$$
$$\forall m \in M$$

$$\sum_{m\in M}\sum_{i\in I} P(m) \times X_{n,m,i} \leq A(n) \qquad (C2)$$
$$\forall n \in N$$

$$X_{n,m,i} = 0 \qquad (C3)$$
$$\forall n \in N, \forall m \in M, \forall i \in I | L_{n,m} = 0 \text{ or } E_{m,i} = 0$$

$$\sum_{w'\in V} Y_{u,v,w'} \times Q(w') \leq C(w) \qquad (C4)$$
$$\forall w = (u, v) \in F$$

$$\sum_{u\in N}\sum_{v\in N} Y_{u,v,w} \times D(u, v) \leq K(w) \qquad (C5)$$
$$\forall w \in V, (u, v) \in F$$

$$\sum_{m \in M} \sum_{i \in I} X_{n,m,i} \times G(i) \leq B(n) \qquad (C6)$$
$$\forall n \in N$$

$$\sum_{u \in N} \sum_{v \in N} \sum_{n \in N} \sum_{i \in I} Z_{u,v,n,m,i} \times D(u,v) + \qquad (C7)$$
$$\sum_{u \in N} \sum_{v \in N} \sum_{n \in N} \sum_{i \in I} \frac{Z_{u,v,n,m,i} \times G(i)}{C(u,v)} +$$
$$\sum_{n \in N} \sum_{i \in I} X_{n,m,i} \times T_{n,i} \leq S$$
$$\forall m \in M, (u,v) \in F$$

$$Z_{u,v,n,m,i} \leq X_{n,m,i} \qquad (C8)$$
$$\forall u, \forall v, \forall n \in N, \forall m \in M, \forall i \in I$$

$$Y_{u,v,w}, Z_{u,v,n,m,i} = 0 \qquad (C9)$$
$$\forall u, \forall v, \forall n \in N, \forall w \in V, \forall m \in M, \forall i \in I | (u,v) \notin F$$

$$\sum_{f \in N} Y_{n,f,w} - \sum_{f \in N} Y_{f,n,w} = \qquad (C10)$$
$$\sum_{i \in I} X_{n,a,i} - \sum_{i \in I} X_{n,b,i}$$
$$\forall w = (a,b) \in V, \forall n \in N$$

$$\sum_{j \in N} Z_{u,j,n,m,i} - \sum_{j \in N} Z_{j,u,n,m,i} = \qquad (C11)$$
$$X_{n,m,i} \times R_{u,i} - X_{n,m,i} \times (1 - \lceil \frac{|u-n|}{\alpha} \rceil)$$
$$\forall m \in M, \forall i \in I, \forall n, u \in N, \alpha = |N|$$

$$X_{n,m,i}, Y_{u,v,w}, Z_{u,v,n,m,i} \in \{0,1\} \qquad (C12)$$
$$\forall u, \forall v, \forall n \in N, \forall m \in M, \forall w \in V, \forall i \in I$$

This objective function minimizes the bandwidth allocated by requests for the establishment of virtual network. By doing that, the formulation maximizes the bandwidth available for future requests.

The constraint ($C1$) establishes that each virtual router is allocated to a single physical router and that a single image is used to instantiate it. The constraint ($C8$) prevents an image from being transferred to a physical router if it will not be used by it. The constraint ($C10$) ensures that the set of physical links mapped by a virtual link is a valid path. The constraint compares the in-degree and the out-degree of each physical router n, calculated

respectively by $\sum_{f \in N} Y_{f,n,w}$ and $\sum_{f \in N} Y_{n,f,w}$. The constraints $(C11)$ establishes that the set of physical links allocated for the transfer of an image consist of a valid path in the substrate network.

The constraints $(C2)$ and $(C6)$ express limitations of the physical routers, with $(C2)$ ensuring that each physical router has, at least, as many cores as the sum of the number of cores requested by all virtual routers allocated to it and $(C6)$ establishing that the amount of memory available for each physical router is sufficient to store the images of all the virtual routers allocated to it.

The constraint $(C3)$ guarantees that the virtual routers will only be instantiated using images that meet all the software requirements as well as any geographic location defined by the client requesting the virtual network.

The constraints $(C4)$ and $(C5)$ express the limitations of the physical links. The constraint $(C4)$ ensures that the bandwidth available for each physical link is sufficient to satisfy the bandwidth requirements for all virtual links using it. The constraint $(C5)$ establishes that the total delay in the physical path allocated to a virtual link does not exceed the delay threshold allowed for that virtual link.

The constraint in $(C7)$ ensures that the time to instantiate the virtual network does not exceed the time threshold established by the client. The time needed to instantiate each virtual router is the sum of the times required to transfer the image and to boot the operating system of the image assuming that two or more images can be transferred simultaneously in the same physical link.

The solution is found by using a search tree that is traversed by the branch and cut method. The two algorithms proposed in this paper differ by the node where the search is finished.The algorithm that traverses all the tree is an optimal algorithm and it is called here "optimal algorithm". The algorithm that terminates the search at the root of the tree is called "root heuristic algorithm".

4 Performance Evaluation

Numerical examples presented in this section compare the performance of the two algorithms which were evaluated in dynamic scenarios, where requests arrive during a certain time interval and the availability of resources in the substrate network varies with time. The results in the dynamic scenario demonstrate the impact of the differences between the two proposed algorithms when requests arrive dynamically. Simulations were performed to evaluate the run time of the algorithms, the amount of bandwidth allocated to the virtual networks requested, and the number of requests blocked. Comparisons with existing algorithms [4] [11] were not performed, since these do not consider all of the parameters considered by the algorithms presented here.

All codes were executed on a computer running the operating system Debian GNU/Linux Squeeze. The computer was equipped with two Intel Xeon 2.27GHz processors, with 8 cores each one, and 6GB of RAM. Both algorithms were implemented in C++. The 0–1 integer programming was implemented using the CPLEX optimization library version 12.0.

4.1 Configuration of the Experiments

The configuration of the scenarios simulated took the following into consideration:

- Number of routers in the substrate network: 5, 7, 10, 12, 15, 17, 20, 22 and 25. Using this variation, it is possible to evaluate the performance of the algorithms as a function of the number of physical routers;
- Number of routers with image servers attached to it: 20% of the number of routers of the substrate network. This value was experimentally found by the authors to avoid a large number of infeasible allocations;
- Number of cores available in the physical routers: 6, which is the actual number found in real routers [12];
- Available bandwidth in real links; this number was determined from a uniform distribution between 1Gbps and 10Gbps, which is the interval common in substrate networks [13];
- Available memory in the physical routers: 256MB; this number was based on the actual amount of flash memory in existing real routers [14];
- Size of images: 128MB. This value was based on the recommended amount of flash memory for the software defined in [15], an operating system for routers;
- Time needed to boot an image in a physical router: 10 seconds;
- Time threshold to instantiate each virtual network: 100 seconds;
- Type of request: Type 1, Type 2 and Type 3. This depends on the amount of resources required. Table 1 describes the requirements for each Type of virtual network.

Table 1: Types of virtual networks.

Type	**# of virtual routers**	**# of cores**	**Bandwidth (uniform distribution)**
1	5	2	100Mbps–200Mbps
2	8	3	200Mbps–300Mbps
3	10	6	300Mbps–400Mbps

Both the topology of the substrate networks and that of the virtual networks were randomly generated by using the topology generator `BRITE` [16], with the BA-2 algorithm. For the substrate network, the link delays were the values given by `BRITE`. Since the requested delay of the links of the virtual networks must be greater than the delay in the links of the substrate network, these were defined by multiplying the value returned by `BRITE` by a random number derived from a uniform distribution. The delay of the virtual network was calculated as the value given by `BRITE` multiplied by 15, 10 and 5, respectively for Types 1, 2 and 3.

4.2 Results

Each scenario simulation involves 10000 seconds of simulation time. The arrival time and the duration of the requests were defined randomly following an exponential distribution with means of 50 and 1000 seconds, respectively. The number of physical routers was 20. Results for other sizes of the substrate network were derived, but they are not shown due to space limitations; the results for this substrate network of 20 routers were representative of the overall results. All virtual network requests involved 4 virtual routers.

Results for requests of Type 3 are omitted due to space limitation; comments for this type of request is very similar to that made for requests of Type 2.

Figures 1 and 2 present the execution time of the algorithms for the three types of requests as a function of time. The execution time decreases throughout the simulation. This happens because the resources of the substrate network became occupied and, as a consequence, various points of the search space results in infeasible mapping, and thus diminishing the search time. The decrease in the execution time is smoother for the Type 1 requests (Figure 1), these virtual networks require fewer resources than do the demands of other types of requests. As a consequence, valid mappings for a higher number of requests can be found. The root heuristic algorithm runs faster than the optimal algorithm, and this is specially evident for the Type 1 requests. The execution time of the optimal algorithm decreases as the simulation progresses since the number of points in the search space that yield infeasible solutions increases, thus reducing the search space. The maximum run time reductions when using the root heuristic algorithm were 95.40%, 96.81% and 73.80%, of the run time of the optimal algorithm for Types 1, 2 and 3, respectively.

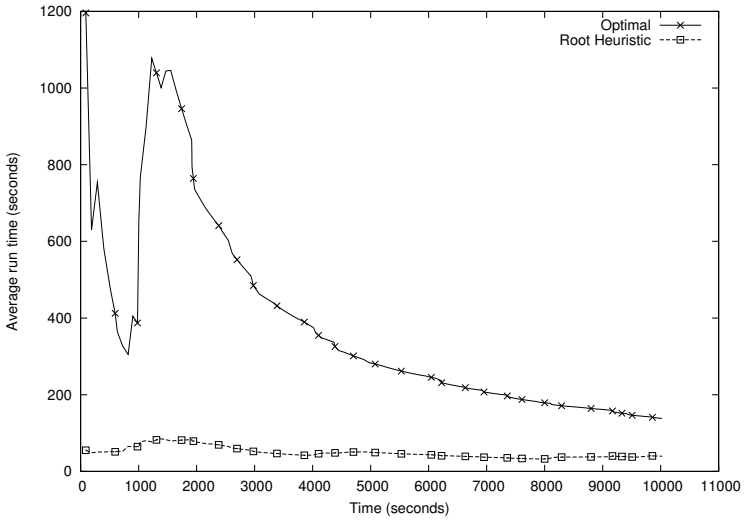

Figure 1: Run time - Type 1 - Dynamic Scenarios.

Figures 3 and 4 and Table 2 show the bandwidth allocated by the algorithms. For Type 1 requests, the root heuristic algorithm allocated at most 17.60% more than the optimal algorithm. For more complex requests (Types 2 and 3), the two algorithms allocated the same amount of bandwidth, with this bandwidth decreasing as the saturation of the

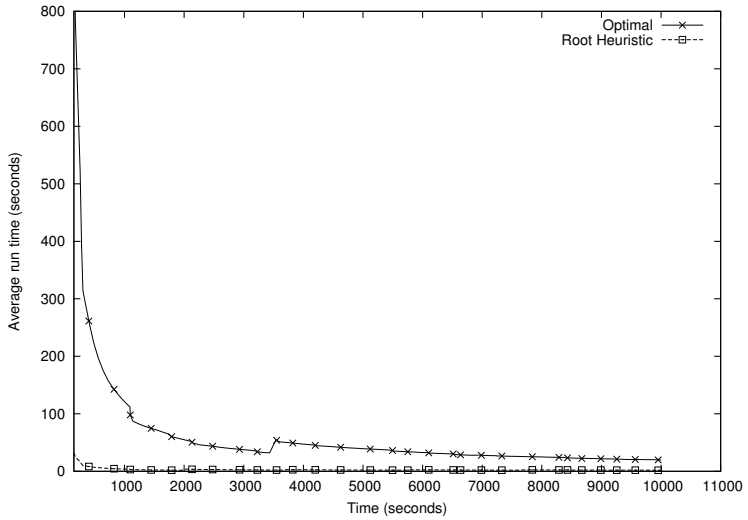

Figure 2: Run time - Type 2 - Dynamic Scenarios.

substrate network increases.

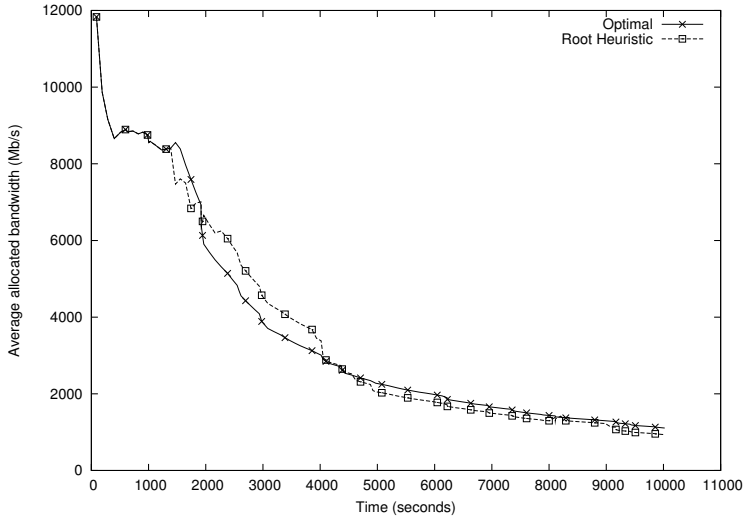

Figure 3: Average allocated bandwidth - Type 1 - Dynamic Scenarios.

Figures 5 and 6 and Table 2 present the blocking ratio of requests. As the substrate network elements are allocated, resource availability decreases, leading to an increase in probability of blocking. Requests of Type 2 saturate the substrate earlier than do requests of Type 1. It can be seen that the blocking rates for the two algorithms are very similar for the scenarios with Type 2 (and 3) virtual networks (Figures 6). The main difference in blocking ratio of the two algorithms is found with Type 1 virtual networks (Figure 5), for which the root heuristic algorithm presents a blocking rate 41.53% lower than that of the optimal algorithm. Since for Type 1 request the execution time of the root heuristic

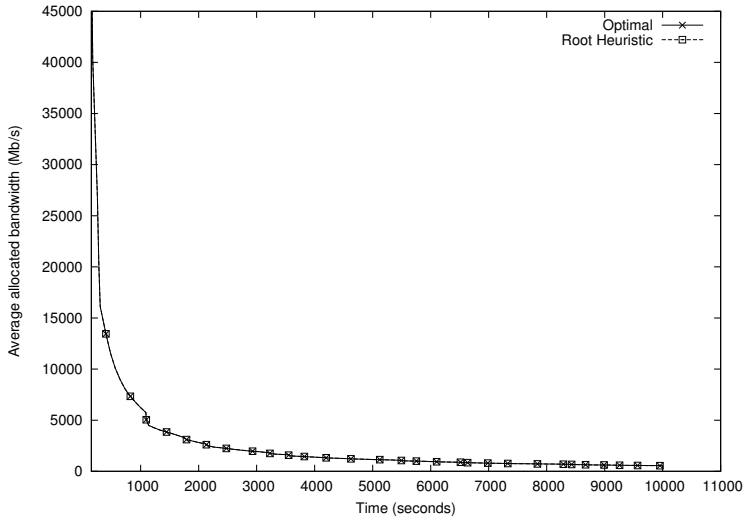

Figure 4: Average allocated bandwidth - Type 2 - Dynamic Scenarios.

algorithm is quite slower than that of the optimal one, requests are processed faster; virtual networks are allocated earlier than when the optimal algorithm is used and, consequently, resources are released earlier so they are available for new requests. These dynamics imply that less blocking occurs with the use of the root heuristic algorithm is used. The maximum differences between the blocking ratio given by the two algorithms for Type 1, 2 and 3 are, respectively, 100%, 3.57% and 3.44% of the blocking ratio given by the optimal algorithm.

Since the algorithms demand different execution times and requests are processed one by one, we have plotted the mean waiting time for a request to be processed. It can be observed that processing requests takes much longer to be processed when the optimal algorithm is used. For the Type 1 requests, the average waiting time reduction was 98.95%. The cumulative effect of the small differences in execution time for requests of Type 2 (and 3) leads to very large differences in the average waiting time for requests. The root heuristic algorithm reduces at most 93.78% of the waiting time given by the optimal algorithm for Type 3 and 100% for Types 1 and 2.

5 Conclusions and Future work

This paper introduced two novel algorithms based on 0-1 linear programming: an optimal one and a heuristic. These algorithms differ from previous proposals by the accountability of a large number of characteristics existing in real systems. It was shown through numerical examples that the heuristics provides solutions similar to those of the optimal algorithm for several metrics but demands lower computational effort which can have a significant impact on the blocking probability as well as on the waiting time to be processed a request is subject to. The algorithms were evaluated for different number of routers in the substrate network and similar results were found. They are not shown in this paper due to space limitation. A comparison between the performance of the two algorithms were also conducted in scenarios

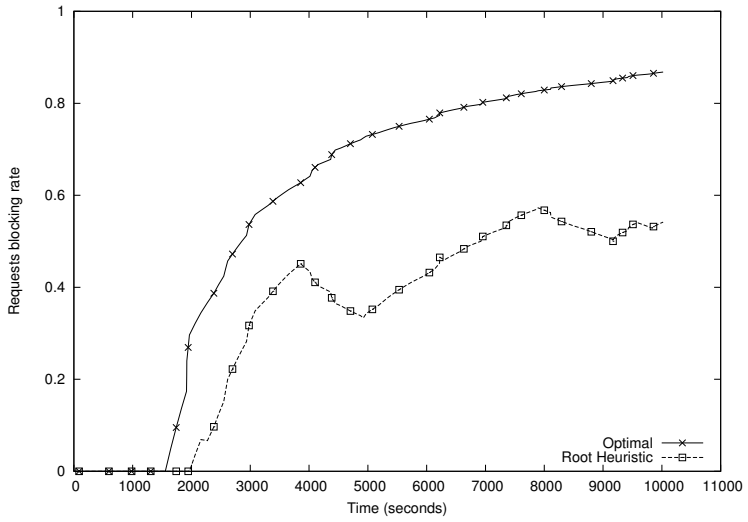

Figure 5: Average blocking rate - Type 1 - Dynamic Scenarios.

with substrate networks with unallocated resources. The heuristic algorithm also showed to be a good trade-off between quality of solution and computational complexity. Again these results are omitted for the same reason.

Currently, faster approximation based on randomized rounding as well as on iterative randomized rounding technique are under evaluation. Solutions based on path splitting is also under development.

References

[1] G. Alkmim, D. Batista, and N. da Fonseca, "Optimal mapping of virtual networks," in *2011 IEEE Global Telecommunications Conference (GLOBECOM 2011)*, 2011, pp. 1–6.

[2] N. Feamster, L. Gao, and J. Rexford, "How to Lease the Internet in Your Spare Time," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 1, pp. 61–64, 2007.

[3] D. Trossen, "Invigorating the Future Internet Debate," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 5, pp. 44–51, 2009.

[4] N. Chowdhury, M. Rahman, and R. Boutaba, "Virtual Network Embedding with Coordinated Node and Link Mapping," in *IEEE INFOCOM*, Abril 2009, pp. 783–791.

[5] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 17–29, 2008.

[6] I. Houidi, W. Louati, and D. Zeghlache, "A Distributed and Autonomic Virtual Network Mapping Framework," in *ICAS '08*, 2008, pp. 241–247.

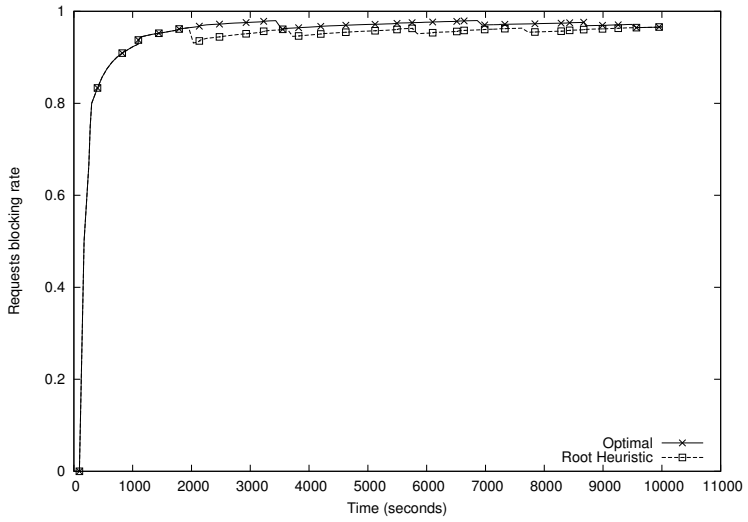

Figure 6: Average blocking rate - Type 2 - Dynamic Scenarios.

[7] J. Lu and J. Turner, "Efficient Mapping of Virtual Networks onto a Shared Substrate," Washington University, Tech. Rep. WUCSE-2006-35, 2006, http://www.arl.wustl.edu/~jst/pubs/wucse2006-35.pdf. Accessed at 12/20/2010.

[8] Y. Zhu and M. Ammar, "Algorithms for Assigning Substrate Network Resources to Virtual Network Components," in *IEEE INFOCOM*, Abril 2006, pp. 1–12.

[9] J. Fan and M. H. Ammar, "Dynamic Topology Configuration in Service Overlay Networks: A Study of Reconfiguration Policies," in *IEE INFOCOM*, Abril 2006, pp. 1–12.

[10] P. Padala, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, and K. Salem, "Adaptive Control of Virtualized Resources in Utility Computing Environments," in *ACM EuroSys '07*, 2007, pp. 289–302.

[11] J. Lischka and H. Karl, "A Virtual Network Mapping Algorithm Based on Subgraph Isomorphism Detection," in *ACM VISA '09*, 2009, pp. 81–88.

[12] C. Systems, "Cisco multiprocessor wan application mode [cisco catalyst 6500 series switches]," 2010. [Online]. Available: http://www.cisco.com/en/US/prod/collateral/modules/ps5510/product_data_sheet0900aecd800f8965_ps708_Products_Data_Sheet.html

[13] RNP, "RNP Backbone map ," 2010, http://www.rnp.br/en/backbone/index.php. Accessed at 12/20/2010.

[14] Cisco Systems, "Cisco 7200 Series Routers Overview [Cisco 7200 Series Routers]," 2010, http://www.cisco.com/en/US/prod/collateral/routers/ps341/product_data_sheet09186a008008872b.html. Accessed at 12/20/2010.

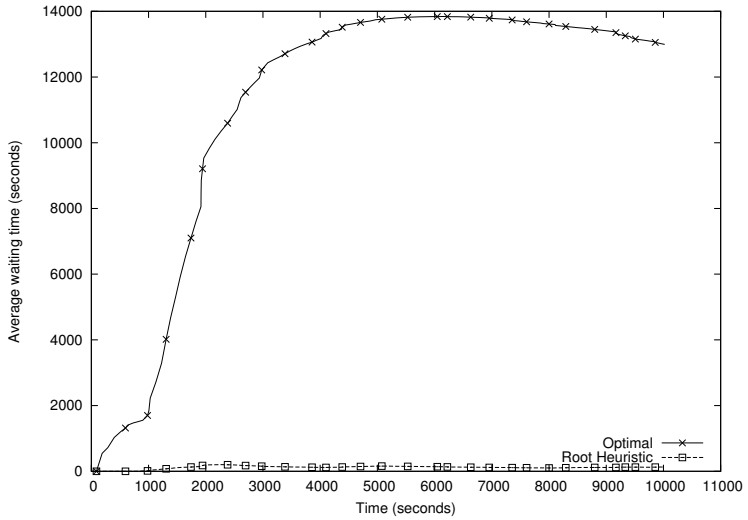

Figure 7: Average waiting time - Type 1 - Dynamic Scenarios.

[15] Cisco Systems, "Download Software," 2010, http://www.cisco.com/cisco/software/release.html?mdfid=278807391&flowid=956&softwareid=280805680&release=12.4.2\discretionary{-}{}{}XB11&rellifecycle=GD&relind=AVAILABLE&reltype=latest. Accessed at 12/20/2010.

[16] A. Medina, A. Lakhina, I. Matta, and J. Byers, "Brite," 2010, http://www.cs.bu.edu/brite/. Accessed at 12/20/2010.

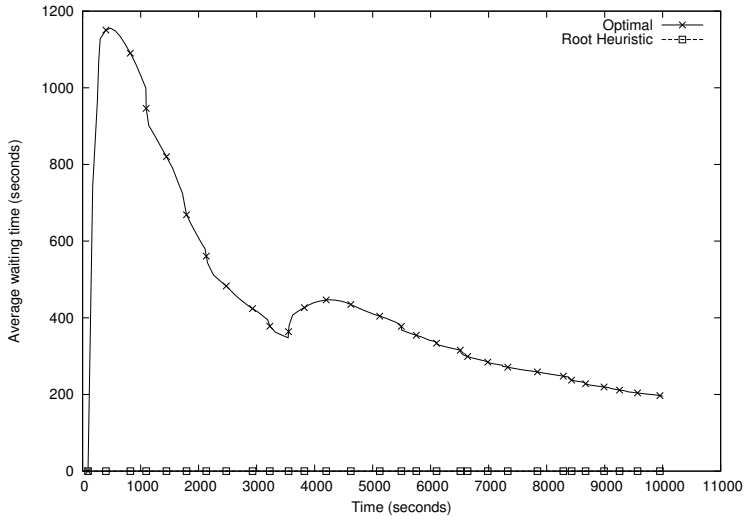

Figure 8: Average waiting time - Type 2 - Dynamic Scenarios.

Table 2: Summary – Dynamic Scenarios.

Optimal Algorithm

Type	**Average run time (s)**	**Average allocated bandwidth (Mbps)**	**Average blocking rate**	**Average service delay (s)**
1	369.23	3390.58	60.38%	11471.87
2	68.45	3057.55	95.06%	439.54
3	33.61	5283.17	91.54%	171.47

Root Heuristic Algorithm

Type	**Average run time (s)**	**Average allocated bandwidth (Mbps)**	**Average blocking rate**	**Average service delay (s)**
1	48.62	3418.49	35.30%	120.68
2	3.22	3057.55	93.80%	0
3	15.34	5283.17	90.78%	28.95