Green SDN: A Power Optimization Approach

Gokcan Cantali, *2016700027,* and Cem Ersoy, *Professor*

***Abstract*—The energy consumption problem is getting bigger as people keep depending on energy exploiter products and services. With the continuous development of the Internet, data center networks have become one of such energy consuming products. This project focuses on data center networks that use Software-Defined Networking (SDN), and proposes an approach to optimize the energy consumption in such environment.**

I. Introduction

As the need to access the Internet keeps increasing every day, organizations and companies are searching better ways to provide their clients services through the Web. As the number of their clients scale up, such organizations face difficulties regarding the network configuration and traffic congestion in their data centers. With the emergence of Software-Defined Networking (SDN), programmable network devices along with decoupling of data plane and control plane begin to attract data center owners. Therefore, many network designers have started to replace their traditional networking architecture with an SDN architecture for such easiness.

Although SDN can provide more manageability and flexibility for controlling forwarding devices (e.g. routers, switches etc.), there still exists an important issue that is available in any type of network design: energy efficiency. Nowadays, energy consumption is becoming concern of everyone since more energy consumption leads to more emissions of CO_2, increasing the effect of global warming and triggering climate change. Furthermore, increasing amounts of energy consumption will eventually lead to depletion of non-renewable energy resources. Moreover, more energy consumption also results in more maintenance and operation cost, leading to an unpleasant situation for the organization. Therefore, many companies are now trying to reduce their power consumption[1] by taking measures. One of such measures is to reduce the power consumed by the operational network and its devices. The approach of minimizing energy consumption of the network as much as possible is also called Green Networking.

In this project, we[2] try to find an optimum solution to the problem of power minimization of a network in SDN environment. Since data centers use huge amounts of energy to provide services [1], they are main focus of the project. Thus, we build a model representing the network of a data center and based on the model, we implement an algorithm to find the optimal (or near optimal in some cases) design that minimizes energy consumption.

[1]Since such companies keep their services active all the time, power reduction implies energy reduction in the context. Hence, 'power' will be used interchangeably with 'energy' hereafter

[2]For the sake of formalism, 'we' is used instead of 'I' even though only one person is working on this project right now

II. Background and Related Work

Many researchers have shown interest for constructing a Green SDN environment by proposing energy-efficient frameworks which do not have much performance degradation. Based on the study in [2], energy consumption optimization approaches can be divided into main four category: traffic-aware solutions, TCAM (Ternary Content Addressable Memory) compacting solutions, rule placement solutions and end host aware solutions. Since the most popular one among these categories is the traffic-aware energy efficiency approach, we focus on the related work regarding such solutions.

One of the studies regarding power consumption minimization is conducted by Heller et al. in [3]. The scheme called ElasticTree shuts down the network devices which are not used. The study proposes three different scheduling algorithms that aim to minimize the energy consumption. The first one is a formal model and uses Mixed Integer Programming (MIP). The second is a greedy algorithm and called Greed Bin-Packing. The third is a topology-aware heuristic they have developed. Even though the MIP solution provides the best value, its complexity is not low enough to scale up to big data center networks. The heuristic solution has linear complexity and gives sub-optimal values for the objective function. Nevertheless, it fails at some peak points where traffic increases.

Another solution to the energy efficiency problem is given in the study [4]. The authors propose a correlation-aware power optimization framework, called CARPO, which adjusts the flows on the links and shuts down the unnecessary devices. Although they build a MIP model to find a near-optimal solution, the complexity of the problem make the solution hard to use in real applications. Therefore, they also develop a heuristic based on the greedy bin-package algorithms. The approach of shutting down the unnecessary network components is similar to the one used in ElasticTree. However, CARPO also considers the correlation among the flows to save more energy. Moreover, CARPO adopts link rate adaptation to reduce the power consumption even further. As a consequence, CARPO manages to outperform ElasticTree by 19.6%.

Bolla et al. [5] proposes a different approach, compared to ElasticTree and CARPO, for optimizing energy consumption in a data center network. The idea in the study is simply as follows: First, the algorithm pre-computes the all the possible paths between each pair of nodes. During the process, the nodes in the paths are stored in the system, along with how many times they appear in a path. The nodes are ordered based on the number of paths in which they appear. Then, the algorithm retrieve nodes from the list one by one, and decide whether they can handle the traffic requirements under consolidation. Nodes with such capability are chosen until all the network traffic requirements are satisfied. Consolidation is then applied to these selected nodes, and test results show that energy savings can reach to around 80%.

The problem of optimizing power consumption while maintaining the flow requirements has even attracted the undergraduate students. In the bachelor thesis [6], Schaap et al. build a model to represent a data center network in SDN environment. They formulate the network problem based on linear programming and implement a scheduling algorithm which uses Gurobi as the linear solver. They also use an Open Shortest Path First (OSPF) algorithm for routing as a baseline to compare their linear programming scheduling algorithm. Based on the results, the linear programming solution saves up to 30% compared to the OSPF algorithm. However, since the linear programming complexity is not low enough to apply to real world applications, the scalability of the solution remains as a question.

III. Problem Formulation

Even though there are various different network architectures, one of the most popular among data centers is the FatTree topology (refer to [7] and [8] for more details). Many organizations, especially the ones adopt SDN, use FatTree network architecture in their data centers. With the intention of running a simulation as close as to the real applications, we build our model on a SDN environment with FatTree topology. On top of our model, we implement a network design tool which takes the given workload parameters as input, calculates the minimum possible energy consumption and finds out how to achieve this objective without violating the constraints.

A. Workload Parameters

Workload parameters are the input variables given to our network design tool. The tool accepts the following workload parameters:

- N: The number of clients (hosts) that will be served in the network.
- k: The number of ports in a switch.[3]
- $[r_{i,j}]_{NxN}$: The traffic requirement matrix that shows which host will be sending how many packet per second to another host.
- C_l: The bandwidth capacity of the links. For simplicity, we assume that every link has the same capacity. Hence, from at this point, we refer to C_l as C.

B. Objective Function

Objective function is what we are trying to achieve in our model. In order to achieve a "greener" state regarding the network devices, our design tool aims to reduce the power consumption as much as possible:

$$min\ P_{total}$$

[3]The number of switches in a FatTree architecture depends on the number of ports in a switch. Therefore, there does not exist another variable that represents the number of switches

In our model, P_{total} is the sum of the consumed power by the switches:

$$P_{total} = \sum_{i=1}^{N_{activeswitches}} P_i^{switch} \tag{1}$$

Where $N_{activeswitches}$ is the number of active switches in the network and P_i^{switch} is the power consumed by only the switch i.

In on our model, the power consumption of a switch arises from its base power and the power consumed by its ports. The base power is the power used by a switch when it is online, even when it does not relay any packets. The port power depends on the rate of the flow on the port. Therefore, we can estimate the power consumption of a single switch as follows:

$$P^{switch} = P_{base} + \sum_{i=1}^{N_{ports}} P_i^{port} \tag{2}$$

Where N_ports is the number of ports in a switch, P_{base} is the power consumed by a switch when it is online and P_i^{port} is the power consumption by the port i in the switch. We calculate P_i^{port} by

$$P_i^{port} = P_{full}^{port} * \rho_i^{port} \tag{3}$$

Where P_{full}^{port} is the power consumed by a port when it uses all the bandwidth capacity of the link the port is tied to. ρ^{port} is the utilization of the port, i.e. the amount of bandwidth it is using compared to the capacity of the link.

Combining the equations (1), (2) and (3), we arrive at our objective function:

$$min \sum_{i=1}^{N_{activeswitches}} (P_{base} + \sum_{j=1}^{N_{ports}} P_{full}^{port} * \rho_i^{port}) \tag{4}$$

C. Decision Variables

Decision variables are the variables each of which will be given a value by our network design tool in order to achieve optimum value of the objective functions. In other words, the value of decision variables are decided by the optimizer tool such that it can find a more suitable value for the objective function.

The decision variables in our network design tool is the followings:

- $C_{i,j}$: Whether or not the core switch in jth place on the ith group is active. (Binary Variable)

$$\mathrm{C}_{i,j} = \begin{cases} 1 & \text{If the } jth \text{ core switch in } ith \text{ group is on} \\ 0 & \text{Otherwise} \end{cases}$$

- $A_{i,j}$: Whether or not the aggregate switch in jth place on the ith pod is active. (Binary Variable)

$$\mathrm{A}_{i,j} = \begin{cases} 1 & \text{If the } jth \text{ aggregate switch in } ith \text{ pod is on} \\ 0 & \text{Otherwise} \end{cases}$$

- $E_{i,j}$: Whether or not the edge switch in jth place on the ith pod is active. (Binary Variable)

$$\mathrm{E}_{i,j} = \begin{cases} 1 & \text{If the } jth \text{ edge switch in } ith \text{ pod is on} \\ 0 & \text{Otherwise} \end{cases}$$

- $f_{m,n}$: The flow from port m of a switch (or from host m) to port n of another switch (or to host n), based on number of packets per second. (Integer Variable)

D. Constraints

Constraints are the bounds for our decision variables. They are the set of equations and inequalities that limits the solution space of the problem. Our network design tool will try to find a minimum power consumption value, without going out of the borders set by the constraints.

The constraints in our model are the followings:

1) *Topology Constraint*: Our model dictates that the topology of the network must be a FatTree architecture. The solutions which cause the network architecture to be something different, will not be accepted.
2) *Link Capacity Constraint*: The flow on a link from m to n, i.e. flow between two switches or between a host and a switch, cannot be more than the capacity of the link:

$$f_{m,n} \leq C \quad \forall m, n \in HS \cup PS \tag{5}$$

Where HS is the set of all hosts and PS is the set of all ports (of all switches) in the network.

3) *Flow Constraint*: The flow on the links must satisfy the network requirement matrix. To put it more simpler, the amount of flow going into a switch should be the same amount of the flow going out of the same switch. Before focusing on the constraint, we first define the following notation:

$$L = HS \cup PS \tag{6}$$

Where HS and PS have the same definition as the previous constraint. Therefore, L is considered the set of all the endpoints of the links in the system. Now, we can write a more formal expression of the constraint as the following:

$$\sum_{n \in PS_i} \sum_{x \in L} f_{x,n} = \sum_{m \in PS_i} \sum_{y \in L} f_{m,y} \quad \forall i \in SS \tag{7}$$

Where PS_i is the set of all ports belonging to the switch i and SS is the set of all switches in the network.

IV. The Planned Solution

In order to solve the power minimization problem in a feasible time, we intend to use Simulated Annealing (SA) algorithm. Since SA is a non-deterministic method, the result might vary during different runs for the same case. In order to reduce the probability of encountering an edge case, we might run the algorithm multiple times for a single case, and accept the more frequent solution.

SA needs a neighborhood structure and a cooling schedule to be implemented. For now, the planned neighborhood structure is the following: two states are considered neighbour when a unit flow of a switch is moved to another switch. Similarly, only one switch is deactivated/activated between two states, they are also considered as neighbours. The cooling schedule is not clear as of now, though we plan to set the initial temperature to 99% and the stopping condition to an arbitrary number of iterations during which the optimal state has not changed. This arbitrary number, and the temperature decrement rate, are planned to be set depending on the input.

Even though we plan to use SA for the solution to our problem, we might face serious difficulties in the implementation. In such case, we might switch to developing our own heuristics for a simpler solution. Moreover, if time allows, we might consider implementing an integer programming solution to the problem as well. Since the integer (or linear) programming solution to our problem will likely have scalability issues, we might consider to use the solution only to compare the effectiveness of our main method (SA or our own heuristics).

References

[1] "US Data Center Energy Consumption In the Past and Prediction for the Future," https://www.nrdc.org/resources/americas-data-centers-consuming-and-wasting-growing-amounts-energy, accessed: 2016-10-18.

[2] B. G. Assefa and O. Ozkasap, "State-of-the-art energy efficiency approaches in software defined networking," in *ICN 2015 : The Fourteenth International Conference on Networks*, ser. ACM-ICN '15. New York, NY, USA: ACM, 2015, pp. 256–260.

[3] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "Elastictree: Saving energy in data center networks," in *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 17–17. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855711.1855728

[4] X. Wang, X. Wang, K. Zheng, Y. Yao, and Q. Cao, "Correlation-aware traffic consolidation for power optimization of data center networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 4, pp. 992–1006, 2016. [Online]. Available: http://dx.doi.org/10.1109/TPDS.2015.2421492

[5] R. Bolla, R. Bruschi, F. Davoli, and C. Lombardo, "Fine-grained energy-efficient consolidation in sdn networks and devices," *Network and Service Management, IEEE Transactions on*, vol. 12, no. 2, pp. 132–145, 2015.

[6] M. Schaap, "Saving energy in openflow computer networks," B.S. Thesis, University of Amsterdam, 1098 XH Amsterdam, Holland, 6 2015.

[7] C. E. Leiserson, "Fat-trees: Universal networks for hardware-efficient supercomputing," *IEEE Trans. Comput.*, vol. 34, no. 10, pp. 892–901, Oct. 1985. [Online]. Available: http://dl.acm.org/citation.cfm?id=4492.4495

[8] K. S. Solnushkin, "Automated design of two-layer fat-tree networks," http://arxiv.org/abs/1301.6179, January 2013, arXiv:1301.6179 [cs.DC].