An SDN-based energy-aware traffic management mechanism

Alex B. Vieira · Wallace Nascimento Paraizo · Luciano Jerez Chaves · Luiz H. A. Correia · Edelberto Franco Silva

Received: 17 February 2020 / Accepted: 3 June 2021 / Published online: 30 June 2021
© Institut Mines-Télécorn and Springer Nature Switzerland AG 2021

Abstract
Green computing is a central theme in many computer science areas, including computer networks. Dynamic solutions that can properly adjust network resources can prevent infrastructure over-provision and mitigate power consumption during low-demand periods. In this work, we propose DTM (Dynamic mechanism for Traffic Management), an energy-aware dynamic mechanism for traffic management, built upon the SDN paradigm. DTM continuously monitors the use of network links to concentrate traffic and disconnect idle equipment without degrading the offered quality of service. Our simulations show that the mechanism can save up to 46% of energy, on average, in the links' capacities of homogeneous and heterogeneous scenarios. In scenarios with average to high traffic demands, the mean energy savings are 36.72% and 17.86%, respectively. Compared to a well-known existing mechanism, our approach is up to 7% better for medium-demand scenarios, and approximately 4% better for high-demand scenarios.

Keywords Green computing · Power consumption · Software-defined networking · Traffic management

1 Introduction
Energy-aware computing is a central theme of research today, especially in computer networks [1, 2]. Resources are often overprovisioned to cope with demands and spike in network traffic, resulting in wasted resources and high power consumption. Part of this problem occurs due to the existing infrastructure of legacy networks, where stability is crucial to business success. In these networks, the control and routing are inseparable and embedded in network elements. In this sense, the adoption of new technologies and documents in production environments is virtually unviable [3].

The new software-defined network (SDN) paradigm makes a clear separation between data and control planes. A centralized view and the network programmability allow one to easily monitor network and to program its flow routing. SDN can assist in shutting down (or hibernating) network devices as a way to save power.

Note that there are several ways to save power in networks such as optimizing the network infrastructure, the dynamic switch on-off of network elements, the scaling of the frequency of network elements, and the controlling of the link capacity (i.e., data rate). However, support for some of these approaches is not common for commercial network devices. For example, scaling the frequency of network elements or controlling the link capacity was initially part of the IEEE 802.3az standard. However, due to the implementation challenges, the functionality for configuring Ethernet network elements in different data rates has been excluded from this standard final specification [4]. In this sense, most of the existing solutions to save power on networks rely on the optimizing of data routing.

In this paper, we present DTM: an Energy-Aware Dynamic Traffic Management. DTM employs SDN to improve traffic routing between switches. Briefly, DTM
monitors network flows and uses fewer routes as possible to transmit the existing end-to-end flows, which enables the disconnection of links and the turning off of idle network devices. Moreover, to avoid the recurrent connection/disconnection of devices—which may shorten equipment life—our mechanism uses a traffic aggregation policy. However, a policy only aggregates flows, allowing link disconnection, in case its utilization level does not indicate a future increasing demand.

We emphasize that the existing works in the literature focus on topology optimization and the minimization of the number of links and switches. Existing solutions do not consider the dynamics of flows [5–8]. In fact, only a few works like [9] and [7] explore the seasonality of traffic.

We have evaluated DTM by emulating an SDN environment from a typical university campus topology (Section 5). We consider different traffic levels as well as distinct overloaded scenarios. Our evaluation results evidence an energy-saving from 17 to 46%, depending on the emulated network demand. Compared with an existing solution, our results are, on average, 7 to 4% better for medium- and high-demand scenarios, respectively.

In sum, our contributions are the following:

- We propose an energy-aware dynamic mechanism for traffic management.
- The mechanism has been evaluated through the emulation of networks using Mininet, considering a realistic topology with homogeneous and heterogeneous links’ capacities.
- Our evaluation results show that our mechanism provided an average energy saving of approximately 46% in a scenario with low network demand, similar to a nighttime pattern.
- In scenarios with average to high traffic demands, the mean energy savings are 36.72% and 17.86%, respectively.
- Compared to a well-known existing mechanism, our approach is up to 7% better for medium-demand scenarios and approximately 4% better for high-demand scenarios.

The remainder of this article is organized as follows. In Section 2, we discuss the related work. In Section 3, we introduce the common scenario we consider in this work. In Section 4 we describe the protocol we propose. Evaluation and results of our proposal are presented in Section 5. Finally, Section 6 concludes this work.

2 Related work

The topic of energy-saving is worthy of investigation [1, 2]. Traditionally, energy-saving works on computer networks have focused on mathematical models of topology and minimizing attributes (e.g., number of links, switches, or CPU frequency) [5–8]. The authors of [10] formulate an optimization model to minimize the number of network links and network assets, choosing the position of the controller. According to them, these are key issues for energy saving. Many of these formulations involve heuristics, once resource optimization in this environment is considered an NP-Complete problem. The authors of [6] and [8] propose heuristics for disconnecting aggregated link network cards, which can be disabled independently. The authors do not turn off the entire equipment or an entire link because they believe this reduces network connectivity. In [11], the authors consider bandwidth, link load, and traffic arrays to hibernate switches, rearranging the network topology as needed.

Fernández-Fernández et al. [10] formulate an optimization model that considers the routing requirements for data and control planes. This model considers the reduction of network assets and also considers driver location, which, according to the authors, is a key issue for energy savings.

These problem formulations often involve heuristics because resource allocation optimization in this environment is an NP-Complete problem. In [6] and [8], the authors proposed heuristics to reduce power consumption by disconnecting aggregated link network cards, which can be disabled independently. The authors do not turn off equipment or entire links because they believe this reduces network connectivity. Habibullah et al. [11] investigate the reduction of network infrastructure power consumption through the hibernation of switches, rearranging the network topology as needed. For this, they consider bandwidth, link load, and traffic arrays as input parameters for their decision algorithms.

Rather than rearranging the network topology or shutting down the networking assets, the authors of [5] adjust the speed of underused links. The lower priority traffic is redirected and real-time traffic is kept on the minimum path to satisfy the desired QoS. Note that the reduction in power consumption is achieved only by reducing link speeds, resulting in marginal gains. Moreover, also note that link speed reduction is not commonly supported on commercial devices.

Only a few works consider the dynamics of the network. Clearly, nowadays, network dynamics and infrastructure is a topic that must be taken into account for many purposes, from energy-saving to the coexistence of multiple high-performance distributed applications [12]. In this context, Heller et al. propose the ElasticTree, a network power manager for datacenter that monitors traffic and chooses the set of network elements that must remain active to meet the goals of performance and fault tolerance. The ElasticTree turns off unnecessary links and switches as
much as possible. For this, a formal model, a greedy algorithm, a topology-sensitive heuristic, and forecasting methods were adopted. In the same way, the authors of [7] present a model and a greedy heuristic to find a minimal path concerning the number of hops and an increase in energy consumption.

In this sense, Markiewicz et al. [7] consider that network elements can be turned off. The power consumption minimization problem is formulated using an IP model, and the objective function jointly minimizes the sum of the power consumption of the links and the switches. The authors introduced four greedy heuristics algorithms to solve the problem, namely, shortest path first, longest path first, smallest demand first, and highest demand first. The algorithms are evaluated over two sample topologies, a campus network and a mesh network for low, medium, and high traffic. Experimental results show that the longest path first gives close to optimal energy savings of up to 35% and is better than the other three algorithms for both topologies.

Recently, a number of works use SDN to save energy in networks, especially in cloud-based environments. Son et al. [13] also avoid overbooking of resources, and as a consequence, save energy, through the use of SDN. Authors, in this case, strategically allocate a more precise amount of resources to VMs and traffic (in SDN-Based Cloud Data Centers) according to the demand of Quality of Service (QoS). Despite the similar objective (the energy-saving), DTM focuses on network resources and topology, while Son et al. focus on the allocation of Cloud Data Centers resources. In a different way, Xu et al. [14] schedule the flow order to save energy in an SDN network compliant environment. In this way, they can optimize the selection of links, avoiding defragmentation. Finally, Jia et al. [15] also use an SDN approach to re-route flows. Initially, they treat the problem as a min-cost problem, which is NP-hard. Moreover, they do not dynamically define network link utilization levels, which may induce network to shift between on-off link states, in a ping-pong effect. Finally, we highlight that we provide close-to-real implementation, instead of simulation.

It is possible to see on Table 1 a comparison between all related works. We also highlight both Markiewicz et al. and our proposal, once this work will be used to compare directly with our proposal ahead. Differently from DTM, most existing works do not consider the energy-save characteristics or only present costly solutions that rely on specialized mechanisms. Moreover, these studies only reassess the proposed optimization/heuristic in the face of changes in flows but do not consider a smoothing interference with network assets or possible traffic aggregations. Thus, they only disconnect links and switches when they become naturally idle.

| Work                        | Environment | Traffic             | Topology | Dynamicity | Switch | Link/Port |
|-----------------------------|-------------|---------------------|----------|------------|--------|-----------|
| Markiewicz et al. [7]       | Testbed     | Real and Synthetic  | Mesh     | X          | X      | X         |
| Fisher et al. [6]           | CPLEX       | Real and Synthetic  | Abeline and Hierarchical | X | – | – |
| Lin et al. [8]              | CPLEX       | Real and Synthetic  | Abeline and Hierarchical | X | – | – |
| Fernández-Fernández et al. [10] | Testbed | Real | SNDLib | X | – | X |
| Sasaki et al. [5]           | Simulation  | Real and Synthetic  | Random   | X          | – | – |
| Son et al. [13]             | Simulation  | Real and Synthetic  | Mesh     | X          | – | – |
| Xu et al. [14]              | Simulation  | Synthetic           | Fat-Tree and BCube | X | X | X |
| Oliveira et al. [12]        | Simulation  | Synthetic           | Fat-Tree and BCube | X | – | – |
| Jia et al. [15]             | Simulation  | Real                | Rocketfuel | X | X | – |
| Our proposal                | Simulation  | Real and Synthetic  | Mesh     | X          | X | X |
switch \( c \) directly attached to a host \( i \in \mathcal{N}_h \); and \( \mathcal{N}_c \) the forward switch, where \( \forall l_{cj} \in \mathcal{L} \) connects the switch \( c \) directly to another switch \( j \in \mathcal{N}_c \).

DTM reduces the power consumption by shutting down (or hibernating) selectively ports on switches that have their idle links. If all ports on a switch are turned off, the switch can also be turned off. For this, traffic management algorithms need to concentrate flows on a minimum number of links, but without degrading the quality of service offered. This strategy is called resource consolidation, according to [16].

For all switch \( c \in \mathcal{N}_c \), denote \( E_t(c) \) as the energy consumption of \( c \) at time \( t \), following Eq. 1 [5]:

\[
E_t(c) = E_{b_t}(c) + E_{p_g}(c) \times N_{p_g}(c) + E_{p_f}(c) \times N_{p_f}(c)
\]

where \( E_{b_t}(c) \) shows the base consumption of \( c \) at time \( t \), necessary to keep the switch working (processor, cooler, etc.); \( E_{p_g}(c) \) and \( E_{p_f}(c) \) show the energy consumption for each port of 1 Gbps and 100 Mbps in \( c \), respectively; \( N_{p_g}(c) \) and \( N_{p_f}(c) \) represent the number of 1-Gbps and 100-Mbps ports active in \( c \) at time \( t \), respectively. Note that the energy consumption of each port changes according to its speed [7]. Then, the energy consumption can be estimated for network total amount \( E_t(\mathcal{G}) \) at time \( t \) following Eq. 2:

\[
E_t(\mathcal{G}) = \sum_{c \in \mathcal{N}_c} E_t(c)
\]

4 DTM: a dynamic traffic management

In this article, we propose an SDN-based energy-aware traffic management mechanism. This DTM engine presents three main components: (i) active network monitoring, which maintains an up-to-date network resource usage model; (ii) the new flows installation algorithm, which reactively identifies new traffic and allocates appropriate routes; and (iii) the active stream redirection algorithm, which aggregates flows in the least amount of links to shut down idle resources, avoiding to overload the remaining links.

In what follows, we detail each one of these components.

4.1 Active network monitoring

DTM uses the OpenFlow controller to access network information. In this sense, the network controller maintains a detailed and up-to-date network view, including information on resource use and energy consumption of active elements. Since DTM can shut down and reconnect links and switches over time, we denote as \( \mathcal{G}_t' \subseteq \mathcal{G} \) nodes and links set that is on and available for use in an instant \( t \). To keep the model up to date, the controller periodically sends OpenFlow messages to the switches requesting information and statistics. The \( \Delta t \) interval between requests can be adjusted to a good compromise between model update and network overhead. With the answers, it is possible to update \( \mathcal{G}_t' \), the estimated power consumption \( E_t(\mathcal{G}) \), the instantaneous throughput of data streams and also the utilization rate of network links. This information is posteriorly used by the new flow installation algorithm and active flow redirection algorithm.

Let \( U_t(l) \) be the link utilization rate \( l \in \mathcal{L} \), calculated by the bandwidth ratio at the instant \( t \) for the transmission capacity of \( l \). In this work, we use Table 2 as a reference to classify the link utilization and any number of link states could be used. This categorization is used by algorithms to prevent overloaded links from receiving new flows, avoiding packet loss. Besides, low-load links prove to be favorable candidates for shutdown ports on switches. Thus, the mechanism aims to reallocate all flows of these links to alternative routes, making them idle and energy-saving. We

| State \( S_t(l) \) | Link state | Utilization rate \( U_t(l) \) |
|------------------|------------|-----------------------------|
| \( s_0 \)        | Low load   | \( 0\% \leq U_t(l) < T_{0 \to 1} \) |
| \( s_1 \)        | Normal load| \( T_{0 \to 1} \leq U_t(l) < T_{1 \to 2} \) |
| \( s_2 \)        | High load  | \( T_{1 \to 2} \leq U_t(l) < T_{2 \to 3} \) |
| \( s_3 \)        | Overload   | \( T_{2 \to 3} \leq U_t(l) \leq 100\% \) |

Table 2 \( S_t(l) \) state as a function of \( U_t(l) \) use
let as future work the investigation of the impact of varying
the possible number of link states.

Once the controller collects information of network
elements (links) and updates the network model, it builds
the $P_{ij}$ set with the $k$ minimum paths connecting host $i$ to
$j$, $\forall i, j \in N_h$. The controller also calculates, $\forall c \in N_c$, the $Q(c)$ indicator representing the total paths in the $P$ sets passing the $c$ switch. Minimum paths and the $Q(c)$ indicator make sense when viewed from an energy-saving perspective because the shorter and more concentrated the paths, the fewer network elements are used and the more switches can be turned off.

4.2 Flow installation

The new flow installation by the controller is reactive. In
other words, when a host “a” starts a new flow to host
“b,” the network controller detects this new flow routes
accordingly.

Let $f_{ab}$ be a flow between hosts $a, b \in N_h$. When the
first packet of $f_{ab}$ reaches the access switch, the controller
is notified and starts searching for the best path $p_{ab}^{best} \in P_{ab}$ with all links in state $S_i(l) \leq s_{max}$, as per Algorithm 1. The
initial search is restricted to paths that do not use overloaded
links ($s_{max} = s_2$). The algorithm prioritizes the optimal
path $p_i$ that can be accommodated in $G'_i$ and does not result
in increased power consumption. In the case of several
ideal paths, the shortest one is prioritized $|p_i|$, having as a
tiebreaker the indicator $Q(p_i) = \sum_{c \in P_{ab}} Q(c)$ . If there is
no $p_i \in G'_i$, then the energy impact $E_i(p)$, $\forall p \in P_{ab}$, and
choose the $p_e$ path that results in the smallest final energy
increment. In this case, we will need to update $G'_i$, enabling
links or switches to accommodate the new flow.

Algorithm 1 PATHFINDER.

| Input: $P_{ab}$ ways; Active subgraph $G'_i$; $s_{max}$ state. |
| Output: best path $p_{ab}^{best} \in P_{ab}$ with $S_i(l) \leq s_{max}$ links. |
| $p_i \leftarrow \infty$; $p_e \leftarrow \infty$; |
| foreach $p \in P_{ab}$ do |
| if $\forall l \in p / S_l(l) \leq s_{max}$ then |
| if $p \in G'_i$ then |
| if $(|p| < |p_i|)\ OR(|p| = |p_i|)\ AND(Q(p) > Q(p_i))$ then |
| $p_i \leftarrow p$; |
| if $E_i(p) < E_i(p_e)$ then |
| $p_e \leftarrow p$; |
| if $p_i \neq \infty$ then return $p_i$ else return $p_e$; |

In case it does not exist a path $p_{ab}^{m}$ with at least one
overloaded link, the controller relaxes the maximum state of
link constraint ($s_{max} = s_3$), and perform a new search for
$p_{ab}^{m}$. Once the path is set to $f_{ab}$, flow rules are installed on
all $c \in p_{ab}^{m}$ switches.

4.3 Active flows redirection

The new flow installation considers the instantaneous link
utilization rate. However, traffic behavior over time can lead
to undesirable use in some links (i.e., link overloading). In
this case, the active flow redirection algorithm redistributes
traffic across the network. To do this, it evaluates the
$S_i(l)$ state that each link can assume, as shown in Fig. 2
(which reflects the states we previously defined in Table 2).
In this work, we assume that flows allocated to a link
in an overload state $S_i(l) = s_3$ may be penalized with
longer delays and packet losses. In this case, the algorithm
redirects from this to other links the smallest amount of
flows required for the utilization rate to decrease to some
state $S_i(l) < s_3$. This strategy prevents links from remaining
overloaded, reducing possible damage to network QoS
indicators.

On the other hand, we also consider that links at the
low charge state $S_i(l) = s_0$ are underutilized. In this
case, the algorithm tries to redirect all of these flows to
other links in the network and then disable this link which
in turn saves network energy. When a link is completely
idle, it is possible to shut down its ports on the adjacent
switches. If all ports on a switch are off, then the switch
can also be turned off completely. If there is no way able
to receive any of the redirecting flows, the process is aborted
with the understanding that it is not possible to vacate the
link without damaging the QoS indicators of the active
traffic.

Flow redirection is performed as a trigger whenever
active network monitoring classifies a link as low load
or overload. The search for alternative paths is done by
the Algorithm 1, with the restriction of $s_{max} = s_1$. This
restriction requires new paths to use only links in the normal
or low load state, preventing links in the $s_2$ (high load) state,
receiving more flows, and being classified as overloaded
in sequence, which would result in new redirections and,
in a subsequent “ping-pong” effect between the $s_2$ and
$s_3$ states.

5 Evaluation and results

In this section, we evaluate the DTM and the energy it saves.
More precisely, we evaluate DTM in three distinct scenar-
ios. First, we evaluate it in a static scenario (Section 5.1),
where we illustrate the very basic functionality of DTM.
Then, we evaluate DTM in a dynamic, yet synthet-
cenario (Section 5.2), where we show the performance of
DTM under a controlled experiment. Finally, we evaluate
DTM under a realistic scenario (Section 5.3), where we
mimic a typical campus network and evaluate the perform-
ance of our proposal.
We have evaluated DTM using the network emulator Mininet\(^1\) (ver. 2.3.0), which allows the creation of a virtual host network, switches, links, and controllers in a single computer. The OpenFlow driver used was POX (ver. 0.2.0) and the switches were run on instances of OpenVSwitch\(^2\) (ver. 2.5). The experiments were performed on a computer with 7th generation Intel Core i5@3.1GHz, 8GB RAM and Lubuntu 16.04 operating system.

5.1 Scenario A: static evaluation of DTM

First, we statically evaluate DTM to check its internal mechanism accuracy. This illustrative example guides readers to follow the basic DTM mechanisms. In a static built scenario, we are able to test (i) the DTM network topology discovery process; (ii) the active network monitoring process; (iii) the flow installation process. Figure 3 presents the static scenario we use to check DTM basic functionality. According to this figure, the topology we evaluate presents three hosts (h1, h2, h3) and for switches (S1, S2, S3).

First, the DTM controller performs a network topology discovery process. It detects all switches and links. Initially, only access switches will be kept turned on (S1, S3, and S4), and all reminder switches will be turned off to save energy (S2). Note that in Fig. 3a S2 is red-colored to represent it in off state.

After a few seconds, as shown in Fig. 3b, we intentionally start a flow between hosts H1, H2, and H3. The network controller then detects these flows and starts the flow installation process. In this case, the controller must activate the switches S2; otherwise, there will no path between communicating endpoints. In sum, the active nodes and links are colored in green in Fig. 3b.

Flows cease after 10 s. In this case, hosts and links present no inactivity, as we depict in Fig. 3c. In this experiment, we set up a 2-min threshold that triggers the DTM network controller. The controller, in turn, turns off all unnecessary switches, saving as much as energy as it could. By the end, Fig. 3d presents the same network state as the initial setup.

5.2 Scenario B: dynamic evaluation of DTM

Second, we dynamically evaluate DTM by changing the average network load in a controlled network environment. In this scenario, we evaluate the energy-savings, as well

---

\(^1\)http://mininet.org

\(^2\)https://www.openvswitch.org/
as the overhead DTM, imposes to the network due to flow redirections.

Figure 4 illustrates the network we evaluate. The switches are represented by circles and the hosts by squares. All links are Gigabit Ethernet. Initially, as shown in Fig. 4a, end hosts are idle and, as a consequence, switches are in the low load state we previously defined. The controller initiates the automatic network topology detection and then it builds the minimum spanning tree which connects all hosts. In this case, note, in Fig. 4a, that switch S3 is off, which saves network energy.

During our experiments, the host H3 acts as a server. Host H1 connects to H3, demanding on average 500 Mbps. The controller calculates the minimum spanning tree and, as shown in Fig. 4b, the suggested shortest path between H1 and H3 is S1-S2-S4. In this case, switches S1, S2, and S4 are low loaded and they accept the flow. The controller then finally installs the route in these switch tables.

In a third step, shown in Fig. 4c, the host H2 initiates communication with the server H3. H2 demands, in this case, 300 Mbps. In short, the controller identifies the path S2-S4 in the minimum spanning tree. These switches (i.e., S2 and S4) are in the normal load state and then, they accept the flow; the route is installed and hosts communicate with each other.

Any network link may become overloaded. For example, as shown in Fig. 4d, the link between S2 and S4 turns overloaded which imposes the controller to redirect traffic, according to the policies we previously defined.

In short, the controller performs an active flows redirection policy and it notes that the flow between H1 and H3 has an optional route through S1-S3-S4. The switch S3 can handle the overloaded traffic from the other switches. The controller then binds and installs this route, as shown in Fig. 4e. At this moment, there is no one overloaded switch; however, all switches are turned on, which imposes the highest energy consumption to the network.

Suppose flow between H1 and H3 ends. In this case, switch S3 (and its links) will turn to the low load state. The DTM controller may redirect all remaining flows from S3 table to other network switches (which are on normal or low load states). As a consequence, in case the controller successfully transfers all S3 flows to other switches, it turns off S3, as shown in Fig. 4f.

Figure 5 shows the percentage of energy-saving, and the network load, during the experiment we previously depicted. As expected, the energy economy is closely related to the network load. In fact, the higher the network load, the higher the number of switches we expect to turn on. Note that the flow redirection mechanism may impose a latency on energy savings. For example, during the period from 17 to 21 s in Fig. 5, the DTM controller turned on switches due to overloaded links, which impacted the...
network energy consumption. During our experiments, the DTM controller reaction time was about 0.2 ms.

Flow installation process and flow redirections may injure network traffic. For example, during a flow redirection, end-to-end communication may experience higher latencies. Figure 6 presents the end-to-end latency between network endpoints during our experiments. This figure clearly corroborates our previous comment. In special, during flow installation/redirections, as we noticed around the 15 to the 28 s of the experiment. More precisely, during a flow installation, a switch must consult the controller for a forwarding rule (reactive mode). Controller answer switch and, only after that, switch installs the new rule and flow can be performed. The second peak corresponds to a flow redirection due to an overloaded link. As the controller had proactively installed the rules, the overhead is smoothly lower. The third peak is similar to the second. That is, the controller observes that there is a low load on the routing switch links, triggering an event that leads to a new stream redirection section, intending to turn off the switch.

5.3 Scenario C: realistic evaluation of DTM

DTM can be used in arbitrary topologies as long as the nodes are SDN compatible. In this paper, we consider a realistic scenario represented by Fig. 7, which shows a campus network topology. This kind of topology has already been largely explored in similar works, as [7, 12, 14, 15]. Over the past decade, data centers and large network topologies, as a campus network, remain roughly stable. We have not noticed any disruptive technology and, as a consequence, this kind of topology remains updated, representing a wide variety of networks.

In this work, we consider a network topology with 95 links and 45 nodes (i.e., switches). More in deep, nodes include routing switches (nos. 1 through 4) and access switches (nos. 5 through 18). The others are host nodes (nos. 19 to 45), forming two distinct groups (the upper and the lower portions of the figure). Client/server application pairs for traffic generation and consumption are always positioned one in each host group.

We evaluate DTM considering two distinct scenarios. First, we consider the network with homogeneous links. In this case, all links have the same negotiated speed of 250 Mbps, which is close to the previous works [7].
This allows us to compare DTM to an existing approach to save energy in networks. Then, we evaluate DTM considering heterogeneous network links. In this case, the links interconnecting the routing switches have a speed of 1 Gbps, while the other links maintain the speed of 250 Mbps. Again, this scenario follows close to previous existing work.

5.3.1 Evaluation methodology

In this article, we consider the power consumption parameters (in watts) the same as [5] and [17]. More specifically, the base switch consumption is \( E_{b}(c) = 146 \), the 1-Gbps port consumption is \( E_{pg}(c) = 0.87 \), and the 100-Mbps port consumption is \( E_{pf}(c) = 0.18 \). New flow arrival rate follows a Poisson process with an interval expectation \( \lambda = 3 \) s. New flows have a fixed duration of 15 s and can assume one of the load levels described in Table 3. These values are equivalent to those used in [7] and take into account night traffic behavior, average daytime traffic, and annual peak traffic. According to the authors, annual peak traffic is five times higher than night traffic, and average daytime traffic is three times higher than night traffic. The traffic was generated by the D-ITG (Internet Traffic Generator) [18] tool.

The \( T_{0\rightarrow1}, T_{1\rightarrow2}, \) and \( T_{2\rightarrow3} \) thresholds used to classify link states were set at 20%, 60%, and 80% respectively. These values define a good compromise between the states we define in this work. We let future work further investigate these parameters. Moreover, we consider \( \Delta t = 1 \) s. In a real environment, we must fine-tune this value to not overload the network with control messages. We consider building \( P_{ij} \) sets with \( k = 16 \) minimum paths connecting host \( i \) to \( j \), \( \forall i, j \in N_h \). All minimum paths will have five hops between source and destination in this topology.

Our evaluations evidence the percentage of power savings by comparing DTM to a network where all devices are turned on (i.e., there is no energy-saving mechanism). We repeat each experiment 50 times, and each one remains for 120 s. Unless we tell otherwise, we present mean values and confidence intervals, for a 95% confidence level.

5.3.2 Evaluation

We evaluated DTM in the homogeneous and heterogeneous scenarios, considering the three traffic demand levels.

| Level | Traffic demand | Description          | Flow throughput |
|-------|----------------|----------------------|-----------------|
| \( N_1 \) | High            | Annual peak          | [150, 250] Mbps |
| \( N_2 \) | Average         | Daytime average      | [50, 150] Mbps  |
| \( N_3 \) | Low             | Nightly average      | [0, 50] Mbps    |

Figure 8a and b show the energy-saving cumulative distribution functions in each evaluated configuration. Intuitively, the higher demand, the lower savings achieved, as more links are in use and fewer opportunities to shut down idle network elements.

Under low traffic demand, both scenarios show similar behavior. In these cases, the minimum links required to maintain network connectivity are active. As a result, DTM achieves a significant average energy savings of...
over 46%. On average demand, energy savings differ. In a heterogeneous scenario, the average savings were \( \approx 40\% \). In a homogeneous scenario, this value is near 36\%, \( \approx 4\% \) lower than heterogeneous scenario. As the heterogeneous scenario has some higher capacity links, it can absorb this average demand without linking new links. In the homogeneous scenario, however, it is necessary to connect more links and eventually more switches, which increases energy consumption. This is even more evident with a high work demand, where more links and switches are required to support full traffic, minimizing the opportunity to save energy. Energy savings fall considerably for both cases but are most evident in the homogeneous scenario. In this situation, the homogeneous scenario consumed \( \approx 7\% \) more energy than the heterogeneous one, and the latter achieved no more than 25\% of the total average energy savings.

Figure 9 presents the network map and illustrates which switches are turned on/off. In this figure, we present in green the switches typically turned on during our simulations. The Fig. 9(a) to (c) present situations of low, medium and high network demands, respectively, for the homogeneous scenario. The Fig. 9(d) to (f) refer to the heterogeneous scenario.

As discussed earlier, in a high demand situation, there are more switches turned on for both scenarios. On the other hand, in a low demand scenario, the flow between the network links is low, which eliminates the redirection of active flows and results in the highest energy savings.
The heterogeneous scenario has greater energy savings compared to the homogeneous one. This is quite evident when comparing the Fig 9(b) and (e), or the Fig. 9(c) and (f). Again, heterogeneous scenarios have some links with larger capacity that are capable of supporting large volumes of flows without having to enable alternate paths for possible redirections.

Table 4 shows the flow observed in the network during simulations. Homogeneous and heterogeneous scenarios present equivalent behavior for low and medium demands. For high demand, the flow observed in the heterogeneous scenario is higher. This is because the homogeneous scenario core links reach their load limit and become a bottleneck for the network. In contrast, the heterogeneous scenario core links support the generated flows, providing energy savings and higher flow between switches.

Finally, we compare DTM with a state of the art solution, specifically [7]. Figure 8c shows that both algorithms have the same energy-saving gain in low traffic demand scenarios. In fact, the minimum number of links for network connectivity is capable of ensuring all low demands experience flows. Thus, there are scenarios, some solutions are at their maximum level of the economy. On average traffic demand, note a single difference between the algorithms for both homogeneous and heterogeneous links. DTM was, on average, 7% more economical than the homogeneous scenario core links support the generated flows, providing energy savings and higher flow between switches.

Finally, we compare DTM with a state of the art solution, specifically [7]. Figure 8c shows that both algorithms have the same energy-saving gain in low traffic demand scenarios. In fact, the minimum number of links for network connectivity is capable of ensuring all low demands experience flows. Thus, there are scenarios, some solutions are at their maximum level of the economy. On average traffic demand, note a single difference between the algorithms for both homogeneous and heterogeneous links. DTM was, on average, 7% more economical than the homogeneous scenario core links support the generated flows, providing energy savings and higher flow between switches.

When the traffic demand is high, DTM was \( \approx 4\% \) and \( 5.79\% \) better than [7] method, considering the homogeneous and heterogeneous scenarios, respectively. In short, even when energy-saving opportunities are rare, DTM has the advantage. The gains are significant, especially considering the ultimate impact given computer network-scale usage today.

### 6 Conclusions and future work

In this paper, we present DTM, an Energy-Aware Dynamic Traffic Management. DTM employs SDN to improve traffic routing between switches. The mechanism has been evaluated through the emulation of networks using Mininet, considering a realistic topology. Our evaluation results show that DTM provided an average energy saving of 46% in a scenario with low network demand, similar to a nighttime pattern. In this low-demand scenario, as expected, we achieved the highest energy savings. In scenarios with average to high traffic demands, the mean energy savings are 36.72 and 17.86%, respectively.

Compared to a well-known existing mechanism, our approach is up to 7% for medium-demand scenarios and approximately 4% better for high-demand scenarios. Future work includes the investigation of DTM scalability in other topologies, ranging the topology density, for example.

### References

1. Assefa BG, Özkasap Ö (2019) A survey of energy efficiency in sdn software-based methods and optimization models. J Netw Comput Appl 137:127–143
2. Tuyuz MF, Ankarali ZK, Gözüpek D (2017) A survey on energy efficiency in software defined networks. Comput Netw 113:188–204
3. Costa Leonardo C., Vieira Alex B., Silva Erikde Britto E, Macedo Daniel F., Gomes Geraldo, Correia Luiz H. A., Vieira Luiz F. M. (2017) Performance evaluation of OpenFlow data planes. In: IFIP/IEEE IM, pp 470–475
4. Addis B, Capone A, Carello G, Gianoli LG (2016) B Sansó. Energy management in communication networks: a journey through modeling and optimization glasses. Comput Commun 91–92:76–94
5. Sasaki S, Ogura K, Bista BB, Takata T (2015) A proposal of QoS-aware power saving scheme for SDN-based networks. In: NBISS, pp 405–410
6. Fisher W, Suchara M, Rexford J (2010) Greening backbone networks: reducing energy consumption by shutting off cables in bundled links. In: ACM SIGCOMM Workshops, pp 29–34
7. Markiewicz A, Tran PN, Timm-Giel A (2014) Energy consumption optimization for software defined networks considering dynamic traffic. In: IEEE CloudNet, pp 155–160
8. Lin G, Soh S, Chia K-W, Lazarescu M (2013) Efficient heuristics for energy-aware routing in networks bundled links. Comput Netw 57(8):1774–1788
9. Heller B, Seetharaman S, Mahadevan P, Yiakoumis Y, Sharma P, Banerjee S, Elastictree NM (2010) Saving energy in data center networks. In: USENIX NSDI, pp 1–17
10. Fernández-Fernández A, Cervelló-Pastor C, Ochoa-Aday L (2016) Improved energy-aware routing algorithm in software-defined networks. In: IEEE LCN, pp 196–199
11. Habibullah KM, Rondeau E, Georges J-P (2017) Reducing energy consumption of network infrastructure using spectral approach. In: Dastbaz M, Arabnia H, Akhgar B (eds) Technology for Smart Futures. Springer, Cham, pp 235–250
12. Oliveira AT, Martins BJCA, Moreno MF, Gomes ATA, Ziviani A, Vieira AB (2019) SDN-based architecture for providing quality of service to high-performance distributed applications. Int J Netw Manag e2078. https://doi.org/10.1002/nem.2078
13. Son J, Dastjerdi AV, Calheiros RN, Buyya R (2017) Sl-a-aware and energy-efficient dynamic overbooking in sdn-based cloud data centers. IEEE Trans Sustain Comput 2(2):76–89
14. Xu G, Dai B, Huang B, Yang J, Wen S (2017) Bandwidth-aware energy efficient flow scheduling with sdn in data center networks. Future Gener Comput Syst 68:163–174
15. Jia X, Jiang Y, Guo Z, Shen G, Wang L (2018) Intelligent path control for energy-saving in hybrid sdn networks. Comput Netw 131:65–76
16. Bianzino AP, Chaudet C, Rossi D, Rougier J-L (2010) A survey of green networking research. IEEE Commun Surv Tutor 14(1):3–20
17. Mahadevan P, Sharma P, Banerjee S, Ranganathan P (2009) Energy aware network operations. In: IEEE INFOCOM, pp 25–30
18. Botta A, Dainotti A, Pescapè A (2012) A tool for the generation of realistic network workload for emerging networking scenarios. Comput Netw 56(15):3531–3547

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.