Development of optimized algorithm for virtual to physical resources mapping in SDN

Mikhail Lebedev
Faculty of computational mathematics and cybernetics
Lomonosov Moscow State University
Moscow, Russia
mlebedev@lvk.cs.msu.ru

Alexander Shalimov
Faculty of computational mathematics and cybernetics
Lomonosov Moscow State University
Moscow, Russia
ashalimov@lvk.cs.msu.ru

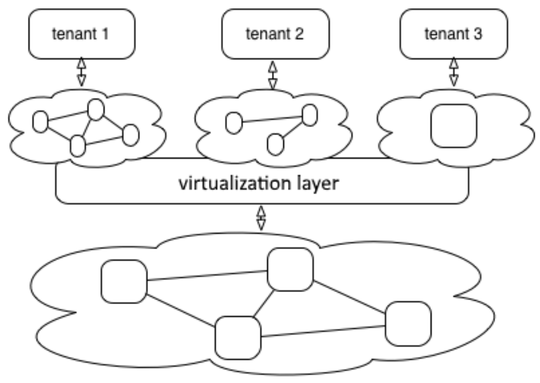
INTRODUCTION

Virtualization is present in a lot of areas of IT. For networks virtualization is a provision of virtual network for user, which is abstracted from physical resources, but looks like real network. And on the same physical resources we can place several virtual networks.

In traditional networks, the virtualization problem is solved by adding some information in header or encapsulation by special protocols. But all these solutions introduce limitations on hardware and don't allow to create custom virtual topologies.

Appear of software-defined networks resolve these problems. Nowadays there are some virtualization systems for SDN like FlowVisor and OpenVirteX, which organize program interlayer between physical network and controllers, with abstraction of topology and address space. However, these platforms do not allow automatically map virtual networks to physical resources.

For mapping custom virtual topologies in [4] the authors have developed algorithm for OpenVirteX platform, which allow user to send to OVX requests for create virtual network without information about physical network. But algorithm work not in all cases and for practical use algorithm requires more successful mappings.



PREVIOUS WORK

Algorithm:

Let's use the following notations:

PS – set of physical switches

PL – set of physical links

VBS – set of virtual border switches with connected devices

VS – set of virtual switches without devices

VL – virtual links

TN – terminal nodes (groups of physical switches)

PN – physical network graph

VN – virtual network graph

Thus problem reduced to problem of mapping virtual network weighted graph to physical network weighted graph with *SLA* on bandwidth for each link.

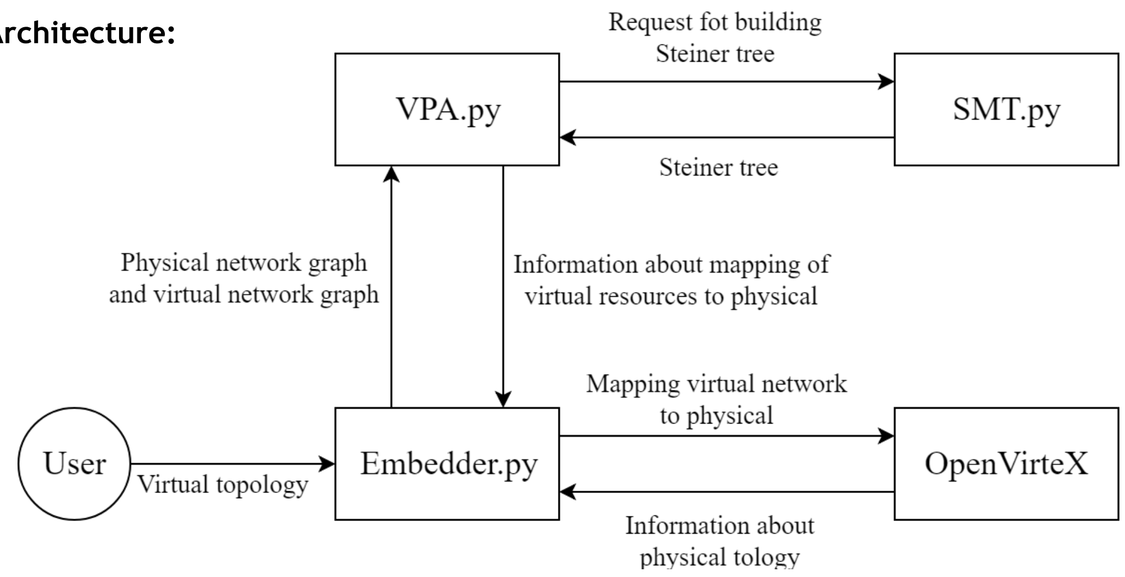
Our model has two restrictions set by OVX platform:

1. Each physical switch can be used in only one virtual switch (*vs* or *vbs*)
2. Virtual switch can be mapped to one physical switch or to group of physical switches with tree structure.

The original algorithm consists from three main steps:

1. Mapping of virtual border switches. For each *vbs* we have group of terminal nodes (physical switches). For this nodes we need to build Steiner minimal tree. For each node from the current group metric closure is built in PN graph. Then on gotten metric closure we need to build minimal spanning tree. Approximate Steiner minimal tree building is finished. After building all *vbs* we check first restriction of model.
2. Building direct links between *vbs*. For all direct links between *VBS* the following algorithms is starting: for each pair ps_i and ps_j from first and second *vbs* of link shortest path is computed with 1/bandwidth metric by Dijkstra's algorithm; the shortest path is selected for current link mapping.
3. Mapping of *VS* and other links. For each subset PS^k of *PS* such that $|PS^k| = |VS|$, which consist from not used *ps*, being built all possible mappings. Links between *vs* is build similar to second step. If mapping satisfies the requirements, then stop algorithm.

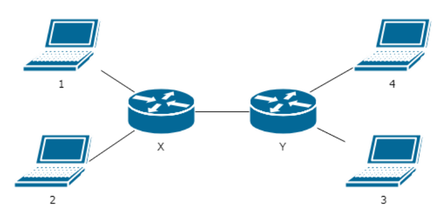
Architecture:



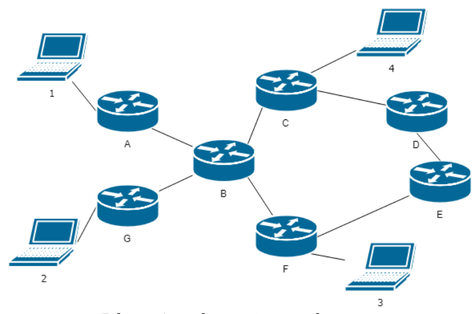
PROPOSED MODIFICATION

All cases of stopping algorithm occurred at first step. After analysis of results of work became clear the problem. Algorithm always stops after the first step, because one physical switch is used by several virtual switches in that at first step Steiner trees were built separately. For solving this problem algorithm was modified. After building every tree for each *vbs*, used physical switches is removed from set of available *ps*. Also for each *vbs* set of available *ps* do not consist terminal vertices of others *vbs*. During the research was tested two algorithms of building minimal spanning tree – Prim's and Kruskal's, but results were similar.

Example:



Virtual network



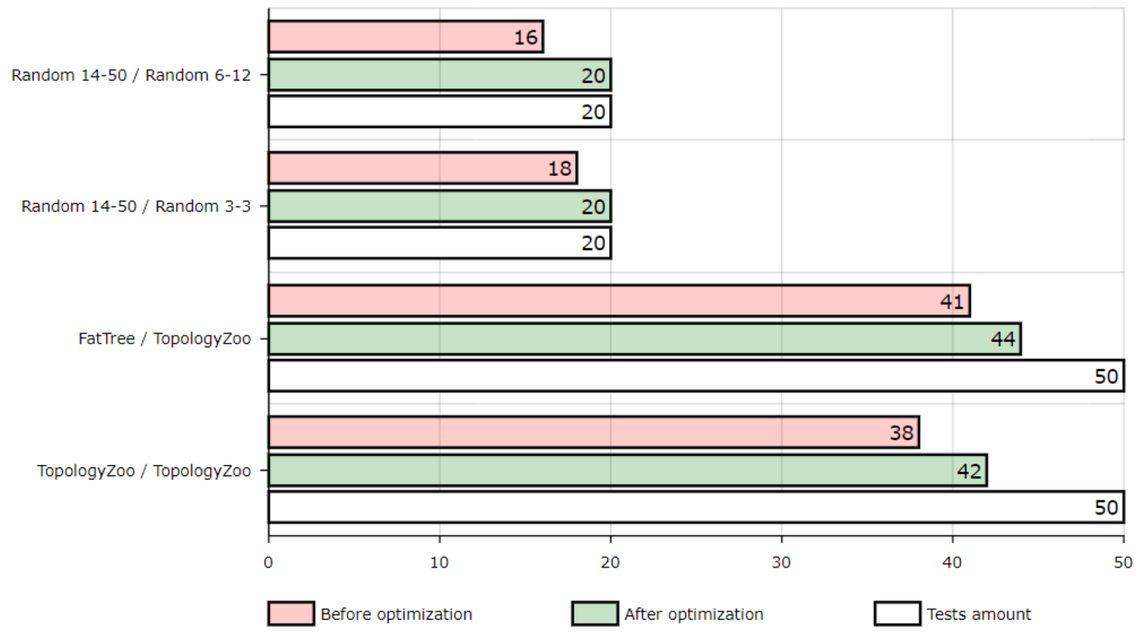
Physical network

There are two virtual border switches – X, Y. *VBS* X will be mapped to ABG, Y to CBF. This mapping at first step will lead to conflict situation, because two *vbs* use the same physical switch B, but solution exists (X to ABG and Y to CDEF).

RESULTS

Algorithm was tested on four group of tests:

1. Physical – PLOD graph (14 vertices, 50 edges); Virtual – PLOD graph (6 vertices, 12 edges); 2 groups of terminal vertices by 3 in each
2. Physical – PLOD graph (14 vertices, 50 edges); Virtual – PLOD graph (3 vertices, 3 edges); 2 groups of terminal vertices by 2 in each
3. Physical – FatTree; Virtual – TopologyZoo; 2 groups of terminal vertices by 2-4 in each
4. Physical – TopologyZoo; Virtual – TopologyZoo; 2 groups of terminal vertices by 2-4 in each



Algorithm modification helped to exclude conflict situations on first step and achieve high results on two first groups of tests. However, on second two groups results were not so significant, because tests have sophisticated topologies, which is can't be checked manually.

FUTURE WORK

For further improve the results, it is necessary:

- To research the influence of the display order in the first stage.
- To develop heuristic algorithm for *PS* mapping for third step.
- To explore ways to change method of mapping switches in OpenVirteX platform for removing restrictions.

REFERENCES

1. Ali Al-Shabibi, Marc De Leenheer, Ayaka Koshibe, Guru Parulkar and Bill Snow, Matteo Gerola and Elio Salvadori – "OpenVirteX Make Your Virtual SDNs Programmable", 2014
2. Bang Y. W., Kun-Mao C. Steiner Minimal Trees – "Spanning Trees and Optimization", Chapman & Hall/CRC Press, USA, 2004.
3. Brandon Heller – "OpenFlow Tutorial", Open Networking Summit, Li Ka Shing Center, Stanford University, Oct 17, 2011
4. Evgenij Bykovec – "Development of the algorithm of mapping virtual resources to physical resources in software-defined networks" Lomonosov Moscow State University, Russia, 2015
5. Source code – github.com/MikhailLebedev/MappingAlgorithm

RESEARCH POSTER PRESENTATION DESIGN © 2011
www.PosterPresentations.com