Mapping Strategy of Simulation Nodes Based on Topology Partitioning

Xingman Chen [a], Yanhui Guo, Qi Li

Beijing University of Post and Telecommunications, Beijing, 100876, China

[a]x-man@bupt.edu.cn

Keywords: network simulation, mapping of simulation nodes, topology partitioning, load balance.

Abstract. In the visualized network simulation system, both hardware and software resources are partly occupied to run the simulation nodes. And distributed simulation service is required to be provided by multiple physical servers if the scale of simulation network is rather large. In order to solve the problem of mapping from simulation nodes to physical servers in the distributed environment, this paper presents a mapping strategy of simulation nodes based on topology partitioning. First, we transform the topology network into the weighted graph according to the degree of the nodes and links. Then we map the simulation nodes based on the weighted graphs. By using the proposed method, the loading balance among the multi-physical servers can be achieved well, and fewer physical servers' resources are consumed.

Introduction

With the development of computer network, most large enterprises have run their own communication network with growing scale considering the networks security. Therefore, maintaining and expending the communication network are rather difficult for those enterprises. With visualized simulation technology, the simulation nodes can be given such similar functions as real network devices have, so that users can operate the simulation nodes as they do to the real network devices. However, the scale of communication network run by enterprise is quite large, and there might be hundreds of or more simulation nodes in the network. Thus, the visualized simulation experiment can't be conducted with only one but several physical simulation servers to provide distributed simulation service. Under that condition, it needs to map the simulation network to the physical one. So far many strategies have been raised to conduct that mapping [1, 2, 3], which deal with the problem that, with certain constraints, maps simulation nodes and links to the physical ones. Visualized simulation platforms with those strategies must map their simulation links to the corresponding physical ones. In other words, if two simulation nodes happen to be placed into the same physical node, the communication between them must be achieved through physical links. Therefore, the constraints between mappings in nodes and links must be considered.

Based on the above analysis, this paper addresses a new mapping strategy of simulation nodes based on network topology partitioning. First, set weights for simulation nodes and links according to their degree of core in the network topology. Then, partition the topology graphs by using the tool, and map the simulation nodes. It turns out that with this strategy, mapping from simulation nodes with large traffic to the same physical servers can be achieved, and meanwhile, the simulation nodes in different servers come into approximately equal amounts. In addition, under the same experimental environment, with this strategy, it takes up fewer hardware resources of physical simulation servers compared with the simple random mapping, and thus the efficiency of operation are promoted to a large extent.

Mapping Strategy of Simulation Nodes Based on Topology Partitioning

K-way graph partitioning is defines as: given an undirected graph $G = (V, E)$, including $V = \{v \mid v\ is\ the\ vertex\ of\ G\}$, the set of vertices, and $E = \{(v_i, v_j) | v_i, v_j \in V, v_i\ and\ v_j\ are\ adjacent\}$, the set of edges in G. K-way graph partitioning is a method that partitions V into k subsets, namely $V_1, V_2, \cdots V_k$. Among the k subsets $V_i \cap V_j =$

$\emptyset \; when \; i \neq j$, and$\cup_i V_i = V$. Let $|V|$ be the number of vertices in G, and each subset contains $|V|/k$ vertices. The related vertices belonging to different subsets of E is in its minimal number.

In this paper, first we evaluate the degree of core for each node and link in the simulation network, and hereby set weights for them, as a result, we manage to transform the mapping problem into one of topology partitioning. Specialized partitioning tool is employed here to achieve the ideal partitioning results. At last, we can map the simulation nodes based on what we have achieved in the last steps.

Set Weights for the Simulation Nodes and Links. In the real network, the closer network device is to the core, the more loads it carries, which is directly proportional to its performance. Therefore, to keep the balance of simulation loads is to achieve equilibrium of total simulation resources that nodes in the subnets take up. In the distributed simulation environment, the target to partition the network topology is to minimize the number of the packets in the network transmission. In result, the results of mapping of simulation nodes are closely related with the degree of core for nodes and links in the network.

The degree of core for nodes and links can be measured by Betweenness Centrality, which is derived from the evaluation of individual importance in social network.

The Betweenness Centrality for arbitrary node v in graph $G = (V, E)$ is defined as following:

$$C_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \quad (1)$$

In the equation above, $C_B(v)$ is the Betweenness Centrality of v. σ_{st} is the number of the shortest paths between s and t. $\sigma_{st}(v)$ is the number of the shortest paths between s and t which go through node v.

If

$$\delta_{st}(v) = \frac{\sigma_{st}(v)}{\sigma_{st}}, \quad (2)$$

$$C_B(v) = \sum_{s \neq v \neq t \in V} \delta_{st}(v). \quad (3)$$

Betweenness Centrality of the edge e is defined as following:

$$C_B(e) = \sum_{s \neq v \neq t \in V} \delta_{st}(e), \delta_{st}(e) = \frac{\sigma_{st}(e)}{\sigma_{st}}. \quad (4)$$

$C_B(e)$ is the Betweenness Centrality of the edge e. σ_{st} is the number of the shortest paths between s and t. $\sigma_{st}(e)$ is the number of the shortest paths between s and t which go through node e.

According to the definition of the Betweenness Centrality, the higher degree of core for nodes and links in the network is, the more the shortest paths which go through this node or link is, and the larger the Betweenness Centrality of the node and link is and vice versa. So we set the value of Betweenness Centrality as the weight of node and link in the topology partitioning.

In this paper, we use the algorithm proposed by Ulrik Brandes[6] to calculating the value of Betweenness Centrality, which has been the highly efficient one so far. The algorithm concludes such steps as: first, take arbitrary node as the source node; second, searche the shortest paths between the source node and other nodes by using BFS (Breadth First Search); at last, calculate the Betweenness Centrality of those shortest paths. We can obtain the final values of Betweenness Centrality of all nodes and edges by accumuling all the values of Betweenness Centrality of the source nodes in the graph.

We take node as an example, and the algorithm can be illuminated as follows.

The Betweenness Centrality of the arbitrary node *v* dependent on the source node *s* is defined as $\delta_{s\cdot}(v) = \sum_{t \in V} \delta_{st}(v)$, in which is the set of nodes in the graph. The deduction can be drawn as following.

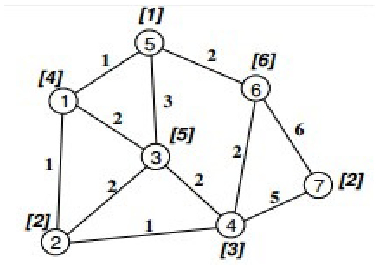
$$C_B(v) = \sum_{s \neq v \neq t \in V} \delta_{st}(v) = \sum_{s \in V} \sum_{t \in V} \delta_{st}(v) = \sum_{s \in V} \delta_{s\cdot}(v) \quad (5)$$

And by BFS, we can work out $\delta_{s\cdot}(v)$ taking s as *Root Node*.

Topology Partitioning Method. So far there have been many topology partitioning methods [7,8]. In this paper, we partition the topology network graph by using Metis tool [9]. It is well acknowledged. And what's more, it has brought the multi-layered K-way graph partitioning model proposed by George Karypi and Vipin Kumar [7,8] into practice. With Metis tool, we can partition the undirected graph with weights of both nodes and edges properly into subgraphs with equal scales. In addition, the connectivity among those subgraphs comes to the minimum. For the topology graph concluded in 1.1, we partition the graph by using Metis tool to achieve balance of loads and minimize the traffic among all subnets. Finally, we map the simulation nodes to each physical servers.

We conduct it with following 3 steps.

1) We abstract the network topology graph as the processible input file, including graph and mesh files. And 4 types of graph file are subdivided by that the node or edge weights are concluded or not. What mesh file contains is only the topology information among vertices, but excludes information of edge weight. We take the weighted graph file as the input file. In sequence, the 3 parameters in the first line represent the numbers of vertices, edges, and the model of weights respectively in the graph—011 represents that node and edge are both weighted. The parameters in the i-th line($i \in (2, |V| + 1)$) represent the related information that the vertices and edges are adjacent to the vertex i-1. In Fig.1-1, we give an example to show the Corresponding Relationship between Weighted Graph and the Graph File



```
7 11 011
4 5 1 3 2 2 1
2 1 1 3 2 4 1
5 5 3 4 2 2 2 1 2
3 2 1 3 2 6 2 7 5
1 1 1 3 3 6 2
6 5 2 4 2 7 6
2 6 6 4 5
```

Fig.1 Corresponding Relationship between Weighted Graph and the Graph File

2) We partition topology graph by using Metis tool. As conducting it, not only the designated graph file, but also the specific value k, which is the number of the subsets partitioned, i.e. the number of physical servers, is required. In addition, unbalance factor—the degree of unbalance among those partitioned subsets, is also required. With Metis tool, we can draw correspondence between each vertex and the subgraph it is in, and output the result to *. graph. part file, whose total number of rows is equal to that of the vertices in the graph. Besides, the parameters in the i-th line of the file is the subgraph G the vertex i belongs to, and $i \in [1, |V|], G \in [0, k - 1]$. Finally, the result of the topology graph partitioning can be got from the correspondence between network topology graph and numbered dot-linear one.

3) We map the simulation nodes to the physical servers on the basis of what we get from the above steps.

Test and Analysis

Experiment Description. In the same experimental environment and scale, we employ two mapping strategies of simulation nodes. One is bsed on random mapping, and the other is the mapping based on topology partitioning. Then we contrast the amounts of resources in physical servers their simulation service programs have taken up. We are given 3 physical simulation servers with identical hard and soft ware configurations to conduct the experiment, and all the simulation nodes concluded in the network topology are routing nodes, whose exact numbers are 20, 30, and 40. The consumption of resources in physical servers is demonstrated by the utilization that simulation service programs are in CPU and memory of physical servers.

Results and Analysis. With those two mapping strategies, we map the simulation nodes with different amounts to the physical servers. And till it achieves stability to operate the simulation nodes, we compute the CPU and memory utilizations whose results are shown in Table. 2-1.

Table 2-1 CPU and memory utilizations in physical servers for two mapping strategies

Simulation Nodes	Random Mapping		Mapping of Topology Partitioning	
	CPU Utilization	Memory Utilization	CPU Utilization	Memory Utilization
20	34.45%	14.54%	24.53%	12.72%
30	45.76%	14.98%	31.51%	12.98%
40	57.68%	15.34%	47.76%	13.82%

In addition, we contrast the utilization of CPU with that of memory in physical servers with those two strategies, which are presented in Fig. 2-1 and Fig. 2-2.

From Table. 2-1, Fig. 2-1 and Fig. 2-2, we can draw the conclusion that, in the same experimental environment and scale, with the mapping strategies of simulation nodes based on topology partitioning, the simulation service programs take up fewer CPU and memory resources of physical servers than with the random mapping strategies.

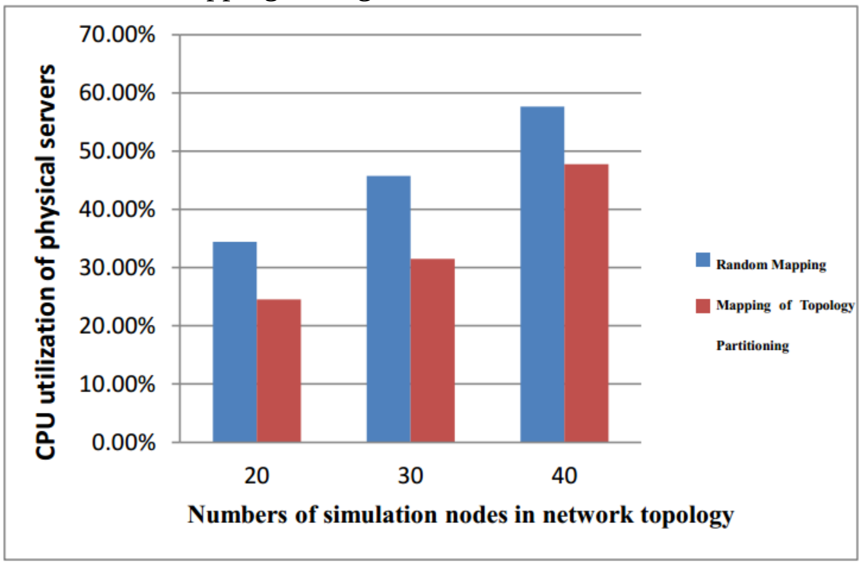


Fig. 2-1 CPU utilizations of simulation nodes in physical server for two mapping strategies

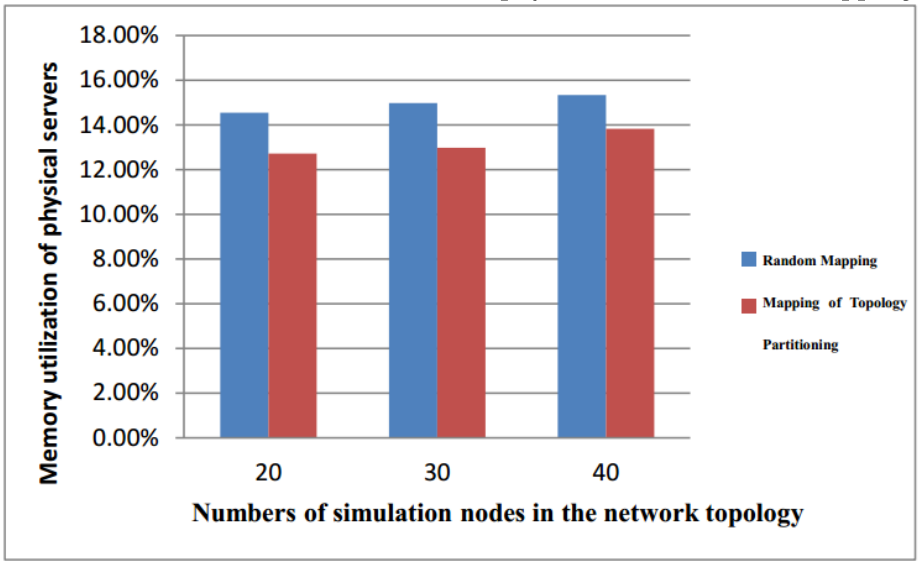


Fig. 2-2 Memory utilizations of simulation nodes in physical servers for two mapping strategies

Conclusion

Considering the degree of core for nodes and links in the network, and combined with Metis tool, in this paper we will testify the mapping strategies of simulation nodes based on topology partitioning. By the experiment, it results that with this strategies, the load balance among physical servers can be achieved, what’s more, the traffic of simulation nodes crossing several physical servers is largely reduced and thus it raises the efficiency of network simulation, because it can map those closely related simulation nodes to the same server as far as possible. Therefore, it contributes to improve the resources utilization in distributed simulation system. Furthur, combining this mapping strategies with the distributed system and taking other related factors into account can lead to the automation of mapping simulation nodes for higher efficient operation of distributed simulation system.

Acknowledgment

This work is supported by *National Natural Science Foundation of China Project* (61302087, 61401038).

References

[1] M.Yu, Y.Yi, J. Rexford, M.Chiang. Rethinking virtual network embedding: Substrate support for path splitting and migration. ACM SICCOMM Computer Communication Review, 2008, 38 (2), pp. 17-29.

[2] Ricci,R. Alfeld,C. & J.Lepreau. A Solver for the Network Testbed Mapping Problem [J]. University of Utah Flux Group Technical Note 2002-05.

[3] J. Fan, M.Ammar. Dynamic topology configuration in service overlay networks: A study of reconfiguration Policies. IEEE. INFOCOM, 2006: 1-12.

[4] Amine Abou-Rjeili, George Karypis.Multilevel algorithms for partitioning power-law graphs[R].American:Department of Computer Science and Engineering, University of Minnesota, Army HPC Research Center and Digital Technology Center, 2005.

[5] Freeman L C. Centrality in Social Networks: Conceptual Clarification [J]. Social Networks, 1979, 1: 215- 239.

[6] Brandes U. A Faster Algorithm for Betweenness Centrality [J]. Journal of Mathematical Sociology, 2001, 25: 163-177.

[7] G. Karypis and V. Kumar. Parallel Multilevel K-Way Partitioning Scheme for Irregular Graphs. Technical Report TR 96-036, Department of Computer Science, University of Minnesota, 1996.

[8] G. Karypis and V. Kumar. A Fast and Highly Quality Multilevel Scheme for Partitioning Irregular Graphs. SIAM Journal on Scientific Computing, 1998 (To Appear).

[9] Karypis G, Kumar V Metis manual [EB/OL]. http://glaros. dtc.unto. edu/ gkhome /metis/ Metis/download.