DSNRA: a Dynamic Substrate Network Reconfiguration Algorithm

Xinliang LV, Xiangwei ZHENG*, and Hui ZHANG
School of Information Science and Engineering
Shandong Normal University, 250014, Jinan, CHINA

Abstract

Considering the low acceptance rate and unbalanced load issue in virtual network embedding (VNE), a new reconfiguration algorithm in dynamic substrate network environment, called DSNRA, was proposed. The general process of DSNRA is as follows: a new physical node and related links would be added at first when a virtual request was rejected due to bottleneck substrate devices. Then DSNRA found the bottleneck node and other high load rate physical nodes within the distance constraint and put them in a corresponding set. Next, comparing the revenue with the migration cost of the already mapped virtual nodes which were in the above mentioned set, the decision would be made whether to relocate them and their related virtual links. Finally, adding another new physical node and related links and repeat the before steps continuously until the node load rate of the set is under the pre-set index. Experimental results demonstrated that DSNRA could improve acceptance rate and balance the load.

Keywords: Virtual Network Embedding, Dynamic, Reconfiguration and Load Balance;

1 Introduction

With the development of the network virtualization, a number of continuously improving virtual network mapping (VNM) algorithms enhance the acceptance rate remarkably. Infrastructure Providers (InPs) and Service Providers (SPs) could obtain considerable benefit when InPs used the physical network efficiently [1].

However, when a virtual network request's life cycle runs out, substrate network would generate some fragments. These fragments make a result that: even this mapping algorithm is optimal under the current network mapping topological, there is no guarantee that the mapping structure will be always excellent. Virtual network (VN) request acceptance rate drops visibly, a lot of physical nodes or physical links run inefficiency, and only a minority of virtual network requests are embedded on this physical network. The concept of virtual network reconfiguration (VNR) is introduced to redeploy the substrate network. VNR gathers the debris resources and it makes the physical network more potential than this network before it migrated [9].

Since virtual network reconfiguration puts forward, diverse reconfiguration algorithms constantly appear. Different methods (integer linear programming, mixed linear programming, heuristic algorithm etc.) are triggered in different periods (off-line, life cycle run out, VN requests rejected or lacked of resources) [5]. This paper presents a novel reconfiguration algorithm about load balance and acceptance rate, this algorithm increases acceptance rate and balance the load through adding new physical devices around the bottleneck location and doing a series of related migratory operation.

This paper will be divided into the following several parts: the second part is related work; the third part is formulations for Dynamic Substrate Network Reconfiguration Algorithm (DSNRA); the fourth

*Corresponding author: School of Information Science and Engineering, Shandong Normal University, 250014, Jinan, CHINA part is description of the DSNRA; the fifth part is the simulation results and analysis; the last part is conclusion and future work.

2 Related Work

How to relocate virtual network resources is a NP-hard problem [7]. Kinds of researches use different methods to achieve diverse goals. The virtual network reconfiguration algorithm is a kind of greedy algorithm proposed in literature [4], it will be triggered when a virtual network request is rejected. This reconfiguration algorithm reduces the rejection rate (up to 83%) when it has low virtual request arrival rate. But it doesn't consider the energy saving and it has a poor performance when arrival rate is high.

In the literature [8], Phuong Nga Tran etc. take the service disruption's punishment into account. It suggests a solution to increase the acceptance rate under the physical network change least condition. The algorithm adopts a kind of integer linear programming (ILP) method, would be triggered only when the virtual requests lacked of physical resources. Advantages of the algorithm are able to significantly enhance acceptance rate while ensure the new arrival virtual network request is assigned least resources totally. The deficiencies are selection strategy of node migration is not enough precise, and it does not consider the energy saving, link balance and etc.

Literature [3] proposes a novel reconfiguration scheme. This scheme is based on the number, behavior and preferences of network users. It adjusts the network dynamically according to the traffic demand management, the network scale and resource distribution. This algorithm makes the substrate network work efficiently, improves the quality of network service and the user experience. For example, network flows are mainly distributed in the industrial zone in the daytime, but it is necessary to assign more resources to the residential regions at night.

To settle dynamic virtual network reconfiguration problem, the literature [8] proposes a periodic reconfiguration algorithm. Its re-embedding strategy is: real-time monitoring load conditions (including substrate nodes and links). When the load rate is too high, the virtual nodes or virtual links in the heavy load substrate devices would be relocated to the lower load rate physical nodes or links to catch the load-balance goal. Moreover, in order to improve the efficiency of solving dynamic mapping problem, literature [8] presents a reconfiguration method based on path splitting and its core thought is putting virtual links into several sub links. Further, the virtual links need to be remapped could have more chance to be re-embedding, thus this method could increase the acceptance rate.

The literature [6] proposes a dynamically self-adapting virtual networks customized internet (Da Vinci). In this customized internet, the physical links monitor the traffic status of each virtual network, and the shared substrate bandwidth (BW) resources are reallocated to these virtual links periodically based on some local information including the congest level and performance index and so on.

To overcome the issue that the virtual network acceptance rate decreased caused by the bottleneck physical nodes like literature [4] referred and the unbalanced load literature [8] proposed, this paper puts forward the DSNRA to add new physical nodes and its related physical links and remove the already mapped virtual nodes in the bottleneck and high load rate physical nodes to these new physical nodes. DSNRA enhances the acceptance rate and makes the physical load more balance.

3 Formulations for Dynamic Substrate Network Reconfiguration Algorithm

This paper abstracts the underlying network as an undirected graph $G = (n, e)$, where n and e are the sets of physical nodes and links. A request of the virtual network is also abstracted as an undirected graph

Table 1: Parameters

Parameters	Description
NodeNum	The total number of physical nodes
$Load_i$	The load rate of physical node i

$Gv = (n_v,\ e_v)$, where n_v and e_v represent the sets of virtual nodes and links respectively. Table 1 shows the some other parameters.

The model of the algorithm is similar to literature [8], as followed:

$$\min \sqrt{\frac{\sum\limits_{i \subseteq n} \left(Load_i - \overline{Load}\right)^2}{NodeNum}} \tag{1}$$

When the computation formula of $\overline{Load}$ is as follows.

$$\overline{Load} = \frac{\sum\limits_{i \subseteq n} Load_i}{\text{NodeNum}} \tag{2}$$

4 Description of the Dynamic Substrate Network Reconfiguration Algorithm

It's known that two cases could be suffered the following in the substrate network: 1) some virtual nodes densely mapping in one physical node, thus the acceptance rate declined and the whole substrate network load would be unbalanced; 2) some of substrate devices failure, the network service would be interrupted. Due to the second situation mostly adopts backup method which may wastes resources, but it is a better method so far than other already known algorithm. So this paper focuses on the first situation instead of the second situation.

This method monitors the whole physical network load condition in real time, when the physical node load rate is greater than a threshold, it would be remarked as a bottleneck substrate node. When the virtual requests are rejected due to some virtual nodes can't embed in these bottleneck substrate nodes, DSNRA would be trigger. New physical devices would be created and DSNRA tries to reconfigure the virtual request on the bottleneck location to these new devices. To do this DSNR algorithm decreases the congested physical devices' load rates and the load of whole substrate network would be more balance. The pseudocode of DSNR algorithm is as algorithm 1 shown.

Algorithm 1: The Pseudocode of DSNR algorithm

```
Input : nodeNum, threshold $\alpha$, $\beta$, Distance, a set Near and its subset NearSlim, a link set
        workslink, reRevenue, reCost, the number of virtual node mapped in physical node i -
        $nodeNum_i$, checked.
for all $i = 1$ to nodeNum do
    If $Load_i > \alpha$
        create a new physical node nearby i, it has the same CPU resource with i;
        create a set Near, this set includes all the one-hop connected physical nodes with the
        bottleneck node;
        screen out the physical links between bottleneck node and $n \subseteq Near$ which are at least one
        virtual link mapped in them and putting these links into workslink;
        Another end substrate node of links $l \subseteq workslink$ except the bottleneck node end puts
        into NearSlim;
    end if
    for all $j = 1$ to nodeNum do
        If $dist(i, j) < Distance$
            If $Load_j >= \beta$
                for $k = 1$ to $nodeNum_j$ do
                    If $reRevenue_k > reCost_k$
                        reconfigure k to the new created node;
                        $checked_k = 1$;
                    Else
                        $checked_k = 1$;
                    end if
                end for
            end if
    end for
    for all $j = 1$ to nodeNum do
        for $k = 1$ to $nodeNum_j$ do
            If $Load_j >= \beta$
                GOTO 3;
            end if
        end for
    end for
end for
```

Where the function $dist(i, j)$ is to calculate the geographical distance between substrate node i and j. *Distance* is the max distance of the virtual node in one node could remap to another substrate node.

5 Simulation and Analysis

5.1 Environment

Experiments of this paper used C++ programming language, run in the eclipse tools and developed in the Linux operating system. GNUPLOT was used to be the draw tool drew these simulation result figures.

5.2 Parameters

Most of experiment configurations are shown in the table 2.

Table 2: Configuration Table

Parameters Description	Value
The initial of the underlying network	100 substrate nodes and 467 substrate links
Every physical node's CPU resource and every physical link's BW resource	Fixed
Total virtual requests	100
The virtual node number of a virtual request	2–10 (average 5)
The probability to produce a virtual link between every two virtual nodes	0.75
The upper bound of a virtual node or link	10% of a substrate node's CPU resource or a substrate link's BW resource
The path splitting ratio	50%
The value of α	0.8
The value of β	The average load rate at this moment

5.3 Index

5.3.1 Acceptance rate

The acceptance rate computation formula is as follows:

$$AR = \frac{aCount}{tCount} \tag{3}$$

When the aCount represents the number of virtual requests which are accepted, tCount represents the total coming virtual requests number.

5.3.2 Total embedding cost

It represents the virtual requests embedding cost including node and link migration cost. Computation formula is as follow:

$$tot\cos t = \sum_{i=1..vnrNodeNum} vNode_{iCPU} + \sum_{j=1..vnrEdgeNum} vLink_{jBW} \tag{4}$$

When the *vNode$_i$CPU* represents the CPU demand of virtual node i. *vnrNodeNum* represents the number of already mapped virtual nodes. $vLink_{jBW}$ represents the BW demand of virtual node j. *vnrEdgeNum* represents the number of already mapped virtual links.

5.4 Experiment Data and Analysis

In this paper, it adopts the following mapping strategy: Greedy method [10] is adopted in node mapping section. This method means that under the distance constraint, the substrate node will be chosen which

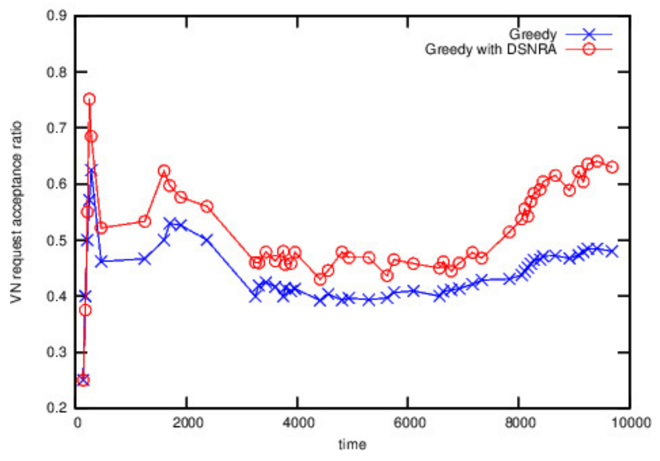

Figure 1: Acceptance Rate

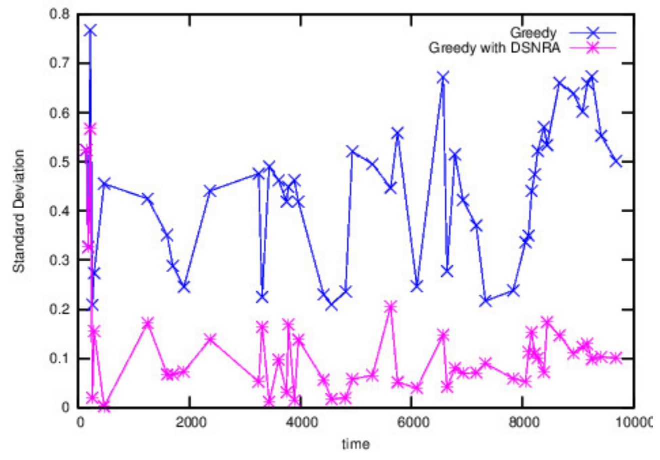

Figure 2: Standard Deviation

has maximum CPU resources. And this paper adopts R-Vine [2] as link mapping strategy. The experiment would compare Greedy combined with DSNR algorithm with the original Greedy method without reconfiguration method from acceptance rate, standard deviation and total cost these three aspects. The analyze simulation results and related analyses is as followed.

Figure 1 demonstrates Greedy combined DSNRA's acceptance rate is much higher than without this reconfiguration strategy. It is due to the more balanced load, the new coming virtual requests would decrease the reject chance caused by a few virtual nodes or links couldn't find suitable embedding target. In other words, the whole virtual request had a higher probability of being mapped.

Figure 2 shows the standard deviation of Greedy and Greedy with DSMRA. The lower the standard deviation is, the more balance the total substrate network would be. It is observed that when VNM cooperated with DSNRA, its standard deviation is lower than only Greedy about 68%.

Figure 3 reflects the total embedding cost. It is noticed that there is one more line besides the two this experiment referred named Greedy DSNRA before, it is this algorithm initial used method: when new substrate node created, it would be connected with all one-hop neighborhood of bottleneck node. Compared with this paper's latest Link policy, it may waste more cost than DSNRA. Of course, they are both taken more cost than the simple VNM (Greedy) without migration. But considering the two main

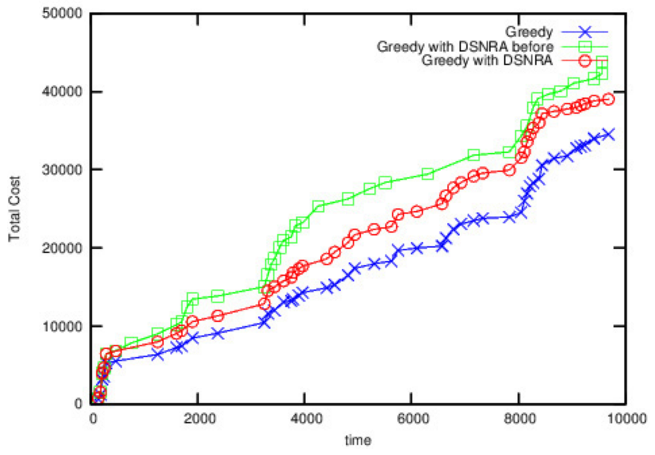

Figure 3: Total Embedding Cost

goals are well implemented, this slightly extra cost could be accepted.

6 Conclusion and Future Work

The following conclusion can be got through the experimental simulation. In our study, we designed a novel dynamic substrate network method which is more similar to the practical application scene. Meanwhile, a precise link generation strategy is proposed to decrease unnecessary cost. When new substrate nodes are added, virtual nodes in high load rate physical nodes are relocated and therefore the load becomes more balanced and acceptance rate is improved. Our future work will consider dynamic substrate part and virtual part. In addition, we will study other newly emerging index such as QoE, QoS, energy consumption, and so on.

References

[1] X. Cheng, Z.-B. Zhang, S. Su, and F.-C. Yang. Survey of virtual network embedding problem. *Journal of China Institute of Communications*, 32(10):143–151, 2011.

[2] M. Chowdhury, M. R. Rahman, and R. Boutaba. Vineyard: Virtual network embedding algorithms with coordinated node and link mapping. *IEEE/ACM Transactions on Networking (TON)*, 20(1):206–219, 2012.

[3] S. Ci, B. Yu, and Y. Han. Main of service migration technologies in virtual networks. *ZTE Technology Journal*, 20(3):32–40, 2014.

[4] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann. Vnr algorithm: A greedy approach for virtual networks reconfigurations. In *Proc. of the 2011 IEEE Global Telecommunications Conference (GLOBECOM'11), Houston, Texas, USA*. IEEE, December 2011.

[5] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach. Virtual network embedding: A survey. *IEEE Communications Surveys & Tutorials*, 15(4):1888–1906, 2013.

[6] J. He, R. Zhang-Shen, Y. Li, C.-Y. Lee, J. Rexford, and M. Chiang. Davinci: Dynamically adaptive virtual networks for a customized internet. In *Proc. of the 4th Conference on Emerging Network Experiment and Technology (CoNEXT'08), Madrid, Spain*, pages 15:1–15:12. ACM, December 2008.

[7] X.-L. Li, H.-M. Wang, B. Ding, C.-G. Guo, and X.-Y. Li. Research and development of virtual network mapping problem. *Ruanjian Xuebao/Journal of Software*, 23(11):3009–3028, 2012.

[8] P. N. Tran, L. Casucci, and A. Timm-Giel. Optimal mapping of virtual networks considering reactive reconfiguration. In *Proc. of the 2012 IEEE 1st International Conference on Cloud Networking (CLOUDNET'12), Paris, France*, pages 35–40. IEEE, November 2012.

[9] X.-g. Wang, X.-w. Zheng, and D.-j. Lu. A heuristic virtual network mapping algorithm. In *Proc. of the 10th International Conference on Intelligent Computing (ICIC'14), Taiyuan, China*, volume 8589 of *Lecture Notes in Computer Science*, pages 385–395. Springer International Publishing, August 2014.

[10] Y. Zhu and M. H. Ammar. Algorithms for assigning substrate network resources to virtual network components. In *Proc. of the 25th IEEE International Conference on Computer Communications (INFOCOM'12), Barcelona Spain*, volume 12. IEEE, April 2006.

Author Biography

Xinliang LV is pursuing MS degree at the School of Information Science and Engineering, Shandong Normal University, Jinan, China.

Xiangwei ZHENG is currently serving as a Professor in the School of Information Science and Engineering, Shandong Normal University, Jinan, China. His research interests include computational intelligence, computer networks, cloud computing.

Hui ZHANG is pursuing MS degree at the School of Information Science and Engineering, Shandong Normal University, Jinan, China.