SDN Controller Placement in LEO Satellite Networks Based on Dynamic Topology

Jianming Guo*, Lei Yang*, David Rincón†, Sebastià Sallent†, Chengguang Fan*, Quan Chen*, and Xuan Li*
*College of Aerospace Science and Engineering,
*National University of Defense Technology, Changsha, Hunan, 410073, China
†Department of Network Engineering,
†Universitat Politècnica de Catalunya (UPC), Castelldefels, 08860, Barcelona, Spain
Email: {guojianming15, yanglei, chengguangfan, chenquan11, lixuan13a}@nudt.edu.cn,
{drincon, sallent}@entel.upc.edu

***Abstract*—Software-defined networking (SDN) logically separates the control and data-forward planes, which opens the way to a more flexible configuration and management for low-Earth orbit satellite networks. A significant challenge in SDN is the controller placement problem (CPP). Due to characteristics such as the dynamic network topology and limited bandwidth, CPP is quite complex in satellite networks. In this paper, we propose a static placement with dynamic assignment (SPDA) method without high bandwidth assumption, and formulate it into a mixed integer programming model. The dynamic topology is taken into account by effectively dividing time snapshots. Real satellite constellations are adopted to evaluate the performance of our controller placement solution. The results show that SPDA outperforms existing methods and can reduce the switch-controller latency in both average and worst cases.**

***Index Terms*—software-defined networking, low-Earth orbit, controller placement, satellite network, dynamic topology.**

I. Introduction

AIMING to provide global Internet services, low-Earth orbit (LEO) satellite networks have gained more attention in recent years. Some LEO constellation projects with hundreds or thousands of satellites are being invested and developed, such as Telesat, OneWeb, Starlink, to name a few. However, subject to the rigid architecture, satellite networks now also confront challenges [1]. On the one hand, network management and configuration are not very flexible, due to the inherent coupling of software and hardware on traditional satellite networking devices [2]. On the other hand, satellite networks are hindered from directly merging with new terrestrial network technologies (e.g., 5G), because their protocols evolve differently and are developed independently [3]. The introduction of software-defined networking (SDN) in satellite networks may overcome the aforementioned problems. With a logically centralized control plane, which is decoupled from the data-forward plane, SDN provides the capability to simplify network programming and management. Nonetheless, embracing SDN can also incur new challenges, among which the controller placement problem (CPP) is of importance. CPP consists in deciding how many controllers are needed for network management and where to locate them (which switches are assigned to a specific controller).

To the best of our knowledge, CPP is first proposed in [4], where the authors use switch-controller latency to evaluate placement results in terrestrial networks. A survey on CPP of terrestrial SDN is presented in [5], where a classical CPP formulation is described as an integer linear programming (ILP) model. Different Objectives of CPP are summarized and discussed, mainly including control latency, load balancing, resiliency, reliability, cost efficiency, etc. Furthermore, the authors sum up both heuristic and optimal methods to solve CPP, among which the greedy algorithm and ILP are the most popular methods. However, these methods cannot be aodopted directly to satellite circumstances due to the dynamic network topology. There also exist some studies on satellite network CPP, which can be divided into two categories: i.e., static controller placement (SCP), and dynamic controller placement (DCP). SCP aims to deploy controllers on fixed nodes and maintain stationary switch-controller assignment as shown in [6], where controllers are statically placed on the ground together with satellite gateways. By contrast, DCP allows a controller to migrate from one node to another, and thus the switch-controller assignment is changeable [7].

Both SCP and DCP struggle to adapt to the characteristics of satellite networks. Owing to the dynamic network topology, SCP cannot satisfy all the topological configurations, which causes long switch-controller latency in some time instances. On the other hand, controller migrations in DCP consume time, bandwidth and power. When a satellite periodically orbits around the Earth, its traffic load will change over time due to the dynamic topology and the Earth's regular rotation. Thus, in order to keep traffic load balanced, previous studies tend to dynamically migrate controllers [7], [8], which requires frequent and synchronous controller migrations and large amount of calculation for the real-time controller placement scheme. Therefore, traffic-based DCP might not be quite

This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB0502402, and by the Agencia Estatal de Investigación of Spain under project PID2019-108713RB-C51/ AEI/10.13039/501100011033.

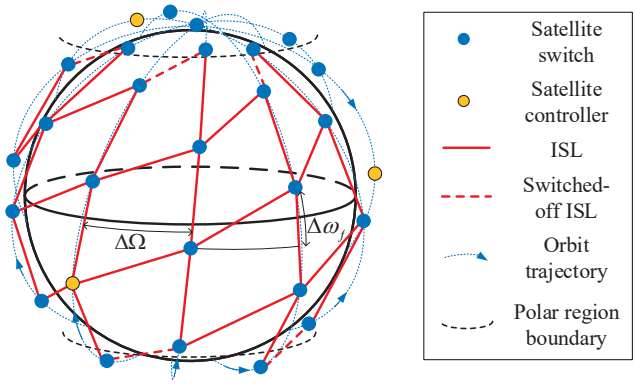


Fig. 1. Architecture of a typical LEO satellite network (*Polar* constellation). Some ISLs and satellites in the polar region are omitted for simplicity.

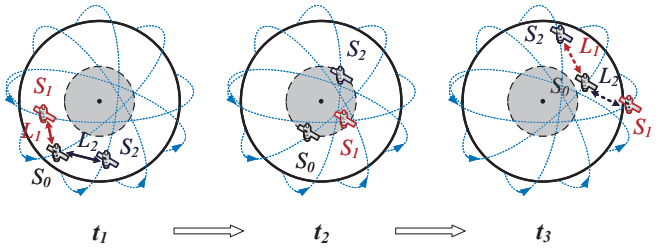


Fig. 2. ISLs are switched off in the polar region (in grey) where two orbits intersect each other (from the view of the North/South pole).

appropriate for satellite networks. A combined static-dynamic method for CPP is proposed in [9], where controllers are statically placed on ground control centers and geostationary Earth orbit (GEO) satellites, but dynamically placed on two LEO satellites of each orbit, which are closest to 0° latitude. Nevertheless, the mobility of LEO satellites is not considered, and no details are provided on the controller migration process.

In this paper, we first give a SDN-based LEO satellite constellation architecture, and use a set of parameters to represent the constellation. Then we analyze the changeable and periodical topology of satellite constellations and present an improved scheme of time snapshot division. Accordingly, we propose a new method, i.e., static placement with dynamic assignment (SPDA), which aims to statically place controllers on certain nodes and dynamically assign switches to them based on the dynamic topology. Moreover, we formulate SPDA into a mixed integer programming (MIP) model. Finally, we adopt two types of real constellations to test SPDA and compare the results with previous works, which indicate that our method outperforms both SCP and DCP.

II. System Model

A. SDN-based Satellite Constellation Architecture

We take both *Walker-δ* and *Polar* constellations into account, and these two types can cover almost all existing systems. A *Walker-δ* constellation usually has inclined orbits, while the inclination angle of a *Polar* constellation is around 90° [10]. The architecture of a typical LEO satellite network is depicted in Fig. 1. Satellites are all SDN switches, and some of them also take the function of SDN controllers. In other words, controllers are physically co-located with satellite switches but logically separated from them. A controller is in charge of switches within its domain and manages their flow tables. Switches only need to handle data packets according to instructions from their controllers.

In a LEO constellation, all satellites are organized into several orbit planes. Let us assume there are P planes and S satellites in each plane; thus, the total number of satellites is $N = P \times S$. A LEO constellation can be denoted by a set of parameters, i.e., h:i:$P/S/F$, where h is the orbit altitude, i is the inclination angle, and F is an integral phase factor, representing the inter-plane phase difference of two neighboring satellites, namely $\Delta\omega_f$ (see Fig. 1). For a *Walker-δ* constellation, $F = 0, 1, ..., P-1$, and $\Delta\omega_f = 2\pi F/N$, while for *Polar*, $\Delta\omega_f = \pi/S$, which is half of the intra-plane phase difference of neighboring satellites. Moreover, $\Delta\Omega$ is the longitude difference between adjacent planes, and it usually equals $2\pi/P$ or π/P for *Walker-δ* or *Polar* constellations, respectively.

Inter-satellite links (ISLs) are established between neighboring satellites either in the same orbit or in adjacent orbits, namely intra-orbit or inter-orbit ISLs, respectively. It should be noticed that in our model inter-orbit ISLs only exist between two satellites moving in the same direction, otherwise they will be unstable owing to high Doppler Shift and have a short duration time, which incurs a so-called seam between two counter-rotating orbits in *Polar* constellations.

B. Dynamic Topology and Snapshot Division

Due to the relative motion of satellites, a LEO satellite network has a dynamic topology. We take a *Polar* constellation as an example to elaborate it. As shown in Fig. 2, before flying into the polar region at t_1, satellite S_0 can maintain inter-orbit ISLs L_1 and L_2 with S_1 and S_2, respectively. When these satellites enter the polar region at t_2, L_1 and L_2 will be temporarily switched off, because their range, azimuth and elevation angles change too fast, which makes antenna tracking difficult. After passing through the polar region, S_0 reestablishes L_1 and L_2 at t_3, but their targeting nodes are exchanged. Similarly, the ISL switch-off will still exist in a *Walker-δ* constellation, because adjacent orbits intersect each other at the latitude near the orbit inclination angle. For the sake of simplification, we will likewise use the polar region to represent the orbit intersection area of *Walker-δ* constellations in the following paragraphs.

In order to deal with the dynamic topology, existing studies have proposed two kinds of methods, namely virtual node (VN) and virtual topology (VT) [11]. Although VN method shields the topological dynamics by dividing the Earth surface

into several cells and binding them with certain satellites, it cannot reflect the intrinsic changes of ISLs, and thus the network dynamic topology. By contrast, VT method maintains the topological changes by dividing the satellite motion period into several time snapshots, which is indeed an effective way for CPP. However, dividing the snapshots evenly by a fixed time step Δt, like in [7], might be unable to capture some topological configurations with a duration shorter than Δt. Moreover, if we divide the time period by topological changes and only keep one instance for each snapshot [8], the duration of topological configurations will have no effect on the CPP results.

To address the snapshot division problem, we modify the VT method in our model. Let us assume that the time period that a satellite cycles around the Earth is T, which is also the period of the constellation, and the network topology changes at a sequence of time instances T_i $(i = 1, 2, ...)$. Then the network snapshots are intervals between two consecutive time instances, $\delta_i = T_{i+1} - T_i$. We first calculate the minimum snapshot length, namely δ_{min}, based on the method in [10], where a virtual orbital plane is created, and all satellites of the LEO constellation are mapped into it. Then we set the fixed time step satisfying $\Delta t < \delta_{min}$. Thus, we can sample a sequence of time instances, i.e., t_k $(k = 1, 2, ...)$, and $t_{k+1} - t_k = \Delta t$.

C. Controller Placement Scheme

We aim to statically place controllers on fixed nodes while dynamically assigning switches to controllers. Therefore, the SPDA consists of two steps: first, we choose appropriate controller locations within a constellation period; second, we calculate the dynamic switch-controller assignment according to the topological changes. Fig. 3 illustrates the reassignment process of switches. When the topology changes between t_1 and t_2, switch S_0 will be detached from the previous controller C_1 and reassigned to its new controller C_2 owing to topology changes.

In terms of the evaluation metrics, the switch-controller latency is most commonly used for CPP [4]. It is also intrinsically equivalent to the flow setup time defined in [7], which is the sum of twice the switch-controller propagation latency plus the end-to-end flow forwarding latency, because the latter actually has no effect on controller placement. We adopt the switch-controller latency as the evaluation metric, and consider the trade-off between optimizing for worst-case and average-case of latency. Although heuristic algorithms can find feasible solutions [5], we apply the linear programming framework and formulate the CPP as a MIP model.

The cost of SPDA only involves the switch migration process, which is quite small [12], compared with the controller migration process illustrated in [13], [14], where the total data store size of a controller is estimated as 100 MB. Although the bandwidth of migration links is set to 1 Gbps (too large for current satellites), the time cost of controller migration is at least 0.8 s, which will be larger if we consider the propagation delay between the previous and newly selected controllers.

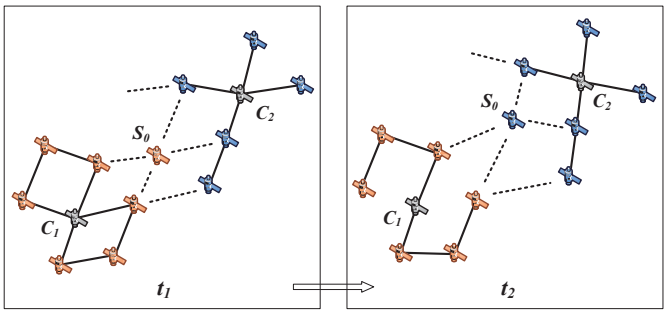


Fig. 3. Switches are reassigned to controllers when network topology changes. Here, solid and dashed lines represent ISLs and indirect connections of satellites, respectively.

III. Controller Placement Problem Formulation

A. Static Controller Placement

We define a time-varying graph $\boldsymbol{G}(\boldsymbol{V}, \boldsymbol{E})$ to describe the network topology of the constellation, where $\boldsymbol{V}$ and $\boldsymbol{E}$ are two sets of satellite nodes and ISLs, respectively. Let us assume that a controller c_i co-locates with the switch v_j, where $c_i, v_j \in \boldsymbol{V}$ and the total number of satellite nodes is N. Thus, we define an N-dimensional vector of 0-1 elements, i.e., $\boldsymbol{x} = (x_1, x_2, ..., x_N)$, as the design variables to represent controller locations,

$$x_i = \begin{cases} 1, & \text{if } c_i \text{ exists} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

The switch-controller assignment is defined as

$$y_{ij} = \begin{cases} 1, & \text{if } v_j \text{ is assigned to } c_i \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The latency between controller c_i and switch v_j will change when satellites are orbiting, and it is denoted as $d_k(c_i, v_j)$ in time instance t_k. We use f_1 to represent the average latency of switch-controller pairs in all time instances,

$$f_1 = \frac{1}{|\boldsymbol{V}|} \sum_{v_j \in \boldsymbol{V}} \bar{d}(c_i, v_j) y_{ij} \tag{3}$$

where $\bar{d}(c_i, v_j) = \frac{1}{T} \sum_k d_k(c_i, v_j)$. Thus, the first objective is to minimize f_1.

Similarly, we denote f_2 as the maximum latency of switch-controller pairs in all time instances:

$$f_2 = \max_{i,j} d^{\max}(c_i, v_j) y_{ij} \tag{4}$$

where $d^{\max}(c_i, v_j) = \max_k d_k(c_i, v_j)$. The second objective aims to minimize f_2, i.e., $\min(\max_{i,j} d^{\max}(c_i, v_j) y_{ij})$. Due to the $\min\max(*)$ function, f_2 needs to be linearized by introducing a new continuous variable z, which satisfies $f_2 = z$. Thus, new constraints should be added as follows:

$$z \geq d^{\max}(c_i, v_j) y_{ij}, \ \ (\forall c_i, v_j \in \boldsymbol{V}) \tag{5}$$

Therefore, we have the total objective function: $\min \boldsymbol{f} = \min(f_1, f_2)$.

The constraints come from several different aspects [5]. First, the total number of controllers K should be:

$$\sum_i x_i = K \tag{6}$$

Second, we need to ensure that one switch is assigned to exactly one controller,

$$\forall v_j \in \boldsymbol{V},\ \sum_{c_i \in \boldsymbol{V}} y_{ij} = 1 \tag{7}$$

Third, c_i should exist if switch v_j is assigned to it,

$$\forall c_i, v_j \in \boldsymbol{V},\ y_{ij} \leq x_i \tag{8}$$

Fourth, the latency between each switch-controller pair should be smaller than a preset threshold Δ_{th},

$$\forall c_i, v_j \in \boldsymbol{V}, k \in T,\ d_k(c_i, v_j) y_{ij} \leq \Delta_{th} \tag{9}$$

B. Dynamic Assignment Method

The dynamic assignment occurs when the topology changes drastically. In other words, when controllers enter or leave the polar region, switches need to be reassigned to new controllers. Because satellites move regularly and periodically, we can pre-calculate the dynamic assignment to save on-orbit computing resources, before launching the satellites. With the static controller placement result $\boldsymbol{x}$, we calculate the time periods that require dynamic assignment. Based on that, we develop a shortest-path based dynamic assignment (SDA) algorithm to calculate the new assignment y_{ij} in those time instances while keeping x_i constant.

Algorithm 1 Shortest-path based dynamic assignment.

Input: $\boldsymbol{x}$, $d_k(c_i, v_j)$
Output: $\boldsymbol{y}_k$

$ResNode_j = 0$ $(j = 0, 1, ..., N)$, $Hops = 1$
while $\sum_i ResNode \neq 0$ **do**
  for $i = 1$ to K **do**
    Find neighboring set of i within $Hops$, $AdjSat$
    for all $j \in AdjSat$ **do**
      $PathLen = ShortestPath(d_k(c_i, v_j), \boldsymbol{x}, j)$
      if $ResNode_j \neq 0$ **then**
        $ResNode_j = 0$, and assign j to controller i
      else
        Calculate current path length of j, $OLen$
        if $PathLen < OLen$ **then**
          Assign j to controller i
        end if
      end if
      Update $\boldsymbol{y}_k$
    end for
  end for
  $Hops = Hops + 1$
end while

SDA is illustrated in Algorithm 1, in which K controllers find their assigned switches simultaneously within a given number of hops, and $ResNode$ labels the residual switches that are not assigned. While there exist unassigned switches, we first find the neigoboring node set, $AdjSat$, within given $Hops$ for a certain controller i. We then use the shortest path algorithm to calculate the current $PathLen$, which is recorded as the shortest if node j is unassigned; or else, we compare $PathLen$ with the current path length and decide the best controller. The computational complexity of SDA is $O(N)$, which is better than that of traversing all node pairs, i.e., $O(N^2)$.

TABLE I
Parameters of the constellations

Constellation	Iridium	Celestri
N	66	63
P	6	7
S	11	9
h (km)	780.0	1400.0
i (°)	86.4	48.0

IV. Case Study

A. Simulation Settings

We adopt Iridium and Celestri (a *Polar* and *Walker-δ* constellation, respectively) to evaluate our SPDA method. The parameters of the two constellations as defined in Section II-A are listed in Table I, and the phase factor F of Celestri is set to 1 for simplicity. According to the constellation parameters, we automatically generate the satellite orbit elements in STK through a MATLAB program script and export the orbit files containing satellite positions in each time slot. We then build the MIP model and use Gurobi 9.0.1 to solve the CPP problem. The simulation settings for Iridium and Celestri, respectively, are given as follows:

- The polar region is defined by a latitude boundary, β, which is smaller than i. Here, we set β to $80°$ and $45°$.
- The simulation time is set to the period of constellations T, i.e., 6030 s and 6840 s. The minimum time interval of snapshots, δ_{min}, is calculated as 133.2 s and 24.8 s.
- The fixed step for sampling time instances, Δt, is set to 30 s and 20 s to obtain a more precise snapshot division. Subsequently, the number of time instances is calculated as 202 and 343.
- Because the average length of ISL is at most thousands of kilometers for both Iridium and Celestri, the switch-controller latency threshold, Δ_{th}, can be set to 60 ms to ensure the multihop transmission delay.
- The number of controllers, K, is initially set to 6, and it will change in our following simulations.

B. Results and Discussion

We compare our method (i.e., SPDA) with SCP and DCP regarding to the cumulative density function (CDF) of both the average and maximum switch-controller latency. The results are presented in Fig. 4 and 5 for Iridium and Celestri, respectively. Sub-figures (a) and (b) show the average and maximum

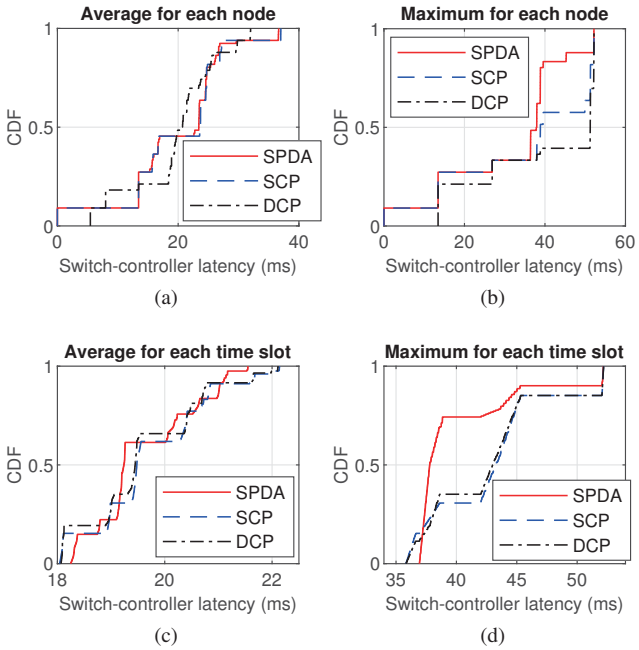


Fig. 4. CDF of switch-controller latency in Iridium.

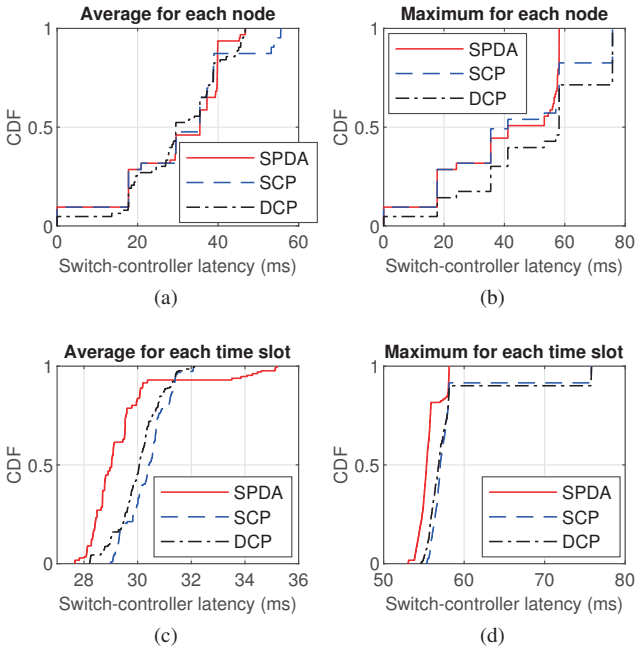


Fig. 5. CDF of switch-controller latency in Celestri.

values of latency of each node for all time instances, while sub-figures (c) and (d) show those of each time instance for all nodes. For the average case of Iridium in Fig. 4 (a) and (c), the three methods generate almost the same results, while for the maximum case, both SCP and DCP are bounded by SPDA as shown in Fig. 4 (b), and about 80% nodes have a switch-controller latency lower than 40 ms for SPDA. According to Fig. 4 (d), although no time instance has lower latency than 37 ms for SPDA, 80% of them are between 37 and 39 ms. By contrast, only 40% of time instances are lower than 39 ms for SCP and DCP. In Fig. 5 (a) and (c), the largest average latency for each node in SPDA is smaller than that of SCP and DCP, and SPDA also generates about 90% time instances with a much smaller average latency, i.e., 30 ms. In Fig. 5 (b) and (d), SPDA also incurs lower maximum latency for not only each node but also each time instance. Overall, SPDA has the best performance because of the improved time division method, and it has relatively low bandwidth utilization rate as well as low computational costs.

We then set $K = 8$ controllers and evaluate the trade-off between the average and worst case by changing the priority and weight of the two objectives. The priority varies from 1 to 9 while the weight varies from 0 to 3, and the MIP optimizer in Gurobi is independently run for several times. We record the best solutions of each trial and show the results in Fig. 6. Each red point represents a solution, and the results are approximately distributed on several lines quantized by values of y-axis. This is because the maximum latency reflects the number of hops between controllers and switches. Moreover, the Iridium case shows smaller latency on both the maximum and average value than the Celestri case. It is reasonable

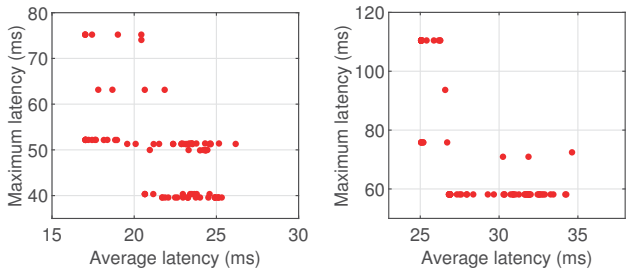
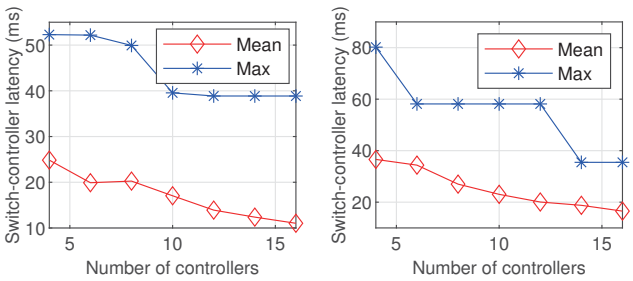


Fig. 6. Trade-off between the worst and average latency.

that the maximum latency maintains stable while the average latency varies a lot for different controller placement schemes. Therefore, our aim is to reduce the average latency while maintaining a relatively low maximum latency.

We now use SPDA to calculate the best controller placement schemes for different K (the number of controllers), and the results are shown in Fig. 7. When K increases, the average switch-controller latency decreases for both the Iridium and Celestri cases. However, the maximum latency of Iridium remains stable for the case of 10 or more controllers. By contrast, the maximum latency of Celestri reaches 40 ms when the controller number is more than 14, and it will predictably stay at 40 ms even with more controllers. Moreover, with the same number of controllers, Iridium always has lower latency of both the average and maximum cases than Celestri, because the ISL length of Iridium, determined by the constellation, are generally shorter than that of Celestri.



(a) Iridium (b) Celestri

Fig. 7. Relations between switch-controller latency and the number of controllers.

V. Conclusion

Considering the limited bandwidth and dynamic topology in LEO satellite networks, we propose the SPDA method to statically place controllers and dynamically adjust the switch-controller assignments. We formulate SPDA into a MIP model, and develop the SDA algorithm to effectively assign switches. Simulation results show that our method outperforms both SCP and DCP. The trade-off between the average and worst case of latency is also evaluated, and the maximum latency maintains stable while the average latency changes a lot for different controller placement schemes. Furthermore, the average latency will decrease while the maximum latency is hard to reduce with more controllers. Future work of CPP in satellite networks could involve traffic demands of terrestrial users and the evaluation of node reliability. Other controller placement metrics could also be considered in future work.

References

[1] Y. G. Bi, G. J. Han, S. Xu, X. W. Wang, C. Lin, Z. B. Yu, and P. Y. Sun, "Software defined space-terrestrial integrated networks: Architecture, challenges, and solutions," *IEEE Network*, vol. 33, no. 1, pp. 22–28, 2019.

[2] L. Bertaux, S. Medjiah, P. Berthou, S. Abdellatif, A. Hakiri, P. Gelard, F. Planchou, and M. Bruyere, "Software defined networking and virtualization for broadband satellite networks," *IEEE Communications Magazine*, vol. 53, no. 3, pp. 54–60, 2015.

[3] R. Ferrus, H. Koumaras, O. Sallent, G. Agapiou, T. Rasheed, M. A. Kourtis, C. Boustie, P. Gelard, and T. Ahmed, "SDN/NFV-enabled satellite communications networks: Opportunities, scenarios and challenges," *Physical Communication*, vol. 18, pp. 95–112, 2016.

[4] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, ser. HotSDN '12. ACM, 2012, Conference Proceedings, pp. 7–12.

[5] T. Das, V. Sridharan, and M. Gurusamy, "A survey on controller placement in SDN," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 472–503, 2020.

[6] J. Liu, Y. Shi, L. Zhao, Y. Cao, W. Sun, and N. Kato, "Joint placement of controllers and gateways in sdn-enabled 5g-satellite integrated network," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 2, pp. 221–232, 2018.

[7] A. Papa, T. D. Cola, P. Vizarreta, M. He, C. M. Machuca, and W. Kellerer, "Dynamic SDN controller placement in a LEO constellation satellite network," in *IEEE Global Communications Conference (GLOBECOM)*, 2018, Conference Proceedings, pp. 206–212.

[8] S. Wu, X. Chen, L. Yang, C. Fan, and Y. Zhao, "Dynamic and static controller placement in software-defined satellite networking," *Acta Astronautica*, vol. 152, pp. 49–58, 2018.

[9] S. Xu, X. Wang, B. Gao, M. Zhang, and M. Huang, "Controller placement in software-defined satellite networks," in *2018 14th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN)*, 2018, Conference Proceedings, pp. 146–151.

[10] J. F. Wang, L. Li, and M. T. Zhou, "Topological dynamics characterization for LEO satellite networks," *Computer Networks*, vol. 51, no. 1, pp. 43–53, 2007.

[11] Q. Chen, J. Guo, L. Yang, X. Liu, and X. Chen, "Topology virtualization and dynamics shielding method for LEO satellite networks," *IEEE Communications Letters*, vol. 24, no. 2, pp. 433–437, Feb 2020.

[12] Y. Xu, M. Cello, I. C. Wang, A. Walid, G. Wilfong, C. H. P. Wen, M. Marchese, and H. J. Chao, "Dynamic switch migration in distributed software-defined networks to achieve controller load balance," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 3, pp. 515–529, 2019.

[13] M. He, A. Basta, A. Blenk, and W. Kellerer, "How flexible is dynamic SDN control plane?" in *2017 IEEE Conference on Computer Communications Workshops*, 2017, Conference Proceedings, pp. 689–694.

[14] A. Papa, T. d. Cola, P. Vizarreta, M. He, C. Mas-Machuca, and W. Kellerer, "Design and evaluation of reconfigurable SDN LEO constellations," *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1432–1445, 2020.