Int. J. Communications, Network and System Sciences, 2017, 10, 264-273
http://www.scirp.org/journal/ijcns
ISSN Online: 1913-3723
ISSN Print: 1913-3715

Distributed Middlebox Placement Based on Potential Game

Yongwen Li, Zhiyi Qu

School of Information Science & Engineering, Lanzhou University, Lanzhou, China
Email: liyw14@lzu.edu.cn, quzy@lzu.edu.cn

How to cite this paper: Li, Y.W. and Qu, Z.Y. (2017) Distributed Middlebox Placement Based on Potential Game. *Int J Communications, Network and System Sciences*, **10**, 264-273.
https://doi.org/10.4236/ijcns.2017.105B026

Received: May 3, 2017
Accepted: May 23, 2017
Published: May 26, 2017

Abstract

In this paper, we analyse the deployment of middlebox. For a given network information and policy requirements, an attempt is made to determine the optimal location of middlebox to achieve the best performance. In terms of the end-to-end delay as a performance optimization index, a distributed middlebox placement algorithm based on potential game is proposed. Through extensive simulations, it demonstrates that the proposed algorithm achieves the near-optimal solution, and the end-to-end delay decreases significantly.

Keywords

Middlebox Placement, Performance Optimization, Potential Game, Matching Graph, Distributed Algorithm

1. Introduction

In order to manage users' communication, improve safety and performance the operators widely deploy middlebox services in their own network, such as deep packet inspection (DPI), firewall, proxy, intrusion detection and prevention (IDP), network address conversion (NAT), etc. The service chain is defined as a middlebox sequence that should be traversed in a pre-specified order. Recently, the academic and industrial domains have done a lot of efforts on how to implement the service chain efficiently [1] [2] [3]. According to the predefined sequence of strategies, the routing is transmitted from one middlebox to another. However, no matter what program is used to achieve the service chain, there will be such a problem that in the network where the deployment of middlebox is placed in order to make the performance of the overall strategy for the implementation of overall strategy.

This paper mainly solves the problem of middlebox deployment. In particular, according to the traversal sequence of the policy, we aim to find the best location

of the deployment of middlebox. Instead of the shortest path passing from the entry point to the exit point, in which traffic route from the entrance to a number of middlebox one by one, and then exit the outlet. Obviously, the overall path will be exaggerated, which will result in a long end-to-end delay. Therefore, in this work, we consider the total end-to-end delay as the optimization index of the policy implementation, which is also adopted in [4], and use this optimization index to evaluate the performance of proposed middlebox deployment scheme. In the network function virtualization (NFV) [5], the problem is more meaningful since you can easily move one form of middlebox software running on commodity hardware. In NFV, you can use the deployment algorithm more often to determine the best location of middlebox periodically.

In this paper, a distributed middlebox placement based on potential game is proposed to decrease computation complexity. Through the definition of the service queue middlebox for the participants in the game, to minimize the total end-to-end delay as the goal, we design the game strategy and utility function, give the potential game proved that, and according to the definition of ordinal function and analysis of change, it can be proofed that Nash equilibrium point exists in the model. Based on this potential game model, a distributed algorithm is proposed. Through numerical results, the proposed distributed algorithm can reduce the system end-to-end delay significantly compared with the random placement. On average, the end-to-end delay will be reduced by 34%. In addition, it shows that the performance of the proposed distributed algorithm obtains the near-optimal solution.

2. Methodology

2.1. System Model

These $G=(S,E)$ is utilized to express underlying network, where S denotes the switch set, E denotes the link set. For each switch $s_l \in S$, $C(s_l)$ represents the available resource of switch s_l in deploying middlebox, and resource represents the number of available ports in server or the computing resources of the server. In addition, $d^p_{l_1,l_2}$ represents the delay from route s_1 to route s_2, that is the total delay of all links on the route. Then, a format model is given to describe the strategy. The P is used to describe provision set of policy, the provision of each policy $p_k \in P$, which is defined as $p_k=\{i_k, m_k^1, m_k^2, \cdots, m_k^{n_k}, e_k\}$ where i_k and e_k denotes input switch and output switch, respectively, m_k^i represents the data stream of this policy adopted by the *i-th* middlebox, and n_k represents the number of middlebox of this policy. Further, we use Q to represent the set of middleboxes to be deployed, and $R(q_i)$ to represent the required resource to deploy middlebox $q_i \in Q$.

2.2. Problem Formation

From the above definition, we need to determine the location of each middlebox. $x_{i,l}$ is utilized to represent the deployment scenario of the middlebox. When $x_{i,l}=1$, it represents that middlebox q_i connected to switch s_l. Otherwise,

$x_{i,l} = 0$. Then, the following constraints are obtained.

$$\sum_{\forall s_l \in S} x_{i,l} = 1, \forall q_i \in Q. \tag{1}$$

$$\sum_{\forall q_i \in Q} R(q_i) x_{i,l} \le C(s_l), \forall s_l \in S. \tag{2}$$

$$x_{i,l} = 0, \forall q_i \in Q, \forall s_l \in S \setminus S_i. \tag{3}$$

Condition (1) ensures that each middlebox is only deployed in a switch condition (2) guarantees that the total resource requirements in a local deployment middlebox do not exceed resource performance. Condition (3) indicates that middlebox with special requirements can only be deployed in a specific area.

a) Induction problem using matching graph

The main problem of this paper is how to deploy the middlebox to minimize the system delay. This paper mainly uses the matching graph to explain how to deploy middleboxed, as shown in **Figure 1**. Where ω_{q_i,s_j} represents the weight of middlebox q_i located in switch s_j, which is defined as sum link delay from the middlebox q_i to the next hop. When the last hop of middlebox q_i is the end node, the ω_{q_i,s_j} contains the link delay of starting node to q_i. From the above definition, the overall delay of the system is:

$$D^{tot} = \sum_{q_i \in Q} \omega_{q_i,s_j}, \tag{4}$$

where the weight ω_{q_i,s_j} is related to the matching of other middleboxes, which are determined by the decision variables $x_{i,l}$.

b) Analysis of the reasons for the problem difficulty

From the above analysis based on matching graph, we can observe that if the matching position of any middlebox q_i changes, and the weight value of other middleboxes may vary with unchanged matching position. The edge weight of the matching graph is influenced by each other, thus it becomes very difficult to obtain the maximum matching problem as shown in **Figure 1**. Furthermore, we can see that if middlebx q_i changes its matching position, then the weight of edges, whose next hop is q_i, also changes. Therefore, we modify the definition of weights, and use the growth trend to design distributed algorithm.

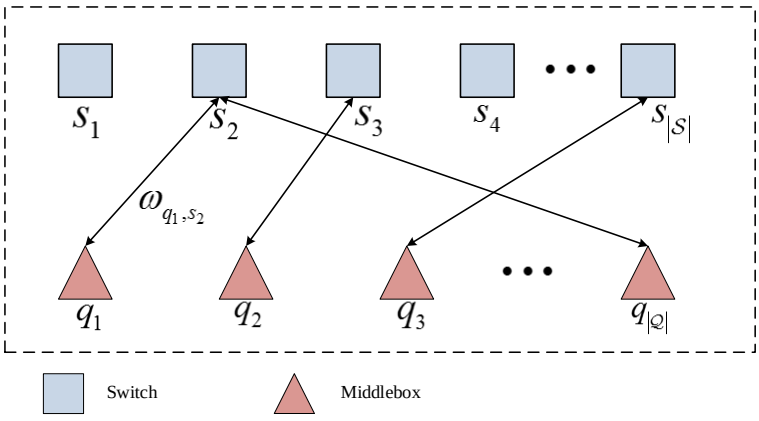

Figure 1. Problem model with matching graph.

2.3. Definition of Potential Game

Potential game (PG) [6] is an effective model for the change of the trend, and the most important one is the definition of utility, the effective policy space and the definition of the optimal policy.

1) According to the above analysis, the game model is defined as $\mathcal{G}=(\mathcal{N},\mathcal{X},\mathcal{U})$, where $\mathcal{N}$ represents the number of game players, $\mathcal{X}$ represents strategy space, and $\mathcal{U}$ represents network performance of any player. From the above analysis, the utility of middlebox q_i is related to the start edge and the weight of the next hop edge, so the utility of any player is defined as:

$$\mathcal{U}_{q_i}=\sum_{s_l\in\mathcal{S}}x_{i,l}\omega_{q_i,s_l}+\sum_{p_k\in\mathcal{P}}\sum_{q_j\in\mathcal{Q}}I_{p_k,q_j}d_{p_k,q_j}, \tag{5}$$

Compared with ω_{q_i,s_j}, this definition increases the weight of the edge with the next hop to the middlebox. Here $I_{a,b}$ is the indicative symbol that if $a=b$, $I_{a,b}=1$, or $I_{a,b}=0$.

2) It is clear that the feasible strategy space of any middlebox q_i is to select some switch, which also meets the constraint of the switch port capacity.

Definition 1: The optimal strategy of the participants. The optimal policy for any middlebox q_i is the minimum utility that can be obtained in a feasible space when the location of the other middlebox is not changed, that is

$$\max_{x_n\in\mathcal{X}}U_n(x_n,x_{-n}),\forall n\in\mathcal{N}, \tag{6}$$

where n represents middlebox, x_n represents strategy of n, x_{-n} represents the strategy of other middlebox except n. The above formula represents the best strategy for the current middlebox, which is based on the same location of the other middlebox.

3) According to the definition of the above utility function and the strategy space, we also need to prove that this game model is a potential game.

Theorem 1: *The game model defined as* $\mathcal{G}=(\mathcal{N},\mathcal{X},\mathcal{U})$ *is a potential game.*

Proof: *We* first define the ordinal function as $F(x)=\sum_{q_i\in\mathcal{Q}}\sum_{s_l\in\mathcal{S}}x_{i,l}\omega_{q_i,s_l}$. Then, we prove that any varying in the layout of the middlebox will bring a change in its own utility function and change the value of ordinal function with the same trend. Assuming that the layout of a middlebox q_i has changed, from the switch s_l to switch $\tilde{s}_l$, the utility function of the middlebox varys as follows:

$$\begin{aligned}
&\mathcal{U}_{q_i}-\tilde{\mathcal{U}}_{q_i}=\sum_{s_l\in\mathcal{S}}x_{i,l}\omega_{q_i,s_l}+\sum_{p_k\in\mathcal{P}}\sum_{q_j\in\mathcal{Q}}I_{p_k,q_j}d_{p_k,q_j}\\
&-\sum_{s_l\in\mathcal{S}}\tilde{x}_{i,l}\tilde{\omega}_{q_i,s_l}-\sum_{p_k\in\mathcal{P}}\sum_{q_j\in\mathcal{Q}}I_{p_k,q_j}\tilde{d}_{p_k,q_j}\\
&=\underbrace{\sum_{s_l\in\mathcal{S}}x_{i,l}\omega_{q_i,s_l}-\sum_{s_l\in\mathcal{S}}\tilde{x}_{i,l}\tilde{\omega}_{q_i,s_l}}_{\Delta U_1}\\
&+\underbrace{\sum_{p_k\in\mathcal{P}}\sum_{q_j\in\mathcal{Q}}I_{p_k,q_j}d_{p_k,q_j}-\sum_{p_k\in\mathcal{P}}\sum_{q_j\in\mathcal{Q}}I_{p_k,q_j}\tilde{d}_{p_k,q_j}}_{\Delta U_2}
\end{aligned} \tag{7}$$

where ΔU_1 represents the delay varying of the edge with q_i as the starting point and ΔU_2 represents the delay varying of the edge with q_i as the next hop. Similarly, the ordinal function of the change is shown below:

$$\begin{aligned} F(x)-\tilde{F}(x) &= \sum_{q_i\in\mathcal{Q}}\sum_{s_l\in\mathcal{S}} x_{i,l}\omega_{q_i,s_l} - \sum_{q_i\in\mathcal{Q}}\sum_{s_l\in\mathcal{S}} \tilde{x}_{i,l}\tilde{\omega}_{q_i,s_l} \\ &= \underbrace{\sum_{s_l\in\mathcal{S}} x_{i,l}\omega_{q_i,s_l} - \sum_{s_l\in\mathcal{S}} \tilde{x}_{i,l}\tilde{\omega}_{q_i,s_l}}_{\Delta U_1} + \Delta F \end{aligned} \tag{8}$$

where ΔF_{22} indicates the edge delay of next hop as q_i for all of the strategies, as the delay of other edge independent of q_i will not change, $\Delta F_2 = \Delta U_2$. Therefore, it is proved that the game model defined as $\mathcal{G} = (\mathcal{N}, \mathcal{X}, \mathcal{U})$ is a potential game. According to the above definition, it can be concluded that this model exists Nash equilibrium.

2.4. Distributed Algorithm

Based on the above potential game, it can be seen that any middlebox can choose its best strategy to reduce the system delay. The specific algorithm can be depicted extensivley as follows (**Algorithm 1**):

In the algorithm, each middlebox selects its best strategy in a random order. In each iteration, the system uniformly randomly chooses one middlebox $q \in \mathcal{Q}$, the selected middlebox obtains its best strategy from Definition 1. If no new strategy is obtained from all middlebox, the stop flag is set to be 0, and the Algorithm 1 ends.

From the analysis of these algorithms, it can be seen that any middlebox can reduce system latency only considering its utility, therefore, saving the overall

---

Input: The capacity of switches, middlebox and traffic.
1) **Initialization:**
2) Select the feasible strategy randomly;
3) Calculate system delay, *Stopflag* = 1;
4) **WHILE** *Stopflag* **DO**
5) **FOR** $q \in \mathcal{Q}$ **DO**
6) Select its optimal strategy;
7) **IF** $\mathcal{U}_q - \tilde{\mathcal{U}}_q > 0$
8) Update the strategy of q and its corresponding system delay.
9) **ELSE**
10) Select middlebox randomly;
11) **END IF**
12) **END FOR**
13) If no middlebox changes its strategy,
14) Set *Stopflag* = 1;
15) **END WHILE**
Output: System delay and its corresponding strategy.

Algorithm 1. Distributed algorithm based on potential game.

optimization of the system overhead, and overcome the convergence problem of heuristic algorithm (such as reference in the literature of the simulated annealing algorithm [4]).

2.5. Proof of Convergence

Theorem: *the proposed distributed algorithm certainly converges to the Nash equilibrium point.*

Proof: Each a iteration of the algorithm generates a new strategy by adopting the best response strategy. Since there are only a limited number of middlebox, the maximum number of policies for each middlebox is limited. Therefore, the system can achieve the final strategy by finite iteration with probability 1.

3. Results

In this section, we evaluate the performance of the proposed algorithm for middlebox layout. In the evaluation, we introduce random layout for comparison. Random placement does not consider the impact of middlebox placement on system delay, and only guarantees that the constraint conditions are satisfied. In the design of network topology and service chain strategy, the previous works [2] [7] used well known topology (such as Abilene and FatTree) in strategy execution due to the lack of openly available information. In this paper, we choose the Abilene [8] network as the reference; Abilene is the core network of Internet 2, which is the irregular topology of the network (like most ISP network). The network has 11 nodes, 14 links. We adopted the same approach as the literature [2] and [7] to generate policy rules. Specifically, we assume that there are diffeent number of applications, and the traffic flow required for each application is required through multiple middlebox. We gave each application a random assignment of a middlebox sequence. Then, a number of applications are randomly selected for the traffic flow between the two switches to generate the policy requirements. We distributed the link delay according to the uniform distribution. Based on the link delay, the network controller will select the shortest path passing from one switch to another through route.

In the evaluation, we generated a total of 400 simulation scenarios. In each scenario, we randomly selected the value of each parameter from the parameter set in **Table 1**, and generated the network settings and policies. The cumulative distribution function (CDF) associated with the optimal value is as shown in **Figure 2**. As shown in the graph, the performance gap between the distributed algorithm and the optimal solution is very small. Specifically, the performance gap of the distributed algorithm for 92.5% of the simulated scenarios is less than 20%. In addition, there is a large gap between the distributed algorithm and random layout, which shows that the proposed algorithm greatly improves the performance of the strategy. On average, the distributed algorithm reduces the end-to-end delay of 34% compared to random placement.

In order to further investigate the effect of different layout schemes on the performance, we studied the distribution of end-to-end delay of each strategy.

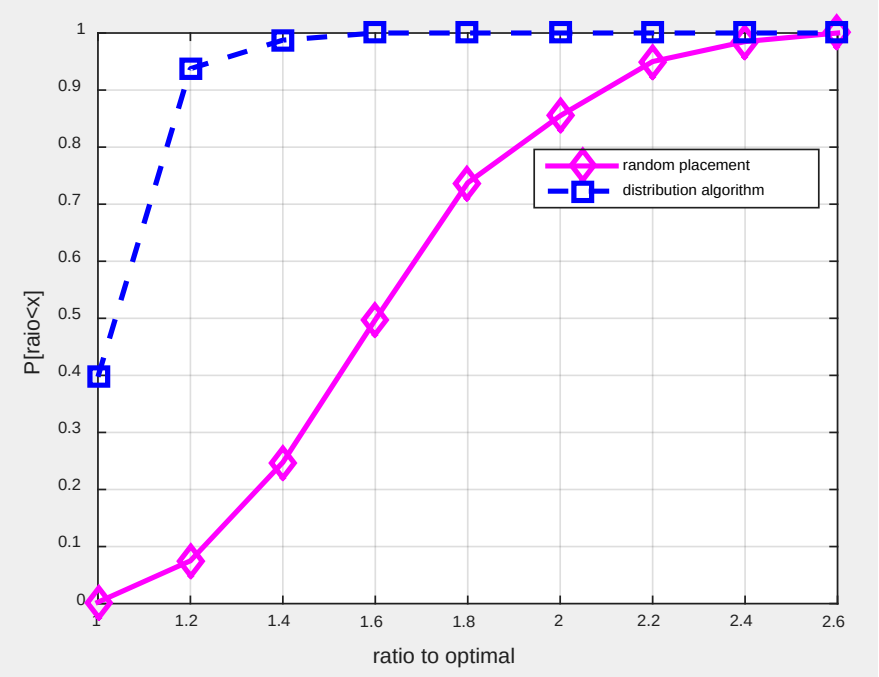

Figure 2. CDF of end-to-end delay with different placement over optimal results.

Table 1. Parameters settings in the performance evaluation.

Parameter	Distribution	Mean	Var
Link delay(ms)	Uniform	1, 1.5, 2, 2.5	0.4, 0.5, 0.6, 0.7
Available switch ports	1, 2, 3	N/A	N/A
The number of middleboxes to deplay	7, 8, 9, 10	N/A	N/A
The number of middleboxes in each policy	3, 4, 5, 6	N/A	N/A
The number of total applications	5, 10, 15	N/A	N/A

The results of a scenario are shown in **Figure 3**. These results clearly indicate that the layout scheme has a significant impact on the system performance. Middlebox placement by proposed distribution algorithm can obtain near optimal performance, about 70% of end-to-end delay of strategy is less than 100 ms. However, Middlebox deployment according to random arrangement is only 25%. In the schemes generated by distribution algorithm, 100% of the delay of the strategy is less than 150 ms, and there is only 65% in the schemes generated by randomly placement.

The performance results for each switch with the number of available ports varying are described in **Figure 4**. In this simulation, each policy consists of 5 middlebox. As shown in **Figure 4**, the performance is improved with the incensement of the number of available ports. This is because those more available ports provide more feasible strategy for each middlebox. We observe that that the distributed algorithm has can utilize the increased resources more efficiently. **Figure 5** provides a comparison of computation complexity between the distributed algorithm and the optimal placement algorithm. It can be clearly ob-

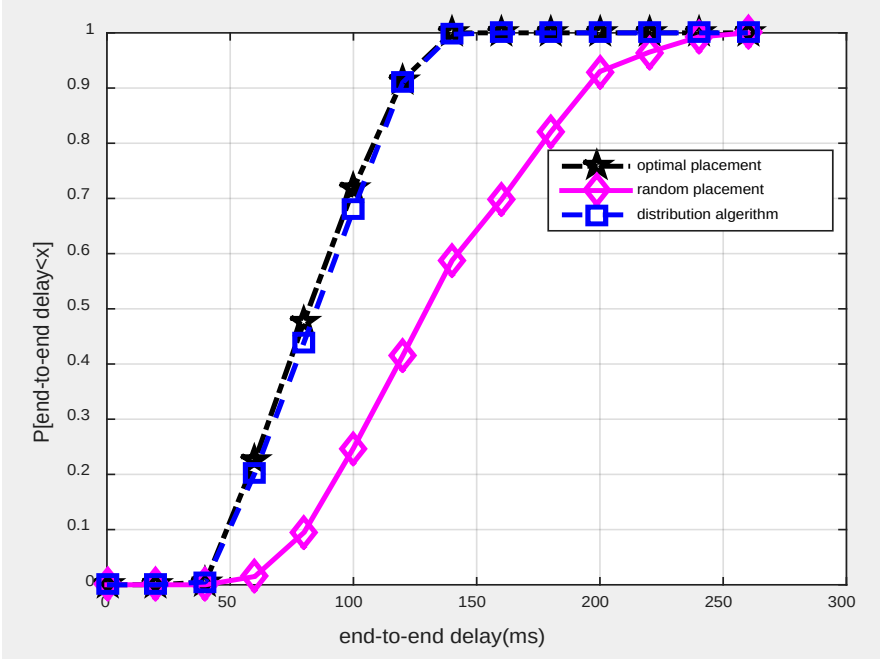

Figure 3. CDF of end-to-end delay of each policy.

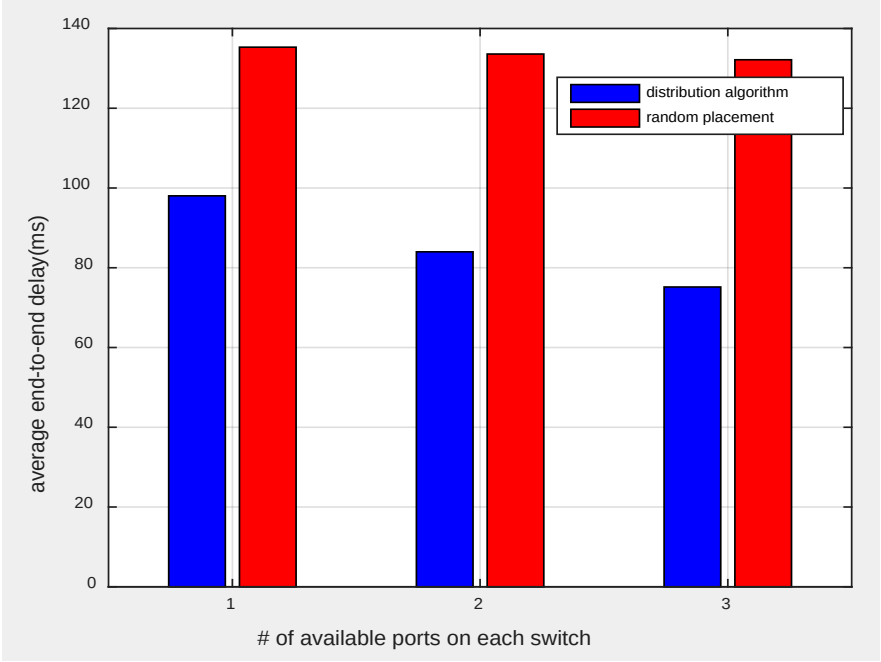

Figure 4. Average end-to-end delay varying the number of available ports on each switch.

served that the distributed algorithm reduces the computation complexity significantly from two aspects. One is to reduce the number of overall iterations, the other is to share the computing tasks among all switch.

4. Conclusion

In this paper, we mainly analyze the optimization of middlebox deployment to minimize the end-to-end delay. The matching graph was utilized to formulate

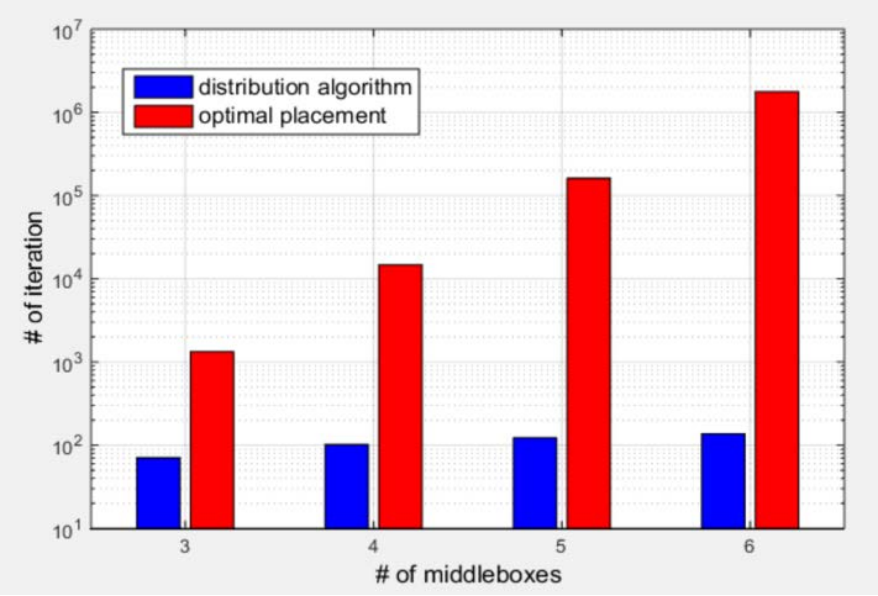

Figure 5. The number of iteration when varying the number of middleboxes.

the problem, which is proved as a potential game and exists the Nash equilibrium. Thus, a distributed algorithm is proposed based on potential game. Through extensive simulations, it is proved that the proposed algorithm can reduce the end-to-end delay effectively and obtain the near-optimal solution.

Acknowledgements

The authors would like to acknowledge all the members that participation in the exhaustive field measurement campaign for their valuable effort. Thanks also anonymous reviewers for their perspicacious comments.

References

[1] Grillo, G., Joseph, D.A., Tavakoli, A. and Stoica, I. (2008) A Policy-Aware Switching Layer for Data Centers. *Proc. ACM SIGCOMM*, 51-62. https://doi.org/10.1145/1402958.1402966

[2] Qazi, Z.A., Tu, C.-C., Chiang, L., Miao, R., Sekar, V. and Yu, M. (2013) Simple-Fying Middlebox Policy Enforcement Using Sdn. *Proc. ACM SIGCOMM*, 27-38. https://doi.org/10.1145/2534169.2486022

[3] Zhang, Y., Beheshti, N., *et al.* (2013) Steering: A Soft- ware-Defined Networking for Inline Service Chaining. *Proc. IEEE ICNP*, 1-10. https://doi.org/10.1109/icnp.2013.6733615

[4] Liu, J., Li, Y., Zhang, Y., *et al.* (1939) Improve Service Chaining Performance with Optimized Middlebox Placement. *IEEE Transactions on Services Computing*, 1-1.

[5] Sdn and Openflow World Congress Introductory White Paper, Network Functions Virtualisation. https://portal.etsi.org/NFV/NFVWhitePaper.pdf,2012.

[6] Hu, X.H., Gao, H.W., Wang, D.Y., Li, Y.M. and Ji, Z.H. (2013) Two Classes of Potential Games and the Solving Method of the Equilibria. Information Engineering Research Institute, USA. *Proceedings of* 2013 *International Conference on Intelligent Materials and Mechatronics* (*IMM* 2013). Information Engineering Research Institute, 5.

[7] Sekar, V., Egi, N., Ratnasamy, S., Reiter, M.K. and Shi, G. (2012) Design and Implementation of a Consolidated Middlebox Architecture. *Proc. USENIX NSDI*, 323-336.

[8] Abilene Core Topology. https://itservices.stanfo-rd.edu/service/network/internet2/abilene