On the Placement of Management and Control Functionality in Software Defined Networks

Daphne Tuncer, Marinos Charalambides, Stuart Clayman, and George Pavlou
Department of Electronic and Electrical Engineering
University College London
Email: {d.tuncer, m.charalambides, s.clayman, g.pavlou}@ee.ucl.ac.uk

***Abstract*—In order to support reactive and adaptive operations, Software-Defined Networking (SDN)-based management and control frameworks call for decentralized solutions. A key challenge to consider when deploying such solutions is to decide on the degree of distribution of the management and control functionality. In this paper, we develop an approach to determine the allocation of management and control entities by designing two algorithms to compute their placement. The algorithms rely on a set of input parameters which can be tuned to take into account the requirements of both the network infrastructure and the management applications to execute in the network. We evaluate the influence of these parameters on the configuration of the resulting management and control planes based on real network topologies and provide guidelines regarding the settings of the proposed algorithms.**

I. Introduction

Over the past few years, the Software-Defined Networking (SDN) paradigm has attracted a lot of interest from both the industry and the research community, who envision SDN-based solutions as a key enabler towards the simplification of the management of today's network infrastructures.

The main principle of SDN lies in the decoupling of network control from forwarding hardware [1]. In the SDN architecture, control functions are moved away from the network devices, which are represented as basic forwarding elements, towards external dedicated software-based components, referred to as the controllers, forming a unified control platform [2]. This can be seen as a logically centralized control plane which operates on a global network view and which implements a range of functions. The controllers interact with the network resources via a standardized interface which is used to collect network state information and distribute control commands to be enforced in the network devices.

A direct consequence of the migration of control functions from network elements to remote controllers is the risk of creating new bottlenecks and potentially significant overhead, depending on the type of management applications considered [3]. While using a centralized controller with a network-wide view has the benefit of facilitating the implementation of the control logic, it also presents limitations, especially in terms of scalability as the size and dynamics of the network increase. In addition, resource management in fixed networks is usually performed by external offline centralized systems, which optimize network performance over long timescales. While they are well-suited for supporting network operations that do not require frequent reconfigurations, centralized/offline approaches are not adequate for applications that adapt to traffic and network dynamics (e.g. online traffic engineering). To overcome these limitations, dynamic management applications should rely on a distributed framework [4] [5]. However, the implementation of decentralized management and control functionality raises several challenges, as the decision on how to distribute the managers and controllers can be driven by several factors, such as the characteristics of the underlying physical infrastructure and/or the type of applications considered.

In this paper, we extend our previous work on the placement of management functionality in the context of the SDN-based resource management framework presented in [5]. In the proposed framework, management operations are realized through a set of local managers (LMs), which implement the logic of management applications. The LMs are responsible for computing the configuration of a set of network resources under their supervision according to the objective of the applications which they implement. In [5], we designed a placement algorithm to decide on the allocation of LMs (number and location) with the objective of minimizing the average distance between the network devices and the LMs. While focusing on minimizing the distance is relevant for applications sensitive to delay, this may lead to an unbalanced allocation of LMs to network devices (i.e. unbalanced number of devices per LM). As such, this may not be appropriate for applications with stringent requirements in terms of volume of information to maintain and process (e.g. cache management).

This paper extends our previous work in two main directions. We first propose two variants of the placement heuristic presented in [5] that can cater for the requirements of applications with stringent needs in terms of balanced cluster size and evaluate of their performance with respect to different parameters. In addition, we further elaborate on and consolidate the preliminary results reported in [5] by evaluating the performance of the proposed algorithms based on a wider range of parameters (types and values). Based on the results, we refine the recommendations reported in [5] and provide guidelines that can be used to decide how to configure the associated algorithms.

The remainder of this paper is organized as follows. Section II briefly describes the management framework developed in our previous work and discusses related work. Section III presents the proposed placement approach and algorithms. The

978-3-901882-77-7

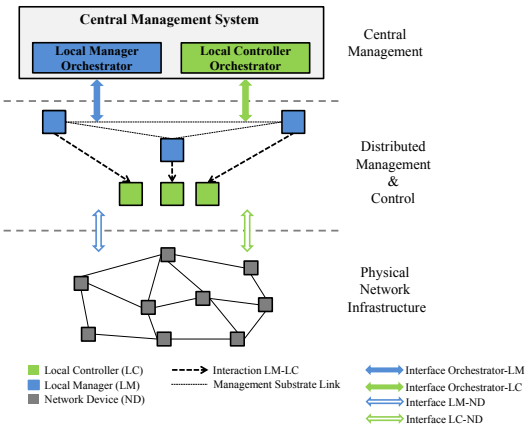


Fig. 1. Proposed framework.

evaluation of the different heuristics is presented in Section IV. Finally, we discuss the obtained results and provide a summary of the paper in Section V.

II. Background and Related Work

A. Management and Control Framework

In [5], we developed a SDN-based solution for network resource management and control that supports both static and dynamic resource management applications in fixed backbone infrastructures. The proposed framework follows a layered architecture as depicted in Fig. 1. The bottom layer represents the underlying network infrastructure. The middle layer implements distributed management and control functionality. Finally, the top layer represents the central management system. As shown in Fig. 1, the components of the architecture are implemented in a modular fashion. This allows a clear separation between the management logic and the control logic, on one hand, and between short to medium term and long term management operations, on the other hand.

More specifically, the network infrastructure is managed and controlled by a set of software-based Local Managers (LMs) and Local Controllers (LCs), forming the distributed management and control planes, respectively. The LMs implement the logic of management applications which are executed at short to medium timescales (e.g. traffic engineering), and are responsible for making decisions regarding the settings of network parameters for the set of resources under their responsibility. Configuration decisions taken by LMs are provided to the LCs, which define and plan the sequence of actions to be enforced for updating the network parameters. These actions are then translated to instructions sent to and executed by the relevant network devices. The centralized management system is responsible for longer term operations, for example those that pertain to the life cycle of LMs and LCs. In particular, they are used to determine the number of LMs and LCs to deploy, their location, as well as their zone of responsibility.

B. Related Work

In order to overcome the limitations of the single centralized controller model, various approaches have been proposed in the literature. These mainly fall into two categories: fully distributed architectures (e.g. [2] [6]) and hierarchical models (e.g. [7] [8] [9]).

A key issue when designing a distributed management and control approach is to decide on the placement of the relevant functionality. The problem of the allocation of network functions has been investigated in a wide range of contexts, ranging from the placement of monitoring agents (e.g. [10]) to the selection of network service gateways (e.g. [11]). In the context of SDN, Heller et al. argue in [12] that one of the key parameters to consider for large networks is the propagation delay between the controller(s) and the network devices. In [13], the authors have proposed an approach to dynamically determine the number and the location of controllers based on the network conditions. A different objective has been considered in [14] where the goal of the placement is to maximize the reliability in terms of control paths. Their approach assumes that the number of controllers to deploy is given, which may not be easy to determine *a priori* and is considered as a variable in our work. The placement problem has also been tackled from the perspective of fault tolerant SDNs in [15], where authors have focused on an approach to achieve at least five nines reliability in the southbound interface between the controllers and the nodes. Finally, the controller placement problem has recently been modeled in [16] as a multi-objective optimization problem. This presupposes that the requirements of each application can be precisely identified, which is not trivial.

III. Placement Algorithms

A. Manager/Controller Placement Problem

A key challenge when deploying a distributed management and control infrastructure is to determine the degree of distribution of the decision-making points (i.e. the LMs[1]), which can depend both on the physical infrastructure as well as the type of management applications to consider. On one hand, increasing the degree of distribution has the benefits of (i) decreasing the delay of communication between the network devices and the LMs/LCs and (ii) improving the level of redundancy. On the other hand, this may come with a cost in terms of communication overhead (both at the intra and inter layers level) and as such, may compromise the scalability of the framework. In [5], we developed an algorithm to determine the placement of functionality in the proposed distributed management and control layer, which applies to scenario in which the structure of the management entities does not change frequently. Given a network topology, the objective of the proposed approach is to compute the number of LMs to deploy, their location, as well as the devices these are connected to, with the objective of minimizing the distance (in terms of hop count) between the network devices and the LMs.

The proposed algorithm follows a greedy approach so that LMs are iteratively added in the network one-by-one. The

[1]In this paper, we assume that there is a one-to-one mapping between the LMs and the LCs.

Pseudo-code $P_{distance}$ Algorithm
Inputs: Set of nodes; Terminating condition.
0. Select initial LM location.
1. Compute Pressure score of all nodes not already selected as a LM location.
2. Select the node with the highest score.
3. Check if the selection satisfies the terminating condition.
if it is satisfied **then**
End algorithm.
else
Add selected node to the list of LM locations and go to step 1.
end if
Outputs: Set of LM locations; Switch-LM mapping.

Fig. 2. Pseudo-code of the $P_{distance}$ algorithm (from [5]).

TABLE I
NETWORK CHARACTERISTICS.

Network	# Nodes	# Bidirectional Links
Abilene [17]	12	15
Geant [18]	23	37
Germany50 [19]	50	88
Deltacom [20]	92	129

TABLE II
AVERAGE UNBALANCE DEGREE.

	Abilene	Geant	Germany50	Deltacom
δ	1.75	3.87	8.32	13.87

decision on to which node to attach the next LM is based on a metric, called *Pressure* score [10], defined for each node, which measures the benefits of selecting the node as a LM location in terms of average LM-node distance reduction. More specifically, the *Pressure* score of node i is defined as:

$$P(i) = \sum_{j \in \mathcal{N}} \max(0, l_j - d_{i,j}) \quad (1)$$

For all j in $\mathcal{N}$, l_j represents the distance between node j and the LM to which it is currently connected and for all i and j in $\mathcal{N}$, $d_{i,j}$ is the distance from node i to node j. The algorithm relies on two tunable parameters which control its initialization and termination steps. In particular, the terminating condition is represented by an ending threshold which is based on a measure of the average distance reduction at each addition of an extra LM. We refer to this version of the placement algorithm as $P_{distance}$. The pseudo-code of the $P_{distance}$ algorithm is presented in Fig. 2.

In practice, the set of devices attached to a LM (i.e. cluster) can affect the volume of information which needs to be maintained and processed by the LM. Given that the main objective of algorithm $P_{distance}$ is to minimize the average distance between the LMs and the network resources, it may lead to the formation of non-homogeneous clusters. In the following subsections, we propose two variants of the algorithm to take into account constraints in terms of cluster balance degree.

B. Placement Algorithm - $P_{cluster}$

To investigate the effect of algorithm $P_{distance}$ on the cluster size distribution, we compute, for the four networks presented in Table I, the value δ of the difference in terms of size (i.e. number of nodes) between the largest and the smallest clusters. δ represents the unbalance degree of the cluster size. The values obtained with a distance reduction improvement threshold equal to 5% are reported in Table II. As can be noted, the algorithm can lead to the formation of unbalanced clusters. In particular, the degree of unbalance tends to increase as the number of nodes in the network increases.

In order to take into account the trade-off between latency reduction and homogeneity of the cluster size distribution, we design a second version of the placement algorithm. In the new algorithm, called Placement Algorithm $P_{cluster}$, two conditions are considered to decide whether the algorithm should terminate. The first condition is similar to the one implemented in algorithm $P_{distance}$ and is represented by the distance reduction improvement threshold. The second condition is about the unbalance degree of the cluster size. More specifically, at each step, the algorithm evaluates the value of δ and compares it to an input unbalance degree threshold, which defines the maximum authorized difference between the number of nodes in the largest and smallest clusters. The algorithm terminates if δ is lower than the input threshold. The influence of the threshold on the output configuration is discussed in detail in Section IV-C.

C. Placement Algorithm - P_{weight}

Algorithm $P_{cluster}$ relies on the implicit assumption that the volume of information to maintain and process depends on the number of nodes under the supervision of a LM. However, there exist management applications for which the input data associated with each network node depends on the characteristics of the node itself (e.g. geographical location). In this case, the cost imposed to an LM is not driven by the number of nodes *per se* but by the actual set of nodes. Cache management applications are examples of this type of applications (e.g. [21]). In this example, the complexity of the decision-making process is directly affected by the number of requested content items, which depends the location of the nodes.

To account for these requirements, we modify the *Pressure* score presented in Eq. 1 by considering a weighted sum of the distance reduction, i.e.

$$P(i) = \sum_{j \in \mathcal{N}} w_j \cdot \max(0, l_j - d_{i,j}) \quad (2)$$

where w_j represents the weight associated with node j. In practice, the weight of a node needs to be computed in accordance with the characteristics of the management application to consider. In the case of a cache management application, these can be defined, for instance according to the density of the region to which each node is mapped in the physical network infrastructure.

TABLE III
DISTRIBUTION OF THE NUMBER OF LOCAL MANAGERS.

Network	1^{st} quartile	Median	3^{rd} quartile
Abilene	4	4.5	8
Geant	5	6	7
Germany50	6	6	7
Deltacom	5	5.5	6

IV. EVALUATION

In this section, we investigate the influence of different parameters on the performance of the proposed placement algorithms based on the four topologies reported in Table I.

A. Influence of the Initialization Criteria

The proposed placement algorithms work by iteratively selecting the location on which LMs are deployed. The resulting configuration is therefore affected by the order according to which LM locations are selected and, in particular, by the choice of the first LM location. In Table III, we report the distribution of the number of LMs obtained by applying algorithm $\mathrm{P}_{\text{distance}}$ with a terminating threshold equal to 5%. A variation between the first, second (median) and third quartiles can be observed, which demonstrates that the algorithm is sensitive to the initial conditions.

To decide on the first LM location, we investigated in [5] the existence of a correlation between the average distance factor of the nodes and the number of selected LMs. Other topological metrics are also usually considered in the literature to characterize the location of the nodes in the network. In this paper, we take into account two additional important and well known graph theory metrics and discuss their incidence on the choice of the first LM location.

- **Betweenness centrality**: the betweenness centrality of a node i represents the number of times node i is in the shortest path between any two other network nodes. It is calculated as follows:

$$\forall i \in \mathcal{N},\ B(i) = \sum_{s,d \in \mathcal{N}\setminus\{i\}} \frac{\sigma_{sd}(i)}{\sigma_{sd}} \tag{3}$$

  where σ_{sd} is the number of paths from node s to node d and $\sigma_{sd}(i)$ the number of paths from s to d that go through i. The larger the value of $B(i)$, the larger the number of paths passing through node i.
- **Clustering coefficient**: the clustering coefficient of a node i quantifies the density of the connections in the neighborhood of node i. More specifically, it represents the total number of connections that exist between all the neighbors of node i. It is calculated as follows:

$$\forall i \in \mathcal{N},\ C(i) = \frac{2 \cdot Y(i)}{Z(i) \cdot (Z(i)-1)} \tag{4}$$

  where $Z(i)$ is the number of neighbors of node i and $Y(i)$ the number of links between the neighbors. The larger the value of $C(i)$, the denser the connections between the neighbors of i.
- **Average distance factor**: the average distance factor of a node i represents the proximity of a node in terms of average distance to the rest of the network nodes. It is calculated as follows:

$$\forall i \in \mathcal{N},\ D(i) = \frac{\sum_{j \in \mathcal{N}\setminus\{i\}} d_{i,j}}{|\mathcal{N}|^2} \tag{5}$$

  where $d_{i,j}$ is the distance in terms of hop count between node i and node j. The smaller the value of $D(i)$, the closer node i is, on average, to the other network nodes.

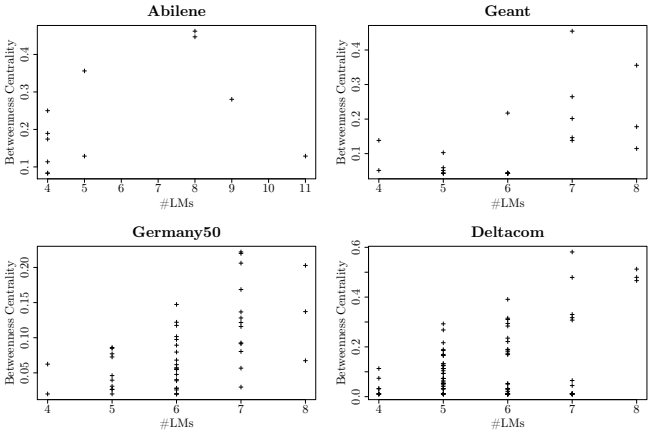


Fig. 3. Number of selected LMs vs. betweenness centrality.

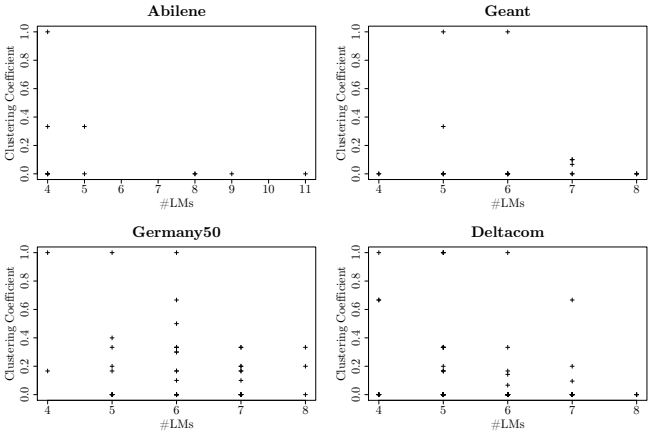


Fig. 4. Number of selected LMs vs. clustering coefficient.

For each network node i, we compute the values $B(i)$, $C(i)$ and $D(i)$ and record the number of selected LMs when the first LM is connected to node i. The correlation between the values of each metric for the initial LM location and the number of selected LMs obtained with algorithm $\mathrm{P}_{\text{distance}}$, using a distance reduction improvement threshold of 5%, for the four considered networks is depicted in Fig. 3, Fig. 4 and Fig. 5.

It can be observed in Fig. 4 that there is no correlation between the clustering coefficient and the number of selected LMs. Fig. 3 shows that while the number of selected LMs tends to increase as the value the betweenness centrality increases in the case of Germany50 and Deltacom, Abilene and Geant do not exhibit such a trend. A trend common to all the networks

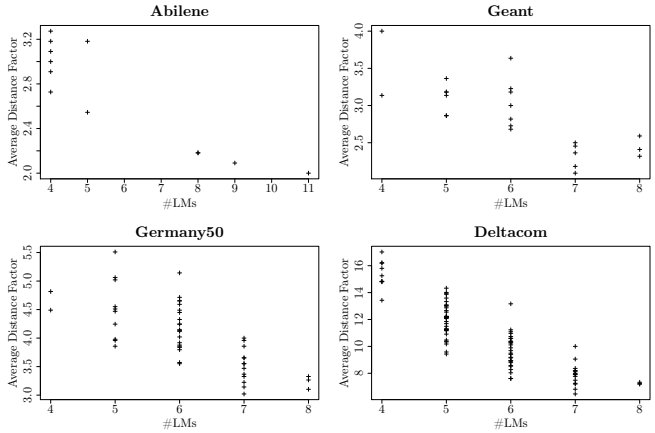


Fig. 5. Number of selected LMs vs. average distance factor.

can, however, be observed in Fig. 5 where it can be noted that the number of LMs tends to decrease as the average distance factor increases. This means that when the initial location is on average close to every other node, a larger number of LMs tends to be selected. These results suggest that the average distance factor is the most relevant metric to consider to decide on the initial placement. By carefully choosing the initial LM location based on this metric, it is possible to control the output of the placement algorithm.

B. Influence of the Distance Reduction Improvement Threshold

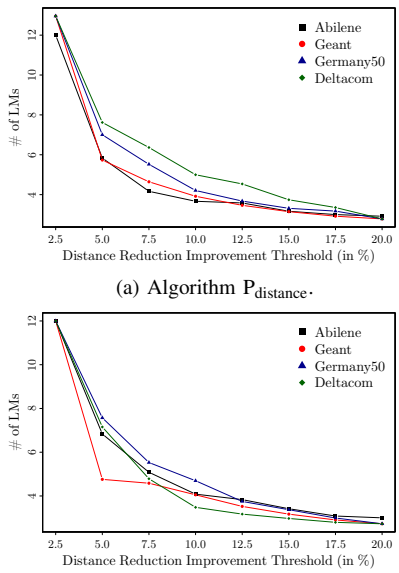
In this subsection, we investigate the influence of the distance reduction improvement threshold on the number of LMs selected with algorithm $P_{distance}$ and P_{weight}[2]. We apply the algorithm in the four topologies with threshold values ranging from 2.5% to 20%. Given that the output is affected by the first LM location, we consider, for all topologies, all possible initial locations, covering as such the totality of the input space. The results are depicted in Fig. 6. For the sake of clarity, the figures present the values averaged over all possible configurations.

As can be observed, all the networks follow the same trend. The number of LMs decreases as the value of the threshold increases. Given that P_{weight} aims at balancing the weight of the nodes between the LMs, it tends to select more LMs than $P_{distance}$ on average. The minimum number of LMs is equal to two and is obtained for all the networks when the threshold is set to 20%. It can also be noted that there is not a significant difference in the number of LMs selected in the four topologies. This can be explained by the definition of the terminating threshold which considers distance reduction gain in absolute values [5].

C. Influence of the Cluster Size Distribution Threshold

By design, algorithm $P_{cluster}$ is *more constrained* than algorithm $P_{distance}$ given that an additional terminating condition is taken into account. In this subsection, we evaluate the influence of the cluster size unbalance threshold on the number of

[2]In this case, the weights are defined based on the population of the city associated with each node in each topology, which can represent a cache management application.



(a) Algorithm $P_{distance}$.

(b) Algorithm P_{weight}.

Fig. 6. Influence of the distance reduction improvement threshold on the number of local managers.

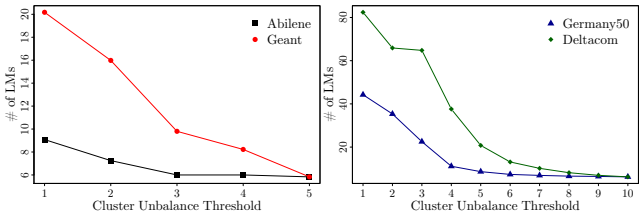
selected LMs. The distance reduction improvement threshold is set to 5%[3] and all initial configurations are considered. In order to better highlight the characteristics of each topology, we categorize the networks into two groups: small size networks with a number of nodes less than 30 (i.e. Abilene and Geant) and medium size network with a number of nodes less than 100 (i.e. Germany50 and Deltacom). For small size networks, the value of the threshold is varied from 1 to 5 and for medium size networks, from 1 to 10. The lower the threshold value, the stricter the constraint imposed to the unbalance degree of the cluster size. The results are depicted in Fig. 7. In a similar fashion to Fig. 6, these represent the average values.

As can be observed, in all cases, the number of LMs increases as the value of the threshold decreases, i.e. the stricter the constraint, the larger the number of selected LMs. It can also be noted that when the threshold is equal to 1 (i.e. the maximum difference authorized between the number of nodes in the largest and smallest clusters is one node only), LMs are deployed on almost all network nodes. Finally, it should be noted that while data on bigger networks (more than 100 nodes) was not available, it is expected that similar results would be obtained.

V. Discussion and Summary

The decision on which placement algorithm to use and how to set the associated parameters should be driven both by the characteristics of the underlying infrastructure and the type of applications to consider. In particular, algorithm $P_{distance}$ is well suited for applications whose main objective is to minimize the average distance between the distributed management and control layer and the network resources, and whose performance is not strongly dependent on the structure

[3]Same observations were made with other threshold values.



(a) Small size networks. (b) Medium size networks.

Fig. 7. Influence of the cluster size unbalance threshold on the number of local managers.

of the computed clusters. The load-balancing approach or the energy management solution presented in [22] and [23], respectively, are examples of such applications. In contrast, the requirements of applications for which the maintenance of a certain level of homogeneity in terms of cluster structure is essential should be tackled with algorithm $P_{cluster}$ or P_{weight}, depending on the weight associated with the nodes. These can be applications which operate at very short timescales and are characterized by frequent communication between the LMs and the network devices (e.g. online monitoring/sampling [24]). These can also be applications sensitive to the volume of information maintained and processed by the LMs, such as cache/content management approaches (e.g. [21]).

In addition, the settings of the algorithm parameters should also be determined according to the requirements of the considered applications. For example, the stronger the requirements in terms of cluster size homogeneity, the stricter the unbalance degree constraint and as such, the smaller the value of the unbalance degree threshold. In a similar fashion, the stricter the constraint in terms of LM-device distance, the lower the value of the distance reduction improvement threshold.

The time complexity of the proposed placement algorithms is dominated by the number of nodes in the network. In the case of algorithms $P_{distance}$ and P_{weight}, this is $O(N^2)$. The complexity of algorithm $P_{cluster}$ depends on the value of the cluster size distribution threshold. The smaller the value, the longer it takes for the algorithm to converge. In the worst case, the complexity is in the order of $O(N^3)$. Given that the computation of the placement of LMs is an offline process, the above computational complexity is acceptable for the size of traditional network domains.

In summary, the proposed approach aims at determining how to allocate management functionality in the context of a SDN-based distributed management and control framework. We compare different placement algorithms and show how the requirements of the physical network infrastructure and management applications can be taken into account to decide on the distribution of the decision-making entities. It is worth noting that, although limitations in terms of single point of failure and lag in reactions can be overcome by distributed management approaches, these often rely on collaborative decision-making processes, which can incur additional communication overhead. We are currently investigating these issues and are developing a framework to support coordination between deciding entities.

ACKNOWLEDGMENT

This research was funded by the EPSRC KCN project (EP/L026120/1) and by the Flamingo Network of Excellence project (318488) of the EU Seventh Framework Programme.

REFERENCES

[1] "Software-Defined Networking: The New Norm for Networks," Apr. 2012, https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf.
[2] T. Koponen et al., "Onix: A Distributed Control Platform for Large-scale Production Networks," in *Proc. USENIX*, ser. OSDI'10, 2010, pp. 1–6.
[3] S. Yeganeh, A. Tootoonchian, and Y. Ganjali, "On scalability of software-defined networking," *Communications Magazine, IEEE*, vol. 51, no. 2, pp. 136–141, February 2013.
[4] M. Charalambides, G. Pavlou, P. Flegkas, N. Wang, and D. Tuncer, "Managing the future Internet through intelligent in-network substrates," *IEEE Netw.*, vol. 25, no. 6, pp. 34–40, nov. 2011.
[5] D. Tuncer, M. Charalambides, S. Clayman, and G. Pavlou, "Adaptive Resource Management and Control in Software Defined Networks," *Network and Service Management, IEEE Transactions on*, vol. 12, no. 1, pp. 18–33, March 2015.
[6] A. Tootoonchian and Y. Ganjali, "HyperFlow: A Distributed Control Plane for OpenFlow," in *Proc. of INM/WREN'10*.
[7] S. Hassas Yeganeh and Y. Ganjali, "Kandoo: A Framework for Efficient and Scalable Offloading of Control Applications," in *Proc. of HotSDN'12*.
[8] R. Ahmed and R. Boutaba, "Design considerations for managing wide area software defined networks," *Communications Magazine, IEEE*, vol. 52, no. 7, pp. 116–123, July 2014.
[9] S. Jain et al., "B4: Experience with a Globally-deployed Software Defined Wan," *SIGCOMM'13*, vol. 43, no. 4, pp. 3–14, Aug. 2013.
[10] R. Clegg, S. Clayman, G. Pavlou, L. Mamatas, and A. Galis, "On the Selection of Management/Monitoring Nodes in Highly Dynamic Networks," *Computers, IEEE Transactions on*, vol. 62, no. 6, pp. 1207–1220, June 2013.
[11] R. Cohen and G. Nakibly, "A Traffic Engineering Approach for Placement and Selection of Network Services," *Networking, IEEE/ACM Transactions on*, vol. 17, no. 2, pp. 487–500, April 2009.
[12] B. Heller, R. Sherwood, and N. McKeown, "The Controller Placement Problem," in *Proceedings of HotSDN'12*, 2012, pp. 7–12.
[13] M. Bari, A. Roy, S. Chowdhury, Q. Zhang, M. Zhani, R. Ahmed, and R. Boutaba, "Dynamic Controller Provisioning in Software Defined Networks," in *Proc. of CNSM'13*, Oct 2013, pp. 18–25.
[14] Y.-N. Hu, W.-D. Wang, X.-Y. Gong, X.-R. Que, and S.-D. Cheng, "On the placement of controllers in software-defined networks," *The Journal of China Universities of Posts and Telecommunications*, vol. 19, Supplement 2, no. 0, pp. 92 – 171, 2012.
[15] F. J. Ros and P. M. Ruiz, "Five nines of southbound reliability in software-defined networks," in *Proc. of HotSDN'14*, ser. HotSDN '14, 2014.
[16] S. Lange, S. Gebert, T. Zinner, P. Tran-Gia, D. Hock, M. Jarschel, and M. Hoffmann, "Heuristic Approaches to the Controller Placement Problem in Large Scale SDN Networks," *Network and Service Management, IEEE Transactions on*, vol. 12, no. 1, pp. 4–17, March 2015.
[17] "The Abilene Internet 2 Topology," http://www.internet2.edu/pubs/200502-IS-AN.pdf.
[18] "The GEANT topology," 2004, http://www.dante.net/server/show/nav.007009007.
[19] "The Germany50 topology," 2004, http://sndlib.zib.de/.
[20] "The Deltacom topology," 2010, http://www.topology-zoo.org/maps/Deltacom.jpg/.
[21] D. Tuncer, M. Charalambides, R. Landa, and G. Pavlou, "More Control Over Network Resources: An ISP Caching Perspective," in *Proc. of CNSM'13*, 2013.
[22] D. Tuncer, M. Charalambides, G. Pavlou, and N. Wang, "Dacorm: A coordinated, decentralized and adaptive network resource management scheme," in *Proc. of NOMS'12*, apr. 2012, pp. 417–425.
[23] M. Charalambides, D. Tuncer, L. Mamatas, and G. Pavlou, "Energy-aware adaptive network resource management," in *Proc. of IM'13*, 2013, pp. 369–377.
[24] S. Clayman, R. Clegg, L. Mamatas, G. Pavlou, and A. Galis, "Monitoring, aggregation and filtering for efficient management of virtual networks," in *Proc. of CNSM'11, mini-conference*, Oct 2011, pp. 1–7.