Optimization of Power Consumption in SDN Networks

Adrián Flores de la Cruz, Juan Pedro Muñoz-Gea, Pilar Manzanares-Lopez, Josemaria Malgosa-Sanahuja
Department of Information and Communication Technologies
Universidad Politécnica de Cartagena
Campus Muralla del Mar, 30202. Cartagena, Spain
Email: {adrian.flores, juanp.gea, pilar.manzanares, josem.malgosa}@upct.es

***Abstract*—Software-Defined Networking (SDN) offers the possibility to carry out a more direct control of network behavior and to interact directly with the elements of the network. In this paper, the problem of optimizing the power consumption in SDN networks is addressed by looking for the most appropriate set of active switches and links, their associated rates, and the number of flow entries at each SDN switch. We present a formulation for this optimization problem and a heuristic algorithm to reduce time complexity in large topologies. Energy saving values of more than 40% are reached using real traffic demands data.**

Keywords–SDN; Energy efficiency; Optimization.

I. INTRODUCTION

Nowadays, the optimization of network power consumption is considered as one promising field of application for Software-Defined Networking (SDN). This new networking technology offers the possibility to carry out a more direct control of network behavior and it gives us the possibility to interact directly with the elements of the network. Considering that strategies, such as turning off network switches, links, or reducing the link rate, can result in energy savings without affecting the service quality, SDN appears as a proper tool to act over network devices and configure them following these prerequisites.

It is known that the highest energy saving is achieved when entire interconnection devices are turned off. However, the energy consumption of data networks can also be minimized by reducing the number of other active elements. For example, this feature can be implemented by putting into a low-power sleep state (sleep mode) elements, such as line cards or port interfaces whenever a link is not tranferring data.

In addition to this, the link rate can also be configured by the controller. Taking into account that a higher link rate means a higher power consumption, ports can be configured with the most appropriate rate considering the traffic rate that is transferred by the link. Finally, it is also necessary to take into account the power consumption associated to flow routing. In this case, this power consumption directly depends on the number of flows configured in the flow table of a switch, as a higher number of flow entries represents a higher number of flows that traverse the corresponding switch.

Different from previous works [1][2], which focus on power-minimization in SDN considering only the number of active links, in this work, we also consider all the previous factors that affect the switch power consumption. That is, in this paper, the problem of optimizing the power consumption in an SDN network is addressed by looking for the most appropriate set of active switches and links, their associated rates, and the number of flow entries at each SDN switch.

The major contributions of this paper are as follows:

- It has been developed an optimization program that minimizes the total switch power consumption considering all the previously presented factors.
- A heuristic algorithm to optimize the network power consumption is also implemented.
- The implemented heuristic is evaluated in different topologies.

The rest of this paper is structured as follows. In Section II, we introduce previous works about the use of SDN technology to reduce the network power consumption. In Section III, we introduce our model and present the formulation of our ILP model. The developed heuristic algorithm is presented in Section IV. Then, the simulation results are analyzed in Section V. Finally, in Section VI we present the conclusions.

II. RELATED WORKS

In [1], the authors faced the problem of optimizing the energy consumption of SDN networks. Such goal is achieved by an energy-aware routing approach that minimizes the number of active links used to route a given traffic demand in SDN. They propose an Integer Linear Program (ILP) model, as well as a heuristic considering the traffic routing requirements. In [2], the previous model is improved. In addition, results showed that the heuristic algorithm converges much faster and it can handle larger network sizes for which the exact model cannot find solutions in reasonable time.

A state-of-the-art study of energy efficiency strategies in SDN is presented in [3]. Some research works, such as [4] and [5] investigate the power consumption of SDN-related system through measurement studies. For instance, the authors of [4] derived power consumption models based on the measurements of two OpenFlow switches, considering the effect of configuration, management, and the managed traffic. In [5], the authors analyze the implications of different software data planes on the power efficiency, as part of Network Function Virtualization (NFV) implementations.

On the other hand, other research works [6][7] explore energy eficiency in SDN-enabled Data Center Networks (DCN). In a typical data center, energy consumption mainly consists of two parts: servers and network devices. In [6] authors investigate how to minimize the energy consumption by carefully scheduling the multi-path routing, according to the data center traffic demands. They take into account the TCAM (Ternary Content-Addressable Memory) size limitation and formulate the problem into an Integer Linear Problem. In [7], the authors study data center energy optimization with joint consideration of Virtual Machine placement and forwarding rule placement. They also formulate the problem in the form of ILP and propose a low-complexity two-phase heuristic algorithm.

 ISBN: 978-1-61208-602-6

III. ENERGY-AWARE APPROACH

According to [8], the power consumption of an OpenFlow switch (P_{switch}) is modeled as:

$$P_{switch} = \quad P_{base} + P_{config} + P_{control} + P_{OF} \qquad (1)$$

It consists of the base power P_{base}, the power of the configuration of the switch P_{config} (it depends on the number of active ports and the configured rates), the power consumption of the control traffic $P_{control}$ (it depends on the rate of outgoing *PacketIn* messages -they are a way for the switch to send a captured packet to the controller-, the rate of incoming *FlowMod* messages -they allow the controller to modify the state of an OpenFlow switch-, and their respective energy consumption per packet), and the power consumption of the processed OpenFlow traffic P_{OF} (it depends on the matches and actions of active OpenFlow rules).

A. Network Model

In this paper, we consider a topology as an undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ includes both the host set $\mathcal{H}$ and the set $\mathcal{V}$ of *candidate* SDN-enabled switches, i.e., $\mathcal{N} = \mathcal{H} \cup \mathcal{V}$, and edge set $\mathcal{E}$ represents the set of *candidate* links between the nodes in $\mathcal{N}$.

We also consider a set of offered unicast demands $\mathcal{D}$ between the hosts in $\mathcal{H}$. For each demand d, h_d denotes its known offered traffic.

B. Formulation

We are interested in creating an algorithm that solves the problem that finds (i) the SDN-enabled switches to be activated in the network, (ii) the links to be installed, (iii) their capacities, and (iv) how the traffic must be routed over the links. The optimization target is to minimize the total power consumption of SDN switches in the network.

The problem is formulated as follows:

- Input parameters:
  - $\mathcal{H}$: Set of network hosts.
  - $\mathcal{V}$: Set of *candidate* SDN-enabled switches.
  - $\mathcal{E}$: Set of *candidate* network links. From this information, $\delta^+(v)$ denotes the set of candidate links outgoing from node v, and $\delta^-(v)$ the set of candidate links to v.
  - $\mathcal{D}$: Set of offered unicast demands.
  - $h_d, d \in \mathcal{D}$: Offered traffic of a demand d. From this information, $a(d)$ denotes the origin host of a demand d and $b(d)$ the destination host.
  - P_{base}: The base power of SDN-enabled switches.
  - $P_{config-port}$: The power associated to the activation of both extreme switches ports of a link. It is a term of contributing to the overall P_{config}.
  - $P_{config-speed}$: The power associated with the configured speed rate. It is a term of contributing to the overall P_{config}.
  - $P_{OF-control}$: The power associated to the control and OpenFlow traffic.
  - U: Maximum capacity of a link.
  - M: An auxiliary really big number.
- Decision variables:
  - $z_e, e \in \mathcal{E}$: 1 if candidate link e is actually installed, and 0 otherwise (there is no link there, and then the capacity of this candidate link must be zero).
  - $u_e, e \in \mathcal{E}$: Capacity of candidate link e.
  - $x_{de}, d \in \mathcal{D}, e \in \mathcal{E}$: Traffic of demand d that traverses candidate link e.
  - $\hat{x}_{de}, d \in \mathcal{D}, e \in \mathcal{E}$: 1 if demand d traverses candidate link e, and 0 otherwise.
  - $x_v, v \in \mathcal{V}$: 1 if candidate node v is actually installed, and 0 otherwise.
  - $d_v, v \in \mathcal{V}$: Number of OpenFlow flows installed in candidate switch v.
- Formulation:

$$\min \quad (P_{base} \sum_v x_v + P_{config-port} \sum_e z_e + \qquad (2a)$$
$$+ P_{config-speed} \sum_e u_e + P_{OF-control} \sum_v d_v),$$

subject to:

$$\sum_{e \in \delta^+(v)} x_{de} - \sum_{e \in \delta^-(v)} x_{de} = \begin{cases} h_d, & \text{if } v = a(d) \\ -h_d, & \text{if } v = b(d) \\ 0, & \text{otherwise} \end{cases},$$
$$\forall d \in \mathcal{D}, \forall v \in \mathcal{V} \qquad (2b)$$

$$\sum_d x_{de} \leq u_e, \quad \forall e \in \mathcal{E} \qquad (2c)$$

$$u_e \leq U z_e, \quad \forall e \in \mathcal{E} \qquad (2d)$$

$$\sum_{e \in \delta^+(v)} u_e + \sum_{e \in \delta^-(v)} u_e \leq M x_v, \quad \forall v \in \mathcal{V} \qquad (2e)$$

$$x_{de} \leq M \hat{x}_{de}, \quad \forall d \in \mathcal{D}, \forall e \in \mathcal{E} \qquad (2f)$$

$$\sum_{e \in \delta^+(v)} \sum_d \hat{x}_{de} = d_v, \quad \forall v \in \mathcal{V} \qquad (2g)$$

$$u_e \geq 0, x_{de} \geq 0, d_v \geq 0, \quad \forall d \in \mathcal{D}, e \in \mathcal{E}, v \in \mathcal{V} \qquad (2h)$$

$$z_e, \hat{x}_{de}, x_v \in \{0, 1\} \quad \forall d \in \mathcal{D}, e \in \mathcal{E}, v \in \mathcal{V} \qquad (2i)$$

The objective function (2a) minimizes the total power consumption of SDN switches in the network.

Constraints (2b) are the flow conservation constraints. Constraints (2c) mean that for each link, the traffic carried in the link is less or equal than its capacity (and thus, no link is oversubscribed). Constraints (2d) make that (i) if a candidate link e is off ($z_e = 0$), then there cannot be a capacity in it ($u_e = 0$), and (ii) if a candidate link is on ($z_e = 1$), the link capacity is limited to U.

Constraints (2e) make that if a candidate switch v is off ($x_v = 0$), then the capacity of its outgoing and incoming links has to be equal to zero ($u_e = 0$). Constraints (2f) make that if a demand d does not traverse a link e, then the associated traffic of the demand has to be equal to zero ($x_{de} = 0$).

Constraints (2g) set the number of OpenFlow flows installed in candidate switch v. Finally, constraints (2h) forbid that a link carries a negative amount of traffic of a demand, since this has no physical meaning.

 ISBN: 978-1-61208-602-6

Require: $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ network graph, $\mathcal{D}$ data traffic demands
Ensure: $\mathcal{V}$ active switches, $\mathcal{X}$ active links, $\mathcal{U}$ links capacity, $\mathcal{F}$ flows in each switch

```
1:  for all demands ∈ D do
2:      if first demand then
3:          path=SHORTESTPATH(demand)
4:          CALCULATEPOWER(path, demand)
5:          UPDATE(V, X, U, F)
6:      else
7:          P=ALLPOSSIBLEPATHS(demand)
8:          for all paths ∈ P do
9:              if AVAILABLECAPACITY(path) then
10:                 CALCULATEPOWER(path, demand)
11:             end if
12:         end for
13:         SELECTMINIMUMPOWERPATH(P)
14:         UPDATE(V, X, U, F)
15:     end if
16: end for
```

Figure 1. Heuristic algorithm

```
1: for all links ∈ path do
2:     UPDATEACTIVELINKS(link, demand)
3:     UPDATEINSTALLEDFLOWS(demand)
4:     if origin(link) or destination(link) ∉ V then
5:         UPDATEACTIVESWITCHES( )
6:     end if
7:     UPDATETOTALPOWER(demand)
8: end for
9: return totalPower
```

Figure 2. CALCULATEPOWER(path, demand)

This optimization problem has been implemented in a network planning tool that is publicly available through GNU public license, called Net2Plan [9]. The contributed Java Optimization Modeler (JOM) library allows Java to interface from Net2Plan with GLPK and IPOPT optimization engines. In this way, it is possible to create and configure custom optimization problems within the Net2Plan environment.

However, the previous optimization problem becomes challenging to be solved even on medium-scale topologies. This is because the difficulty of this problem is NP-Hard, so the consumption of resources grow exponentially with the network size. To solve this problem in an alternative way, we develop a heuristic algorithm in the next section.

IV. HEURISTIC ALGORITHM

The proposed algorithm is shown in Figure 1. The main loop of the algorithm consists in determining the minimum power path for each traffic demand, that is, the path that suposses a minimum increment in the total power consumption in the network. In the case of the first demand (line 2), the selected path is the shortest one. Then, the power consumption associated to the configuration of this path in the network is calculated, and the sets of active switches ($\mathcal{V}$), active links ($\mathcal{X}$), links capacities ($\mathcal{X}$), and flows in each switch ($\mathcal{F}$) are updated.

For the rest of traffic demands (line 6), first of all, the set of admissible paths associated with each demand is obtained. Among all these paths with enough capacity to establish the new connection, we select the path that has a lower increment in the total power consumption in the network (line 13). Then, the sets of active switches ($\mathcal{V}$), active links ($\mathcal{X}$), links capacity ($\mathcal{X}$), and flows in each switch ($\mathcal{F}$) are updated.

Figure 2 shows the method to obtain the power increment that supposes the establishment of a specific demand in a certain path. Line 2 considers if a new link must be activated or an existing link needs to increment its configured speed. Line 3 considers the energy consumption required by the adding of new flow entries in the switches that form the path. Line 5 takes into account if it has been necessary to activate new switches.

V. SIMULATIONS AND RESULTS

To evaluate the performance of our algorithm, a simulator in Python language has been developed. Network topologies are obtained using the NetworkX [10] library. NetworkX is a Python software package that is used for the creation and study of the structure and performance of complex networks. Our tests are performed in two different topologies, one with $|\mathcal{V}| = 20$ candidate switches and another with $|\mathcal{V}| = 70$. On the other hand, the traffic demands have been obtained from SNDlib library [11].

Figure 3 shows the total power consumption in a network with 70 nodes, as a function of the number of demands. In this figure, we can observe that the power consumption using the heuristic is lower than the power consumed when all the swiches and links are active. The saved power values are between 25% and 50%. On the other hand, in this figure it is clear that the increment in the number of demands represents an increment in power consumption.

Figure 4 shows the total power consumption in a network with 20 nodes. In this figure, we can observe the same patterns as in Figure 3. In this figure we can see how the heuristic continues to give us better performance regarding scenarios where only the links that are not used are disconnected. When the number of demands is small, the saved power values are similar to those presented with $|\mathcal{V}| = 70$. However, when the number of demands increases, it can be seen that the percentage of saved power is lower in the case $|\mathcal{V}| = 20$. This is due to the fact that in the case of $|\mathcal{V}| = 20$, as demand between nodes increases, most nodes in the network must be activated to supply such demand. While in the case of $|\mathcal{V}| = 70$, having more nodes available, we can try to route the demands in a higher number of available routes, giving us a higher percentage of saved power.

Figure 6 shows the increment in the total power when new demands are activated in the network. It can be seen that when the first demands are established, the increment in the total power is high. This is because it is the necessary to activate several switches to connect the source and the destination nodes. As new demands are established, the switches have already been activated, so the required power is not so high. There are some peaks in the last demands, though. This is because there is a new source or destination node that has not been activated before, or that simply the links that were already connected cannot give the necessary performance to satisfy the demands and it is necessary to look for another route with a new switch that was not connected.

 ISBN: 978-1-61208-602-6

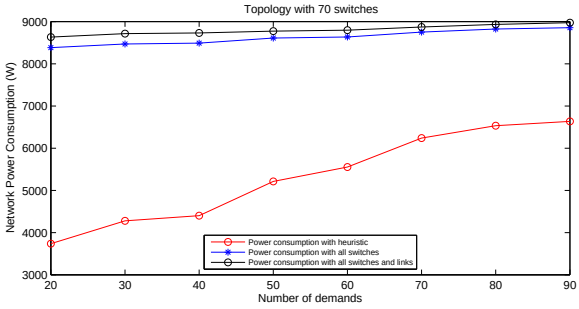

Figure 3. Total power consumption in a network with 70 nodes.

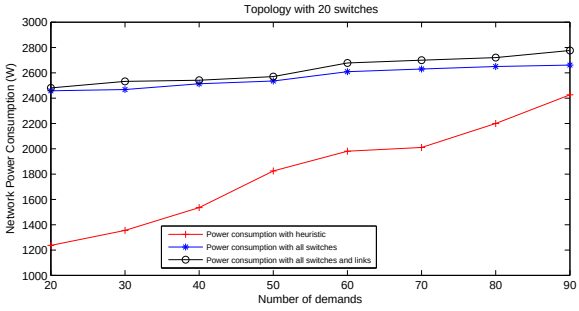

Figure 4. Total power consumption in a network with 20 nodes.

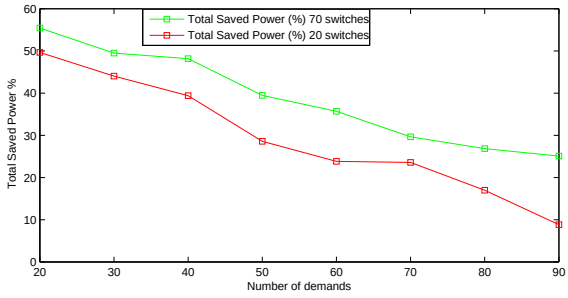

Figure 5. Percentage of saved power.

VI. Conclusion

In this paper, we have proposed an approach that minimizes the total power consumption in SDN networks. To achieve this, first of all an optimization program has been formulated. In this program, different constraints have been modeled and implemented. We have also developed a heuristic algorithm that has been evaluated in two different topologies with real traffic demands. Based on experimental simulations, we have proved that our apporach achieves energy savings of up to 50%. As future work, we plan to implement this heuristic in Smart Cities environments, where SDN is a key element.

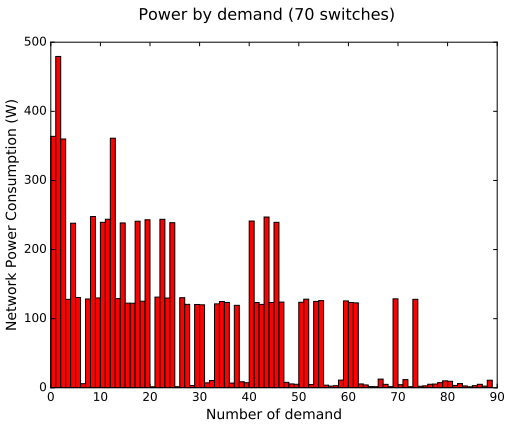

Figure 6. Increment in the total power when new demands are activated.

Acknowledgment

This research has been supported by the AEI/FEDER,UE Project Grant TEC2016-76465-C2-1-R (AIM). Adrian Flores de la Cruz also thanks the Spanish Ministry of Economy, Industry and Competitiveness for a FPI (BES-2014-069097) pre-doctoral fellowship.

References

[1] A. Fernandez-Fernandez, C. Cervello-Pastor, and L. Ochoa-Aday, "Improved energy-aware routing algorithm in software-defined networks," in *2016 IEEE 41st Conference on Local Computer Networks (LCN)*, Nov 2016, pp. 196–199.

[2] ——, "Achieving energy efficiency: An energy-aware approach in sdn," in *2016 IEEE Global Communications Conference (GLOBECOM)*, Dec 2016, pp. 1–7.

[3] B. G. Assefa and O. Ozkasap, "State-of-the-art energy efficiency approaches in software defined networking," in *SoftNetworking'15*, April 2015, pp. 268–273.

[4] F. Kaup, S. Melnikowitsch, and D. Hausheer, "Measuring and modeling the power consumption of openflow switches," in *10th International Conference on Network and Service Management (CNSM) and Workshop*, Nov 2014, pp. 181–186.

[5] Z. Xu, F. Liu, T. Wang, and H. Xu, "Demystifying the energy efficiency of network function virtualization," in *2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS)*, June 2016, pp. 1–10.

[6] D. Zeng, G. Yang, L. Gu, S. Guo, and H. Yao, "Joint optimization on switch activation and flow routing towards energy efficient software defined data center networks," in *2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6.

[7] H. Y. et al.i, "Joint optimization of vm placement and rule placement towards energy efficient software-defined data centers," in *2016 IEEE International Conference on Computer and Information Technology (CIT)*, Dec 2016, pp. 204–209.

[8] F. Kaup, S. Melnikowitsch, and D. Hausheer, "Measuring and modeling the power consumption of openflow switches," in *10th International Conference on Network and Service Management (CNSM) and Workshop*, Nov 2014, pp. 181–186.

[9] P. P.-M. et al., "Net2plan-aire: gained experience in an ad-hoc sdn development for a metro carrier network," in *17th International Network Strategy and Planning Symposium (Networks 2016)p*, 2016, pp. 181–186.

[10] "Networkx," http://networkx.github.io/, accessed: June 2017.

[11] "Sndlib," http://sndlib.zib.de/, accessed: June 2017.

 ISBN: 978-1-61208-602-6