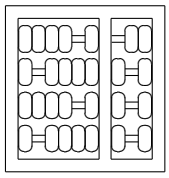

INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Approximated Algorithms for Mapping Virtual Networks on Network Substrates

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

Technical Report - IC-13-29 - Relatório Técnico

October - 2013 - Outubro

Approximated Algorithms for Mapping Virtual Networks on Network Substrates

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 introduction of new features into network elements at low cost. An open issue in virtualization is how to search for an efficient mapping of virtual network elements onto those of the existing physical network. Mapping is an NP-hard problem and existing solutions take long time to find a solution. This paper presents four new approximated algorithms based on two integer linear programming formulations that runs fast and, also, consider various real network characteristics, which is neglected by other proposals in the literature.

1 Introduction

The minimalist approach and the independence of the specific network technology at the link layer 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. The impossibility of including new features in the core of the Internet has prevented the development of many applications and services. This has often been called 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]. Several of these are based on network virtualization which allows the definition of virtual networks (VNs) 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, "Approximated algorithms for mapping virtual networks on network substrates," in 2012 IEEE International Conference on Communication (ICC 2012), 2012, pp. 1460–1465. [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 VNs onto the substrate network [3] [4]. This mapping determines the allocation of routers and links of a VN onto the routers and links of the substrate network. However, the search for the optimal mapping of VNs is an NP-hard problem [5].

Various solutions have been proposed for the problem of mapping VNs [4] [3] [6] [7] [8]. However, most of them assume restrictions to make the problem tractable, such as previously know requests [6] [7], infinite substrate capacity [7] and restricted network topology [6]. Moreover, the modeling of the network [4] [3] [5] consider only the available bandwidth and the processing capacity of the routers. Some papers [9] suggested the inclusion of other characteristics as a topic for future work; nonetheless, no solution has been proposed so far. An exception is the algorithms in [8], which considers 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. Some of the previous work [4] [5] also proposes the separation of the mapping of virtual routers from the mapping of virtual links.

However, the algorithm in [8] is based on a linear programming formulation which can take considerable time to return the optimal mapping. In this paper, four new algorithms based on two integer linear programming formulations are proposed with the aim to reduce the computational complexity of the algorithm presented in [8]. These two integer formulation differ from that in [8] since they correspond to a partition of the original formulation that represent a first step in reducing the computational complexity of the problem. Techniques of relaxation of integer linear program (ILP) formulations are then employed in these new algorithms.

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

2 Proposed Algorithms

The algorithms in this paper models requests for virtual network establishment on network substrates that arrive dynamically. Each request specifies the topology of the virtual network, the resource demanded by the virtual network elements, and the QoS requirements which include a bound on the time to instantiate the VNs and location constrains to instantiate the nodes of the VN.

The proposed algorithms are based on two 0-1 Integer Linear Programming (ILP) formulations. The algorithms consider parameters related to realistic environments, such as the existence of router images located in repositories distributed over the network. Before presenting the approximated algorithms, it will be presented the ILP formulations. The implementation of these formulations, without any relaxation techniques, leads to an optimal

solution. However, in some scenarios, the search for the optimal solution can take a long time. The formulation in this paper differ from that in [8] by the introduction of a a two step formulation which reduces memory dependencies. The approximated algorithms are proposed in order to reduce the complexity of the problem without a significant loss in the quality of the solutions. The following notation is used for the formulation of the problem:

N is the set of physical routers

F 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 is the set of virtual routers;

V 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 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 is the set with the number of available cores in the physical routers; $A(n)$, $n \in N$, gives the number of cores of router n;

P 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 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 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 is the set of values of delay in the physical links; $D(f)$, $f \in F$, gives the delay in link f;

K 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;

$L_{n,m}$ are binary values that give restrictions on locations. 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. For example, if the user requesting the virtual router m does not want this router mapped to a physical router located in a certain country, the value of $L_{n,m}$ with physical routers located in the country, must be 0;

$R_{n,i}$ are binary values that provide details about the location where images are 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;

$E_{m,i}$ are 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 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 is the set of image sizes; $G(i)$,$i \in I$, represents the size of the image i;

S is the time threshold for instantiation of the VN;

$T_{n,i}$ represents the time the physical router n takes to boot the image i;

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 on the location the image shall 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 VN 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;
- $Z_{u,v,m}$: if the physical link (u, v) is used to transfer the image requested by the virtual router m, this value is 1; otherwise, it is 0.

2.1 ILP Formulations

All the algorithms proposed in this paper are based on two ILP formulations that must be executed sequentially. The first (ILP-Mapping) searches for a solution of the problem of mapping VNs (routers and links) on the substrate. The second (ILP-Image) searches for routes in the substrate to transfer the images of the virtual routers from the repositories to the nodes in the substrate hosting the virtual nodes. The utilization of two ILPs reduces the execution time to find the solution when compared with a solution that tries to find the values of all variables in just one ILP [8].

The ILP-Mapping is formulated as follows:

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

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

$$\sum_{m\in M}\sum_{i\in I} X_{n,m,i} \leq 1 \qquad (C2)$$
$\forall n \in N$

$$\sum_{m\in M}\sum_{i\in I} P(m) \times X_{n,m,i} \leq A(n) \qquad (C3)$$
$\forall n \in N$

$$X_{n,m,i} = 0 \qquad (C4)$$
$\forall n \in N, \forall m \in M, \forall i \in I | L_{n,m} = 0 \text{ or } E_{m,i} = 0$

$$\sum_{w'\in V} Y_{u,v,w'} \times Q(w') \leq C(w) \qquad (C5)$$
$\forall w = (u, v) \in F$

$$\sum_{u\in N}\sum_{v\in N} Y_{u,v,w} \times D(u, v) \leq K(w) \qquad (C6)$$
$\forall w \in V, (u, v) \in F$

$$\sum_{m\in M}\sum_{i\in I} X_{n,m,i} \times G(i) \leq B(n) \qquad (C7)$$
$\forall n \in N$

$$Y_{u,v,w} = 0 \qquad (C8)$$

$\forall u, \forall v, \forall w \in V | (u, v) \notin F$

$$\sum_{f\in N} Y_{n,f,w} - \sum_{f\in N} Y_{f,n,w} = \qquad (C9)$$
$$\sum_{i\in I} X_{n,a,i} - \sum_{i\in I} X_{n,b,i}$$
$\forall w = (a, b) \in V, \forall n \in N$

$$X_{n,m,i} \in \{0, 1\} \qquad (C10)$$
$\forall n \in N, \forall m \in M, \forall i \in I$

$$Y_{u,v,w} \in \{0, 1\} \qquad (C11)$$
$\forall u, \forall v, \forall w \in V$

The objective function of the ILP-Mapping minimizes the bandwidth allocated by requests for the establishment of VN. 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. Constraint ($C2$) limits the amount of virtual routers that can be allocated on a physical router per request. Only one virtual router can be allocated to a physical router per request. The constraint ($C9$) 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. The constraints (C3) and (C7) express limitation of the physical routers ensuring that in each physical router the number of cores and the amount of memory allocated are not exceeded.

The constraint ($C4$) guarantees that the virtual routers will be instantiated using only images that meet all the software requirements as well as any geographic location defined by the client requesting the VN.

The constraints ($C5$) and ($C6$) 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 of all virtual links using it. The constraint ($C6$) 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. Constraint ($C8$) guarantees that if (u, v) is not a physical link, it will never be used in the mapping.

Constraints ($C10$) and ($C11$) define the domain of the binary variables.

After the solution of the ILP-Mapping is found, the values of $X_{n,m,i}$ e $Y_{u,v,w}$ can be used as input to the second formulation, the ILP-Image.

The ILP-Image is formulated as follows:

$$\text{Minimize} \sum_{m \in M} \sum_{u \in N} \sum_{v \in N | (u,v) \in F} Z_{u,v,m} \times D(u,v)$$
$$+ \frac{Z_{u,v,m} \times G(i | X_{n,m,i} = 1)}{C(u,v)} \text{ subject to}$$

$$\sum_{m \in M} Z_{u,v,m} = 0 \qquad (C12)$$
$$\forall u, \forall v | (u,v) \notin F$$

$$\sum_{j \in N} Z_{u,j,m} - \sum_{j \in N} Z_{j,u,m} = \qquad (C13)$$
$$X_{n,m,i} \times R_{u,i} - X_{n,m,i} \times (1 - \lceil \frac{|u-n|}{\alpha} \rceil)$$
$$\forall m \in M, \forall i \in I, \forall n, u \in N, \alpha = |N|$$

$$Z_{u,v,m} \in \{0, 1\} \qquad (C14)$$
$$\forall u, \forall v \in N, \forall m \in M$$

The objective function of the ILP-Image minimizes the time to instantiate the VN. 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 on the same physical link.

Constraint ($C12$) guarantees that (u, v) will not be used in the mapping in case it is not a physical link in the substrate. The constraint ($C13$) establishes that the set of physical links allocated to the transfer of an image consists of a valid path in the substrate network. Since $|u - n|$ is greater then 0 iff $u \neq n$ and $|N|$ is the total of physical routers, $\lceil \frac{|u-n|}{\alpha} \rceil$) values 1 if $u \neq n$ and 0 otherwise. Constraint ($C14$) defines the domain of the variables.

2.2 Optimal and Root Approximated Algorithms

To evaluate the performance of the approximated algorithms, their solutions are compared with those of the optimal solution of the problem. In this section, two algorithm, not described in details, are introduced for the solution of the ILP formulations presented in the last subsection. The first algorithm implements the ILP formulation presented in the Subsection 2.1 without any change, so it is called the Optimal Algorithm. Preliminary experiments with the Optimal Algorithm show that it can not be used in all scenarios since it takes too much time to find solutions to problems involving networks with more than 100 routers. Because of that, we implemented another algorithm to serve as a reference to the approximated algorithms, called the Root Approximated Algorithm.

Both the Optimal and the Root Approximated algorithms use the Branch and Cut method. In this method, a search tree is transversed to find solutions. The Optimal Algorithm transverse all the nodes of the tree, while the Root Approximated Algorithm can stop the search at the root node of the tree. By stopping the search early, it is expected a reduced execution time when compared to that of the Optimal Algorithm.

2.3 Random and Deterministic Approximated Algorithms

In this subsection, the four new approximated algorithms are presented. These algorithms employ the relaxation of the integer constraints of the two ILP formulations. This consideration tends to reduce the execution time of the Optimal Algorithm. The relaxation modifies the constraints ($C11$), ($C12$) and ($C14$), respectively to the constraints ($C11'$), ($C12'$) and ($C14'$) and it is presented bellow:

$$X_{n,m,i} \in \mathbb{R}_{[0,1]} \quad \forall n \in N, \forall m \in M, \forall i \in I \qquad (C11')$$

$$Y_{u,v,w} \in \mathbb{R}_{[0,1]} \quad \forall u, \forall v \in N, \forall w \in V \qquad (C12')$$

$$Z_{u,v,m} \in \mathbb{R}_{[0,1]} \quad \forall u, \forall v \in N, \forall m \in M \qquad (C14')$$

The new constraints modify the domain of each variable from $\{0, 1\}$ to $\mathbb{R}_{[0,1]}$. There is a needed to execute an algorithm after the search of the solution to approximate the real numbers to integer ones. The four algorithms for mapping VNs differ by the algorithm implemented to approximate the real numbers. Table 1 summarizes the method used in each algorithm to approximate the variables to integer values.

Table 1: Description of the Approximated Algorithms

Algorithm	**Description**
Random	Executes the relaxed ILP and, for each virtual router m, draws the integer value of variable $X_{n,m,i}$ by considering the real values returned by the ILP as probabilities.
Deterministic	Executes the relaxed ILP and, for each virtual router m, sort the variables $X_{n,m,i}$ in descending order. The value of the first X variable in the list is changed to 1 and the others to zero.
Iterative Random	Similar to the Random algorithm, with the difference that, after each virtual router has its allocation defined, the relaxed ILP is re-executed.
Iterative Deterministic	Similar to the Deterministic algorithm, with the difference that, after each virtual router has its allocation defined, the relaxed ILP is re executed.

3 Performance Evaluation

Numerical results presented in this section compare the performance of the four approximate algorithms: Random, Deterministic, Iterative Random and Iterative Deterministic for the two algorithms previously described: the Optimal and the Root Approximated. In the rest of the paper they will be referenced as Rand Approx, It Rand Approx, Det Approx, It Det Approx, Opt, and Root Approx respectively. They were evaluated in dynamic scenarios, in which requests arrive during a certain time interval and the availability of resources in the substrate network varies with time. Simulations were performed to evaluate the execution time of the algorithms, the amount of bandwidth allocated to the VNs requested and the number of requests blocked.

All the algorithms and the simulator were implemented in C++. The linear program formulations were implemented using the CPLEX optimization library version 12.0. All programs were executed on a computer running the operating system Debian GNU/Linux Squeeze. The computer was equipped with two Intel Xeon 2.27GHz processors each with 6 cores capable of running 12 simultaneous threads and 40GB of RAM.

3.1 Configuration of the Experiments

Both the topology of the substrate networks and that of the VNs were generated by using the topology generator `BRITE` [10], with the BA-2 [11] algorithm, a widely used method to generate pseudo-random network topologies similar to the Internet. For the substrate network, the link delays were the values returned by `BRITE`. The requested delay in the links of the VNs was defined as the value returned by `BRITE` multiplied by 15.

Two different set of experiments were conducted in order to evaluate the performance of the algorithms. The first set of experiments evaluates the performance of the algorithms as a function of the number of physical routers. The second set of experiments evaluates the performance as a function of different arrival rates of requests. The last set of experiments evaluate the scalability of the algorithms. The range of the number of physical routers in this last set of experiments is greater than in the first set.

Table 2 presents the parameters used in the first set of experiments. Similar parameters were used in the second set of experiments. The second set of experiments differs in relation to the first set by the number of physical routers, fixed in 60, and the mean time between requests, which varied in the interval [25,300] seconds. It is important to highlight that the

Table 2: Parameters of the first set of experiments

Parameter	**Value**
# of physical routers	[20,200]
# of cores of each physical router	6
Size of the memory of each physical router	768 MB
Bandwidth of each physical link	∼10 Gbps
Delay of each physical link	Defined by the BRITE
# of virtual routers of each request	4
# of cores requested by each virtual router	2
Bandwidth of each virtual link	∼1 Gbps
Delay of each virtual link	(Defined by the BRITE) × 15
Mean time between requests	25 s
Mean "life-time" of requests	3000 s
Maximum time to instantiate the VN	100 s
Simulation time	5000 s
# of different images	3
Size of each image	128 MB
Time taken to instantiate each virtual router	10 s

parameters used in the experiments were based in values of real equipments and network links [12], [13], [14], [15].

3.2 Results

The Opt Algorithm was executed only in scenarios with 100 or less physical routers given the long time they take to produce a solution.

Figure 1 shows the execution time of the algorithms for the first set of experiments. The execution time of the Opt Algorithm as a function of the number of physical routers increases with a exponential rate. Such increase did not happen with the Root Approx. The approximated algorithms had different performance. The Rand Approx and the Det Approx were similar to the Root Approx, i.e., they increase the execution time almost linearly. The It Rand Approx and the It Det Approx had also an almost linear increase but these increase was sharper than the other two approximated algorithms. This shows that the Rand Approx and the Det Approx have execution time similar to the Root Approx. This was expected since they run the two ILPs just once and after that they go to the step to find the integer solutions. The It Rand Approx and the It Det Approx demand longer execution time since they run the ILPs several times. Even fixing some variables in each iteration before the next execution of the ILP, their execution time was long.

Figure 2 shows the execution time of the algorithms for the second set of experiments. In this set, the number of physical routers was set to 60 and the mean arrival time of requests for VNs varied from 25 to 300s. As can be observed, the execution time of the Det. Approx and the Rand Approx algorithms are similar to those of the Root Approx Algorithm, since all these algorithms run the ILPs just once. Several executions of the ILPs affected the performance of the It. Rand Approx and It Det Approx algorithms, since they demanded longer execution time than the other approximated algorithms. All the approximated algorithms achieved good performance when compared with the Opt algorithm. For example, the execution time of the Det Approx is, on average, 1.60% of the execution time of the Opt algorithm. The lower the network load, the longer are the

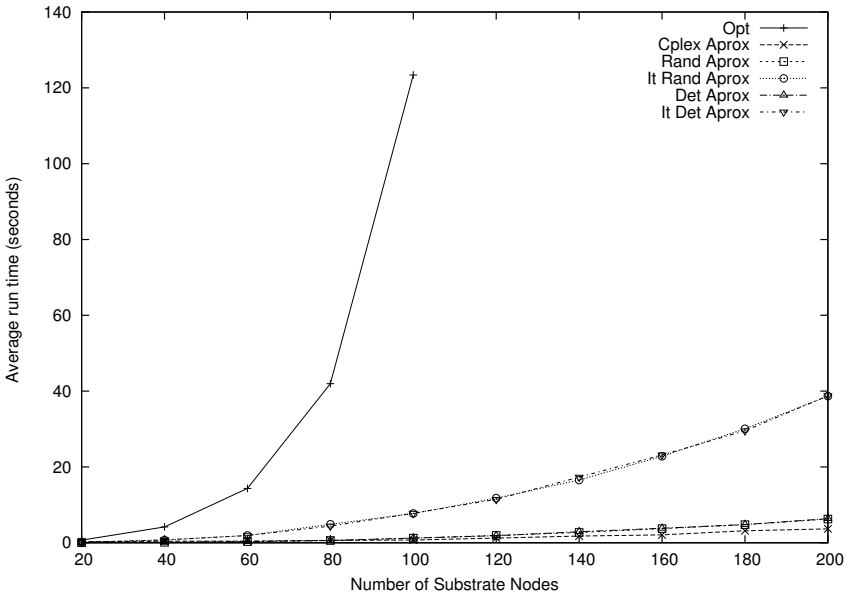

Figure 1: Execution time (First set of experiments)

execution times since the search space of the solution is larger.

Figure 3 and Figure 4 show the average allocated bandwidth as a function of the number of physical routers (first set of experiments) and as a function of the mean arrival time of requests (second set of experiments). Results given by the It Det Approx and by It Rand Approx are not shown since they were similar to those of the Det Approx and the Rand Approx (the curves would be hard to difference if they were included in the figures). As expected, the Opt Approx presented the best performance, but its long execution time (Figure 1 and Figure 2) shows that this algorithm is not a good candidate for real environments. Besides, it was not feasible to wait until the end of its execution when the network had more than 100 physical routers. Results show that the approximated algorithms produce worst results but the gain in execution time compensates this loss of precision. However, several executions of the iterative algorithms with consequent long execution time, did not reduce the average allocated bandwidth. This justifies the utilization of the Det Approx and the Rand Approx in real environments (The results of the Det Approx were on average 3.18% better than those of the Rand Approx).

The blocking ratio was also measured. In the first set of experiments, there was no significant difference between results given by the algorithms. In the second set of experiments, the average blocking ratio of the iterative algorithms was greater than those given by the non iterative algorithms. Although the algorithm based on relaxation increased the blocking ratio, such increase was not significant, of the order of 0.04% when comparing result given by the Root Approximated with those given by the Approx Det.

Table 3 summarizes all the results. The Root Approximated algorithm presented the best performance. The allocated bandwidth and the utilization produced by the Root Approximated algorithm were, on average, only 20.22% greater than those obtained by the

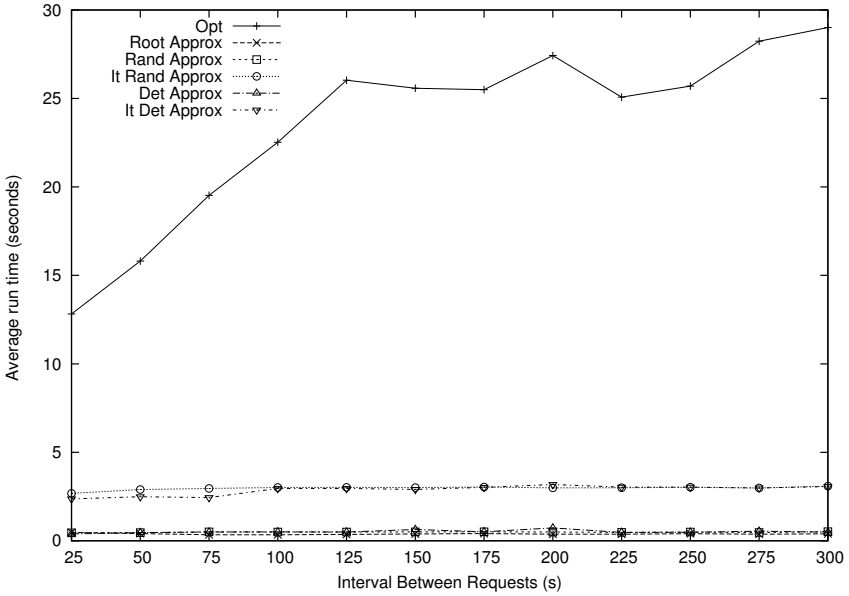

Figure 2: Execution time (Second set of experiments)

Optimal algorithm. Besides, the execution time of the Root Approximated algorithm was the smallest one in all the simulated scenarios and the blocking rate was the same of that given by the Optimal algorithm. The Iterative Random and the Iterative Deterministic demanded the longest execution times, which was a consequence of several executions of the ILP.

4 Conclusions

This paper introduced four novel algorithms based on two 0-1 linear programming formulations. These algorithms differ from previous proposals by the accountability of a large number of characteristics existing in real systems and by the low execution time achieved by them. The algorithms were evaluated for different number of routers in the substrate network and arrival rates. It was shown via numerical examples that the Root approximated provides solutions similar to those of the optimal algorithm for several metrics and demands less computational effort. The Root approximated algorithm also showed to be a good choice for trade-off between quality of solution and computational complexity.

5 Acknowledgment

This research was partially financed by Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), process 2010/03422-5

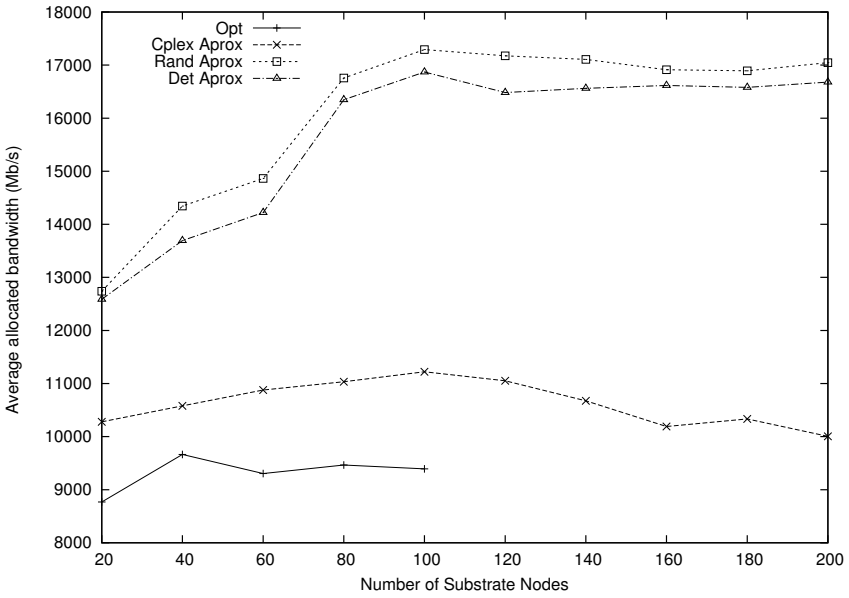

Figure 3: Average allocated bandwidth (First set of experiments)

References

[1] G. Alkmim, D. Batista, and N. da Fonseca, "Approximated algorithms for mapping virtual networks on network substrates," in *2012 IEEE International Conference on Communication (ICC 2012)*, 2012, pp. 1460–1465.

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

[4] 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.

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

[6] 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.

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

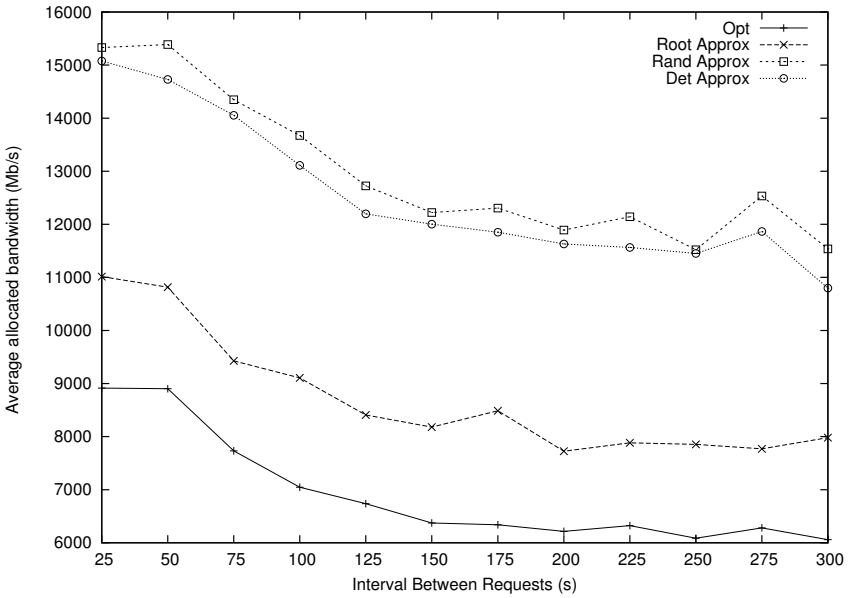

Figure 4: Average allocated bandwidth (Second set of experiments)

[8] 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.

[9] 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.

[10] A. Medina, A. Lakhina, I. Matta, and J. Byers, "Brite," 2011, http://www.cs.bu.edu/brite/. Accessed at 09/19/2011.

[11] R. Albert and A. L. Barabási, "Topology of Evolving Networks: Local Events and Universality," *Physical Review Letters*, vol. 85, no. 24, pp. 5234–5237, Dec 2000.

[12] Cisco Systems, "Cisco Multiprocessor WAN Application Mode [Cisco Catalyst 6500 Series Switches]," 2010, http://www.cisco.com/en/US/prod/collateral/modules/ps5510/product_data_sheet0900aecd800f8965_ps708_Products_Data_Sheet.html. Accessed at 12/20/2010.

[13] RNP, "RNP Backbone map ," 2011, http://www.rnp.br/en/backbone/index.php. Accessed at 09/19/2011.

[14] Cisco Systems, "Cisco 7200 Series Routers Overview [Cisco 7200 Series Routers]," 2011, http://www.cisco.com/en/US/prod/collateral/routers/ps341/product_data_sheet09186a008008872b.html. Accessed at 09/19/2011.

Table 3: Numerical Comparisons (Average values)

First set of experiments			
Algorithm	**Run time (s)**	**Bandwidth (Mbps)**	**Blocking ratio (%)**
Opt	36.92	9319.26	0.56
Root Approx	0.49	10799.00	0.56
Rand Approx	0.46	15199.26	0.56
Det Approx	0.46	14743.84	0.56
It Rand Approx	3.08	15117.86	0.56
It Det Approx	2.99	14729.80	0.56

Second set of experiments			
Algorithm	**Run time (s)**	**Bandwidth (Mbps)**	**Blocking ratio (%)**
Opt	23.60	6917.39	0.06
Root Approx	0.38	8720.91	0.05
Rand Approx	0.48	12967.91	0.09
Det Approx	0.51	12527.41	0.09
It Rand Approx	2.97	12820.58	0.11
It Det Approx	2.86	12442.75	0.10

[15] C. Systems, "Download software," 2011, http://www.cisco.com/cisco/software/release.html?mdfid=278807391&flowid=956&softwareid=280805680&release=12.4.2-XB11&rellifecycle=GD&relind=AVAILABLE&reltype=latest. Accessed at 09/19/2011.