University of Rennes 1
Computer Science
Cloud Computing and Services

Betsegaw Lemma Amersho

Simulating Energy-Aware Networks in Large-Scale Distributed Systems

Master's Thesis
Rennes, June 26, 2017

Supervisors: Professor Martin Quinson, ENS Rennes
Dr. Anne-Cécile Orgerie, CNRS
Advisor: Assoc. Prof. Keijo Heljanko, Aalto University

University of Rennes 1
Computer Science
Cloud Computing and Services

ABSTRACT OF
MASTER'S THESIS

Author:	Betsegaw Lemma Amersho		
Title:			
Simulating Energy-Aware Networks in Large-Scale Distributed Systems			
Date:	June 26, 2017	**Pages:**	58
Major:	Cloud computing and services	**:**	
Supervisors:	Professor Martin Quinson Dr. Anne-Cécile Orgerie		
Advisor:	Assoc. Prof. Keijo Heljanko		

Due to the emergence of new technologies such as IoT, the number of devices connected to the data-center infrastructures is rapidly increasing. In response to this growing trend, the data-centers are also expanding. This expansion led to a major energy consumption issue. To tackle this issue researchers are designing and running experiments on real network, test-bed environments and simulation software. The first two experiment environments are limited interms of availabilty, flexibility and cost for conducting large-scale energy consumption estimation experiments. Simulation is the viable alternative in the context of large-scale networks. Currently proposed simulation software that are capable of estimating energy consumption of a network are in the category of packet-level simulators. The main problem of this kind of simulators is that they fail to scale well in the area of large-scale networks. In this paper we have proposed a reasonably accurate and scalable flow-level simulator alternative in the context of SimGrid, a large-scale distributed network simulator. Our flow-level simulator registered relative error less than 1% in accuracy compared to a packet-level simulator and it is also at least 243 times more faster.

Keywords:	Flow-level, Energy, SimGrid, ECOFEN
Language:	English

Acknowledgements

I am grateful to Martin Quinson, Anne-Cécile Orgerie, and Keijo Heljanko for the useful comments, remarks and engagement through the learning process of this master thesis, and to Guillaume Pierre for helping me find my internship and for following up my work. I thank the Myriads team, IRISA for financial support during my internship.

Last but not the least, I would like to thank my family and friends who have supported me throughout my study.

Rennes, June 26, 2017

Betsegaw Lemma Amersho

Contents

Chapter 1

Introduction

According to the report released by Cisco in June 2017, "The Zettabyte Era" [10], the number of networked devices is expected to increase from 17.1 billion in 2016 to 27.1 billion in 2021. In another report entitled "Cisco Global Cloud Index, 2015 to 2020" [9], Cisco estimates that global cloud IP traffic will grow more than three times by 2020 and, among all the workloads processed in data-centers, by the year 2020, 92% of them will be handled in cloud data-centers. The remaining 8% will be processed in traditional data-centers. In response to these growing trends, the data-centers are continuously expanding. This expansion raises a primary concern on the amount of energy required to support the added data-center components and the growing service demands.

The energy consumption issue is further aggravated due to the fact that current servers and network devices are energy inefficient. Of the total power consumed by a given computing or communication device, the idle power consumption takes the greater proportion [27]. IT equipment is considered energy efficient when it consumes power proportionally to the amount of computing or data transfer task it performs. Currently there are different techniques implemented at a device level to tackle the energy inefficiency problem. For computing device, for instance, the operating frequency of the CPU can be lowered when the amount of computing goes below some threshold value [35]. Similarly, for a communicating device, the data transferring rate can be lowered (a.k.a, adaptive link rate mode) depending on the traffic [16] or the device can be set to sleep (a.k.a, low power mode) when there is no traffic [29]. However, these techniques are not fully utilized, as they induce performance penalty when switch from one mode to another.

The energy consumption issue is primarily driven by economical factor – to save energy consumption bills – and environmental factors – to reduce CO_2 emission. There are also other secondary factors, such as reducing the heat generated by a given IT device. The more energy inefficient a device is the more heat it generates. This internally affects the life time and proper functioning of the device.

Currently researchers are tackling the energy consumption issue at different levels. At device level, for instance, it consists in finding new energy saving techniques or optimizing the existing ones. At infrastructure level, it can range from finding energy aware routing algorithms for network devices to energy efficient load balancing and workload assignment of servers.

There are three approaches that are commonly in use for conducting energy related research experiments at the infrastructure level. The first approach is to experiment on a real network. Though in this approach one might get the most real picture of the situation at a given moment, it will be very difficult to repeat the experiment on other real networks due to the transient nature of the experimental parameters such as workload and network traffic. Furthermore, the real network might not be available for experimentation. The second approach is to use an experimental test-bed. This approach gives full control over the experiment parameters. However, when the platform under investigation becomes very large, it will be infeasible to set-up a test-bed for it due to the hardware cost involved. In addition, experimenting on a new hypothesis might require setting up a new test-bed with a new hardware and different configuration. This also becomes costly and time consuming. The third approach is to use simulation software for experimentation. This approach gives the ultimate control, flexibility and scalability compared to the other two. The main challenge though is accurately modeling the real characteristics of the components involved in the problem at hand.

In the context of computer networking, we can classify simulators into two kinds: packet-level and flow-level simulators. Packet-level simulators strive to capture fine-grain details of a given network phenomenon. Flow-level simulators, on the other hand, use analytical equations that approximate the behavior of the phenomenon being modeled using few parameters. Compared to flow-level simulators, packet-level simulators are considered more accurate due to the detailed information they try to capture. However, they fail to scale well due to the time and storage they need to process and store the captured information. A typical example of packet-level simulator is NS-3, and of flow-level simulator is SimGrid, a large-scale distributed network simulator.

Despite the advantages network simulators can offer for studying the energy consumption problem that exists in large-scale networks, search of the literature revealed only few packet-level simulators proposed to address the issue. As we have mentioned earlier, packet-level simulators can not scale well in the area of large-scale networks. Currently, to our knowledge, there is no flow-level simulator proposed that can simulate the energy consumption of computing and communication components of large-scale networks.

Therefore, the purpose of this study is to investigate the level of accuracy of flow-level energy consumption simulators for estimating energy consumption of large-scale networks. To fulfill this purpose we use the SimGrid simulator. SimGrid already has an energy consumption model for computing components, our work is limited to adding flow-level energy model for communicating devices such as switches and routers. Furthermore, we are only concerned with wired network components and concepts.

Our contribution in this work is twofold: (1) outlining a research method that help us achieve our objective successfully and (2) using this method, designing flow-level energy consumption model, implementing it in SimGrid and validating it against NS-3 simulator.

The rest of this thesis is organized as follow. In Chapter 2, we first describe relevant concepts that are related to our study and then we review other proposed simulators. In Chapter 3, we explain about our experimental environment. Then in Chapter 4, we compare the advantages and disadvantages of commonly used methods that are used to study the energy consumption problem and then we outline the method that we followed. In Chapter 5 we present the implementation of our flow-level model and then in Chapter 6 we discuss about the validation experiments that we have conducted to evaluate the implementation. Following the validation, in Chapter 7, we discuss the whole study, point out the limitations of the study and based on the limitations we also give recommendations for future work. Finally, in Chapter 8 we give concluding remarks.

Chapter 2

Background

In this chapter we first start by describing the current trend in global electricity consumption in the IT field and then we describe the energy consuming components involved in large scale distributed networks. In related to the components, we describe the concept of energy proportionality, which explains why servers and network components are considered energy inefficient. We then mention the approach used to study this energy inefficiency problem. We gave particular emphasis on packet-level and flow-level simulators. Next, we describe SimGrid as a large-scale distributed network simulator, its role in estimating energy consumption of large-scale distributed networks, and what we are planning to do for it and using it. Finally, we review existing simulators that are proposed for estimating large-scale network energy consumptions.

2.1 Energy consumption of ICT equipments

ICT equipment consume a significant amount of electricity. A survey conducted by Heddeghem et al. [18] shows the electricity consumption and growth trends of three classes of ICT equipment: personal computers, communication networks, and data centers. Personal computers include devices such as desktop, laptop and external monitors. Communication networks includes residential network access devices (such as WiFi routers and modems), network equipment used in offices (such as routers and switches) and telecom-operator network equipment (such as base stations, routers and optical amplification systems). Data-centers house storage and computing servers, communication network equipment, and power provisioning and cooling facilities. In this classification there are overlaps, for instance, telecom-operator can have office network equipment and data-centers. After carefully avoiding possible redundant measurements, the researchers estimated absolute electricity consumption and annual consumption growth rate of each category of equipment for the period 2007 and 2012. The results of the study shows that the global electricity consumption share of personal computers is 1.6%, communication networks is 1.7%, and data centers is 1.4%. The estimated annual growth rate of each category is 5% for personal computers, 10% for communication networks, and 4% for data-centers. These growth rates are higher than that of the total global electricity consumption, which is 3%. This trend signifies the need for energy saving research in all the three categories.

2.2 Large-scale network energy consumption

In Section 2.1 we described data-center's global share in electricity consumption. In this section we describe the components involved within the data center itself.

Electricity consumption units within a typical data-center can be classified into two broad groups [13]: The first group is IT equipment, (which includes computing servers, storage servers and networking components) and the other group is infrastructure facilities (which includes power provisioning, cooling and lighting components).

Figure 2.1 from [13] shows the electricity consumption proportion of the data-center components. This value differs significantly from one data-center to another [2], for instance, due to architectural difference [17] or energy efficiency of the components. The infrastructure facility components take the large proportion (e.g., 65%) of the consumption.

Though the infrastructure facility consumes relatively larger amount of electricity, the focus of this study is on the IT equipment components, particularly on the network equipment.

If we further zoom in on the IT equipment part, we can find computing servers, storage servers and network devices. A data-center servers consist of one or more CPU cores, memory and I/O devices. The energy consumption relationship among these components is shown in Figure 2.2. Combined, memory and CPU units consume the larger amount of energy relative to other components. The fact that CPU is the dominant electricity consuming unit is exploited by Fan et al. in [14] to model the dynamic power usage of thousands of servers by using only CPU utilization as a parameter. The result of their study was very accurate, with error as low as 1%. The energy consumption contribution of storage servers in a typical data center is shown in Figure 2.1 together with computing servers.

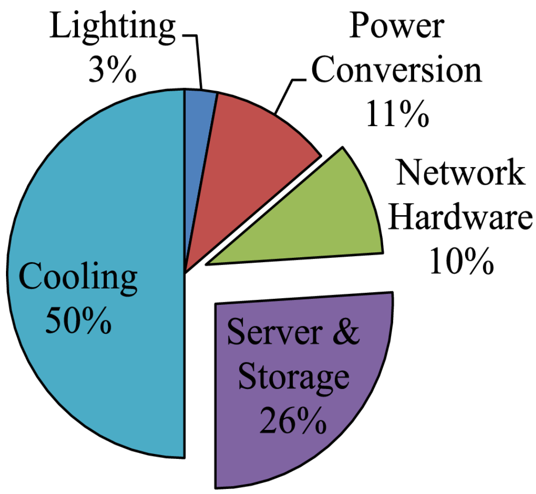


Figure 2.1: Energy consumption percentage of data-center components from [13]

Network devices are the other part in the IT equipment component of a data center which contribute to energy consumption as shown in Figure 2.1. Shehabi et al. in [32], from Berkeley National Laboratory, produced a report which show the annual energy consumption of network devices deployed in data centers found in the United State. The historical and the forecast energy consumption is shown in Figure 2.3 for the period 2006 up to 2020. In the figure, the absolute electricity consumption of network equipment is shown grouped by port speed of 100 Mbps, 1000 Mbps, 10 Gbps, and 40 Gbps.

In large-scale distributed networks, network devices are deployed with in and outside the data center. Our study is not limited only to network devices residing in a particular data center, it also includes network devices residing outside a data center.

2.3 Energy proportionality

The primary reason the study of energy consumption management of network equipment becomes so important is that, in general, ICT equipment do not consume energy proportional to their workload. An ideal ICT equipment is the one which consume zero electricity when it is idle, and it consumes electricity proportional to its workload when it is active. However, the reality is, even power efficient servers consume about 50% of their peak power [3], even when they are doing nothing. This percentage can even reach 85% for network switches [15]. Figure 2.4 from [24] shows the ideal and the typical energy consumption characteristics of a network equipment.

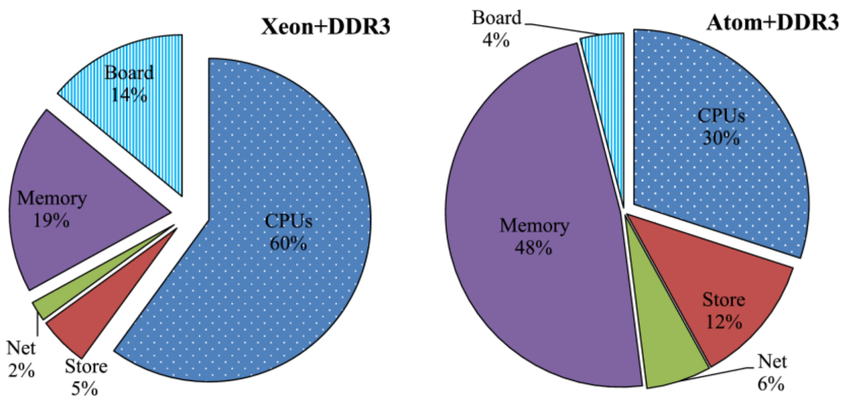


Figure 2.2: Energy consumption percentage of Xeon based (on the left) and Atom based (on the right) servers [13]

Three approaches are in common use to deal with this situation [5]. The first one is re-engineering network devices so as to make them more energy proportional, device vendors are the prime role player in this aspect. The second approach is related to the operating rate of a network equipment port. A typical switch can operate on different transmission rate (100 Mbps, 1 Gbps or 10 Gbps). An active port transmitting at 10 Gbps can consume more energy than if it transmit at 100 Mbps. Rate adaptation is the approach devised to take advantage of this situation. Instead of transmitting at the maximum rate all time, the network port can be made to adapt to the actual traffic load. This energy saving approach is known as Adaptive Link Rate (ALR) [25]. The third approach, which is referred to as Low Power Idle (LPI), allows a network device to send data as fast as possible and then enter low power mode between transfers [3]. The low power mode can further be extended by a technique called packet coalescing, which allows more energy saving [5].

2.4 Packet-level and flow-level simulators

One way of conducting an experiment is to use real production environment or to use a test-bed environment, both are referred to as *in vivo* in [8]. In the former case, handling transient and varying conditions would make the data collection and prediction very difficult and often times, a production environment is also not available for experimentation. In the later case, it requires setting-up a separate testing environment designed solely for the purpose of conducting the desired experiment. This approach apart from

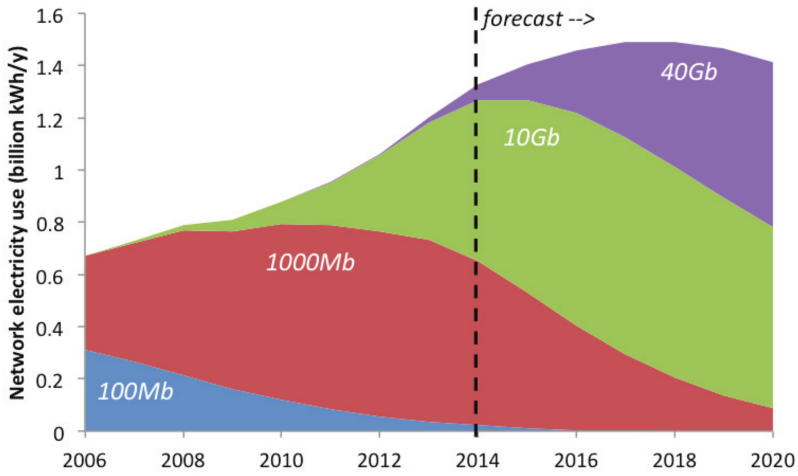


Figure 2.3: Total data center network equipment energy consumption in the United States [32].

being expensive, it requires significant amount of time for experiment setup: and it can also be non-repeatable when experimenting with different scenario that demands a significantly modified or completely new configuration.

The other alternative for experimenting is simulation, also referred to as *in silico* in [8]. Simulation, unlike real environment, allows great flexibility in terms of experiment configuration, control and repetition. In addition, it can also be less time consuming and less expensive.That is why virtually in all computer network related researches simulations are widely used.

Simulators use models to specify the relationship between the variables involved in a particular network phenomenon. Generally, the models are classified as packet-level and flow-level models based on the detail of information the models are trying to capture. We can also refer to simulators as packet-level simulator and flow-level simulators based on the model they use.

Packet-level simulators strives to model a given network phenomenon at the granularity level of individual packets[21]. Due to the detail of information this simulators capture, in general, they are accepted by the research community to be more accurate compared to flow-level ones [8]. One of the most popular packet-level simulator is NS-3, which is categorized under discrete-event simulator with events corresponding to sending and receiving of packets [22]. Though packet-level simulators are accepted to be more accurate, they fail to scale well in the field of large-scale distributed networks due to the computation and storage cost involved in processing and storing each packet.

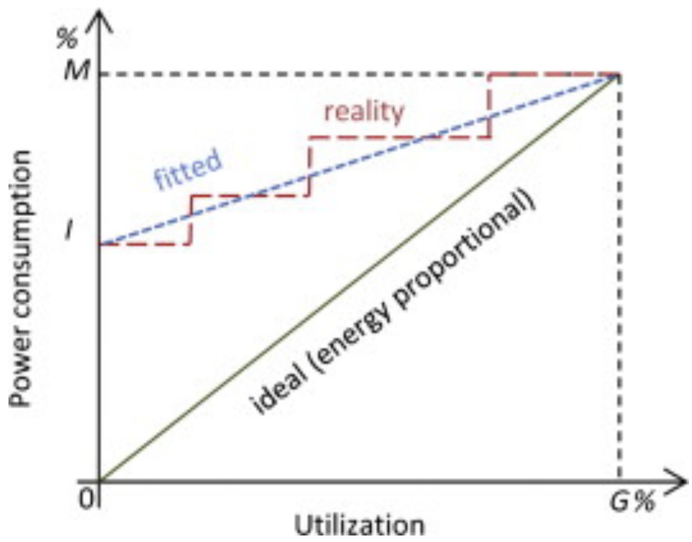


Figure 2.4: Ideal and measured energy proportionality of a network equipment [24]

In the area of large-scale networks, flow-level simulators are the preferred simulation alternative. Rather than modeling a given network phenomenon at an individual packet level, flow-level models treat a set of packets as a single unit [8, 21]. The most commonly used definition for a *flow* in the context of computer networking is coined by Claffy et al. in [11]:

"...a *flow* ...a unidirectional traffic stream with a unique [source-IP-address, source-port, destination-IP-address, destination-port, IP-protocol] tuple ..."

In addition to the five tuple mentioned in the definition, a flow also has a limited time duration. Claffy et al. [11] used a time limit of 64 seconds as a flow duration in their study. Researchers such as Carneiro et al. [7], adopted this definition to develop flow monitoring module for NS-3, a module that can generate information such as amount of packets or bytes transferred, packets dropped or transmission start and end time for each flow. Barakat et al. in [1] also used this definition to model traffic at the flow-level for the Internet backbone link. By abstracting away fine details, flow-level models provides easy way to instantiate experiments and they also scale very well for conducting large-scale network simulations [1, 8].

The flow definition given above is not the only one. Any analytical model which capture the characteristics of a given network phenomenon can also be considered a flow-level model. In SimGrid, for instance, TCP flow is characterized primarily by bandwidth and end-to-end latency [8].

2.5 Simulating and modeling energy consumption of large-scale networks

In this study we simulate energy-aware large scale distributed networks using SimGrid (more description about SimGrid follows in the next section). When we say large-scale distributed network, we are referring to a set of networks residing inside in the distributed data centers and also the networks that are used to connect them.

The energy consumption E of an equipment depends on the operating power P at time t. The total energy consumption for a time period T is given by Equation 2.1 [26].

$$E(T) = \int_0^T P(t)dt \tag{2.1}$$

Due to the energy proportionality characteristic described in Section 2.3, the common approach used to compute the energy consumption is to divide the power component into two parts: static/idle power (P_{static}) and dynamic power ($P_{dynamic}$) as shown in equation 2.2. Then the total energy is obtained by multiplying the total power, P_{total} by the time duration [13, 20, 24, 26].

$$P_{total} = P_{static} + P_{dynamic} \tag{2.2}$$

For a typical network equipment such as a switch, the static part constitutes the power consumption of the chassis and the line-cards (when all the ports on the line-cards are switched off). The dynamic part, on the other hand, constitutes the power consumption of the switch ports running at a given rate multiplied by the utilization factor [24]. Equation 2.3 shows how to compute the total power for a switch, where P_{switch}, is the total power consumption of a switch, $P_{chassis}$ and $P_{linecard}$ is the idle power consumption of the chassis and the line card, respectively. P_{rate}, is the power consumption of a given port at a given rate and $numports_{rate}$ is the number of ports running at a given rate. The rate can take values such as 10 Mbps, 100 Mbps, 1 Gbps or 10 Gbps.

$$\begin{aligned} P_{switch} = P_{chassis} + (numlinecards \times P_{linecard}) + \\ \sum_{rate=min}^{max} (numports_{rate} \times P_{rate} \times utilizationFactor) \end{aligned} \tag{2.3}$$

2.6 SimGrid

SimGrid is one of the simulator available for simulating large-scale distributed networks such as grid, cloud, volunteer and HPC [38]. It employs flow-level models in its core for simulating different network resources and phenomenon. In subsequent paragraphs we give overview of its architecture, the pros and cons of the employed TCP flow-level model and its current status in relation to energy consumption models.

Figure 2.5 shows the structure of SimGrid and how its core works. The top three components are the APIs that users can use to develop their simulation. Both MSG and SMPI are used to specify simulated applications as concurrent processes. The difference is that using MSG, users can simulate any arbitrary application, whereas, using SMPI users can simulate existing MPI applications, the MPI processes are created automatically from C or Fortran MPI programs. SIMDAG, on the other hand, does not use concurrent processes. It allows users to describe their application as communicating task graph. The next layer, SIMIX, implements the mechanisms that are required to simulate the concurrent process of MSG and SMPI applications. It also provides process control and synchronization functionalities. The bottom layer, SURF, is the simulation core, it simulates the execution of activities on computing or communication resources [8].

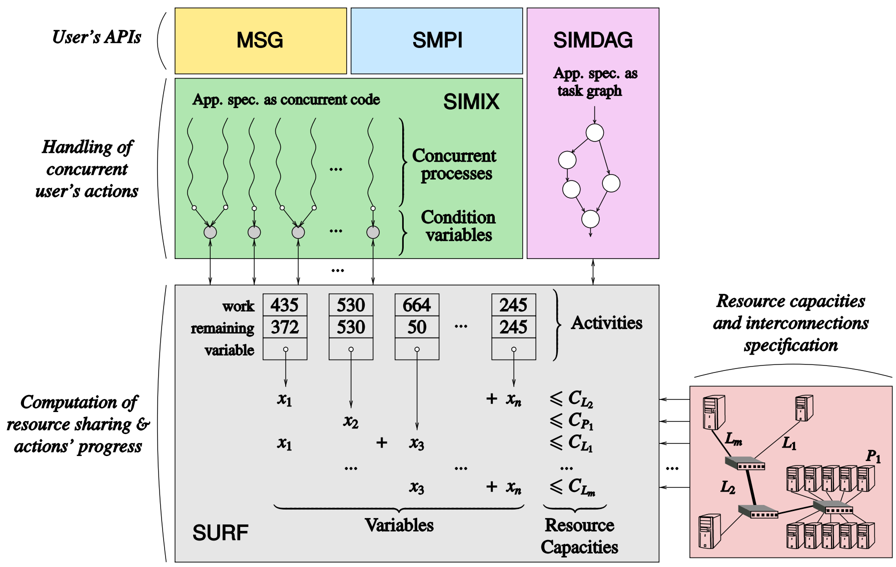


Figure 2.5: Architecture of SimGrid [8]

In SimGrid for each simulated activity, such as computation or data transfer, there is a corresponding condition variable, in Figure 2.5 it is shown in SIMIX box. This condition variable synchronizes the concurrent processes of the simulated applications. The computing (P_x) and the communication (L_x) resources are shown on the bottom-right side of the figure. Computing resources are defined in terms of computing power, whereas, communication resources are defined in terms of bandwidth and latency. As shown in the SURF box, multiple activities can share the same resource (e.g., (x_1, x_n), (x_1, x_3) or (x_3, x_n)) or one activity can use multiple resources (e.g., x_1 or x_3 or x_n). Activities that share the same resource are limited by the capacity of that resource. Each activity is defined by the total and remaining work to be executed. When the work associated with the activity completes, the corresponding upper layer components receive a notification signal [8].

As we have already pointed out in Section 2.4, the primary advantage of flow-level simulation is its scalability in terms of speed and memory usage [28]. SimGrid uses flow-level analytical model for simulating TCP network phenomenon [8]. To show the scalability of the flow-level model, the SimGrid team compared it with other widely used simulators such as GridSim and OverSim. After simulating 500,000 tasks both on GridSim and SimGrid, the results demonstrate that SimGrid is 257 times faster and 26 times more memory efficient. Similarly, the comparison result with OverSim shows that SimGrid is 15 times faster and it can also simulate scenarios 10 times larger. Concerning the accuracy, though the simulator gives very good accuracy in most of the case studies explained in [8], there are situations where it fails to give accurate results. As an example, the comparison study of SimGrid with packet-level simulator GTNetS shows that for data size less than 100 KiB there is a significant difference in prediction.

Currently, SimGrid has energy consumption model for CPU. Using this CPU model researchers can simulate energy consumption of single or multi core CPUs running at different operating frequencies. Concerning network equipment however, SimGrid has no energy consumption model. Therefore, the focus of this study is to propose and implement network energy consumption model for SimGrid. The implementation of this model, together with the existing CPU energy model, allows us to estimate the energy consumption of large-scale networks that reside within or outside a data-center.

2.7 Related Simulators

In this section we review existing simulators that are proposed for estimating energy consumption of large-scale networks.

2.7.1 ECOFEN

Orgerie et al. [26] proposed ECOFEN, an Energy Consumption mOdel For End-to-end Networks. It is a packet-level simulator designed for estimating energy consumption of large-scale networks. Initially the simulator was developed as NS-2 module but currently it is also available as NS-3 module [12]. ECOFEN provides three models for simulating energy consumption at different levels of granularity: *basic*, *linear* and *complete*.

The basic model allows to simulate energy consumption of a network interface card (NIC) at coarse level of granularity. It only accepts energy consumption value for ON and OFF state of the NIC. The linear model, on the other hand, accepts energy consumption value for the idle state of the NIC and for each bytes processed. This model allows to compute power consumption of a given network traffic. The complete model like the linear model also considers traffic in its power consumption computation. The difference is that it offers added flexibility in terms of energy parameters. Different energy consumption values can be assigned for bytes received or send and also packets received or send. All the three models produce the estimated average energy consumption at milliwatt precision level in the chosen time interval and the time interval can be set as small as a millisecond.

ECOFEN module has two main limitations, mainly due to the limitation of the underlying NS-3 simulator. The first limitation comes from the lack of CPU abstraction in NS-3. As we have discussed in Section 2.2, server energy consumption is the second dominant part in typical data center and CPU is the main contributer among server parts such as memory and storage. Furthermore, since the energy consumption of CPU is linearly dependent on its operating frequency and its workload, the energy consumption increases more as the workload increases. As a consequence of absence of CPU from NS-3, the energy estimation we get from ECOFEN is partial. It only able to simulate energy consumption of network components such as NICs, switches and routers. The second limitation is concerned with the scalability issue. Being a packet-level simulator, the performance of ECOFEN is affected significantly as the number of processed packets grows large or as the network size become large. Cornea et al. [12] noticed this scalability problem during their study of the energy consumption of data transfers in clouds using ECOFEN module. It took them 5 hours to capture 1 minute of simulated network activity for a large-scale network.

2.7.2 GreenCloud

Kliazovich et al. [20] proposed GreenCloud, a simulator that can estimate energy consumption of cloud computing data centers. GreenCloud is developed as extension to NS-2 packet-level network simulator. This simulator contains power consumption models both for the computing and communicating components of a typical data center. The power consumption model used for the computing component is shown in Equation 2.4. This equation contains power consumed by the fixed parts (such as bus, memory and disk) which consume power independent of the operating frequency f of the computing component CPU and the power consumed by the CPU (P_f) operating at a given frequency *f*. This model allows for lowering the operating frequency of the CPU when workload becomes below some predefined threshold in order to decrease the power consumption.

$$P_{computing} = P_{fixed} + P_f \times f^3 \tag{2.4}$$

The power consumption model used in GreenCloud for the communicating components is the one shown in Equation 2.3. The equation shows the static power consuming parts (such as the chassis($P_{chassis}$)and the active line cards($P_{linecard}$) and the dynamic part (P_{rate}) is the energy consumed by the port running at a particular line rate for a given traffic load.

This simulator is limited in three aspects: (1) in the number of allowed CPU cores, (2) in versatility, and (3) in scalability. The first limitation is that only one CPU core is allowed per simulated node. This hinders the study of energy consumption of multi-core computing nodes. The second one is that we can not use this simulator outside the cloud computing domain such as grid, volunteer, peer-to-peer or HPC, at least that is not the authors original intention when they develop this simulator. The available features of the simulators are tuned towards cloud computing applications only. This limits its versatility. The third limitation deals with the scalability issue. The fine grain details provided by GreenCloud and the packet-level processing approach of the underlying NS-2 simulator is advantageous for getting accurate result when simulating relatively small networks. However, for large-scale distributed networks, it is not scalable. In related to this, the authors have mentioned that the simulation speed gets slower and slower as the number of simulated nodes increases beyond few thousands and also as the number of processed packets increase. The GreenCloud's underlying simulator NS-2, is known for its scalability problem. Currently NS-3 is available as a better performing alternative [40] however, we could not find any upgrade for GreenCloud.

Both of these simulators, since they compute energy-consumption at a packet-level, suffer from scalability issue when the size of simulated nodes and traffic size increases. Therefore, the aim of our study is to investigate if flow-level models are reasonably accurate and more scalable for estimating energy consumption of large-scale distributed networks.

Chapter 3

Environment

In this study we employed SimGrid, ECOFEN, FlowMonitor modules of NS-3 simulator and other tools. This chapter explains the main features of these tools from the perspective of our simulation experiment needs.

3.1 SimGrid

In Section 2.6 of Chapter 2 we discussed the software architecture of SimGrid at a higher level. We will give low-level details of the implemented flow-level model and related concepts in later chapter. In this section, our plan is to discuss features of SimGrid that are related to setting up and running energy consumption experiments.

In Figure 2.5 of Chapter 2 we have presented three user APIs that SimGrid users can use to develop their simulation experiments. Currently there is another API named S4U that is under development. This API is similar in usage to MSG API. One main difference, from users perspective, is that MSG is in C while S4U is in C++ language. S4U is the API that we have used in this study.

Designing and running simulation experiments in SimGrid using MSG or S4U APIs involve creating three files: a C/C++ simulation Script, an XML file for specifying the simulated platform topology and another XML file for specifying the deployment options, such as, identifying the host that send or receive data, the size of data, and the number of processes sending the data.

In a typical simulation experiment of energy estimation as a function of data transfer, we can have three sections in the simulation script. In the first section we write a function which specify what the sender do to send the data, in the second section we write what the receiver do to receive the simulated data or what action to take when the simulated data arrives. In the third section we tell to SimGrid's simulation engine about the two functions and we also pass to the engine the platform and the deployment file names.

In SimGrid, simulated network resources such as, NICs, switches and routers are represented with an abstraction called Link. In SimGrid platform file we represent a Link as follows:

```
<link id="SWITCH1" bandwidth="100MBps" latency="10ms">
      <prop id="watt_range" value="305:550"/>
</link>
```

From the link XML tag we can see the basic characteristics of a simulated switch, such as, bandwidth and latency. We also see the idle and busy power consumption range of the switch. Other additional information can also be added such as how the link should be shared when multiple traffic cross the link and tracing information to control the bandwidth and latency property while the simulation is progressing.

In a similar manner we can specify deployment information such as the role of hosts, the number of processes and the size of transfered data as follows.

```
<process host="H1" function="sender">
<argument value="10" />
</process>
<process host="H2" function="receiver"/>
```

This separation of concern among the simulation script, the platform and the deployment configuration files offers great flexibility for designing and running large scale experiments. The platform can be scaled up or down without changing the simulation script, for instance.

Another feature of SimGrid that we would like to mention here is that SimGrid provides access to NS-3 simulator. This has at least two main advantages. The first one is that for SimGrid users who like to have low level packet information about the simulated network phenomenon, they can launch their experiment from SimGrid interface while using their platform and deployment files that they have created within SimGrid. The second one is that for studies similar to ours this feature helps a lot during the validation process of newly implemented model. This feature allowed us to run the validation comparisons with NS-3 using the same platform and deployment file that we have created in SimGrid. SimGrid automatically maps the topology and the corresponding parameters into NS-3's abstraction.

3.2 NS-3

NS-3 is a discrete-event packet-level simulator, events corresponding to, for instance, arrival and departure of packets. NS-3 is structured in a modular manner. The core and the network modules are two of the modules that serve as generic simulation core that can be used for Internet-based or different network type simulation. These two modules, being generic, are independent from any device models. The core module provides features such as tracing, callbacks, smart pointer, random variables, events and schedules. The network module consists components such as packets, node, addresses (e.g., IPv4 and MAC) and network devices. The components provided by the simulation core modules can be used to create other modules. This feature allows researchers to add their own models for the network phenomenon that they want to simulate. We will visit two of the modules that are constructed in this way in the next two subsections [22].

The NS-3 core and other modules are built in C++ language as a set of libraries. The user can access these libraries in their main C++ program to configure the simulated topology and other simulator parameters. The libraries are also available as Python API for those researchers who prefer Python programming language.

3.2.1 ECOFEN Module

ECOFEN is one of the two non-core NS-3 modules that we used in our experiments. We explained the power consumption simulation features provided by this module in the Related Simulators section of Chapter 2 and we will give detailed explanation about why and where we have used it in our Method chapter. In this section, we only give brief description about how it is related to NS-3 and how we have used it.

NS-3 in its core provides an abstraction such as Node, Net Device, Channel and Application. A Node represents network communication and computing devices (currently NS-3 do not have CPU abstraction) such as servers, switches and routers. To a Node a Net Device, which represent devices such as network interface card (NIC), can be attached. Two or more Nodes can be linked to each other through a Channel, which is a representation of Ethernet or Wi-Fi link. These three abstractions: Node, Net Device, and Channel, together they can be used to define the simulated network topology. Application, on the other hand, is an abstraction that represent user program that perform some simulated activity such as sending or receiving UDP packets [22].

Using the core abstractions provided by NS-3, such as Node, Net Device, and Packet, the ECOFEN module implemented three power consumption models that enable users to simulate power consumption as a consequence of packets transmission at different levels of granularity as discussed in Chapter 2 and Chapter 4.

In a typical NS-3 simulator script, in its main function, we can recognize four common sections: (1) the section where we find statements that import the required core or other modules, (2) the section where the topology of the simulated network is defined, (3) the section where the simulated user application is defined, and (4) the section where statements related to running, starting, stopping and cleaning the simulation is specified. This is a rough approximation, certainly there are other statements such as those that are related to logging and tracing.

The NS-3 scripts that we used in our power consumption simulation experiments imported the ECOFEN module in their first section, set up and configured the energy consumption models in the second section, configured an application that send and receive UDP or TCP packets in the third section, and finally, in the fourth section, we stated when the simulation should start and end.

3.2.2 FlowMonitor Module

FlowMonitor is the other non-core NS-3 module that we have employed in our study. This module is designed with the aim of providing generic network traffic inspection facility. It provides researchers, who want to measure the simulated network efficiency, with standard performance metrics such as bit-rate, duration, delay, packet-size and packet loss ratio [7].

Among the performance metrics that are available in FlowMonitor module, the following are the ones that we have used in our simulation experiments.

- ***rxBytes*** to get the received bytes by a node,
- ***txPackets*** to get the transmitted packets by a node,
- ***timeFirstRxPacket*** and ***timeLastRxPacket*** to get the absolute time when the first packet and the last packets in the flow was received,
- ***timeFirstTxPacket*** and ***timeLastTxPacket*** to get the absolute time when the first and last packets in the flow was transferred and
- ***lostPackets*** to check if there are lost packets.

We used the above performance metrics to compute throughput (T) with the unit of Mega-bits per second (Mbps) and Packets per second(Pps) as shown in Equation 3.1 and Equation 3.2.

$$T_{Mbps} = rxBytes \times 8.0 \times 10^{-6} / (timeLastRxPacket - timeFirstRxPacket) \tag{3.1}$$

$$T_{Pps} = txPackets/(timeLastTxPacket - timeFirstTxPacket) \tag{3.2}$$

3.3 Other tools

We followed, partly, the literate programming and reproducible research approach proposed in [31, 36]. In this approach, the authors used two well-known tools: Git and Org-mode. A Git branching model is proposed in [36] that ease the synchronization of data and the code that generated the data. Org-mode, on the other hand, is employed as a literate programming tool for managing a laboratory notebook.

Org-mode[1] is a plain text mark-up language which is available as an extension to Emacs text editor. An Org-mode document can have different sections: a plain text, an executable code block and/or a data block. The code and the data blocks are active, meaning they can be evaluated (or executed) and as a result they can output the code or the data block as passive (plain text) form and/or the computational result of the evaluated code or the data block. This feature allows Org-mode to be a powerful tool for literate programming.

Whenever we wanted to do some experiment, we used Org-mode in our laboratory notebook to capture the experimental environment, such as, the objective of the experiment, the assumptions we made, the parameters used, the links referenced, and any other information relevant to our experiment. Within the same document we also put chunks of codes wherever we want them using programming languages such as bash shell, python, and R. What is more amazing is that Org-mode allowed us to name and call the executable codes from anywhere within the document with different input parameters, hence we were able to reuse previously written code blocks.

We used the shell script to run NS-3 and SimGrid simulation scripts and also to capture their outputs. We used Python to extract and format the data we want from the raw data produced by the simulators. Then we used R with its ggplot2 package to generate different plots and to do statistical analysis.

[1] http://orgmode.org/

Chapter 4

Methods

In this chapter we begin by first describing the common approaches followed by researchers for a variety of energy consumption experiments, their advantages and disadvantages. Then we present the approach we followed for our study and its justification.

4.1 Common Approaches

One approach for estimating energy consumption of a given network is by employing actual power meter to measure the power drawn by involved network and computing components. A good example for such case is the measurement that Fan and his team conducted [14]. In this study the authors have managed to monitor power consumption of several thousand of servers over a period of six months on real live workload. Mahadevan et al., in [24], have also done a similar power measurement on a production environment for studying power consumption behavior of networking devices such as switches and routers. If the measurements are done correctly, this approach produces the most real picture of the network under investigation compared to the other two approaches that we will discuss in subsequent paragraphs. However, this approach has certain inherent drawbacks. First, real production networks might not be available for experimentation. Even if they become available, the transient and varying nature of the production environment makes it hard to repeat the experiments. Second, we have little or no control over factors affecting the measured power consumption. We do not have the privilege of injecting or modifying the traffic or the workload in order to test different experimental hypothesis. To have a full control we need another approach.

Experimental testbed is another approach that researchers have used to study power consumption characteristics of different computing and networking devices. In this approach first a separate network is setup and configured solely for the purpose of conducting experiments. Then researchers make measurements by manipulating factors that affect power consumption according to the hypothesis that they want to test. Unlike the previous one, this approach offers greater flexibility over the experimental parameters. In the power measurement study scenario that we are discussing, the researcher can change parameters such as traffic rate, packet size, inter-packet time interval and transmission protocol used (TCP/UDP). Sivaraman et al. in [34], for example, setup experimental testbed for determining per-packet processing and per-byte receipt, storage, queuing, and transmission power consumption. The experiment setup involved hardware-based traffic generator (which gives fine grain control over parameters such as the packet size, inter-packet interval and data rate), NetFPGA[1] experimental router and digital oscilloscope for measuring the power draw of the NetFPGA router. A similar experiment but with commercial switches of different vendors is explained in [33]. The primary advantages of this approach is that the researcher can have full control over the experimental parameters provided by the tools involved in the testbed and experimental result can also be very accurate. The first disadvantage though is that it can easily become very expensive when we want to experiment on large-scale level. The second disadvantage is that experimenting on different scenario might require considerable reconfiguration and even a completely new testbed, which apart from limiting the flexibility, it can also be very costly, time and effort consuming. We need an approach which overcome these shortcomings. That is, we need an approach which gives full control over the experiment, which is reasonably accurate, less expensive and very flexible.

Simulation is the most widely used approach in computer network research [40]. It has several advantage compared to the other two approaches mentioned before. First, it makes it relatively easy, for instance, the study of the performance of non-existing network protocol or algorithm. One can propose and validate, by simulation experiment, a new energy-aware routing protocol or algorithm for wired or wireless networks. This is what Swain et al. [37] did in their new energy-aware routing protocol proposal for wireless sensor networks. Second, though it depend on the design of the particular simulator used, in general, simulation approach allows running large scale experiments that involve hundreds and thousands of nodes with less effort and cost compared to the other two approaches. In [26] and [12], the NS-3 module, ECOFEN, is used to simulate energy consumption of large-scale

[1]http://www.netfpga.org/

networks with nodes more than 600 and 1000, respectively. In [20] Kliazovich et al. studied energy consumption of data center networks with two-tier and three-tire architectures that encompasses 1536 nodes. Third, in simulation scaling does not incur monetary cost, though it is limited by performance factors such as runtime and memory usage [40]. Fourth, the researcher has great flexibility and full control over the simulation experiment. Finally, simulators makes output data management extremely easy by providing mechanisms such as logging, tracing and visualization [8, 22].

Though simulation experiment has quite a lot of advantages over experiments done on production environment or experimental testbeds, it faces one big challenge, accuracy. In the process of approximating the real network phenomenon in the simulation model, some less significant concepts are abstracted away, for instance, to reduce complexity or to gain performance improvement, which results in unavoidable loss of accuracy. However, in other instances the models used in a given simulator might fail to correctly capture the simulated real network phenomenon. In [39] the authors demonstrated incorrect modelings found in popular simulators such as OptorSim, GridSim and CloudSim. Therefore, (in)validating the correctness of a simulator is important task that should be undertaken before any simulation experiment for two related reasons. Either to know the boundaries within which the simulator used produce reasonably accurate results, or to know if the simulator produce the expected or the accurate result. The validation can be done either by comparing the output of the simulator against accurate measurements obtained from real networks or by comparing the output against another simulator whose accuracy is already known [19].

4.2 Our Approach

The goal of this study is to investigate the accuracy and scalability of flow-level models, as compared to packet-level models, in estimating energy consumption of large-scale distributed networks. To achieve this goal, we first search literature to find and propose a suitable flow-level model. Second, we implement the model in SimGrid. Finally, we run different experiments to test the accuracy and the scalability of the implemented flow-level model by comparing it against a packet-level model. For our experiments, we chose the simulation approach among the three alternatives discussed above.

Before describing the details of our approach, let us first justify why we end up with the relatively complex method shown in Figure 4.1. There is experimental test-bed (Grid'5000[2]) in France that we have access to. Grid'5000 is experimental test-bed specifically designed for studying large-scale distributed networks [6]. However, we could not used it for our purpose (i.e., for studying large-scale flow-level relationship of power consumption and traffic) as the network devices are not equipped with power meters accurate enough (current power meters on Lyon site of Grid'5000 provide one measurement per node and per second). As a result, we opted to use a packet-level simulator with power consumption models obtained from literature. Subsequent paragraphs describe the specific steps we followed in our approach.

As we have discussed in Chapter 2, SimGrid already have energy consumption model for CPU which corresponds to the computing part of a given large-scale network. What we wanted to add is energy consumption model for communication components such as switches and routers. Therefore, the initial task in our approach is to study literatures (Ⓐ in Figure 4.1) in order to find and propose a model which describe the power consumption characteristics of communication equipments such as switches and routers. Our search returned the linear relationship that we have described in Equation 2.2 [4, 23, 24, 34]. This equation tells us that the power consumption of a network equipment constitutes the idle and dynamic components. The idle power consumption represents the power drawn by the equipment while it is on but with no traffic. The dynamic consumption, on the other hand, represent the additional power drawn due to network traffic. The next task (Ⓒ in Figure 4.1) is to implement this linear model for SimGrid and (in)validate its accuracy against ECOFEN module (Ⓓ in Figure 4.1) [12, 26]. This task is done iteratively by switching between model implementation and accuracy validation. The final task (Ⓖ in Figure 4.1) is to show the scalability of the implemented flow-level model against the existing packet-level model in ECOFEN. For this, we designed and run two kinds of experiments (Ⓔ and Ⓕ in Figure 4.1), one for speed and one for memory usage.

The purpose of the accuracy and the scalability experiments shown in Ⓓ and Ⓖ in Figure 4.1 is to test our *hypothesis*, which states: *flow-level energy consumption models can give reasonably accurate estimation and they can also be significantly more scalable than packet-level models.*

We chose to use ECOFEN as packet-level simulator to compare the accuracy and performance of the implemented model for two primary limitations that exist in the other alternative simulator, GreenCloud [20]. The first limitation is that GreenCloud is designed for a cloud computing environment. This is in contrary to one of SimGrid's main design principles, versatility [8].

[2]https://www.grid5000.fr/mediawiki/index.php/Grid5000:Home

ECOFEN, on the other hand, is not tied to one particular large-scale networking paradigm, therefore, suits more for our purpose. The second limitation of GreenCloud is that it is build on top of currently obsolete NS-2 simulator. In comparison, though ECOFEN was also initially built as NS-2 simulator module, currently it is rewritten for NS-3 [12]. One of the major advantage of using NS-3 over NS-2 is that NS-3 performs considerably better in both runtime and memory-usage metrics [40].

In the accuracy-validation and scalability-comparison experiments mentioned in our approach, we are comparing the newly implemented flow-level model in SimGrid simulator against another packet-level simulator model implemented in ECOFEN module. This simulator-to-simulator comparison is valid only if the later simulator model, against which the new implementation is to be validated, is known to be accurate. However, we could not find any information that tell us the accuracy of the ECOFEN module. Therefore, we designed a validation experiment (Ⓑ in Figure 4.1) for ECOFEN as described in the next section.

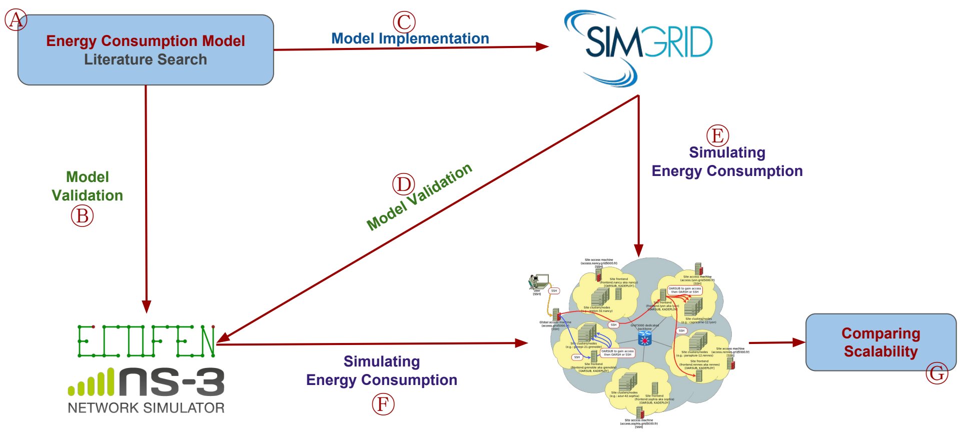


Figure 4.1: Summary of the experimental method we followed in this study

4.3 Validating ECOFEN

ECOFEN has three models with names *basic*, *linear* and *complete* that we have discussed in Section 2.7 of Chapter 2. Both the *linear* and *complete* models can produce values of power consumption as a function of traffic. The *basic* model, on the other hand, produces power consumption values based on the ON or OFF state of a node, it does not consider network traffic. Therefore we describe the validation experiments for both the *linear* and the *complete* models in this section.

The basic procedure for the validation experiment is first to simulate, using ECOFEN, power consumption in response to traffic sent or received and then to compare the results against data obtained from literature where actual measurement is conducted.

4.3.1 Validating the Linear Model

In the work of Sivaraman et al. [34] we can find the result of a power consumption experiment that is shown in Figure 4.2. The figure displays the linear relationship that exist between traffic volume (in Mbps) and power consumption (in watts) for a fixed packet sizes of 100, 576, 1000, and 1500 bytes. Furthermore, in the figure, the linear fit equations (models) for each of the packet sizes are also displayed.

The authors intention in this experiment is to determine values of the per-byte and the per-packet processing energy consumption, however, our intention is to use the per-byte energy consumption value that they have experimentally determined and to use it in ECOFEN to get power consumption values for a given volume of traffic. Then compare the results we obtained with the linear fit models that are shown in Figure 4.2. The linear fit equations shown in Figure 4.2 are derived from actual power measurements.

In their experiment, the authors used three kinds of hardware devices: (1) NetFPGA router card that has four 1 Gbps Ethernet ports, (2) IXIA hardware traffic-generator for generating packets with the desired packet-size and data-rate, and (3) high-fidelity oscilloscope for measuring the power consumed by the NetFPGA card as a consequence of the packets send or received.

The linear model of ECOFEN module accepts energy consumption values for the idle state of a simulated network interface card (NIC) and the per-byte processing. The underlying NS-3 platform, in addition, provides us with more parameters such as packet-size and data-rate, which among other parameters, enable us to have full control over the generated traffic.

In this validation experiment we wish to simulate the experiments conducted by Sivaraman et al. as closely as possible. With this in mind, we setup, in our NS-3 simulation script, a three node simple network with first and third nodes connected to the second node. All the three nodes are connected to each other by links that have maximum bandwidth capacity of 1 Gbps and delay of 10 ms.

We got the idle consumption values for each of the packet-size models

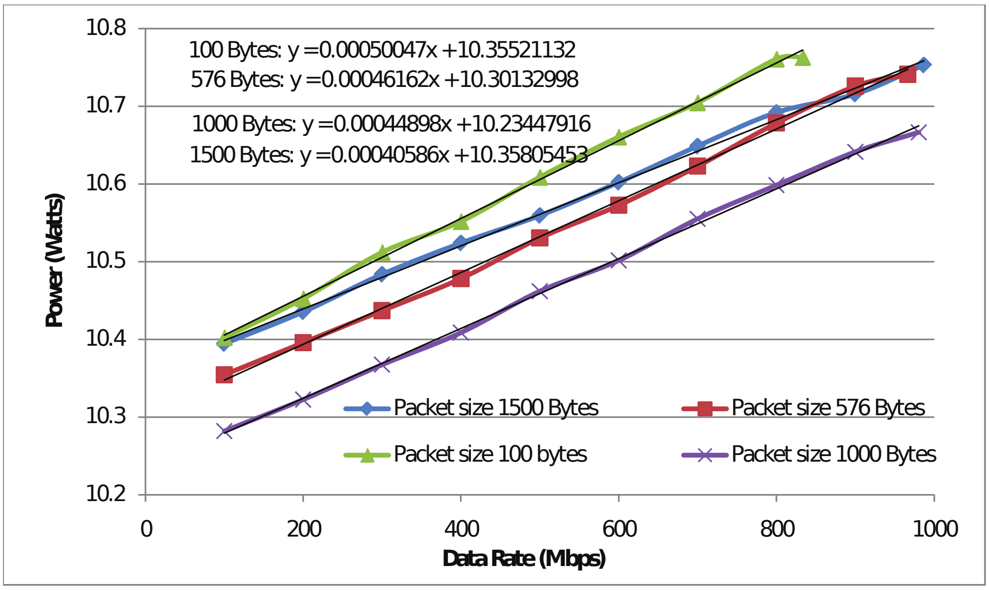


Figure 4.2: Power consumption vs data-rate for fixed packet size from [34].

shown in Figure 4.2 by setting the x component (the data rate value) of the equations to zero and for a per-byte processing energy consumption value we used 3.4 nJ. This is the value that the researchers experimentally determined.

For the generated traffic volume in the simulation, we used uniform random number generator provided by NS-3 in order to get integer values between 1 and 1000. Our NS-3 script, in addition to packet-size and data-rate values, also requires number of packets to be send and also inter-packet interval time values. These values are derived from packet-size and data-rate parameters.

Since there might be unexpected results, for instance, due to wrong network configuration, we have employed NS-3's FlowMonitor module to monitor the actual traffic transfered in the simulated network. Using this flow monitoring module, we have confirmed if all the traffic generated by the sending end are also received at the receiving end. We have also used this module to compute the actual traffic rate (throughput) both at the receiving and the sending ends as shown in Equation 3.1 and Equation 3.2 in Chapter 3.

Finally, we set the remaining simulation environment configuration settings such as starting and stopping time and then run the experiment 40 times, each time with different run value for the random number generator. The result obtained is depicted in Figure 4.3. In the graph the expected power consumption values from the linear fit models shown in Figure 4.2 along with the simulated values for each of the packet sizes (100, 576, 1000,

Packet Size	Confidence Interval of difference in mean	Mean of Expected	Mean of Simulated	P-Value
100	[-0.027, 0.110]	10.640	10.599	0.230
576	[-0.039, 0.082]	10.544	10.523	0.480
1000	[-0.043, 0.073]	10.466	10.451	0.6131
1500	[-0.062, 0.048]	10.566	10.573	0.796

Table 4.1: Unpaired t-test results for simulated and measured power consumption values for ECOFEN's linear

and 1500 bytes) are displayed.

Visually, the simulated and the expected values seem to agree very well, even though the gap between them starts to grow slightly larger (especially when the packet size is 100 bytes) for larger data-rates. In order to be more sure, we run unpaired t-test statistical test using the produced data. The summary of this test is shown in Table 4.1.

The 95% confidence interval values shown in Table 4.1 of difference in mean between the measured and simulated values are very close to zero and in fact zero is also one of the values. The P-values are also confirming the same thing, the null hypothesis that the difference in mean between the simulated and the expected values is zero is not rejected.

The conclusion in this validation test is that the linear model of ECOFEN is accurate in predicting the power consumed by NetFPGA router for a given volume of traffic.

4.3.2 Validating the Complete Model

Roughly, in this validation experiment, we have used the same experimental configuration and procedure as that of the linear validation experiment that we have described in the previous section. Therefore, in this section our focus is more on the result than on the configuration.

One of the main difference between the complete and the linear model of ECOFEN is that the complete model distinguishes between the received and sent bytes. Which means that different energy consumption values can be assigned to bytes based on the direction of transfer. The linear model, on the other hand, assigns same value for both. The other main difference is that the complete model considers the packet processing energy consumption cost both for the sent and received packets.

Sivaraman et al. [34] also conducted experiments to determine energy consumption values for per-byte receive or transmit and per-packet process-

Packet Size	End	**Confidence Interval of difference in mean**	**Mean of Expected**	**Mean of Simulated**	**P-Value**
576	Rx	[-0.067, 0.039]	10.770	10.784	0.598
1500	Rx	[-0.010, 0.095]	10.778	10.736	0.114
1000	Tx	[-0.096, 0.053]	10.750	10.773	0.560

Table 4.2: Unpaired t-test results for simulated and measured power consumption values for ECOFEN's complete model

ing. The experimentally determined values for per-byte receive is 1.3 nJ, for per-byte transmit is 2.1 nJ, and for per-packet processing is 197.2nJ.

We have slightly modified the NS-3 script that we have used in the previous section to make it suitable for this experiment. Now we have configured our script for the ECOFEN's complete model to use energy consumption values for per-byte receive or send and per-packet processing. Further more, we have upgraded the link capacity between the nodes from 1 Gbps to 2 Gbps. Finally, we set the traffic rate in terms of packets per second in the sending end and in terms of Mbps in the receiving end in order to comply with the experiments done by mentioned authors. The results for the sending and receiving are is shown in Figure 4.4 and Figure 4.5, respectively. The linear fit models we have used for this validation experiment are also available in [34]. There is only one linear fit model (for packet size 1000 bytes) for the sending end and there are three for the receiving end. Table 4.2 shows the unpaired t-test result for one packet-sizes in the transmitting side (Tx) and for two packet-sizes in the receiving side (Rx).

Again in this case the 95% confidence interval values shown in Table 4.2 of difference in mean between the measured and simulated values are very close to zero and in fact zero is also one of the values. The P-values are also confirming the same thing, the null hypothesis that the difference in mean between the measured and the simulated values is zero is not rejected.

The conclusion from this validation experiment is also the same as the previous one, the ECOFEN's complete energy consumption model accurately predicts power consumed by NetFPGA router for a given volume of sent or received traffic.

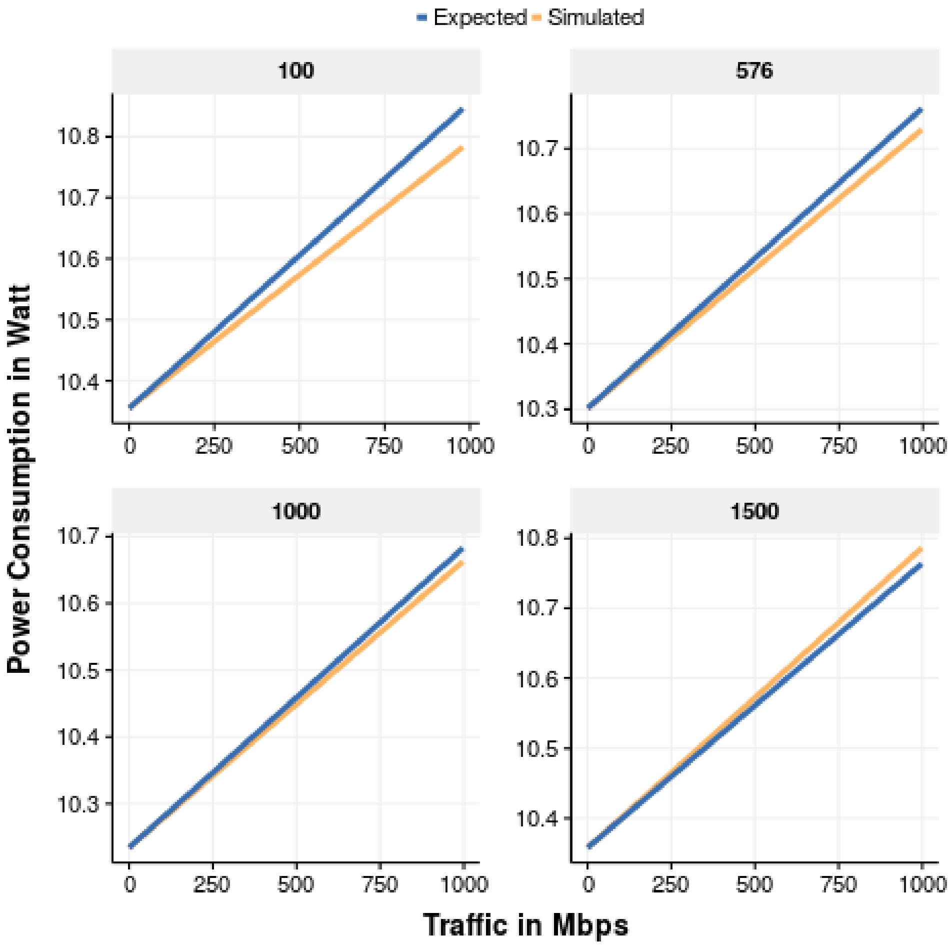


Figure 4.3: Power consumption vs data-rate comparison between expected (or measured) values (in red color) and simulated values (in light blue color) for a fixed packet size of 100, 576, 1000, and 1500 Bytes

.

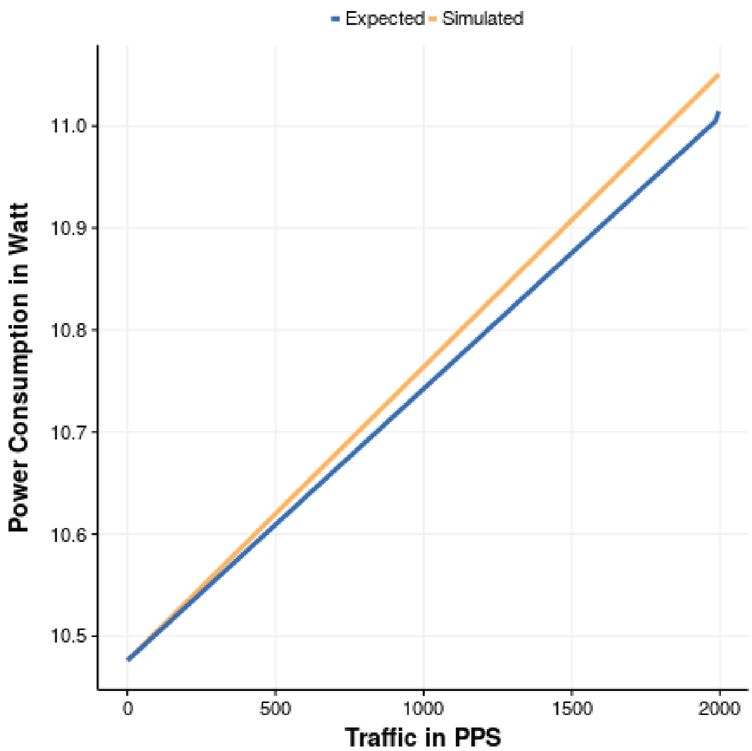


Figure 4.4: Power consumption vs data-rate comparison between expected (or measured) values (in red color) and simulated values (in light blue color) for a fixed packet size of 1000 Bytes for the sending end

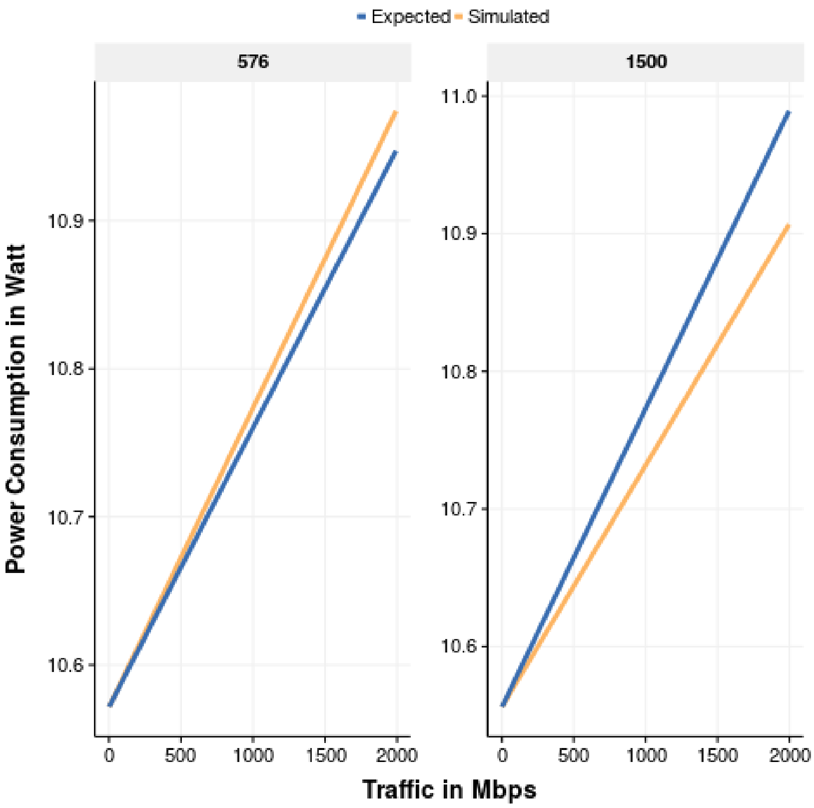


Figure 4.5: Power consumption vs data-rate comparison between expected (or measured) values (in red color) and simulated values (in light blue color) for a fixed packet size of 576 and 1500 Bytes for the receiving end

Chapter 5

Implementing flow-level model

In this chapter we start by describing how we designed the energy model, we then move to the discussion of the implementation details. Finally, we present the challenge we faced during implementation.

5.1 Modeling energy consumption

By definition, as we have shown in Equation 2.1, the energy consumption of a given electronic equipment is given by the product of power and time. We have also shown, in Figure 2.4 and Equation 2.2, the linear relationship that exist between network equipment load and power consumption. Combining Equation 2.1 and Equation 2.2, we get Equation 5.1 for energy consumption of a given device for a given time duration T, where P_{idle} is the power that the equipment consumed when there is no traffic and $P_{dynamic}$ is the additional power drawn due to network traffic.

$$E(T) = \int_0^T (P_{idle} + P_{dynamic})(t)dt \tag{5.1}$$

5.2 SimGrid's link energy model

To implement the model shown in Equation 5.1 in SimGrid, we need to determine the values of the three variables. We can directly read the idle power consumption value from the SimGrid's link property that we described in Section 3.1. For the $P_{dynamic}$, we need to describe how SimGrid computes load in its core.

Briefly described, for a set of simulated activities running on a given simulated network resource such as a switch, SimGrid computes, using its bandwidth sharing algorithm, the amount of resource share that each activity can get. The sum of the resource share that all activities can get for a given resource at a given moment cannot exceed the capacity of the resource on which the activities are running. Figure 2.5 depicts this concept symbolically. We can access how much of the resource is currently in use, i.e., its load, from SimGrid's core library. SimGrid dynamically recomputes the resource usage when any of the allocated activities finishes their data transfer task.

We can compute the dynamic power consumption, $P_{dynamic}$, of a link at any given instance as shown in equation Equation 5.2. Similar to the idle value, we can also read the busy power consumption value, P_{busy}, directly from the link property description and u is link utilization computed by dividing the load (used bandwidth) by the maximum bandwidth. The maximum bandwidth value is also available on the link description.

$$P_{dynamic} = (P_{busy} - P_{idle}) * u \tag{5.2}$$

where:

u : is a normalized utilization factor obtained by dividing the current Link load with its full capacity, and

($P_{busy} - P_{idle}$) : is the slope of the relationship between load and power consumption as shown in Figure 2.4.

In Chapter 3, we mentioned that SimGrid provides an interface to NS-3. In order to take advantage of this feature, especially during the validation phase, we took into account how the SimGrid's links are mapped to NS-3's abstraction. In SimGrid there is no NIC (or NetDevice in NS-3's term) abstraction. But the interface maps SimGrid's link into two NS-3 NetDevices. Therefore, the idle and busy values of Equation 5.2 are multiplied by two for each link in our implementation.

To compute the total energy, $E(T)$, of a network during the time interval T, the approach we followed is that each time an event happens on a link (link created or destroyed, link turned on or off, or link load change, or simulation ended), we read the link load and multiply it by the time elapsed between the current event and the previous event. This gives the energy consumed between two events. We collect this value for all events happened during time duration T for each link and finally, when the simulation ends, we collect all the energy values for all the links. This value gives us the total energy consumption, $E(T)$ of the simulated network.

5.3 Implementation Challenge

One major problem that we faced during our preliminary validation experiment of this implementation is that there is discrepancy between the simulated time value of SimGrid and ECOFEN. Figure 5.1 shows this discrepancy when 200 MBytes of data is transferred at different bandwidth level (ranging between 10 and 500 Mbps). Approximately, as shown in the Figure 5.1, for bandwidth value of below 100 Mbps both simulators seems to agree on the simulated time but above 100 Mbps, ECOFEN stays constant while SimGrid keeps on decreasing at a slower rate. We have also confirmed that the time predictions stays close below 100 Mbps for varying data sizes (20 to 500 MBytes) by testing at two different bandwidth values (10 and 50 Mbps).

Since the model shown in Equation 5.1 depends on time value and since we are also going to validate this model against ECOFEN, for all accuracy and scalability validation experiments presented in the next chapter, we decided to use bandwidth values residing below 100 Mbps where both SimGrid and ECOFEN seems to agree. The task of figuring out why these two simulators seems to predict time differently is beyond the scope of this work.

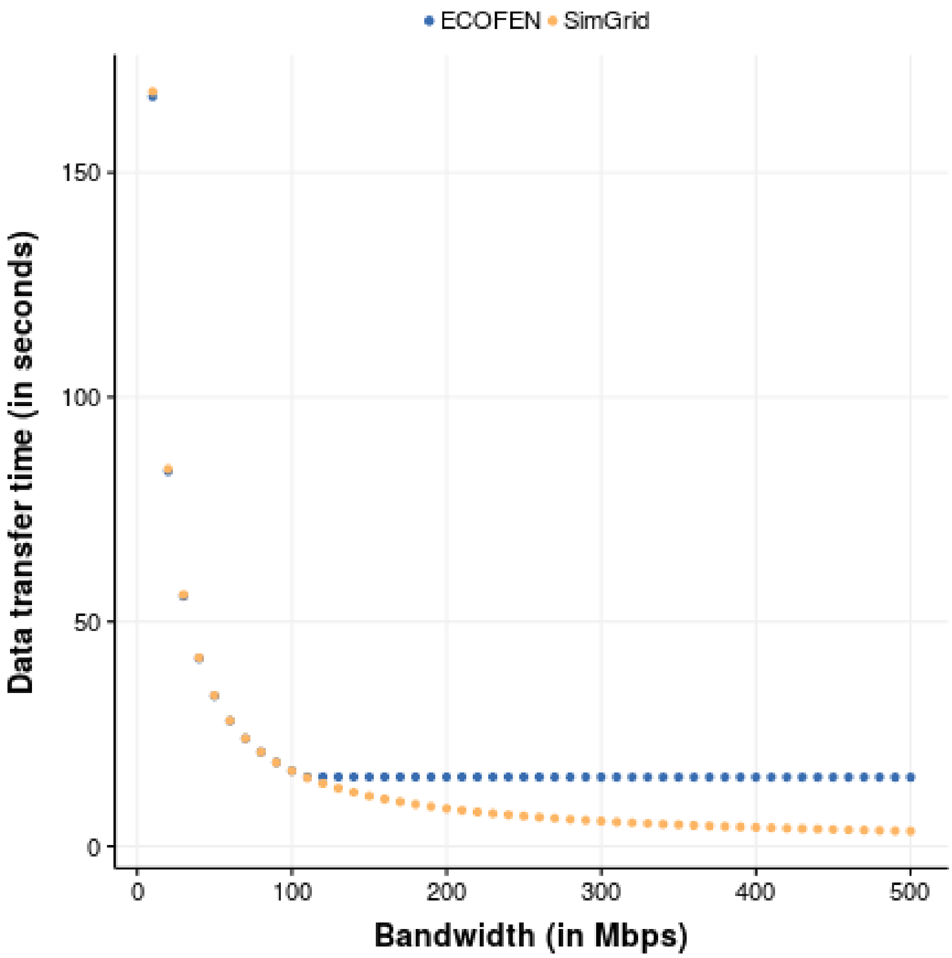


Figure 5.1: Simulated time required to transfer 100 MBytes of data at different bandwidth settings: comparison between ECOFEN and SimGrid simulators.

Chapter 6

Validation

In this chapter we present the accuracy and the scalability experiments we performed to evaluate the implemented flow-level model. As we outlined in our method chapter, we compare our implementation with the packet-level implementation of ECOFEN (based on NS-3).

6.1 Experiment Setup

For all our experiments the version of NS-3 we used is version 3.26 and that of SimGrid is version 3.15.

6.1.1 General ECOFEN related setup

Among the three power consumption models available in ECOFEN module, we used the *linear* one. This model accepts idle power consumption values in Watts and also energy consumption values for each Byte received or send in nanoJoule. Throughout our experiments we used 10.3581 Watt as idle power consumption value and 3.423 nJ as Byte energy consumption value. We took these values from [34], even though any arbitrary values can also suffice for our purpose.

The linear model estimates power consumption value for a given NIC (or NetDevice in NS-3's term) based on these two inputs and the amount of transmitted or received traffic. It then displays the estimated power consumption in Watt at specified time interval. We set the interval to be 0.05 sec (20 estimations per second) in order to get more frequent power estimation.

What we want to get from ECOFEN is the total amount of energy consumed by the simulated network. To get this value, we first compute the average power drawn by the network for a given data transfer task and when the task ends, we multiply the average total network power with the data transfer time.

6.1.2 General SimGrid related setup

For all of our experiments, we used a simple client/server SimGrid application. As described in Chapter 3, a typical SimGrid script contains three sections. Accordingly in our script, the first section represents what the client function does to send data. It simply accepts the number of Bytes to send from command line option or uses a default 1,000 Bytes if no value is specified, or it will generate the Bytes if random option is specified in the command line. Then it passes the Bytes to the SimGrid's send routine. By default there will be only one TCP flow for the specified Bytes, but if we want to send multiple flows we can also specify the number of flows on the command line. The server function simply issues the receive routine if there is data to receive. The third section, in addition to doing the tasks specified in Chapter 3, it is also a gateway to NS-3 or our implemented flow-level model depending on the option specified in the command line.

For the link tag in SimGrid's platform file we used 10 Mbps as a bandwidth value. This value is chosen because it falls within the range where ECOFEN's and SimGrid's simulated time value match closely as we have described by the end of Chapter 5. For latency we used 10 ms and as a Watt-range power consumption value, we used 10.3581 Watt as idle power consumption value and 10.7479 Watt as a busy power consumption value.

SimGrid's interface to NS-3 will set these bandwidth and latency values, together with other parameters, to NS-3's configuration. It means that we have only one SimGrid script to run both simulations (flow-based in SimGrid directly and packet-level in NS-3 with ECOFEN), thus ensuring a fair comparison on the same simulated network with the same generated traffic.

6.2 Accuracy Validation

Our first objective consists in evaluating the accuracy of SimGrid's energy consumption estimates (done with our energy flow-based model) against the values computed by ECOFEN. For this accuracy validation, we conduct two sets of experiments. In the first set, our purpose is to investigate the accuracy of energy consumption estimation difference between ECOFEN and the flow-level model when the size of platform changes. In SimGrid case, it means when the number of links change; in ECOFEN case, when the number of Nodes, NetDevices and the connection between them change. For this

Scenario	Number of Links	Number of Flows	Data size(MB)	Bandwidth (Mbps)	Latency (ms)
1L1F	1	1	[20,500]	10	10
1L2F	1	2	[20,500]	10	10
1L4F	1	4	[10,100]	10	10
3L1F	3	1	[20,200]	10	10
3L2F	3	2	[20,100]	10	10

Table 6.1: Scenarios tested for accuracy validation of the implemented flow-level model against ECOFEN. In the first column L stands for Link and F stands for Flow, hence 1L1F stands for one-link/one-flow scenario.

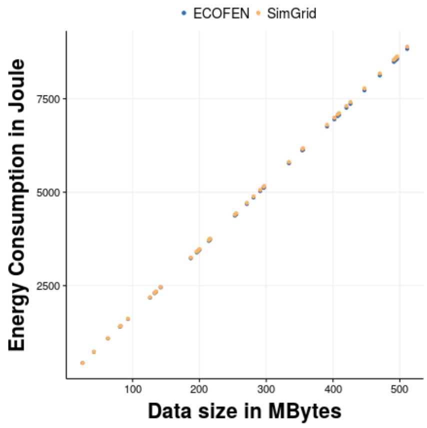
experiment we keep the number of flows and the data volume fixed but we increase the number of Links. In the second set, on the other hand, our purpose is to study the difference between the estimated energy consumption value between ECOFEN and SimGrid when the data size or flow changes while keeping the number of links fixed. Table 6.1 shows all the tested scenarios and Figure 6.1 shows the energy consumption prediction behavior of the implemented model and ECOFEN's model for all of the scenarios summarized in Table 6.1.

In order to compare the energy consumption prediction accuracy of our flow-level model compared to the packet-level model of ECOFEN, we employed the unequal variance t-test method (a.k.a Welch's t-test) as suggested in [30]. The reason for choosing this test is that for our data sets obtained from the two simulators, we cannot assume equal variance, as the two data sets are independent. Table 6.2 contains the statistics obtained from this test using R's built in Welch t-test function.

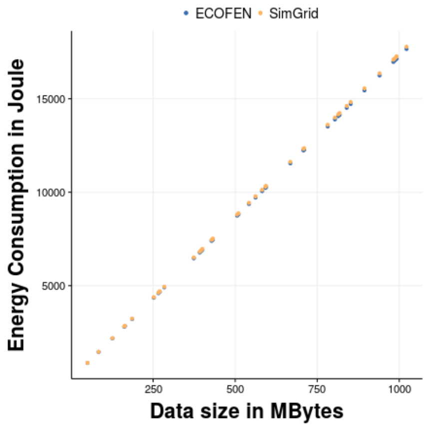
For all accuracy scenarios that we have tested, Figure 6.1 demonstrates the closeness in energy consumption estimation between the implemented flow-level model and the packet-level model used for validation. From the last column of Table 6.2 we can see that the maximum estimation error that our flow-level model registered is approximately 0.3%, which is a very good estimation. The P-values shown in the fifth column also confirms that there is no statistically significant difference between the flow-level and the packet-level estimation.

6.3 Scalability Validation

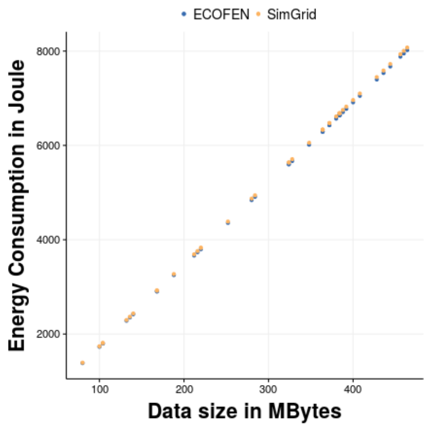
For validating the scalability of the implemented flow-level model, we run also two sets of experiments. In the first set, our goal is to investigate the



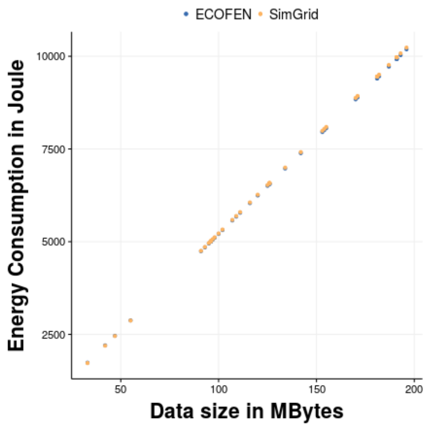
(a) 1 Link 1 Flow



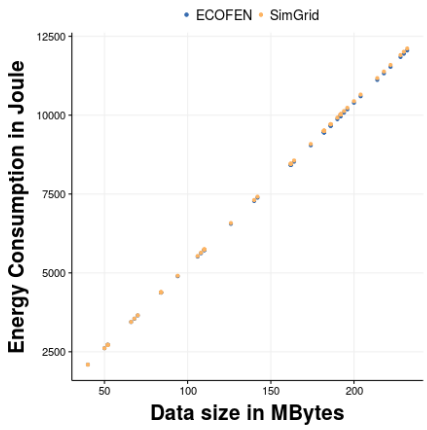
(b) 1 Link 2 Flows



(c) 1 Link 4 Flows



(d) 3 Link 1 Flows



(e) 3 Link 2 Flows

Figure 6.1: Predicted energy consumption comparison of ECOFEN (blue dots) and SimGrid (orange dots) as a function of transferred Bytes for different path length and flow amount.

Scenario	**Mean of ECOFEN**	**Mean of SimGrid**	**CI of Difference in mean**	**P-value**	**% of Difference**
1L1F	4837.2	4869.6	[-1156.3,1091.5]	0.9544	0.283
1L2F	9672.6	9739.0	[-2314.2,2181.3]	0.9532	0.295
1L4F	5250.8	5286.9	[-720.10,647.90]	0.9169	0.297
3L1F	6804.9	6828.8	[-1024.9,977.1]	0.9622	0.124
3L2F	7896.6	7931.9	[-1061.4,990.6]	0.9457	0.168

Table 6.2: Unequal variance t-test statistics obtained using R. The confidence interval (CI) values in the fourth column are computed for the difference of the two energy consumption estimations and the last column values are obtained from mean log error as explained in [39].

scalability of the model when the path length increases. For these experiments we use path length values of 1, 2, 4, 6, 8 and 10. We fix the number of flows at 2 and the data size at 200 MB. In the second set, we examine the scalability as the data size increases. In this case, we keep the path length at 1 and the number of flows at 2, but we vary the data size randomly between 50 and 550 MB. The total number of data size values within this range was 21.

Both of these experiments were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations[1]. The machine we used is SUN FIRE X2270 which have Intel Xeon X5570 2.93 GHz 2 CPUs, 4 cores per CPU and 24 GB RAM running Debian version 8 (jessie) operating system.

For each set of experiments, we use the Debian command, /usr/bin/time, to collect simulation run time and peak memory usage data. We run each experiment within each set seven times i.e., 7 run for each link in the first set and 7 run for each data size in the second set. Figure 6.2 shows the runtime and memory usage comparison as the path length increases. Figure 6.3, on the other hand, shows the runtime and peak memory usage comparison as the data size increases.

The packet-level simulator curves shown in Figure 6.2 and Figure 6.3 follows a linear growth in both simulation time and memory usage metrics whereas the flow-level model stays constant and below the packet-level curve. From Table 6.3 we can see that the flow-level model is at least 243 times faster than the packet-level simulator and it is also at least 2 times more memory

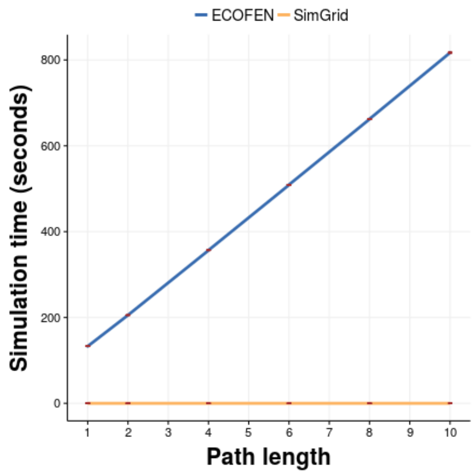
[1]https://www.grid5000.fr/mediawiki/index.php/Grid5000:UsagePolicy

Link	Flow	Data size	Average seconds SimGrid	Average seconds Ecofen	Runtime efficiency of Simgrid
1	2	100	0.3	132.95	443 times
10	2	100	0.3	817.02	2723 times
1	2	111	0.3	74.14	243 times
1	2	530	0.3	351.62	1172 times

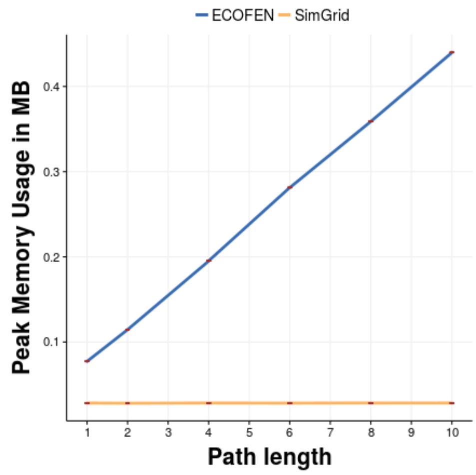
Table 6.3: Simulation time (runtime) comparison of SimGrid and ECOFEN. The first and the second rows compare at minimum and maximum path length values whereas the third and fourth column compare at minimum and maximum data size values. At each row the data size value has to be multiplied by the flow number to get the total data size.

Link	Flow	Data size	Average memory SimGrid	Average memory Ecofen	Memory efficiency of Simgrid
1	2	100	0.028	0.077	2.7 times
10	2	100	0.028	0.44	15.5 times
1	2	111	0.028	0.06	2.12 times
1	2	530	0.028	0.15	5.4 times

Table 6.4: Peak memory usage in mega Bytes (MB) comparison of SimGrid and ECOFEN. The first and the second rows compare at minimum and maximum path length values whereas the third and fourth column compare at minimum and maximum data size values. At each row the data size value has to be multiplied by the flow number to get the total data size.



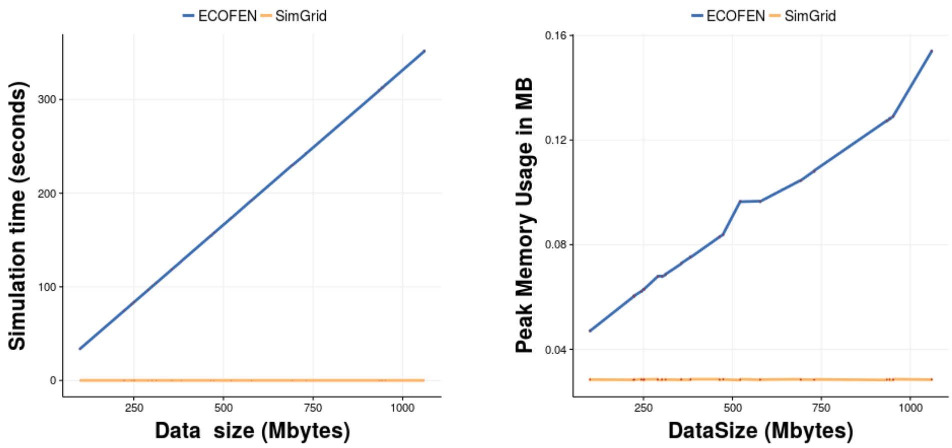
(a) Runtime



(b) Peak Memory Usage

Figure 6.2: Run time and peak memory usage comparison of ECOFEN and the flow-level model as the path length increases. In the figure the confidence interval for each experiment is also shown as a bar.

efficient. These results clearly shows the validity of our model for studying energy consumption of large-scale networks.



(a) Runtime

(b) Peak Memory Usage

Figure 6.3: Run time and peak memory comparison of ECOFEN and the flow-level model as the data size increases. In the figure the confidence interval for each experiment is also shown as a bar.

Chapter 7

Discussion

In this research our objective was to investigate the level of accuracy and scalability obtained from flow-level models for estimating energy consumption of large-scale network. In order to achieve this goal we layout a research method that describes the steps that we followed from start to finish.

Following our method, we started by studying literature to explore the state-of-the-art in the area of energy consumption of large-scale networks and the simulation frameworks available for estimating the consumption. From our literature study, we have learned that the energy consumption of a network equipment is characterized by two properties: idle power consumption and dynamic power consumption as a result of data transfer task. We have also learned that there are very few packet-level simulators proposed for estimating energy consumption of large-scale networks.

It is already known that packet-level simulators are more accurate in modeling a given network phenomenon compared to flow-level simulators but less scalable in terms of runtime and memory usage performance metrics [21]. The accuracy of packet-level simulators comes from the detailed information they capture about simulated network phenomenon, they strive to capture packet-level details of the simulated network phenomenon. Flow-level simulators, on the other hand abstract away low-level details and model a given network phenomenon using analytical equations. Loosing low-level details allows flow-level models to scale well when the size of the simulated network increases.

Due to this trade-off between level of details and scalability between the two simulation approaches, in our research, we stated the following hypothesis in order to investigate the level of accuracy and scalability we can get from flow-level models.

"flow-level energy consumption models can give reasonably accurate estimation and they can also be significantly more scalable than packet-level

models"

In order to test this hypothesis, we first implemented a flow-level energy consumption model that we found in literature for SimGrid and then conducted accuracy and scalability experiments as we have described in Chapter 6.

The results of the accuracy validation experiments show that in fact very good energy consumption estimation accuracy can be obtained using flow-level models. For all the five scenarios run, the observed relative error lies approximately between 0.1% and 0.3% compared to the packet-level model used in ECOFEN. Our unequal variance t-test statistical test with p-value around 0.9 also tells that statistically the estimation difference between the packet-level simulator and the flow-level simulator is not significant. This confirms the first half of our hypothesis that flow-level models can give reasonably accurate estimations. The condition that should be satisfied in our model to let this hypothesis hold true is, correct time prediction, since our analytical equation (flow-level model) uses simulated time as one of its parameter.

During our experimentation, we have noticed that there is significant difference between ECOFEN (or NS-3) and SimGrid on predicting the simulated time required to transfer a given amount of data. For a given data transfer, at a given latency value, SimGrid's predicted time continues to decrease as bandwidth increases whereas ECOFEN stops to decrease approximately when the bandwidth goes above 100 Mbps as shown in Figure 5.1. Since addressing this problem is beyond the scope of our work, the approach we followed to avoid estimation error due to this time discrepancy is that we restricted all our experiments to bandwidth values where the two simulators closely agree on predicted time values.

Another point we like to point out about our accuracy validation experiments is that, there are additional tests that should be conducted to validate the correctness of the implemented model in different data transfer scenarios. For example SimGrid support simulating simultaneous TCP flows in same or different directions. All of the flows that we have used in our experiments were in the same direction.

Concerning the scalability, the runtime and peak memory usage experiment results confirmed the second half of our hypothesis which states that flow-level models are significantly more scalable than their packet-level counter part. With the runtime performance metrics in Table 6.3, we showed that SimGrid is 243 to 2723 times faster than ECOFEN. Similarly in Table 6.4 we have shown that SimGrid is 2 to 15 times more memory efficient than ECOFEN. Actually the scalability performance did not come directly from our implementation. It is due to the scalability of the underlying flow-level model of SimGrid. The scalability of SimGrid is already confirmed in other research works performed in the area of large-scale distributed systems [8, 28].

Two of the limitations of our work is related to SimGrid's TCP model. The first one is that the TCP model in SimGrid can work in both half-duplex and full-duplex mode, however, our implementation works only in half-duplex mode. The second one is that SimGrid uses different bandwidth sharing options that determine how the bandwidth is shared among the flows traversing a given link. In our implementation, we have only considered the option which shares bandwidth fairly among the flows.

The other limitations of our work come from our scope. We have first limited our study to communicating components of large-scale networks. There are other energy consuming components such as data-center infrastructure facilities (which includes power provisioning, cooling and lighting components) that should be modeled in order to give full energy consumption estimation of a given large-scale network. Our work was also limited to the wired network components. As a result, we have not considered any communicating components that are involved solely on the wireless network, such as, Wi-Fi access points.

Using our implementation together with already existing power consumption model of SimGrid for CPUs, it is possible to simulate energy consumption of computing and communicating components of large-scale networks. However, using present implementation, it is only possible to experiment on one kind of energy saving techniques: switching links ON/OFF. One can experiment on the effects of switching network components on or off to study the resulting energy cost difference.

Other experiments, such as, investigating the effects of activating adaptive link rate energy saving mode on a link, can not be conducting using the current implementation. However, the current implementation can be extended to allow this feature by following the P-state approach implemented in the existing CPU energy consumption model. Similarly to CPU's P-state, a link can have multiple data transmission rate levels.

We would like to recommend three areas for future researchers who would be willing to extend our work. First, to include in our model other features of SimGrid's TCP model such as full-duplex and different bandwidth sharing options. Second, to consider other energy saving levers such as adaptive link rate. Third, to propose and implement flow-level energy consumption model for wireless network components following the method that we proposed in this manuscript.

Chapter 8

Conclusions

In this thesis, we aimed at investigating the level of energy estimation accuracy and performance scalability that can be obtained from flow-level energy consumption models for wired network devices. In order to achieve this goal, we outlined a research method and, following this method, we designed, implemented, and validated flow-level energy consumption models for SimGrid simulator.

Using our validation experiments, we have shown that the implemented flow-level model exhibits less than 1% estimation error compared to its packet-level counter part. Furthermore, it is also at least twice faster than the packet-level simulator. From these results, we can conclude that given accurate simulated time prediction, our flow-level model gives very accurate energy consumption estimation and it is also significantly scalable.

These findings suggest that, in general, even though flow-level models capture less detail about the reality they model, compared to packet-level models, the loss of detail might not result in significant loss of accuracy.

Bibliography

[1] Barakat, C., Thiran, P., Iannaccone, G., Diot, C., and Owezarski, P. Modeling Internet backbone traffic at the flow level. *IEEE Trans. Signal Processing 51*, 8 (2003), 2111–2124.

[2] Barroso, L. A., Clidaras, J., and Hölzle, U. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*. Synthesis Lectures on Computer Architecture. Morgan & Claypool Publishers, 2013.

[3] Barroso, L. A., and Hölzle, U. The Case for Energy-Proportional Computing. *IEEE Computer 40*, 12 (2007), 33–37.

[4] Beister, F., Dräxler, M., Aelken, J., and Karl, H. Power model design for ICT systems - A generic approach. *Computer Communications 50* (2014), 77–85.

[5] Bolla, R., Bruschi, R., Davoli, F., and Cucchietti, F. Energy Efficiency in the Future Internet: A Survey of Existing Approaches and Trends in Energy-Aware Fixed Network Infrastructures. *IEEE Communications Surveys and Tutorials 13*, 2 (2011), 223–244.

[6] Bolze, R., Cappello, F., Caron, E., Daydé, M. J., Desprez, F., Jeannot, E., Jégou, Y., Lanteri, S., Leduc, J., Melab, N., Mornet, G., Namyst, R., Primet, P., Quétier, B., Richard, O., Talbi, E., and Touche, I. Grid'5000: A Large Scale And Highly Reconfigurable Experimental grid testbed. *IJHPCA 20*, 4 (2006), 481–494.

[7] Carneiro, G., Fortuna, P., and Ricardo, M. Flowmonitor: A network monitoring framework for the Network Simulator 3 (NS-3). In *International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS)* (2009).

[8] Casanova, H., Giersch, A., Legrand, A., Quinson, M., and Suter, F. Versatile, scalable, and accurate simulation of distributed applications and platforms. *J. Parallel Distrib. Comput. 74*, 10 (2014), 2899–2917.

[9] Cisco. Cisco Global Cloud Index: Forecast and Methodology, 2015-2020. White paper, November 2016.

[10] Cisco. The Zettabyte Era: Trends and Analysis. White paper, June 2017.

[11] Claffy, K., Miller, G., and Thompson, K. The nature of the beast: Recent traffic measurements from an Internet backbone. In *Proceedings of INET* (1998), vol. 98, pp. 21–24.

[12] Cornea, B. F., Orgerie, A., and Lefèvre, L. Studying the energy consumption of data transfers in Clouds: The Ecofen approach. In *IEEE International Conference on Cloud Networking, (CloudNet)* (2014), pp. 143–148.

[13] Dayarathna, M., Wen, Y., and Fan, R. Data Center Energy Consumption Modeling: A Survey. *IEEE Communications Surveys and Tutorials 18*, 1 (2016), 732–794.

[14] Fan, X., Weber, W., and Barroso, L. A. Power provisioning for a warehouse-sized computer. In *International Symposium on Computer Architecture (ISCA)* (2007), pp. 13–23.

[15] Fiandrino, C., Kliazovich, D., Bouvry, P., and Zomaya, A. Y. Performance Metrics for Data Center Communication Systems. In *IEEE International Conference on Cloud Computing, (CLOUD)* (2015), pp. 98–105.

[16] Gunaratne, C., Christensen, K., and Suen, S. Ethernet Adaptive Link Rate (ALR): Analysis Of A Buffer Threshold Policy. In *IEEE Global Telecommunications Conference (GLOBECOM)* (2006), pp. 1–6.

[17] Gyarmati, L., and Trinh, T. A. How can architecture help to reduce energy consumption in data center networking? In *International Conference on Energy-Efficient Computing and Networking (e-Energy)* (2010), pp. 183–186.

[18] Heddeghem, W. V., Lambert, S., Lannoo, B., Colle, D., Pickavet, M., and Demeester, P. Trends in worldwide ICT electricity

consumption from 2007 to 2012. *Computer Communications 50* (2014), 64–76.

[19] Jain, R. *The art of computer systems performance analysis - techniques for experimental design, measurement, simulation, and modeling.* Wiley professional computing. Wiley, 1991.

[20] Kliazovich, D., Bouvry, P., and Khan, S. U. GreenCloud: A packet-level simulator of energy-aware cloud computing data centers. *The Journal of Supercomputing 62*, 3 (2012), 1263–1283.

[21] Liu, B., Figueiredo, D. R., Guo, Y., Kurose, J. F., and Towsley, D. F. A Study of Networks Simulation Efficiency: Fluid Simulation vs. Packet-level simulation. In *IEEE International Conference on Computer Communications (INFOCOM)* (2001), pp. 1244–1253.

[22] LLC, M. NS-3 Simulator. `https://www.nsnam.org`. Online; accessed 2017-02-20.

[23] Mahadevan, P., Banerjee, S., and Sharma, P. Energy proportionality of an enterprise network. In *ACM SIGCOMM Workshop on Green Networking* (2010), pp. 53–60.

[24] Mahadevan, P., Sharma, P., Banerjee, S., and Ranganathan, P. A Power Benchmarking Framework for Network Devices. In *8th International IFIP-TC Networking Conference NETWORKING* (2009), pp. 795–808.

[25] Nedevschi, S., Popa, L., Iannaccone, G., Ratnasamy, S., and Wetherall, D. Reducing Network Energy Consumption via Sleeping and Rate-Adaptation. In *USENIX Symposium on Networked Systems Design & Implementation (NSDI)* (2008), pp. 323–336.

[26] Orgerie, A., Lefèvre, L., Lassous, I. G., and López-Pacheco, D. M. ECOFEN: An End-to-end energy Cost mOdel and simulator For Evaluating power consumption in large-scale Networks. In *12th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WOWMOM* (2011), pp. 1–6.

[27] Orgerie, A.-C., Dias de Assunção, M., and Lefèvre, L. A Survey on Techniques for Improving the Energy Efficiency of Large-scale Distributed Systems. *ACM Computing Surveys 46*, 4 (Mar. 2014), 47:1–47:31.

[28] Quinson, M., Rosa, C. D., and Thiery, C. Parallel Simulation of Peer-to-Peer Systems. In *IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)* (2012), pp. 668–675.

[29] Reviriego, P., Christensen, K., Rabanillo, J., and Maestro, J. An Initial Evaluation of Energy Efficient Ethernet. *IEEE Communications Letters 15*, 5 (2011), 578–580.

[30] Ruxton, G. D. The unequal variance t-test is an underused alternative to Student's t-test and the mann–whitney U test. *Behavioral Ecology 17*, 4 (2006), 688–690.

[31] Schulte, E., Davison, D., Dye, T., Dominik, C., et al. A multi-language computing environment for literate programming and reproducible research. *Journal of Statistical Software 46*, 3 (2012), 1–24.

[32] Shehabi, A., Smith, S., Horner, N., Azevedo, I., Brown, R., Koomey, J., Masanet, E., Sartor, D., Herrlin, M., and Lintner, W. United States data center energy usage report. *Lawrence Berkeley National Laboratory, Berkeley, California. LBNL-1005775 Report 4* (2016).

[33] Sivaraman, V., Reviriego, P., Zhao, Z., Sánchez-Macián, A., Vishwanath, A., Maestro, J. A., and Russell, C. An experimental power profile of Energy Efficient Ethernet switches. *Computer Communications 50* (2014), 110–118.

[34] Sivaraman, V., Vishwanath, A., Zhao, Z., and Russell, C. Profiling per-packet and per-byte energy consumption in the NetFPGA Gigabit router. In *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)* (April 2011), pp. 331–336.

[35] Snowdon, D., Ruocco, S., and Heiser, G. Power Management and Dynamic Voltage Scaling: Myths and Facts. In *Workshop on Power Aware Real-time Computing* (2005).

[36] Stanisic, L., Legrand, A., and Danjean, V. An Effective Git And Org-Mode Based Workflow For Reproducible Research. *Operating Systems Review 49*, 1 (2015), 61–70.

[37] Swain, A. R., Hansdah, R. C., and Chouhan, V. K. An Energy Aware Routing Protocol with Sleep Scheduling for Wireless Sensor

Networks. In *IEEE International Conference on Advanced Information Networking and Applications (AINA)* (2010), pp. 933–940.

[38] TEAM, S. SimGrid Simulator. `http://simgrid.gforge.inria.fr/`. Online; accessed 2017-01-20.

[39] VELHO, P., SCHNORR, L. M., CASANOVA, H., AND LEGRAND, A. On the validity of flow-level tcp network models for grid and cloud simulations. *ACM Trans. Model. Comput. Simul. 23*, 4 (2013), 23:1–23:26.

[40] WEINGÄRTNER, E., VOM LEHN, H., AND WEHRLE, K. A Performance Comparison of Recent Network Simulators. In *IEEE International Conference on Communications (ICC)* (2009), pp. 1–5.