Minimizing Energy Consumption in Software Defined Networks

Kuldeep Kurroliya
Dept. of CSE
NIT, Rourkela
Odisha, India
kuldeep.nitrkl2304@gmail.com

Sagarika Mohanty
Dept. of CSE
NIT, Rourkela
Odisha, India
sagarikam_23@yahoo.com

Bibhudatta Sahoo
Dept. of CSE
NIT, Rourkela
Odisha, India
bdsahu@nitrkl.ac.in

Khushboo Kanodia
Dept. of CSE
NIT, Rourkela
Odisha, India
kkanodia2307@gmail.com

***Abstract*—Software-Defined Network (SDN) isolates the data plane networking equipment from the control plane to speed up the quick distribution, composition, and growth of networks. Controllers in the software-defined network play an important role like managing the tasks performed by the switches which come at a tradeoff of the aim of maximizing the energy saving in SDN. If more edges of SDN networks are switched off, it may lead to select the routes where propagation delays will be larger. In this regard, we refer to the task of allocating switches to the controller with the target of making the maximum number of inactive edges while satisfying the latency. In this paper, we propose a link based genetic algorithm(LBGA) and compared this with Greco. From the evaluation, it is observed that the results of various topologies have shown energy savings up to more than 55% during the hours when the edges are inactive and this is better in comparison to Greco.**

***Index Terms*—Software Defined Network(SDN), Energy Saving, Latency, Link based Genetic Algorithm(LBGA)**

I. Introduction

Software-Defined Network (SDN) exploits a new networking epitome which facilitates the utilization of path-finding policies and also helps in accelerating network structure and it's enlargement. Software-defined networking is an architecture for computer networks aimed at decoupling the network control functions(control plane) from the forwarding devices(data plane) [1]. Determining the control logic and implementing the routing protocol is the main responsibility of the control plane. SDN has the logically centralized programmable controller in the control plane which contains all the network information and make it easy for network operators to flexibly adjust and deploy new network architecture. Unlike traditional network in which data and control plane are more tightly coupled and make it very difficult for network administrators to create any changes in the network.

Software Defined Network (SDN) architecture is split into three layers, infrastructure layer, control layer, and application layer. The infrastructure layer consists of various networking equipments like switches, routers, etc. which help in forwarding the traffic of three types of communications such as (switch to switch communications(S2S), switch to controller communications(S2C), controller to controller communications(C2C)). The forwarding of the traffic packets is done based on the rules given by the controller. The communication between this plane and the control plane is done through the northbound interface. The devices in this plane store the information about the various states of the network and then transfers it to the controller. Switches stores the data supplied by the controller in the flow tables. Every row contains information like statistics, action, and identifiers. When a packet arrives, the packet header is matched with the entries in the row of the flow table. If the information matches, then the required action takes place for the particular packet otherwise, the request is sent to the controller for further processing.

The control layer of Software-Defined Network (SDN) manages the data plane of the network. The controller maintains all information of the network which is sent to it by the devices(switches, routers) in the data plane. The responsibility of the controller is to maintain various rules in the flow tables. The controller interacts with the data plane devices by using the southbound interface like OpenFlow. Controllers interact among themselves using the westbound interface and east interface. Few open-source and community-driven implementations of the controller which follows the standard of OpenFlow are Flood-light, POX, Maestro, and Ryu.

The application layer of the Software-Defined Network (SDN) implements the functionality of various middleboxes like NAT, firewalls and load balancers as an application on a server software.

The overall contributions of the proposed work are as follows. Random generation of routes for every (s, d) pair of S2C traffic flow demands. The total number of switches assigned to a controller according to the "Opcon" value considering the average and worst-case latency. We proposed an LBGA algorithm which satisfies the S2C demands with less number of edges, thereby saving more energy than Greco [2]. The main objective is to minimize the number of active edges for the given S2C demands. The topology datasets are taken from the internet topology zoo and the algorithms are executed on different topologies and the corresponding comparison results on three topologies are shown in the results.

The contents of the paper are organized in the following manner. Section II discussed the related works. Section III tells about the network model. Section IV states the Problem statement(objective function) of the network model and constraints. Section V discuss the proposed genetic algorithm

TABLE I
NOTATIONS

Symbols used	Description
N	Set of nodes
L	Set of edges
S	Set of switches in data plane
C	Set of controllers
c_l	capacity
Θ	Set of traffic flow between switch-to-switch(S2S), controller-to-controller(C2C), switch-to-controller(S2C)
θ	Set of traffic flow between switch-to-controller(S2C)
dem_{sd}	traffic request between s and d
P_{sd}	Set of random paths for every (s, d) $\in$ S2C demands
Q_{sd}	Set of paths selected by LBGA for every S2C demands.
p_k^{sd}	k-th path from the given random paths
ρ_k^{sd}	propagation delay of k-th path
$\delta_{l,p}^{sd}$ ϵ {0, 1}	Binary variable set to 1 if link l is used
D_{sd}	average propagation delay for every (s,d) ϵ θ
u_{max}	Maximum link utilization
u_l	Edge utiilization of link l

(LBGA). Section VI shows the various results and comparison between Greco and LBGA. Section VII concludes the paper.

II. Related Works

In [3], Adriana et. al discussed the problem of placing the controllers in a distributed way for a control plane to minimize the energy reduction in SDN. For small and medium-scale networks BIP model was applied and for large scale networks IGCPA algorithm was applied which gives the suboptimal solution for placing the controllers. IGCPA gives a better solution. In [4], Madhukrishna et. al discussed the technique for maximizing energy-saving while adopting an energy-saving path with less latency. In [5], Jaime et. al discussed how the power saving is maximized of an IP/SDN hybrid network through an ILP based model and genetic-based algorithm. In [6], Beakal et. al discussed the method of energy-saving based on maximizing the RESDN value of the edges. The RESDN ratio also takes into account the contribution of maximum utilization of edges parameters. In [7], the author proposed a mechanism for allocating the resources to the SDN network according to the flow demand request. In [8], the energy issues related to the location of facility allocation were discussed. In [9], Ying et. al discussed the minimization of energy consumption by a green design or arrangement and paths for satisfying the demand requests based on the current status of the network. In [10], the energy-aware mechanism was proposed by the author that maximizes the energy-saving in SDN networks up to 50%. In [11], Heller et al. discussed the metrics of average latency, worst latency, latency bound. In [12] and [13] the authors discussed about the K-paths for routing the flow traffic between switch to controller(S2C) and controller to controller (C2C) communications to increase reliability. In [14], the author's main aim is to increase reliability. In [15], other metrics like redundancy and load balancing are considered by the authors. In [16], the author discussed the balancing of load via K center problem considering the capacity of the controller. In [17], Samir et. al focused to minimize the worst-case latency between controller and switches.

Considering all the above-related works, a different traffic-engineering energy-aware method is proposed through the genetic approach. In this approach, we try to switch off several edges to maximize the energy saving in the network while still maintaining a path for the flow traffic between the controller and switches while satisfying the latency and controller load. Energy has become a major factor as the carbon imprints in the various networks is expected to rise in a significant amount. Hence various researchers are trying to develop solutions to minimize the cost of energy expenditure in the network. From the literature review of the above work, we can assume that this is the first work to simultaneously consider the load of the controller, propagation delay in the path from the controller to switches, maximum link utilization (MLU) as well as a reduction in energy consumption of SDNs network. We have obtained the results of our algorithm on three topologies: Abilene, AT&T, GEANT and compared our results against the GreCo algorithm. The experimental results show that the Genetic approach is able to make 60%(approx.) of the total number of inactive edges in the topology.

III. Network Model

Software defined network is represented as an undirected graph G(N, L). N is set of nodes. L is the set of edges / links. Set S = $\{s_1, s_2, s_3, ..., s_{|S|}\} \subset$ N is the set of switches and C = $\{c_1, c_2, c_3, ..., c_{|C|}\} \subset$ N is the controller set. Every link between node i and node j has capacity c_l. We assume that equal volume of energy is consumed by every link. Let Θ set represents the demands of the flow traffic between S2C, C2C and S2S where S2C represents switch to controller traffic flow demands, C2C represents controller to controller traffic flow demands and S2S represents switch to switch traffic flow demands. Let $\theta \in \Theta$ represents only S2C traffic flow demands. Every (s, d) $\in \Theta$ has demand dem_{sd}, where (s,d) $\in$ N denotes the node pairs contained in S2C, C2C or S2S traffic flow demands. The set consisting of random paths equal to population size for every (s, d) pair is P_{sd}, each has an index k. The particular path can be represented as p_k^{sd}. The associated propagation delay is ρ_k^{sd}. ps denotes the population size and this much number of random paths are generated for every (s, d) pairs. Let $\delta_{l,p}^{sd} \in \{0, 1\}$ is a binary variable whose value is 1 if route p $\in$ P_{sd} consists of link l to pass the flow traffic of demand dem_{sd}. The route set passing through the links l in (s, d) $\in$ Θ. For a particular edge the overall flow traffic passing through is given by $L_l = \sum_{p \in P_l} D(p)$, where $D(p)$ is the function which denotes the demand of the total traffic carried by the route p. Edge utilization of every edge is given by $u_l = L_l/c_l$ and the maximum value of edge utilization is $u_{\max}$. The average propagation delay for each demand (s, d) $\in$ θ is given by

$$D_{sd} = \frac{\sum_{p_k^{sd} \in P_{sd}} \rho_k^{sd}}{|P_{sd}|} \tag{1}$$

IV. Problem Statement

Software-defined network(SDN) is designed as G(N, L) with $|C|$ number of controllers. Each pair of traffic flow $\in$ θ has a demand dem_{sd}. Our main aim is to switch off as many edges as possible when no traffic is passing through them. Let $y_l \in \{0, 1\}$ denotes that edge l is on, i.e, it denotes that some flow is passing through it. Variable p_k^{sd} is a binary variable used to denote that the particular shortest path is selected among the population of random paths computed by the algorithm. Arithmetically, our objective function can be stated as,

$$min \sum y_l, \qquad \forall l \epsilon L \tag{2}$$

satisfying the constraints,

$$\sum_{k=1}^{ps} p_k^{sd} = 1, \qquad \forall (s,d) \in \Theta \tag{3}$$

$$y_l \geqslant p_k^{sd}, \qquad \forall l \in L, \forall p_k^{sd} \in P_l \tag{4}$$

$$\sum_{(s,d) \in \Theta, p_k^{sd} \in P_l} d_{sd} p_k^{sd} \leqslant c_l u_{max} \qquad \forall l \in L \tag{5}$$

$$\sum_{p_k^{sd} \in P_l} p_k^{sd} \leqslant Th, \qquad \forall c \in C \tag{6}$$

$$AverageLatency(c') = \frac{1}{N} \sum_{c \in C} \min_{s \in S'} distance(c, s) \tag{7}$$

$$WorstLatency(c') = \max_{c \in N} \min_{s \in S'} distance(c, s) \tag{8}$$

The main aim is to maximize the total number of switched off links which contribute to the energy saving of the network and also satisfying the constraint that every demand $\in$ Θ is satisfied by the switched-on links. The other constraints include: one route can be chosen for every (s, d) pair from (3), an edge is in on state only when some flow traffic is passing through it otherwise it is in off state(4). The overall edge utilization is less than the maximum link utilization(MLU)(5), and the maximum switches that can be handled by a controller at any instant are bounded by value Th(6). Switches are assigned to the controller according to the constraints defined in equation (7) and (8).For every pair (s, d) $\in$ θ random paths equal to population size ps are computed. Each path may be routed through a different controller $\in$ Q_{sd} satisfying the delay condition D_{sd}. In this way, S2C demands delay conditions are also taken into account. Here the case of making the switches in data plane inactive is not taken into account as they may again be allocated to different controllers depending on the future traffic demands. The delays involved in making the edges active are assumed to be negligible. The value of Threshold Th is considered in the range $(|S|/|C|, |S|)$.

V. Heuristic

Algorithm 1 LBGA

Input : T, ps, θ.
Output : Q_{sd}

```
 1: $T \leftarrow$ Topology Matrix
 2: $ps \leftarrow$ Population size
 3: $itr \leftarrow$ total number of iterations
 4: $pop \leftarrow \phi$
 5: for $i = 1$ to $ps$ do
 6:     $r \leftarrow$ randomly select an edge/route from $L$
 7:     $pop \leftarrow pop \cup r$
 8: end for
 9: for $i = 1$ to $itr$ do
10:     $n\_pop \leftarrow \phi$
11:     $scores \leftarrow fitness(pop, T)$
12:     $best = min(scores)$
13:     $par_1, par_2 \leftarrow selection(pop, scores)$
14:     $r_1, r_2 \leftarrow crossover(par_1, par_2)$
15:     $n\_pop \leftarrow n\_pop \cup \{r_1, r_2\}$
16:     for $j = 1$ to $length(n\_pop)$ do
17:         $r[j] \leftarrow mutation(n\_pop[j])$
18:         $n\_pop[j] = r[j]$
19:     end for
20:     $n\_pop \leftarrow n\_pop \cup (par_1, par_2)$
21:     for $k = length(n\_pop)$ to $ps$ do
22:         $r \leftarrow$ randomly select an edge/route from $L$
23:         $n\_pop \leftarrow n\_pop \cup r$
24:     end for
25:     $pop \leftarrow n\_pop$
26:     $i \leftarrow i + 1$
27: end for
28: $scores \leftarrow fitness(pop, T)$
29: $best = min(scores)$
30: $Q_{sd} \leftarrow best$
31: $return\ Q_{sd}$
```

A genetic approach, LBGA is proposed to solve the problem. The main target is to maximize the energy-saving of the network by making more number of inactive edges satisfying the constraints of S2C traffic flow, balancing the load and edge utilization. The topology matrix T, population size ps, and number of iteration itr are set as input parameters for LBGA algorithm. This approach selects the path with a minimum number of edges and low latency. If there is not sufficient bandwidth to route a demand then it is "$rejected$" otherwise "$selected$". The controllers have the dual capability of behaving like a controller as well as a switch. To ensure that all controllers control an equal number of switches, the switches are assigned to the controller based on the 'Opcon" value,i.e., ($|S|/|C|$).

The initial population of some valid paths is randomly generated from the topology matrix T. Then based on the number of hops present in the various paths generated, scores or fitness values are calculated for each path. For example, [[1,3,7,11], [1, 4, 6, 8, 9, 11]] is the population of pair (1, 11).Based on the $scores$, the path with the lowest latency and a minimum number of hops is selected as the best path and stored in $best$. Based on the $scores$ assigned to the paths, two best routes par_1, par_2 are selected as the parent routes for crossover to generate new routes r_1, r_2. Invalid routes are rejected. Then the population is updated. Mutation is performed on the routes of the new population n_pop to generate improved new routes having latency less than the previous routes. Few members of the previous generation are kept in case there are no improved routes generated in the successive iteration. If the population size is still less than the desired value new random routes are generated and added to the new population. Then the new population is updated. The above procedure is repeated for the total number of iterations defined. The best route selected after the last iteration is used to satisfy the current (s, d) pair $\in \theta$. In this process, the algorithm checks that if the switch assigned to a particular controller can be moved to another controller where the latency between the switch and controller is low.

The $best$ routes in Q_{sd} refers to the path used to satisfy the demands contained in θ and the links of the corresponding paths are known as "active links". If any path or link whose threshold is greater than the MLU then that path is rejected for the demand and some other path is selected where MLU is less than 80%.

The most time-consuming step in Greco algorithm was the Yen algorithm to calculate K shortest paths. So, the time complexity of Greco algorithm is O($K|N|(|L|+|N|log|N|)$), where K is K paths computed for each request in θ. The overall time complexity of Greco algorithm is O($K|N|^5$). In LBGA, the time complexity of the algorithm depends on the number of iterations taken by the genetic algorithm to compute the best path for each demand in θ. So, the proposed genetic approach takes less time than Greco algorithm.

VI. Evaluation

The network topologies considered are Geant(37 nodes, 58 edges), AT&T(25 nodes, 56 edges), Abilene(11 nodes, 14 edges). Topology and traffic matrices are obtained from the internet zoo topology dataset in graphml format and then converted into distance matrix using haversine formula. The experiments were conducted in a Python 3.7 environment. The controllers and switches were selected according to equation (7) and (8). Routes selected by the algorithm are used to transmit the data packets between S2C till the utilization of any link on the route does not go beyond MLU. The MLU is kept as 80% for active links. As a result, the total number of active links may vary on each execution of the algorithm. For a particular controller, the corresponding switches select the different routes because the paths are selected on the criteria of propagational latency randomly and not on the basis of link utilization. Therefore, given (s, d) pair, the route p_k^{sd} does not remains the same. The energy savings can be calculated by the formula

$$\frac{Number\,of\,inactive\,edges}{Total\,number\,of\,edges} \tag{9}$$

TABLE II
Topology Information

Topology	Number of nodes	Number of edges
Abilene	11	14
AT&T	25	56
Geant	37	58

TABLE III
Genetic Approach Parameters

Parameters / Topology	Abilene	AT&T	Geant
Population Size	7	28	30
# iterations	6	10	10
Mutation probability	0.06	0.06	0.06

Fig. 1 shows the impact of the percentage of energy savings on an increasing number of controllers. From the simulations, it is observed that LBGA performs better in terms of percentage of energy savings in comparison to Greco algorithm

It is noticed that the total number of (s, d) routed demands decreases when the load on the network increases. All the topologies considered shows a similar kind of behavior on increasing the network load. This is the expected behavior of our approach given that the traffic requests are satisfied within link utilization, i.e., 80%.

VII. Conclusion

By making more number of inactive links may heavily load some other edges and may increase the chances of increasing the total energy consumption.So, therefore, the main aspect to be focussed is the maximization of energy savings by making more number of inactive links while considering the average and worst-case latency. In Greco, energy savings of up to 55% is achieved during the hours when the links are inactive. In LBGA, energy savings of up to 60%(approx.) is achieved during the hours when the links are inactive. In LBGA, population size below half the total number of edges, i.e. ($ps \leq 12L$), as well as several iterations below 10, do not give good results in most of the topologies. To solve this issue, the main focus is on minimizing the active

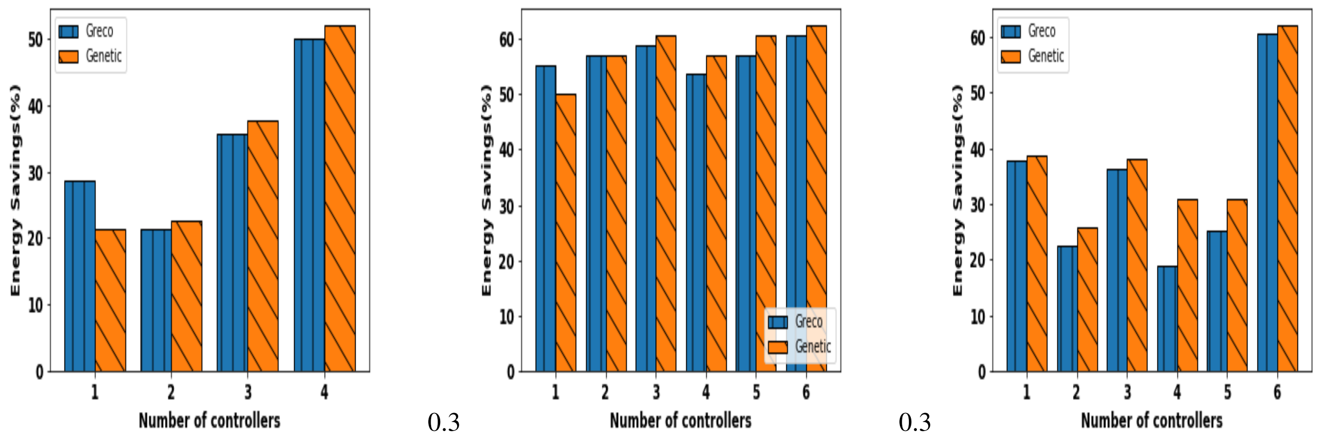

Fig. 1. Percentage of shut down links for Abilene, AT&T, Geant topologies considering average latency

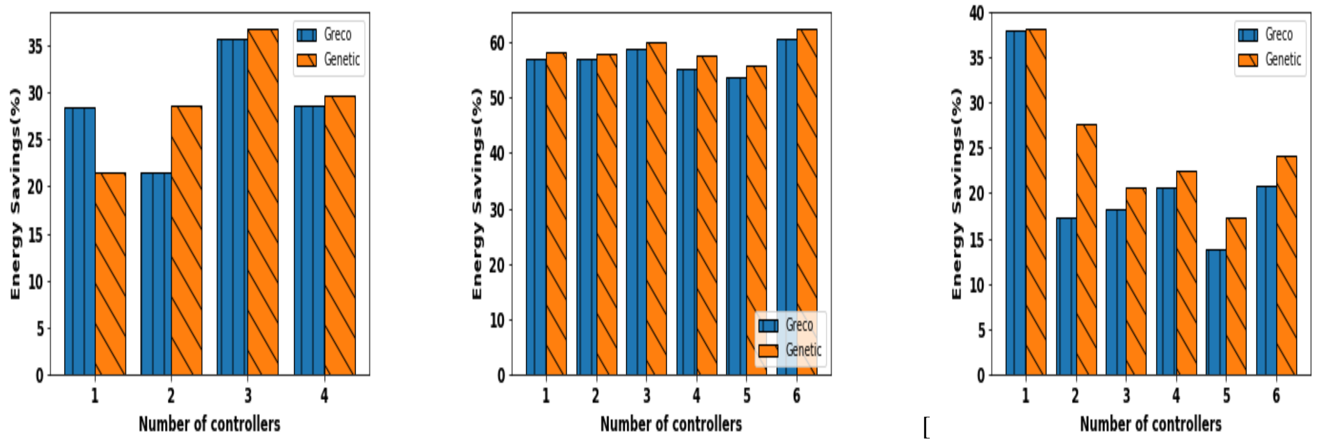

Fig. 2. Percentage of shut down links for Abilene, AT&T, Geant topologies considering worst latency

edges satisfying the constraints of link utilization, a load of the controllers, the average propagation delay of the data packets to be transmitted between switch and controller. From the results, we can conclude that there is an average of 60% energy savings during the hour's edges are inactive. For future work, a different model of energy consumption which focuses on minimizing the rate of energy consumption of links along with minimizing the cost can be considered.

References

[1] Y. Jarraya, T. Madi, and M. Debbabi, "A survey and a layered taxonomy of software-defined networking," *IEEE communications surveys & tutorials*, vol. 16, no. 4, pp. 1955–1980, 2014.

[2] A. Ruiz-Rivera, K.-W. Chin, and S. Soh, "Greco: An energy aware controller association algorithm for software defined networks," *IEEE communications letters*, vol. 19, no. 4, pp. 541–544, 2015.

[3] A. Fernandez-Fernandez, C. Cervello-Pastor, and L. Ochoa-Aday, "Achieving energy efficiency: An energy-aware approach in sdn," in *2016 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2016, pp. 1–7.

[4] M. Priyadarsini, P. Bera, and M. A. Rahman, "A new approach for energy efficiency in software defined network," in *2018 Fifth International Conference on Software Defined Systems (SDS)*. IEEE, 2018, pp. 67–73.

[5] J. Galan-Jimenez, "Minimization of energy consumption in ip/sdn hybrid networks using genetic algorithms," in *2017 Sustainable Internet and ICT for Sustainability (SustainIT)*. IEEE, 2017, pp. 1–5.

[6] B. Assefa and O. Ozkasap, "Resdn: A novel metric and method for energy efficient routing in software defined networks," 05 2019.

[7] R. Lin and Z. Ye, "A bat algorithm for sdn network scheduling," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, 12 2018.

[8] H. Jia, Y. Xu, G. Tian, M. Zhou, J. Zhang, and H. Zhang, "Random energy-efficient models for sustainable facility location subject to carbon emission, economical, capacitated and regional constraints," *IEEE Access*, vol. 6, pp. 72 757–72 765, 2018.

[9] Y. Hu, T. Luo, W. Wang, and C. Deng, "Gresdn: Toward a green software defined network," in *2016 18th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, 2016, pp. 1–6.

[10] M. Mihoubi, A. Rahmoun, P. Lorenz, and N. Lasla, "An effective bat algorithm for node localization in distributed wireless sensor network," *Security and Privacy*, vol. 1, no. 1, p. e7, 2018.

[11] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012, pp. 7–12.

[12] Y. Hu, W. Wang, X. Gong, X. Que, and S. Cheng, "On reliability-optimized controller placement for software-defined networks," *China*

Communications, vol. 11, no. 2, pp. 38–54, 2014.

[13] 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, pp. 92–171, 2012.

[14] F. J. Ros and P. M. Ruiz, “Five nines of southbound reliability in software-defined networks,” in *Proceedings of the third workshop on Hot topics in software defined networking*. ACM, 2014, pp. 31–36.

[15] Y. Hu, W. Wang, X. Gong, X. Que, and S. Cheng, “Balanceflow: controller load balancing for openflow networks,” in *2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems*, vol. 2. IEEE, 2012, pp. 780–785.

[16] G. Yao, J. Bi, Y. Li, and L. Guo, “On the capacitated controller placement problem in software defined networks,” *IEEE Communications Letters*, vol. 18, no. 8, pp. 1339–1342, 2014.

[17] S. Khuller and Y. J. Sussmann, “The capacitated k-center problem,” *SIAM Journal on Discrete Mathematics*, vol. 13, no. 3, pp. 403–418, 2000.