Simulated Annealing Based Resource Allocation for Cloud Data Centers

Wenbo Wang, Xiaolin Chang, Jiqiang Liu, Bin Wang
School of Computer and Information Technology
Beijing Jiaotong University, P.R. China
e-mail: {12120458,xlchang,jqliu, 08283021}@bjtu.edu.cn

ABSTRACT
Network virtualization technology can be applied to provide the performance guarantee by creating a virtual network for running the user application. One of the most important issues in the network virtualization is the *virtual network embedding* (VNE) problem, which deals with the efficient allocation of physical resources in the cloud data center to virtual networks. In this paper, we investigate the ability of simulated annealing technique in handling the VNE problem. The simulation results show that SA technique outperforms both genetic algorithm and particles swarm optimization techniques in handling the cost-aware VNE problem.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search–*Heuristic methods*; I.2.4 [**Computer Communication Networks**]: Distributed Systems–*Distributed applications*

Keywords
Cloud data center, Network virtualization, Virtual network embedding, Metaheuristic, Resource allocation.

1. INTRODUCTION
Network virtualization is regarded as a promising technology to create an ecosystem for cloud computing applications [1]. One of the most important issues in the network virtualization is the *virtual network embedding* (VNE) problem, which deals with the efficient allocation of physical resources in the cloud data center to virtual networks. This paper considers the VNE problem in the peak-demand situations, in which the primary concern of Cloud Infrastructure Provider (InP) is how to embed more VN requests in order to improve the InP's long-term revenue (defined in Equation (1) of Section 2).

When the node and link constraints (including CPU, memory, network bandwidth, and network delay) are both taken into account, the VN embedding problem is NP-hard, even in the offline case. Various techniques, including Meta-heuristic techniques, have been explored to handle the VNE problem. The simulation results in both [3] and [4] demonstrated the better performance of metaheuristics-based VNE algorithms in terms of the InP's long-term revenue and embedding cost (defined in Equation (2)), compared to the existing state-of-the-art algorithms which are not metaheuristics-based.

GECCO'13 Companion, July 6–10, 2013, Amsterdam, The Netherlands.
ACM 978-1-4503-1964-5/13/07.

This paper aims to examine the ability of the basic simulated annealing technique in handling the VNE problem. To the best of our knowledge, no SA-based algorithm has been presented previously to deal with the VNE problem with both node and link constraints. In the rest of the paper, we first present the VNE problem and then describe the details of the proposed CB-SA algorithm in Section 2. We present the evaluation results in Section 3. Here CB denotes the simple node ranking method, proposed in [2]. A node ranking scheme not only determines the virtual node mapping order but also determines the selection order of the candidate physical nodes which can host a virtual node.

2. CB-SA VNE ALGORITHM

2.1 VN embedding
Both the substrate network and the virtual network are modeled as a weighted undirected graph and are denoted by $G_S = (N^S, E^S)$ and $G_V = \left(N^V, E^V\right)$, respectively. Here N^S / N^V is the set of physical/virtual nodes and E^S / E^V is the set of physical/virtual links. Each physical node $n^S \in N^S$ is associated with processing power resources $c(n^S)$ and geographical location. Each physical link $e^S \in E^S$ is associated with bandwidth capacity and delay. Usually, the QoS requirements of a virtual node include the processing power demand and a preferred value expressing how far a virtual node $n^V \in N^V$ can be placed from the specified location. Virtual network embedding for a VN request is defined as a mapping from G_V to G_S with the constraints: ① Each virtual node is mapped to a physical node in a one-to-one manner, and the virtual node QoS requirements are satisfied. ② Each virtual link is mapped to a physical path such that the $e^V(v,w)$ bandwidth requirement is below the total available bandwidth of the physical path or the flow. The revenue $R(G_V(t))$ of serving G_V at time *t* as

$$R(G_V(t)) = \sum_{e^V \in E^V} b(e^V) + \omega \sum_{n^V \in N^V} c(n^V) \qquad (1)$$

where ω is the weight to determine the relative importance of the CPU and bandwidth resources. The embedding cost $C(G_V(t))$ is defined as follows:

$$C(G_V(t)) = \sum_{e^S \in E^S} \sum_{i} \beta(e^S, i) \cdot f^i_{e^S} + \sum_{n^V \in N^V} c(n^V) \qquad (2)$$

Here $f^i_{e^S}$ is defined to denote the total amount of flows from physical node *v* to *w* on the physical link (*v*,*w*) for the VN's i^{th} virtual link, $i \in \left\{1 \ldots \left|E^V\right|\right\}$. $\beta(e^S, i)$ is weight.

2.2 CB-SA Algorithm

We define a vector $H=\left(h_1,h_2,\cdots,h_{|N^S|}\right)$ to denote a solution to the virtual node mapping. H is a $|N^S|$-length integer vector. The value of h_i is set to k if it hosts the k^{th} virtual node, where $1\le k\le|N^V|$. The value of h_i is set to -1 if no virtual node is hosted on it. Whenever H is updated, its feasibility must be checked by using the *k-shortest-path* algorithm to map all the virtual links in the VN. H is called as being feasible if all the virtual links find their hosted physical paths; otherwise H is infeasible. The *fitness* function $f(x)$ is defined as in Equation (2). If H is infeasible, the $f(x)$ value at this temperature is set to $+\infty$. Thus, a feasible H represents a VNE solution. The variable *gBest* is defined to denote the best solution obtained so far.

CB-SA consists of two loops: (1) The inner loop aims to find out a new VN mapping solution. (2) The outer loop is responsible for finding an optimized VN mapping solution. The inner loop uses the Roulette Wheel method to find a physical node to host a virtual node, and applies the simple *k-shortest-path* [6] link mapping algorithm to map virtual links. In addition, CB-SA uses the node ranking method proposed in [2] to determine the virtual node mapping order. Like GA and PSO, CB-SA does not know when an optimal solution is reached. Therefore we define CB-SA stops when the final temperature T_{min} is reached. T_{max} is defined to denote the initial temperature. At each temperature, if $f(x)$ of the new solution is better, the previous solution is replaced. If it is worse it can still be chosen with a probability that depends on the change in the cost function and parameter T, which is gradually decreased during the process. The probability is defined in Equation (3), where *lastH* denotes the VNE solution in last temperature and ε is a weight.

$$p=\exp\left(\varepsilon\frac{f(lastH)-f(H)}{T}\right) \qquad (3)$$

At each T iteration, if $f(H)<f(gBest)$, then *gBest=H*.

3. PERFORMANCE EVALUATION

We compare the performance of CB-SA with CB-GA [4], CB-PSO[3], R-ViNE, and CB-SP [1]. This R-ViNE applies the R-ViNE proposed in [5] to map the virtual nodes but applies the *k-shortest-path*-based method [6] to map virtual links. The parameters in the existing algorithms are set as in the corresponding literature. ω used in Equation (1) and all β used in Equation (2) are set to 1. In CB-SA, T_{max}=50, T_{min}=0. The population size is set to 5 and the number of iterations is set to 20. The configurations of the physical network are set as in [5]. VN requests arrive in a *Poisson process* and the lifetimes of the VN requests follow an exponential distribution. We fix the average VN lifetime to 1000 time units and evaluate the five algorithms by varying the VN request arrival rate from 4 to 8 VN requests per 100 time units. Each simulation is run for 50000 time units. Figure 1 - Figure 3 describe the simulation results. The simulation results indicate that CB-SA works better than the compared VNE algorithm in terms of *Acceptance Ratio*, *Average Revenue*, and *R/C Ratio*. The main reason is that CB-SA can explore the VN embedding solution in a larger searching space. CB-SA enlarges the searching space at least from the following two aspects. The first is its positive probability to a move to a worse solution. The second is that SA's next solution is generated randomly from the current solution. This "randomly" feature may contribute enlarging the searching space.

4. ACKNOWLEDGMENT

The work described in this paper has been supported in part by Beijing Municipal Natural Science Foundation (No. 4123103), Program for New Century Excellent Talents in University (NCET-11-0565), Program for Innovative Research Team in University of Ministry of Education of China (IRT201206).

5. REFERENCES

[1] http://nicira.com/

[2] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration," In Proc. ACM SIGCOMM, 2008.

[3] X. Cheng, S. Su, Z.B. Zhang, K. Shuang, F.C. Yang, Y. Luo, and J. Wang, "Virtual Network Embedding Through Topology Awareness and Optimization," In Elsevier Journal of Computer Networks, 2011.

[4] X.M Mi, X.L. Chang, and J.Q. Liu, "Embedding Virtual Infrastructure Based on Genetic Algorithm," In Proc. IEEE PDACT, 2012.

[5] N. M. M. K. Chowdhury, M. R. Rahman, and R. Boutaba, "ViNEYard: Virtual Network Embedding Algorithms with Coordinated Node and Link Mapping," In IEEE/ACM Transaction on Networking, 2011.

[6] N. Katoh, T. Ibaraki, and H. Mine. An efficient algorithm for k-shortest simple-paths.Networks,12:411–427,1982.

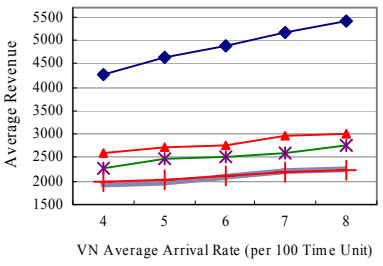

Figure 1. Average revenue over VN average arrival rate

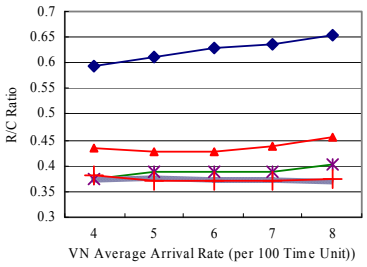

Figure 2. R/C Ratio over VN average arrival rate

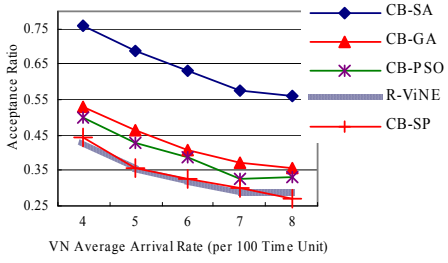

Figure 3. Acceptance ratio over VN average arrival rate