Engineering Applications of Artificial Intelligence 26 (2013) 2540–2550

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Performance evaluation of artificial intelligence algorithms for virtual network embedding ☆

X.L. Chang [a,*], X.M. Mi [a], J.K. Muppala [b]

[a] School of Computer and Information Technology, Beijing Jiaotong University, Beijing, PR China
[b] Department of Computer Engineering, Hong Kong University of Science and Technology, PR China

ARTICLE INFO

Article history:
Received 11 January 2013
Received in revised form
19 May 2013
Accepted 12 July 2013
Available online 9 August 2013

Keywords:
Cloud computing
Network virtualization
Virtual network embedding
Artificial Intelligence

ABSTRACT

Network virtualization is not only regarded as a promising technology to create an ecosystem for cloud computing applications, but also considered a promising technology for the future Internet. One of the most important issues in network virtualization is the *virtual network embedding* (VNE) problem, which deals with the embedding of virtual network (VN) requests in an underlying physical (substrate network) infrastructure. When both the node and link constraints are considered, the VN embedding problem is NP-hard, even in an offline situation. Some Artificial Intelligence (AI) techniques have been applied to the VNE algorithm design and displayed their abilities. This paper aims to compare the computational effectiveness and efficiency of different AI techniques for handling the cost-aware VNE problem. We first propose two kinds of VNE algorithms, based on Ant Colony Optimization and genetic algorithm. Then we carry out extensive simulations to compare the proposed VNE algorithms with the existing AI-based VNE algorithms in terms of the VN Acceptance Ratio, the long-term revenue of the service provider, and the VN embedding cost.

1. Introduction

Cloud computing emerges as a new computing paradigm that offers IT-related capabilities and resources as dynamically configurable services, via the Internet and on-demand, to satisfy users' needs. However, the current cloud data center networks have been a barrier to achieve the promise of cloud computing (http://nicira.com/). Network virtualization is regarded as a promising technology to overcome this barrier and then create an ecosystem for cloud computing applications (http://nicira.com/; Baroncelli et al., 2010). Baroncelli et al. (2010) proposed a network-virtualization-based mediation layer, which is able to provide Network as a Service (NaaS) to cloud computing. Moreover, network virtualization has been regarded as a promising technology for the future Internet (Anderson et al., 2005). Feamste et al. (2007) talked about the evolution of a future Internet architecture consisting of *infrastructure providers* (InPs) and *service providers* (SPs).

One of the most important issues in the network virtualization is the *virtual network embedding* (VNE) problem, which deals with the mapping/embedding of VN requests onto specific physical nodes and paths of the substrate network. When both the node and link constraints are considered, the VN embedding problem is NP-hard, even in an offline situation (Zhu and Ammar, 2006). Various solutions to the VNE problems were proposed based on the different techniques, classified into CPLEX (http://www.ilog.com/products/cplex/)/GLPK (http://www.gnu.org/software/glpk) solver-based exact algorithms and heuristic algorithms. There are at least two advantages of heuristic VNE algorithms: (1) do not impose requirements on the linearity of the QoS composition operators (and thus of objective function and constraints) and (2) scalable. Some authors, such as in Chang et al. (2012) and the references therein, have formulated the VNE problem as mixed-integer programs and designed the CPLEX/GLPK-based exact algorithms. However, it is generally difficult for an exact algorithm to produce an optimal solution in a reasonable amount of time for medium or large size problem instances.

Artificial Intelligence (AI) techniques, such as ACO (Ant Colony Optimization) (Dorigo and Caro, 1999) and PSO (Particles Swarm Optimization) (Eberhart and Kennedy, 1995), have been applied successfully to design heuristic VNE algorithms. The simulation results in Cheng et al. (2011) demonstrated the better performance of PSO-based VNE algorithms in terms of the InP long-term revenue and embedding cost, compared to the existing state-of-the-art algorithms, which are not AI-based. Genetic algorithm (GA) (Goldberg, 2005) is also an excellent approach to solve the complex optimization problems. To the best of our knowledge, no GA-based VNE algorithm has been presented previously and the existing VNE algorithms based on the different AI techniques have never been compared to each other.

☆An earlier version of this paper appeared in IEEE PDCAT 2012 (Mi and Chang, 2012), described in Section V.

* Corresponding author. Tel.: +86 1051688490.
E-mail addresses: xlchang@bjtu.edu.cn (X.L. Chang), Muppala@cse.ust.hk (J.K. Muppala).

0952-1976/$ - see front matter © 2013 Published by Elsevier Ltd.
http://dx.doi.org/10.1016/j.engappai.2013.07.007

There are two objectives of this paper: (1) Compare the computational effectiveness and efficiency of different AI techniques for handling the cost-aware VNE problem in the various situations of peak resource demands. We focus on three AI techniques, namely ACO, GA and PSO. Each AI technique possesses distinctive features in their strategies for searching the solution state space. This paper only considers the classical versions of these AI techniques. (2) Investigate the effect of a node ranking scheme on the performance of an AI-based VNE approach. A node ranking scheme can be used to determine the virtual node mapping order and to determine the possibility of a physical node to host a virtual node, by ranking the relative importance of a virtual/physical node. This paper considers two node ranking methods (denoted as CB and RW), proposed in (Yu et al. (2008) and Cheng et al. (2011). Note that we consider the cost-aware VNE problem in the scenarios where the physical resource demands of each VN request are certain.

The major contributions are summarized as follows:

(1) We propose two new ACO-based VNE algorithms, named as CB-ACO and RW-ACO. There are some differences between our ACO-based VNE algorithms and the ACO-based VNE algorithm proposed in Fajjari et al. (2011), such as the virtual node mapping order and the pheromone trail computation method. The detailed differences are described in Section 3. Our simulation results (unpublished) indicated that the performance of the VNE algorithm proposed in Fajjari et al. (2011) is even worse than that of the greedy VN mapping method, CB-SP. CB-SP algorithm is described in Section 6. The difference between the two proposed ACO-based algorithms is the node ranking method. CB-ACO adopts the CB node ranking method. RW-ACO uses the RW node ranking method. In order to make a fair comparison of the abilities of each AI technique in handling the cost-aware VNE problem, both the CB-ACO and RW-ACO algorithms apply the simple *k-shortest-path* link mapping algorithm (Katoh et al., 1982) to map virtual links.
(2) We propose two GA-based VNE algorithms. To the best of our knowledge, this is the first work of investigating the ability of GA technique in handling the cost-aware VNE problem. We describe the design of the selection, crossover and mutation operators used in the proposed GA-based VNE algorithms in detail. Note that there are some similarities between the proposed ACO-based and the proposed GA-based VNE algorithms, such as individual representation and the definitions of local and global best solutions. Such designs aim to compare the abilities of the AI techniques in a fair way.
(3) We perform a comprehensive comparative study of the proposed VNE algorithms and the existing state-of-the-art AI-based VNE algorithms in various scenarios, including different network topologies (flat random topologies and transit-stub topologies), different sized substrate networks (small-sized, medium-sized and large-sized networks), different physical network configurations, and different VN configurations. Note that the existing research on designing AI-based VNE algorithms compared their designed algorithms only with the VNE algorithms which are not designed based on the AI techniques. In addition, these existing evaluations were carried out only in the small-sized or medium-sized scenarios with a fixed VN arrival rate and a fixed average lifetime. It is possible that one VNE algorithm performs better in a particular scenario, but may not work well when there is a change in the setting of environment. Moreover, the existing AI-based VNE algorithm evaluations were carried out only under flat random topologies.

1.1. Our simulation results indicate that:

(1) An AI technique working well in one area may not work well in the other areas. For example, the authors in Windisch et al. (2007) and the references therein demonstrated that particle swarm optimization is even better than genetic algorithms for solving a number of test problems. However, GA-based VNE algorithms proposed in this paper perform better than PSO-based VNE algorithms in terms of *Average Revenue* (defined in Section 6.2) in all simulations.
(2) In terms of *Average Revenue*, the features of an AI technique have more impact on the performance of the VNE algorithm based on this technique, compared to the node ranking method employed in the VNE algorithm. We also explain the simulation results. The simulation results and the result analysis presented in this paper not only provide suggestions for an infrastructure provider to choose an AI-based VNE approach, but also provide suggestions for designing a robust VNE algorithm based on the AI techniques.

The rest of the paper is organized as follows: Sections 2 and 3, respectively, present the background knowledge and related work in this area. The details of the ACO-based and GA-based VNE algorithms are described in Sections 4 and 5, respectively. We evaluate the AI-based VNE algorithms in Section 6. In Section 7, we summarize the work presented in the paper.

2. Network model and problem description

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 $G_V=(N^V,E^V)$, 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 CPU resources $c(n^S)$ and geographical location $l(n^S)$. The system resources of a physical node include memory, processing power, storage space and so on. Without loss of generality, this paper only considers the processing power. All the work presented in this paper can be applied directly to the environment where virtual nodes have other resource demands besides CPU demand.

Each physical link $e^S(v,w) \in E^S$ between physical nodes (v, w) is associated with bandwidth capacity. All the physical resources (i.e. bandwidth and CPU) in G_S are limited. Usually, a virtual node′s (denoted as n^V) QoS (Quality of Service) requirements include the CPU demand $c(n^V)$ and a preferred value d^{n^V} expressing how far this virtual node can be placed from the specified location $l(n^V)$. The QoS requirements of a virtual link $e^V(v,w) \in E^V$ between virtual nodes (v, w) include the bandwidth requirement $b(e^V)$ and a delay demand. 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 $e^V(v,w)$ is mapped to a physical path (an unsplittable model) or a flow (a splittable model) in G_S between physical nodes which host v and w, with at least two requirements. One is that the $e^V(v,w)$ bandwidth requirement is below the total available bandwidth of the physical path or the flow. The second is that the delay constraint of the virtual link is met.

Each VN request has a lifetime. After the VN′s lifetime is ended, the physical resources allocated to this VN must be released.