Clustered Candidate Path (CCP) Algorithm for Higher Scalability in Virtual Networks

Hitoshi Yabusaki
Hitachi, Ltd., Yokohama Research Laboratory
292, Yoshida-cho, Totsuka-ku,
Yokohama-shi, Japan
+81-45-860-3084
hitoshi.yabusaki.vw@hitachi.com

Daisuke Matsubara
Hitachi, Ltd., Yokohama Research Laboratory
292, Yoshida-cho, Totsuka-ku,
Yokohama-shi, Japan
+81-45-860-3084
daisuke.matsubara.pj@hitachi.com

Abstract—**The diversity of applications has highlighted the virtual networks that provide communications of different qualities at lower cost. To realize it, a network management system (NMS) needs to set paths satisfying constraints while efficiently using network resources. We proposed the Clustered Candidate Path algorithm, which is a scalable traffic engineering algorithm that satisfies both multiple constraints network-resource efficiency. The proposed algorithm solves the drawback of the conventional Candidate Path algorithm that needs to recalculate all the candidate paths between every pair of edge nodes when the topology changes. The point of the proposed algorithm is to cluster nodes and to limit the recalculated area. The simulation result showed the proposed algorithm cut 90% of paths to be recalculated.**

Keywords-component; path calculation; cluster network; candidate path.

I. INTRODUCTION

The diversity of applications such as online stock transaction and Virtual Machine (VM) migration has highlighted the network virtualization that allow multiple virtual networks of different qualities and capacities to coexist in a single physical network. To realize it, network management system (NMS), which manages resources and provisions paths to each node, needs to set paths satisfying constraints such as delay, bandwidth, pairwise-disjoint, and designated-node constraints while efficiently using network resources.

The Candidate Path (CP) algorithm enables traffic engineering for higher-resource efficiency while satisfying the multiple constraints. It keeps several candidate paths between every pair of edge nodes and select active and backup paths from the candidate paths considering constraints and residual bandwidth of all the links [1]. The drawback of the CP algorithm is that it needs to recalculate all the candidate paths between every edge nodes when nodes or links are added or eliminated. Furthermore, as the network size increases, the recalculation time increases exponentially. To solve this problem, we have proposed clustering candidate path (CCP).

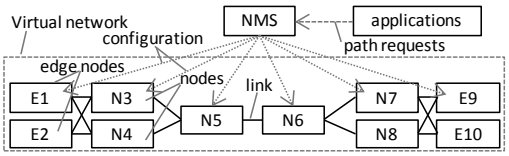

Figure 1. Network Architecture.

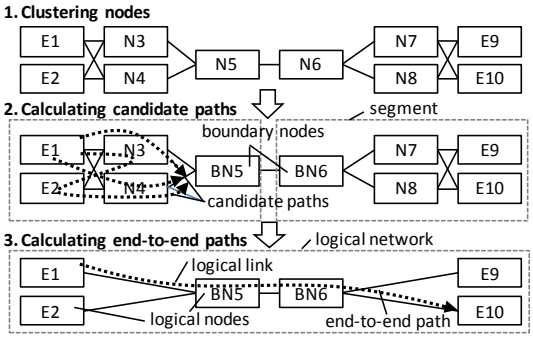

Figure 2. Clustered Candidate Path Algorithm.

II. CLUSTERED CANDIDATED PATH ALGORITHM

A. Overview

The CCP algorithm avoids recalculating every candidate path by clustering nodes and limiting the recalculation segment. First, the NMS clusters the nodes. Then, it calculates several candidate paths between every pair of edge and boundary nodes in the same segment. The NMS manages the logical network composed of logical links and nodes; the logical links include the calculated candidate paths and the links between boundary nodes; the logical nodes include the edge and boundary nodes. When it receives a path request, the NMS calculates an end-to-end path on the basis of the logical network. The point of the algorithm is that each logical link has attributes of the top several candidate paths.

B. Clustering Nodes

To minimize the calculation cost of end-to-end paths, the nodes should be clustered so that the number of the logical links connecting boundary nodes becomes the smallest. To achieve this, we cluster nodes so that modularity maximizes.

The modularity is designed to measure the strength of division of a network into segments. Networks with high modularity have dense connections between the nodes within segments but sparse connections between nodes in different segments. The modularity Q is defined as

$$Q = \sum_{i \in I} \left(e_{ii} - \left(\sum_{j \in I, j \neq i} e_{ij} \right)^2 \right), \quad = \sum_{i \in I} \left(e_{ii} - {a_i}^2 \right) \tag{1}$$

where e_{ii} is the number of links between nodes within the segment i; e_{ij} is the number of links between a node within the segment i and one within the segment j; and a_i is the number of links between a node within segment i and one within the others. The modularity is known as NP-hard to maximize; therefore we applied the well-known greedy heuristic algorithm [2]. In the algorithm, instead of Q, ΔQ is calculated with

$$\Delta Q = e_{ij} + e_{ji} - 2a_i a_j \quad = 2(e_{ij} - a_i a_j) \tag{2}$$

In the initial condition, all the nodes belong to different segments. The algorithm repeats merging the two segments with the largest ΔQ until all merges would reduce the modularity. Thus, nodes are clustered into several segments.

Then, the NMS creates a logical network. The logical network is composed of logical nodes and links. The logical nodes include edge and boundary nodes; the other nodes are abbreviated in the logical network. The logical links connect two boundary nodes or a boundary node to an edge node. The attributes of logical links connecting a boundary node to an edge node are calculated in the next section C.

C. Calculating Candidate Paths

In each segment, the NMS calculates multiple paths, or candidate paths, between every pair of boundary and edge nodes. First, it searches paths by the width first search while the number of hop is under the limit. The limit can be calculated as the proportional function of the number of minimum hop. After searching multiple paths, the NMS eliminates paths without pairwise-disjoint paths. Then, the NMS sorts the paths in order of the number of hops. Thus, candidate paths with pairwise-disjoint paths are calculated.

Then, NMS maps the top several candidate paths and their disjoint paths to the logical links. The point of the algorithm is that each logical link has attributes of the top several candidate paths. Thus, a single logical link has several combinations of communication characteristics such as delay, bandwidth, and redundancy.

D. Calculating End-to-end Paths

When receiving path requests, the NMS calculates an end-to-end path on the basis of the logical network. First, it searches paths by the width first search while the number of hop is under the limit and eliminates paths without pairwise-disjoint paths, as the same with the section C. Then, it selects the best combination of attributes of each logical link that satisfies all the constraints and has the lowest cost.

When a node is newly added to the initial network topology, instead of clustering nodes from scratch, the NMS only selects the segment with the largest ΔQ for the additional node to join and change the boundary nodes if necessary. Then recalculate the candidate paths in the selected segment and maps the calculated paths and the logical links.

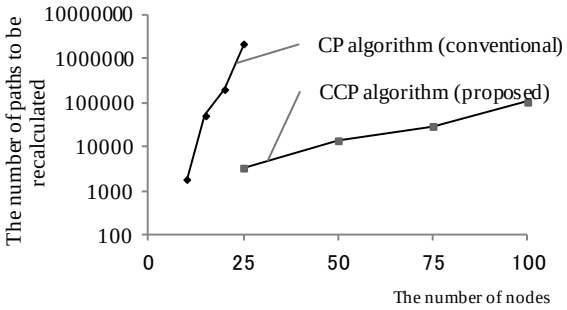

Figure 3. The number of paths to be calculated.

III. SIMULATION

The drawback of the CP algorithm is that it needs to recalculate all the candidate paths between every edge nodes when the nodes or links are added. Hence we have performed simulation and evaluated the number of paths to be calculated. We compared the proposed algorithms with the path calculation without clustering nodes, i.e. Candidate Path algorithm. We adopted the BA model as a network topology where 10% of nodes are connected with links.

Figure 3 shows the number of paths to be calculated. It shows that the CCP algorithm cut more than 90% of paths to be calculated compared to the conventional CP algorithm. In the case of CCP algorithm, the number of segments is 3 where the number of nodes is 25 and 50, and 4 where it is 75 and 100.

IV. CONCLUSION

We proposed the Clustered Candidate Path (CCP) algorithm, which is a scalable traffic engineering algorithm for higher-resource efficiency while satisfying multiple constraints. The proposed algorithm solves the drawback of the conventional Candidate Path (CP) algorithm that needs to recalculate all the candidate paths between every edge nodes when the topology changes. The point of the proposed algorithm is to cluster nodes and to limit the recalculated segment. The simulation result showed the CCP algorithm cut 90% of paths to be recalculated compared to the conventional CP algorithm.

REFERENCES

[1] H. Yabusaki and D. Matsubara, “Study on Path Control for Reliable Transport,” IPSJ SIG Notes, vol. 2009, No. 21, pp. 85-90, Mar., 2009.

[2] Newman, M. E. J., "Modularity and community structure in networks," PROCEEDINGS- NATIONAL ACADEMY OF SCIENCES USA vol. 103 No. 23: 8577-8696, 2006.