Imperial College London
Department of Electrical and Electronic Engineering

Queuing Analysis and Optimization Techniques for Energy Efficiency in Packet Networks

Christina Morfopoulou

June 2013

Supervised by Prof. Erol Gelenbe

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Electrical and Electronic Engineering of Imperial College London
and the Diploma of Imperial College London

Abstract

Energy efficiency in all aspects of human life has become a major concern, due to its significant environmental impact as well as its economic importance. Information and Communication Technology (ICT) plays a dual role in this; not only does it constitute a major consumer itself (estimated 2-10% of the global consumption), but is also expected to enable global energy efficiency through new technologies tightly dependent on networks (smart grid, smart homes, cloud computing etc.). To this purpose, this work studies the problem of energy efficiency in wired networks. As this subject has recently become very active in the research community, there is parallel research towards several research directions. In this work, the problem is being examined from its foundations and a solid analytical approach is presented.

Specifically, a network model based on G-network queuing theory is built, which can incorporate all the important parameters of power consumption together with traditional performance metrics and routing control capability. This generalized model can be applied for any network case to build optimization algorithms and estimate the performance of different policies and network designs. Composite optimization goals functions are proposed, comprising both power consumption and performance metrics. A gradient descent optimization algorithm that can run in $O(N^3)$ time complexity is built thereof. Using power consumption characteristics of current and future equipment, several case studies are presented and the optimization results are evaluated. Moreover, a faster gradient-descent based heuristic and a decentralized algorithm are proposed.

Apart from the routing control analysis, the case of a harsher energy saving solution, namely turning off the networking equipment, is also experimentally explored. Applying a tradeoff study on a laboratory testbed, implementation challenges are identified and conclusions significant for future work are drawn. Finally, a novel admission control mechanism is proposed and experimentally evaluated, which can monitor and manage the power consumption and performance of a network.

Acknowledgments

First and foremost, I would like to thank my supervisor, Professor Erol Gelenbe, for his constructive guidance and invaluable support, both financial and academic. Thanks to his intelligence, knowledge and inspirational ideas, his suggestions and remarks have been ever challenging and influential for work and for life.

I also wish to thank my examiners, Professor Jean-Michel Fourneau and Dr Yiannis Demiris, for reviewing this thesis and for their constructive comments.

Warm regards to the General Arnaoutis Foundation for the scholarship awarded to me and Mrs Nathene Arnaouti for creating an active network of the scholars.

It is a pleasure to acknowledge my colleagues at the Intelligent Systems and Networks group. The support and helpful discussions have been very important for creating a friendly and collaborative working environment. Special thanks to Georgia, Avgoustinos, Niki and Katerina and also Ricardo, Omer, Gokce and Antoine for the willingness to help and our friendly discussions.

I must also thank Alkmini, Ersi, Giota and Myrsini with whom I have shared flats and memories in London. Thanks to all my friends for the happy moments and for being there when needed.

I am deeply thankful to my sister Maria, my mother Theodora and my father Pavlos, to whom this work is dedicated. Without the love and unconditional support of my family, this work would not have been possible. Special thanks to Iason for his love, patience and for standing next to me through this journey.

Contents

List of abbreviations

AC Admission Control

ALR Adaptive Link Rate

CPN Cognitive Packet Network

DVS Dynamic Voltage Scaling

$EAAC$ Energy Aware Admission Control

EEE Energy Efficient Ethernet

$FIFO$ First In First Out

ICT Information and Communication Technology

IP Internet Protocol

ISP Internet Service Providers

$MFDL$ Minimum First Derivative Length

$MPLS$ Multi-Protocol Label Switching

PS Processor Sharing

QoS Quality of Service

RNN Random Neural Network

TCP Transmission Control Protocol

UDP User Datagram Protocol

List of Tables

List of Figures

1 Introduction

1.1 Motivation and objectives

The power consumption of Information and Telecommunication Technology (ICT) has become a major issue in the last few years. According to some studies, ICT covers 2-4% of the global energy consumption, while this figure rises up to 10% when considering only developed countries like the United Kingdom. Surprisingly, the CO_2 emissions of ICT are almost equivalent to the emissions of the aviation industry and are reported to double every 5 years [16]. As shown in Figure 1.1, among the other sectors of ICT, the communication networks consume around 25% of the ICT power consumption. This figure rises further, when one takes into consideration that a significant percentage of the power consumed within datacenters is due to their own communication network.

Moreover, the communication networks play a complex role in the global energy consumption. They can offer an alternative to land and air travel as people start to rely heavily on videoconferencing, or through the massive use of e-commerce, e-learning or even e-work. New energy saving technologies like smart grids, smart homes and cloud computing, depend on communication networks and impose additional traffic in the network. Hence, not only does ICT constitute a major power consumer itself, but it is also responsible to enable and support the reduction of the global energy and carbon emissions.

Apart from the "green" aspect of the reduction of ICT energy consumption, the economic aspect is also very important. Internet service providers, data centers, educational and research organizations, banks, industries as well as customers have huge energy costs for their ICT needs. Even a fraction of energy savings can lead to major economic savings, significant not only for the industries but also to the countries and the end customers themselves. Therefore, it is widely understood that research in communication networks should extend to energy saving solutions.

Today's networks are designed and provisioned to sustain peak time demands and are operated at full capacity at all times. Moreover, a network is usually built with increased redundancy in order to endure potential failures or sudden bursts. Thus, the conventional goal of network architectures and protocols is to provide reliability and to

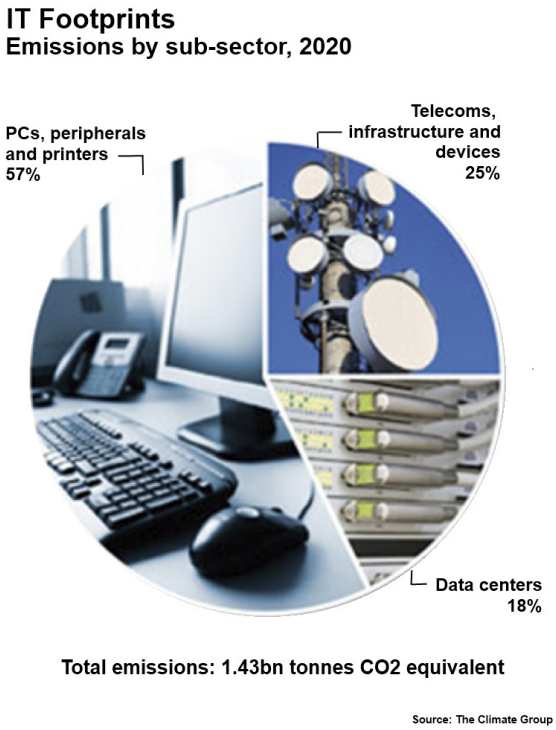


Figure 1.1: Predicted carbon emissions per ICT sector for the year 2020 [78]

attempt to reduce delays and packet losses in a way that guarantees quality of service (QoS) provided. These principles of overprovisioning and operating at maximum capacity at all times are intrinsically opposed to energy efficiency and reveal the opportunity for large energy savings.

The motivation of this work is to examine this novel requirement in network operation and introduce the parameter of energy efficiency in the network service objectives. Since this research subject has only arisen the last couple of years, there is so far no systematic solution. Moreover, as there is parallel research work in other aspects of the problem, such as in the hardware and component level, one needs to be aware of the possible future directions and trends.

In this context, this work studies the problem of energy efficiency in wired communication networks. The parameter that constraints the arbitrary energy efficient tuning of networks is of course the quality of the provided service. Using queuing network analysis, a very useful tool for evaluating the performance of packet networks, a model capturing all the important parameters is built and optimization algorithms are proposed. Moreover, utilizing the PC-based laboratory network located in the Intelligent Systems and Networks group in Imperial College London, experiments are performed to evaluate the proposed mechanisms.

1.2 Contributions

Firstly, the novel research area of energy efficiency in wired networks is reviewed and research directions are identified and categorized. The research problem of network routing control as a means of reducing energy consumption while still remaining aware of QoS considerations is identified. As a first step, we have managed to formalize a theoretical framework which can reflect all the important parameters and can enable the design of optimization algorithms. Thus, the developed model allows us to represent the QoS and power consumption as well as the effect of the control and represents the traffic equations in closed form. To the best of the author's knowledge, this is the first attempt to build a network model on energy efficient routing control under QoS considerations and present an analytical approach to the problem using queuing theory.

Moreover, the usage of composite optimization goals is proposed, comprising both of power consumption and QoS metrics. Using the network model, a gradient descent optimization algorithm is built, which can run in $O(N^3)$ time complexity in order to optimize the composite cost function. Based on power consumption characteristics of current and predicted future networking equipment, several case studies are presented with different optimization goals. The optimization results are evaluated and faster gradient-descent based heuristic algorithms are proposed.

Next, the possibility of harsher energy saving solutions, namely turning off network devices, is experimentally examined. A tradeoff study between delay and energy savings by turning off devices is presented. Finally, a novel energy aware admission control mechanism is proposed and evaluated. This mechanism is responsible to monitor and manage the network in an energy efficient way while still respecting users' QoS needs.

1.3 Thesis outline

The remainder of this thesis is organized as follows. In **Chapter 2**, the background work related to queuing networks and specifically G-networks is reviewed, and a survey on recent work in energy efficiency of communication networks is presented. In **Chapter 3**, the developed network model is presented, which is based on G-networks with triggered customer movement. The optimization problem is built and composite optimization goals are proposed. Moreover, a gradient descent based optimization algorithm is designed. In **Chapter 4**, the proposed model is applied in a real network and the results of the optimization algorithm for different case studies are examined, based on various power consumption characteristics of the nodes and different optimization objectives. A faster heuristic and a distributed approach are also explored and compared. In **Chapter 5**, the case of turning off devices is examined and the tradeoff between the

delay and packet loss and network power consumption are experimentally explored. In **Chapter 6**, the use of an energy aware admission control mechanism is proposed and experimental results in the laboratory testbed are presented. Finally, **Chapter 7** concludes this work and discusses future directions towards energy efficient communication networks.

1.4 Publications

Journal Publications

J 1 Christina Morfopoulou: Network routing control with G-networks. Perform. Eval. 68(4): 320-329 (2011) **(Chapter 3)**

J 2 Erol Gelenbe, Christina Morfopoulou: A Framework for Energy-Aware Routing in Packet Networks. Comput. J. 54(6): 850-859 (2011) **(Chapter 3)**

J 3 Erol Gelenbe, Christina Morfopoulou: Power Savings in Packet Networks via Optimised Routing. MONET 17(1): 152-159 (2012) **(Chapter 4)**

J 4 Georgia Sakellari, Christina Morfopoulou, Erol Gelenbe: Investigating the Tradeoffs between Power Consumption and Quality of Service in a Backbone Network. Future Internet 2013, 5(2), 268-281. **(Chapter 5)**

Conference Papers

C 1 Christina Morfopoulou, Georgia Sakellari, Erol Gelenbe: Energy Aware Admission Control for Wired Networks. ISCIS 2013: *to appear*.

C 2 Georgia Sakellari, Christina Morfopoulou, Toktam Mahmoodi, Erol Gelenbe: Using Energy Criteria to Admit Flows in a Wired Network. ISCIS 2012: 63-72 **(Chapter 6)**

C 3 Erol Gelenbe, Christina Morfopoulou: Gradient Optimization for Network Power Consumption. GreeNets 2011: 125-134 **(Chapter 4)**

C 4 Erol Gelenbe, Christina Morfopoulou: Routing and G-Networks to Optimize Energy and Quality of Service in Packet Networks. ICST E-Energy 2010: 163-173 **(Chapter 3)**

Poster Presentations

P 1 Christina Morfopoulou, Erol Gelenbe: Improving energy efficiency in networks. ACM Sigmetrics/Performance 2012, London June 11-15, 2012 **(Chapter 3,4,6)**

2 Background theory

2.1 Introduction

This chapter attempts to briefly yet comprehensively present the existing and background research, which is related both on the theoretical and the practical aspects of this work. Thus, the contribution of this chapter is twofold: Firstly, the theory of queuing networks and specifically G-networks, on which the network model of Chapter 3 is based, is reviewed. Secondly, the background work on traditional control requirements in communication networks is discussed and a survey on the recent and ongoing research of the relatively novel subject of energy efficiency in wired networks is presented. The chapter is organized as follows: In Section 2.2, a review of the queuing networks and G-networks theory is presented along with the mathematical model and significant extensions. Literature on the traditional problem of applying control in networks in order to improve Quality of Service (QoS) is examined in Section 2.3. Following, in Section 2.4, is a review of recent work on the new requirement of energy efficiency in networks, classified according to different research directions and goals. The chapter concludes in Section 2.5.

2.2 Queuing theory

In this section, the G-network theory is presented, which is used in the network model of Chapter 3. This section starts with the basic theory of queuing systems and is followed by the G-network theory.

2.2.1 Basic queuing theory

A queuing system can be described as customers arriving for service, waiting for service and leaving after being served. The term 'customer' is used in a general sense and differs in each application. In the context of communication networks it usually represents the unit of information or packet. A queuing process is described by a series of symbols such as A/B/X/Y/Z, where A represents the interarrival time distribution and B the probability distribution of the service times. For example, M is Markovian-Poisson for

the arrival process or Exponential for the service time, D is deterministic, G stands for General and GI for General and Independent arrival process. X represents the number of parallel servers, Y the capacity of the waiting queue and Z the queue discipline [56].

Denoting the average rate of customers entering the queuing system as λ and the average rate of serving customers as μ, a measure of traffic congestion for 1-server systems is $\rho = \lambda/\mu$. It turns out that for steady-state results to exist, ρ must be strictly less than one. What is usually interesting in solving queuing models is to find the probability distribution for the total number of customers in the system at time t, $N(t)$, which consists of the number of customers in queue $N_q(t)$ plus those currently receiving service $N_s(t)$. Let $p_n(t) = Pr\{N(t) = n\}$ and $p_n = \lim_{t\to\infty} Pr\{N(t) = n\}$ in the steady state. The expected value of customers in the system is

$$L = E[N] = \sum_{n=0}^{\infty} np_n \tag{2.1}$$

One of the most powerful relationships in queuing theory developed by John Little relates the steady state mean number of customers in the system L to the average waiting time as follows

$$L = \lambda T \tag{2.2}$$

where $T = E[W]$ and W represents the total time a customer spends in the system, including waiting in queue W_q and in service $W_s = 1/\mu$ [56].

The simplest queuing system, the M/M/1 system (with FIFO service) consists of a single server, and an infinite waiting line. The customer interarrival times are i.i.d. and exponentially distributed with some parameter λ and the customer service times are also i.i.d. and exponentially distributed with some parameter μ. In general, the M/M/1 queue leads to easier calculated quantities of interest of the queuing system, thus the sequel will focus on M/M/1 queues.

Performance metrics of M/M/1 queue

M/M/1 queue is the simplest Markovian queue which has only a single server and infinite buffer. The two 'M's refer to the fact that the arrivals are Poisson distributed with a mean rate λ and the interarrival times are exponential with mean $1/\mu$, where $\lambda < \mu \Rightarrow \rho < 1$ to guarantee stability.

One is mainly interested in steady state solutions, where the system after a long running time tends to reach a stable state. The steady state probabilities of the M/M/1

markov chain are given by

$$p_0 = 1 - \rho \tag{2.3}$$

$$p_n = \rho^n(1 - \rho) \tag{2.4}$$

Some significant performance metrics for M/M/1 queue are as follows:

i) **Utilization**

The utilization gives the fraction of time that the server is busy. In the M/M/1 case this is simply the complementary event to the case where the system is empty. The utilization can be seen as the steady state probability that the system is not empty at any time in the steady state, thus

$$\text{Utilization} = 1 - p_0 = \rho \tag{2.5}$$

ii) **Mean number of customers in the system**

The average number of customers in the system is given by

$$L = E[N] = \sum_{n=0}^{\infty} np_n = \frac{\rho}{1 - \rho} \tag{2.6}$$

iii) **Mean response time**

The mean response time T is the average time a customer spends in the system, i.e. the time that a customer spends in the queue as well as the time that customer spends at the server. By applying Little's law

$$T = \frac{L}{\lambda} = \frac{1}{\mu - \lambda} \tag{2.7}$$

iv) **Packet loss**

Together with the above described metrics, in communication networks it is useful to also estimate the probability of packet loss. The M/M/1 queue model assumes an infinite buffer, and thus zero packet loss. However, the probability that the system has K or more customers can be used as an approximation of packet loss. It is implied that this probability equals the probability of a congested queue and if a customer arrives to such a system it will be lost. The probability that the system has K or more customers $P_r[N > K]$, and thus the approximated probability of loss P_{loss} is given by

$$P_{loss} = P_r[N > K] = 1 - Pr[N \leq K] = 1 - \sum_{n=0}^{K} p_n = 1 - (1 - \rho)\frac{1 - \rho^{K+1}}{1 - \rho} = \rho^{K+1} \tag{2.8}$$

Jackson networks

Jackson network refers to a general network system of queues where customers can arrive from the outside at any node i according to a Poisson process with rate $\lambda(i)$ and the service rates are exponential with mean $\mu(i)$. When customers finish service at node i, they move to anther node j with probability $p(i,j)$ or leave the network upon completion of service with probability $p(i,0)$. The total arrival rate of the ith queue is

$$\Lambda(i) = \lambda(i) + \sum_{j=1}^{n} \Lambda(j)p(j,i) \tag{2.9}$$

Defining $\rho = \Lambda(i)/\mu(i)$ Jackson showed that the steady state solution for this system is

$$p_{\mathbf{k}} = p_{k1,k2,..,kn} = (1-\rho_1)\rho_1^{k_1}(1-\rho_2)\rho_2^{k_2}..(1-\rho_n)\rho_n^{k_n} = \prod_{i=1}^{n} p_{k_i} \tag{2.10}$$

This result says that the network acts as if each node was an isolated $M/M/1$ queue with external arrivals equal to the total arrivals expressed in equation 2.9. Even if the internal flows might not really be Poisson, this relationship holds for the network behavior. This very useful feature is described as product form solution for the system.

2.2.2 G-Networks

G-Networks were initially inspired by neural networks. In the random neural network (RNN) model presented in [38], signals can be positive, representing excitatory spikes, or negative, representing inhibitory spikes. A positive spike arriving at a neuron increases its potential by one, while a negative spike arriving at a neuron reduces its potential by one, if its potential is positive, or has no effect, if it is zero. When the potential of a neuron is positive, it can send positive or negative signals at random intervals to other neurons or to outside the network, and in the process it reduces its own potential by one. This model was proved to have product form solution leading to simple analytical expressions for the steady state.

The analogy of this model to a packet queuing network, led to the extension of the RNN model into queuing networks [39] and the concept of G-networks as a unifying model for neural and queuing networks was born. In the queuing network analogy, a node is the equivalent of a neuron. Positive and negative customers traveling in the network correspond to exhibition and inhibition signals which increase or decrease the potential of a neuron, in the same way as a positive and a negative customer increases or decreases the queue length respectively [43].

The simplest instance of G-networks, describes an open network of n queues with

i.i.d. exponential service times $\mu(1), ..\mu(n)$. In this network, two types of customers exist: positive customers with Poisson external arrivals to the ith queue of rate $\lambda^+(i)$, and negative customers with Poisson external arrivals of rate $\lambda^-(i)$ to the ith queue. A positive customer is the normal customer that adds 1 to the queue length and travels through the network until arriving at its destination, where it exits the network. On the other hand, a negative customer arriving at queue i reduces by 1 the length of this queue, if the queue is non empty, and it disappears. In case the queue is empty the negative customer has no effect and just disappears. Thus, negative customers 'cancel' or 'delete' a positive customer and do not receive service themselves. Note that a positive customer leaving queue i after finishing service can join the queue of node j as a positive customer, with probability $p^+(i,j)$, or can head for node j as a negative customer $p^-(i,j)$. It can also depart from the network with probability $d(i)$. Thus, $\sum_j (p^+(i,j) + p^-(i,j)) + d(i) = 1$. The traffic equations are given by

$$\Lambda^+(i) = \lambda^+(i) + \sum_j q_j \mu(j) p^+(j,i) \tag{2.11}$$

$$\Lambda^-(i) = \lambda^-(i) + \sum_j q_j \mu(j) p^-(j,i) \tag{2.12}$$

where

$$q_i = \frac{\Lambda^+(i)}{\mu(i) + \Lambda^-(i)} \tag{2.13}$$

These traffic equations are nonlinear, however it has been proved [39] that there exists a unique solution and the stationary probability distribution has the product form

$$p(k) = \prod_{i=1}^{n} (1 - q_i) q_i^{k_i} \tag{2.14}$$

where $k = (k_1, .., k_n)$ is the vector of queue lengths and $q_i < 1$ for $i = 1, .., n$.

Extensions

Starting from this model with positive and negative customers, several extensions were developed. In [41], the case where the negative customers can carry out batch customer removal is presented. In contrast with the simple case, this extension allows the negative customer to delete one or more positive customers of the queue, thus deleting a batch of customers of random size. If the current queue length is less than the batch size, then queue is emptied.

Many applications of G-networks have been presented since they were introduced. An

early novel application was in the generation of image texture [7]. Other applications include minimum graph covering [48] and other combinatorial optimization problems [43], as well as modeling of defective parts in a flow system [33]. More recent publications present applications to gene regulatory networks [45] and extensions to chemical reaction networks [46]. Several applications inspired by the initial neural network model include resource allocation optimization [55] and the Cognitive Packet Network (CPN) routing algorithm [44, 82].

In another extension [40], the concept of triggered customer movement in G-networks is introduced. In this case, a negative customer arriving at a non empty queue i may reduce the queue length by one with probability $d(i)$, or may trigger the instantaneous passage of a customer to another queue j with probability $q(i,j)$. In this case $\sum_j q(i,j) + d(i) = 1$. The traffic equations in this case become

$$\Lambda^+(i) = \lambda^+(i) + \sum_j q_j \mu(j) p^+(j,i) + \sum_j q_j \Lambda^-(j) q(j,i) \tag{2.15}$$

$$\Lambda^-(i) = \lambda^-(i) + \sum_j q_j \mu(j) p^-(j,i) \tag{2.16}$$

where

$$q_i = \frac{\Lambda^+(i)}{\mu(i) + \Lambda^-(i)} \tag{2.17}$$

This model is very useful for applications where a control mechanism exists that may decide the displacement of tasks from one node to another. The case of multiple classes of both positive customers and signals has also been studied in [32, 49].

The model used in Chapter 3 is a special case of *G-Networks with multiple classes of signals and positive customers*. Thus, the results of the original model [32, 49] are presented in detail in the following section.

2.2.3 G-Networks with multiple classes of signals and positive customers

First consider a network with an arbitrary number n of queues. This model includes a set of positive customer classes $\boldsymbol{U}$ and a set of negative customer (signal) classes $\boldsymbol{S}$. External arrivals streams to the network for both positive customers of some class $k \in \boldsymbol{U}$ and signals of some class $l \in \boldsymbol{S}$ are considered as independent Poisson processes. The external arrival rate of positive customers of class k to queue i are denoted by $\Lambda_{i,k}$ while the external arrival rate of *signals* of class m to queue i are denoted by $\lambda_{i,m}$.

Only positive customers are served and after service they may change service center, class, and nature (become negative), or depart from the system.

When a signal arrives in a non-empty queue it selects a positive customer as its 'target' in the queue, according to the queue discipline at this node. If the queue is empty, the signal disappears. A negative customer of class m arriving at node i tries to trigger the movement of a selected customer of some class k and may succeed with some probability $K_{i,m,k}$, or not succeed with probability $(1 - K_{i,m,k})$. A signal disappears as soon as it tries to trigger the movement of its targeted customer. Note that this signal can either be external or obtained by the transformation of a positive customer after it leaves a queue of the network.

A positive customer of class k leaving queue i after finishing service may

- move to queue j as a positive customer of class l with probability $P^+[i,j][k,l]$
- move to queue j as a negative customer of class m with probability $P^-[i,j][k,m]$
- depart from the network with probability $d[i,k]$

For all i, k

$$\sum_{j=1}^{n}\sum_{l=1}^{U} P^+[i,j][k,l] + \sum_{j=1}^{n}\sum_{m=1}^{S} P^-[i,j][k,m] + d[i,k] = 1 \tag{2.18}$$

All service centers are assumed to have exponential service time distributions and each class of positive customers may have a distinct service rate r_{ik}.

- When the service discipline of the service center is **first-in-first-out (FIFO)** the following constraints should be satisfied:
  - The service rate and the movement triggering rate due to incoming signals should satisfy:

$$r_{ik} + \sum_{m=1}^{S} K_{i,m,k}\lambda_{i,m} = c_i \tag{2.19}$$

  - For classes of signals m such that $\sum_{j=1}^{n}\sum_{l=1}^{U} P^-[j,i][l,m] > 0$ all classes of positive customers a and b should satisfy:

$$K_{i,m,a} = K_{i,m,b} \tag{2.20}$$

  Note that these constraints have the effect of producing a single positive customer class equivalent for service centers with FIFO discipline. As services are expo-

nentially distributed, positive customers of a given class are indistinguishable for movement triggering because of the Markovian property of service time.

- When the service discipline of the service center is **processor sharing (PS)** the probability that one positive customer of the queue selected by the arriving signal is $1/c$, if c is the total number of customers in the queue.

It has been proved that the above described network has a product form solution and the steady state probability that the node i contains one or more positive customers of class k is given by:

$$q_{i,k} = \frac{\Lambda_{i,k} + \Lambda^{+}_{i,k}}{r_{i,k} + \sum_{m=1}^{S} K_{i,m,k}[\lambda_{i,m} + \lambda^{-}_{i,m}]} \tag{2.21}$$

where $\Lambda^{+}_{i,k}$ is the internal arrival rate of positive customers of class k at node i and is given by

$$\begin{aligned} \Lambda^{+}_{i,k} &= \sum_{j=1}^{n}\sum_{l=1}^{U} P^{+}[j,i][l,k] r_{j,l} q_{j,l} + + \sum_{j=1}^{n}\sum_{m=1}^{S}\sum_{s=1}^{U} \lambda_{j,m} K_{j,m,s} q_{j,s} Q[j,i][s,k] \\ &\sum_{j=1}^{n}\sum_{l=1}^{U}\sum_{h=1}^{n}\sum_{m=1}^{S}\sum_{s=1}^{U} r_{j,l} q_{j,l} P^{-}[j,h][l,m] K_{h,m,s} q_{h,s} Q[h,i][s,k] \end{aligned} \tag{2.22}$$

and $\lambda^{-}_{i,m}$ is the internal arrival rate of negative customers of class m at node i and is given by

$$\lambda^{-}_{i,m} = \sum_{j=1}^{n}\sum_{l=1}^{U} r_{j,l} q_{j,l} P^{-}[j,i][l,m] \tag{2.23}$$

2.3 Control in networks

Internet traditionally runs at a 'best effort' manner and the dominating idea of the core of the network is to 'keep it simple'. Moreover, conventional routers and switches lack the ability to provide QoS guarantees. Traditionally, network operator requirements are aimed to guarantee stable and reliable data transfer within time limits and avoid congestion which can cause severe service degradation. So, in order to satisfy specified user QoS there is the need for additional control on top of the networks.

The problem of finding optimal routes in a packet-switched computer network can be formulated as a nonlinear multicommodity flow problem. Early analytical work on the idea of solving this problem in order to reduce a desired cost function, goes back to the Flow Deviation Method [34]. According to this algorithm, given a feasible path flow vector and traffic matrix, part of the flow can be shifted to other paths gradually

based on the minimum first derivative length (MFDL) for each source-destination pair. Several other optimization algorithms based on MFDL applied to optimal routing are described in [11]. Apart from optimal routing problems, there has been a lot of research in flow or congestion control methods which include call blocking, packet discarding and packet scheduling [11].

In practice, the Internet TCP protocol provides a way to reliable data transfer as well as flow and congestion control. TCP maintains a congestion window that controls the number of outstanding unacknowledged data packets in the network and adjusts its sending rate accordingly, maintaining a low congestion in the network. However, as the number of on-line applications like audio/video streaming, video conferencing, IP telephony, medical teleoperation and similar real-time applications is constantly growing, TCP control mechanisms are not enough as a large percentage of traffic is turning to UDP. In order to avoid problems due to unfair sharing of the available resources, fairness and priority need to be guaranteed using additional mechanisms. Thus, a lot of recent research is also focused on approaches to apply control in the networks in order to offer and guarantee the desired QoS to the users.

A way to control traffic congestion and guarantee QoS through the lifetime of a connection is admission control (AC). Admission control mechanisms are responsible to admit or decline the request of a user to enter the network. The new traffic is admitted in the network if QoS constraints of this user and all current users can be respected. Depending on the type of the algorithm, the decision is taken according to a priori set of characteristics and estimations or after probing the network and relying on actual measurements. The first type of algorithms based on a priori specified parameters (parameter-based) [64] can rely on peak bandwidth reservation [57] or statistical allocation [50]. The second category (measurement-based) relies on measurements of actual traffic load using probe traffic of the size of the user traffic [19, 13], or in a percentage of the actual traffic, estimating the actual effect on the users QoS of the new traffic [53, 83].

Other mechanisms preventing congestion rely on network elements such as routers to detect congestion and inform the sources. There is plenty of literature in active queue management algorithms, such as Random Early Detection (RED) and variants or TCP-friendly congestion control schemes as summarized in the surveys [81] and [88]. Multiprotocol label switching (MPLS) traffic engineering can also be used for improving efficiency of bandwidth resources and ensure desirable path for all traffic [85].

The need for reliable networks offering QoS has also lead to the idea of new, smarter networks. A smart or self-aware network is a network able to collect data from distributed points and adaptively take action based on the several QoS objectives of all

users. For example, Cognitive Packet Network (CPN) [44] is a network designed to perform self-improvement based on random neural networks with reinforcement learning, using a special category of smart packets responsible to discover and distribute network information.

The QoS aware networks is a large and active research area. However, in many cases the simpler solution of overprovisioning is used as a means to increase performance. Overprovisioning implies to provision enough capacity to a network in order to be able to serve the peak traffic load estimates. Moreover, the network must be designed to work well under a variety of link and router failures. It is not surprising that most networks today are enormously overprovisioned with very low typical utilizations [93].

2.4 Energy efficiency in networks

As described in the previous section, the lack of QoS guarantees in the communication networks has lead to overprovisioning and redundancy of the deployed hardware. However, together with the need for low congestion and QoS guarantees, lately the need for low energy consumption of the networks has arisen. Even if much work has been presented for energy efficiency in wireless networks ([18, 77, 84]), energy efficiency in wired networks has only recently drawn attention. The problem of energy aware Internet was first addressed by Gupta and Singh in [59]. The authors suggest the idea of energy conservation in Internet systems, proposing possible research directions: putting subcomponents such as line cards into sleep or clock the hardware at a lower rate, change routing so as to aggregate traffic along a few routes only while allowing devices on idle routes to sleep and modify the Internet topology in a way that supports route adaptation and sleeping.

As described in recent surveys [17, 15, 92] several techniques have been recently proposed in order to enable energy efficiency in networks. Although many other classifications would be possible, the research on green ICT can be classified in the following branches

- *Measurements and power consumption models*: There is still little knowledge on how each networking component (hardware, software/applications, network traffic) contributes to overall energy consumption, which is vital for the design of energy-saving systems and architectures. Thus, a lot of work has been devoted on measuring different networking equipment and building models for energy consumption of network equipment

- *Energy efficient hardware*: This branch of research attempts to propose improvements on hardware to increase energy efficiency. Adaptive Link Rate (ALR),

changing operating rate through Dynamic Voltage Scaling (DVS) as well as enabling sleep modes are being examined towards the ideal case of energy proportionality [9].

- *Energy-aware routing and network management*: In this category, research focuses on the potential energy savings by modifying network states and routing policies, depending on different assumptions of network power consumption behavior. In other words, this research area is based on results and trends of the previous two research categories. The focus is on quantifying possible energy savings, under the condition of different hardwares and power models, and proposing algorithms for network energy optimization. This research work of this thesis, falls within this category, thus relevant proposed methods are extensively analyzed in Section 2.4.3.

Each of these categories are further presented in the following sections.

2.4.1 Measurements and power consumption models

A very important aspect of the problem is to first measure and model the power consumed in network components. In [20] the authors measure the power consumption of routers under various configurations and operating conditions. From these measurements it is shown that the base system is the largest consumer, so it is best to minimize the number of chassis at a given point of presence (PoP) and maximize the number of cards per chassis. Then, a generic model for power consumption of router power consumption is built based on a system with different configuration and operating conditions. This model, which reflects the dependence on the power needed for the chassis, the installed line cards and the traffic profile on the device, is applied to a set of network topologies and traffic matrices. The authors also build a design problem where line cards and chassis can be powered on/off, overlaying a multicommodity network-flow problem with flow-balance constraints. This mixed integer program is solved using an offline optimization method that estimates the potential power savings. While traffic was shown to have some impact on the power consumption of a line card, it is measured to be only around 2%. A similar power consumption model is presented in [71], where also the effect of individual ports configured on each line card are taken into consideration. The authors present a power measurement study of a variety of networking devices and such as hubs, edge switches, core switches, routers and wireless access points in both stand-alone mode and within a production data center. They observed that energy consumed by a switch increases linearly with the number of linecards plugged into the switch, the number of active ports on each card and the port line speed. They also suggest that the impact of port utilization on power consumption is under 5% and iden-

tify the challenge for device manufacturers to ensure that networking devices are energy proportional. Moreover, the authors attempt to set a benchmarking suite for measuring and comparing the power consumption of different machines. In [68] the power consumption of computer-based routers is measured focusing on the impact of traffic, with respect to different rates and packet lengths. The results reveal the dependency of power consumption on packet rate and indicate that each device can be characterized by its power consumption versus packet rate profile (power profile).

In [8] a network-based model of the power consumption of the Internet is formulated with data from major equipment vendors focusing on telecommunications network rather than home network equipment. The model of ISP network used comprises of three main sections: the access network, the metro network and the network core. For the estimation of the power consumption of the various routers they use the heat dissipation. To calculate cooling requirements, it is assumed that for every watt of power consumed, another watt of power is required for cooling. The authors result in the observation that the energy bottleneck in the Internet is the routers and not the optical fiber links.

2.4.2 Energy efficient hardware

In this category several works consider hardware changes at the individual PC, switch or router level in order to achieve energy savings [58, 5]. The problem of dynamic link shutdown in Ethernet Links is examined in [60]. The dynamic Ethernet link shutdown algorithm (DELS) designed, makes sleeping decisions for the links depending on traffic arrivals, buffer occupancy and a maximum delay in order to reduce power consumption in Ethernet LAN. The algorithm is tested for different distributions and network traffic loads and the results showed that for loads up to 5% the energy savings are significant while for larger loads the time spent sleeping gradually diminishes enough to be of little significance, while the delay stays within reasonable bounds. Rate adaptation for network power reduction is compared to sleeping in [74]. The first approach is based on rate adaptation of individual links based on their utilization. These performance states try to reduce the energy consumed when actively processing packets by lowering the rate at which work is processed. The second approach puts network interfaces to sleep during idle periods. The sleep states try to reduce energy consumed in the absence of packets. For the realization of this approach small amounts of buffering are introduced and the resulting bursts and delays are moderated. According to the results both sleeping and rate adaptation can lead to significant energy savings, with the tradeoff between them depending primarily on power profile of hardware capabilities and network utilization. Another technique is that of interface proxying, which transfers

the management of traffic to a dedicated entity. This external proxy stores all packets and replies to requests, enabling power-hungry network nodes to sleep for longer periods [4]. A working group of IEEE recently established IEEE 802.3az-2010 standard [63] also known as Energy Efficient Ethernet (EEE) which describes the node mechanisms for enabling sleeping of links, leaving space for research in relevant policies. As Ethernet is a widely adopted networking interface, a fraction of savings in the operation of Ethernet will translate in large overall energy savings. In the legacy Ethernet standards for interfaces of 100M and higher, the circuitry is required to remain powered up whether or not data is being transmitted. The reasoning behind this was that the link must be maintained with full bandwidth signaling so that it is ready to support data transmission at all times. When there is no data they transmit an auxiliary signal called IDLE, used to align transmitters and receivers. This active idle state results in comparable power consumption regardless of whether there is data on the link. Moreover, as the complexity of the interfaces increases for larger data rates, the power consumption also increases significantly.

Energy Efficient Ethernet uses a signaling protocol that allows a transmitter to indicate that there is a gap in the data. This is done by a Low Power Idle (LPI) signal used instead of the IDLE signal and provides a lower consumption energy state that can be employed during periods of low-link utilization. The typical Ethernet traffic has low average link utilization (typically less than 10%) with occasional bursts associated with network activity. EEE takes advantage of the high percentage of idle time on the link and large energy savings can be achieved during long idle periods of time. The standard is defined for the mainstream "BASE-T" interfaces, i.e. 100Base-TX, 1GBase-T and 1GBase-T.

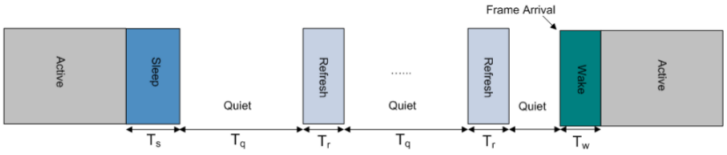


Figure 2.1: Operation of Energy Efficient Ethernet [79]

Figure 2.1 displays the operation of the EEE standard. When a new packet arrives at the device, the Ethernet device transits from a sleep mode to an active mode and the time needed for this transition is defined as T_w (wake-up time). In active mode the device transmits all the packets in queue until the queue is empty and it transits to the sleep mode. The time needed to transit to the sleep mode is T_s. The standard also defines a refresh signal sent every T_r when the link is idle which is important to maintain

compatibility, keep transmitter and receiver aligned and ensure that both partners know that the link is active. For the remaining period T_q the transceiver is in low power mode at which it typically consumes a small fraction (about 10%) of the power consumption when awake.

In [79] the authors evaluated the performance of EEE in terms of energy saving with results showing large energy saving potential. Though, the authors point out that the savings largely depend on the shape of the traffic as the time for putting to sleep and waking up the interface can in some cases be crucial. So, there can be substantial savings when there is a burst of traffic or a lot of back-to-back packets followed by a long idle interval but this is not the case when there are single packet transmissions like in the case of TCP ACKs, in which case the interface might go to sleep and have to wake up for every single packet.

A proposition for further energy savings is to introduce a burst transmission mechanism. In this case when the interface is in sleep mode, incoming packets are accumulated and are released into bursts. The rule for accumulation can be based on a timer or a packet counter or both. In this case, when the first packet arrives in an empty queue a timer/packet counter is started. When the timer has expired or the maximum packet counter is reached the device transits into the active mode and starts transmitting. In [80] and [25] the effect of this mechanism is compared to the EEE standard and the legacy Ethernet and the large energy and economic savings are estimated. Analytical models for the EEE standard and the burst mechanism addition are presented in [72], [62] and [66]. These models are based on Poisson traffic assumption [65] and can be used to evaluate the energy savings.

Other work focuses on aspects outside the network part, e.g. research efforts towards 'greener' applications and end devices. A lot of work also focuses on energy efficiency within data centers. Virtualization is the key idea towards that direction. A typical example of virtualization consists in sharing servers in data centers, thus reducing hardware costs, improving energy management and reducing energy and cooling costs, ultimately reducing data center carbon footprint. Energy savings within data centers and server farms are examined in [61, 70, 73] and in the Cloud Computing environment in [10].

2.4.3 Energy-aware routing and network management

These studies examine the potential energy savings by modifying routing policies, taking into consideration the specific power consumption characteristics of the network components. In the early work of [59] a network-wide as well as a link-layer approach to energy efficiency are examined. According to the network-wide approach (coordinated

sleeping), the decisions for which interfaces to put to sleep can be global and the network protocol will aggregate traffic in the rest of the devices. On the other hand, in the link layer approach (uncoordinated sleeping) an interface would decide to go to sleep based on local data. Moreover, the authors discuss the changes in Internet protocols that should be done in order to support this sleeping.

More recent work deal with the reduction of the power consumption in Backbone Networks using simple heuristics which can solve the corresponding NP-hard problem [24] in a rather network-wide prospect. The problem they consider has as inputs a given physical network topology, the average traffic exchanged by any source-destination pair and the power consumption of each link and node, and the goal is to find a set of routers and links that must be powered on in order to minimize the power consumption. The constraints are the flow conservation and the maximum link utilization. Similar integer programming problem formulation and heuristics are presented in [91]. A case study based on specific backbone networks is discussed in [23], and an estimation of the overall potential energy savings in the Internet is presented in [21].

In [75] the authors focus on the power consumption of the access nodes and propose a technique that adapts their capacity in order to meet the traffic demand and limit the waste of energy. A cooperative approach between the Internet Service providers (ISPs) and the Content Providers (CPs) in order to reduce the total power consumption is examined in [22].

In [54] the reduction of power consumption in wired networks is studied in the presence of users' QoS constraints and experiments with dynamic traffic management in conjunctions with the turning on/off of link drivers and/or routers are discussed. In [36] an autonomic algorithm that utilizes adaptive reinforcement learning in the context of Cognitive Packet Network (CPN) [44, 47] routing protocol is proposed and attempts to minimize the power consumption while meeting the requested end-to-end delay bounds.

An energy-aware online technique is proposed in [86] that aims to reduce energy consumption of the backbone Internet by spreading the load among multiple paths. Energy-Aware Traffic Engineering (EATe) technique is based on the assumption that the hardware is designed to automatically switch to one of the 4 possible operating rates according to its load and that the power consumption of the hardware would follow one of the two curves shown in Figure (2.2). In the first case, there is a great saving in putting the device to sleep so the algorithm tries to shift links to lower energy regions while trying not moving other links to higher energy regions. In the second case rate adaptation is preferable to sleeping, so this approach tries to aggregate traffic to as few links as possible, considering the energy level between the idle state and the bottleneck link utilization state as one large energy level. To a further step of this approach, traffic

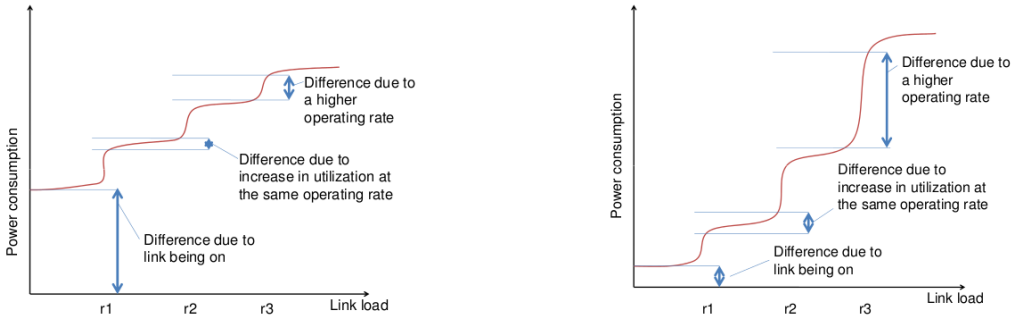


Figure 2.2: Predicted power consumption of a link vs load [86]

is shifted away from routers as a total, so that the entire router can be put to sleep. Another category involves dynamic topology optimization where the alterations of the load are exploited in order to select among all the topologies that ensure connectivity and satisfy the traffic demands the one that has the lowest overall power consumption [67, 87]. In [26], the authors propose an enhancement to the OSPF routing protocol in which a subset of IP routers are selected to be switched off during low traffic periods and in [29] shutting down of cables in bundled links is examined. In [35] a simple energy efficient congestion control extension is proposed and its examined in terms of stability.

ILP formalization of network energy efficiency

As discussed in the previous section, the problem of energy aware routing is usually formalized in the literature as an Integer Linear Programming (ILP) problem [6]. There exist several modifications, so here a generalized formalization will be discussed. Specifically, the problem presented follows the formalizations in [24, 23] and [14], which can informally be expressed as:

- Given
  i)a physical network topology comprising of routers and links with given capacities
  ii) the average amount of traffic exchanged by any source/destination pair
  iii) the power consumption behavior of each router and link

- Find
  i) the routing matrix of the network
  ii)the set of routers and links that must be powered on

- So that
  to minimize the total power consumption of the network

- Subject to the constraints of
  i) flow conservation
  ii) maximum link utilization

First, the network infrastructure is represented as a directed graph $G = (R, L)$, where R is the set of network nodes and L is the set of links. The capacity of the link from node i to node j is represented by c_{ij}. In order to ensure the constraint (ii) a parameter $\alpha \in \{0, 1\}$ is defined which is multiplied with the capacity to give the maximum link utilization that can be tolerated. This model assumes that all nodes and links can be turned off , so the binary variables y_i and x_{ij} are defined which take the value 1 if a node i exists and is powered on and if a link from node i to node j exists and is powered on respectively. r^{sd} represents the average amount of traffic going from source node s to destination node d. The variables $f_{ij}^{sd} \in [0, r^{sd}]$ represent the amount of traffic flowing from s to d through the link i, j while the total amount of traffic on the link i, j is denoted by $f_{i,j}$, where

$$f_{ij} = \sum_{s=1}^{R} \sum_{d=1}^{R} f_{ij}^{sd} \quad \forall\ i, j \tag{2.24}$$

Thus the ILP formalization of the above described problem is

Minimize:

$$P_N = \sum_{i=1}^{R} y_i P_R(i) + \sum_{i=1}^{R} \sum_{j=1}^{R} x_{ij} P_L(i, j) \tag{2.25}$$

Subject to:

$$\sum_{j=1}^{R} f_{ij}^{sd} - \sum_{j=1}^{R} f_{ji}^{sd} = \begin{cases} r^{sd}, & \forall\ (s, d), i = s \\ -r^{sd}, & \forall\ (s, d), i = d \\ 0, & \forall\ (s, d), i \neq s, d \end{cases} \tag{2.26}$$

$$f_{ij} \leq \alpha c_{ij} x_{ij} \quad \forall\ i, j \tag{2.27}$$

$$\sum_{j=1}^{R} x_{ij} + \sum_{j=1}^{R} x_{ji} \leq M y_i \quad \forall\ i \tag{2.28}$$

P_N denotes the total network power consumption and comprises of the power consumption of the routers P_R and the power consumed by links P_L as shown in equation 2.25. Note that the power consumption of nodes and links in this equation is multiplied

by the binary variables y_i and x_{ij}. The constraint 2.26 ensures the flow conservation while the constraint 2.27 forces the total traffic on an active link to be smaller than the maximum link utilization. The constraint of 2.27 makes sure that a node can be turned off only if all connected links are off, where M is a big number $M \geq 2R$.

Discussion

This ILP formulation is known to be NP-hard, so exact methods can only be used to solve trivial cases. Thus, in the existing literature the authors proceed to propose heuristics based on assumptions and specific cases. The heuristics attempt to turn off as many links and/or nodes as possible, assuming that this is the critical factor for energy efficiency. However, the possibility of turning off networking equipment is not feasible with existing equipment. In Chapter 5 it is experimentally shown that there is a large associated tradeoff with quality of service when trying to turn off devices. Moreover, these studies ignore the implementation obstacles when one has to deal with a network with devices that turn on and off. For example, a mechanism should be responsible to inform those machines to wake up when they are needed to route traffic. In addition, turning on devices entails energy and delay costs, until the machine is ON and fully operational. In Chapter 6 an admission control mechanism is proposed, which can manage a network with on/off capabilities. In fact, it is experimentally observed that a solution which tries to turn off as many devices as possible is not the most energy efficient over time.

Thus, an aggressive approach which would shut down network equipment is not only infeasible with the current equipment, but could also lead to major degradation of the performance. In contrast, in Chapter 3 the application of routing control on top of the networks is proposed, in order to change routing decisions in a more energy efficient way. The presented model is based on queuing theory, instead of using ILP, as previous works. This provides us with a very powerful model in terms of network performance. Instead of dealing with a maximum capacity constraint as the formulation presented above, using queuing analysis well known performance metrics as queuing delay and packet loss can be used, as presented in Section 2.2.1. Adding multiple classes in the model provides additional flexibility, as the end-to-end delay of each flow can be tracked and the optimization goal can be adapted to include these QoS metrics. The selection of G-network model with triggered customer movement, presented in Section 2.2.3, grants additional flexibility as the control act and its effect on the cost function can be incorporated in the model.

Moreover, a gradient descent optimization algorithm is proposed which runs in $O(N^3)$ time complexity. Instead of having to built different model and heuristics based on

the specific optimization goal, as in previous works, this algorithm can be used to optimize power consumption or a combined optimization goal comprising both of power consumption and delay, as shown in Chapter 4.

2.5 Conclusions

In the first part of this chapter, the basic queuing theory is presented together with the G-network theory which is used in Chapter 3 to build a model for energy efficient routing control. Then, the traditional network objectives for applying control in networks are presented. This is followed by an extended survey of recent work on network energy efficiency.

All this recent work on green networking research reveals the shifting objectives of network design and operation from the conventional ones, presented in Section 2.3, towards energy efficiency, as well as the importance of this new objective. As there is still no systematic approach for achieving energy efficiency in networks, in this work an attempt to examine the problem from its foundations is made. Using the tools presented in Section 2.2, first an analytical model based on G-networks is presented, which can incorporate all the important parameters, including QoS metrics based on queuing theory. The proposed queuing model provides a very powerful tool in order to examine the network performance together with power consumption, in contrast to ILP formulations of previous works.

The proposed model can distinguish between different users with different source-destination pairs, traffic inputs and QoS needs. Also, thanks to the product form solution it can estimate the traffic at each node in steady state and thus estimate power consumption at each node as well as queue length and QoS metrics. In contrast to the mixed integer programming problem formulations of the existing works which result in NP-hard problems, this model can be used to optimize a cost function in $O(N^3)$ time complexity using a gradient descent algorithm. Finally, the effect of the control traffic and control decisions is included.

Moreover, the majority of the presented approaches, present specific cases of network routing policies and hardware capabilities, i.e. sleep modes, adaptive rates etc, which could become obsolete in case future hardware design follows different direction. In contrast, in this work the problem of a given packet network, with given hardware capabilities and power consumption characteristics is first presented and a generalized model for energy efficient routing control is built. Having in hand the network model, a gradient descent optimization algorithm is used, in order to explore the potential savings from routing control. Several specific cases of power consumption characteristics and

objectives are examined. However, the model can be applied for any network case to build optimization algorithms and estimate the performance of different policies and network designs.

Apart from the routing control analysis, the possibility of a harsher energy saving solution, namely turning off network devices, is experimentally examined. In contrast to using estimations and simulations as in other existing works, a laboratory testbed is used and a real case scenario is examined, where devices can be turned off and power consumption and delay can be measured online. This enables the observation of the significant tradeoff between power savings and QoS in this case. Realizing the limitations and obstacles of such a scheme of turning off devices that had been neglected so far, a novel energy aware admission control mechanism is proposed, which is able to monitor and manage the power consumption and performance of the network.

3 Network model and optimization for energy efficient routing control

3.1 Introduction

As described in Chapter 2, possible research directions towards network energy efficiency span from changing the routing decisions through the network to turning down or turning off the equipment. Even if the power consumption measurements of current networking devices indicate that the decision to shut down machines may appear as an effective solution, an aggressive approach which would shut down all unused equipment is not only infeasible with the current equipment, but could also lead to major degradation of the performance, as shown in Chapter 5. Moreover, the additional energy needed for rebooting and reconnecting the machines in the network could raise unpredictable increase in the power consumption which could cancel out the positive effects. Thus, it is clear that the network changes should be carefully designed and examined before the decisions for the appropriate techniques are taken.

On the other hand, changing routing decisions in an energy efficient way could be deployed in every network and can exploit the relationship between power consumption and carried traffic in each node. It is also in agreement with hardware designers who focus on improving the performance of routers such that the power consumption is significantly reduced when received traffic is small([89]), so the effect of carried traffic will be even more sizable in the future devices.

In order to be able to achieve a network-wide solution towards energy efficiency, firstly one should have a good understanding of the consequences that a change can cause in the whole network. Thus, the first step in this research is to build a network model which can reflect the changes that will be made, the power consumption and the Quality of Service (QoS) metrics and could be in hand for examining the possible optimization techniques.

In the sequel, a network model is presented, which offers a theoretical framework to design algorithms that apply control in computer networks by re-routing traffic so as to optimize a given cost function. This model, which is a special case of G-Networks with triggered customer movement, (see Section 2.2.3) , does not just consider the effect of

the control via route modifications for the user traffic, but also allows one to represent the control traffic itself as a flow of packets which transform themselves into triggers when they reach the target node of the control. This enables one to analyze not just the effect of the control, but also the overhead and delays that the control packets themselves introduce, as well as the effect of these physically important aspects in the details of the control algorithm and their impact on the resulting cost function that one is minimizing.

The chapter is organized as follows: In Section 3.2, the queuing network model is presented which is a special case of G-networks with triggered customer movement and is specifically built to include the important parameters of the energy efficient routing control optimization. Performance metrics of the network are presented in Section 3.3, including power consumption as well as quality of service metrics, as presented in Section 2.2.1. Following, in Section 3.4, the optimization goals are discussed and a gradient descent algorithm is built to optimize energy efficient routing control. Finally, the chapter concludes in Section 3.5.

3.2 Network model description

3.2.1 Main points

In this section, a queuing network model is presented including all the parameters which are important, when one seeks to observe and optimize network energy efficiency under Quality of Service (QoS) considerations. Thus, using queuing theory, the performance of the network, in terms of QoS, is also taken into account in this work, in contrast to previously presented literature (Section 2.4). Since the QoS is the primary objective in today's networks, a clear view of the consequences on QoS should be available in any proposal for energy efficiency.

More specifically, the model described in the sequel, is a special case of G-networks with multiple classes and triggered customer movement (see Section 2.2.3). The multiple classes correspond to different source-destination pairs, with different input traffic rates, different QoS requirements and different routes. The triggers enable the depiction of the control act, when a decision to change routing is forced, for a given class and at a specific node. Modifying the original model, the control traffic here is also considered as physical traffic which travels through the network and recieves service. This way, the effect of the control traffic can be observed and can also be included in the cost function. The effect of control traffic does not only correspond to the possible increase in congestion due to the control traffic, but also to the additional power consumption that this traffic and the effect of control can induce. Moreover, since it has been observed

in literature that links and routers can have a distinct effect on power consumption, the model is extended by separating routers from links, and defining both as nodes of the queuing system. By doing so, the impact on energy consumption and QoS can be modeled separately for routers and links. Finally, as in modern router hardware connections typically are handled separately by threads of executions, in the modified model a separate queue is assumed in routers for each class of user traffic. This enables the modeling of router queues as first-come-first-served (FIFO) and the modeling of control traffic classes which are specifically targeted on certain user traffic classes. In links, all packets are handled in first-come-first-served order.

3.2.2 Model description

Consider a network with N queues denoted $\mathbf{N} = \{1, \ ... \ , N\}$ which is carrying a set of *user traffic classes* $\mathbf{U}$. Let $N+1$ be the "inexistent node" or the outside of the network. Thus a packet that reaches node $N+1$ has simply exited the network or has been lost. Here the term node is not restricted to the store and forward nodes of a network: a subset $\mathbf{R}$ of these nodes will be the usual store and forward nodes or routers, while the remaining set of nodes $\mathbf{L}$ will represent links which connect store and forward nodes, i.e. $\mathbf{N} = \mathbf{R} \cup \mathbf{L}$. Thus, to travel from some store and forward router node to another such node, a packet will transit through the "other type of node" i.e. a link. This separation of the N nodes into the set of routers and the set of links has two advantages:

- It allows us to model separately the impact of routers and links (e.g. delay and loss) on QoS and on the *energy consumption*, and
- Secondly it allows us to explicitly represent packet re-routing as the modification of a path that goes (say) from router r via output link l, to a new path from router r via some other output to l'. Thus, re-routing control can then be viewed as taking place by an action on a packet in r that changes the next link that the packet must enter.

Links have a single predecessor node (which is a router) and a single successor node (which is also a router), and routers typically have multiple predecessor and successor nodes that are links. In our notation the two directions of a physical link are viewed as two distinct links, but they may be coupled because they may share the same power supply and some of the same hardware. Since router hardware will typically use "multicores", connections that are being concurrently processed by a router will often be handled by separate threads of execution, and packets within the same connection may be handled in first-come-first-served order, while those of distinct connections may be processed in parallel.

User traffic

Each user traffic class is denoted by $k = U(s, d, \sigma)$ where (s, d) is a given source and destination router pair and σ is a QoS requirement of this traffic class from the network. The most common QoS metrics are be presented in Section 3.3.

Let $\lambda(r, k)$ denote the *external* arrival rate of packets of user class k to router r so that obviously $\lambda(r, k) = 0$ if $k = U(s, d, q)$ and $r \neq s$, while $\Lambda_R(r, k)$ and $\Lambda_L(l, k)$ are the total arrival rates of user traffic of class k at router r and link l respectively. Each class of users will, at any given instant of time, have a path to its destination so that the probability that a packet of user class k travels from some node i to some node j is denoted by $P(i, k, j)$; if all packets of a given user class only travel over a single path at a time, then this quantity will be either equal to 1 or 0. Note also that if i is a router, then j is a link, while if i is a link, then j must be a router. If user traffic is assumed to travel on a single path, then obviously $\Lambda_R(s, k) = \lambda(s, k)$ when s is source router, and $\Lambda_R(r, k) = \lambda(s, k)$ when r is a router that lies on the path of this traffic class and $\Lambda_L(l, k) = \lambda(s, k)$ for a link l lying on this path respectively.

Control traffic

In addition to user traffic, we also have *control traffic classes* where each of these classes is in charge of selecting the next hop that a packet of a given user traffic class will take at some router. In other words, packets that belong to a control class are in charge of telling a given router where the packets of a given user traffic class must go. Since packets are re-routed by changing the selection of the outgoing link in a router, this selection will be signified by these control packets. Thus control traffic classes are denoted (i, k) where i is a router and k is a *user* traffic class since a control class acts on a specific user class at some given router. However, control traffic can travel over multiple hops just like any traffic class until it reaches the node where it is supposed to take action. The actual number of control traffic classes will be small because at a given instant of time a class of user traffic will only transit through a small number of network nodes. We denote by $\lambda^-(r, (i, k))$ the *external* arrival rate of control traffic class (i, k) to router r, and such arrivals can only occur at routers. The *total arrival rate* of control traffic class (i, k) at router r or a link l, which will be computed below, is denoted by $\Lambda_R^-(r, (i, k))$ and $\Lambda_L^-(l, (i, k))$. This allows us to represent control traffic that may originate at different routers and act at the router where they originate, or they may act at other routers. Thus, control packet of class (i, k) may move from some router r to some link l with probability $p((i, k), r, l)$ provided $i \neq r$ so that this particular control packet cannot act at the router r to redirect traffic. Similarly, the control packet moves from some link l to some router r, with probability $p((i, k), l, r)$. Note that this probability would be

just 0 or 1 whenever there are no losses, simply because the output of a link is only connected to a single router. It is also assumed that once a control packet has acted at some router, then it is destroyed; in other words, each control packet can only act *once* on a single user packet at some specific router. Furthermore, if a control packet of class (i, k) arrives at router i when that router does not contain *any* user packets of class k, then the control packet is again destroyed.

The *control function* exercised by the control packet will be represented by $Q(i, k, j)$ which is the probability that a user packet of class k at router i is *directed* by the corresponding control packet of type (i, k) to the link j. Note that $Q(i, k, j)$ is only defined at a router i for the control class (i, k): in other words it need not be specified how the control policy acts at a node where this particular control class is not empowered to act. Also, the control packets can only act at routers, so that $Q(i, k, j) = 0$ if $i \in \mathbf{L}$.

The control traffic classes that we introduce may be seen in two ways:

- As physical flows of signaling information in the form of specific packets, that are sent out from certain decision nodes, to routers where it may be necessary to re-route traffic, or
- As a virtual and mathematical representation of re-routing decisions. Thus the arrival rate of control packets at some router may be used to represent the rate at which control decisions are made at this router.

Constraints

The routing probabilities satisfy the following constraints, for $k \in \mathbf{U}$, if $r \in \mathbf{R}$:

$$\sum_{l \in \mathbf{L}} P(r, k, l) + P(r, k, N{+}1) = 1, \tag{3.1}$$

$$\sum_{l \in \mathbf{L}} Q(r, k, l) = 1, \tag{3.2}$$

$$\sum_{l \in \mathbf{L}} p((i, k), r, l) + p((i, k), r, N{+}1) = 1, r \neq i \in \mathbf{R} \tag{3.3}$$

while if $l \in \mathbf{L}$:

$$\sum_{r \in \mathbf{R}} P(l, k, r) + P(l, k, N{+}1) = 1, \tag{3.4}$$

$$\sum_{r \in \mathbf{R}} p((i, k), l, r) + p((i, k), l, N{+}1){=}1, \ i \in \mathbf{R} \tag{3.5}$$

and note that there are no control packets classes of the form (l, k) where $l \in \mathbf{L}$. The flow constraints can be expressed by

$$\sum_{r \in \mathbf{R}} \lambda(r,k) = \sum_{r \in \mathbf{R}} \Lambda(r,k) P(r,k,N+1) \tag{3.6}$$

which implies that all incoming traffic of class k should exit the network at the destination node d, where for $r = d \;\; P(r,k,N+1) = 1$.

It is also assumed that all user or control packets that travel *through* a node i have the same service rate μ_i at that node, but control packets act instantaneously when they act as control packets, rather than when they are simply transiting through a node when they experience the usual queuing phenomenon. Finally, all packets are processed in first-come-first-served mode both in the nodes and links, so that there is no priority difference for transiting purposes between the user and control packets. On the other hand, when a control packet arrives at a node where it is supposed to take the control action, it does this instantaneously on its "target" packet class, selecting the target packet (i.e. of the appropriate class) which is first in queue within its own class. If the router where this is supposed to happen contains no packets of the target class, then the corresponding packet is destroyed. The model that we use is a special case of G-Networks [43, 41, 31] with triggered customer movement, where the control classes embody the triggers of the mathematical model [40], including multiple classes [32, 49], as presented in Section 2.2.3. Thus, the corresponding theory can be applied, and in particular the steady-state probability that the queue of node i contains at least one user packet of class k can be obtained as follows.

3.2.3 Flow equations

Since all the parameters have been described, the equations that govern our model can now be detailed based on the G-networks model as referred to above. The steady-state probability that a router or link contains at least one packet of user class k is given by:

$$q(r,k) = \frac{\Lambda_R(r,k)}{\mu_r + \Lambda_R^-(r,(r,k))}, \;\; if \;\; r \in \mathbf{R} \tag{3.7}$$

where it is assumed that each of the user classes are handled by separate queues in routers; recall that we assume that these different user class queues within the same router are processed concurrently with the help of a multicore architecture so that the queues can indeed be viewed as separate entities running in parallel. On the other hand, all packets within a link are handled in first-come-first-served order. The steady-state

Table 3.1: Table of main variables used

User Traffic	
Variable	explanation
k	user traffic class $k \in \mathbf{U}$
$\lambda(r,k)$	external arrivals of class k to router r
$\Lambda_R(r,k)$	total arrivals of class k to router r
$\Lambda_L(l,k)$	total arrivals of class k to link l
$q_R(r,k)$	steady-state prob. for router r and class k
$q_L(l,k)$	steady-state prob. for link l and class k
$B(l)$	steady-state prob. that link l is busy
Control Traffic	
Variable	explanation
(i,k)	control traffic class where $i \in \mathbf{R}, k \in \mathbf{U}$
$\lambda^-(r,(i,k))$	external arrivals of class (i,k) to router r
$\Lambda_R^-(r,(i,k))$	total arrivals of class (i,k) to router r
$\Lambda_L^-(l(i,k))$	total arrivals of class (i,k) to link l
$c_R(r,(i,k))$	steady-state prob. for router r and class (i,k)
$c_L(l(i,k))$	steady-state prob. for link l and class (i,k)

probability that the link l contains at least one packet of user class k is given by:

$$q(l,k) = \frac{\Lambda_L(l,k)}{\mu_l},\ if\ l \in \mathbf{L} \tag{3.8}$$

The total arrival rates of user packets of class k to routers $r \in \mathbf{R}$ and links $l \in \mathbf{L}$ are

$$\begin{aligned} \Lambda_R(r,k) &= \lambda(r,k) + \sum_{l \in \mathbf{L}} q(l,k) P(l,k,r) \mu_l \\ &= \lambda(r,k) + \sum_{l \in \mathbf{L}} P(l,k,r) \Lambda_L(l,k) \qquad (3.9) \\ \Lambda_L(l,k) &= \sum_{r \in \mathbf{R}} [P(r,k,l) q(r,k) \mu_r \\ &\quad + \Lambda_R^-(r,(r,k)) q(r,k) Q(r,k,l)] \qquad (3.10) \end{aligned}$$

The arrival rate to router $r \in \mathbf{R}$ or link $l \in \mathbf{L}$ of control traffic class (i,k) is given by

$$\Lambda_R^-(r,(i,k) = \lambda^-(r,(i,k)) + \sum_{l \in \mathbf{L}} p((i,k),l,r) c_L(l,(i,k)) \mu_l \tag{3.11}$$

$$\Lambda_L^-(l,(i,k)) = \sum_{r \in \mathbf{R}} p((i,k),r,l) c_R(r,(i,k)) \mu_r, i \neq r \tag{3.12}$$

where the steady-state probability that router r contains at least one control packet of class (i,k) for $r \in \mathbf{R}$ and $r \neq i$ is:

$$c_R(r,(i,k)) = \frac{\lambda^-(r,(i,k)) + \sum_{l\in\mathbf{L}} p((i,k),l,r)c_L(l,(i,k))\mu_l}{\mu_r} \tag{3.13}$$

and the steady-state probability that link $l \in \mathbf{L}$ contains at least one control packet of class (i,k) is:

$$c_L(l,(i,k)) = \frac{\sum_{r\in\mathbf{R}} p((i,k),r,l)c_R(r,(i,k))\mu_r}{\mu_l} \tag{3.14}$$

Finally, let us point out that the steady-state probability that link l is busy is simply:

$$B(l) = \sum_{k\in\mathbf{U}} [q(l,k) + \sum_{i\in\mathbf{R}} c_L(l,(i,k))] \tag{3.15}$$

3.3 Performance metrics

The users' QoS needs are typically expressed in terms of packet delay, probability of loss, jitter, and similar metrics that depend on the congestion at routers and links which in turn depend on the probabilities that the nodes or links are busy. Thus, $q(r,k)$, $B(l)$, $c_R(r,(i,k))$ and $c_L(l,(i,k))$ are the key quantities that can be used to obtain other QoS metrics.

3.3.1 Average delay through routers and links

From the previous discussion, the average queue lengths at routers and links can easily be derived (see Section 2.2.1). Unbounded queue lengths are assumed throughout the discussion, and starting with links, the average number of packets at link l is given by:

$$N_q(l) = \frac{B(l)}{1-B(l)}, l \in \mathbf{L} \tag{3.16}$$

The average queue lengths at the routers will be consistent with the preceding discussion which assumes that each category of packets, whether of user type or of control type, will be handled in a separate queue at each router r. So, the average number of packets at router r, for user traffic and control traffic respectively, are:

$$N_q(r,k) = \frac{q(r,k)}{1-q(r,k)} \tag{3.17}$$

$$N_q(r,(i,k)) = \frac{c_R(r,(i,k))}{1-c_R(r,(i,k))}, i \neq r \tag{3.18}$$

The probabilities that a user packet of class k, or a control packet of class (i,k), enters router r or link l are defined:

$$\pi(r,k) = \frac{\Lambda_R(r,k)}{\lambda^+(k)}, r \in \mathbf{R} \tag{3.19}$$

$$\pi(l,k) = \frac{\Lambda_L(l,k)}{\lambda^+(k)}, l \in \mathbf{L} \tag{3.20}$$

$$\pi(r,(i,k)) = \frac{\Lambda_R^-(r,(i,k))}{\lambda^-(i,k)}, r,i \in \mathbf{R}, i \neq r \tag{3.21}$$

$$\pi(l,(i,k)) = \frac{\Lambda_L^-(l,(i,k))}{\lambda^-(i,k)}, l \in \mathbf{L}, i \in \mathbf{R} \tag{3.22}$$

where:

$$\lambda^+(k) = \sum_{r \in \mathbf{R}} \lambda(r,k) = \lambda(s,k) \tag{3.23}$$

is the total user traffic of class k entering the network, s being the source router of class k, and

$$\lambda^-(i,k) = \sum_{r \in \mathbf{R}} \lambda^-(r,(i,k)) \tag{3.24}$$

is the total control traffic of class (i,k).

Let $\Lambda^+(i)$ be the total traffic of user packets entering node i, while $\Lambda^-(i)$ is the total control transiting that node, and they are given by:

$$\Lambda^+(i) = \sum_{k \in \mathbf{U}} \Lambda_R(i,k),\ if\ i \in \mathbf{R} \tag{3.25}$$

$$= \sum_{k \in \mathbf{U}} \Lambda_L(i,k),\ if\ i \in \mathbf{L} \tag{3.26}$$

$$\Lambda^-(i) = \sum_{j \in \mathbf{R}, j \neq i} \sum_{k \in \mathbf{U}} \Lambda_R^-(i,(j,k)),\ if\ i \in \mathbf{R} \tag{3.27}$$

$$= \sum_{j \in \mathbf{R}} \sum_{k \in \mathbf{U}} \Lambda_L^-(i,(j,k)),\ if\ i \in \mathbf{L} \tag{3.28}$$

and the total traffic of packets $\Lambda(i)$ transiting through a node i will be :

$$\Lambda(i) = \Lambda^+(i) + \Lambda^-(i) \tag{3.29}$$

Using Little's formula [51, 52] the total average delay through the network for a user

packet of class k is:

$$T(k) = \sum_{l \in \mathbf{L}} \pi(l,k) \frac{N_q(l)}{\Lambda(l)} + \sum_{r \in \mathbf{R}} \pi(r,k) \frac{N_q(r,k)}{\Lambda_R(r,k)} \tag{3.30}$$

If the propagation delay of the link $d(l)$ is not negligible, this formula should be adjusted to

$$T(k) = \sum_{l \in \mathbf{L}} \pi(l,k) \left(\frac{N_q(l)}{\Lambda(l)} + d(l) \right) + \sum_{r \in \mathbf{R}} \pi(r,k) \frac{N_q(r,k)}{\Lambda_R(r,k)} \tag{3.31}$$

The total average delay experienced by a control packet of class (i,k) is:

$$\begin{aligned} T^-(i,k) &= \sum_{l \in \mathbf{L}} \pi(l,(i,k)) \left(\frac{N_q(l)}{\Lambda(l)} + d(l) \right) \\ &+ \sum_{r \in \mathbf{R}, r \neq i} \pi(r,(i,k)) \frac{N_q(r,(i,k))}{\Lambda_R^-(r,(i,k))} \end{aligned} \tag{3.32}$$

The average overall network user packet delay can be expressed by:

$$\overline{T_N} = \sum_k \frac{\lambda^+(k)}{\Lambda_T^+} T(k) \tag{3.33}$$

where $\lambda^+(k)$ is given by equation (3.23), $T(k)$ is given by (3.31) and $\Lambda_T^+ = \sum_k \lambda^+(k)$. In this expression each flow is weighted by its percentage of traffic volume in the network so that the average in expressed in the per-packet level.

3.3.2 Packet loss

As discussed in Section 2.2.1, the unbounded queue assumption can be used to estimate the probability that the node has K or more packets, which in turn can be used to approximate the probability that a packet is lost. Thus, the probability of packet loss for a link l can be approximated by

$$S_L(l) = B(l)^{K+1} \tag{3.34}$$

where K can be adjusted according to the maximum queue length, or according to the maximum utilization above which there is a significant probability of loss. For example, if $K = 10$ is assumed, the probability of loss becomes significant for utilization values larger than 0.7. Similarly, the packet loss can be defined for the routers, though usually the limited capacity of the links is the crucial parameter for packet loss.

The end-to-end packet loss will be the product of the losses on its path, thus

$$S(k) = \prod_{l \in \mathbf{L}} S_L(l)\ 1[\Lambda_L(l,k) > 0] \tag{3.35}$$

where the expression $1[x]$ is 1 if x is true and 0 otherwise.

3.3.3 The power consumption model

As discussed in Chapter 2, the energy consumption characteristics of real network nodes are fairly known and are expected to change in the future due to extensive research towards more energy saving network equipment. Previous work ([20],[71]) have indicated that the power consumption of a network router consists of a fixed part that depends on the particular hardware type, another part determined by the line cards and ports that are configured on the router and a small part that depends on the traffic carried by the device. A recent survey [15] identifies power profiles depending on the utilization, spanning from the ideal case of energy proportional to the extreme case of energy agnostic. The measurements reported in [68], for two distinct PC-based routers and different fixed packet lengths, are shown in Figure 3.1.

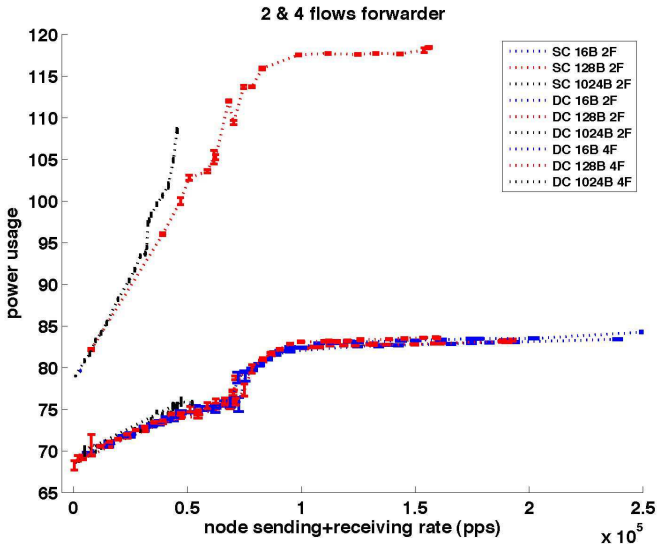


Figure 3.1: Power consumption measurements of 2 distinct PC-based routers [68]. These results showcase the variations between different machines and indicate that each device can be characterized by its power consumption versus packet rate profile, which is defined as the power profile.

These results show that for older single core technology (upper curve) the power consumption increases monotonically with the rate at which *packets* are processed in the router. The lower curve show similar results for a more recent multicore technology with

much lower power consumption, and have a distinct step upwards when an additional core kicks in as packet rate increases. In all cases *packet length* has little effect on the power consumption, as power consumption results with different packet lengths but same packet rates fall on the same curve. These results showcase the variations between different machines and operational states and indicate that each device can be characterized by its power consumption versus packet rate profile, which is defined as the power profile of a machine. Based on this background work, a generalized model is built which describes the devices' power consumption as a function of their load. More specifically, for a router, the amount needed to keep it on, the power needed for processing individual packets and the operating cost for applying the route changes, are included. As a result, the following power consumption formula for a router will be used:

$$P_i = \alpha_i + g_R(\Lambda(i)) + \gamma_i \sum_{k \in \mathbf{U}} \Lambda^-(i,(i,k)), i \in \mathbf{R} \tag{3.36}$$

where α_i corresponds to the router's static power consumption, $g_R(.)$ is an increasing function of the packet processing rate, as in Figure 3.1, while γ_i is a proportionality constant related to the power cost of processing being carried out in the router for re-routing control.

Moreover, the separation of the nodes into the set of routers $\mathbf{R}$, and the set of links $\mathbf{L}$ allows us to model and handle separately their power consumption. The link power consumption depends on the traffic rate in bytes or bits per second and includes the needs for operating the interface within the router, and for transmitting data on the line. Additionally, one could include the power consumed for propagating or "repeating" data on the line, but this may be negligible [8]. The proposed link power model is then:

$$P_i = \beta_i + g_L(\Lambda(i)), i \in \mathbf{L} \tag{3.37}$$

where β_i corresponds to the static power consumption, and $g_L(.)$ is an increasing function. Thus, (3.36) and (3.37) can be used as a generalized model for network device power consumption, since they incorporate static and dynamic effects, they separate link power consumption and also include the possible power cost of changing routing decisions. The total network power consumption will be then given by the summation of power consumption of all network equipment, i.e.:

$$P_N = \sum_{i \in \mathbf{N}} P_i \tag{3.38}$$

3.3.4 Other performance metrics

Another performance metric that could be of interest is the path availability of a specific class k which can be expressed as the minimum availability of all links on the path

$$P_a(k) = \min_{l \in \mathbf{L}} (1 - B(l)\pi(l,k)) \tag{3.39}$$

Similarly, a measure of the average link availability in the network can be

$$P_a(N) = \frac{1}{L} \sum_{l \in \mathbf{L}} (1 - B(l)) \tag{3.40}$$

or if one is interested in average link utilization

$$\overline{U_N} = \frac{1}{L} \sum_{l \in \mathbf{L}} B(l) \tag{3.41}$$

Note that in $B(l)$ (eq.3.20) the control traffic overhead is also incorporated and thus the effect of the control traffic is taken into account in these quantities.

Control traffic overhead

In addition to the presented performance metrics, using this model it is also possible to calculate the actual effect of the control traffic in the performance of the network. For example, denote by $B^0(l) = \sum_{k \in \mathbf{U}} [q_L(l,k)]$ the utilization of the link ignoring the control traffic and $B'(l) = \sum_{k \in \mathbf{U}} [q_L(l,k) + \sum_{i \in \mathbf{R}} c(l,(i,k))]$ the utilization when the control traffic is present. Then, using equation 3.41 the average overhead on the link utilization is simply calculated as following

$$\Delta\overline{U_N} = \frac{1}{L} \sum_{l \in \mathbf{L}} \frac{B'(l) - B^0(l)}{B^0(l)} \tag{3.42}$$

Similarly, it is possible to estimate the increase due to the control traffic in the average network delay etc. In general, the effect of the control traffic on a cost function C will be easily calculated by

$$\Delta C = \frac{C'(\mathbf{q}_k, \mathbf{c}^R_{i,k}, \mathbf{c}^L_{i,k}) - C^0(\mathbf{q}_k)}{C^0(\mathbf{q}_k)} \tag{3.43}$$

Energy per packet

Another metric that could be used instead of the total power consumption is the energy per packet, which can be expressed by

$$epp = \frac{P_i(\Lambda(i))}{\Lambda(i)} \tag{3.44}$$

where $\Lambda(i)$ (eq.(3.29)) is the total traffic carried by node i in pkts/sec and P_i is the function giving the power consumption profile of the node in Watts. Thus, this quantity is measured in $(Joules/sec)/(pkt/sec) = Joules/pkt$ and can give the energy consumed for each packet at node i. One could also be interested in the average energy per packet of a specific class k spent through the network, i.e.

$$F_{epp}(k) \quad = \quad \sum_{i \in \mathbf{N}} \pi(i,k) \frac{P_i(\Lambda(i))}{\Lambda(i)} \tag{3.45}$$

The average energy per packet of a network can be expressed by

$$N_{epp} \quad = \quad \frac{1}{U} \sum_{k \in \mathbf{U}} \frac{\lambda^+(k)}{\Lambda_T^+} F_{epp}(k) \tag{3.46}$$

The energy per packet of a specific class F_{epp} is very useful to illustrate the efficiency of an end-to-end route compared to another. However, in case of the average network energy per packet N_{epp}, it should be noted that the nodes which carry no traffic at all are not taken into consideration. Thus, if the empty nodes still consume power (static power consumption) the N_{epp} metric is not appropriate measure for the network, since the energy waisted for those devices is not incorporated.

3.4 Optimization

3.4.1 Optimization goals

The routing optimization can now be expressed as the minimization of a function f that includes both the network power consumption and the average delay:

$$\text{Minimize } f = c_1 P_N + c_2 \overline{T_N} \tag{3.47}$$

where the parameters c_1 and c_2 are used to tune the relative importance of the two parameters. The minimization can be achieved by selecting appropriate route control parameters $Q(x, m, y)$.

Another approach would be to have a bounded delay goal. This can be expressed by

$$\text{Minimize } f = c_1 P_N + c_2 \overline{T_N} \times 1[(\overline{T_N} - T_{max}) > 0] \tag{3.48}$$

where T_{max} is the average delay bound and the expression $1[x]$ is 1 if x is true and 0 otherwise.

Note that some nodes' power consumption could be more critical than others. Also, the delay of some flows might be more crucial depending on their individual QoS needs. So, a more generalized optimization goal would be

$$\text{Minimize } f = \sum_{i \in \mathbf{N}} c_1(i) P_i + \sum_{k \in \mathbf{U}} c_2(k) T(k) \tag{3.49}$$

Finally, note that some flows may be delay-sensitive and others could be sensitive to packet loss. The cost function can be differentiated to a different QoS constraint for each user. If $\sigma(k)$ denotes the specific QoS constraint of each user, then the cost function will be modified as follows

$$\text{Minimize } f = \sum_{i \in \mathbf{N}} c_1(i) P_i + \sum_{k \in \mathbf{U}} c_2(k) \sigma(k) \tag{3.50}$$

where $\sigma(k)$ can be end-to-end delay (eq. 3.31), packet loss (eq. 3.35), path availability (eq. 3.39) etc.

Other optimization goals could include other metrics presented before and could be adjusted to the specific problem requirements.

3.4.2 Gradient descent optimization

In this section, an optimization algorithm that attempts to minimize the cost function of eq.(3.47) is built. The minimization can be achieved by selecting appropriate route control parameters $Q(x, m, y)$. Since we are interested in gradual optimization in the presence of ongoing flows, the partial derivative of f is computed:

$$\frac{\partial f}{\partial Q(x,m,y)} = c_1 \sum_{i \in \mathbf{N}} \frac{\partial P_i}{\partial Q(x,m,y)} + c_2 \frac{\partial \overline{T_N}}{\partial Q(x,m,y)} \tag{3.51}$$

where the average power consumption is used from (3.36), (3.37):

$$\begin{aligned}
\sum_{i\in\mathbf{N}} \frac{\partial P_i}{\partial Q(x,m,y)} &= \sum_{i\in\mathbf{R}} \frac{\partial g_R(\Lambda_i)}{\partial(\Lambda_i)} \frac{\partial(\Lambda_i)}{\partial Q(x,m,y)} + \sum_{i\in\mathbf{L}} \frac{\partial g_L(\Lambda_i)}{\partial(\Lambda_i)} \frac{\partial(\Lambda_i)}{\partial Q(x,m,y)} \\
&= \sum_{i\in\mathbf{R}} \frac{\partial g_R(\Lambda_i)}{\partial(\Lambda_i)} \sum_{k\in\mathbf{U}} \frac{\partial \Lambda_R(i,k)}{\partial Q(x,m,y)} \\
&\quad + \sum_{i\in\mathbf{L}} \frac{\partial g_L(\Lambda_i)}{\partial(\Lambda_i)} \sum_{k\in\mathbf{U}} \frac{\partial \Lambda_L(i,k)}{\partial Q(x,m,y)} \Rightarrow \\
\sum_{i\in\mathbf{N}} \frac{\partial P_i}{\partial Q(x,m,y)} &= \sum_{k\in\mathbf{U}} [\sum_{i\in\mathbf{L}} \frac{\partial g_L(\Lambda_i)}{\partial(\Lambda_i)} \frac{\partial q(i,k)}{\partial Q(x,m,y)} \mu_i \\
&\quad + \sum_{i\in\mathbf{R}} \frac{\partial g_R(\Lambda_i)}{\partial(\Lambda_i)} \frac{\partial q(i,k)}{\partial Q(x,m,y)} (\mu_i + \Lambda^-(i,(i,k)))] \qquad (3.52)
\end{aligned}$$

and the average delay from (3.33):

$$\begin{aligned}
\frac{\partial \overline{T_N}}{\partial Q(x,m,y)} &= \sum_{k\in\mathbf{U}} \frac{\lambda^+(k)}{\Lambda_T^+} \Bigg[\sum_{r\in\mathbf{R}} \frac{\pi(r,k)}{\Lambda_R(r,k)(1-q(r,k))^2} \frac{\partial q(r,k)}{\partial Q(x,m,y)} \\
&\quad + \sum_{l\in\mathbf{L}} \frac{\pi(l,k)}{\Lambda_L(l,k)(1-B(l))^2} \Big((1-B(l)) \frac{\partial q(l,k)}{\partial Q(x,m,y)} \\
&\quad + q(l,k) \sum_{i\in\mathbf{U}} \frac{\partial q(l,i)}{\partial Q(x,m,y)} \Big) \Bigg] \qquad (3.53)
\end{aligned}$$

Since $\frac{\partial P(i,k,j)}{\partial Q(x,m,y)} = \frac{\partial p((i,k),j,n)}{\partial Q(x,m,y)} = 0$, the portions $\frac{\partial q(r,k)}{\partial Q(x,m,y)}$ and $\frac{\partial q(l,k)}{\partial Q(x,m,y)}$ are calculated. Define $h(i,j) = 1$ if there is a physical connection from node i to j and $h(i,j) = 0$ otherwise. Note that for evaluating $\frac{\partial \mathbf{X}}{\partial Q(x,m,y)}$ only the cases where $h(x,y) = 1$ need to be taken under consideration, since when $h(x,y) = 0$ the partial derivatives will be 0. Now define the vector $\mathbf{q}_k = (q(1,k), q(2,k), .., q(N,k))$ and the $N \times N$ matrices $\mathbf{A}_k = [A_k(l,r)], \mathbf{D}_k = [D_k(r,l)], \mathbf{B}_k = [B_k(l,r)], \mathbf{C}_k = [C_k(r,l)]$, where $r \in \mathbf{R}$ and $l \in \mathbf{L}$:

$$A_k(l,r) = \frac{P(l,k,r)}{\mu_r + \Lambda_R^-(r,(r,k))} \qquad (3.54)$$

$$D_k(r,l) = [P(r,k,l)\mu_r + \Lambda_R^-(r,(r,k))Q(r,k,l)] \qquad (3.55)$$

$$B_k(l,r) = \mu_l P(l,k,r) \qquad (3.56)$$

$$C_k(r,l) = \frac{[P(r,k,l)\mu_r + \Lambda_R^-(r,(r,k))Q(r,k,l)]}{\mu_l[\mu_r + \Lambda_R^-(r,(r,k))]} \qquad (3.57)$$

and the $1 \times N$ row vectors :

$$\mathbf{M}(l) = 1/\mu_l \tag{3.58}$$

$$\mathbf{H}_k^{xmy}(l) = \begin{cases} \Lambda_R^-(x,(x,k))q(x,k) & k = m, l = y \\ -h(x,l)\Lambda_R^-(x,(x,k))q(x,k) & k = m, l \neq y \\ 0 & \text{otherwise} \end{cases} \tag{3.59}$$

Then the partial derivatives can be expressed as:

$$\frac{\partial \mathbf{q}_k(r)}{\partial Q(x,m,y)} = \frac{\partial \mathbf{q}_k(r)}{\partial Q(x,m,y)}\mathbf{D}_k\mathbf{A}_k + \mathbf{H}_k^{xmy}\mathbf{A}_k \tag{3.60}$$

$$\frac{\partial \mathbf{q}_k(l)}{\partial Q(x,m,y)} = \frac{\partial \mathbf{q}_k(l)}{\partial Q(x,m,y)}\mathbf{B}_k\mathbf{C}_k + \mathbf{H}_k^{xmy}\mathbf{M} \tag{3.61}$$

Thus, equations (3.60) and (3.61) can be written as:

$$\frac{\partial \mathbf{q}_k}{\partial Q(x,y,m)} = \frac{\partial \mathbf{q}_k}{\partial Q(x,y,m)}\mathbf{W}_k + \boldsymbol{\gamma}_k^{xmy} \tag{3.62}$$

where the matrix $\mathbf{W}_k$ and the vector $\boldsymbol{\gamma}_k^{xmy}$ are given by

$$W_k(i,j) = \begin{cases} \sum_{l\in\mathbf{L}} D_k(i,l)A_k(l,j) & i,j \in \mathbf{R} \\ \sum_{r\in\mathbf{R}} B_k(i,r)C_k(r,j) & i,j \in \mathbf{L} \end{cases} \tag{3.63}$$

$$\gamma_k^{xmy}(n) = \begin{cases} \sum_{l\in\mathbf{L}} H_k^{xmy}(l)A_k(l,n) & n \in \mathbf{R} \\ H_k^{xmy}(n)M(n) & n \in \mathbf{L} \end{cases} \tag{3.64}$$

So,

$$\frac{\partial \mathbf{q}_k}{\partial Q(x,y,m)} = \boldsymbol{\gamma}_k^{xmy}(\mathbf{I} - \mathbf{W}_k)^{-1} \tag{3.65}$$

where $\mathbf{I}$ is the $N \times N$ identity matrix. Using (3.65) the $\partial f/\partial Q(x,m,y)$ can be calculated from equations (3.51), (3.52) and (3.53), and the matrix inversion is of time complexity $O(N^3)$.

The corresponding gradient descent algorithm to obtain the parameters $Q(i,k,j)$ that reduce the cost function at a given operating point $\underline{X} = [\underline{\lambda}, \underline{\lambda^-}, \underline{\mu}, \underline{P^+}, \underline{p}]$ of the network is then determined by its n^{th} computational step:

$$Q_{n+1}(i,k,j) = Q_n(i,k,j) - \eta \frac{\partial f}{\partial Q(i,k,j)}|_{Q(i,k,j)=Q_n(i,k,j)} \tag{3.66}$$

where $\eta > 0$ is the "rate" of the gradient descent and the partial derivative is computed with the nth updated values of the weights.

The steps of the learning algorithm are then:

1. First initialize all the values $Q(i,k,j)$ and choose $\eta > 0$.
2. Solve the U systems of the N equations (3.7)-(3.14) based on the current state to obtain the values of $q(i,k)$.
3. Solve the U systems of N linear equations (3.65) using the $q(i,k)$.
4. Using the results from steps 2 and 3 update the values $Q(i,k,j)$ using (3.66)
5. Repeat until the change in cost function or in the values of $Q(i,k,j)$ is smaller than some predetermined value ϵ.

The gradient descent algorithm has been inspired by [42], where the gradient descent has been used for learning in the random neural network (RNN) [38]. Taking advantage of the similarity of the G-networks to the RNN, this algorithm has been modified to perform optimization, instead of learning, in the energy efficient routing control context.

Summary of important equations

In this section, the equations of the node utilization are simplified and all the important quantities are expressed in matrix form for easiest usage in calculations. The vector $\mathbf{q}_k$ can be divided into the router vector $\mathbf{q}_k^R$ and the link vector $\mathbf{q}_k^L$. The vector for the routers will then be:

$$\mathbf{q}_k^R = [q_k(r)]_{1\times R}, \quad q_k^R = (q^R(1,k), q^R(2,k), .., q^R(R,k)) \tag{3.67}$$

and using the following matrices

$$\mathbf{X}_k = [X_k(r)]_{1\times R}, \quad X_k(r) = \frac{\lambda(r,k)}{\mu_r + \Lambda_R^-(r,(r,k))} \tag{3.68}$$

$$\mathbf{A}_k = [A_k(l,r)]_{L\times R}, \quad A_k(l,r) = \frac{P(l,k,r)}{\mu_r + \Lambda_R^-(r,(r,k))} \tag{3.69}$$

$$\mathbf{D}_k = [D_k(r,l)]_{R\times L}, \quad D_k(r,l) = P(r,k,l)\mu_r + \Lambda_R^-(r,(r,k))Q(r,k,l) \tag{3.70}$$

the vector $\mathbf{q}_k^R$ can be expressed as:

$$\begin{aligned} & \mathbf{q}_k^R = \mathbf{X}_k + \mathbf{q}_k^R \mathbf{D}_k \mathbf{A}_k \\ \Rightarrow \quad & \mathbf{q}_k^R = \mathbf{X}_k (\mathbf{I} - \mathbf{D}_k \mathbf{A}_k)^{-1} \end{aligned} \tag{3.71}$$

The vector $\mathbf{q}_k^L$ is respectively defined by:

$$\mathbf{q}_k^L = [q_k(l)]_{1\times L}, \quad \mathbf{q}_k^L = (q^L(1,k), q^L(2,k), .., q^L(L,k)) \tag{3.72}$$

The equations of the links are coupled with the equations of the routers so the vector $\mathbf{q}_k^L$ can be expressed as:

$$\mathbf{q}_k^L = \mathbf{q}_k^R \mathbf{E}_k \tag{3.73}$$

where the matrix $\mathbf{E}_k$ is defined as

$$\mathbf{E}_k = [E_k(r,l)]_{R\times L}, \quad E_k(r,l) = \frac{P(r,k,l)\mu_r + \Lambda_R^-(r,(r,k))Q(r,k,l)}{\mu_l} \tag{3.74}$$

The partial derivatives of the vector $\mathbf{q}_k^R$ defined as:

$$\frac{\partial \mathbf{q}_k^R}{\partial Q(x,m,y)} = (\frac{\partial q^R(1,k)}{\partial Q(x,m,y)}, \frac{\partial q^R(2,k)}{\partial Q(x,m,y)}, .., \frac{\partial q^R(R,k)}{\partial Q(x,m,y)}) \tag{3.75}$$

can be expressed as shown before

$$\begin{aligned} \frac{\partial \mathbf{q}_k^R}{\partial Q(x,m,y)} &= \frac{\partial \mathbf{q}_k^R}{\partial Q(x,m,y)} \mathbf{D}_k \mathbf{A}_k + \mathbf{H}_k^{xmy} \mathbf{A}_k \\ \Rightarrow \quad \frac{\partial \mathbf{q}_k^R}{\partial Q(x,m,y)} &= \mathbf{H}_k^{xmy} \mathbf{A}_k (\mathbf{I} - \mathbf{D}_k \mathbf{A}_k)^{-1} \end{aligned} \tag{3.76}$$

where

$$\mathbf{H}_k^{xmy} = [H_k^{xmy}(l)]_{1\times L} \quad H_k^{xmy}(l) = \begin{cases} \Lambda^-(x,(x,k))q(x,k), & \text{k=m,l=y} \\ -h(x,l)\Lambda^-(x,(x,k))q(x,k), & \text{k=m,l}\neq \text{y} \\ 0, \qquad \text{otherwise} \end{cases} \tag{3.77}$$

The derivatives of the vector $\mathbf{q}_k^L$ defined respectively as:

$$\frac{\partial \mathbf{q}_k^L}{\partial Q(x,m,y)} = (\frac{\partial q^L(1,k)}{\partial Q(x,m,y)}, \frac{\partial q^L(2,k)}{\partial Q(x,m,y)}, .., \frac{\partial q^L(l,k)}{\partial Q(x,m,y)}) \tag{3.78}$$

are expressed using (3.76) as:

$$\frac{\partial \mathbf{q}_k^L}{\partial Q(x,m,y)} = \frac{\partial \mathbf{q}_k^R}{\partial Q(x,m,y)} \mathbf{E}_k + \mathbf{S}_k^{xmy} \tag{3.79}$$

where the matrix $\mathbf{E}_k$ is defined in (3.74) and $\mathbf{S}_k^{xmy}$ is given by:

$$\mathbf{S}_k^{xmy} = [S_k^{xmy}(l)]_{1\times L}, \quad S_k^{xmy}(l) = \frac{H_k^{xmy}(l)}{\mu_l} \tag{3.80}$$

The equations for the control traffic can be also expressed in matrix form

$$\mathbf{c}_{(i,k)}^R = [c_{(i,k)}^R(r)]_{1\times R}, \quad c_{(i,k)}^R(r) = c_R(r,(i,k)) \tag{3.81}$$

$$\mathbf{p}_{(i,k)}^R = [p_{(i,k)}^R(r,l)]_{R\times L} \quad p_{(i,k)}^R(r,l) = p((i,k),r,l), \mathrm{r} \in \mathbf{R}, \mathrm{l} \in \mathbf{L} \tag{3.82}$$

$$\mathbf{p}_{(i,k)}^L = [p_{(i,k)}^L(l,r)]_{L\times R} \quad p_{(i,k)}^R(l,r) = p((i,k),l,r), \mathrm{r} \in \mathbf{R}, \mathrm{l} \in \mathbf{L} \tag{3.83}$$

$$\begin{aligned} & [\mathbf{c}_{(i,k)}^R \circ \boldsymbol{\mu}_r] = \boldsymbol{\lambda}_{(i,k)}^- + [\mathbf{c}_{(i,k)}^R \boldsymbol{\mu}_r^T] \mathbf{p}_{(i,k)}^R \mathbf{p}_{(i,k)}^L \\ \Rightarrow \quad & [\mathbf{c}_{(i,k)}^R \circ \boldsymbol{\mu}_r] = \boldsymbol{\lambda}_{(i,k)}^- (\mathbf{I} - \mathbf{p}_{(i,k)}^R \mathbf{p}_{(i,k)}^L)^{-1} \end{aligned} \tag{3.84}$$

where $\boldsymbol{\mu}_r = (\mu(1), .., \mu(R))$ is the vector of the service rates and $[\mathbf{c}_{(i,k)}^R \circ \boldsymbol{\mu}_r]$ denotes the element-by-element product of the two vectors. For the links respectively:

$$[\mathbf{c}_{(i,k)}^L \circ \boldsymbol{\mu}_l] = [\mathbf{c}_{(i,k)}^R \circ \boldsymbol{\mu}_r] \mathbf{p}_{(i,k)}^R \tag{3.85}$$

3.5 Conclusions

In this chapter, a network queuing model is presented, which is based on G-networks with multiple classes and triggered customer movement. This model offers a theoretical framework capturing all the important parameters of the energy efficient routing control problem under QoS considerations. Firstly, our model can distinguish between different users with different source-destination pairs, traffic inputs and QoS needs. Also, thanks to the product form solution it can estimate the traffic at each node in steady state and thus estimate power consumption at each node as well as queue length and network performance metrics. The effect of control traffic and control decisions is also included such as the overhead both on power consumption and on performance can be taken into account. A generalized power consumption model is presented and links are also treated as nodes, so that their individual power consumption can be incorporated. This model can be applied for any network case to design algorithms that apply control in networks so as to optimize a given cost function and estimate the performance of different policies and network designs.

Having built the network model, a gradient descent optimization algorithm is designed, which can run in $O(N^3)$ complexity to gradually re-route traffic. Several QoS

metrics are presented and discussed. A composite cost function is proposed, comprising both of the network power consumption and QoS, which can be used to tune the relative importance of power savings and performance. To the best of the author's knowledge, this is the first time a queuing network model on energy efficient routing control under QoS considerations is proposed.

4 Evaluation of energy aware routing

4.1 Introduction

The evaluation of any energy saving solution largely depends on the energy consumption characteristics of the network infrastructures. As discussed in Chapter 2, obtaining energy consumption characteristics for real network nodes is challenging and has yet to be fully achieved. Moreover, as there is ongoing research towards energy saving solutions at the hardware level, the overall picture of the network power consumption behavior is expected to change. In the analysis in Chapter 3, a generalized power consumption model is presented which describes the devices' power consumption as a function of their load (eq. 3.36) and can be adapted to the specific characteristics of each scenario. Thus, in this chapter the previously presented network model is used to examine several case studies depending on power consumption behavior of the network nodes and traffic volumes. The gradient descent optimization algorithm is applied and power consumption savings are estimated for each case.

The power savings often come at the cost of network performance degradation in terms of increased delays. Thus, combined optimization goals are proposed and examined, consisting both of power consumption and delay metrics, in order to moderate the increase in delay when required.

Finally, as the gradient descent optimization would be slow for online calculations, a faster heuristic that could be used online is proposed. This heuristic can perform a routing change that leads to a more energy efficient state, instead of searching for the optimal state that would be time consuming. The case of decentralized decisions is also explored, where each source has partial information for a subnetwork and searches for the best routing within this subnetwork. The heuristic results are compared to the optimization results and advantages and drawbacks are examined. In the last section, our laboratory PC-based network topology is used and experiments are performed with real measured power profiles and in combination with a self-aware routing protocol.

Our aim in this chapter is to examine under which circumstances the energy efficient routing will have significant savings and its effect on the network delay. Also, we seek to explore the limitations of the gradient descent and propose heuristics and simplified solutions.

4.1.1 Setting

For the evaluations in this section, the realistic topology shown in Figure 4.1 is used, which is the topology of Czech Republic's National Research and Education Network (CESNET) [2]. The end nodes are used as source and destination nodes and it is assumed that the power consumption profiles of the nodes is known. The source destination pairs used were randomly selected: (105-104), (020-002), (026-109),(108-021), (106-114), (114-105), (104,020), (109-106), (028-026). Random propagation delays are added to the links, distributed between 1-100ms.

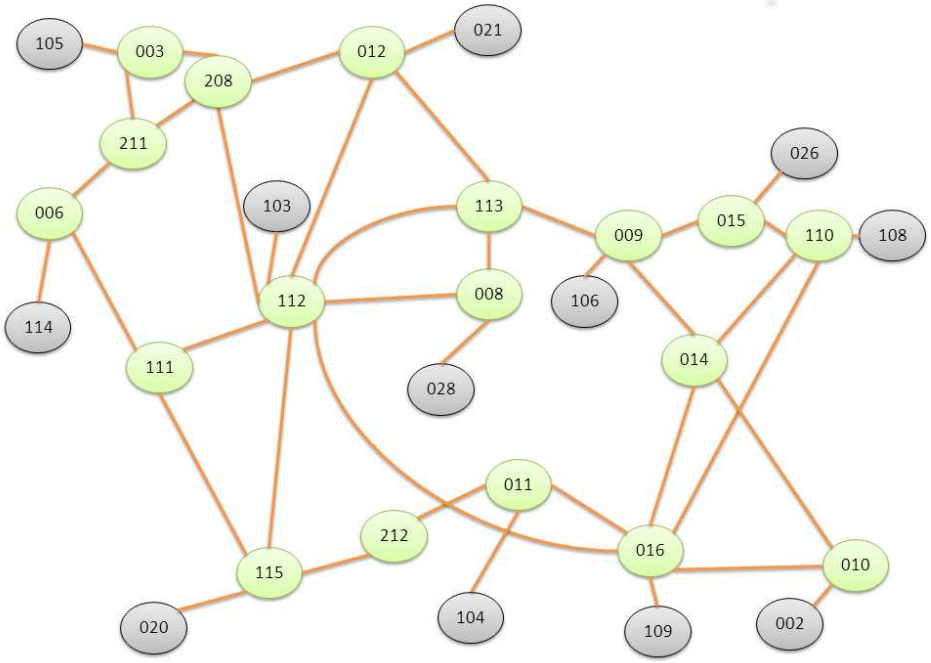


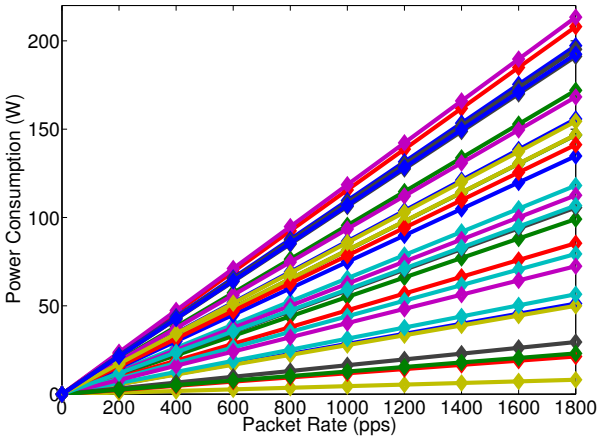
Figure 4.1: Network topology used in the evaluation of energy aware routing. It consists of 29 nodes, connected in the realistic CESNET topology [2]

Regarding the node power consumption characteristics, the following cases of node power profiles were examined:

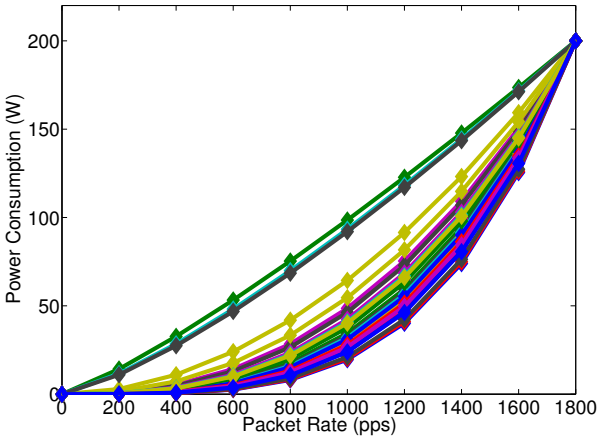
(i) linear power profiles (Figure 4.2a)

(ii) convex power profiles (Figure 4.2b)

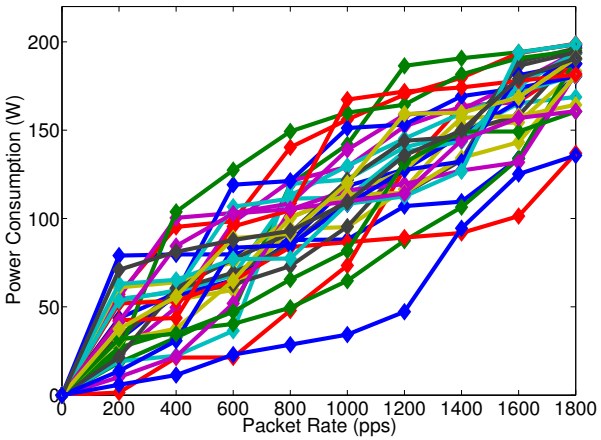
(iii) randomly increasing multi-step power profiles (Figure 4.2c)



(a) Linear power profiles



(b) Convex power profiles



(c) Multi-step power profiles

Figure 4.2: Different cases of power profiles used in the evaluation examples. A power profile curve among those was assigned to each node in a random manner.

These cases of power profiles were selected, based on a recent survey [15] which identifies and categorizes power profiles, as shown in Figure 4.3, spanning from the ideal case of energy proportional or the case of low-traffic optimized convex profile, to the extreme case of energy agnostic.

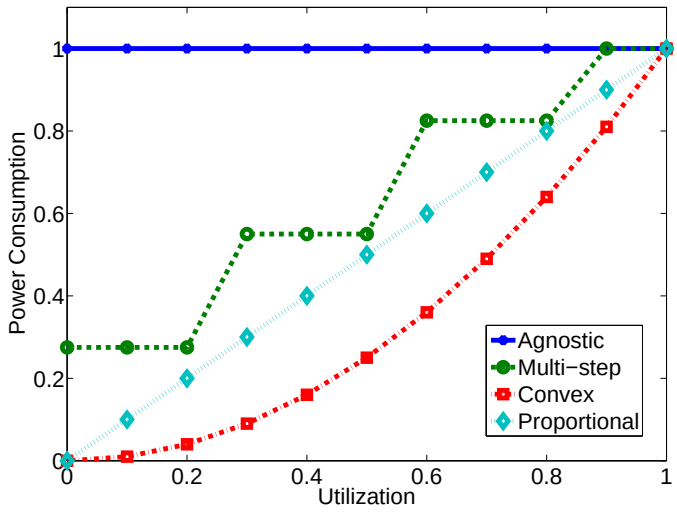


Figure 4.3: Power consumption as a function of utilization. Different power consumption behaviors of nodes versus the utilization, as presented in [15]

In the last section, the case of real measured profiles of PC-based routers (Figure 4.27) is also examined. These correspond to the most typical scenarios, as also shown before (Figure 4.3), and cover all cases from ideal to real and predicted future characteristics of devices.

4.2 Power optimization case studies

4.2.1 Algorithm initialization

First, the result of the optimization algorithm when used to optimize *average delay* is examined. The results of the optimization algorithm are compared to the shortest path solution calculated by Dijkstra algorithm. The aim of this section is to show the importance of the initialization of the routing matrices during the first step of the described algorithm (section 3.4.2) and how it may influence the algorithm's efficiency. Thus, different initializations of the routing matrices may lead to different solutions as local minima may be reached. We compare 3 different initializations to the algorithm:

- Perturbed shortest path : In this case the routing decisions were initialized to a

state slightly perturbed from the shortest path. Thus, 70% of the traffic is routed according to the shortest path, while the rest is equally split among the other possible routing decisions.

- Load balance : In this case, at each node we equally split all the traffic to all the alternative outgoing links.
- Random : In this case an arbitrary initial state is selected where each flow follows one random path.

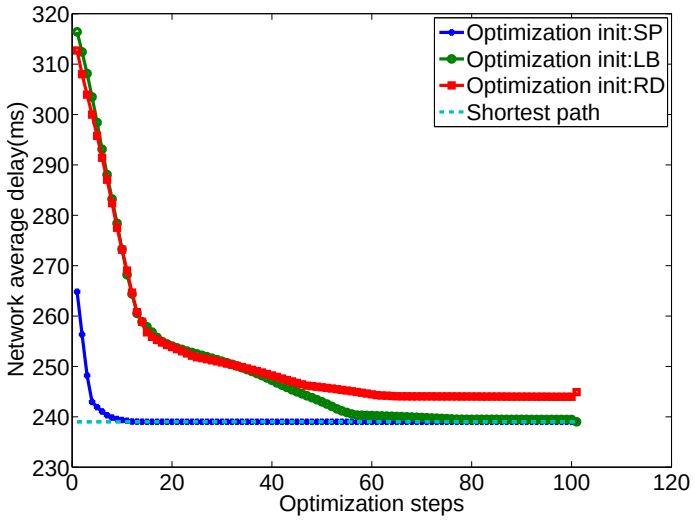


Figure 4.4: Average network delay results : gradient descent algorithm initializations comparison when used to optimize delay. Random initialization resulted in a local minimum.

As can be observed from Figure 4.4, the algorithm converges very quickly to the shortest path (dotted line) for the case of the perturbed shortest path initialization (solid blue line) and also successfully converges to the shortest path for the load balance initialization (solid green line) after approximately 70 steps. Though, for the case of random initialization (solid red line) it gets stuck to a local minimum, increased by 3% comparing to the optimal shortest path solution. Thus, the initialization of the algorithm is critical for its efficiency in such a complex system and it is important to select the initial state wisely, having in mind the local minima that may arise. The load balance initialization, although it was not close to the optimal solution as was the case for the perturbed shortest path, succeeded in converging to the shortest path.

4.2.2 Optimizing power consumption

Linear power profiles

In this section, the behavior of a network which comprises of nodes whose power profiles increase linearly to their utilization is studied. This case corresponds to the ideal case of energy proportional networking devices as mentioned in section 2.4.2 and examined in [9]. Specifically, the case where each node has a different linear power profile is examined and each network node is randomly assigned to one of the linear profile curves of Figure 4.2a.

A cost function comprising only of the total network power consumption of all network nodes is used, i.e.

$$\text{Minimize } f = \sum_{i \in \mathbf{R}} P_i \tag{4.1}$$

Again, the 3 initializations presented in the previous section are compared and a 100pps traffic for each of the nine source-destination pairs is assumed. The total power consumption results are plotted versus the steps of the optimization algorithm in Figure 4.5. As can be observed, in this case the load balance initialization of the algorithm yields the best results with total savings of approximately 8% in power consumption. The load balance initialization will be thus used in the sequel, unless otherwise stated.

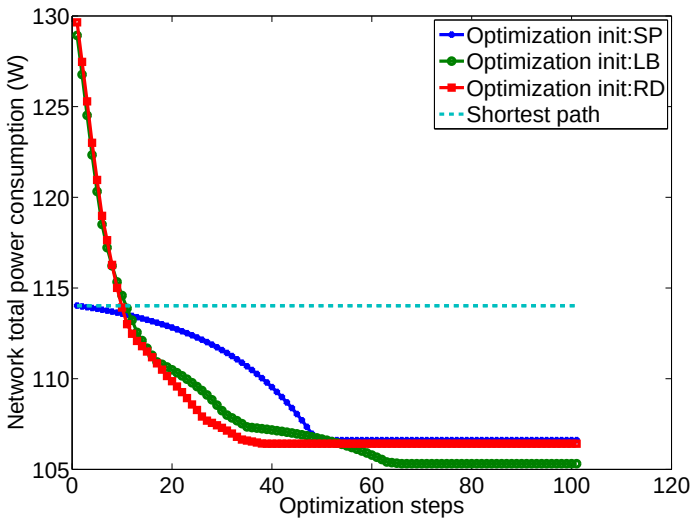


Figure 4.5: Power consumption: Optimization results over the optimization steps, for linear power profiles. Power optimization with load balance initialization leaded to the best results, saving 8% in total power consumption compared to the shortest path routing.

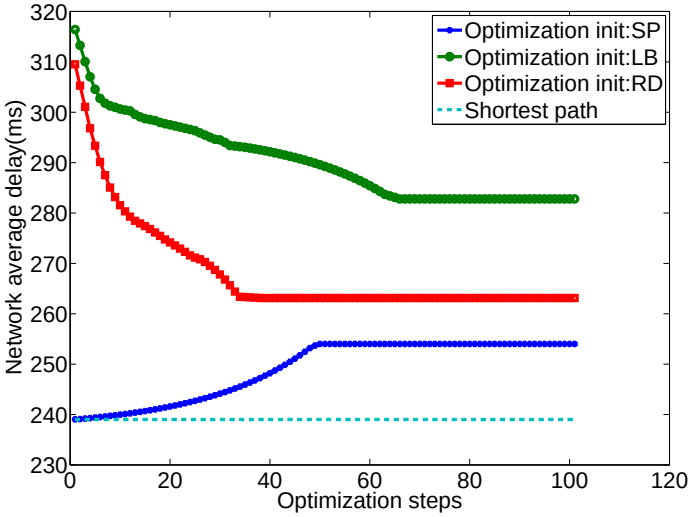


Figure 4.6: Average delay: Optimization results over the optimization steps, for linear power profiles. The power optimization resulted in increased average packet delay, for all initializations.

On the other hand, the average delay is shown in Figure 4.6. An increase in delay is observed as the routing matrix is selected to optimize energy efficiency and the selected paths are no longer the shortest ones. Note that for the load balance and the random initialization the average delay is being decreased in the course of the optimization, but all three cases converge at an increased average delay value comparing to the shortest path (dotted line). Thus, even for the case of ideal linear power profiles, significant savings in power consumption can be achieved by using routing control, possibly in expense of increased delay.

Next, the optimization is repeated for traffic matrices scaled by a factor 0.3 to 3 times the initial value. The average power consumption results are plotted in Figure 4.7 versus the total traffic input in the network. It can observed, that the total wattage savings become greater as the traffic in the network increases. Though, as shown in Figure 4.8, the resulted average delay is exactly the same for all values of total traffic input. This indicates that the algorithm converges to the same routing matrix solution for all cases. In other words, for the case of linear power profiles, there is one optimal solution for every set of traffic and the optimization needs not to be re-run in case of traffic variations. The savings in this case correspond approximately to 8% of the total network power while the delay is increased by around 18%.

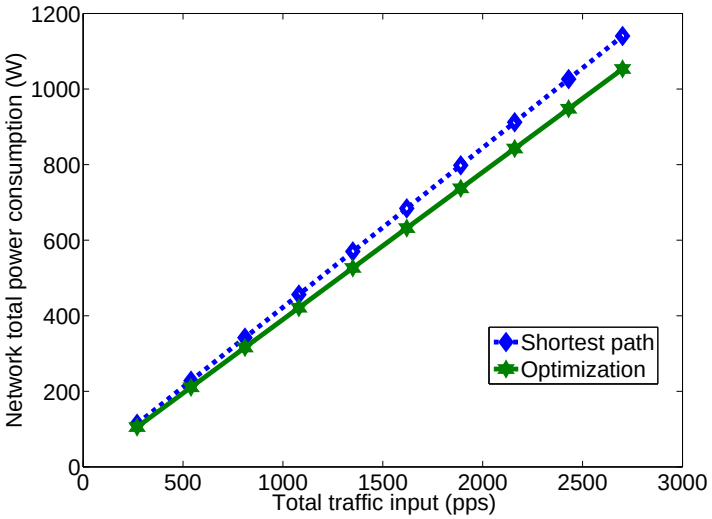


Figure 4.7: Power consumption: Optimization results versus increasing total traffic input for linear power profiles. A power saving of 8% was observed in all cases.

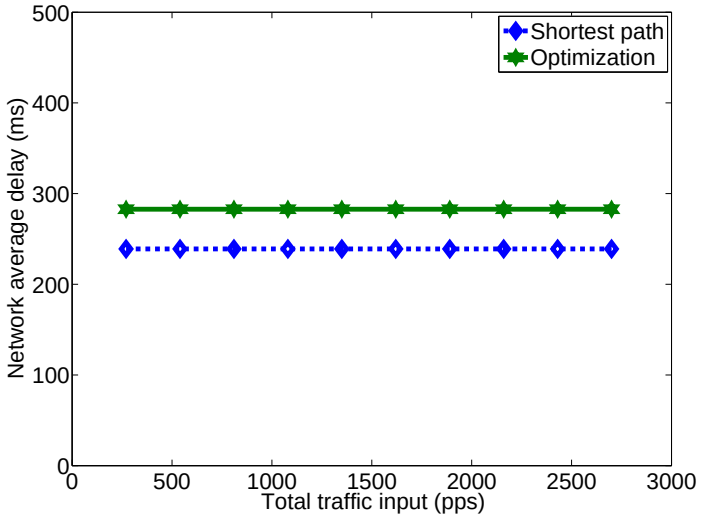


Figure 4.8: Average delay : Optimization results versus total traffic input for linear power profiles. Average packet delay was increased by 18%.

To summarize, we have observed that for the case of linear power profiles, routing control can be used to reduce power consumption at the cost of increased delay. In fact, due to linearity, a fixed volume of traffic evokes the same increase in power consumption in a node, regardless of the total carried traffic. Thus, each source-destination pair will have one power-optimal path regardless of the the rest of the traffic in the network and traffic variations. The power optimization in this case should be run only once and

is not dependent on traffic changes. Note also that if all network nodes had identical linear power profiles, the optimal solution would simply be the minimum-hop path routing. Though, linear behavior is the ideal case while the optimization becomes more complicated for more realistic power profiles, as will be shown in the sequel.

Convex power profiles

In this section, the convex power profiles as shown in Figure 4.2b are examined. This type of power profiles could result from the frequency scaling techniques [90] and would be optimized for low utilization. The optimization is repeated with traffic matrices scaled by an increasing factor of 0.3 to 3 times the initial value and the power consumption results are plotted in Figure 4.9. In this case the savings in power consumption are much more significant than the previous case of linear power profiles. The average power consumption is reduced by 40%-60%. In Figure 4.10 an increase in delay is observed, varying between 13% and 20%.

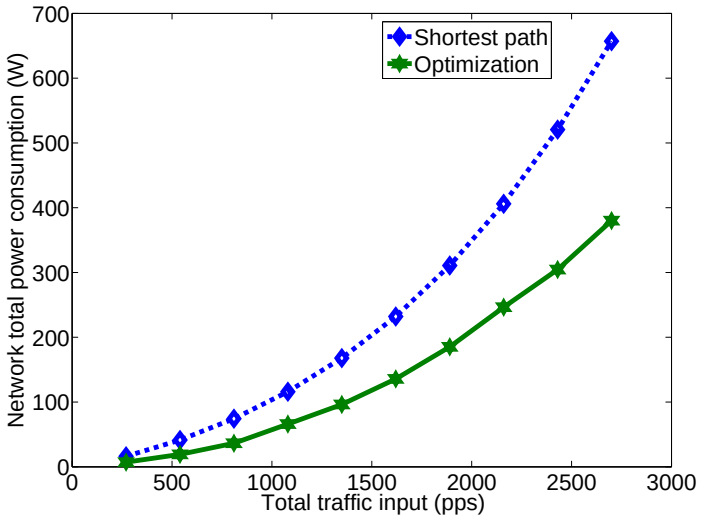


Figure 4.9: Power consumption: Optimization results versus total traffic input for convex power profiles. Power savings ranged from 40%-60%.

Thus, this type of power profiles would offer large room for energy savings through energy efficient routing policies. Moreover, as observed in Figure 4.10, the resulted average delay is different for different values of total traffic, which indicates that the routing solution is not the same. Thus, in this case the traffic variations will affect the optimization solution. The power consumption of a flow is largely dependent on the

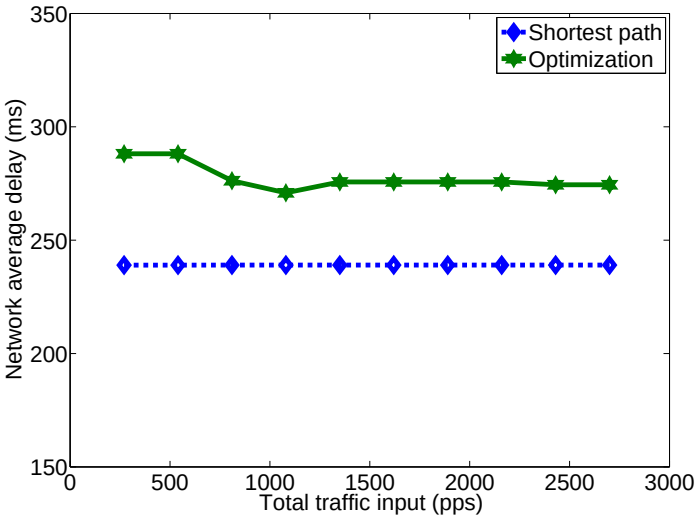


Figure 4.10: Average delay: Optimization results versus total traffic input for convex power profiles. Average packet delay was increased up to 20%.

rest of the traffic carried by the nodes of its path and even a small change in the traffic volume of one single flow may affect the energy-optimal routing of other existing flows.

Multi-step power profiles

In this section, the case of randomly increasing multi-step power profiles is examined, which would correspond to a more general category of power profiles. This type of power profiles could represent the realistic result after the implementation of energy saving techniques described in Section 2.4.2. It is assumed that distinct measurements of the power consumption have been collected, which are randomly increasing as shown in Figure 4.2c, and the profiles are approximated with polynomial fitting to the data.

Again, the optimization is run for traffic matrices scaled by an increasing factor of 0.3 to 3 times the initial values to investigate the effect of different traffic volume in the network. The power consumption results shown in Figure 4.11 indicate that the savings vary from 6% to 23%, as total input traffic increases.

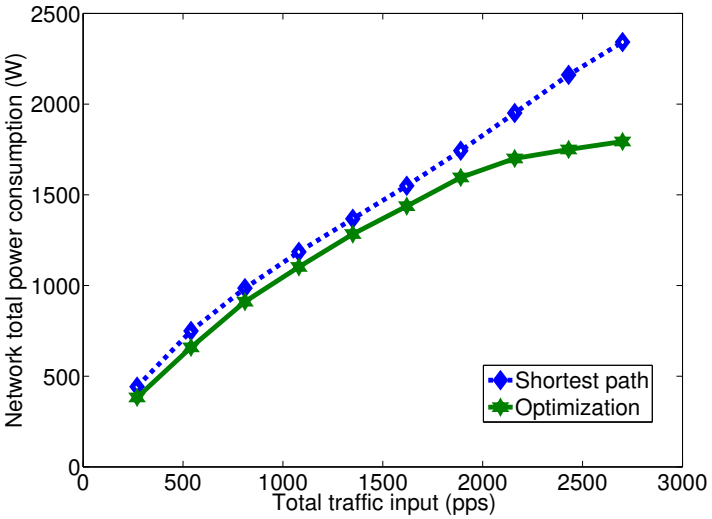


Figure 4.11: Power consumption: Optimization results versus total traffic input for randomly increasing power profiles. Power consumption savings ranged from 6%-23%.

The delay result, as shown in Figure 4.12, is different for different traffic input volumes which again indicates that as the traffic changes the algorithm results in a different routing matrix. Average delay increase varies from 17%-28%.

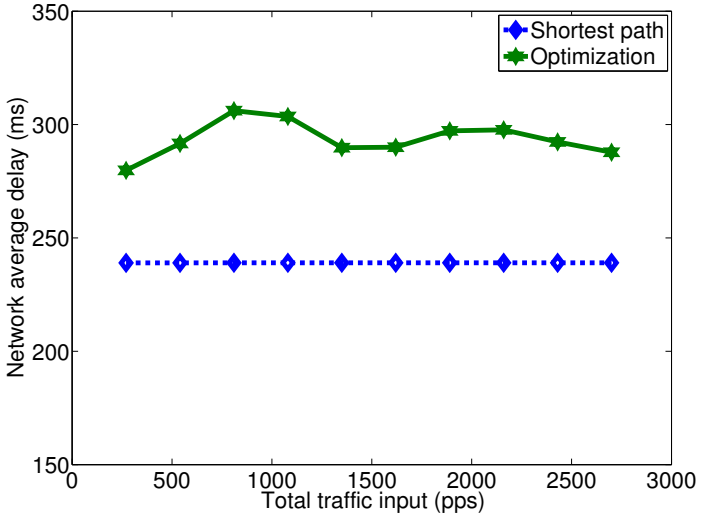


Figure 4.12: Average delay: Optimization results versus total traffic input for randomly increasing power profiles. Average packet delay was increased up to 28%.

4.2.3 Combined optimization goal

Note that so far the presented results account for the optimization of the cost function 4.1 which only includes the total power consumption. As shown in the presented examples, in all cases the power savings come at an expense of increased network average delay. Thus, as was proposed in Section 3.4.1, it would be sensible to use a combined cost function, consisting both of the power consumption and the average delay, i.e.:

$$\text{Minimize } f = c_1 P_N + c_2 T_N \tag{4.2}$$

By tuning the parameters c_1 and c_2 in the cost function, the trade off between power consumption and delay can be adjusted. For example, the delay cost weight c_2 is variated for the case of previously examined randomly increasing power profiles. The power consumption and delay results versus the c_2 are plotted in Figures 4.13 and 4.14.

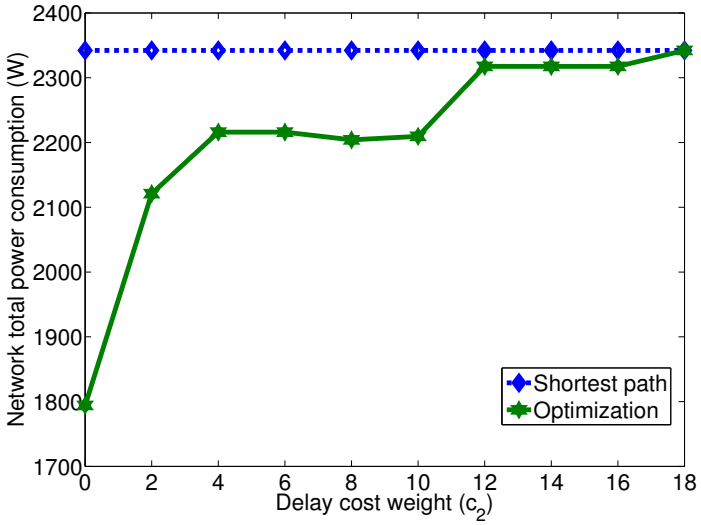


Figure 4.13: Power consumption: Optimization results for combined power and delay optimization cost. As the weight of the average delay c_2 increased, the power savings became smaller.

For the power only optimization ($c_2 = 0$) we observe 23% power savings and 20% delay increase. Though, as we increase the value of c_2, the savings drop gradually to 0% (Figure 4.13) and the delay increase can be moderated to 9%-1% (Figure 4.14). For large enough values of c_2 ($c_2 \geq 18$) the optimization results in the shortest path solution, with no power savings and is equivalent to delay-only optimization.

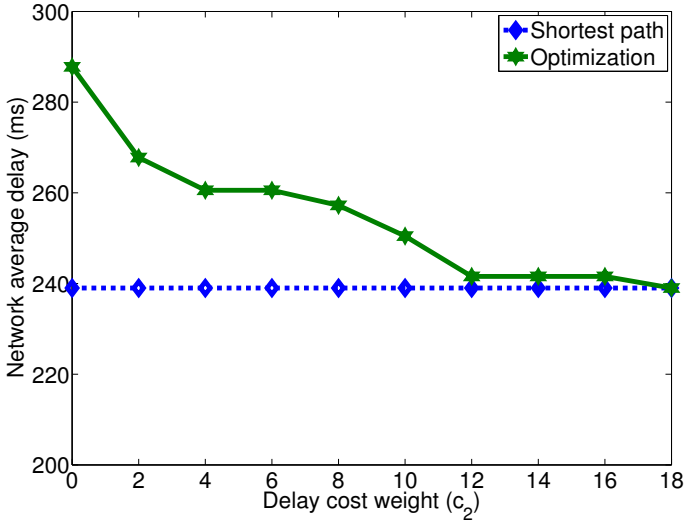


Figure 4.14: Average delay: Optimization results for combined power and delay optimization cost. As the weight of the average delay c_2 increased, the increase in delay was moderated.

Thus, the proposed combined optimization goal can be used to moderate the increase in average delay while still achieving significant power savings. The cost weights should be selected according to the relative values of the optimizing quantities and the requirements of each scenario.

QoS-differentiated goal

In the previous sections, the average end-to-end delay of a packet in the network was examined. Even though the average delay is a very simple and useful metric to evaluate the network performance, it does not reflect the specific behavior of the network to different users. In fact, different users would have different QoS requirements or constraints. In the presented model, this fact is anticipated as user traffic classes are included. Each user class can have different source-destination pair and QoS requirements. So, in the case of differentiated delay requirements, a cost function reflecting these needs will be more appropriate, as presented in Chapter 3:

$$\text{Minimize } f = c_1 P_N + \sum_{k \in \mathbf{U}} c_2(k) T_k \tag{4.3}$$

where $c_2(k)$ can be different for each one of the traffic classes.

The previous general case of randomly increasing power profiles is examined and a close look is taken to the delay results in Figure 4.15, which compares the shortest path

delay and the end-to-end delay per flow after the optimization. The delay of most of the flows is increased after the power optimization, while flow no.4 suffers the largest proportional increase, which is approximately 56%.

Assuming that flow no.4 is the only one sensitive to delay, the optimization is repeated with $c_2(4)$ varying from 0 to 10. The network power consumption results are presented in Figure 4.16a and the resulted end-to-end delay of flow no.4 in Figure 4.16b. As $c_2(4)$ increases, the increase in the delay of flow no.4 is moderated and finally for large enough value of $c_2(4)$ ($c_2(4) > 10$), flow no.4 follows its shortest path. Observing Figure 4.16a, for $c_2 = 10$ the optimization results still achieve significant energy savings of 11% (instead of 23% that was for power-only optimization).

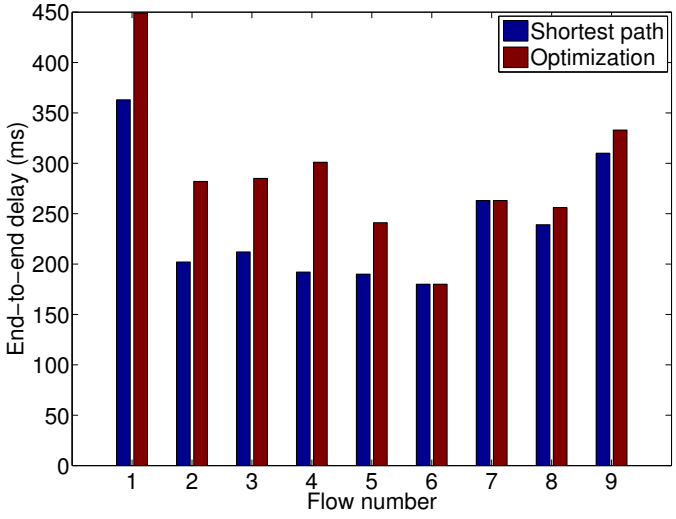
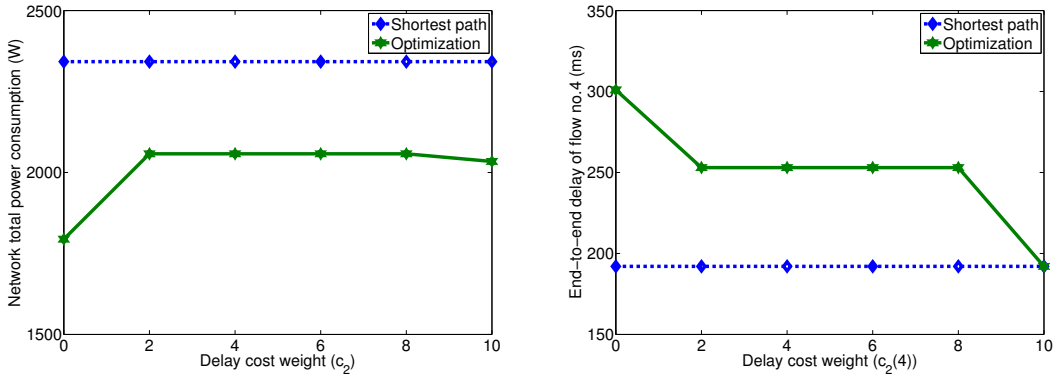


Figure 4.15: Per flow end-to-end delay for shortest path and power optimization results. Flow no.4 suffered the largest proportional increase, approximately 56%.

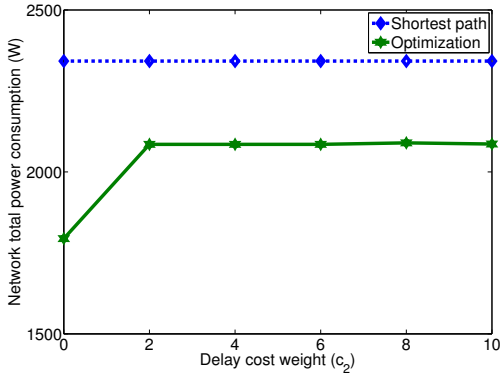
Similarly, the delay of more than one flows can be selected to be moderated, for example of both flow no.4 and flow no.3. The network power consumption results in Figure 4.16c show that the power savings are reduced to 11% as the weight of the delays of the two flows increases. For $c_2(k) = 10$ the average delay (Figure 4.16d) is increased by 17% relatively to the shortest path solution. Though, as shown in Figure 4.17 for $c_2(k) = 10$, both flows no.4 and no.3 have lowest possible end-to-end delay.

Note that, a cost function containing different QoS metric for each of the flows could also be used, as expressed in equation 3.50.

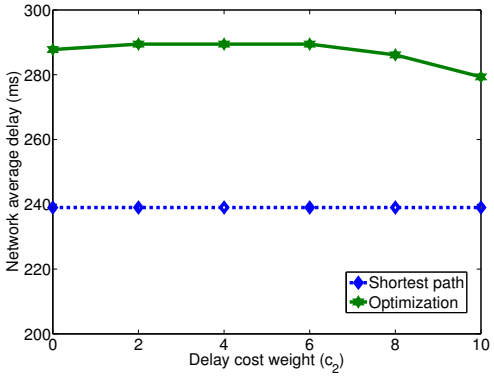


(a) Power consumption

(b) End-to-end delay of flow no.4



(c) Power consumption



(d) Network average delay

Figure 4.16: Optimization results with delay differentiated goal, where only flow no.4 was considered to be sensitive in delay (up), or both no.4 and no.3 (down). As the weight of the delays $c_2(4)$ and $c_2(3)$ increased, the increase in delay of these flows was moderated, while still achieving significant power savings.

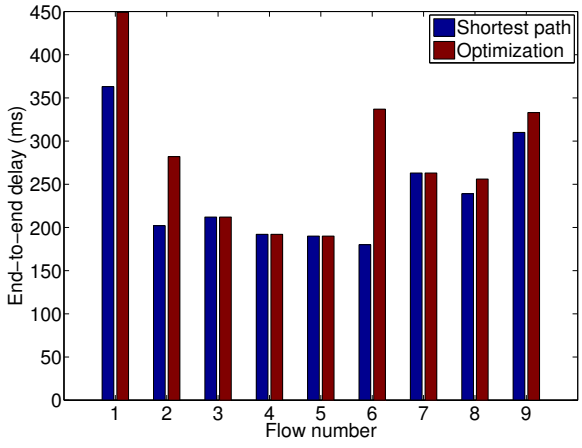


Figure 4.17: Per flow end-to-end delay for shortest path and optimization results with delay differentiated goal, where both no.4 and no.3 were considered to be sensitive in delay. The resulted delay of these flows is minimized.

Per-node power differentiated goal

Similarly, the cost function could include different weights of the power consumption of each one of the network nodes. Power consumed at specific nodes may induce a different cost, e.g. financial, environmental or other external cost. For example some nodes could be powered by renewable sources, thus it may not be required to lower their consumption. Others might have large cooling needs due to their specific hardware or local characteristics, thus the importance of their power consumption would be larger than others'. In such cases a differentiated optimization goal in terms of per-node power consumption could be useful. This would be expressed as a minimization of the following cost function

$$\text{Minimize } f = \sum_{i \in \mathbf{N}} c_1(i) P_i + c_2 T_N \tag{4.4}$$

where $c_1(i)$ can be different for each one of the nodes.

4.2.4 Discussion

To summarize, in this section different types of power profiles of network nodes are used in order to estimate potential energy savings by using the presented gradient descent energy efficient routing optimization. Moreover, it is demonstrated that a combined optimization goal, comprising both of power consumption and average delay, can be used to avoid excess increase in average delay and tune the relative weight of the two quantities in each scenario. As different users have different QoS needs, it is shown how the proposed model and optimization can be used to prioritize the QoS of more delay-sensitive flows while still achieving significant power savings.

The presented algorithm, as presented in the previous chapter, is of polynomial complexity ($O(N^3)$, in contrast with other mixed integer programming formulations of the network energy saving optimization problem that are NP-hard [6]. Though, as has been observed here, the gradient descent algorithm proposed requires numerous optimization steps until convergence. Although it could be important for estimation of potential savings, this optimization algorithm could be very time consuming for fast online calculation. Thus, in the next section we will present a faster heuristic and a decentralized mechanism.

4.3 Heuristic algorithms

4.3.1 Gradient descent based heuristic

In this section, a faster heuristic is proposed, based on the previously described gradient descent optimization algorithm 3.4.2. As the gradient optimization requires several steps to converge, a way to reduce computation time, i.e. an algorithm that could be useful for online application is presented. The idea here is that, instead of waiting for the optimization algorithm to converge, the partial derivatives can be used to guess a better routing choice. The variables in our model are the routing probabilities that range from 0 to 1 during the optimization and are forced to be 0 or 1 in the end of the optimization in order to keep traffic on a single path. Thus, according to our heuristic, after calculating the partial derivatives needed for the gradient descent, instead of making a gradient change to the routing matrices, the change that appears to be 'best' is found and the routing matrix is modified accordingly. More specifically, the algorithm follows the steps:

1. First initialize all the values $Q(i,k,j)$ and choose $\eta > 0$.
2. Solve U systems of the N equations (3.7)-(3.13) based on the current state to obtain the $q(i,k)$.
3. Solve U system of N equations (3.65) using the $q(i,k)$.
4. Find *minimum* of the $\frac{\partial f}{\partial Q(x,m,y)}$ (eq. 3.51). This gives the triplet (x',m',y'). If $min < 0$ then update as following

$$Q_{n+1}(x',m',y') = 1,\ Q_{n+1}(x',m',y) = 0 \text{ for } y \neq y' \tag{4.5}$$

   Thus, we change the routing decision at the node with the minimum negative partial derivative.
5. Repeat from step 3 until no change is made.

In this case, as the aim is a fast online solution, the algorithm is initialized with the shortest path values of the routing decisions, so that even from the first step of the algorithm the result achieved will be better than the initial state. The heuristic algorithm is run for the case of linear power profiles of Figure 4.2a and the results versus the algorithm steps are presented in Figures 4.18 and 4.19.

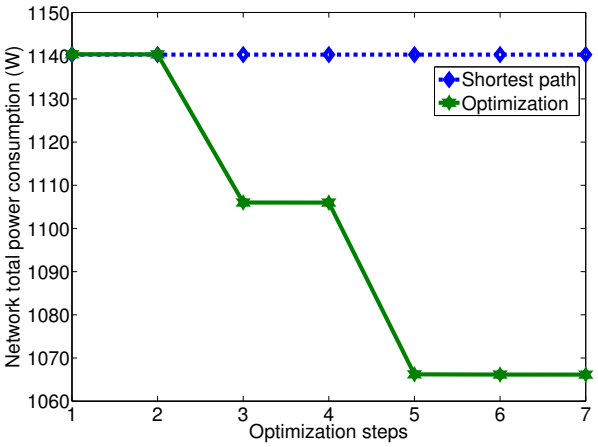


Figure 4.18: Power consumption: Power optimization results using the gradient-based heuristic, over the optimization steps, for linear power profiles. Power savings of 7% were observed in 7 steps, compared to 8% savings for the gradient descent optimization, which needed approximately 70 steps to converge.

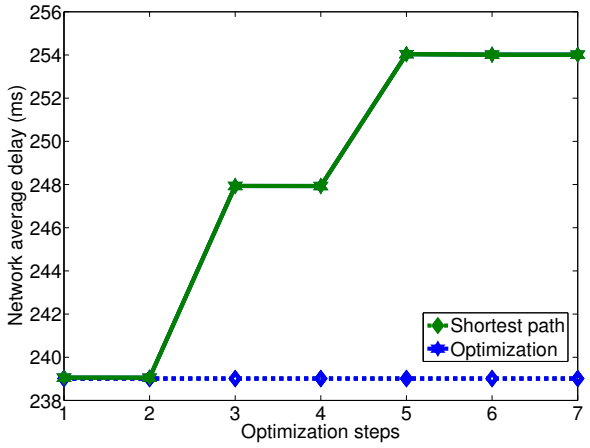


Figure 4.19: Average delay: Power optimization results using the gradient-based heuristic, over the optimization steps, for linear power profiles.

As can be observed the algorithm stops after only 7 steps with power savings 7%. Comparing with Figure 4.5, it can be seen that the the gradient descent optimization algorithm achieved 8% in approximately 70 steps in this case while now 7% now reached in only 7 steps.

The total input traffic is variated as before, scaling the input traffic matrix by a factor 0.3 to 3 times the initial value and the results of the heuristic algorithm are compared

to the gradient optimization and the shortest path case in Figure 4.20. As can be observed, for the linear profiles the results of the heuristic algorithm are very close to the results of the gradient optimization. The comparison is repeated for the case of the convex power profiles 4.2b and the randomly increasing power profiles 4.2c and the results are plotted in Figures 4.21 and 4.22 respectively. The heuristic algorithm is shown to have very good results for the case of the convex profiles. On the other hand, for the randomly increasing multi-step power profiles the results are not so close to the gradient optimization. This is expected as in this case the derivative of the power profile can be changing abruptly, thus the estimation of step 4 is not so accurate. Still, the heuristic achieves at least 4% savings in all cases.

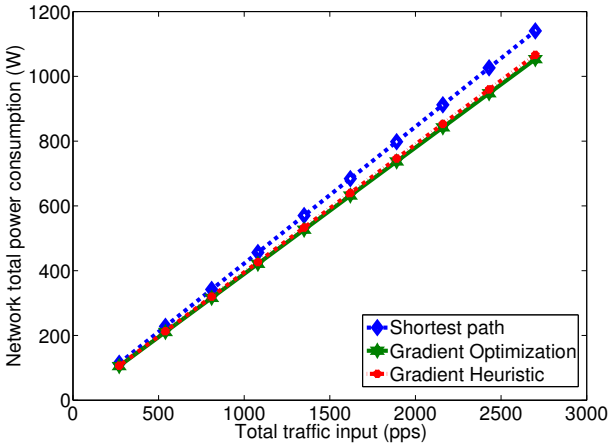


Figure 4.20: Power consumption: Optimization results comparing gradient descent and gradient based heuristic to shortest path, for linear power profiles

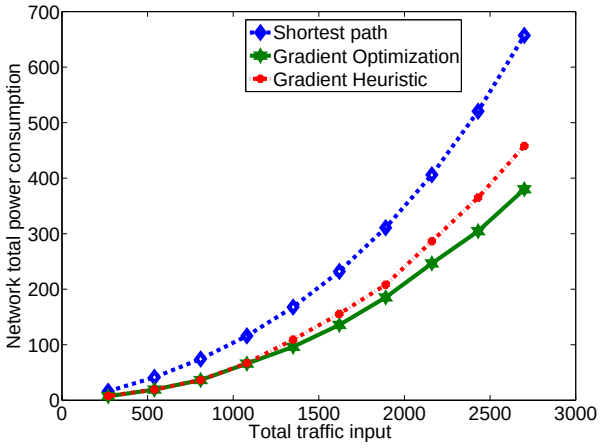


Figure 4.21: Power consumption: Optimization results comparing gradient descent and gradient based heuristic to shortest path, for convex power profiles

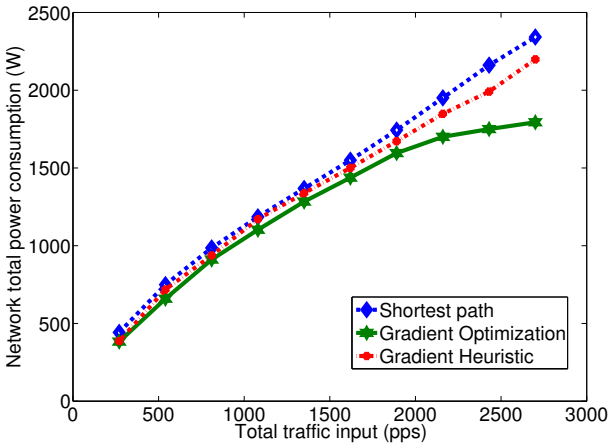


Figure 4.22: Power consumption: Optimization results comparing gradient descent and gradient based heuristic to shortest path, for randomly increasing power profiles

To summarize, the resulted average savings are gathered in Figure 4.23. It is observed that in all cases significant power savings can be achieved through energy efficient routing control. Convex power profiles offer the largest space for power savings. Moreover, the heuristic results are close to the optimization ones. Even in the most difficult case of multi-step power profiles the heuristic achieves 5% savings on average.

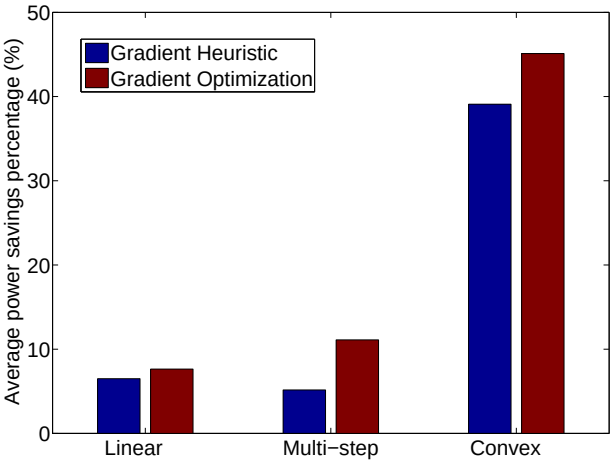


Figure 4.23: Power consumption savings relative to the shortest path routing for heuristic and power optimization results and each case of power profiles. Highest savings were observed for convex power profiles, but significant savings were achieved in all cases. The gradient-based heuristic performed well for convex and linear profiles, and fairly worse for multi-step profiles.

4.3.2 Distributed decisions

Another way to reduce complexity than reducing the number of optimization steps, is to try to reduce the size of the matrices used. To this end, in this section the case of distributed decisions at each source node is examined. It is assumed that the source knows a set of paths $\mathbf{\Pi}_k^*$ to its destination. The nodes that are included in these paths form the subnetwork N^* with routers R^* and links L^* such as $N^* = R^* \cup L^*$. So, the size of the above presented equations is reduced to the size of this subnetwork N^*. It is also assumed that each source can collect information about the traffic carried by the nodes and links of its subnetwork.

The cost function of each subnetwork N* will be given by

$$f^* = c_1 \sum_{r \in \mathbf{R^*}} P_r(\Lambda(r)) + c_2 \frac{\lambda(k)}{\Lambda_T^+} \left[\sum_{r \in \mathbf{R^*}} \pi(r,k) \frac{N_q(r,k)}{\Lambda_R(r,k)} + \sum_{l \in \mathbf{L^*}} \pi(l,k) \Big(\frac{N_q(l)}{\Lambda(l)} + d(l) \Big) \right] \tag{4.6}$$

The first term accounts for the power consumption of the subnetwork while the second term considers the delay of flow k^*, weighted by the flow's relative importance. Note that in the quantities $\Lambda(r)$ and $\Lambda(l)$ the traffic of other flows in the subnetwork should be also taken into account as background traffic.

The case where each flow optimizes the cost function of its subnetwork is examined, using equation 4.6. Thus, for each flow an optimization is run independently from others to optimize f^*. This optimization should be done sequentially, so a mechanism should be responsible for giving the right to one flow per time to run the optimization. As noted, the values of the traffic that belong to other flows and are carried by this subnetwork are regarded as constant background traffic.

For comparison, the gradient descent optimization when optimizing only power is run, thus $c_2 = 0$ in 4.6. The results are presented only for the randomly increasing power profiles, which is the most difficult case and the total input traffic is variated as before, scaling the input traffic matrix by a factor 0.3 to 3 times the initial value. The results of the sequential optimization of each source are compared to the centralized gradient descent algorithm and the shortest path solution in Figure 4.24. As can be observed, the savings in this case are very close to the centralized results. For this experiment it was assumed that $|\mathbf{\Pi}_k^*| = 5$ alternative paths are known in advance.

However, the efficiency of this algorithm depends on the number of known alternative paths for each source. Thus, the average power savings for the examined cases of varying traffic input are compared, for 2, 5, 10 alternative known paths as well as for the centralized case (Figure 4.25).

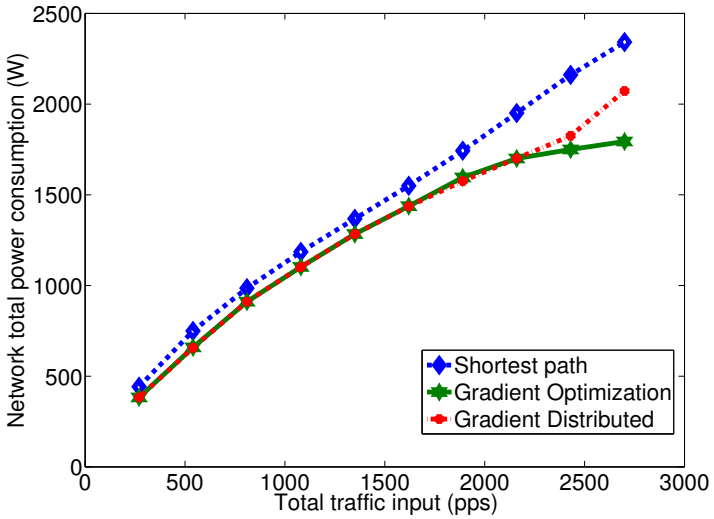


Figure 4.24: Power consumption: Optimization results over increasing traffic input, comparing centralized and distributed gradient descent to shortest path, for randomly increasing power profiles.

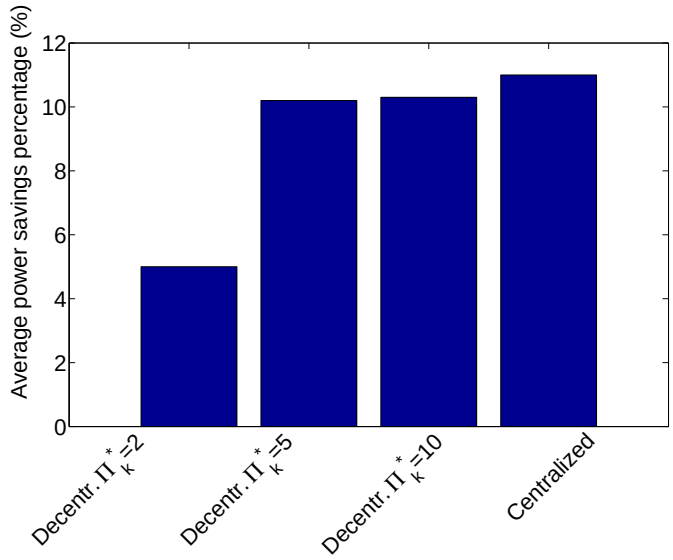


Figure 4.25: Average power savings for different number of initial known alternative paths of the decentralized algorithm and for the centralized case.

In case of 2 alternative paths, the savings are approximately 5%, while we can observe that the savings when we increase the alternative paths from 5 to 10 are negligible.

In Table 4.1 we report the average subnetwork sizes for these cases. Interestingly, for the case of 5 alternative paths where the power savings of the algorithm were very close

to the centralized algorithm results, the size of the equations is remarkably reduced comparing to the centralized case. Note that the optimization could result in any of these 5 initial paths, or any combination of these within the subnetwork.

To summarize, a decentralized version of the gradient descent algorithm is proposed, where each source optimizes its routing within a subnetwork. Each source should have some information about alternative paths leading to its destination, in order to build its routing matrix within this subnetwork and collect information for carried traffic on the subnetwork nodes. The optimization should be run sequentially for all sources to avoid conflicting decisions and a mechanism should be able to give the right to each source. It has been shown that the results are very close to the centralized case and the size of the equations is dramatically reduced, which would improve significantly the computation time.

Table 4.1: Size of the equations for centralized and decentralized cases

	R^*	L^*	N^*
Centralized	29	78	107
Decentralized-$\lvert\mathbf{\Pi}_k^*\rvert = 10$	17	25	42
Decentralized-$\lvert\mathbf{\Pi}_k^*\rvert = 5$	12	15	27
Decentralized-$\lvert\mathbf{\Pi}_k^*\rvert = 2$	9	9	18

4.4 Experimental results

4.4.1 Improving upon shortest-path routing

In this section, the power optimization is applied on our 23 node laboratory testbed, shown in Figure 4.26. The testbed comprises of PC-based routers and allows monitoring of the power consumption and delay. The service rate of the links are their 100 Mbps speeds and virtual propagation delays have been added to service times so as to introduce more realistic values of delay; indeed the very short physical links used in the laboratory do not provide realistic values of network delays so that we have to delay each packet in software at the nodes so as to arrive to a more realistic value. The power consumption profiles of the fourteen nodes located on the circle have been measured with a power meter and are approximated with polynomial fitting to the data (Figure 4.27). It can be observed that these PC-based routers have small power variation with respect to the carried traffic. As they are old technology energy-agnostic devices they have a large static power consumption even when carrying no traffic. They offer small room for energy savings thus represent the worst case of our case studies.

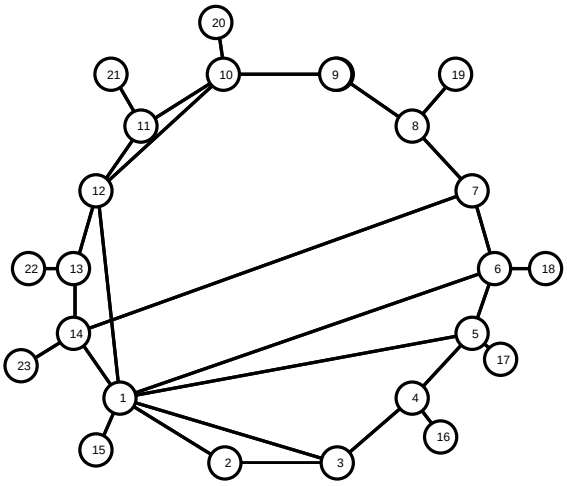


Figure 4.26: Experimental PC-based laboratory network topology. End nodes are used as source-destination pairs and the power profiles of the nodes on the circle have been measured using a power meter.

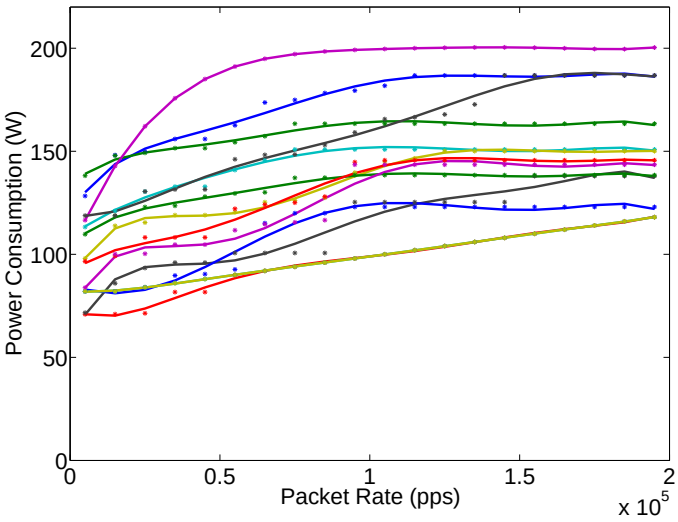
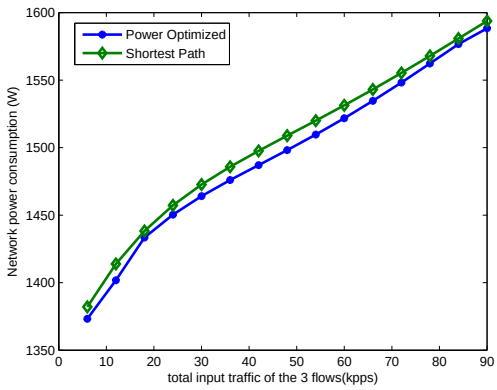


Figure 4.27: Measured power consumption profiles of 14 nodes positioned on the circle.

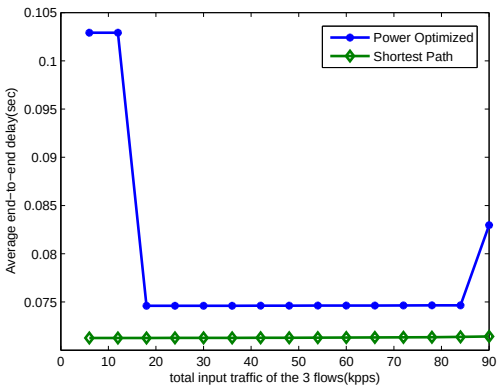
The comparisons are carried out in the presence of flows traveling from source to destination with average traffic rates: Flow 1 (22,18) traffic rate 30kpps, Flow 2 (23,19) traffic rate 10kpps, Flow 3 (21,17) traffic rate 20kpps. First, we apply the optimization algorithm to the network and we focus on power ($c_2 = 0$). As expected, as these devices show a very small power variation with respect to the carried traffic, the optimization

yields a saving of 10 Watts, down from 1531 watts, at the cost of an increase in average end-to-end delay of 3.3ms.

Then we vary the input traffic of the 3 flows from 0.1 to 1.5 times their initial value and present the results in Figure 4.28a. The power optimization achieves a modest average power saving of 8.2 Watts. The average packet delay, shown in Figure 4.28a is in most cases slightly increased, though in the first two and the last experiments, the increase is significant. If we opt for both power and delay optimization by adjusting c_2 in the cost function (4.2) we can avoid the increases in average delay seen in Figure 4.29 and the average power savings are decreased to 6.4 Watts.

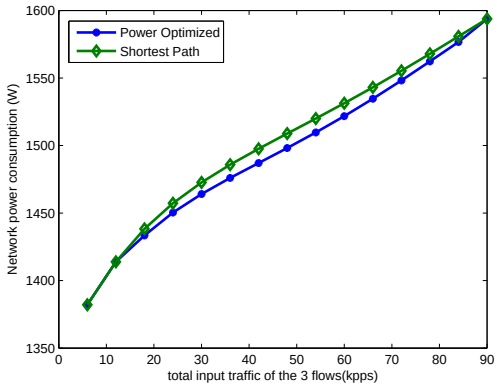


(a) Network power consumption

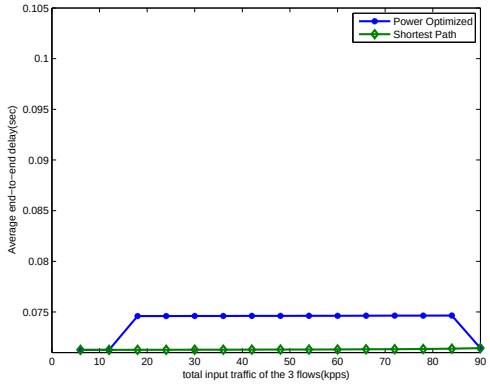


(b) Average delay

Figure 4.28: Power consumption and average end-to-end packet delay against varying traffic load in kpps for power-optimized versus shortest path routing



(c) Network power consumption



(d) Average delay

Figure 4.29: Power consumption and average end-to-end packet delay against varying traffic load in kpps for power-and-delay-optimized versus shortest path routing

These results indicate that the energy aware routing optimization can achieve modest savings even in the small testbed with only 3 flows present and in the worst case of the power profiles of Figure 4.27 that are energy-agnostic, with a negligible degradation in delay. It is expected that as more flows are added in the network the savings will become more significant. For example, if we add another flow Flow 4 (20,16) at traffic rate 5kpps, and vary the traffic of the four flows from 0.1 to 1.5 times their nominal values, the average power savings increase to 15 Watts as shown in Figure 4.30, which corresponds to approximately 1% decrease.

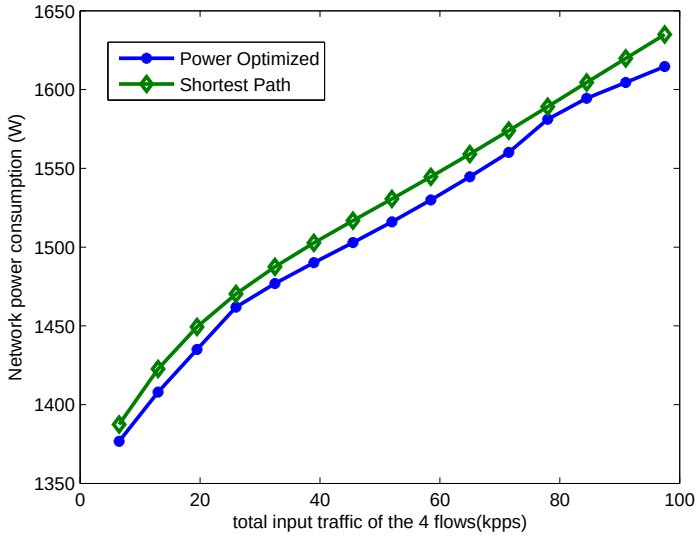


Figure 4.30: Power consumption for proposed algorithm compared to shortest path with four flows and power optimization

4.4.2 Improving upon CPN

In this section, we utilize the cognitive packet network (CPN) which is a self-adaptive network protocol that uses random neural networks in order to discover better paths [44]. CPN uses 'smart' packets to discover paths in the network that are efficient with respect to the required metric. The full paths are reported back to the source, which are then used to route the information packets. Here, we use log files available from experiments with the topology and set of flows described in the previous section with EARP [36]. EARP is the energy-aware version of CPN, which uses CPN to search for the paths that minimize power consumption.

In our experiments, we assume no prior knowledge of the network topology but rather we use the log files of observed paths to build the routing matrices. Specifically, every time a new path is reported back to the source, the routing matrices of the flows are

updated and the algorithm is initialized with the current routing state of the network. In this experiment we use the heuristic version of the gradient descent based algorithm presented in Section 4.3.1 that can be used online to provide a fast one step decrement of the power consumption.

The results comparing to the EARP are shown in Figure 4.31 where the dashed lines represent the average values. It can be observed that each time EARP uses different paths for the carried flows, the power consumption changes. Our algorithm is initialized with the routing matrix selected by EARP and one step optimization was used, that is why the results follow the jitter of the EARP. Though, we observe a significant average power saving with respect to EARP.

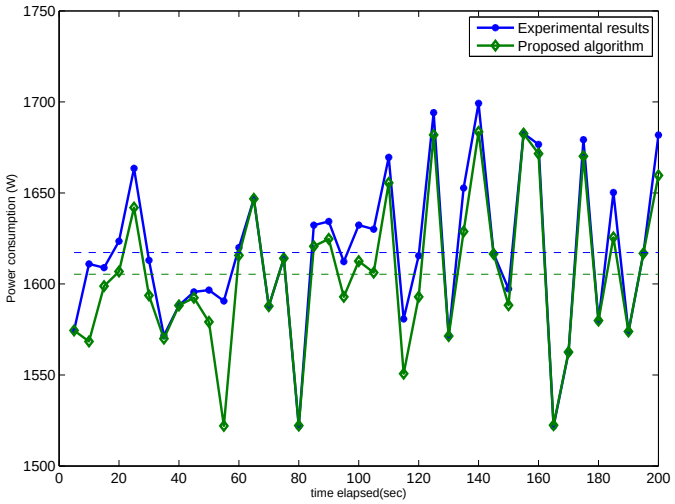


Figure 4.31: Power consumption for the gradient heuristic algorithm (green) compared to power-based EARP [36] initialized with EARP

In this case, EARP makes very frequent changes in the routing matrices and our heuristic algorithm is always initialized at that state. Though, in order to limit out of order packet arrival and jitter, the algorithm could be initialized at its previous state and thus change paths only when the path modification improves. We then have the greater power savings of Figure 4.32.

To summarize, in this section the usage of energy aware routing is demonstrated in a laboratory testbed with real measured power profiles. It has been observed, that in this case of old-technology energy agnostic devices, only small power savings can be achieved. As the power characteristics of real devices are changing, the importance of energy efficient routing will become crucial. Moreover, the fast heuristic algorithm

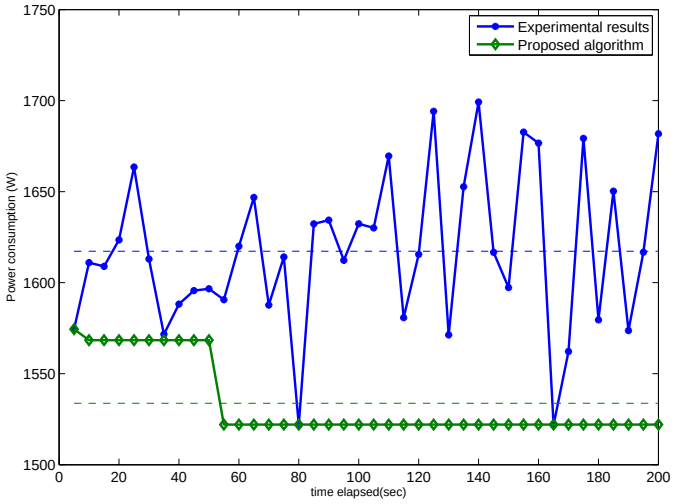


Figure 4.32: Power consumption for the gradient heuristic algorithm (green) compared to power-based EARP [36] initialized with previous state

has been used in conjunction EARP. It has been observed observed that, using the fast changing routing matrix, the gradient heuristic can be used to provide an one step change that improves power consumption.

4.5 Conclusions

In this chapter, the gradient descent optimization algorithm presented in Chapter 3 is evaluated for different network power consumption characteristics. In each case, it is shown that a significant amount of energy can be saved by just changing the routing decisions, without disrupting service. As expected, the routing changes might result in longer or more congested paths, which might add unacceptable delay to the traffic. To deal with that restriction, the combined optimization goal function proposed, consisting of both power consumption and propagation as well as queuing delay, is shown to restrict the delays imposed while still achieving significant savings. The case studies examined were chosen to showcase potential savings depending on power consumption characteristics and different optimization goals. As our model is generalized, this analysis can be easily extended to other cases to include different power consumption behaviors, power consumption of links, the effect of control traffic etc.

As the complexity of the gradient descent algorithm would be large for online application, a faster heuristic is proposed and compared to the gradient descent optimization results. Moreover, the case of distributing the optimization to subnetworks is also pro-

posed and examined. Both simplifications are shown to achieve good performance.

In the last section, we have tested a small laboratory testbed with PC-based nodes and measured power profiles. As the power profiles of these nodes are energy-agnostic, only small savings can be achieved. However, the research undertaken has demonstrated that our proposed heuristic can be used to apply a fast one step improvement in power consumption in a fast changing environment.

5 Investigating energy - QoS tradeoff

5.1 Introduction

As discussed in Section 2.4, several research works are based on the assumption that the networking devices are equipped with a sleep mode, or similar capability of low-power-when-idle enabled. However, current networking devices do not have these capabilities and it is still unclear as to what extent future devices will enable low power modes and their specific characteristics. Even though the decision to shut down machines may appear as an effective solution, an aggressive approach which would shut down all unused equipment is not only infeasible with the current equipment, but could also lead to major degradation of the performance. Since previous work ignore such implementation obstacles, the objective here is to experimentally display the effect that deep sleep based power saving technologies have on the quality of service (QoS), and more specifically, the tradeoff between power consumption, network delay and packet loss.

In this chapter, using a PC-based laboratory testbed in which nodes have the capability to suspend their operation and be waken up remotely, the effect of sleep modes on QoS is experimentally demonstrated. In contrast to other work [17], the intention is not to measure one single device, but rather a whole network and thus the effect on the total power consumption and end-to-end delays and packet losses. More specifically, the aim is to explore the drawbacks of introducing sleep modes in the network. Moreover, by presenting graphically the tradeoff relationship between power consumption and QoS, we propose a way to identify on a high-level the appropriate operating state of the network based on the specific requirements.

In the sequel, first the setting of the experiments and the additional functionalities that were built in the network in order to enable the exploitation of the sleep modes are described (Section 5.2). In Section 5.3 the power consumption savings and the effect on network delay and packet loss are displayed for different parameters. The tradeoff curves are presented and discussed. Finally, in 5.4 the conclusions are drawn and challenges are highlighted.

5.2 Setting

The experiments are conducted on a 29-node PC-based laboratory testbed, set up in a topology that mimics the Czech Republics National Research and Education Network (CESNET)[2], as shown in Figure 5.1. Grey nodes are sources/ destinations and are always on, while green intermediate nodes can be put into a low-power sleep mode and their power consumption is being measured using the Watts up?.Net power meter [3]. The testbed nodes run Ubuntu Linux 2.6.32-25 and have the ACPI S3 suspend-to-RAM state [1], which is a low wake-latency sleep state, where only the RAM remains powered. Also, their BIOS is configured to accept wake-on-lan packets, so that they may be remotely put to sleep and woken up.

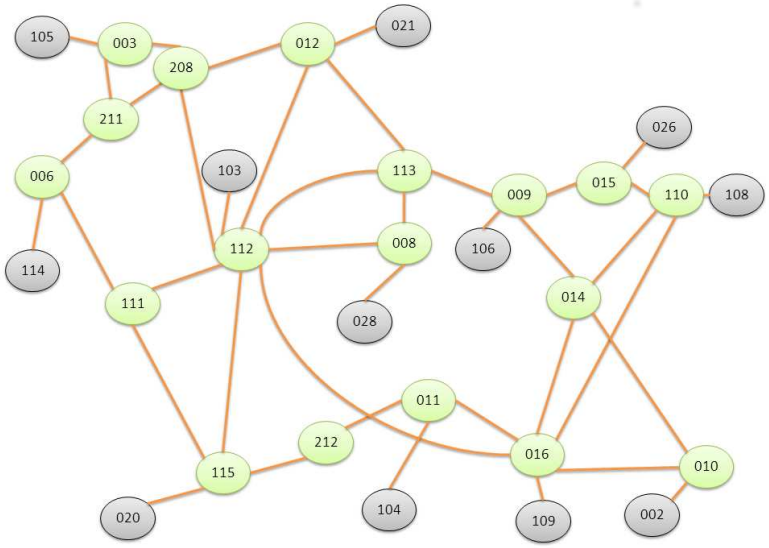


Figure 5.1: Experimental testbed consisting of 29 PC-based routers connected in the CESNET topology [2]. Green nodes can be put to sleep and their power consumption is being measured using a power meter. Grey nodes are sources/destinations and are always ON

Queue and neighbor discovery signaling

In this section, the additional functionalities that were built in the network in order to enable the exploitation of the sleep modes are described. The first challenge that needs to be faced, is that a node has to be notified when a neighbor node is in sleep mode. Moreover, packets which need to use a node, while this node is in sleep mode, will have to be stored, until the node becomes again fully operational. The mechanism built to

face these challenges consists of the following modules:

- An enhanced neighbor-discovery signaling mechanism for rapidly establishing whether a node's immediate neighbors are on or off,
- A queue for storing packets at the previous hop if a node is suspended.

The neighbor discovery signaling mechanism runs in each node and is implemented by exchanging frequent "hello" packets between neighbors. This enables fast identification when the neighbor becomes unavailable by entering the energy-saving sleep mode. Therefore this signaling mechanism is responsible for sending and receiving hello messages, keeping a state table of all its neighbors and relevant time stamps and add/remove a neighbor in the list accordingly. It also informs the queuing mechanism for the addition of a new neighbor, so that the packets waiting can be sent to their destination.

The queuing mechanism on the other hand, is responsible to create a queue as soon as it is identified that one of its neighbors are unavailable. Packets of the same next-hop are queued until this neighbor becomes available again. As soon as the next hop wakes up, the packets in the corresponding queue are released towards their destination. At each node, the queuing mechanism includes the steps shown in Figure 5.2:

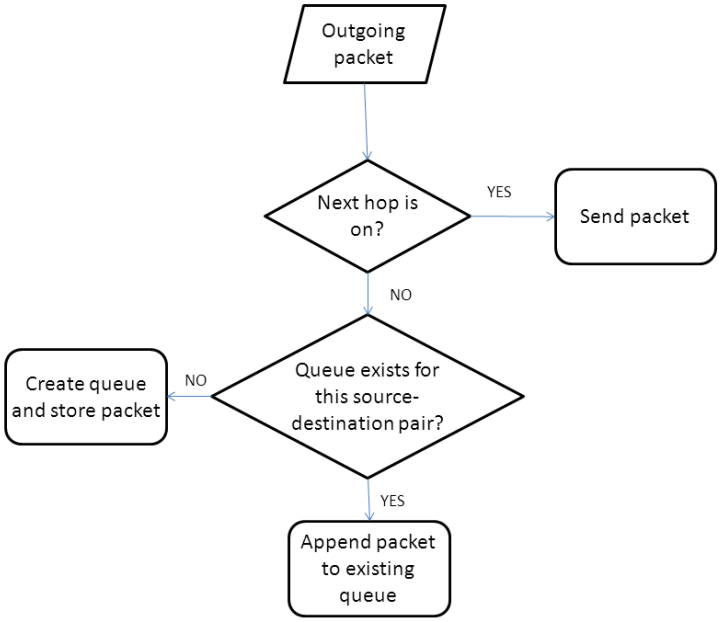


Figure 5.2: Flow chart of the queuing mechanism built in each node in order to enable the exploitation of sleep modes

Note that a node that has packets in such queues may also enter sleep mode and the packets in the queues shouldn't be lost. This is addressed by using the 'suspend' capability, with which memory is the only remaining function that consumes energy, so as to avoid losing the packets in the queues.

Also note that a predefined shortest-path routing is used and it is not attempted to change paths when a node is off. This is because the objective is to study the effect of turning nodes off in a network regardless of the routing protocols and available alternative paths of the specific source and destination pairs. This also avoids cases such as the overloading of alternative paths and additional packet losses.

5.3 Experiments

The experimental setting that is used including the mechanisms described involves

- Nine users, each corresponding to a source-destination (S-D) pair of continuous Poisson UDP traffic of 50 Kbps, to avoid congestion situations. Note that these pairs of source-destinations correspond to gray nodes in Figure 5.1 and were selected to cover the whole network when their traffic is routed simultaneously.

- The sources and destinations are always kept on, while intermediate nodes are controlled by an automated mechanism which decides when to suspend a node and turn it back on again. Each node is controlled independently and ssh commands are sent to suspend the node, while wake-on-lan messages are sent to awake the node from the suspend state.

- During each experiment the power consumption of the intermediate nodes, the packet loss and the average end-to-end delay experienced by the packets are being measured. Each experiment lasted 200 seconds, which corresponds to a reasonably long period in order to observe the average results. Note that in the course of the experiments, turning off the nodes of the experimental testbed often resulted in machine failures. As the total times the nodes turned off during each experiment is very high (see Figure 5.6), each of the final experiments presented here was conducted only once. The average values presented are the average over time for each experiment.

Experiments are run with the auto-hibernation mechanism following a random pattern of turning off and on the nodes. The duration that any node stays on T_{on} and the duration that it stays in sleep mode T_{off} are selected randomly between two values T_{on}^{min},T_{on}^{max} and T_{off}^{min},T_{off}^{max} respectively. The average cycle time $T_{on}^{avg} + T_{off}^{avg} = T_{cycle}$

is kept constant for comparison in this section and is $T_{cycle} = 25$. The results of seven experiments with the parameters are reported in Table 5.1. During each experiment, the power consumption of the intermediate nodes, the packet loss and the average end-to-end delay experienced by the packets were measured.

Table 5.1: Parameters used in set of experiment with $T_{cycle} = 25$

	T_{on}^{min}	T_{on}^{max}	T_{off}^{min}	T_{off}^{max}
experiment 1	the nodes are always off			
experiment 2	5	10	5	30
experiment 3	5	15	5	25
experiment 3	5	20	5	20
experiment 5	5	25	5	15
experiment 6	5	30	5	10
experiment 7	the nodes are always on			

In the first experiment the nodes are kept always on, while in the last experiment the nodes are always off and are turned on at the 200-th second. The minimum time that a node remains off (T_{off}^{min}) was chosen to be 5 seconds because this was the time it takes for a node to switch from normal operation into the suspend state.

In Figure 5.3 , the sixth experiment is examined in detail and the measured power consumption over time is plotted. At the beginning of this experiment and after the 200-th second all nodes are on, while during the experiment nodes turn off and on at random. The measured end-to-end delays for all source-destination pairs in this experiment are shown in Figure 5.4 in the form of a histogram. It can be seen that most packets face very small delays, while as expected, there are some packets that face large delays due to being queued up in the network nodes when nodes on their path are in sleep mode. Thus, the power savings observed in this case (Figure 5.3) come at the expense of increased delay for some packets, while the majority of packets do not face any degradation in service.

5.3.1 Effect of sleep modes on packet loss

Let R_{on} be the proportion of cycle time that nodes are on, which is

$$R_{on} = \frac{T_{on}^{avg}}{T_{cycle}} \tag{5.1}$$

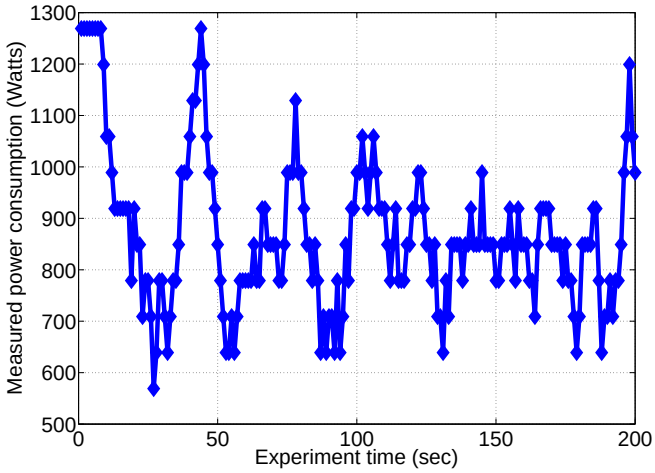


Figure 5.3: Network power consumption during experiment no.6 where $R_{on} = 0.7$ and $T_{cycle} = 25$. Power consumption fluctuates over time as nodes are turned off and on.

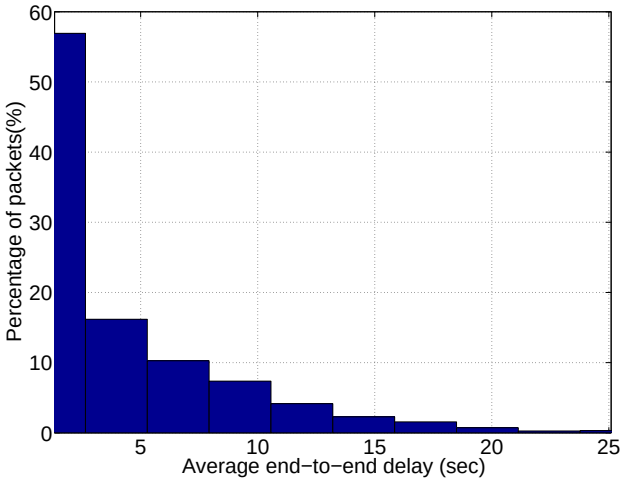
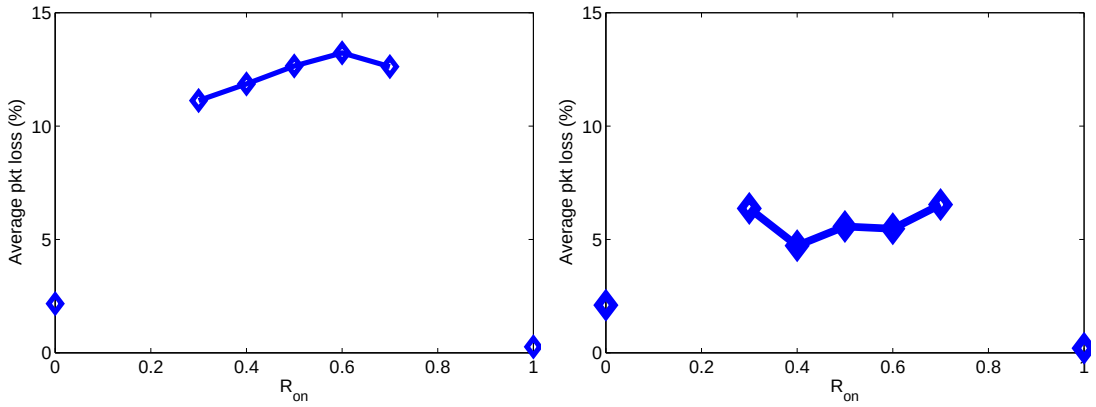


Figure 5.4: Histogram of delay measurements during experiment no.6. It can be observed that 58% of packets faced very small delays, while for the rest of the packets there was a significant degradation in performance due to encountering nodes in sleep mode.

where $T_{on}^{avg} = (T_{on}^{max} + T_{on}^{min})/2$ is the average time the nodes are on. All the results will be presented versus R_{on}, unless otherwise stated. First, the impact of sleep modes on packet loss is examined, which is mainly due to the packets lost during the intervals between the instant that a neighbor goes to sleep and the time that it discovers that the neighbor is sleeping (no hello-messages have been received during the time-out interval),

after which it will start queuing the packets; packets may also be lost while the neighbor is entering the suspend state and is still replying to the hello messages but does not process the incoming traffic.

In Figure 5.5a the packet loss is plotted versus R_{on}. For the purposes of analysis, only experiments 2-6 are taken into consideration and the two extremes where the nodes are always off (experiment 1) or always on (experiment 7) are excluded. It can be observed that for the experiments no.2-6, the packet loss is relatively stable, approximately 12%. This is expected, since the traffic is sent out at a low rate to avoid network congestion and therefore packet loss is mainly due to the time it takes for a node to discover that a neighbour is asleep and start queuing the outgoing packets. Thus, packet loss seems to depend on the frequency of the hello messages and the timeout interval set. For comparison reasons, the experiments were repeated with $T_{cycle} = 50s$ and the same values of R_{on} (details described in section 5.3.3) and as can be observed in Figure 5.5b the packet loss is again relatively stable, at approximately 6%. This suggests that indeed the packet loss depends on the time it takes for a node to realise that a neighbour is asleep and is stable for the same T_{cycle}, while for larger T_{cycle} the packet loss is smaller as the total number of times the nodes are turned off are fewer. For the experiments, the hello messages between neighbours were sent every 1s and the timeout was double that value, so as to avoid falsely identifying a neighbour as asleep and creating unnecessary queues. The packet losses could be further reduced if the nodes signaled their neighbours just before they entered a sleep state.



(a) $T_{cycle} = 25s$ (b) $T_{cycle} = 50s$

Figure 5.5: Network average packet loss versus average on rate R_{on}, for $T_{cycle} = 25s$ and $T_{cycle} = 50s$. Packet loss is relatively stable for the same T_{cycle} value as it depends on the total times the nodes were turned off during each experiment. For smaller T_{cycle}, more turn offs are happening for the same time period, thus smaller packet losses are observed.

Figure 5.6 shows the average number of times that the nodes are turned on/off in all experiments. The intermediate nodes turn on/off approximately 125 times during each experiment, with the exception of experiments 1 and 7 where either all nodes are constantly on or nodes are kept off until the end of the experiment. This was expected since the T_{cycle} is kept constant. This allows consistency and enables the comparison of the QoS values of the difference experimental cases. Moreover, it justifies the relatively stable values of the packet losses mentioned before.

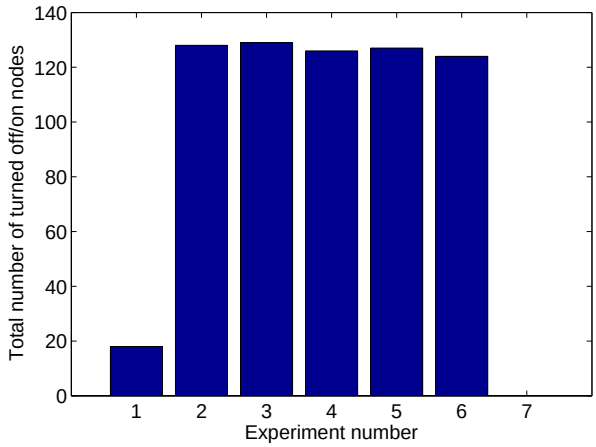


Figure 5.6: Total number of times that nodes were turned off/on during each experiment for $T_{cycle} = 50s$. For all experiments approximately the same number of turn offs is observed, with the exception of experiments 1 and 7 where either nodes are kept off until the end of the experiment or all nodes are constantly on. This was expected since the T_{cycle} is kept constant.

5.3.2 Power consumption against delay

The average value of the measured power against the R_{on} for all experiments is shown in Figure 5.7 (blue solid line). As can be observed, power consumption increases almost proportionally to R_{on}. Since R_{on} is the proportion of time that nodes stay on, in order to save energy (smaller power consumption) R_{on} needs to be as small as possible. Obviously this comes at the price of a performance degradation since if nodes stay on for less time (smaller R_{on}) the packet delay increases. This can be observed from the average delay curve in the same figure (green dotted line).

Another way of presenting this relationship is by plotting the average measured value of power against the observed average packet delay for each experiment, shown in Figure 5.8. A clear tradeoff can be observed between the power savings that can be achieved and the degradation in QoS that packets may experience presented in terms of delay.

Power consumption is reduced by up to 88%, at the expense of increased delay which can be unacceptable for some of the traffic in the network. Through this relationship, one can choose the appropriate operating state of the network that offers the desired power savings and QoS constraints. For instance, in Experiment No. 6 ($R_{on} = 0.7$) a 35% savings is measured in network power consumption while the end-to-end packet delay shown in Figure 5.4, is small for the most of the traffic.

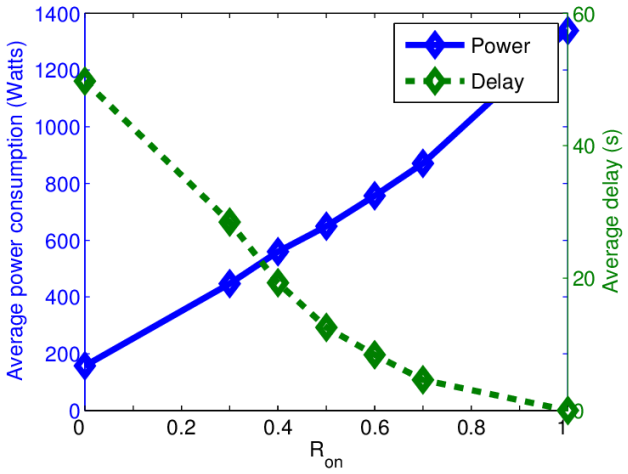


Figure 5.7: Average power consumption and average delay versus R_{on}. Power consumption increases almost proportionally to R_{on} while the effect on average delay is reverse.

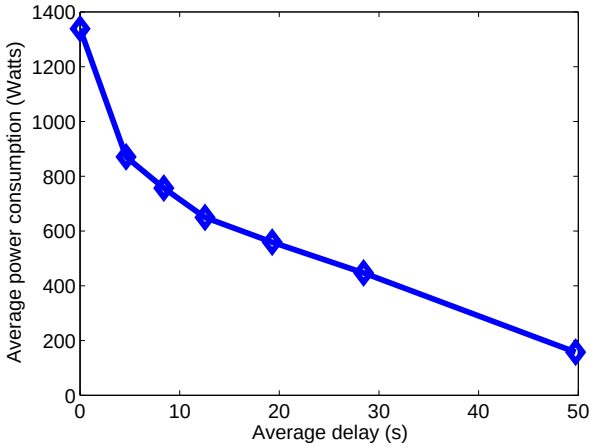


Figure 5.8: Graphical representation of the observed tradeoff between average power consumption and average delay $T_{cycle} = 25$. Each point corresponds to a different value of R_{on}.

5.3.3 The impact of the on/off cycle

To investigate the effect that T_{cycle} has on the QoS of the packets the experiments are repeated with $T_{cycle} = 50s$ and the same values of R_{on}. The parameters of this second set of seven experiments are shown in Table 5.2.

Table 5.2: Parameters used in set of experiment with $T_{cycle} = 50s$

	T_{on}^{min} (s)	T_{on}^{max} (s)	T_{off}^{min} (s)	T_{off}^{max} (s)
experiment 1	the nodes are always OFF			
experiment 2	5	25	5	65
experiment 3	5	35	5	55
experiment 3	5	45	5	45
experiment 5	5	55	5	35
experiment 6	5	65	5	25
experiment 7	the nodes are always ON			

Figure 5.9 shows the measured power consumption versus the measured end-to-end delays for $T_{cycle} = 25s$ and $T_{cycle} = 50s$. In this figure, the system behavior and the relative tradeoff between power consumption, delay and packet loss can be observed. Comparing between the two experiments it can be seen that for larger T_{cycle} the delay increases. Note that the results for packet loss were presented in Section 5.3.1 (Figure 5.5b), where the packet loss was relatively stable at approximately 6%, smaller than when $T_{cycle} = 25s$. Thus, small T_{cycle} values result in smaller delays but larger packet losses.

Through these curves, one can think about the required operating state of the network and have a rough estimation on how the energy savings could affect network delays and packet losses.

The results presented here correspond to the case of the specific PC-based routers which are not optimized for fast sleeping and waking up. The large observed delays, reveal the challenge for future network devices to have fast sleeping states in order to utilize mechanisms that turn off and wake up devices in the network, as have been previously proposed in literature.

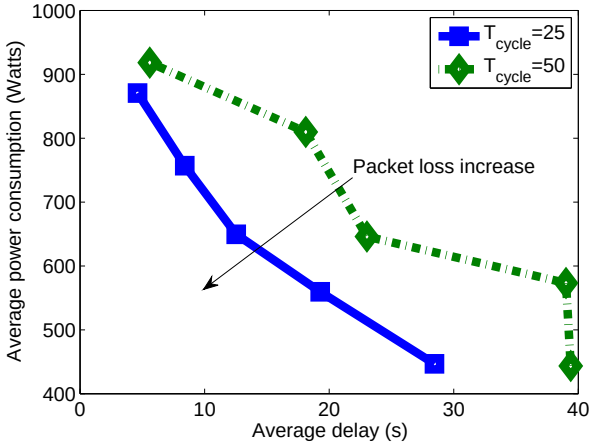


Figure 5.9: Graphical representation of the observed tradeoff between average power consumption and average delay $T_{cycle} = 25$ and $T_{cycle} = 50$. Smaller T_{cycle} resulted in lower average delay but larger packet loss.

5.4 Conclusions

In this chapter, the effect of the introduction of sleep modes on the energy consumption and QoS of a wired network is demonstrated. The intermediate nodes are put to sleep or are waken independently of their traffic using a separate control mechanism to suspend or wake up a node at random intervals with a specified time cycle. Experiments are reported for a PC-based laboratory testbed using a shortest path routing protocol.

This study showcases how the limitations of the hardware with sleep modes can tradeoff the power savings in terms of increased delays and packet losses. In such a scenario, the proposed way of representing tradeoff curves (Figure 5.9), dependent on the specific characteristics, can help decide the desired operating state of the network based on power consumption saving requirements and QoS constraints. These results can provide the basis for future modeling studies of similar systems to identify optimal operating points for energy efficiency and QoS.

6 Energy aware admission control

6.1 Introduction and background work

Admission control in wired networks has been traditionally used as a way to control traffic congestion and guarantee QoS [76]. This, usually, requires estimation of the level of QoS that a new user session will need and investigation of whether there are enough resources available to service that session without affecting the QoS of the existing users of the network [30, 69]. Thus, when a flow requests real-time service, the network needs to be able to characterize the requirements of the new flow and make an admission decision based on an estimation of its current and projected state. The metrics considered in the decision of whether to accept a new flow into a network are mainly bitrate, delay, packet loss and jitter [12, 37]. Admission control has never been used as a tool to restrict user entrance in a wired network in order to minimize energy consumption. However, the concept of admitting users according to their power consumption has been used in wireless networks where flows are accepted based on the estimated residual energy or the transmit power of the nodes along a routing path [27, 28].

In this chapter, an energy efficient admission control for wired networks is proposed. In contrast to previously proposed admission control mechanisms for congestion control, in this case it is not desirable to reject users from entering the network. The objective here is to delay - if possible - the admission of flows in the network and admit them at a later, more energy efficient, time. The challenge entailed in this case is the unpredictability of the future states of the network and thus, the uncertainty on whether a more energy efficient condition actually exists in the future or not.

Thus, the idea of energy aware admission control implies the shaping of the input traffic rate so that the network 'reacts' in a more energy efficient way, instead of trying to change the routing as has been proposed in Chapters 3,4. Firstly, the problem is defined in Section 6.2. In the sequel, two different scenarios are examined and experimentally evaluated. In the first one (Section 6.3), a nonlinear power profile of the nodes is assumed, where the operational state of the node is automatically adjusted according to the carried traffic rate. When a new user is expected to increase the power consumption of the network significantly, this user is not serviced until its maximum acceptable time

expires or until a more energy-efficient time to admit it is found. In the second scenario (Section 6.4), a small testbed with on/off capabilities is used and its power consumption is being measured during the experiments. The decision to delay a user is taken every time the admission of the new user would require the wake up of a node, which would lead to additional power consumption. The chapter concludes with a discussion of the results on Section 6.5.

6.2 Problem definition

The problem of energy aware admission control can be formulated as the minimization of the energy consumption of the network, while respecting the acceptable maximum waiting times set by the users :

$$\begin{aligned} &\text{Minimize} \quad E_N = \int P_N dt \\ &\text{subject to} \quad w_i \leq W_i \end{aligned}$$

where $P_N = \sum_{i \in \mathbf{N}} P(n)$ is the total power consumption of the network and $P(n)$ is the power consumption of node n. w_i is the waiting time of user i and W_i is the maximum time that this user is willing to wait (maximum acceptable waiting time or voluntary waiting time). This means that the target is to minimize the total energy consumed over time E_N, under the condition that all users are finally accepted into the network, and no user waits for admission more than his predefined maximum voluntary waiting time W_i. The notion of voluntary waiting time is proposed to illustrate that longer waiting times can result in larger energy savings and to make sure that the mechanism takes into consideration the individual QoS demands of each flow.

This problem can be formalized as a Markov Decision Process (MDP). At any given point of time, the network is in a particular state S defined by the number of each type of active source-destination pairs and their bandwidth demand. This state specifies deterministically the power consumption of the network. At random times a new event E can occur, which can be a new arrival, a flow termination or an expiration of the waiting time of a flow in the waiting queue. When such an event occurs, the admission control mechanism has to choose whether to admit the new flow or send it to the waiting queue. It also has to decide whether to admit any flows that are already in the waiting queue. At some subsequent random time another event occurs, and the described process is repeated. The task of the admission control mechanism is to determine a policy that maximizes the sum of rewards $R(s, e)$ over an infinite horizon. The reward in this case

will be based on a combined cost related to the energy efficiency of this action and whether the maximum acceptable admission delay of the flows is respected.

Since we want to consider a real time solution, solving the problem in an optimal way is not straightforward, as there is no knowledge of the future traffic demand and consequently no knowledge on whether a more energy efficient state does exist within the limit of maximum acceptable waiting time of each flow. Thus, real time heuristic algorithms described are proposed in the following sections.

6.3 Energy aware admission control mechanism : Scenario 1

In this section a general increasing nonlinear power profile of the network nodes is assumed. The idea is that since some of the operation areas are more power efficient than others, the users' admission in the network could be delayed until a more energy-efficient time in the future.

More specifically, the proposed centralized Energy Aware Admission Control (EAAC) mechanism follows the steps described next:

1. A new user i informs the EAAC about its source s_i, destination d_i and demanded bandwidth bw_i. It also sets a maximum time limit w_i that the user is willing to wait until it is admitted into the network.

2. The EAAC calculates the minimum hop path π_i from s_i to d_i and collects the information about the current power consumption of the nodes n on this path.

3. Using the known power profile and the bandwidth of the flow, it estimates the increase in power consumption after the acceptance of the new flow.

$$\delta P = \sum_{n \in \pi_i} p_n(\lambda_n + bw_i) - \sum_{n \in \pi_i} p_n(\lambda_n) \tag{6.1}$$

   where p_n is the instantaneous power consumption of node n and λ_n is the current packet rate of the node n on path π_i.

4. If the estimated wattage increase δP is smaller than a fixed value Δ, the flow is accepted and admitted into the network (Δ is the threshold in increasing the power consumption that is acceptable by the EAAC). If not, the new flow is sent to a waiting queue. Note that the flows are stored in the waiting queue in a ascending order of their remaining wait time.

5. If a new flow arrives while the mechanism is busy estimating the δP of the previous flow, it joins a request queue. The mechanism checks in first-come-first-served order the flows in the request queue, going back to step 2. If no flow waits in the request queue, the mechanism picks a waiting user from the waiting queue and follows the same process from step 2.

6. If the waiting time of a flow w_i expires, the flow is immediately admitted into the network, irrelevantly of its estimated power increase.

6.3.1 Experiments

Configuration of the experiments

To evaluate the efficiency of this algorithm, experiments are performed in our laboratory testbed located at Imperial College London. The testbed consists of 18 PC-based routers connected according to the realistic topology of Czech Republics National Research and Education Network (CESNET) [2], seen in Figure 6.1.

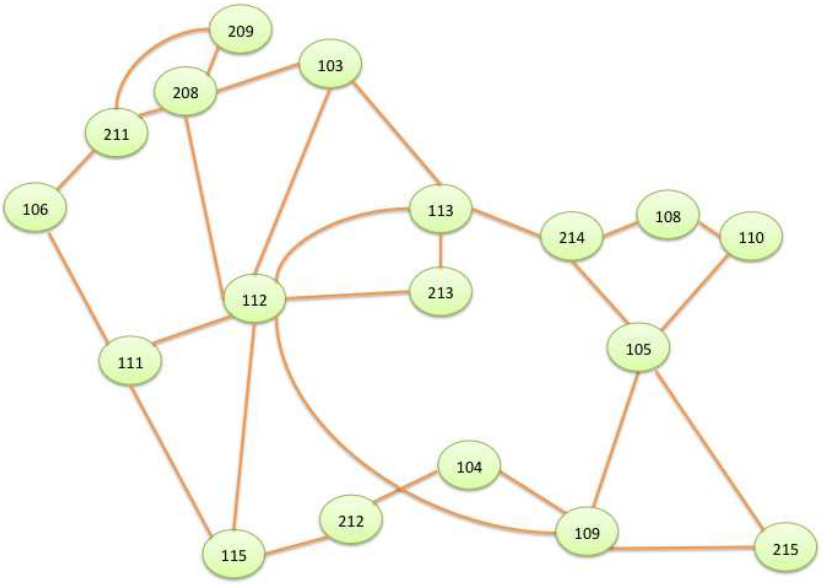


Figure 6.1: Experimental network topology consisting of 18 PC-based routers

The power consumption profile of these machines is assumed to follow a step-like behavior as shown in Figure 6.2, with a minimal power consumption of $10W$ for less than 100 packets/sec. This type of power profiles correspond to devices with rate adaptation where the power consumption coarsely adapts to the load, as described in

[86] and are categorized as multi-step in [15]. Note that for the experiments this specific power profile of the network devices is examined, however the same mechanism could be implemented in case of non-identical power profiles under the condition that some steps in power consumption arise.

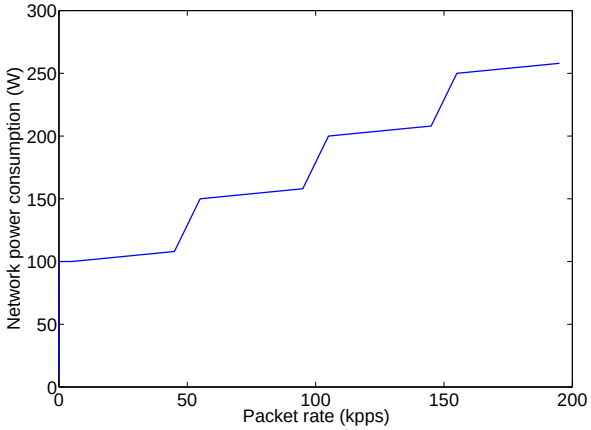


Figure 6.2: Power profile of the nodes used in the experiments. Power consumption adapts coarsely to the load and this results in a step-like behavior [15]

Two sets of experiments are performed, with different amount of users constantly making requests to send traffic; in the first set there are 4 users corresponding to 4 Source-Destination (S-D) pairs and in the second, more demanding set of experiments, there are 9 users independently making requests to send traffic into the network. For the first set of experiments, the impact of the threshold value of Δ on the amount of savings in energy and the admission delay is also examined.

In order to avoid having more than one users requesting to enter the network at the same time, each flow enters a queue ("request queue") at the data gathering point. Thus, all users from all source nodes will queue there in order to enter the network. After making a request, each user waits for a random time *intertime* and then makes a request again. This random *intertime* is set among requests, so as to have different rates for the arrivals.

Note that delays are measured via pinging and the power consumption of the nodes is estimated from the power profile. Thus, the carried traffic is being monitored and mapped to the profile shown in Figure 6.2. Therefore, it is believed that the results obtained will mimic the energy and delay characteristics that one would obtain in a standard IP network with these characteristics. Note that all the presented results are averaged over three runs of the experiment and the 95% confidence intervals are displayed.

Evaluation of EAAC

In this experiment there are 4 source-destination pairs $(103, 209)$, $(108, 212)$, $(111, 214)$ and $(209, 215)$. New flows are generated every *intertime* seconds, where *intertime* is randomly distributed between 10 and 40 secs. A random flow duration of $10-30$ seconds and a randomly distributed bandwidth request of $1-10Mbps$ are assumed. The packet size is set to $100bytes$. Finally, all the users are willing to wait up to 30 seconds before they are admitted into the network.

The experiment ran with our EAAC and without (immediately accepting in the network every new flow). New flows are generated for 300 secs, in order to have a long enough time interval to study the effect of admission control. Note that for the case without EAAC all requests will be accepted until the 300 secs, while for the case with EAAC there may be requests in the waiting queue at that time. In order to compare the total energy spent for serving the same amount of users in the network, for the EAAC case all users in waiting queue are accepted immediately after the 300 secs . The power consumption is observed for 350 secs, so that all flows will have finished using the network in both cases.

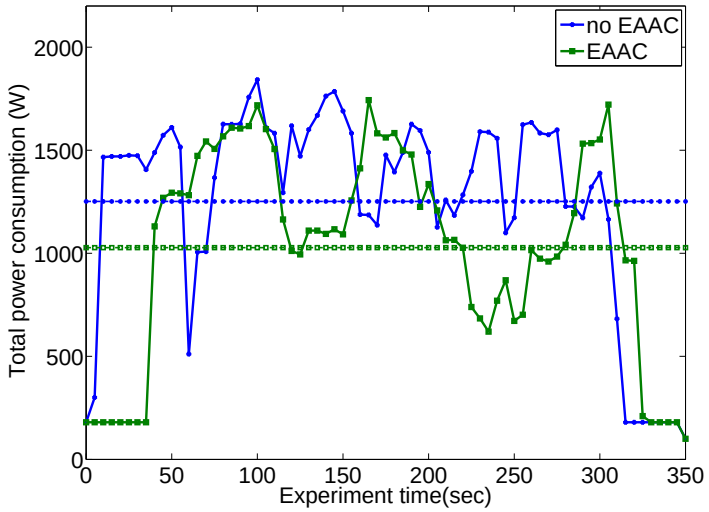


Figure 6.3: Network power consumption with admission control (green line) and without (blue line). Dotted lines represent the average values over time. An average power saving of 17% can be observed for the EAAC case.

The total network power consumption over time, for both cases, is shown in Figure 6.3. A slow start can be noticed in the beginning of the EAAC case (green line) when the network is empty and all flows are forced to wait. On the other hand, the power

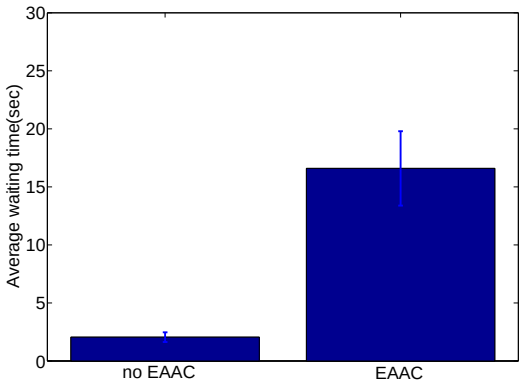


Figure 6.4: User average admission waiting time for the admission control and no admission control case. For the EAAC case, incoming flows are forced to wait for 16 seconds on average. The energy saving comes at a cost of delaying some users before they are admitted into the network.

consumption of EAAC is higher than the no EAAC case after the 300 secs, when all waiting flows are immediately admitted, regardless their energy consumption or waiting times, in order to guarantee fair comparison for the two cases. The dashed lines which correspond to the average values over time indicate a power saving of 17% in average power consumption for the EAAC case.

In Figure 6.4 the average waiting times of the users with and without the EAAC are presented. As expected, the energy saving comes at a cost of delaying the users before they are admitted into the network. More specifically, incoming flows are forced to wait for 16 seconds on average for the EAAC case. Note that in this experiment all users were willing to wait up to 30 seconds and that the EAAC guarantees that this limit is never violated. Thus, this average waiting time is composed of users immediately admitted, others waiting up to the time that a better condition is found in the network and those that are admitted because of expired waiting time.

Taking a closer look at these proportions of users (Figure 6.5), it can be observed that approximately 45% of the users are immediately admitted, while 30% are admitted due to expired waiting times. The remaining 25% are delayed before admission but admitted before their waiting time is expired, thus for those users the mechanism has been successful in finding a more energy efficient time of admission.

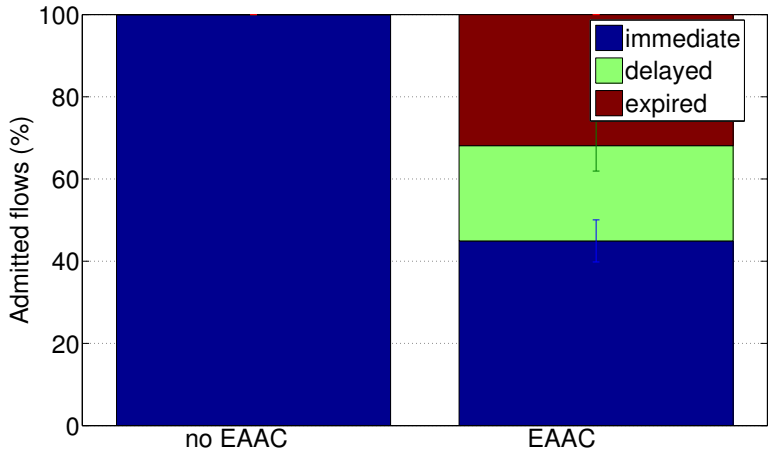


Figure 6.5: Types of admissions for the EAAC and no EAAC case. For the no EAAC case all users are immediately admitted. For the EAAC only 45% of the users are immediately admitted while 30% are admitted due to expired waiting times. The remaining 25% are delayed before admission but admitted before their waiting time has expired, when a more energy efficient state has been identified.

Impact of the Δ value

The pre-selected value of Δ is critical for the efficiency of the EAAC and should therefore be examined carefully. In order to demonstrate the effect of the threshold value Δ, the network is loaded with higher rate of flows' arrivals, i.e. the *intertime* is set randomly between 10 and 20 seconds and examine the values $\Delta = 60$, $\Delta = 100$, $\Delta = 200$ and $\Delta = 300$. These values were selected based on the specific power consumption profile to be a little larger than the possible step-increments of power consumption.

The average values of network power consumption for all cases are presented in Figure 6.6. For $\Delta = 60$ and $\Delta = 100$ the largest saving is observed, approximately 20% comparing to the no EAAC case. For $\Delta = 200$ the power savings are 17% and for $\Delta = 300$ 11% respectively.

Figure 6.7 shows the resulted average admission waiting time for the examined values of the admission threshold Δ. It reveals that the stricter the Δ value is, the greater the average waiting time. It is interesting to observe, that for $\Delta = 100$ the waiting time is 30% less than in case $\Delta = 60$, for which the same power saving was measured. Thus, in the case of $\Delta = 100$ the EAAC mechanism was more efficient, since it resulted in the largest savings with smaller admission delays. Looking at the percentage of successful late admissions (in other words the percentage of flows which were forced to

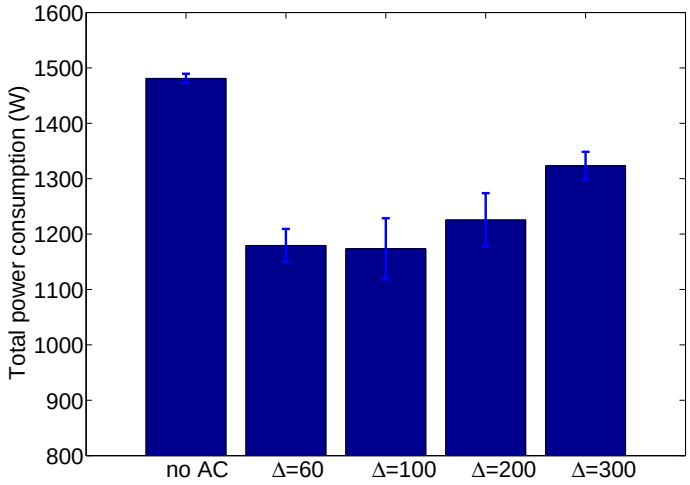


Figure 6.6: Average network power consumption results for several values of the admission threshold value Δ. For $\Delta = 60$ and $\Delta = 100$ the largest savings are observed (approximately 20% comparing to the no EAAC case)

wait and were admitted at a later time when a more energy efficient condition is found) in Figure 6.8 it can be observed that indeed in case $\Delta = 100$ the largest percentage increase is achieved (18%).

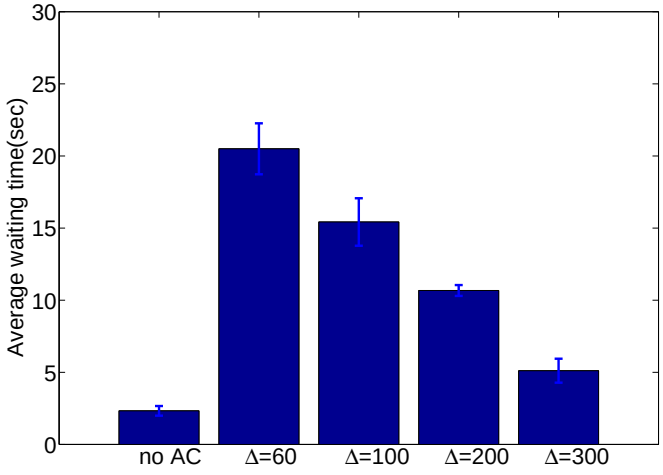


Figure 6.7: User average admission waiting time for several values of the admission threshold value Δ. The stricter the threshold is, the greater the average waiting time. $\Delta = 60$ and $\Delta = 100$ are the most power efficient values, but $\Delta = 100$ results in lower admission delay.

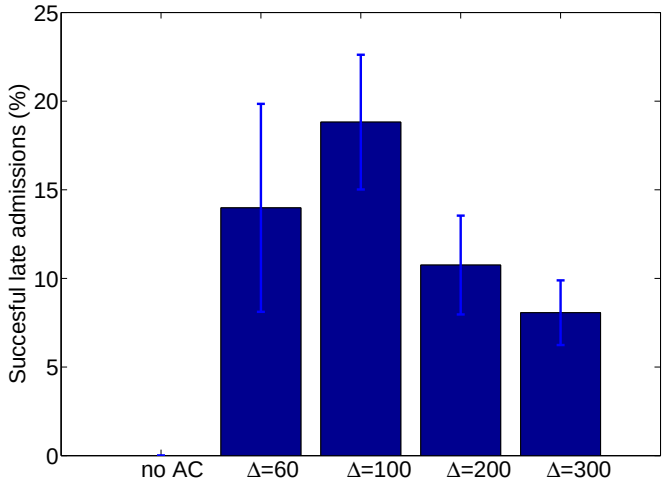


Figure 6.8: Percentage of successful late admissions (percentage of flows which were forced to wait and were admitted at a later time when a more energy efficient condition is found). For $\Delta = 100$ the EAAC mechanism is more successful.

In Figure 6.9, all types of admissions are displayed: a) the percentage of users immediately admitted, b) those waiting up to the time that a better condition is found in the network and c) those that are admitted because of expired waiting time.

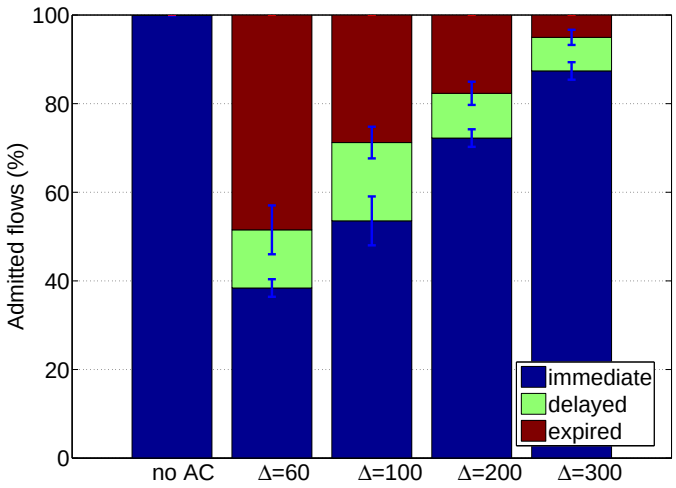


Figure 6.9: Types of admissions for several values of the admission threshold value Δ. As the threshold value increases, the percentage of immediately admitted users is also increasing. Though, the maximum efficiency of the mechanism was reached for $\Delta = 100$.

As can be observed, if one seeks to compromise the service of only a small amount of users, then larger values of Δ are required. On the other hand, for stricter values of the threshold Δ, the energy savings are greater. For the case of $\Delta = 100$, which is observed to be the best in the examined case, 18% of users were forced to wait and were admitted at a later time which was more energy-efficient for the network. However, 30% of users were forced to wait without success, as a more energy efficient state was not found before their expiry time. Taking this into consideration, one could argue that the case of $\Delta = 300$ is more successful, as the percentage of users forced to wait before admission 'in vain' is 40% of the total users that were forced to wait, while in case $\Delta = 100$ this percentage was approximately 62%. However, the EAAC achieved larger power savings for $\Delta = 100$ as in this case more users (in absolute values) were successfully delayed.

Figure 6.10 shows the total number of admitted flows in the network over time. A slow start is observed for stricter Δ values but all flows are finally admitted in the network for all cases.

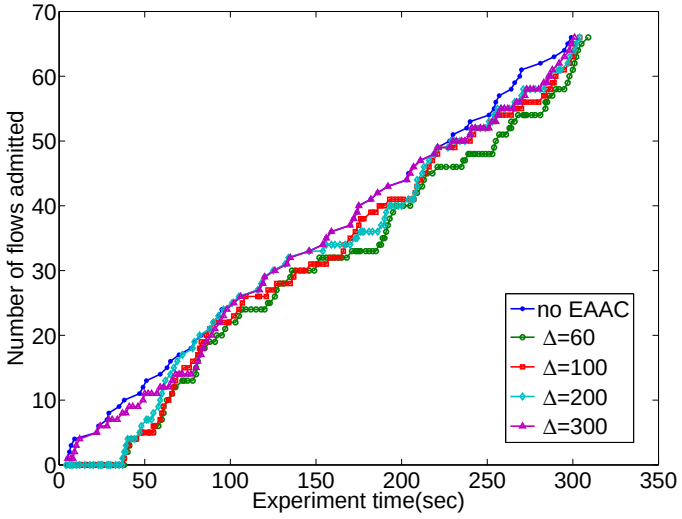


Figure 6.10: Number of admitted flows in the network for the for several values of the admission threshold value Δ over time. The same number of flows are finally admitted in the network for all cases.

To summarize, the value of the admission threshold Δ affects the efficiency of our proposed method and should therefore be selected carefully. Too small Δ values can lead to all flows being queued up before admission, while too large Δ values, can lead to all new flows being admitted which drastically reduces the effect of the EAAC. Regarding the admission waiting time, stricter Δ values will lead to more users being

queued up and will thus increase the average admission delay. Taking a close look at the power consumption and the admission delay results (Figure 6.3 and 6.4 respectively), it can be observed that the best choice in this case was $\Delta = 100$, for which the largest percentage of successful late admissions is achieved. Thus, the value of Δ could be readjusted online, reducing the value of the threshold up to the point that the efficiency of the mechanism starts deteriorating, thus up to the point where the percentage of successful delayed admissions starts decreasing.

Highly loaded network

The goal of this section is to examine the effect of the EAAC on packet loss and network latency. Thus, the case of a highly loaded network is presented. For this experiment the network is loaded with 9 source-destination pairs $(103, 209)$, $(108, 212)$, $(111, 214)$, $(209, 215)$, $(106, 213)$, $(115, 110)$, $(214, 103)$, $(211, 115)$, $(215, 113)$. *Intertime* is distributed between 10 and 20 secs and *bandwidth demand* between $1Mbps$ and $10Mbps$. The value $\Delta = 100$ which resulted to be the best in the previous analysis is used.

In Figure 6.11 the power consumption results for the EAAC and no EAAC case are shown. On average, the EAAC achieves in this case a power saving of 7%.

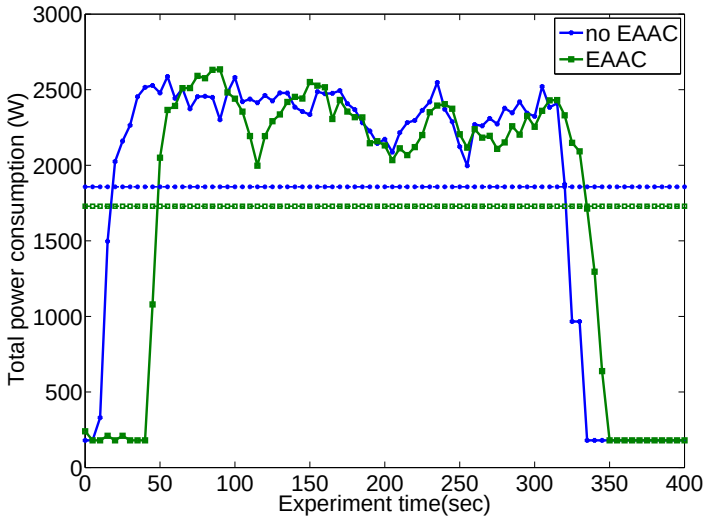
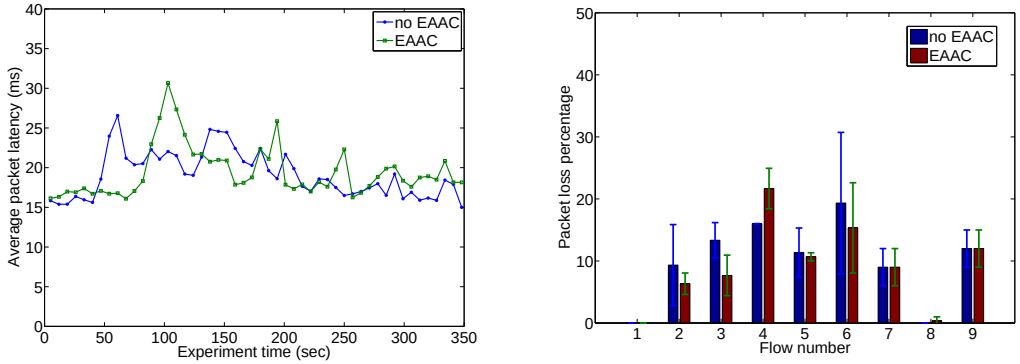


Figure 6.11: Network power consumption for the admission control (green line) and no admission control (blue line) cases in a highly loaded network. Dotted lines represent the average values over time. An average power saving of 7% can be observed for the EAAC case.

Taking a look at the packet loss in Figure 6.12b, it is noted that indeed this traffic input results in a highly loaded network with some very high packet loss rates. How-

ever, the admission control seems to have a neutral effect on the average packet loss (Figure 6.12b), since for some flows it can cause increase and for others decrease in the percentage of packets lost. The average network latency (Figure 6.12a) seems also not to be affected by the EAAC.



(a) Average network latency

(b) Packet loss per flow

Figure 6.12: Results for the admission control and no admission control case in highly loaded network. The EAAC didn't affect significantly neither the network latency nor the average packet loss.

Non-smart admission control

In this section the effect of applying a non-smart admission control mechanism is displayed, according to which users are simply delayed before admission up to their maximum waiting time. This is equivalent to the application of EAAC with $\Delta = 0$, as all flows induce a positive power increase in the network. The maximum waiting times w_j in this experiment are selected uniformly between $[10, 40] sec$. The power consumption over time is compared to the case where all flows are accepted as soon as they arrive, in Figure 6.13. It can be observed that initially the power consumption of the non-smart admission control is lower, as all flows are delayed until their waiting times expire. However, the average values of power consumption (dotted lines) show clearly that there is no energy saving by using such a mechanism. In other words, the selection of the appropriate threshold value Δ is the critical factor for the energy efficiency of the admission control mechanism.

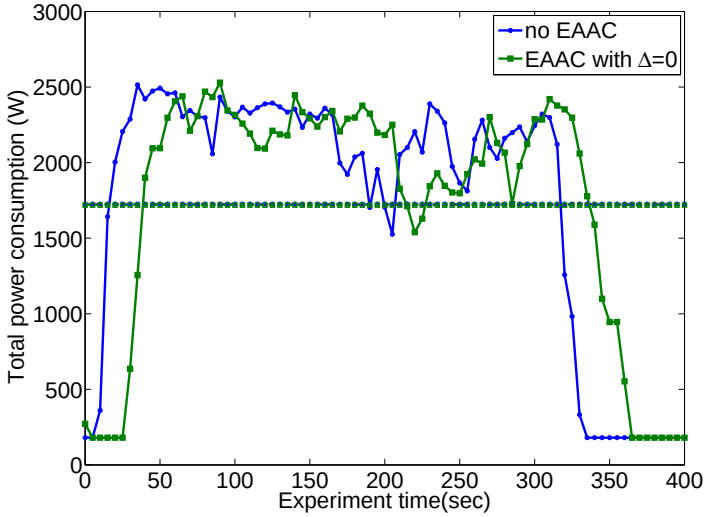


Figure 6.13: Network power consumption of no admission control (blue line) compared to non-smart admission control with $\Delta = 0$ (green line). Lower power consumption is observed for the non-smart admission control in the beginning of the experiment due to slow start. However, the average values (dotted lines) indicate no energy saving by using non-smart EAAC with random admission delays of flows.

6.4 Energy aware admission control mechanism: Scenario 2

In this section the case where nodes can be turned off is examined. Thus, in contrast to the step-like power profile of the previous section, here the examined nodes consume minimal power when they are off, and an almost constant amount of power when they are on, which is the case for our testbed's nodes. In fact, in this set of experiments we are able to measure the real time power consumption of a small experimental testbed that consists of nodes with on/off capabilities. The power consumption is measured using a power meter and realistic results are presented based on the existing equipment.

Similarly to the previous section, the EAAC mechanism decides whether a flow could be admitted into the network without significantly increasing to the energy consumed. If the flow is projected to be energy demanding, it is sent to a waiting queue and delayed until either the network conditions have changed or a maximum waiting time has expired. Instead of setting a value Δ for the admission threshold, here the decision is based on whether all nodes of the required path are ON or not. In addition to the mechanism presented in the previous section, in this case the EAAC is also responsible

for monitoring the state of the nodes (whether they are on or off) and sending out decisions to wake nodes up. Moreover, an independent auto-hibernation mechanism has been implemented which decides to turn off a node if it has been inactive for a long enough period.

Admission control mechanism description

When a new request arrives, the EAAC calculates the shortest path and checks whether this path is available. If at this instant of time all nodes on this path are ON, the flow is admitted. In case one or more nodes on the path are OFF, the flow is sent to a waiting queue. Flows in the waiting queue are periodically re-evaluated for admission, in case the network state has changed. The user can predefine a maximum waiting time that it is willing to wait. When this time expires, the EAAC sends out wake-requests to the nodes on the corresponding shortest path and admits the flow once the full path is established. To summarize, the EAAC mechanism uses the following steps (also shown in the flow diagram of Figure 6.14):

1. A new user i informs the EAAC about its source s_i and destination d_i. It also sets a maximum time limit w_i that the user is willing to wait until it is admitted into the network.

2. The EAAC calculates the minimum hop path π_i from s_i to d_i and collects the information about the current state of these nodes (whether they are currently on or off).

3. If all of the nodes on the shortest path are on, the flow is admitted into the network, else the flow is sent to a waiting queue.

4. If the waiting time of a flow in the waiting queue w_i expires, the EAAC turns on the nodes on the path that are off. Once they are on, the flow is admitted.

5. If the request queue is empty, the requests in the waiting queue are re-evaluated for admittance.

Note that this mechanism not only allows to monitor the state of the network, in order to avoid admitting flows when no path is available, but also tries to consolidate traffic admissions to improve efficiency when nodes are on. Also note, that the maximum waiting time of the users should take into consideration the additional time it would take from the time the EAAC sends out wake-requests until the path is fully established, which is denoted by T_{on}. Thus, for this mechanism to work, all users should be willing to wait for at least this time T_{on}.

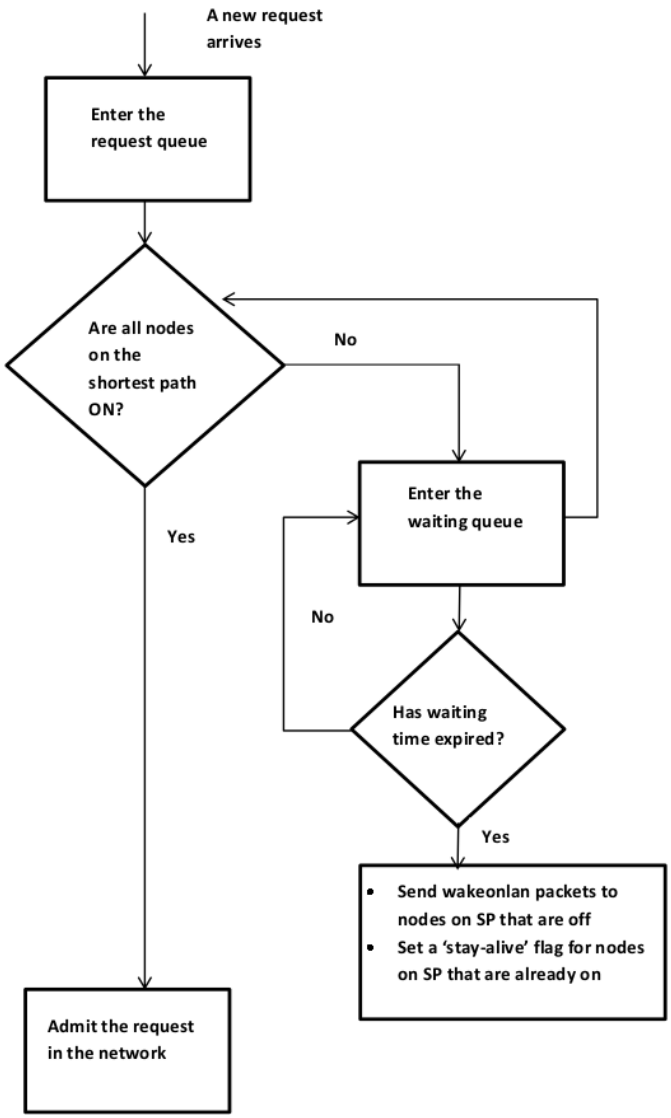


Figure 6.14: The flow diagram of the admission control mechanism

Auto-hibernation / waking-up mechanisms

In parallel to the EAAC mechanism, an auto-hibernation (AH) mechanism is running on all nodes. The AH mechanism is responsible to turn off the machines and its decision is based on whether the node has been inactive for a long enough period of time. A node is considered inactive when it is not processing any traffic and this inactivity state is being monitored by a built-in function of the node. Note that the length of the

inactivity period after which the AH mechanism should decide to turn off a node will be examined in the experiments.

As for the 'waking up' mechanism; when the waiting time of a user request expires, the admission control mechanism sends 'wake on lan' packets to the nodes of the required path that are currently OFF. A 'stay alive' flag is set on nodes that are already ON, to prevent the auto-hibernation mechanism from turning them off, while trying to turn on the rest of nodes on the path.

Energy cost from switching off nodes

Firstly, the behavior of our PC-based routers during their shutdown and wake up times is examined. Figure 6.15 shows the power consumption of one of our testbed's nodes. It is initially turned off and then it receives a "wake on lan" packet (10th second). The "turning on" process lasts from the 10th second to approximately the 40th second. The node is asked to turn off again at the 67th second and its finally off around the 80th second. Significant spikes in power consumption can be observed when turning on and off which result in a large energy waste, since the nodes are not processing any traffic during this time. So, when deciding to turn off a machine one has to take into consideration the additional energy spent for turning off and on and whether this is smaller than the savings during the time for which the node is off. For example, for the node examined in Figure 6.15, it can be calculated that the total energy waste of turning off and on is approximately 600 Joules. Since the power consumed when the node is on is approximately 60 Watts more than when it is off, the node should be turned off for more than 10 seconds in order to obtain positive energy savings.

Also, it is evident that the time needed to wake up or turn off a node in our PC-based routers is quite long (an average of approximately 30 seconds for booting and 10 seconds for shutting down is measured). Although the sleep mode might be preferable to turning the system completely off, in terms of waiting time, the sleep mode was not available in the machines used in our experiments.

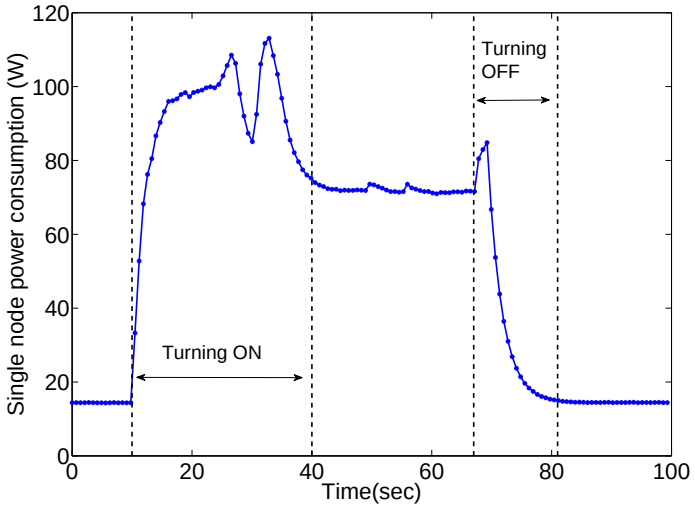


Figure 6.15: Power consumption measurements for turning on and off a node. The node is initially OFF and receives a wake-request on t=10sec. A sharp increase is observed in power consumption until the node is fully operational, on time t=40sec. On t=67sec it is turned off remotely and is finally OFF on t=80.

Existing networking equipment do not possess any sleep or turn off capability. However, current and future research targets to introduce sleep modes in routers or individual router components. Once this is enabled the required time to turn off and back on should be significantly smaller than in our laboratory equipment. However, there would still be the need for a control mechanism capable to ensure that nodes are on and fully operational when needed and that the user enters the network only when a path is established. Moreover, users could have a maximum acceptable waiting time up to which they are willing to wait before using the network, depending on their needs and preferences. This acceptable waiting time could be exploited to further increase the energy savings.

6.4.1 Experiments

Configuration of the experiments

In order to evaluate our mechanisms experiments are conducted on the real testbed located at Imperial College London. The testbed consists of 12 PC-based routers as shown in Figure 6.1. Green nodes can be turned off/on and their power consumption is being measured using Watts up?.Net power meter [3]. Grey nodes are sources/destinations and are always on. In the experiments, 5 users corresponding to 5 Source-Destination

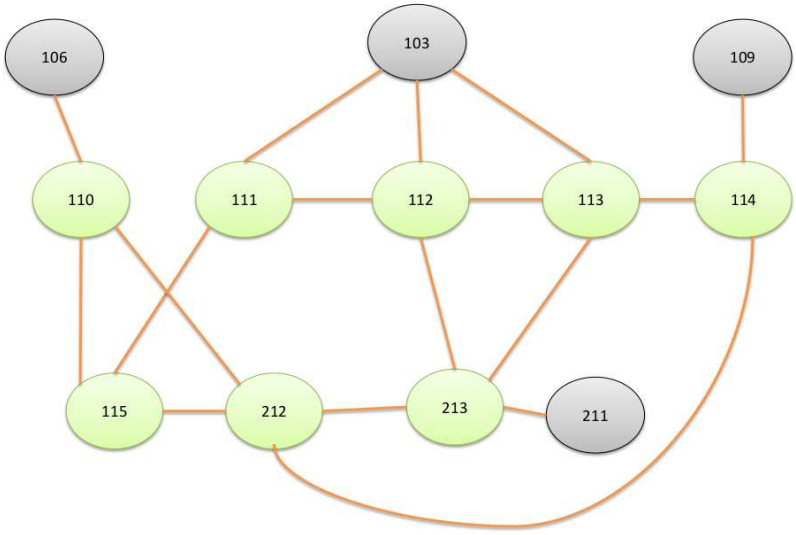


Figure 6.16: Experimental topology. Green nodes can be turned off/on remotely and their power consumption is being measured using a power meter. Grey nodes are sources/destinations and are always ON.

(S-D) pairs are independently making requests to send traffic into the network at random intervals. More specifically, the pairs (106,109), (103,211), (109,106) and (106,103) generate traffic every 150-200 seconds, for 700 seconds, with randomly distributed bandwidth request of 200-500Kbps. Note that these source-destination pairs are selected to cover the whole network and not leave any nodes unused. For the EAAC case it is assumed that all users are willing to wait for T_{on}, which is the time it takes to turn on a node, in case nodes on the required path are OFF on the time of the user's arrival. The experiment ran for 1000 seconds comparing the EAAC with the no admission control case where the users are not willing to wait and thus all nodes are constantly ON, ready to carry traffic.

The total power consumption over time for both cases is shown in Figure 6.17. As expected, the power consumption of the no admission control case where all nodes are ON, is almost constant. On the other hand, when using the EAAC large savings from turning off the nodes are observed, though there are also sharp spikes from the extra power needed to turn on nodes. The average value (measured until the 700 second for fairness) indicates a saving of 80 Watts which corresponds to 13% of savings on total power consumption.

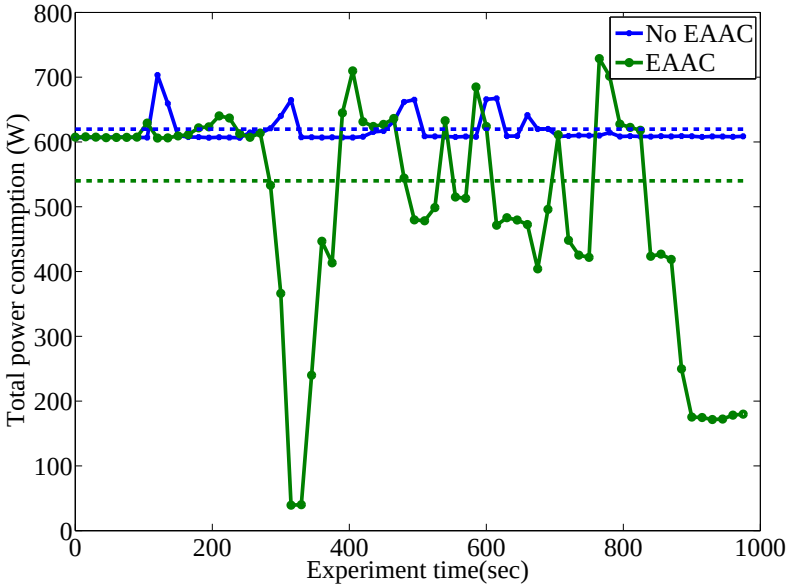


Figure 6.17: Power consumption results for EAAC (green line) and no EAAC case (blue line). For the no EAAC case the power is almost constant as no nodes are turned off. For the no EAAC case the power consumption fluctuates. Sharp falls due to unused machines turning off and also significant rises due to the power needed to turn machines back on are observed. Approximately 13% savings are measured on average (dotted line).

This energy saving comes at the cost of delaying some users before entering the network. The resulted average waiting time for the EAAC in this experiment is 22 seconds. Note that for the PC based routers used in our testbed the time needed to turn a node on was about 30 seconds.

Impact of the voluntary waiting time

In the experiments of section 6.4.1 the time that each user is willing to wait for admission in the network equals the time that it takes for a node to be turned on T_{on}, which in this case is approximately 30 seconds. In this section, the impact that longer voluntary waiting times of the users could have on energy saving is displayed. Therefore the case where the voluntary waiting time is just T_{on} sec is compared to a larger waiting time comprising of the T_{on} plus a random value distributed between 20-40 sec. These cases are compared to the case of zero voluntary waiting time, which is the case of no EAAC mechanism, as all nodes should be constantly ON, ready to carry traffic at any time.

In Figure 6.18 it can be observed that indeed longer voluntary waiting time could lead to larger energy savings. More specifically, 13% power savings for $W = T_{on}$ and

20% for $W = T_{on} + [20 - 40]$ are measured respectively.

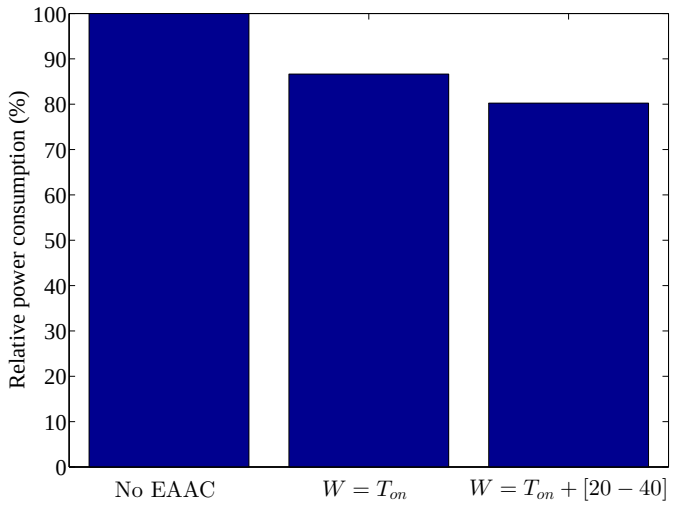


Figure 6.18: Relative power consumption results for different values of the voluntary waiting time W. When $W = 0$ (no EAAC), all nodes should be constantly ON, ready to receive traffic. For $W = T_{on}$, 13% average power savings are observed, while for $W = T_{on} + [20 - 40]$ the savings are 20% respectively.

Figure 6.19 shows the resulted average admission delays for all cases. The blue columns correspond to the overall average admission delay, including flows that were immediately admitted, while the red columns correspond to the average admission delay of the delayed flows. For the no EAAC case, the admission delay times are zero. As expected, as the voluntary waiting time W increases, the resulted average delay is increased. What is interesting to observe, is that for $W = T_{on} + [20 - 40]$ the overall average delay (all flows) is not much larger than the case of $W = T_{on}$. This can be explained looking at Figure 6.20 which shows the percentage of total flows that were actually delayed by the EAAC, i.e. the percentage of total flows which were put to the waiting queue before admission. It is observed that for $W = T_{on} + [20 - 40]$ less flows than in the case $W = T_{on}$ were put to the waiting queue by the EAAC, even though those put in the waiting queue were delayed for more time.

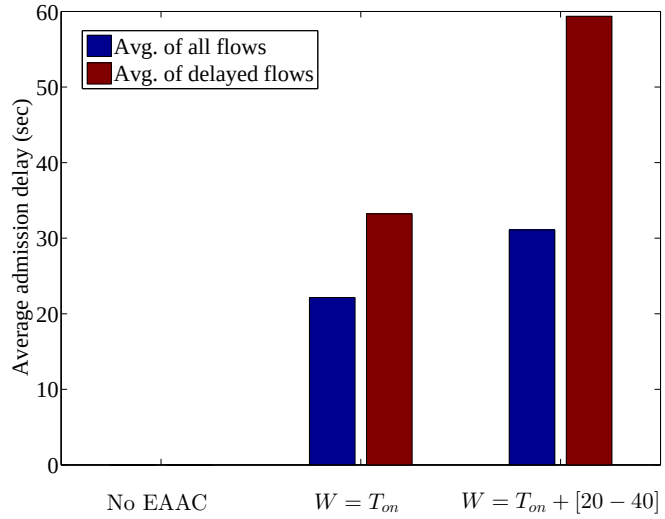


Figure 6.19: Average admission waiting time of flows, for different values of the voluntary waiting time. Blue columns correspond to the overall average admission delay, while red columns correspond to the average admission delay of the *delayed* flows. Interestingly, the overall average delay for the case $W = T_{on} + [20 - 40]$ is not much greater than the case $W = T_{on}$ as is the average delay of the *delayed* flows.

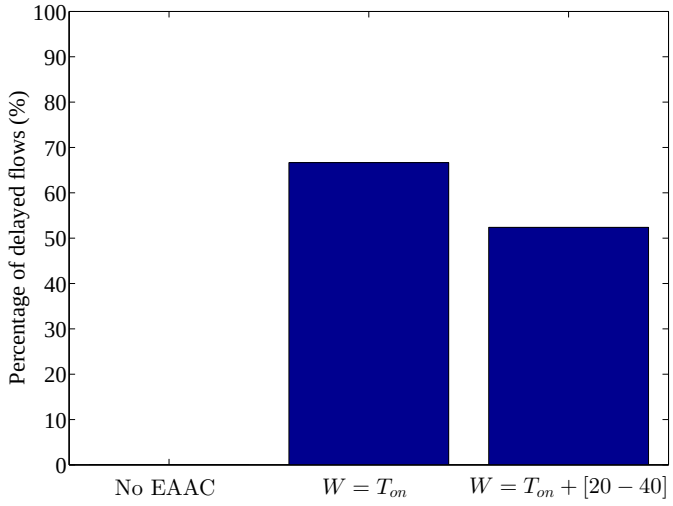


Figure 6.20: Percentage of total flows that were put to the waiting queue by the EAAC in each case of the voluntary waiting time W. What is interesting to observe, is that in case $W = T_{on} + [20 - 40]$ the percentage of delayed flows is smaller than in case $W = T_{on}$.

Impact of the auto-hibernation time

In the experiments of the previous sections, the auto-hibernating mechanism decided to turn a machine off if it was inactive for 50 seconds. In this section, the impact that this auto-hibernation time h has on the achievable energy saving is investigated.

In Figure 6.21, the average power consumption when the auto-hibernation time is relatively short ($h = 20secs$) is compared to a longer auto-hibernation period ($h = 50secs$) and to the case without EAAC, for both cases of voluntary waiting time W examined in the previous section ($W = T_{on}$ and $W = T_1 = T_{on} + [20 - 40]$). It can be observed that for smaller auto-hibernation time $h = 20$, greater savings are possible. The best case is achieved for small auto-hibernation time and large voluntary waiting time, which corresponds to 27%.

In Figure 6.22, the average times the users had to wait in all these cases are compared. Even if short auto-hibernation time is preferable for energy savings, it induces longer average waiting times, as in most of the cases new users have to wait due to one or more nodes on their required path being OFF at the time of their arrival. Taking a closer look at the percentage of delayed flows in Figure 6.23, it can be observed that, indeed, for the smaller auto-hibernation time more than 90% of the users had to wait, while for $h = 50$ this percentage was 50-65%. The average waiting time of the *delayed* flows seems to only depend on their voluntary waiting times W.

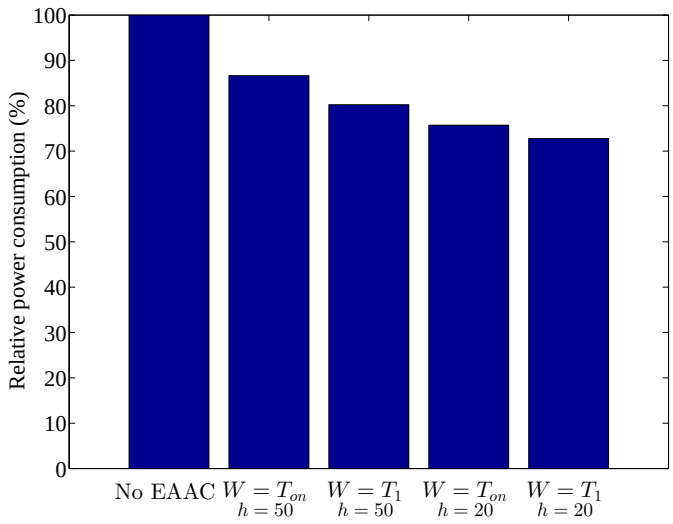


Figure 6.21: Power consumption results for different values of the voluntary waiting time W and auto-hibernation time h. As expected, the shorter h value results in larger savings as the nodes remain unused for shorter period before being turned off. Best case is for $h = 20$ and $W = T_1$ (27% average power savings).

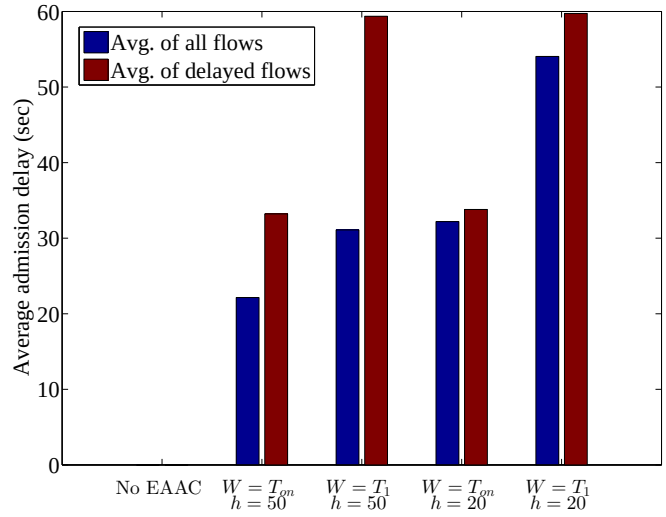


Figure 6.22: Average admission waiting time of flows for different values of the voluntary waiting time W and auto-hibernation time h. The average delay of the *delayed* flows (red columns) is only dependent on the value W, while the overall average delay (blue columns) is significantly increased for small auto-hibernation h.

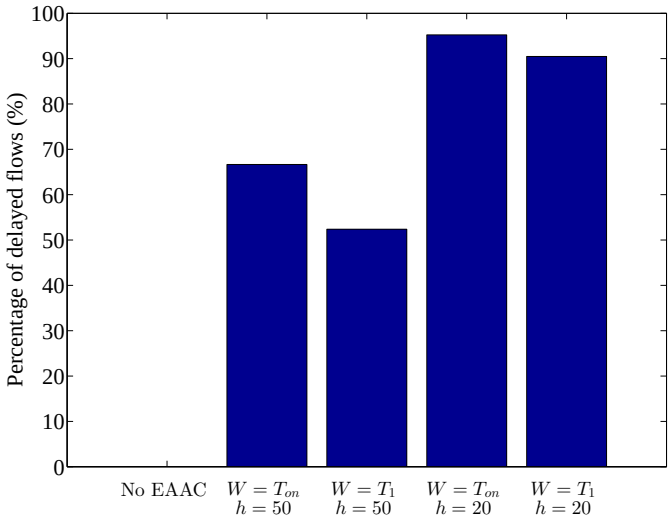


Figure 6.23: Percentage of delayed flows for different values of the voluntary waiting time W and auto-hibernation time h. For short auto-hibernation time $h = 20$ more than 90% of the users are delayed as one or more nodes on their required path are OFF at the time of their arrival. The larger voluntary waiting time $W = T_1$ results in a smaller percentage of flows being delayed, compared to $W = T_{on}$, for the same h value.

On the other hand, the overall average delay as well as the percentage of flows that are delayed are significantly increased for small auto-hibernation h. In both cases of $h = 20$ and $h = 50$, the larger voluntary waiting time $W = T_1$ results in a smaller percentage of flows being delayed and larger energy savings, compared to $W = T_{on}$ for the same h value. Thus, it can be remarked that longer voluntary waiting times give better flexibility to the EAAC mechanism and result in better efficiency.

Frequency of turning the machines on/off

Another parameter that should be taken into account is the frequency in which nodes are being turned off and back on. In the course of our experiments, it has become clear that the number of times the nodes are turned off affects proportionally the probability of failure of the nodes as well as the probability of concurrent decisions for admittance and auto-hibernations and thus packet losses. In Figure 6.24 the total number of node turn offs during each experiment are presented. It can be observed that shorter auto-hibernation time results in significantly larger number of turn offs. Moreover, larger W value allows the EAAC to keep nodes turned off for longer period, thus results in fewer turn offs for the same h value.

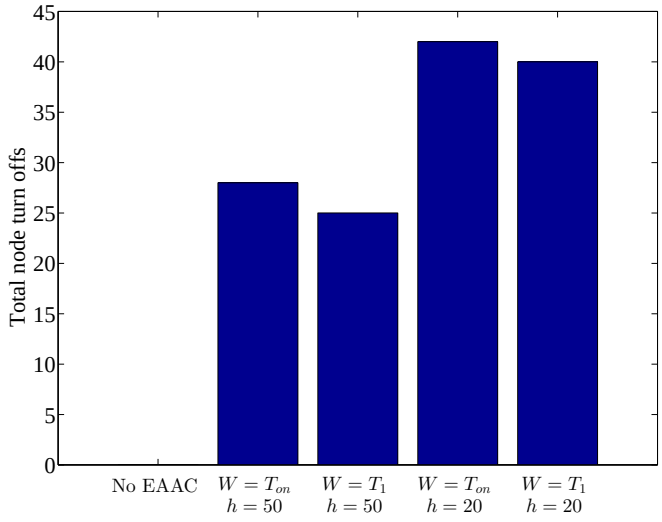


Figure 6.24: Total number of times that nodes have been turned off and back on during each experiment, for different values of the voluntary waiting time W and auto-hibernation time h. Smaller h value results in significantly more total turnoffs. Also, larger W value allows the EAAC to keep nodes turned off for longer period, thus results in fewer turnoffs for the same h value.

One very important thing to observe from Figure 6.24, is that the total number of

turn offs is not related to the average power consumption measured over time. Thus, a static solution that would try to turn off as many nodes as possible wouldn't be the most energy efficient. In fact, it could also lead to greater energy consumption, if nodes stay turned off for very short time. Another important observation is that even if short auto-hibernation time results in much larger energy savings, it increases not only the admission delay but also the total number of turn offs. In this case, one has to also take into consideration the increase in number of turn offs which can provoke a larger probability of failures.

6.5 Discussion of the results

In this chapter, a novel Energy Aware Admission Control (EAAC) mechanism has been proposed and evaluated. The EAAC aims to reduce energy consumption of a network by delaying the admission of users which would result in an increase in power consumption and trades off admission delay for energy efficiency.

In the first part (Scenario 1), the case of a power profile that adapts to the load and has a nonlinear behavior has been examined. In this case, the EAAC mechanism can delay the users that would result in a large increase of power consumption in the network, and admit them at a later time when their effect on the power consumption of the network will be smaller than a fixed threshold Δ. Experimental results show the efficiency of the proposed mechanism, at the cost of increased admission delays, while the critical parameters are the selection of the threshold value Δ and the maximum waiting times W that the users are willing to wait.

In the second part (Scenario 2), the EAAC is evaluated in a testbed with on/off capabilities. In this version the decision is taken based on whether all nodes on the path are ON or not. There is extended literature in energy aware routing for turning off nodes, though one has to deal with the practical problems of such a mechanism, such as the required time to turn a node on, which is quite significant. In addition, turning a machine on comes with an instantaneous additional increase in power consumption which could possibly even out the savings of turning it off. Additionally, a control mechanism should be able to monitor the traffic of the node and decide whether the node is not needed after it has been idle for some time. Our experiments present some realistic results, based on our testbed capabilities, where all the above mentioned points are taken into consideration. More specifically, the EAAC mechanism is able to send wake-requests when a node is needed to route traffic. Moreover, an auto-hibernation mechanism has been implemented which turns off unused machines after a hibernation time h. All users should be willing to wait for $W = T_{on}$, where T_{on} is the time it

needs for a network node to turn on and become operational. It is evident that, in order to implement on/off mechanisms in a real network, the time it takes to turn on a network device should be significantly reduced. Nevertheless, our experiments show that the proposed EAAC is able to monitor and manage a network with on/off capabilities and result in significant power savings, at the cost of delaying users before admissions. Investigations on the voluntary waiting time W and the auto-hibernation time h have also been performed and show the significance of these parameters. This work reveals the potential of an Energy Aware Admission Control mechanism which can offer large room for energy savings and network management.

7 Conclusion

7.1 Summary of thesis contributions

Traditionally, computer networks are designed and operated in a way that optimizes the quality of the provided service (QoS) in terms of delay, packet loss, jitter etc. However, the increasing need for efficiency in power consumption, for both environmental and financial reasons, is pushing research interest towards the energy optimization of networks. Network devices are always fully provisioned to satisfy the maximum demand and guarantee good performance at peak traffic times. Consequently, such devices are underutilized, or even idle, for large periods of time. Despite being underutilized, network devices continue consuming large amounts of energy. Thus, there is an intuitive trade-off between QoS guarantees and energy efficiency. The research question that is identified and addressed in this work, is to examine and propose techniques in order to intervene in a given wired network with certain installed routers and links and make it energy efficient while also respecting the QoS needs of the carried traffic.

As a first and very important step, a model of the network is built which is a special case of G-Networks with triggered customer movement. This model allows us to have a clear view of the parameters that affect the power consumption and the quality of service provided. G-Networks are queuing networks with additional control capabilities (e.g. negative customers, resets, triggers) and have the very important properties of product form solution and unique solution. These control capabilities make these networks important in modeling.

Our presented model has some very important features. Firstly, it represents distinct traffic flows of each customer in the network. This representation, allows us to examine the network at a per-flow level. Moreover, the most important QoS parameters regarding the delay at the routers and links can be represented. Based on the measurements and background knowledge, a power consumption model which can be adjusted to different kinds of devices is built and integrated in the model. What is also very important is that links are also treated as nodes, which grants an added level of flexibility to the model. Finally, we introduce control traffic which can be seen either as physical flows of signaling information that travel throughout the network to reach routers that are not accessible, or as a virtual representation of the rerouting decisions at nodes.

Next, a gradient descent algorithm is presented which operates in $O(N^3)$ time complexity and propose a composite power and QoS cost function. Both the presented thorough network routing control analysis and the optimization algorithm can be adjusted to the characteristics of a network and be used to build control mechanisms for energy efficient routing, taking into consideration the QoS and the cost of changing routing decisions. Although with current power consumption characteristics of devices the savings of applying just routing control might appear small, as more power efficient devices will be deployed in the networks and the behavior of the nodes will be diverse, it is believed that this analysis will be even more important.

In the next chapter, the optimization for different cases of power consumption characteristics of the network is evaluated. Since, the complexity of the presented algorithm is high for online calculation we build a gradient descent based heuristic and compare the results. We also explore the case of decentralized decisions, where each source has partial information for a subnetwork and searches for the best routing within this subnetwork. Finally, we demonstrate how the optimization results can vary depending on the weights of the cost function.

Moreover, the harsher energy saving solution of turning off devices are experimentally evaluated. We first explore the potential savings and tradeoff in delay and packet loss in a real case scenario where nodes are turned on and off and their power consumption is being measured online, along with the delay and packet loss measurements of the flows. This study comes up with tradeoff-curves showing operational states depending on power consumption and delay. This could be useful to showcase an estimation of potential savings and losses for certain network operation requirements.

In the last chapter, a novel energy aware admission control mechanism is proposed. Based on the idea of admission control which was originally introduced to guarantee the QoS levels required, a mechanism is presented and built, which tries to keep the network at the lowest possible level of power consumption, by delaying the admission of flows in it. While it is shown to result in large admission delays, it is expected that it could be an efficient mechanism for future devices, that could change operational states (on-off or between a set of possible rates) faster.

7.2 Future work

The research area of energy efficiency in communication networks is very wide and has only recently drawn attention, though the research interest is growing at a very rapid rates. The research presented in this thesis can be extended in several directions. In this section we discuss open issues and provide possible directions for future work.

In Chapter 3, we have presented a thorough network routing control analysis and a gradient descent optimization algorithm which can be adjusted to the characteristics of a network. It can be used to evaluate the performance of other techniques and control mechanisms for energy efficiency, taking into consideration the QoS and the cost of changing routing decisions. As further work, the optimization algorithm could be extended to also control service rates of the network nodes along with the routing decisions. Thus, the optimization would be responsible to change not only routing of the network traffic but also the rate at which packets are processed in the routers as well as the speed of the line cards. This option will be enabled in reality in case instantly changing adaptive rates and stand-by modes are introduced in network nodes in the future. The slower processing rates will have an effect on packet queuing delay in the node and the proposed combined optimization goal, comprising both of power consumption and delay, will be again essential to tune the relative importance of the two quantities.

In Chapter 4 we have presented case studies depending on different network power consumption characteristics to estimate potential savings via energy efficient routing and evaluate the proposed optimization algorithm. This algorithm can be applied in real network cases where details of power consumption characteristics would be available and include the effect of control, once this is measured. Moreover, the one-step heuristic and the decentralized algorithm proposed can be applied in real time with a realistic evolving traffic matrix. The rate at which decisions should be made has to be examined, depending on the cost of control and the rate at which changes are justified.

In Chapter 5, we demonstrate the tradeoff on network performance of turning off devices with current characteristics. Further efforts in the direction of introducing fast turn off of devices and fast wake up signaling will enable the efficient usage of these mechanisms. The presented tradeoff curves will be useful to decide the required operating state of a network as a total and facilitate pricing techniques according to users requirements and network operator energy costs. These curves can provide the basis for future mathematical models of the behavior of such a system, that could give an estimation on the tradeoff in QoS when a specific decisions on turning off of devices are taken. Moreover, they can be used in conjunction with pricing techniques depending on QoS demands and the cost of power needed for these constraints to be guaranteed.

Finally, the energy aware admission control mechanism described in Chapter 6, is responsible to monitor the network and delay the users until the required network resources become available. Moreover, it attempts to tradeoff admission delay for energy efficiency in a way acceptable to users. The presented results are limited to the laboratory testbed's capabilities and would become more important in future devices with

stand-by modes and rate adaptation. In such a general scenario, the admission control and routing control could be combined to offer the best results in each case of network hardware capabilities and QoS demands.

To conclude, as current trends and new technologies forecast a substantial growth on the usage of the networks and its significance in various aspects of our everyday life, the problem of energy efficient networks is a strong challenge for the research community. This work, consists a throughout analysis of the energy efficiency in communication networks. Since the problem has been modelled and well understood, the goal for future research should be on applying this knowledge and creating an energy efficient network, without compromising the QoS. Firstly, the power profiles of existing networking devices should be measured in agreement with internet service providers and data center operators. Then, using the described methods, the aim should be on the development of plug-ins in the network that would allow the dynamic management of network traffic in the most energy efficient way, while also respecting the QoS demands of the users. These online mechanisms should be able to i) observe traffic flows in the network, monitor the status of the nodes and the networks power consumption, ii) select the network routing configuration that offers lower power consumption, with an acceptable level of QoS to ongoing and predicted flows, and iii) manage and sequence dynamic changes in links and nodes, and reroute traffic, to achieve reduced power consumption at acceptable QoS levels.

Bibliography

[1] *Advanced Configuration and Power Interface Specification, Revision 5.0. Available online : http://www.acpi.info/spec50.htm.*

[2] *CESNET2 Network. Available online: http://www.ces.net/network/.*

[3] *Wattsup circuit controllers. Available online: https://www.wattsupmeters.com.*

[4] A. Adelin, P. Owezarski, and T. Gayraud, "On the impact of monitoring router energy consumption for greening the internet," in *Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on*, oct. 2010, pp. 298 –304.

[5] G. Ananthanarayanan and R. H. Katz, "Greening the switch," in *Proc. of Workshop on Power Aware Computing and Systems (HotPower '08)*, December 2008.

[6] P. Arabas, K. Malinowski, and A. Sikora, "On formulation of a network energy saving optimization problem," in *Communications and Electronics (ICCE), 2012 Fourth International Conference on*, aug. 2012, pp. 227 –232.

[7] V. Atalay, E. Gelenbe, and N. Yalabik, "The random neural network model for texture generation," *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, vol. 6, pp. 131–141, 1992.

[8] J. Baliga, K. Hinton, and R. S. Tucker, "Energy consumption of the internet," in *Optical Internet, 2007 and the 2007 32nd Australian Conference on Optical Fibre Technology. COIN-ACOFT 2007. Joint International Conference on*, June 2007, pp. 1 –3.

[9] L. Barroso and U. Holzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, pp. 33 –37, dec. 2007.

[10] A. Berl, E. Gelenbe, M. Girolamo, G. Giuliani, H. De Meer, M. Dang, and K. Pentikousis, "Energy-efficient cloud computing," *The Computer Journal*, vol. 53, no. 7, pp. 1045–1051, September 2010.

[11] D. Bertsekas and R. Gallager, *Data Networks*. Prentice-Hall International Editions, 1992.

[12] G. Bianchi, F. Borgonovo, A. Capone, L. Fratta, and C. Petrioli, "Endpoint Admission Control with Delay Variation Measurements for QoS in IP Networks," *Computer Communication Review*, vol. 32, no. 2, pp. 61–69, Apr. 2002.

[13] G. Bianchi, A. Capone, and C. Petrioli, "Throughput analysis of end-to-end measurement-based admission control in ip," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, Mar. 2000, pp. 1461 –1470 vol.3.

[14] A. Bianzino, C. Chaudet, F. Larroca, D. Rossi, and J. Rougier, "Energy-aware routing: A reality check," in *GLOBECOM Workshops (GC Wkshps), 2010 IEEE*, 2010, pp. 1422–1427.

[15] A. Bianzino, C. Chaudet, D. Rossi, and J.-L. Rougier, "A survey of green networking research," *Communications Surveys Tutorials, IEEE*, vol. 14, no. 1, pp. 3 –20, quarter 2012.

[16] K. Bilal, S. Khan, S. Madani, K. Hayat, M. Khan, N. Min-Allah, J. Kolodziej, L. Wang, S. Zeadally, and D. Chen, "A survey on green communications using adaptive link rate," *Cluster Computing*, pp. 1–15, 2012. [Online]. Available: http://dx.doi.org/10.1007/s10586-012-0225-8

[17] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti, "Energy efficiency in the future internet: A survey of existing approaches and trends in energy-aware fixed network infrastructures," *Communications Surveys Tutorials, IEEE*, vol. 13, no. 2, pp. 223 –244, quarter 2011.

[18] A. Boukerche, X. Cheng, and J. Linus, "A performance evaluation of a novel energy-aware data-centric routing algorithm in wireless sensor networks," *Wireless Networks*, vol. 11, no. 5, pp. 619–635, 2005.

[19] L. Breslau, E. W. Knightly, S. Shenker, I. Stoica, and H. Zhang, "Endpoint admission control: architectural issues and performance," *SIGCOMM Comput. Commun. Rev.*, vol. 30, pp. 57–69, August 2000. [Online]. Available: http://doi.acm.org/10.1145/347057.347400

[20] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright, "Power awareness in network design and routing," in *INFOCOM 2008. IEEE*, April 2008, pp. 457–465.

[21] L. Chiaraviglio, D. Ciullo, E. Leonardi, and M. Mellia, "How much can the Internet be greened?" in *GLOBECOM Workshops, 2009 IEEE*, 30 2009-Dec. 4 2009, pp. 1 –6.

[22] L. Chiaraviglio and I. Matta, "Greencoop: Cooperative green routing with energy-efficient servers," in *e-Energy 2010 - First International Conference on Energy-Efficient Computing and Networking*, April 2010.

[23] L. Chiaraviglio, M. Mellia, and F. Neri, "Energy-aware backbone networks: A case study," in *Communications Workshops, 2009. ICC Workshops 2009. IEEE International Conference on*, June 2009, pp. 1 –5.

[24] ——, "Reducing power consumption in backbone networks," in *Communications, 2009. ICC '09. IEEE International Conference on*, June 2009, pp. 1 –6.

[25] K. Christensen, P. Reviriego, B. Nordman, M. Bennett, M. Mostowfi, and J. Maestro, "IEEE802.3az: the road to energy efficient ethernet," *Communications Magazine, IEEE*, vol. 48, no. 11, pp. 50 –56, November 2010.

[26] A. Cianfrani, V. Eramo, M. Listanti, M. Marazza, and E. Vittorini, "An energy saving routing algorithm for a green ospf protocol," in *INFOCOM IEEE Conference on Computer Communications Workshops , 2010*, March 2010, pp. 1 –5.

[27] S. Dilip Kumar and B. Vijaya Kumar, "Eaac: Energy-aware admission control scheme for ad hoc networks," *International Journal of Wireless Networks and Communications*, vol. 1, no. 2, pp. 201–219, 2009.

[28] S. El-Dolil, A. Al-Nahari, M. Desouky, and F.-S. Abd El-Samie, "Uplink power based admission control in multi-cell wcdma networks with heterogeneous traffic," *Progress In Electromagnetics Research B*, vol. 1, pp. 115–134, 2008.

[29] W. Fisher, M. Suchara, and J. Rexford, "Greening backbone networks: reducing energy consumption by shutting off cables in bundled links," in *Proceedings of the first ACM SIGCOMM workshop on Green networking*, ser. Green Networking '10. New York, NY, USA: ACM, 2010, pp. 29–34. [Online]. Available: http://doi.acm.org/10.1145/1851290.1851297

[30] S. Floyd, "Comments on Measurement-based Admissions Control for Controlled-Load Services," *Computer Communication Review*, July 1996.

[31] J.-M. Fourneau and E. Gelenbe, "Flow equivalence and stochastic equivalence in g-networks," *Computational Management Science*, vol. 1, no. 2, pp. 179–192, 2004.

[32] J.-M. Fourneau, E. Gelenbe, and R. Suros, "G-networks with multiple classes of negative and positive customers," *Theoretical Computer Science*, vol. 155, pp. 141–156, 1996.

[33] J.-M. Fourneau and M. Hernandez, "Modeling defective parts in a flow system using g-networks," in *Proc. Second Int. Workshop on Performability Modeling of Comput. and Communic. Syst.*, Le Mont Saint-Michel, 1993.

[34] L. Fratta, M. Gerla, and L. Kleinrock, "The flow deviation method: an approach to store-and-forward computer-communication network design," *Networks*, vol. 3, 1973.

[35] L. Gan, A. Walid, and S. Low, "Energy-efficient congestion control," *SIGMETRICS Performance Evaluation Review*, vol. 40, no. 1, pp. 89–100, Jun. 2012. [Online]. Available: http://doi.acm.org/10.1145/2318857.2254770

[36] E. Gelenbe and T. Mahmoodi, "Energy-Aware Routing in the Cognitive Packet Network," in *International Conference on Smart Grids, Green Communications, and IT Energy-aware Technologies(Energy 2011)*, 2011.

[37] E. Gelenbe, G. Sakellari, and M. D' Arienzo, "Admission of QoS Aware Users in a Smart Network," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 3, no. 1, pp. 4:1–4:28, Mar. 2008.

[38] E. Gelenbe, "Random neural networks with negative and positive signals and product form solution," *Neural Computation*, vol. 1, no. 4, pp. 502–510, 1989.

[39] ——, "Product-form queueing networks with negative and positive customers," *Journal of Applied Probability*, vol. 28, no. 3, pp. 656–663, 1991.

[40] ——, "G-networks with instantaneous customer movement," *Journal of Applied Probability*, vol. 30, no. 3, pp. 742–748, 1993.

[41] ——, "G-networks with signals and batch removal," *Probability in the Engineering and Informational Sciences*, vol. 7, pp. 335–342, 1993.

[42] ——, "Learning in the recurrent random neural network," *Neural Computation*, vol. 5, no. 1, pp. 154–164, 1993.

[43] ——, "G-networks: An unifying model for queuing networks and neural networks," *Annals of Operations Research*, vol. 48, no. 1-4, pp. 433–461, 1994.

[44] ——, "Cognitive packet network," *United States Patent*, vol. 6,804,201, October 2004.

[45] ——, "Steady-state solution of probabilistic gene regulatory networks," *Physical review. B, Condensed matter*, vol. 76, no. 3, p. 31903, 2007.

[46] ——, "Network of interacting synthetic molecules in steady state," *Proceedings - Royal Society. Mathematical, physical and engineering sciences*, vol. 464, no. 2096, p. 2219, 2008.

[47] ——, "Steps towards self-aware networks," *Communications of the ACM*, vol. 52, pp. 66–75, 2 2009.

[48] E. Gelenbe and F. Batty, "Minimum cost graph covering with the random network model," in *Proc. Conf. ORSA Techn. Committee Comput. Sci.*, Williamsburg: Pergamon, 1992.

[49] E. Gelenbe and A. Labed, "G-networks with multiple classes of signals and positive customers," *European Journal of Operations Research*, vol. 108, no. 2, pp. 293–305, 1998.

[50] E. Gelenbe, X. Mang, and R. Onvural, "Diffusion based statistical call admission control in ATM," *Performance Evaluation*, vol. 27-28, pp. 411 – 436, 1996.

[51] E. Gelenbe and I. Mitrani, *Analysis and Synthesis of Computer Systems*. Imperial College Press, World Scientific, 2010.

[52] E. Gelenbe and G. Pujolle, *Introduction to Queueing Networks*, 2nd ed. John Wiley & Sons Ltd, 1998.

[53] E. Gelenbe, G. Sakellari, and M. D'arienzo, "Admission of qos aware users in a smart network," *ACM Trans. Auton. Adapt. Syst.*, vol. 3, pp. 4:1–4:28, March 2008. [Online]. Available: http://doi.acm.org/10.1145/1342171.1342175

[54] E. Gelenbe and S. Silvestri, "Reducing power consumption in wired networks," in *24th International Symposium on Computer and Information Sciences (ISCIS'09)*, North Cyprus, 14-16 September 2009, (IEEE Digital Library).

[55] E. Gelenbe and S. Timotheou, "Random Neural Networks with Synchronised Interactions," *Neural Computation*, vol. 20, no. 9, pp. 2308–2324, 2008.

[56] D. Gross and C. M. Harris, *Fundamentals of queueing theory (2nd ed.)*. New York, NY, USA: John Wiley & Sons, Inc., 1985.

[57] R. Guerin, "A unified approach to bandwidth allocation and access control in fast packet-switched networks," in *INFOCOM '92. Eleventh Annual Joint Conference*

of the IEEE Computer and Communications Societies, IEEE, vol. 1, may 1992, pp. 1 –12.

[58] C. Gunaratne, K. Christensen, and B. Nordman, "Managing energy consumption costs in desktop pcs and lan switches with proxying, split tcp connections, and scaling of link speed," *International Journal of Network Management*, vol. 15, no. 5, pp. 297–310, September 2005.

[59] M. Gupta and S. Singh, "Greening of the Internet," *Computer Communication Review*, vol. 33, no. 4, pp. 19–26, 2003.

[60] ——, "Dynamic ethernet link shutdown for energy conservation on ethernet links," in *Communications, 2007. ICC '07. IEEE International Conference on*, June 2007, pp. 6156 –6161.

[61] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "Elastictree: Saving energy in data center networks," in *USENIX NSDI*, 2010.

[62] S. Herreria-Alonso, M. Rodriguez-Perez, M. Fernandez-Veiga, and C. Lopez-Garcia, "A power saving model for burst transmission in energy-efficient ethernet," *Communications Letters, IEEE*, vol. 15, no. 5, pp. 584 –586, May 2011.

[63] IEEE 802.3az, "Energy efficient Ethernet Task Force," *http://grouper.ieee.org/groups/802/3/az*.

[64] S. Jamin, S. J. Shenker, and P. B. Danzig, "Comparison of measurement-based admission control algorithms for controlled-load service," in *Proceedings of the INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution*, ser. INFOCOM '97. Washington, DC, USA: IEEE Computer Society, 1997, pp. 973–. [Online]. Available: http://portal.acm.org/citation.cfm?id=839292.843046

[65] T. Karagiannis, M. Molle, M. Faloutsos, and A. Broido, "A nonstationary poisson view of internet traffic," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 3, March 2004, pp. 1558 –1569 vol.3.

[66] K. J. Kim, S. Jin, N. Tian, and B. D. Choi, "Mathematical analysis of burst transmission scheme for ieee 802.3az energy efficient ethernet," *Performance Evaluation*, 2012.

[67] A. A. Kist and A. Aldraho, "Dynamic topologies for sustainable and energy efficient traffic routing," *Comput. Netw.*, vol. 55, no. 9, pp. 2271–2288, Jun. 2011. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2011.03.008

[68] R. Lent, "A sensor network to profile the electrical power consumption of computer networks," in *GLOBECOM Workshops (GC Wkshps), 2010 IEEE*, dec. 2010, pp. 1433 –1437.

[69] S. Lima, P. Carvalho, and V. Freitas, "Admission Control in Multiservice IP Networks: Architectural Issues and Trends," *IEEE Communications Magazine*, vol. 45, no. 4, pp. 114–121, Apr. 2007.

[70] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "Energy aware network operations," in *Proceedings of the 28th IEEE international conference on Computer Communications Workshops*, ser. INFOCOM'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 25–30. [Online]. Available: http://dl.acm.org/citation.cfm?id=1719850.1719855

[71] ——, "A power benchmarking framework for network devices," in *Proc. of NETWORKING 2009*, May 2009.

[72] M. Marsan, A. Anta, V. Mancuso, B. Rengarajan, P. Vasallo, and G. Rizzo, "A simple analytical model for energy efficient ethernet," *Communications Letters, IEEE*, vol. 15, no. 7, pp. 773 –775, July 2011.

[73] I. Mitrani, "Managing performance and power consumption in a server farm," *Annals of Operations Research*, pp. 1–14, 2011. [Online]. Available: http://dx.doi.org/10.1007/s10479-011-0932-1

[74] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing network energy consumption via sleeping and rate-adaptation," in *NSDI'08: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*. Berkeley, CA, USA: USENIX Association, 2008, pp. 323–336.

[75] C. Panarello, A. Lombardo, G. Schembra, L. Chiaraviglio, and M. Mellia, "Energy saving and network performance: a trade-off approach," in *e-Energy 2010 - First International Conference on Energy-Efficient Computing and Networking*, April 2010.

[76] H. G. Perros and K. M. Elsayed, "Call Admission Control Schemes: A Review," *IEEE Communications Magazine*, vol. 34, no. 11, pp. 82–91, Nov. 1996.

[77] R. Rajaraman, "Topology control and routing in ad hoc networks: a survey," *SIGACT News*, vol. 33, no. 2, pp. 60–73, 2002.

[78] S. . Report, "Enabling the low carbon economy in the information age," Tech. Rep., 2008. [Online]. Available: http://www.theclimategroup.org

[79] P. Reviriego, J. Hernandez, D. Larrabeiti, and J. Maestro, "Performance evaluation of energy efficient ethernet," *Communications Letters, IEEE*, vol. 13, no. 9, pp. 697 –699, Sept. 2009.

[80] P. Reviriego, J. Maestro, J. Hernandez, and D. Larrabeiti, "Burst transmission for energy-efficient ethernet," *Internet Computing, IEEE*, vol. 14, no. 4, pp. 50 –57, July-Aug. 2010.

[81] S. Ryu, C. Rump, and C. Qiao, "Advances in internet congestion control," *Communications Surveys Tutorials, IEEE*, vol. 5, no. 1, pp. 28 –39, 2003.

[82] G. Sakellari, "The Cognitive Packet Network: A Survey," *The Computer Journal: Special Issue on Random Neural Networks*, June 2009, doi: 10.1093/comjnl/bxp053.

[83] G. Sakellari and E. Gelenbe, "A distributed admission control mechanism for multi-criteria qos," in *IEEE GLOBECOM 2010 Workshop on Advances in Communications and Networks, Miami, Florida, USA, 6-10 December 2010*.

[84] P. Santi, "Topology control in wireless ad hoc and sensor networks," *ACM Comput. Surv.*, vol. 37, no. 2, pp. 164–194, 2005.

[85] G. Swallow, "Mpls advantages for traffic engineering," *Communications Magazine, IEEE*, vol. 37, no. 12, pp. 54 –57, dec 1999.

[86] N. Vasic and D. Kostic, "Energy-aware Traffic Engineering," EPFL, Tech. Rep., 2008.

[87] W. Vereecken, W. Van Heddeghem, M. Deruyck, B. Puype, B. Lannoo, W. Joseph, D. Colle, L. Martens, and P. Demeester, "Power consumption in telecommunication networks: overview and reduction strategies," *Communications Magazine, IEEE*, vol. 49, no. 6, pp. 62 –69, June 2011.

[88] J. Widmer, R. Denda, and M. Mauve, "A survey on tcp-friendly congestion control," *Network, IEEE*, vol. 15, no. 3, pp. 28 –37, May 2001.

[89] M. Yamada, T. Yazaki, N. Matsuyama, and T. Hayashi, "Power efficient approach and performance control for routers," in *Communications Workshops, 2009. ICC Workshops 2009. IEEE International Conference on*, 14-18 2009, pp. 1 –5.

[90] B. Zhai, D. Blaauw, D. Sylvester, and K. Flautner, "Theoretical and practical limits of dynamic voltage scaling," in *Design Automation Conference, 2004. Proceedings. 41st*, July, pp. 868–873.

[91] M. Zhang, C. Yi, B. Liu, and B. Zhang, "Greente: Power-aware traffic engineering," in *Network Protocols (ICNP), 2010 18th IEEE International Conference on*, oct. 2010, pp. 21 –30.

[92] Y. Zhang, P. Chowdhury, M. Tornatore, and B. Mukherjee, "Energy efficiency in telecom optical networks," *Communications Surveys Tutorials, IEEE*, vol. 12, no. 4, pp. 441 –458, quarter 2010.

[93] R. Zhang-Shen and N. McKeown, "Designing a predictable internet backbone network," in *HotNets III*, 2004, pp. 58–64.

Declaration of originality

This thesis is submitted for the degree of Doctor in Philosophy in the Department of Electrical and Electronic Engineering at Imperial College London. This is to certify that the research work reported in this thesis is the product of the author's work, unless otherwise acknowledged, carried out in the Department of Electrical and Electronic Engineering at Imperial College London. No part of this thesis has been submitted in support of a degree or qualification of this or any other educational establishment.

Copyright declaration