Computer Communications and Networks

Adrian Popescu *Editor*

Greening Video Distribution Networks

Energy-Efficient Internet Video Delivery

Springer

Computer Communications and Networks

The **Computer Communications and Networks** series is a range of textbooks, monographs and handbooks. It sets out to provide students, researchers, and non-specialists alike with a sure grounding in current knowledge, together with comprehensible access to the latest developments in computer communications and networking.

Emphasis is placed on clear and explanatory styles that support a tutorial approach, so that even the most complex of topics is presented in a lucid and intelligible manner.

More information about this series at http://www.springer.com/series/4198

Adrian Popescu
Editor

Greening Video Distribution Networks

Energy-Efficient Internet Video Delivery

Springer

Editor
Adrian Popescu
Blekinge Institute of Technology
Karlskrona
Sweden

ISSN 1617-7975 ISSN 2197-8433 (electronic)
Computer Communications and Networks
ISBN 978-3-319-71717-3 ISBN 978-3-319-71718-0 (eBook)
https://doi.org/10.1007/978-3-319-71718-0

Library of Congress Control Number: 2017961100

Printed on acid-free paper

This Springer imprint is published by the registered company Springer International Publishing AG part of Springer Nature
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Foreword

The concept of the CONVINcE project grew out in 2013 of two observations: the Internet's carbon footprint was expected to exceed in 2020 that of air travel by a factor of two and the Internet traffic will be driven by video. Should people not be convinced of the necessity to reduce carbon dioxide emission, they will need to react to the economic aspect of the problem coming from the price of electricity, which will irremediably increase in the future.

Made up of 16 partners from four European countries, CONVINcE is a Celtic-Plus project addressing the challenge of reducing the power consumption in IP-based Video Distribution Networks (VDN). Partners followed an end-to-end approach, from the headend where contents are encoded and streamed to the terminals where they are consumed, embracing the Content Delivery Networks (CDN) and the core and access networks.

Partners' efforts concentrated on several topics. First on architectures, with a theoretical approach aiming at defining trade-offs between energy consumption and final user's Quality of Experience (QoE). Then, several concrete use cases were studied and some of them implemented. Particular focus was given to energy savings in mobile terminals, edge-cloud and green routing in the core network. Prefiguring tomorrow's networks, virtualization and Software Defined Networks (SDN) were not dismissed, and their influence on power consumption was assessed as well. In order to have accurate measurements of power consumption of devices, hardware and software measurement tools were developed. Industrial partners also conducted a techno-economic analysis with the objective to estimate potential energy savings brought by the project and subsequent gains for service and network operators. CONVINcE project contributed additionally to four different standardization forums: 3GPP, IEEE, IETF and MPEG. In these forums, the project contributed to six different working groups with 38 technical contributions.

An important conclusion of the project is that there is no 'magic solution' solving all consumption issues from the headend to the terminal. Optimization is the sum of several—even small—contributions properly implemented to save energy. Another important result of the project is that you must sometimes accept to consume more energy in one part of the system to save a lot in other parts. This is

typically the case where using a new encoding standard HEVC—in the headend increases significantly the energy consumption in this part, but divides by two the bitrate, leading to substantial savings in networks and terminals, with a high positive impact on the end-to-end consumption.

This book is a useful guide for those who want to reduce energy consumption in IP-based VDNs. Most of the chapters are direct results of the CONVINcE project, showing that use of new approaches and technologies can actually reduce the energy consumption in these networks, contributing so to reducing the carbon footprint of the Internet. I would like to thank the authors for their valuable contribution to this huge task.

Raoul Monnier
CONVINcE Project Coordinator, Harmonic France
Cesson-Sevigne, France
email: raoul.monnier@harmonicinc.com
URL: http://convince.wp.tem-tsp.eu/

Preface

Introduction

Internet video delivery refers to Video Distribution Networks (VDNs). These are networks composed of several parts that together create a chain model from encoding and packaging the content, its transport and distribution to end users and to its consumption in terminals. The most important functionalities in these networks are video encoding/transcoding, adaptive bit streaming, core/metro networks, access networks, Content Delivery Networks (CDNs) and routing protocols.

Given the huge increase in video traffic expected in the next few years, this means one needs to solve associated problems like tempering the significant increase expected in energy consumption and to provide the expected end user Quality of Experience (QoE). For instance, it is mentioned that the European Council has set forth an important target in form of the so-called '20 20 by 2020' initiative, which means the goal is to reduce the greenhouse gases by 20% in European Union (EU) as well as to obtain a 20% share of renewable energies in EU. This further means there is a strong need to consider problems related to this, to study and to solve them.

Reducing energy consumption in VDNs also means addressing topics like software best practices and eco-design as well as power and Quality of Experience (QoE) based design.

Towards this goal, the Celtic-Plus project CONVINcE, financed 2014–2017, with participating industrial and academic partners from four European countries (France, Finland, Sweden and Turkey) brings in important contributions. This book reports an overview of the main research results obtained in CONVINcE.

Organization

The book is organized in a number of ten book chapters and a Foreword chapter that cover several of the most important elements of Video Distribution Networks. Nine book chapters have been written by people involved in the CONVINcE project, working with different elements and aspects of Video Distribution Networks. Furthermore, one more book chapter has been written by researchers at the University of Genoa, Italy and University of Malta. Particular focus has been given to elements like architectures, models, video encoding and decoding, mobile terminals, wireless sensor networking, Software Defined Networking (SDN) and techno-economic aspects.

Features

Based on the information regarding existing literature, one can state that today there is no other similar book. The uniqueness of the results reported in the book combined with the relevance and importance of the topic gives a unique position. Furthermore, it is expected that the experience obtained in the CONVINcE project will open up for other future relevant projects focused on the stringent problem of reducing the energy in Video Distribution Networks.

Target Audience

The book is addressed to several categories of readers: college students, engineers, researchers, networking scientists, computer scientists and industry people. The broad area of topics covered by the book combined with the practical insight creates good premises for opening the interest of many people involved in computer and telecommunication systems.

Karlskrona, Sweden Adrian Popescu

Contents

Abbreviations

3GPP	3rd Generation Partnership Project
ACM	Association for Computing Machinery
API	Application Programming Interface
AVC	Advanced Video Coding
CAM	Video CAmera attached to a Computer
CCN	Content Centric Networking
CDN	Content Distribution Network
Cloud Computing	Internet-based computing
CONVINcE	Consumption OptimizatioN in VIdeo Networks
CPU	Central Processing Unit
DC	Data Center
DNS	Domain Name System
Edge Cloud	Optimizing cloud computing by performing data processing at the edge of the network
GPU	Graphics Processing Unit
GSM	Global System for Mobile Communication
H.323	Protocol for setup, management and termination of a media session
HEVC	High Efficiency Video Coding
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTP-DASH	Dynamic Adaptive Streaming over HTTP
HW	Hardware
I/O	Input/Output
ICT	Information and Communication Technology
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IoT	Internet of Things
IP	Internet Protocol
IPTv	Internet Protocol Television

IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISO	International Standards Organization
ITU	International Telecommunications Union
LLN	Low-Layer Network
MPEG	Moving Picture Experts Group
MVC	Multi-View Coding
NFV	Network Function Virtualization
OTT	Over The Top
PSNR	Peak Signal-to-Noise Ratio
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
RBG	Red/Blue/Green
SDN	Software-Defined Networking
SIP	Session Initiation Protocol
SLA	Service Level Agreement
SNMP	Simple Network Management Protocol
SVC	Scalable Video Coding
TCP	Transport Control Protocol
TV	Television
UHD	Ultra-High-Definition television
VDN	Video Distribution Network
VDS	Video Data Specification
VoD	Video on Demand
VoIP	Voice over IP
VPN	Virtual Private Network
VQM	Video Quality Model
WAN	Wide Area Network
Wi-Fi	Technology for Wireless Local Area Networking based on using IEEE 802.11 standards
WLAN	Wireless Local Area Network
WMSN	Wireless Multimedia Sensor Network
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network

Chapter 1
Video Distribution Networks: Architectures and System Requirements

Adrian Popescu, Yong Yao and Dragos Ilie

1.1 Introduction

Video Distribution Systems (VDS) are basically a set of audio and video devices, peripherals and equipments for interfacing, switching, routing, and distribution of audio, video, and picture signals sourced from multiple devices and distributed to geographically dispersed multiple devices. VDS provide distribution services of different categories like one to many, many to one, and many to many. A set of specifications and design guidelines are used to define these systems. Technical solutions with business value are expected to provide good mixture of performance, repeatability, and easy-to-use facilities. Because of this, VDS have today almost completely moved from switched networks to IP networks, also combined with the use of IP-capable endpoints and H.323 that defines the protocols to provide audiovisual communication sessions on any packet network. At the same time, the enterprise voice over IP (VoIP) environment continues the massive transition to Session Initiation Protocol (SIP) for functions like call setup, management, and termination of multimedia sessions, also requiring interworking with H.323-based video endpoints.

The current Video Distribution Networks (VDNs) and the Internet are today under big pressure to provide exponential growth in bandwidth demands, mainly because of very large traffic demands associated with TV services like IPTV as well as the appearance of new streaming services like Netflix and Skype-like video communications. At the same time, the Internet has democratized the processes of creation, distribution, and sharing of user-generated video contents through services such as YouTube and Hulu. The situation is further complicated by new developments in the form of adoption of new better video formats, requesting more

A. Popescu (✉) · Y. Yao · D. Ilie
Faculty of Computing, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden
e-mail: adrian.popescu@bth.se

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0_1

bandwidth like, e.g., the Ultrahigh Definition (UHD) defined and approved by the International Telecommunications Union (ITU).

The video traffic increase associated with the streaming services is dramatic. For instance, the so-called Cisco Visual Networking Index (VNI) indicates that today, the video traffic amounts to 90% of the global consumer traffic and also that most data communication is expected to become wireless by 2018 [1].

Another important development is the emergence of versatile apps for TV. Users are expecting a rich mixture of Video on Demand (VoD), live, and other advanced video services to be provided on an increasing number of mobile devices like smartphones and tablets, anywhere, anytime, any device, also with minimum energy consumption. At the same time, content providers are expected to be able to seamlessly operate across mobile and TV networks through app-like interfaces. The interplay between services and TV hardware is expected to increase. New facilities like voice and gesture recognition and hyper personalization are expected to become popular as well. Furthermore, the appearance of new bandwidth-demanding video streaming services like Netflix further complicated the situation.

To cope with the bandwidth growth, the shift to wireless, and to solve other issues (e.g., security), new architectures have been suggested to support a wide range of services, but unfortunately no focus has been given yet to video distribution. This is clearly observed at new suggested architectures like Content-Centric Networks (CCN) [2] and content-based extensions to Software-Defined Networking (SDN) [3]. The conclusion is that there is need for sustained research and development activities on IP-based VDN.

Connected with this, several important questions need to be answered regarding, e.g., future architectures for video delivery with reduced energy consumption, provision of real-time guarantees for live and interactive video streams, subjective, and objective metrics for performance measurement and evaluation, and performance optimization [4].

The book chapter is organized as follows. Section 1.2 presents the high-level architecture of a video distribution system. Section 1.3 presents the main categories of Low-Layer Network architectural solutions. Section 1.4 presents Content Distribution Networks (CDN). Section 1.5 presents the video distribution networks. Section 1.6 presents some of the most important video applications. Section 1.7 presents the middleware used to control the video architectures. Finally, Sect. 1.8 concludes the book chapter.

1.2 High-Level Architecture

High-level architecture of a system is defined to be a general-purpose architecture that describes the system considered for study. The goal is to define a common framework to support analytical studies, simulations, experiments and interoperability. That means such architecture must provide elements for, e.g., analytic

simulations, test, and evaluation. Among others, the high-level architecture of a system must provide information about the system model and also information about requirements, assumptions, and different constraints. In a larger perspective, aspects related to business models and deployment elements may become relevant as well.

1.2.1 Basic Operations

The creation of video content and the distribution over IP networks is a sophisticated process that follows a chain model from the acquisition at the video source, through the production and packaging of the content to the final element of distribution to viewers [5]. A video distribution system contains therefore several parts, which are about media contribution, the primary distribution, and the secondary distribution.

Video contribution refers to functions like capturing and initial processing of the video content as well as the initial transport prior to distribution. For example, the video content may be captured at a football stadium and it may be transported to a central broadcast network facility, which is placed in another location. The video streams used in this phase may be compressed or uncompressed, depending upon the particular situation in terms of the existing bandwidth and the particular demands for quality. The distribution video is sent from the central broadcast facility to other broadcast facilities for ultimate transmission to end users. Distribution configurations of type one to one (unicast) and one to many (multicast) are typically used in this case.

There are two categories of distribution systems, the primary distribution and the secondary distribution [1, 5]. The primary video distribution providers are responsible for the transport of video content from the production entity to the secondary distribution entity for ultimate transmission to end users. Primary distribution services are normally using compressed video formats, e.g., MPEG-2 or MPEG-4 or JPEG 2000. The secondary video distribution providers manage the transport of video content from the primary video distribution providers to the end consumers. Examples of secondary distribution services are IPTV, cable TV, and video streaming. Secondary distribution services are normally compressed with MPEG-2 or MPEG-4 standards, with rates of, e.g., 2–4 Mbps for Standard Definition (SD) systems and 8–20 Mbps for high-definition (HD) systems. The distribution configurations can be of type one to one (unicast) and one to many (multicast).

Basic operations done on the video signals along the video distribution chain are video coding and compression, encapsulation, transmission, reception and decapsulation, forward error correction, and decompression. Similar to other categories of service provider networks, video over IP distribution systems are expected to provide services with a good mix of simplicity, scalability, security, manageability and cost-effectiveness. Service-level agreement (SLA) requirements for video are used for parameters like network delay, network jitter, packet loss, availability, and loss recovery.

The requirements for the greening of IP-based video distribution networks further complicate the situation. One needs to take in this case a closer look at the component elements in the primary and secondary distribution networks and to analyze the greening mechanisms used in particular cases. The major components are the access networks (with different categories of technologies of type wired and wireless), core networks, Data Centers (DCs), and storage networks used to provide Web-based services like IPTV, content distribution, and cloud-based services.

1.2.2 Video Distribution Networks

Video distribution networks are distribution networks typically deployed on top of Content Distribution Networks (CDN) and the so-called low-layer networks (Fig. 1.1). These networks are defined to be IP-based networks managed to provide the levels of service and experience, security, interactivity, and reliability as requested by multimedia services like TV, video, and audio [5].

Today, the general practice is to cache the content in major metro areas, with the help of which the performance variability introduced by the Internet transport or core peering relationships is reduced or even eliminated. Beyond the metro delivery to gateway points, the video is delivered to customers on the access networks. Also, the broadcast channels are typically distributed via satellite.

Another important element at video networks is regarding the video distribution in the network. There are two main categories of video distribution systems, which are using the Internet Protocol TV (IPTV) services or using the Over-The-Top (OTT) services [5].

A video streaming network is composed of various components in the form of hardware, software, and network entities. Examples of hardware are headend, routers, switches, and WiFi access points. The software and the network entities are related to the ways used for accomplishing the video streaming from the source side (e.g., headend) to the destination side (e.g., a mobile terminal). Important elements used in this case are video encoding/decoding algorithms, routing algorithms, edge cloud, and software-defined network (SDN).

The diversity of video streaming network solutions and components indicates the complexity existent in the design and development of optimization solutions for these networks. To alleviate such complexity, a high-level architecture model is used as indicated in Fig. 1.1.

As observed in this figure, the high-level architecture is composed of four different layers and a middleware entity used to control things:

- Applications (App)
- Video Distribution Network (VDN)
- Content Distribution Network (CDN)
- Low-layer network (LLN)
- Middleware

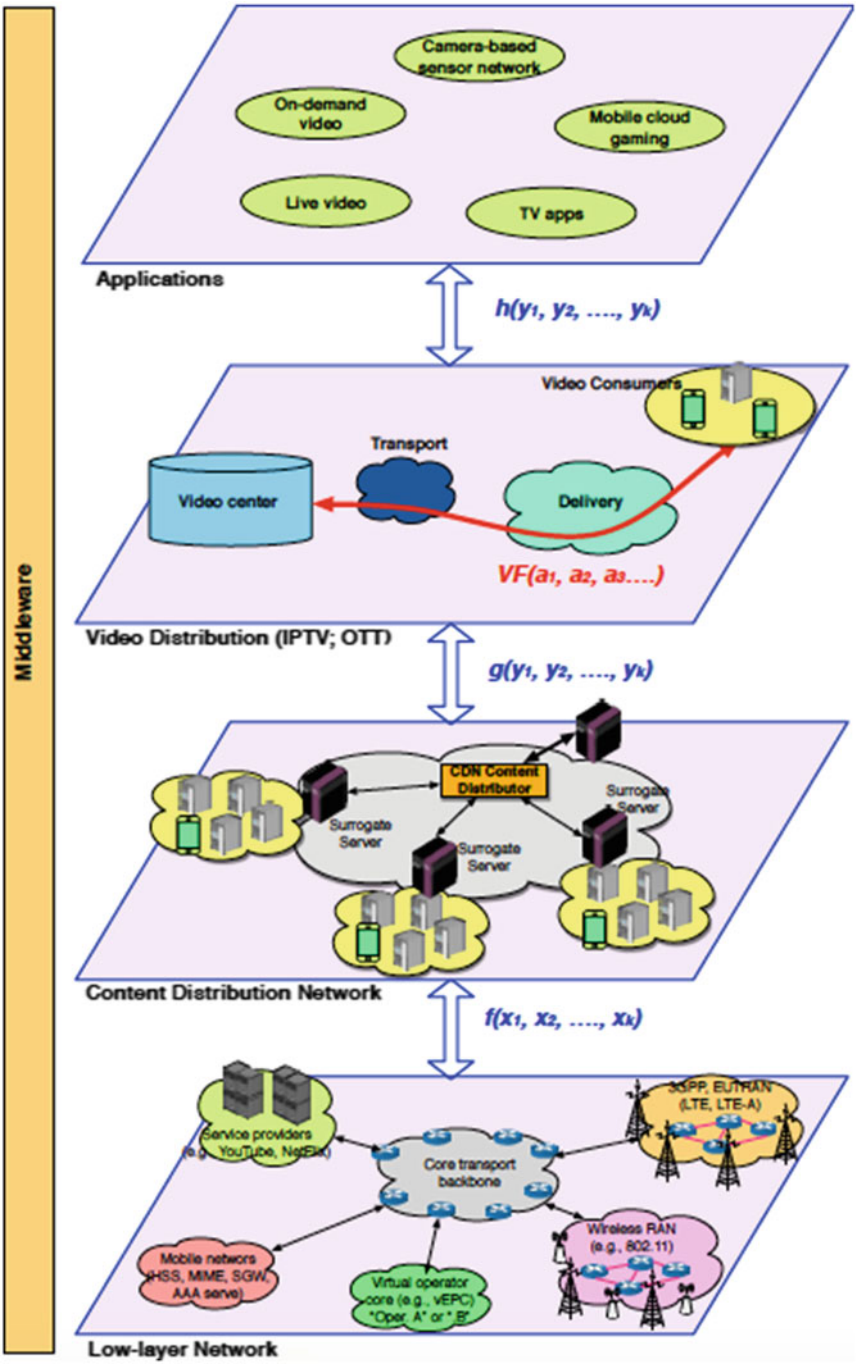


Fig. 1.1 Video distribution networks high-level architecture

As observed in Fig. 1.1, the main characteristics of video distribution networks are diversity and complexity.

A more detailed description of these layers is as follows.

1.2.3 Low-Layer Network

The so-called low-layer network (LLN) refers to L1 (Physical layer), L2 (Data link layer) and L3 (Network layer) in the TCP/IP protocol stack.

Different categories of media and communication systems can be used at low-layer like, e.g., Ethernet, WiFi, coaxial cable, power lines, fiber optics as well as different wireless communications and configurations like WLAN (IEEE 802.11), WPAN (IEEE 802.15), and LR-WPAN (IEEE 802.15.4).

Based on these technologies, different home automation and entertainment services can be offered like Netflix and YouTube.

The most popular architectures are

- Non-cloud-based architecture
- Edge cloud based architecture, and
- Software-defined networking/network function virtualization (SDN/NFV) architecture

Short description of these architectures is as follows.

1.2.3.1 Non-cloud-Based Architecture

The main elements of this architecture (Fig. 1.2) are the video encoding/transcoding entity together with adaptive bit streaming, core/metropolitan networks, routing protocols, fixed and mobile terminals.

Minimizing the energy consumption in this architecture means that the energy is minimized for each individual networking element, typically without giving consideration to other networking elements. Furthermore, transversal solutions can be considered as well like, e.g., software best practices and eco-design.

1.2.3.2 Edge Cloud Based Architecture

The edge cloud based architecture is an architecture, where instead of using a cloud infrastructure at remote, core-deep locations, several small clouds are used at the network edge in partnership with the edge Internet Service Providers (ISPs), as shown in Fig. 1.3. By doing so, the intelligence is shifted away from the network core toward the network edge, also combined with facilities for storage and other

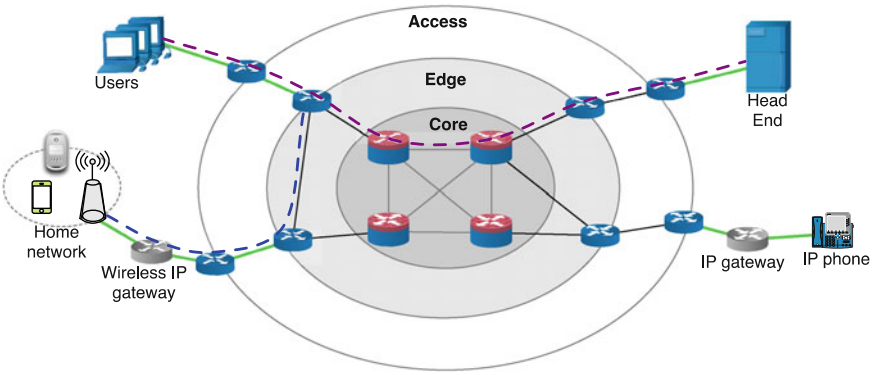


Fig. 1.2 Non-cloud-based architecture

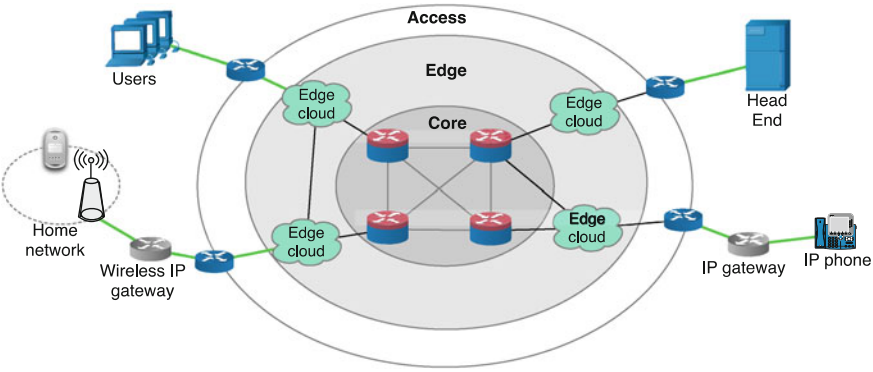


Fig. 1.3 Edge cloud based architecture

functionalities. Important advantages are in the form of offloading energy-intensive operations in access networks, headend, and terminals like, e.g., encoding/transcoding video content, 2D/3D graphical computation, and radio communication in base stations (BSs).

Basically, this architecture is a distributed cloud system complemented with a central cloud to provide services like failback and centralized infrastructure-as-a-service (IaaS) facilities. The distributed cloud system can be used for offloading complex and high energy consuming terminal operations like displaying and communication with the server. The model can be also used as a platform for CDN-like functionality.

Using edge cloud facilities solves several networking limitations like offloading energy-intensive operations in access networks, headend, and terminals. The drawback is because of additional communication needed between terminals and edge cloud, particularly when the communication occurs frequently. This problem

can however be solved by finding out appropriated trade-offs among involved parameters [6].

For instance, video transcoding and media processing in edge clouds are an interesting alternative to processing in the central headend multiple streams at different qualities and bit rates, such as to cope with the variety of terminals and the varying QoS in the networks, as it is done today with an Adaptive Bit Rate (ABR) approach. The headend produces in this case a single high-quality stream, which can be further transcoded "on the fly" in the edge cloud. This provides the advantage of less traffic load in the network combined with good stream quality offered to end users. The drawback in this case is because of additional communication introduced between terminals and edge cloud.

Edge computing makes it possible to increase the quality of services by using the so-called context adaptation, which is a way of profiling within a local context, without compromising the privacy. Today, there is a lot of work on this, known under different denominations like, e.g., cloud computing, edge computing, fog computing, and mobile edge computing [7]. The main question in this case is about the best management system for edge resources such as to provide best performance for end users combined with minimum energy consumption. Parameters like resource type and use combined with the particular objectives are considered in this case.

1.2.3.3 SDN/NFV Based Architecture

Software-Defined Networking (SDN) and the associated Network Function Virtualization (NFV) are basically a new networking paradigm where the control and the forwarding data planes in the network are separated. Connected with this, the OpenFlow technology is suggested to standardize the way a controller communicates with network devices [8].

Basically, SDN separates the networking functions into two groups, the data plane (responsible for data forwarding, based on local decisions) and the control plane (responsible for centralized decision-making algorithms that determine optimal paths for data packets). SDN is focused on optimizing the transport of data and programmability facilities used to centrally control traffic flows, to partition the networks as well as to provide application-level QoS. On the other hand, NFV focuses on implementing network functions in software, to support multi-versions and multi-tenancy, to deploy provisioning and control network functions, and to virtualize and streamline the network function deployment. SDN and NFV are combined to create flexible networks, and they are not dependent on each other. SDN allows programmability of the NFV infrastructure to support development and deployment of network functions whereas NFV allows decoupled network capacity and functionality to create network functions that can be dynamically created and deployed.

It is important to distinguish between NFV and network virtualization. NFV focuses on virtualizing traditional network services into entities referred to as virtual network functions (VNFs) and mandates the use of off-the-shell commodity servers and switches. Network virtualization is used to implement logical networks decoupled from the underlying hardware, which provides a new level of flexibility and agility in network management. One can consider NFV as an approach to build up complex network services by chaining VNFs, where network virtualization provides the means to interconnect the VNFs in an optimal way.

Virtualization also applies to video encoding/transcoding. By this, the hardware resources are adapted in a flexible way to encoding/transcoding requests, especially when these resources are in the edge or for live events where picks of connection to video services are experienced.

NFV is used to transform the existing network functions or services into software applications, to consolidate them and to run them on virtual machines based on commercial off-the-shelf hardware. NFV requires carrier-specific virtualization software with high performance on real-time, scalability, reliability, and availability. The problem is that this demands for large investments.

It must be also emphasized that OpenStack and SDN/NFV are not competing solutions, but rather complement each other. OpenStack focuses on cloud orchestration, where the network is just a part of the platform's set of features. This approach is suitable for service providers offering computation and storage services. SDN/NFV aims to bring more flexibility in network management, efficient use of network resources, and target in particular network operators. OpenStack is now able to support OpenFlow and SDN (through the OpenDaylight project [9]) as plug-ins to its network-as-a-service module.

The SDN/NFV architecture provides several advantages like centralized control over the network, quicker network configuration, decoupling of the system that makes decisions from the underlying system that forwards the traffic as well as openings for energy savings in the network. By providing programmability on traffic and devices, increased flexibility and efficiency can be obtained, also with regard to energy savings in the network. Energy saving is particularly relevant for optical fiber networks, which are typically used with Multiprotocol Label Switching (MPLS). By using SDN in a MPLS environment, the complexity of the dynamic control plane is reduced, allowing thus for flexible service creation [8]. For instance, today different control mechanisms are used for operations regarding to find available paths, to reserve bandwidths, to monitor paths, and to maintain guarantees like, e.g., Open Shortest Path Forward Traffic Engineering (OSPF-TE), Reservation Protocol Traffic Engineering (RSVP-TE), Internal Boarder Gateway Protocol (I-BGP), and MultiProtocol Boarder Gateway Protocol (MP-BGP). The routers are requested in this case to provide sufficient CPU capacity and memory to support these protocols, and this is against the well-known end-to-end (e2e) principle used in Internet. On the other hand, the use of SDN and OpenFlow controllers eliminates the need for this. The consequence is also that the applications interfacing to the controller implement the functions provided by the router resident

protocol code. The energy requirements are therefore reduced due to lower need for memory and computing capacity, combined with less frequent updates [8].

While the application of SDN to wired/optical scenarios is already becoming a reality, things are more difficult in the case of wireless networks. This is especially because of the large diversity of wireless technologies existing today. Figure 1.4 shows an example of SDN/NFV-based architecture of a mobile network operator, where a dashed line indicates a control plane connection and a solid line indicates a user plane connection [10]. As shown in this example, a mobile telecom operator may own and manage one or more heterogeneous Radio Access Networks (RANs), all of them connected to a common core network. Wireless backhaul networks may be involved as well to connect the base stations (BSs) to a core network. Access and/or core network component elements are enhanced in SDN/NFV with programmability facilities to support multiple functionality levels. For instance, L2 switches and L3 routers can become programmable. Also, multiple operators might physically share the transport core network, the backhaul, or the radio part. Key interfaces are in this case the so-called northbound interface (to virtual operators as well as to service and application providers) and the southbound interface (to physical user plane network entities in the core transport backbone, to physical user plane entities in the RAN as well as to mobile nodes) [11]. It is also important to mention that, besides architectures that assume persistent connectivity to cloud or to edge cloud, ad hoc mobile clouds can be developed as well. These are architectures where the neighboring mobile devices are pooled together for resource sharing.

It is also observed (in Fig. 1.4) the presence of Home Networks (HM). These are particular networks that facilitate communication and interoperability among devices present inside or close to a home. These networks are provided with increased functionalities to provide facilities for interaction and to increase so the quality of life at home. Different categories of media can be used in this case like, e.g., Ethernet, WiFi, coaxial cable, power lines, fiber optics (e.g., fiber-to-the-home

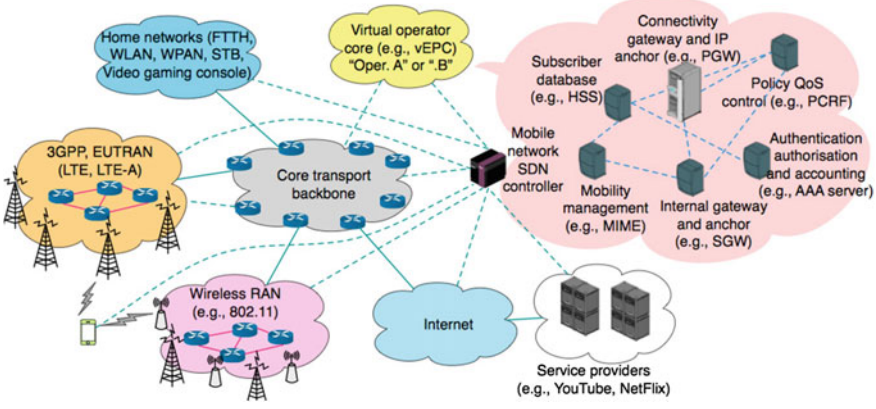


Fig. 1.4 SDN/NFV based network

(FTTH)). Wireless configurations like WLAN (IEEE 802.11), WPAN (IEEE 802.15), and LR-WPAN (IEEE 802.15.4) can be used in the home environment. Based on this, home automation and entertainment services can be offered like, e.g., Netflix and YouTube.

Another important element that must be considered is the so-called *sliced networks*, which is about sharing the network resources among multiple providers. Typically, a user creates in this case a so-called slice and delegates the control to a third party. Every slice is defined to be isolated from other users in terms of traffic, bandwidth, and control. Furthermore, the creation of slices is the requirement needed for opening the network control to more end users to innovate on the network at a time.

1.2.4 Content Distribution Network

On top of the low-layer network, there is the Content Distribution Network (CDN) layer. This is basically a large system of geographically distributed specialized cache servers deployed in different data centers across the Internet, combined with the use of a special routing code to redirect media requests to the closest server, as shown in the example in Fig. 1.5 [11].

CDN is a networking solution where high-layer network intelligence is used to improve the performance in delivering Web and media content over the Internet. An appropriated software architecture is used for this. The main functional elements of CDN are content distribution, request routing, content routing, content processing, authorization, authentication, and accounting. With regard to content delivery, there are three main techniques: HTTP redirection, IP redirection, and DNS redirection (most effective, but need for using DNSSEC) [5].

Furthermore, there are two main categories of content distribution networks, which are based on the way the content is delivered to end user. These are the cache-based CDN and the peer-to-peer (P2P) *C*DN [5].

The cache-based CDN is advantageous for edge cloud architecture, whereas the peer-to-peer CDN solution uses overlay networks and associated routing for content distribution.

The fundamental concept of distributing the content to cache servers located close to end users results in better performance with reference to maximizing the bandwidth, minimizing the latency and jitter, and improved accessibility as well as better balance between the cost for content providers and the QoE for customers. The benefits of using CDN are substantial to Web users, content and application owners as well as network to service providers.

An important issue is to investigate the energy consumption and savings for two distinct cases regarding Data Centers (DCs). These are the case of locating DCs close to end users, which minimizes the transport energy requirements, and the case with centralized DCs, which minimizes the storage energy requirements.

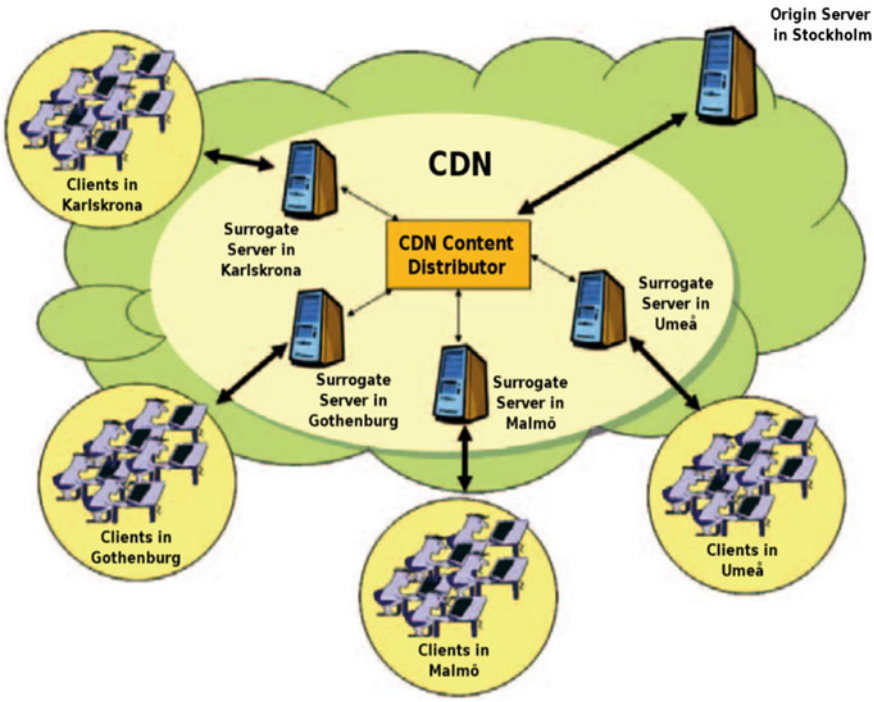


Fig. 1.5 Content distribution network

The combination of cloud-based servers for video content management and distributed client software for optimal enterprise video delivery provide together high-quality business video to end users. Most CDNs are typically operated as an Application Service Provider (ASP) on the Internet.

The owners of high traffic Web site and Internet Service Providers (ISP) hire the services of the CDN company that provides content delivery. On the other hand, the CDN company pays ISP, carriers, and network operators for hosting its servers in their Data Centers (DC). At the same time, content owners like media companies and e-commerce vendors pay CDN operators for delivering their content to end users. Another important observation is that today an increasing number of Internet operators build up their own CDN (like Akamai, Microsoft Azure, Amazon CloudFront) with the goal to improve the quality of content delivery, reduce demands on the own network infrastructure as well as to generate revenues from content consumers. The conclusion therefore is that CDN is today associated with quite sophisticated business models, which need to be investigated and modeled.

1.2.5 Video Distribution Network

Video networks are distribution networks typically deployed on top of CDN networks. They are basically IP-based networks managed to provide the levels of service and experience, security, interactivity, and reliability requested by multimedia services like TV, video, and audio. The basic operations are capturing and initial processing of video content, initial transportation prior to distribution, primary and secondary distribution, and finally delivery to end users.

The networks can be of three categories, namely terrestrial or satellite or cable television networks. The first category is also known as broadband networks and it is based on using the IP technology. These networks are targeting individual users although multicast and broadcast mechanisms can turn them into multicast networks as well.

The video signal is collected at, e.g., a football stadium and transported in compressed form to some broadcast facility that may be placed in another location [5]. The video is then sent to a number of secondary broadcast entities for further transmission to end users. Examples of distributions networks are IP-based TV networks (IPTV) and Over-The-Top (OTT) networks. Networks like terrestrial or satellite networks can be used as well. A multidimensional process of treatment the video content is used, where the most important elements are content acquisition, content exchange, and content distribution. Basic operations are video coding and compression, encapsulation, forward error correction, transmission, reception and decapsulation, error correction, and decompression [5].

Video distribution networks are requested to provide services with a good mix of simplicity, scalability, security, manageability, and cost-effectiveness. Service-Level Agreement (SLA) requirements for video are used to define the service requirements, which are about parameters like network delay, network jitter, packet loss, availability, and loss recovery.

From an end user point of view, there are three categories of video delivery services, namely Live TV, Time-Shifting TV, and Video on Demand (VoD).

There are three fundamental challenges that must be considered in developing good quality video networks. These are the fact that the video is a bandwidth hungry application, the streaming of video demands for real-time consistent network performance combined with sophisticated distribution schemes as well as that the video is perceived by the end user as a continuous long-term experience. Altogether, this poses serious challenges in developing solutions for IP-based video distribution networks that are energy-efficient as well [5]. Furthermore, recent developments in the end user environment for which a complete viewing experience (so-called TV user experience UX) means connecting the Internet directly to large-screen, high-definition TV receivers influence the situation as well.

Today, the general practice is to cache the content in major metro areas, with the help of which the performance variability introduced by the Internet transport or core peering relationships is reduced or even eliminated. In other words, most video

networks are today of category metro networks. Beyond the metro delivery to gateway points, the video is delivered to customers on access networks. Also, the broadcast channels are typically distributed via satellite.

There are three categories of video distribution systems:

- Internet Protocol TV (IPTV)
- Over The Top (OTT)
- Peer to Peer (P2P) based

The first two categories are widely used today in the form of commercial solutions. On the other hand, the P2P-based system is today more of a research topic.

Short presentation of the systems is as follows.

1.2.5.1 IPTV

The Internet Protocol TV (IPTV) is basically a system used for the delivery of video via a private network, where the users are requested to subscribe to a traditional cable service (like Orange or Comcast) or satellite pay-TV service (like Dish or DirectTV) to have access to video services. In other words, this is a system where the content provider, transmission network, and receiver are all of them provided by the (local) telecom company.

The IPTV architecture is a network architecture where different elements are placed in different network categories, e.g., Super Headend (SHE) is placed in the core network, Video Hub Office (VHO)—Regional Headend (RHE) is placed in the metro aggregation and distribution network, Video Switching Office (VSO) is placed in the access network and end entities like home gateway, Set Top Boxes (STB) and TV set are placed in the residential/home network.

The basic IPTV architecture has several components: acquisition and distribution servers, video on demand (VoD) creators and servers, IP routers, and Set-Top Boxes (STB) [12]. Another important component is the IP multicast entity, which contains elements like Super Hub Office (SHO), Video Hub Offices (VHO), Intermediate Offices (IO), Central Offices (CO), and others [12].

The IPTV services are mostly multicast services for Live TV and they are using Internet Group Membership Protocol (IGMP) subscribing features combined with H.264 over RTP over UDP protocols for delivery. This means that this is generally an expensive distribution service but it has the advantage of the provision of interactivity services combined with good QoS/QoE.

On the other hand, the need to provide QoS in Internet demands for the presence of technologies and protocols that provide this possibility like, e.g., bandwidth reservation, latency management, traffic prioritization, traffic shaping, and network congestion avoidance. That means, from the service provider point of view, the main challenge in an IPTV system is to provide high-quality service at a minimum cost by using the existent delivery networks. This means that control mechanisms

like bandwidth reservation and admission control must be present to guarantee QoS demands for video.

IPTV is a closed system owned by some telco, where the purpose is to deliver licensed contents that the telco receives from the content provider. On the other hand, the subscribers can enjoy the content with the associated quality and stability by paying the fee for the content. Compared to the broadcast television services, IPTV sends only one program at a time. When a viewer changes the TV channel, a new video stream is transmitted from the server to the end viewer.

The IPTV architecture has the advantage of tight control combined with guaranteed bandwidth, but this is done at the price of inability to serve viewers wherever and whenever they want. The consequence of this is that the OTT services have now become more popular, because of less complicated architecture and also better capabilities.

1.2.5.2 OTT

The term over the top (OTT) is used to define the delivery of video content (with different video formats) over a public network (via the Internet) without requiring the users to subscribe to a traditional cable service or satellite pay-TV. The OTT apps and services are in this case simply delivered over the best-effort Internet. The content provider can be, e.g., an independent service/company, the receiver can be, e.g., purchased by consumer (box, stick, TV set, computer, mobile device) and the transmission network is provided by the public Internet and the local telecom company.

The most significant difference between IPTV and OTT is therefore that IPTV is delivered over a private controlled network whereas OTT is delivered over a public uncontrolled network. This means that it is no longer necessary to have a managed network to ensure QoS in video delivery in OTT. That also means OTT is not associated with a pay-TV service provider subscription. Furthermore, the use of Internet as a carrier provides larger audience and greater reach at lower costs.

The OTT architecture is a unicast-based architecture where the video content is delivered by broadcaster directly to the end user client. The broadband delivery is done without a multiple system operator that takes care of distribution and has control over it. Several important elements in the OTT architecture are the encoder (placed in the headend), different servers like Web streaming server and Digital Rights Management (DRM) server, Content and User Management System (to direct streaming requests to right servers) as well as the CDN (typically edge servers used for distribution). The customers can access the OTT content by using common entities like Internet-connected devices and phones (Android, iPhone, Windows type), smart TV, set-top boxes, gaming consoles, desktop and laptop computers, and tablets.

Compared to IPTV, where the control over QoS is enabled, OTT transmits streams using HTTP and it works therefore under best-effort network conditions

where no QoS guarantees are provided. End user bandwidth cannot be controlled. The use of the streaming technology in combination with CDN creates the premises needed for OTT streaming. This means that this is a low-cost service but at the price of risk for lower service quality. Furthermore, an important challenge is regarding the live broadcast, which is today not up to the mark. Examples of streaming technologies are Adaptive Bit Rate (ABR), Apple HTTP Live Streaming (HLS), Google WebM, Adobe HTTP Dynamic Streaming, Microsoft Silverlight Smooth Streaming, and MPEG-DASH. These are actually competing standards, the most important being HLS and MPEG-DASH. Basically, both of them provide ways to transport the same information but with a different syntax. MPEG-DASH is more detailed than HLS on several points like, e.g., transportation and parental rating [13].

OTT refers to content that arrives from a third party like, e.g., Netflix, YouTube, Hulu, and Amazon. In other words, the OTT streaming is done through open unmanaged Internet, whereas IPTV uses a dedicated managed network. An interesting development is that major broadcast operators like, e.g., HBO Now, CBS All Access, are now introducing OTT options.

OTT applications like WhatsUp, WeChat, Viber, Skype, and WebRTC have gained immense popularity in the last years, and this is due to the good quality of the service combined with low (often no) cost for these services. The good quality of service is a direct consequence of the fundamental concept of over-dimensioning used today in Internet.

The OTT market is expected to increase fourfold by 2019, according to recent information provided by the BroadbandTVNews. This positive development is however done at the price of the conflictual situation that may exist between companies that provide online video streaming media and VoD (e.g., Netflix) on one side and service providers (e.g., fixed-line broadband providers) on the other side. This situation is further complicated because of the complex relationship (cooperation and/or competition) existing among serving teleoperators. Basically, the conflict is directly related to the charge policies used by service providers in combination with the unpredictable consumption of the OTT services.

1.2.5.3 Peer to Peer

The Peer-to-Peer (P2P) architecture is an architecture that is less demanding with regard to hardware as it uses instead the existing resources in a temporary way. Basically, the P2P architecture can be broadly classified into two categories, each of them with own advantages and drawbacks [14]:

- Multi-tree based streaming
- Mesh-based streaming

The main advantage of the P2P architecture is in the form of simplicity, which is however obtained at the price of need for more dynamic system with the consequence of worse performance, mainly because of the presence of churn.

1.2.6 Applications

There are today a large number of applications developed on top of video distribution networks. Some of the most popular applications are TVapps (content providers embed the technology in their services such as they operate seamlessly across mobile and TV networks through common "app-like" user interfaces), interactive games (use of games for, e.g., education and learning), media distribution and business (distribution, streaming business), camera-based sensor networks, online ads and commerce (advertising, measurable, actionable, interconnection of ads and commerce), health care (digitization of patient data, pharmaceutical testing, medical records), and enterprise (cloud computing, security). In the following, the focus is laid on media streaming, mobile cloud gaming, and TV apps only, where a huge traffic increase is expected in the next 5–10 years, as indicated, e.g., by Cisco and Ericsson traffic forecast.

1.2.6.1 Media Streaming

Basically, media streaming means that the multimedia content (video and/or audio) is sent in compressed form over the Internet and played in real time instead of first being saved to the hard drive. That means, a Web user does not need to wait to first download a file and then to play it. The data stream is played as it arrives [15]. This method is different from the classical method of normal file download where the entire file is first downloaded before starting up the playback.

There are three categories of streaming delivery methods [16]:

- Traditional streaming
- Progressive download streaming
- Adaptive video streaming

A good example of traditional streaming is the Real-Time Streaming Protocol (RTSP) [17], which is a stateful protocol where the users interact with the media content provider. RTSP is used together with the Real-time Transport Protocol (RTP) [18]. The move today is however toward HTTP-based streaming where the media delivery is adapted to the Internet instead of trying to adapt the entire Internet to streaming protocols.

The progressive download streaming is a method based on downloading from a HTTP Web server. The player client allows the media to be played back while the file download is still in progress. This means the user can, e.g., pause the streaming, waiting for the download to finish, allowing so a smooth playback when the user decides to play the media. The drawback is that it is not perfect and not very flexible in selecting the parameters for downloading.

A better method for video streaming is the HTTP-based adaptive video streaming. This is basically a hybrid method of progressive download and streaming. The video is cut into short segments and encoded at the specific delivery

format and rate. The segment length may vary from implementation to implementation, but they are typically couple seconds long. It is the adaptive properties of the streaming solution that is behind the popularity of the method. Adaptive elements like available network resources (e.g., bandwidth), device facilities (e.g., display resolution, available CPU) and current streaming conditions (e.g., playback buffer size) may be used in the streaming process. With these adaptive facilities, a major benefit is that the video streaming client can adjust over time the selection of encoded segments based on, e.g., the actual communication link throughput. A good example of HTTP-based adaptive video streaming standard is the MPEG-DASH, which stands for MPEG Dynamic Adaptive Streaming over HTTP, also known as ISO/IEC 23009-1 [19]. The International Organization for Standardization (ISO) finally published MPEG-DASH in April 2012.

Live streaming requires the presence of some form of source media (e.g., video camera, audio interface), an encoder to digitize the content, a media publisher, and a CDN to distribute and deliver the content. A user also needs a player entity, which is a software that uncompresses and sends video data to display and audio data to speakers. The associated storage size is calculated from the product of streaming bandwidth and media length. Examples of streaming technologies are Adobe Flash, Apple QuickTime, Microsoft Windows Media, and Silverlight. Streaming technologies use compression to shrink the sizes of audio and video files so they can be retrieved and played by remote viewers in real time. The codecs are usually independent of the streaming technology that implements them.

The users can access the OTT content by using, e.g., Internet-connected devices like phones of different categories (Android, iPhone, Windows based), smart TV sets, set-top boxes, gaming consoles, laptops, and tablets.

Furthermore, there are several transport protocols that can be used in video streaming, like Hypertext Transport Protocol (HTTP, this is however not a streaming protocol, but used to distribute small files), Real-Time Streaming Protocol (RTSP), Real-Time Transport Protocol (RTP), Real-Time Transport Control Protocol (RTCP), Real-Time Messaging Protocol (RTMP), and Real Data Transport (RDT).

A broadband speed of at least 2 Mbps is recommended for the streaming standard definition video without experiencing buffering, especially for live video, e.g., Apple TV, Google TV, or Sony TV Blue-ray Disc Player. Under such circumstances, streaming media storage sizes of about 128 MB are needed for one hour of video encoded at 300 kbps [20]. In the case of live video with 3000 viewers, then the media storage sizes increase to almost 2×10^6 MB [20].

With reference to the streaming services offered to end users, there are two categories of services [20]:

- (Social) Live Video Streaming (LVS)
- On-Demand Video Streaming (ODVS)

In LVS systems, the same (music and) video content is disseminated to a given group of receivers in a synchronous way, at most with a (minimal) dissemination delay. The content is sent in compressed form over the Internet and displayed by users in real time. The (audio and) video stream is transmitted straight to (fixed and mobile terminals) receivers with a target to achieve low end-to-end delivery latency. The LVS chain contains a source media (e.g., video camera), an encoder used to digitize the content, a media publisher and a CDN to distribute and deliver the content as well as the receivers. As a general comment, it is observed that live video streaming over the Internet has now opened the door to the era of social streaming by applications like Facebook Live, YouTube, and Twitter, also augmented by applications like Virtual Reality (VR) and Augmented Reality (AR).

On the other hand, in ODVS systems, the receivers request the same or different video contents at different time moments and they (listen to and) watch the contents in an asynchronous way, independent of each other. Even in the case of same content distributed to different receivers at the same time moment, the playback is typically not synchronized among receivers. That means the video streams can be saved to hard disks and/or servers of different categories for different time periods. These memories are supplementary elements compared to the LVS systems.

1.2.6.2 Mobile Cloud Gaming

Mobile cloud gaming is today one of the most demanding applications with reference to the requirements put on terminal and on network as well as with reference to the complex data structures involved in this application. The basic idea is to render and to compress the gaming graphics in a gaming server, thus streaming the encoded gaming video to the mobile device through Internet and wireless access.

The mobile terminal and the game server may have different requirements in terms of hardware and/or software. The mobile users are especially interested in games that run frictionless and playable across various mobile terminals and platforms. On the other hand, the game server is expected to solve several challenges, where some of the most important ones derive from a possible large increase in the number of users running the particular games. Other important requirements are, e.g., regarding the scalability with the number of users and the load balancing over day and night period times. On the other hand, transcoding is not used in this case given that different users may have different contents.

Currently, there are two main categories of cloud-based gaming [21, 22]. These are based on video and on file streaming, respectively. Compared to the first category (video streaming), the second category of gaming (file streaming) means that the game files like, e.g., texture files, executable files, etc, are downloaded only when the mobile user plays the particular game.

1.2.6.3 TV Apps

One of the most important recent developments is in form of smart TV and TV apps. This means that the users are provided with a rich mixture of video services of different categories and characteristics, with high-definition (HD) and ultrahigh-definition (UHD) characteristics, which are run on an increasing number of mobile devices and terminals like multiscreen TV, smartphones and tablets, also under increasing demands like, e.g., anywhere, anytime, any device, and minimum energy consumption. Recent big innovations in digital media like, e.g., Netflix, Spotify, and iTunes are now entering the TV technology. New generation of TV hardware and software like, e.g., smart TV, Apple TV, Chromecast and Amazon Fire, is now entering the market as well. These systems use open, standardized operating systems that seamlessly integrate with mobile devices. They support open app stores and use cloud-based services. An expected consequence of these developments is that the future of TV is expected to be in form of TV apps, where the users expect that the content providers operate seamlessly across fixed and mobile TV by using "app-like" user interfaces. It is also expected that TV apps like Netflix, Apple TV, Facebook, YouTube, Skype, and apps dedicated to particular interest groups like, e.g., shopping, sports, will dominate the market, also combined with TV advertising.

1.2.7 Middleware

As mentioned above, video distribution networks are composed of various elements in form of hardware, software, and networking entities, which may belong to different computer systems. This means individual architectural elements must be integrated to operate together toward the goal of providing the video distribution service. Given the large diversity and complexity of video streaming network solutions and components as well as the diversity of computer systems serving the video distribution networks, this means there is need for a special software system, the so-called middleware, to allow communication and data management for distributed applications.

The middleware is essentially a software that lies between an operating system and the applications running on it. It works as a hidden translation layer, which enables communication and data management for distributed applications. The middleware includes a set of elements of different categories like, e.g., Web servers, application servers, content management systems, and other similar tools that support application development and management. Examples of middleware incorporated in cloud services are Remote Procedure Call (RPC) middleware, database middleware, transaction middleware, content-centric middleware, portals, and embedded middleware.

Services provided by middleware include, among others, transformation of message formats from one application to another, routing and switching of messages, encryption, security, and validation tasks.

There are several categories of middleware services [23]:

- Data management services (database and file system middleware)
- Communication services (Remote Procedure Call and messaging middleware)
- Distribution services (location, time, security)
- Object management services
- Application cooperation services
- Presentation services
- System management services (configuration, change, operations, problem, performance)

Depending upon the particular application, these services are accordingly adapted to provide the expected task and associated performance. For instance, particular focus is given to distribution and application distribution services in the case of video distribution networks.

The middleware provides advantages like real-time information access among systems, streamlines business processes, and maintains information integrity across multiple systems. At the same time, the middleware is associated with drawbacks like high development costs, need of much resources, time consuming implementations.

A good example of middleware element is the mobile middleware, which is a software component that can be integrated with any application to add mobility [23]. Among others, this software must provide service elements like configuration management, service discovery, and event notification. The software must fulfill requirements like

- Provide support for multiple types of mobile platforms, e.g., phone, PDA
- Address the diversity of mobile devices and provide consistent programming environment across them
- Provide different types of profile, e.g., network interface, access network, and flow description
- Provide implementation style invisible to applications
- Provide support for context awareness
- Provide security facilities
- Provide management facilities

Furthermore, there is also need for a so-called mediator layer to solve the problem of diversity of the format of Application Programming Interface (API) that may exist among elements of different types such as the API calls can be transformed from one format to another and the interactions among different elements can so be coordinated.

Another option, particularly interesting for the edge cloud architecture, is to integrate OpenStack (OpenStack) with a mediation framework such as Apache

Camel [24]. OpenStack already includes a message broker (RabbitMQ) and, with the addition of Apache Camel, one can obtain a lightweight Enterprise Service Bus (ESB). The advantage in this case is that one can leverage fully integrated services available from OpenStack such as large-scale configuration management (Puppet), data collection (Ceilometer), database-as-a-service (Trove), Hadoop data mining (Sahara), and SDN functionality through the OpenDaylight OpenStack plugin.

Finally, an important aspect to consider when designing and using the middleware is the energy consumed by the entities that use it. This energy consists of both the computational energy used to process middleware messages and of the communication energy used by networks in carrying the messages. An important goal is to reduce the energy.

1.3 Conclusion

The book chapter has presented an overview of video distribution networks and their components. The overview follows a high-level architecture scheme, where the component elements are low-layer network, content distribution network, video distribution network, applications, and middleware. Detailed information has been presented for every component element, also complemented with some technical details. Given the complexity of these networks, with many component elements, less focus has been given to technical data.

Acknowledgements This research was partially supported by the European Celtic-Plus project CONVINcE and funded by Finland, France, Sweden, and Turkey.

References

1. Cisco Video Networking Index Forecast, Global IP Traffic Growth. http://www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/index.html
2. Kurose J (2012) Content-centric networking: technical perspective. Commun AC 55(1)
3. Software Defined Networks. http://sdn.ieee.org
4. CONVINcE: Consumption Optimization in Video Networks, Celtic-Plus project. https://bscw.celticplus.eu/pub/bscw.cgi/d985/CONVINcE-start_lq.pdf
5. Popescu A (2014) Greening of IP-based video distribution networks: developments and challenges. In: 2014 Conference on Communications, Bucharest, Romania
6. Zhang X, Zhang J, Huang Y, Wang W (2013) On the study of fundamental trade-offs between qoe and energy efficiency in wireless networks. Trans Emerg Telecommun Technol 24 (2013)
7. Satyanarayanan M (2017) The emergence of edge computing. Computer 50(1):30–39
8. Lara A (2014) Network innovation using OpenFlow: a survey. IEEE Commun Surv Tutor 16 (1):2014
9. OpenDaylight: Open Source SDN Platform, collaborative project. https://www.opendaylight.org

10. Bernardos C, Oliva A, Serrano P, Banchs A, Contreras LM, Jin H, Zuniga JC (2014) An architecture for software defined wireless networking. IEEE Wirel Commun 21(3):2014
11. Sezer S, Scott-Hayward S, Chouhan PK, Fraser B, Lake D, Finnegan J, Viljoen N, Miller M, Rao N (2013) Are we ready for SDN? Implementation challenges for software-defined networks. IEEE Commun Mag 51(7)
12. Yim J, Lee G, Lee T, Jeon J (2014) Review of IPTV system architectures. Multimedia. https://doi.org/10.14257/astl.2014.46.19
13. Monnier R, Popescu A, Ljung R (2016) CONVINcE: towards power-optimized video distribution networks. In: 19th International ICIN conference, Paris, France
14. Zhang J, Xing W, Wang Y, Lu D (2014) Modeling and performance analysis of pull-based streaming schemes in Peer-to-Peer Network. Comput Commun 40:22–32
15. Unified Communications Tech Target (2017) Streaming Video. http://searchunified-communications.techtarget.com/definition/streaming-video
16. Zambelli A (2009) IIS smooth streaming technical overview. Microsoft Corporation
17. Real-Time Streaming Protocol (RTSP) (1998) RFC 2326, IETF
18. Real-Time Transport Protocol (RTP) (2003) IETF, 2003
19. Sodagar I (2011) The MPEG-DASH standard for multimedia streaming over the internet. IEEE Multim 18:62–67
20. Storage B et al (2016) Bandwidth and storage. http://www.broadbandchoices.co.uk/guides/internet/watching-tv-online
21. Platform GC (2016) Gaming solutions and online gaming. https://cloud.google.com/solutions/gaming
22. Shea R, Liu J, Ngai E, Cui Y (2013) Cloud gaming: architecture and performance. IEEE Netw 27(4)
23. Sirisha Mandadi (2015) Middleware technologies, overview. http://slideplayer.com/slide/6068906/
24. Camel A (2015) http://www.camel.apache.org

Chapter 2
Energy-Efficient Management and Control in Video Distribution Networks: 'Legacy' Hardware-Based Solutions and Perspectives of Virtualized Networking Environments

Raffaele Bolla, Roberto Bruschi, Franco Davoli and Etienne Victor Depasquale

2.1 Introduction

Energy efficiency in the Internet (and in computing and telecommunication networks in general) has become a significant problem, which has received increasing attention since the early 2000s (see, e.g. [1–3] and references therein), starting from cloud computing infrastructures, and then extending to mobile and fixed networks. Various techniques can be applied to management and control of energy consumption, both as Local Control Policies (LCPs, at the device level or parts thereof) and as Network Control Policies (NCPs, embracing a network domain, with a view on its devices and links), a classification of which has been attempted in [1], among others. However, in general, energy consumption must be viewed just as one of many Key Performance Indicators (KPIs) in a multi-objective environment, along with Quality of Service/Quality of Experience (QoS/QoE) parameters, with which

R. Bolla · F. Davoli (✉)
Department of Electrical, Electronic and Telecommunications Engineering, and Naval Architecture (DITEN), University of Genoa, Genoa, Italy
e-mail: raffaele.bolla@unige.it

F. Davoli
e-mail: franco.davoli@unige.it

R. Bolla · R. Bruschi · F. Davoli
National Laboratory of Smart, Sustainable and Secure Internet Technologies and Infrastructures (S3ITI), CNIT (National Inter-University Consortium for Telecommunications), Genoa, Italy
e-mail: roberto.bruschi@unige.it

E. V. Depasquale
Faculty of Information & Communication Technology, University of Malta, Msida, Malta
e-mail: edepa@ieee.org

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0_2

an optimization trade-off must be sought, or which should be considered as constraints to be met.

Traffic growth is one of the causes of increased energy consumption and Table 2.1 shows the consistency with which Cisco's VNI has been predicting heavy growth in traffic exchanged ***over the access network*** by both businesses and consumers with: (i) endpoints over managed networks and (ii) endpoints over unmanaged networks ('Internet traffic').

This is corroborated by several other researchers, with perspectives varying from traffic at the access segment to traffic in transit between Internet Service Providers (ISPs) [7–9]. Cisco [4] and Sandvine [10, 11] identify 'video traffic' and 'real-time entertainment' as the drivers of this growth; Alcatel-Lucent Bell Labs [12] identifies that growth in the metro core due to video traffic exceeds growth in video traffic crossing the long-haul core. The growth of energy consumption by the network to support video traffic is recognized by researchers, who have been tackling the problem [12–24]. From these works, development of cache architectures emerges as an important approach to controlling energy consumption, but other mechanisms exist and yet more are emerging within the general thrust towards 'future networks' (ITU-T Y.3001 [25]). 'Radical' approaches employing dynamic and reactive control are distinguished from 'reformist' approaches that seek to improve the caching of content and are characterized by investigation of the reduction of the length of the path between source and destination of IP traffic. Our work seeks to enable radical methods by equipping the control plane with a powerful yet lightweight architecture for exploiting extant and emerging green capabilities.

While considering the evolution of networking towards more flexible and programmable 'softwarized' virtualized infrastructures and strongly integrated paradigms as in 5G [26], a significant difference may exist in the techniques to be applied to achieve the above-mentioned trade-off between energy and performance, with respect to 'legacy' infrastructures based on switches and routers. Though there is a general belief that Network Functions Virtualization (NFV) should result in reduced energy consumption, owing to consolidation of resources and increased flexibility in turning unused hardware (HW) on and off as needed, it is also true that 'the massive introduction of general-purpose HW enabled by NFV would tend to increase power requests with respect to specialized HW solutions' [26]. Therefore, there is a need of management and control mechanisms to be put in operation in these environments, taking care not to increase their complexity and/or the level of

Table 2.1 Percentage of Compound Annual Growth Rate (CAGR) reported in Cisco's VNI over three consecutive years

Period	Fixed Internet traffic	Managed IP traffic	Mobile data
2014–2019 [4]	23	13	57
2015–2020 [5]	21	11	53
2016–2021 [6]	23	13	46

human intervention, in order to keep Operational Expenditures (OPEX) within reasonable limits.

In this chapter, we focus in particular on these issues in the context of video distribution networks (VDNs), where additional paradigms, as content delivery and caching operations, play a relevant role [27, 28]. We first provide a short review on the current state and on energy-related aspects in VDNs; then, we focus on LCPs/NCPs in this context, and the possible role of the Green Abstraction Layer (GAL) [29], a recent ETSI standard [30], to provide an abstract interface to convey energy efficiency related parameters to the management and control entities. In doing this, we consider both 'traditional' networking architectures based on specialized HW, and the evolution towards virtualized infrastructures, where the GAL can still play a significant role, if suitably adapted to the specific environment.

2.2 State of the Art: Video Distribution Networks

2.2.1 Section Overview

In this section, we explore in detail about the data plane functions of the 'video distribution network' (VDN), since this, as is generally true, is where most energy use takes place during operations.

The term 'VDN' encompasses a wide variety of system-architectures and technologies that ingest, process, aggregate and distribute video, audio and text information. The VDN is bounded at one end by the ingest to the headend, which egests to the transport network, and at the other end by the (VDN-dependent) network device that transmits the multimedia to the user terminal. We first introduce the genres of VDN. We refer to genres to classify identifiably different combinations of architectural and technological ingredients. Genres, therefore, differ not only in their architectures, but also in the technologies out of which their organizations are assembled. The organization of the VDN's headend is notably genre-dependent, but space restrictions do not permit us to dwell on this aspect. We proceed to suggest a template for the combination of these ingredients and to briefly indicate its variants. The hierarchy we shall use to organize the space of VDNs, therefore, consists of that shown in Fig. 2.1.

A VDN is owned either in full or in part by a Video Service Provider (VSP, used to encompass any organization that delivers video to customers, regardless of the VDN's genre). We will then proceed to anecdotal references of what appears to be the VSP's ultimate destination, i.e. the unified headend.

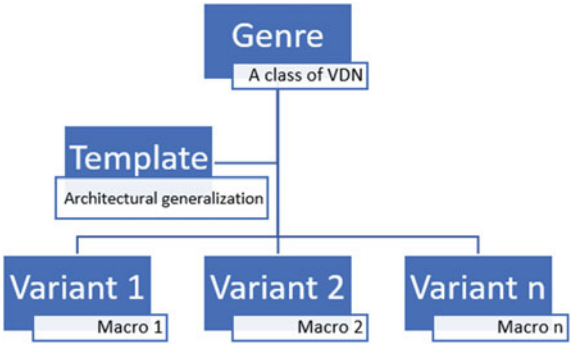


Fig. 2.1 Organisation of the VDN-architecture space

2.2.2 The Genres

2.2.2.1 RFTV and IPTV

The first two genres we distinguish are RFTV (radio-frequency television) and IPTV. RFTV distributes digital television in the fundamental organizational unit of the MPEG Transport Stream (MPEG-TS): it is the common denominator in all RFTV forms. An early moniker of IPTV was 'telco TV' (ca. 2004), emphasizing its origin and its proponent as well as shedding some light on a commercial driver: opening up competition for 'telco' operators with cable TV system operators.

RFTV minimally comprises formatting, source encoding, channel encoding, modulation (carried out at the headend, where the video—and audio and text, where applicable—information enters the VDN for live broadcast transmission) and a channel. Live broadcast transmission is specifically singled out as it requires a shorter introduction, but Video on Demand (VoD)—a unicast form of transmission—can also be delivered from an RFTV headend. Other means of transmission, such as over the top (OTT), are better characterized as part of later generations of headend.

Similarities. IPTV and RFTV are similar in the following ways:

1. ***Common functions.*** IPTV transmission uses formatting and source encoding, like RFTV.
2. ***Managed networks.*** An important common characteristic is that both RFTV and IPTV use ***managed networks*** that are capable of guaranteeing QoS fit for ***tele***-vision.

Differences. IPTV and RFTV are fundamentally differentiated in the following ways:

1. ***Complete abstraction between source bit stream and channel.*** IPTV transmission uses formatting and source encoding, like RFTV, but between the source bit stream and the channel, there is a complete abstraction. The rich diversity of modules that populate the layers between the application and the

link (which includes the channel) of an Internet Protocol Suite architecture [31] may fill in the implementation. This is the current state of evolution of digital communication, where the Internet Protocol Suite architecture has provided the openness necessary for the independent development of competing (same-layer) and complementary (dissimilar layer) modules.

2. ***Path selection***. IP is predicated upon the individually routable packet and does not intrinsically require reservation of path resources like circuit-based communication. Evidently, this does not modify the characteristics of video's demands on its transport, but it does allow the provision of more efficient and possibly more flexible transport than circuit-based one. RFTV uses an isochronous MPEG-TS with no further path selection, and each MPEG-TS packet in RFTV follows the same physical path as its peers. RFTV is ***circuit-like***.
3. ***Multicast versus broadcast***. With IPTV, video is egested from the VDN headend as multicast over UDP for live transmissions and/or stored to be used for VoD. When the stored video is requested by end-users that have passed authentication, authorization and accounting (AAA) management and digital rights management (DRM), the video is unicast using RTP over UDP. We have referred earlier to work that suggests that VoD is most responsible for growth of IP traffic in the metro core [12]. With RFTV, video is egested from the VDN headend as a channel broadcast.

2.2.2.2 Internet TV

A third genre is Internet TV, also referred to as OTT TV, or plainly OTT. The principal differentiator between Internet TV and the preceding two genres regards the network intermediate to the source and the destination. OTT is delivered over an intermediate network that is owned by an arbitrary combination of third parties whereas (as stated earlier) RFTV and IPTV are delivered over a managed network.

The Content Delivery Network (CDN) is an important architectural component of this genre. One taxonomy [32] associates OTT delivery with a specific CDN architecture, where the CDN operator (e.g. Akamai or Limelight) owns the datacenters but not the network nor the content. A commonality is immediately visible; neither the content owner (aka: the rights holder and aka: the contributor), nor the VSP, nor the 'content delivery accelerator' (the OTT CDN) own the intermediate network. This characteristic is pervasive in services delivered over-the-top. While this is not best-effort network service, there is nonetheless a separation between the organization providing the service (the video) and that providing the network service. An evocative phrase often used to distinguish Internet TV from RFTV and IPTV is 'walled garden'. RFTV and IPTV are delivered in a walled garden, where QoS is within the direct control of the VSP. Internet TV is delivered over networks that are outside the direct control of the VSP.

Another observation concerns the range of content accessible over-the-top. Television content may be divided into (a) real time, or linear in this industry's

jargon and (b) stored content, which complements linear—i.e. anything that is not linear. Stored content is a good candidate for delivery through an OTT-CDN. Viewing of stored content, whether through IPTV (where it is commonly referred to as VoD) or OTT, covers a broad range of high-value content. Stored content includes content which was delivered linearly and recorded for delayed viewing. Delayed viewing is a growing behaviour and with at least one major North American television network, was reported as 'the new normal' [33].

Linear content includes some high-value content in television, like sports, news and live events. It seems reasonable to add series premieres to this list, despite the (at least localized) pre-eminence of delayed viewing. It is evidently desirable to include linear content in an OTT-VSP's product range, but technical challenges face aspiring adopters. Linear TV QoE and reliability are difficult problems to solve for an OTT-VSP without the combination of two stages of the delivery network: the Internet core and the regional network. There are at least two broad approaches towards achieving this goal:

- Customized cascading:
  - contracts with Tier-1 ISPs to obtain the reach of a global network. An example [34] illustrates the specialization of a Tier-1 ISP's network for video content distribution;
  - contracts with regional ISPs. An example [35] of such a contract includes the use of the regional ISPs' access network. The OTT reaches the region through the cloud.
- Turnkey cascading: Multi-CDN
  - contracts with multiple CDN operators [36] to ensure good global geographical coverage address both QoE and reliability concerns.

With these two conditions, the differentiator 'managed network' must be replaced by a more focused differentiator to define a clear surface of demarcation between OTT and IPTV. A better differentiation distinguishes between 'managed IP traffic' and 'Internet traffic'. Managed IP traffic is described as that which (a) flows only through a single network and (b) is managed by a single service provider. On the other hand, Internet traffic flows across an Internet backbone, crossing network boundaries between different (Internet) service providers and CDN providers, which therefore means that there are several, uncoordinated control planes.

Indeed, the geographical boundary of a VSP's delivery of linear content is likely to be the same as that of its managed network, whether physical, virtual or some turnkey wide-area networking solution that ingests the content and distributes it. This statement is a consolidation of the geographical scope of delivery of linear content and is valid for IPTV-VSPs as well as OTT-VSPs. The IPTV-VSP typically owns the network and is classifiable as a Network Provider, whereas the OTT-VSP typically owns a part of the network. The IPTV-VSP typically is an incumbent in the telecommunications operators' field, whereas the OTT-VSP typically is an

incumbent in the field of acquisition of content rights, and must enter into contractual relationships (viz., the two approaches identified above) to ensure QoE and reliability.

A consequence of the lower end-to-end predictability of OTT's intermediate network is the development of a variety of application-layer protocols that support adaptive bit rate (ABR) streaming over HTTP. Examples include: Adobe's HTTP Dynamic Streaming (HDS), Apple's HTTP Live Streaming (HLS), Microsoft's HTTP Smooth Streaming (HSS) and ISO/IEC 23009-1 (better known as MPEG-DASH).

2.2.3 *RFTV VDN Templates and Variants*

The RFTV VDN template comprises (a) the RFTV headend and (b) a broadcast channel. The generalized, technology-agnostic template is represented by the well-known block diagram of a digital communication system and is reproduced in several textbooks [37, 38] on digital communications.

The first specialization of the generalized template that leads towards the RFTV template is the MPEG-TS. The MPEG-TS is not part of the generalized form, since its use in practice is specific to multimedia sources. Figure 2.2 illustrates the relationship between the MPEG-TS and the source bits in a digital communications system.

The RFTV VDN template comprises a long-reach transport portion. This is the satellite uplink used by the content source or the content aggregator to distribute to affiliates' headends. The affiliate uses a short-reach transport portion. Further differentiation within the short-reach transport leads directly to the variants. This follows because the principal differentiator from the agnostic form is the broadcast RF channel. Standardization of the RFTV system binds the channel to the carrier modulation and electrical characteristics of the transmitter and the receiver.

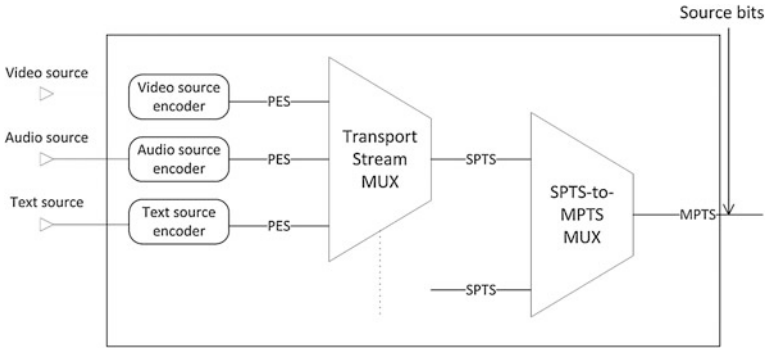


Fig. 2.2 The MPEG-TS is the source bit stream input to an RFTV system. MUX: Multiplexer, SPTS: Single Program Transport Stream, MPTS: Multi Program Transport Stream, PES: Packetized Elementary Stream

Relevant system-level standard views are referred to in the following examples. The interested reader is referred to the indicated standard sources for a system-level view of the variants.

- ***Cable (e.g. CATV)***: RFTV distribution by cable networks [39]
  - Annex B representing US digital cable TV
  - Annex A, equivalent to ETSI EN 300 429 [40], represents European digital cable TV.
- ***Satellite***: Digital Video Broadcasting—Satellite [41]
- ***Terrestrial***: Digital Video Broadcasting—Terrestrial [42] and Advanced Television Steering Committee [43]

Despite the expectations that the OTT genre own the largest percentage of market share in the near future, at least in developed markets like the North American, European and the Far Eastern, a recent survey [44] is giving reason to re-evaluate conventional wisdom on future scenarios of genre adoption. It was found that in the 5 years between 2012 and 2017, homes in the USA that receive TV only through over-the-air (OTA) broadcasts increased by 41% to 15.8 million. A particularly striking note in the press release is 'greater access to … ***affordable*** entertainment' (bold italic font style added). It seems that price sensitivity is high in customers' ultimate selection of a genre for video service.

2.2.4 IPTV VDN Template and Variants

2.2.4.1 Template

IPTV implementations are diverse as a result of the decoupling between IP and the end-to-end path between the headend and the end-user. Generalization is tractable if we are willing to commit some depth of abstraction of the end-to-end path. This path may be coarsely divided into the long-haul IP transport infrastructure and metropolitan area transport infrastructure, respectively. In this basic segmentation, the access segment lies within the metro-area segment. The junction of the long-haul and the metro-area network is typically found in the point of presence of the long-haul network within a metro area, where long-haul equipment (e.g. the P-router) links to the metro-area equipment (e.g. the PE router). Our consideration of the long-haul network's architectural and technological ingredients is somewhat reductionist. There are at least two reasons for this position.

1. The geographical extent of the long-haul network reduces its link-type diversity to two: (a) long-haul optic fibre plant (cabling and erbium-doped fibre amplifiers) carrying DWDM links and (b) satellite links.

2. The use of energy in long-haul IP routing and optical transport, while not easily corroborated by the various studies of the subject, is likely to be an insignificant percentage of the IPTV VDN's energy use. The GreenTouch project's estimate of the long-haul network's (core IP routers and optical transport) energy use is less than 0.5% of the energy used by worldwide telecommunications networks [45].

The long-haul network serves to interconnect the hierarchy of headends. The IPTV template is illustrated in Fig. 2.3. The scope of the variants is that of the metro-area network. The segments of the end-to-end path that lie within this geographical reach are the primary contributors to the diversity referred in the introduction to this subsection. It is therefore apparent that ***the scope for diversity in the IPTV template is the same as the scope for diversity in broadband networks***.

Another significant variation regards the interconnectivity in the metro-aggregation segment. The physical topology of the optic fibre is not restricted to rings (with protection); for example, the Local Exchanges (LEs)/Central Offices (COs)/Distribution hubs of a single Network Provider may be interconnected in mesh form or in hub-and-spoke form with the metro Point of Presence (PoP) on the metro-core ring.

Finally, we observe that the IPTV template may be extended to include the metro-core ring. Each PoP on this ring provides up- and downstream connectivity to at least one specific metro-aggregation segment. This ring covers a larger geographical extent than the metro-aggregation segment and its topology (with protection) is a good trade-off between resilience and capital expenditure. We may therefore reasonably conclude that the scope for variants lies within the access and backhaul (metro-aggregation) segments and most notably within the access segment.

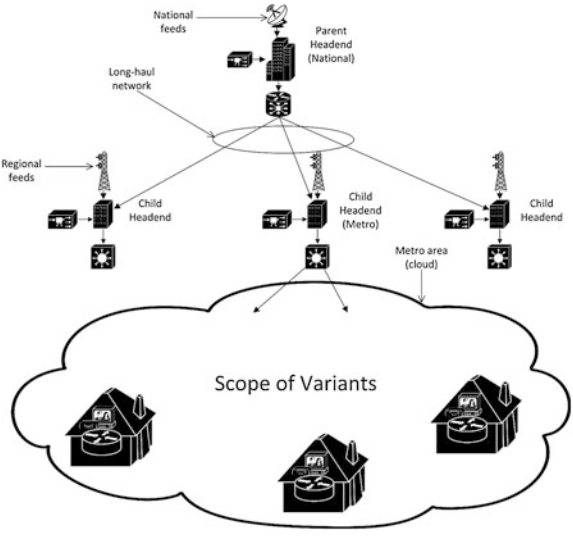


Fig. 2.3 IPTV template. The scope of the variants is collocated with the scope of the metro area

2.2.5 Destination: CORD, HERD or Cloudsource?

In network providers' technology space, the convergence of communications and information technology is shaping the design of the CO/LE (CORD: central office re-architected as a datacentre) and the headend (HE; HERD: head office re-architected as a datacentre).

Netflix's workflow is entirely delegated to AWS except for the CDN. BT has cloudsourced video processing operations from AWS Elemental [46]; BT does, of course, own transport and acquisition infrastructure. These two cases amply exemplify the scalability of cloudsourced operations. Crucially, it illustrates a symbiotic relationship in which huge processing resources are linked to wide-area telecommunication infrastructure and successfully deliver a time-sensitive user application. Since cloudsourcing a service is the alternative to running it on premises, and since COs/LEs/HEs are optimally located (physically) for transmission and switching functions, the justification for CORD and HERD must include prioritization of these two critical functions. These are the functions which distinguish NPs and which, together with ownership of pathways, cable and right of way, support all NPs' services.

A key tenet of CORD and HERD is software-defined functionality. The unifying concept behind SD-x (software-defined anything) regards the flexibility with which the functional as well as the non-functional requirements can be changed and the efficiency with which these requirements can be implemented. Both flexibility and efficiency need qualification by the specific system's context. With specific regard to use-phase energy, it is good to dispose of methods that facilitate the efficient use of energy in a system. Furthermore, it would be better if these methods, alone, or in combination with others, facilitate changes in the implementation of the system (hence: flexibility) to match a dynamic external environment (i.e. external to the system).

When considering the development of virtualization on the Intel architecture, we can see that hardware has been developed to assist the flexibility afforded by virtualization. This development in hardware has been exposed through Intel VT architectures, which provide the means for the software (hypervisor) to switch between one OS and another. The behaviour of the system changes through the collaboration of hardware and software. Development of Intel VT was specifically in response to the need for more efficient virtualization, since VMware's products were using binary translation of Intel binaries to patch critical instructions. Processor utilization efficiency was low. During the life cycle of a Virtual Machine (VM), the total complement of hypervisor-instructions to support guest isolation is a high percentage of the total number of the sum of hypervisor-instructions,

guest-OS-instructions and user-application-instructions. If time spent in memory management for VM isolation is brought into the evaluation of efficiency, then efficiency (without Intel's VT-x2) percentages drop further.

Further understanding of flexibility may be obtained through consideration of a hardware-bound function. There are at least two reasons why a function may be hardware-bound. One is that the function is implemented in an electronic or an optical circuit. Another is that the function is implemented in software which cannot be decoupled from its hardware platform. Suppose that this function is required as part of a system's behaviour. Once the function is integrated into the system through interfacing to the hardware item, the location of utilization of the function is limited by the programmability of the paths to and from the interface, since the function itself cannot move. Conversely, with function virtualization, the function is implemented on purpose-independent (i.e. general-purpose), vendor-independent hardware.

One can conclude that flexibility is conferred through ease of repurposing of an ensemble of functions. Repurposing can be effected in at least two ways: (a) moving one or more functions, achieved through function virtualization; (b) changing the paths to and from one or more functions, achieved through software-defined networking. Efficiency is conferred through hardware support of the new purpose. If this is not to defeat flexibility, then hardware support must be exposed by extending general-purpose architectures.

Function virtualization, software-defined networking and general-purpose processor architectural extensions are the foundations upon which CORD and HERD depend if these are to be the models of future COs and HEs. It seems reasonable to anticipate that the logical culmination of CORD and HERD is a single datacentre architecture in which all functions are implemented on virtual deployment units in a pool of general-purpose servers, under the orchestration of QoE- and energy-oriented video service applications, running on a network operating system.

2.3 A Treatment in Summary of the Green Abstraction Layer

In this section, the energy-performance optimization problem is stated; from this, the need for the Green Abstraction Layer emerges as a corollary. We identify key concepts in this architecture, of which we give an overview.

2.3.1 Focusing on the Optimization Problem

We have emphasized identification of functions over identification of devices in our exposition of the VDN. We have justified this approach on the basis that it is

necessary to investigate whether energy can be saved without taking delay, delay jitter, packet loss rate and bandwidth below the thresholds for video on demand and video broadcast. Both VoD and broadcast (or multicast) video, as well as time-shifted video, generate traffic that is classified as streaming and has the following attributes: inelastic, non-interactive and high-bandwidth (>2 Mb/s/stream) [48]. We have referred to these as the user applications and indicated that the network service used by the NP to carry these applications must match or exceed the QoS. These applications' traffic is also characterized as requiring resilience from the underlying network service and being highly asymmetrical [48].

Taking a first conclusion, it may, therefore, be said that the optimization problem is concerned with: (i) meeting user applications QoS requirements, while (ii) minimizing the total energy used to process the VDN functions that implement the user application.

The second part of the statement requires clarification before the scope of research can be accurately discerned. The energy used by the VDN functions may be calculated in a technology-agnostic manner. Digital communications theory is premised on the principle that signal energy must be sufficiently high to permit reliable detection in the presence of noise. However, signal energy is not the same as system energy. In the worst case (Fig. 2.4), system energy is independent of signal energy [49@16:23]. If (a) the Energy-Efficient Ethernet's Low-Power Idle (LPI) had been implemented in the transmitters used in the cited example [49@16:23] and (b) the system implemented to exploit LPI, then it is expected that the power usage curve would have been different. Evidently, energy use is implementation-dependent. Therefore, approaches cannot be technology-agnostic. On the contrary, a ***realizable technological context*** is mandatory in research into energy efficiency.

The latter requirement has important ramifications. If control of energy use cannot be approached without realizable technological context, then control of

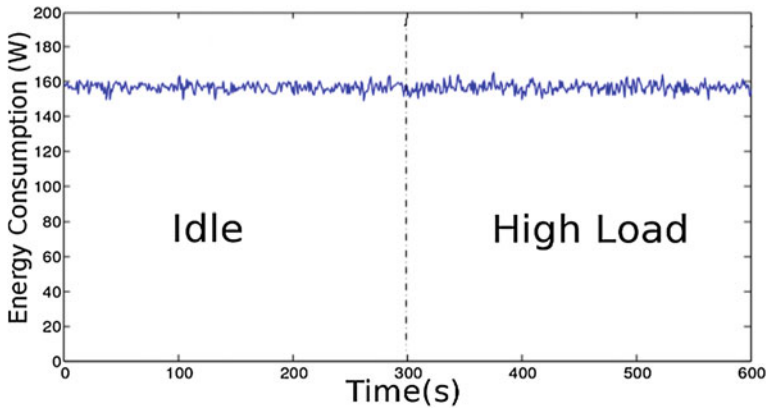


Fig. 2.4 A practical example of energy use that is independent of load (10GbE switch) [49]

energy use is not only technology-dependent but also implementation-dependent. Each implementation must be seen as a unique problem. Evidently, this is not a viable approach, yet it is the context in which we largely operate today, where knowledge about the relationship between performance and energy use is the object of ad hoc research as exemplified above! An important means to tackle this organizational diversity is architectural conformity. Architecture is concerned with the study of structure and behaviour. Behaviour is of interest primarily in so far as a component interfaces with other components (or end-users). A useful interface would provide bidirectional communication between control (control plane) and process (data plane):

Data → Control	Control → Data
1. Performance metrics	1. Commands that change performance and energy use while operational
2. Energy use metrics	2. Commands that change operational status
3. Operational status	3. Queries

Therefore, conformity is achieved through standardization of structure and interfaces that facilitate the development of control systems for optimization of energy use, under the performance constraints that emerge from QoS requirements. This architectural conformity is what has been proposed in the Green Abstraction Layer (GAL) [30].

2.3.2 *A Functional Approach to the Green Abstraction Layer's Architecture*

The basic needs which this architecture must serve are: a) Resolution of scalability of energy-use control; b) Standardization of instrumentation (performance and energy-use measurement); c) Standardization of control messages. The latter two needs regard behavioural aspects of the architecture; the first regards a structural aspect of architecture. We shall first address the structural aspect.

Scalability of energy-use control. The device that is in the scope of energy and performance control may be as simple as a variable-speed fan or as complex as a multi-chassis switching node. In order to address both total quantity of control messages as well as response time, a hierarchical arrangement has been proposed, structured in alignment with the device's aggregation of components into subsystems and subsystems into systems. The depth of the hierarchy is arbitrary and the manufacturer is accommodated by leaving the decision about exposed hierarchical detail within the manufacturer's discretion. We use the example in Fig. 2.5, regarding a multi-chassis switching device, to illustrate the design of scalable energy-use control. Furthermore, in order to give concrete detail, we shall define the concepts in the context of a single 'energy-aware' entity (EAE), which is defined

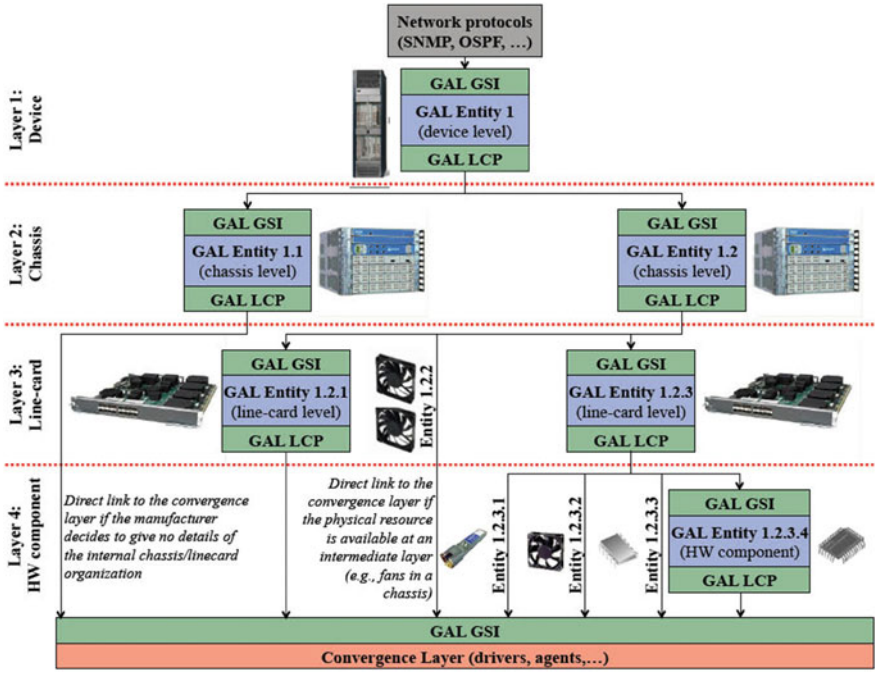


Fig. 2.5 * Structure of scalable energy control in a legacy multi-chassis switching node. *Reforgiato D, Davoli F, Fialho L, Collier M, Enrico M, Donadio P, Bolla R, Bruschi R (2012) Exporting data-plane energy-aware capabilities from network devices toward the control plane: The Green Abstraction Layer. Proc. 17th European Conf. on Network and Optical Communications (NOC 2012), Vilanova i la Geltru, Spain, June 2012

[30] as one that can trade energy use for performance, functions and responsiveness.

The software that implements energy-use control is our proxy on the line card. Consider the line-card level of the hierarchy. Energy-aware entity (EAE) 1.2.3 is a software component that, functionally, balances the use of energy with performance, functions and responsiveness (PFR) for the line card. The software component has two major subcomponents: the Green Standard Interface (GSI) and a local control policy. It is important to emphasize the intention to use 'policy' as the descriptive term. It is a policy for the following reasons.

1. It is an implementation of the manufacturer's reasoning about how much autonomy should be devolved to local control (for the specific entity at the given level). This entails a decision about whether to permit an override by controllers with broader scope.
2. The autonomous control actions are the detailed implementation of policy local to the given level.

3. It is an implementation of the manufacturer's design of orchestration and aggregation of the underlying EAEs.
4. It reflects the manufacturer's choice about the depth of organizational detail to reveal about the directly underlying layer.

EAE 1.2.3 also stores a number of data structures that represent the energy-aware state (EAS) of the line card. We devote part of the sub-section regarding standardization of instrumentation to exploring the EAS. For the time being, its essence may be captured by the simple concept that the EAS is a data-structure-representation of a power-configuration of a particular energy-aware entity. The EAS's relevance to consideration of designing scalability into energy-use control lies in the following. The EAS of EAE 1.2.3 is an aggregation—a summarization, of the resources which it integrates into its organization. These resources include integrated circuits like network processors, function controllers and buffer memory; line interfaces on the line card (e.g. enhanced small form-factor pluggable (SFP+) + modules) and variable-speed fans.

Fine-grained, distributed control is effected through a standardized operating procedure (workflow). Consider the case of a centralized network controller which uses an energy-aware network control policy (NCP). The NCP determines that a logical link (IP layer, say) can be brought down. Here we limit ourselves to expanding upon an earlier narrative [50, p. 84] by illustrating the procedure for sending the relevant port to sleep. We use a data-flow diagram (Fig. 2.6) that shows

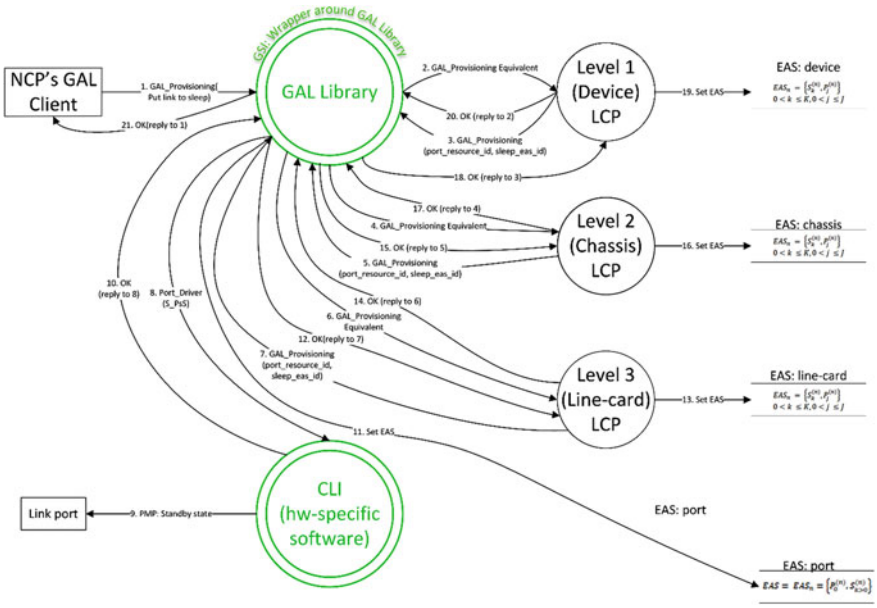


Fig. 2.6 Data-flow diagram showing interaction between components of the GAL architecture

the interaction between the hardware and software components. The numbers on the data flow arrows show the order in the sequence when the data flow takes place.

Note that apart from the EAS for the port-EAE, the EAS for the line-card-EAE is modified too. It must be kept in mind that the line-card-EAE's EAS is a summarization of the lower layer EAEs' states. Therefore, a change in state at a lower level will, in general, affect the EAS of the 'parent' EAEs.

Standardization of instrumentation: the energy-aware state. Control of energy use by a component requires that the component's hardware possesses the intrinsic capability to perform its function at different levels of energy usage. If the energy cost of the function is fixed, then there is no scope for local control of energy usage. The same consideration applies to a system of components (of various types and various performance levels per type). For the system to control energy usage, the hardware must possess the intrinsic capability to perform its function in different ***energy-aware states*** (EASes). We use the term 'energy-aware' to emphasize that the transition $T_{S_i \rightarrow S_j}$ (or simply, T_{ij}) between state S_i and another, S_j, is a transition between two specific combinations of values for each property in a set of properties $\{P_n\}$ ***that includes energy in the set***, i.e. $S = \{P_n\}$. Since we are interested in the performance of the function, then $\{P_n\}$ must also include performance metrics relevant to the function.

The state space can be uncountably infinite if the energy property is defined as a continuous variable. Alternatively, we can rationalize into a countable set of EASes through well-defined ***operating modes*** of the component. This approach is exemplified in the ACPI's (Advance Configuration and Power Interface) global states of a computer system. ACPI G0 is more accurately described as an operating mode that encompasses a whole group of states. Each state in the group is represented by a specific combination (a ***state***) of the form $\left(\left\{C_l 0\left(\left\{P_0^{C_l}, \ldots, P_{0N_{C_l}}^{C_l}\right\}\right)\right\}, \left\{D_m 0\left(\left\{P_0^{D_m}, \ldots, P_{0N_{D_m}}^{D_m}\right\}\right)\right\}\right)$, where $C_l 0$ represents the power-aware processor-core's operating state, $D_m 0$ represents the power-aware system device D_m's on-state and the P-symbols represent the power-performance (trade-off) states which each power-aware processor-core and device is capable of. Returning to the EAS, we may, therefore, identify an EAS as a state within a given operating mode. An EAS defined in this manner will have a lower and an upper bound on the energy used while the component is in that operating mode.

We now seek possible representations of transitions between states (EASes). Our interest stems from the recognition that each such transition is effected by the execution of a control (-plane) action in order to adapt to external conditions. We first recognize that some transitions may not be possible. For the remaining, permitted states, representation should comprise the information required by a controller to effect the execution. We refer back to our earlier differentiation between organization and architecture. If the implementation discloses time cost, energy cost and performance cost in order to effect the transition, then the architecture discloses sufficient information to permit the control action. This can be compared to knowledge of the transfer function of a continuous analogue process. This is

sufficient to effect closed-loop control of the process without knowledge of the process's implementation.

We will add further depth to the use of the EAS later in this section. In the meantime, we can reuse Fig. 2.4's example to support our claim for scope of control of energy usage. That Ethernet switch may have been built out of energy-unaware components. Alternatively, it may have been built out of energy-aware components (e.g. transmitters equipped with LPI), but it may lack energy awareness in its ***local control policy*** (LCP). We express this pithily by stating that: either the component (local scope) or the system (or both) lacks ***green capabilities***.

Primitives. Since the operating mode 'active' is divisible into power-performance substates, then the operating mode 'active' cannot be referred to as a primitive. The other operating modes that represent degrees of sleeping are irrelevant to performance and therefore apparently indivisible, but we would like to coherently describe the primitives and avoid dividing a single dimension (power states or operating modes) into a part that is divisible (the 'active' state) and a part that is indivisible (the sleep states). We, therefore, define the indivisible primitives as follows:

1. power-scaling primitive substates (P-PsSes):
   a. substate number 0: performance (-relevant metrics) and power use are at their highest and
   b. substate number J: performance (-relevant metrics) and power use are at their lowest;
   c. substates between 1 and J-1: these represent intermediate performance states while the component is active;
2. standby primitive substates (S-PsSes):
   a. substate number 0: the component is active, or 'on'
   b. substate number K: the system is off
   c. substates between 1 and K-1: the component is sleeping and therefore is not performing its function.

We now relate the primitive (indivisible) substates to the energy-aware state. A component's EAS is uniquely defined by a pair of primitive substates that collectively express its operating mode and its power-performance level in that mode. We would also like to index the set of EASes using a single scalar descriptor, n. We express this mathematically by describing the nth EAS using the following notation:

$$EAS_n = \left\{ P_j^{(n)}, S_k^{(n)} \right\} \quad 0 \leq k \leq K, 0 \leq j \leq J$$

Definition of the relationship between index n and the primitives' indices may be found elsewhere [30]. There are $\{(K) + 1 + (J)\}$ defined EASes but we build extensibility into this structure by postulating EASes in hybrids of the two operating modes, with $j > 0, k > 0$. The range of values of n above $n = J$ has been reserved

for these custom states. It is expected that transitions to these EASes are internally initiated and that these states are inaccessible to external control processes, whether local to the component (LCP) or external and part of its network's control (NCP). It now remains to introduce an interface through which the primitives and other elements of the data plane ↔ control plane interaction can take place.

The Green Standard Interface. Two saliencies which have been used to describe the Green Standard Interface (GSI) are that it is a 'northbound' interface and that it is 'lightweight'.

It is ***northbound*** because it is intended as the means for the NCP and the LCPs (any level of LCP) to communicate with 'lower' (and hence southwards, or towards the data plane hardware) levels in a manner that frees their development from intimate knowledge of structure. A brief digression is in order to clarify the architectural awareness of the LCP. At any given level-X, LCP-level-X is only pre-built with ***awareness of the logical and physical resources in its 'own' EAE***. Furthermore, LCP-level-X does not drive the physical resources directly. It does so only through the GSI. For interacting with hardware resources, the GSI wraps driver code and exposes it through the same control messages, which it exposes to provide (indirect) access to the LCPs at the several levels. The driver code is formally part of the convergence-layer interface (CLI). As regards the logical resources (or logical entities, as 'entity' and 'resource' are synonymous [30, p.7]), these are software-only elements that are centred around the summary EASes. A logical resource will incorporate a physical resource (which will have a type as defined in RFC 4133) if the manufacturer exposes the logical resource's architecture. The logical resource will also incorporate its EAS (as a complex data structure and private member of the resource).

The GSI is also '***lightweight***'. It comprises only six commands, divided into discovery, provisioning, monitoring and configuration-management categories. The simplicity of the GSI is visible in Fig. 2.7, which is a succinct representation of the functionality in the GSI. Referring back to Fig. 2.5, it may be seen that the GSI is a universal interface in the GAL Architecture. Every inter-entity interaction takes place through the GSI. This simplicity is enabled by two complementary supports:

1. At each EAE, the LCP implements the fundamental data abstraction of the EAS to which the LCP is attached. At the lowest level EAE, this abstraction is converted into its hard currency of power-management primitives. There, the EAS is simply mapped onto the {standby, power-scaling} pair of primitive substates (PsSes). Above this lowest level, say at level-X, the EAS is an artefact of the manufacturer who must create a coherent and consistent relationship between:
    a. on one hand, the power and performance of the subsystem which entity-X represents and
    b. on the other hand, the power gain and the performance gain which each possible value of the EAS can take.

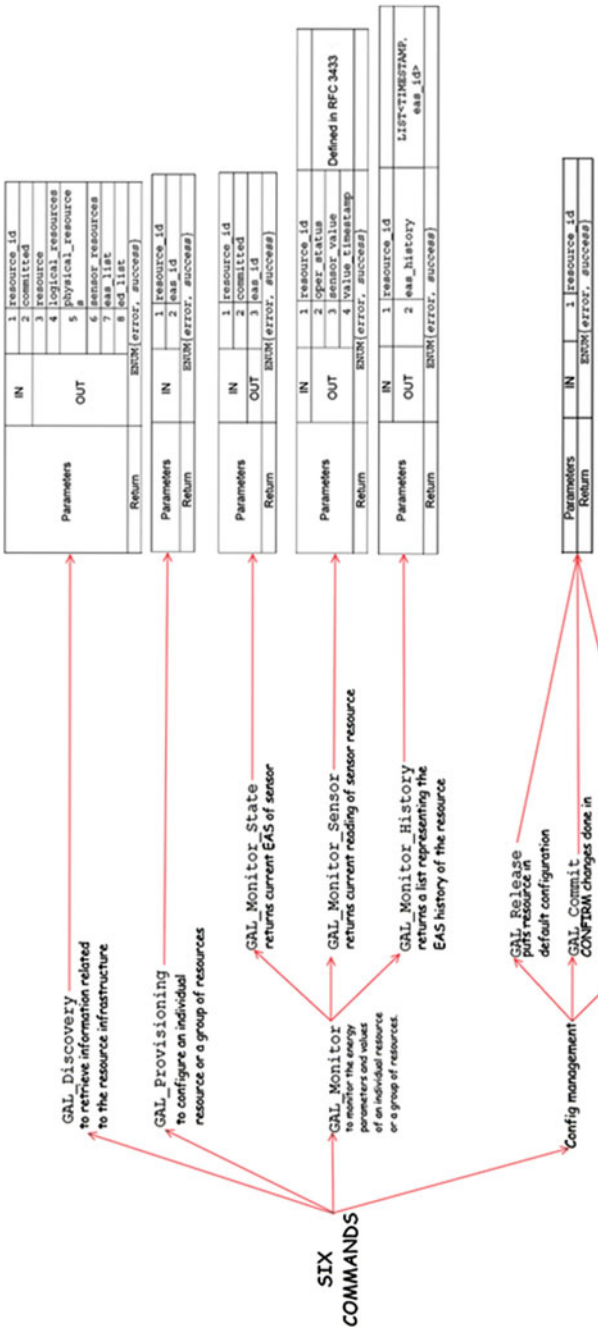


Fig. 2.7 The Green Standard interface's command set

2. At the hardware level, the LCP must be able to interact with drivers and hardware-intrinsic software through the same set of messages. Therefore, a layer must converge the diversity of the hardware with the homogeneity of the GSI; we call this a convergence layer. Essentially, the convergence layer is wrapped by the GSI and the convergence layer then transforms the intention of the GSI command into a sequence of hardware actions appropriate to the hardware being driven.

A description of the commands may be found elsewhere [30]. During real-time control, the GAL_Provisioning and GAL_Monitor comprise the forward and feedback paths of the control loop. As emphasized earlier, the control loops have tight latencies by virtue of their ***localized*** nature. At the same time, a level-X control loop conveys information to higher level control loops to facilitate control at a scope that is beyond the infrastructure in the EAE at level-X.

2.4 Applying the GAL to Future-Generation VDNs

2.4.1 Scope of the Green Abstraction Layer

An important concern surrounding the GAL regards its relevance in the wake of the paradigm shift from conventional infrastructure towards Network Function Virtualization (NFV). By conventional, we refer to the use of bespoke organizations and architectures, with hardware-bound network functions. Conventional infrastructure is exemplified by the PE router (switching node). NFV is built on cloud-datacenter infrastructure: compute, storage and networking devices that combined form an NFV Infrastructure PoP (NFVI-PoP) hosting Virtual Network Functions (VNFs) under the direct lifecycle-management of a virtualized infrastructure manager (VIM), according to the network service requirements input to the orchestrator. The orchestrator relates to the VIM; the VIM invokes the hypervisor (or virtual machine monitor, VMM) in order to deploy, migrate and terminate virtual deployment units (containers or VMs). The NCP relates to the local control policy of ***network functions*** (NFs); here, they are virtualized. This focuses the original concern to a first understanding of the relevance of the GAL to a VM. It has been shown that the architecture of the GAL maps onto the architecture of the appliance that implements the NF. Therefore, ***our concern is with how the GAL relates to the architecture of a virtual machine***.

Some progress towards resolution of this concern may be made through investigation of the binding of the hierarchical architecture to the power-modulating hardware. The GAL's hierarchy is a mapping of the device's hierarchical organization and the example of the multi-chassis switching node was used to illustrate this mapping. It is useful to expand the notion of mapping at this point in our treatment. To do so, we must distinguish between energy-aware-entities in hardware and energy-aware-entities in the GAL architecture. Hardware energy-aware-entities indeed can lower performance for a reduction in power

consumption. The representation of these entities within the GAL architecture is at the discretion of the manufacturer. We concretize this latter statement by drawing equivalence between representation and ***discoverability***. If a hardware EAE is represented in the manufacturer's implementation of the GAL architecture, then its resource_id will be returned by a GAL_Discovery command at the GSI. At best, representation uncovers full architectural detail of the device. In general, this would not be the case, with an exception perhaps for implementations of open architectures. Representation may, therefore, vary between full transparency and near-complete opacity. Near-complete opacity corresponds to sole representation by a device-level LCP. This would interact with the NCP through the GAL and still provide the scalable energy control, standardized instrumentation and control-messaging characteristic of the GAL Architecture. Since only the top level is mandatory [30] and the very presence of a hierarchy may be disposed of, it is possible that near-complete opacity may be a preferred option for the manufacturer.

A hardware EAE may range from device (highest level) to component (lowest level). A GAL high-level EAE's EAS, representing the hardware EAE's power-performance trade-off, is a weighted composite (or aggregation) of the several lowest level EAEs' EASes. We have referred to the low-level EASes as a set of two primitive substates (PsSes). Therefore, the accurate representation of the high-level trade-off depends upon the accurate representation of the primitive substates. The GAL Architecture's use is contingent upon the accurate representation of the primitive substates. If the PsSes are inaccurate, then the entire system for distributed energy-use control collapses.

In both the transparent and the opaque implementations of the GAL architecture, it is the responsibility of the manufacturer to devise the algorithms that aggregate the low-level EASes into high-level EASes. Therefore, the onus of ensuring the accuracy of the low-level EASes is the manufacturer's. Referring back to the data-flow diagram, we see that it is actually the hardware driver in the convergence layer which executes the power-management primitives. The convergence layer is populated by software entities (drivers and similar agents) that expose the GSI as a northbound interface and interact with I/O registers, memory, interrupts and other hardware-specific interfacing mechanisms at the southbound side.

If the driven component is virtual, we immediately detect a problem. In this case, we need to explore the relationship between the virtual component and its physical counterpart, since energy use takes place in the latter. It is necessary to find out how current virtualization practice for I/O devices' components (e.g. link ports, network processors, memory buffers) and for general-purpose processors affects the exertion of primitive substates onto the realizing physical hardware. The sharing of hardware between virtual components is immediately visible as a cause for concern. If a virtual component is set to state $\{S_K, P_0\}$ (off), it is evident that the power gain attainable with a physical component is not attainable here. The physical component hosts functionality required by other components; it will not be switched off.

General-purpose processor virtualization extensions have greatly improved the performance of virtual machines. Among the improvements, Intel VT-x introduced a set of rings of privilege underlying the traditional four of the IA-32 architecture.

These rings are the architectural manifestation of a processor that was enhanced to reduce emulation of instructions, thus increasing the percentage of native execution of guest operating systems' instructions. With Intel VT-x2, efficiency of memory virtualization in VMs was greatly improved. Intel VT-x2 introduced support for second-level address translation. This enables the use of 'guest-physical' addresses in guest OS page tables and eliminates the need for the virtual machine monitor (VMM) software to intercept guest OS access to page tables. I/O virtualization in general-purpose machines is tackled using at least three different technologies:

1. emulated components* (emulation in VMM)
2. physical sharing of components*, such as I/O adapters supporting PCI-SIG's SR-IOV
3. paravirtualised component* drivers, which reduce overhead by using direct communication between the guest OS's component* driver and a driver that handles the I/O component*

   * we use component here instead of the more familiar 'device' to avoid confusion with the earlier use of device where we referred to a 'device' layer/level

In all three cases, isolation of any one guest's I/O from any other guest's I/O is a constraint of paramount importance. The requirement to isolate guests seems to preclude the earlier *modus operandi*, where software running on the physical-network-function device (node) predicates high-level EASes on lowest level primitive substates. This latter information is unavailable to software running on the virtual-network-function-device. It is only available to the VMM. This direct and exclusive relationship between the configuration of power state and the VMM is certainly not incidental. At least since Popek's and Goldberg's seminal paper [51], the role of the VMM has evolved towards a minimalist implementation that carries the concern with ensuring isolation between tenants. Minimalism here conveys efficiency; the ideal VMM sets boundaries for the guests and only intervenes for housekeeping and to prevent malicious or incidental encroachment of boundaries. A critical component of the VMM's minimalist functionality is ***configuration of resources***. This concept defines the sensitive—innocuous complementary instruction classification relevant to the VMM's role. Sensitive instructions are those whose behaviour affects the configuration of resources (control-sensitive) or those whose behaviour is affected by the configuration of resources (behaviour-sensitive). ***The control-sensitive set are clearly within the scope of energy-use control***; we are interested in configuring resources to operate at a performance level sufficient to meet QoS requirements. Therefore, attempting control-sensitive instructions from outside the VMM constitutes an encroachment of boundaries which transfers control to VMM code from guest code. Although virtualization is now assisted by hardware and does not depend on third-generation architectures which were the basis of the sensitive-innocuous classification, the latter still holds. What has changed in hardware-assisted virtualization is that the demarcation between sensitive and innocuous is now ***programmable***. For example, Intel's VT-x uses a virtual machine control structure (VMCS) to program the

boundary between VMM and guest control. The boundary may be positioned to allow deep control by the guest: partitioned usage of a platform is envisaged [52, pp. 33–42], where CPU cores and I/O components are partitioned for the exclusive use of specific VMs.

We have established that the architecture of a virtualization platform (hardware + virtualization layer + VNFs) cannot be used to derive a useful GAL architecture for control of energy use. In its current form, use of the GAL seems to be limited to an NFV-agnostic form that supports the NFVI transport network. We illustrate such a case in Fig. 2.8 [53].

2.4.2 *Extending the Green Abstraction Layer to VNFs*

We have seen that the principal challenge in extending the scope of GAL to NFV is that of extending the GAL to VNFs; an NFV's transport network is still within scope. The relationship between the hardware-EAEs and the virtual objects—the virtual processors, virtual I/O devices, even virtual memory—is not straightforward. The problem may be summarized at one point, condensing our earlier three points. ***There are no known mapping functions between the load on the virtual resource and the load on the physical resource that hosts it***. This problem is rooted in two other unknowns.

- Both (a) the VMM's scheduling policy and (b) the VMM—guest boundary are variables.
- Multiple tenants (independent guests) may use a physical resource at any one time.

Having established the limits of GAL, we turn our attention to approaches for extending the GAL. Here, we need to take into consideration both performance-related (in particular, in the case of video, delay and delay jitter) indicators and energy efficiency indicators, which together form the set of KPIs to

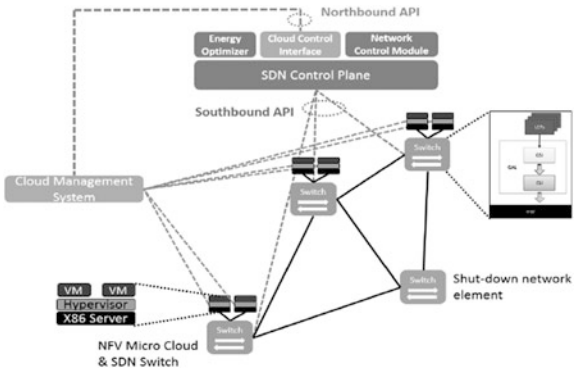


Fig. 2.8 Using the GAL in an SDN/NFV architecture's transport network [53]

be kept within established constraints and/or optimized in their trade-off. Analytical models may be needed, both to predict performance as a function of load and to devise control strategies for resource allocation. Performance indicators may be derived by queueing models, which have to take into account the nature of the traffic (e.g. burstiness levels), but also the enforcement of (packet-level) dynamic adaptation strategies that modulate energy consumption according to the traffic load (typically, Low-Power Idle—LPI—and Adaptive Rate—AR—mechanisms), and whose parameters can be changed by parametric optimization strategies implemented by NCPs (see, e.g. [54]). Such models can be used at different levels of detail, according to the degree of service separation and granularity that they aim to represent.

1. As an aggregate representation of a server (or of a single core, for multi-core architectures), where performance indicators are derived for the aggregate traffic (which necessarily implies a lower detail and rougher approximation); however, such representation has the advantage of bypassing the hypervisor's and scheduler's details in the mapping of the processor's/cores' activity to the traffic, and the very same model (a single queue per processor/core) can be used to derive average energy consumption values, as it relates directly to the hardware behaviour. On and off states of the processor/core correspond to busy/idle states of the queue, and energy parameters can be conveyed between LCPs and the management plane executing coordination and orchestration via the GAL GSI, in a situation that is similar to that of the non-virtualized environment.
2. On the other hand, if we want to adopt more detailed queueing models to represent execution machines of individual functions (e.g. operating on the basis of a single queue per stream), we are faced with the above-mentioned problem of mapping the queue dynamics of a 'virtual processor/core' (which effectively plays the role of the server in this specific queueing system) to the energy consumption of the hardware. To the best of the author's knowledge, this problem has not yet been thoroughly investigated and appears to be a difficult one. A possible solution here, which could still rely on the same GAL structure as in the 'legacy' case, might be the following: i) to use the analytical queueing model in deriving performance indicators for the individual streams; ii) to adopt simpler (aggregate) models to represent energy consumption; the latter can be based, for instance, on Amdahl's law (as done in [55]) or on some form of generalized Amdahl's law (see, e.g. [56]), directly relating energy consumption to the aggregate load, without the mediation of a queueing behaviour.
3. A third possibility consists in a modification of the GAL (currently under consideration in ITU and ETSI working groups), where the concept of EAS is extended to virtualized entities. In this case, detailed 'per stream' queueing models could be used for both energy and performance representation, by introducing a behavioural model of the virtual processor/core and corresponding virtual performance and sleeping states. However, since the latter would have no direct correspondence to the hardware and the real power consumption, a paradigm shift would be needed for their adoption in the overall management

and control scheme. Specifically, we advocate the use of such virtualized energy-related primitive states to create a backpressure from the Infrastructure Provider (which is aware of the real power consumption) towards its tenants: the latter can, in fact, become interested in driving their virtual processor's/cores' behaviour according to LPI/AR LCPs devised on the basis of their virtual primitive EASes (influencing, in turn, their share of real energy consumed), to react to incentives proposed by the Infrastructure Provider (e.g. by means of dynamic pricing strategies also accounting for energy consumption).

2.4.3 *Plumbing the Depths: What Scope for Research into Energy Efficiency Under NFV?*

Our review of the state of the art in VDNs ended with the observation that the future of video processing functions is in virtual deployment units running on virtualization platforms in central offices and headends (CORD, HERD). Figure 2.8 has a useful implication: it illustrates the integration of CORD, HERD and NFV paradigms. Each switch in Fig. 2.8 represents the CO/LE/HE; each micro cloud (or cloudlet) in Fig. 2.8 is representative of exploitation of NFV in the strategic places where it is most required. For example, when an end-user switches between channels, he/she is switching between SPTSes. While the initial delay to start viewing the first channel may be several seconds long, the end-user's QoE will deteriorate markedly should the delay to switch channels be longer than about 2 s. QoE is determining video service QoS constraints and this, in turn, leads to the choice to locate functions in the service chain in respect of these constraints. Caching is a well-researched subject in this regard, but the availability of these virtualization-supporting cloudlets further improves the viability of such theoretical work. Less evident is whether we have a motivation for research into improvement of energy efficiency, given the anticipation of energy savings through consolidation of VNFs into few servers. The first scope of research, therefore, is about whether there is room for further improvement. We have focused on the ETSI NFV use case [57] regarding the virtual Evolved Packet Core (vEPC).

Our work [55] consisted of a theoretical analysis, structured towards the development of a model that enables the comparison of the use-phase energy and operating carbon footprint (OCF) in two scenarios. The first, which we call 'Business-as-usual' (BAU), uses conventional infrastructure to carry out the function of the Serving Gateway (SGW). Our choice consisted of three models from the Nokia 7750 family of service routers (SRs) [58]—the SR7, SR12 and SR12e. The second scenario is one which includes the application of NFV. We consider a two-processor minimalist server, comprising processors from the Intel Xeon Processor E5-2600 v4 family [59], an Intel server board and an Intel 10GbE network interface card. Ten processors in the family are plugged into the model. The primary green capability of NFV is consolidation. We consider this capability in our

comparison to adjust the number of servers dynamically on an hourly schedule of adaptation to load. Furthermore, we adjust the number of servers such that the total service capacity exceeds the hourly peak rate of incoming traffic by less than the maximum capacity of one such server (we use the ceiling function for this purpose). Therefore, a good estimate of the power used by the set of all the active servers is the product of the maximum power used by one server and the number of active servers.

The infrastructures are illustrated in Fig. 2.9 [55]. The BAU scenario is immediately recognizable. The SGW is shown in its position between the radio-access network (RAN) and the EPC, where it interconnects with the Packet Data Network Gateway (PGW) (and the Mobility Management Entity). All network functions are implemented as Physical Network Functions (PNFs) that are inextricably coupled to the bespoke hardware. The NFV scenario requires further clarification. It includes an intra-PoP network, which we postulate as implemented in fat-tree topology. We use knowledge of the relationship between the number of servers and the number of switches in a datacenter interconnected using the fat-tree topology, to calculate the total number of switches required. We first declare the number of active servers per hour:

$$N_h = \left\lceil \frac{a_h.\gamma.A}{\lambda_S} \right\rceil \tag{2.1}$$

- A: Average Traffic Intensity [60]
- γ: Peak—Average conversion factor, currently equal to 3 [61]
- a_h: Hourly factor for peak traffic, where $a_h \in [0, 1]$
- λ_S: Maximum throughput of a server

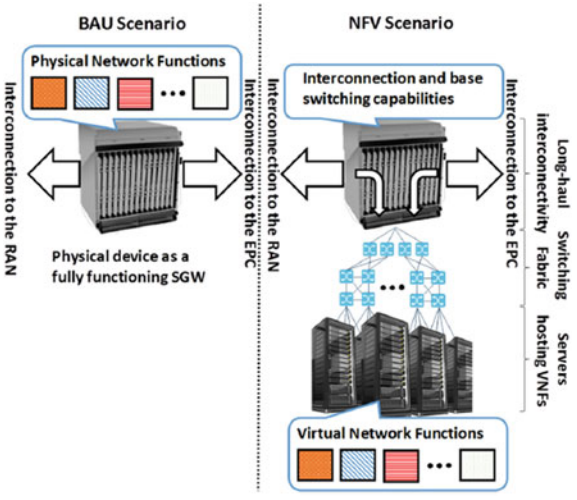


Fig. 2.9 BAU and NFV architectures in the scenario comparison [55]

This gives the peak number of servers required to serve the traffic. We obtain a_h from studies carried out on Telefónica and Orange France networks [62]; we use the average of the two to obtain the normalized hourly factor. Furthermore, to obtain the total complement of servers, we multiply the capacity suggested by (1) (with $a_h = 1$) by a factor of 2, typically used by network operators to overprovision their infrastructure. This leads to Eq. (2).

$$N_S = \left\lceil \frac{\gamma_{o/p}.A}{\lambda_S} \right\rceil \tag{2.2}$$

- $\gamma_{o/p} = (\text{Overprovisioning factor}).\gamma$

Equation (2) is then used to determine the number of switches required, according to the relationship:

$$N_{SW} = k + \left\lceil \frac{k^2}{2} \right\rceil \tag{2.3}$$

where k is defined by [63]:

$$k = \left\lceil 2 + \sqrt{N_S} \right\rceil \tag{2.4}$$

Our final core problem concerns the packet throughput capacity of our minimalist server. For this, we turn first to the work of the European Advanced Networking Test Center Action Group (EANTC), an independent third party [64]. Using a virtual service router (vSR), with worst-case packet processing (64-byte Ethernet frame with an additional 20 bytes/frame of overhead), 57.5 million packets per second (Mpps) were processed. This result can be transformed using Amdahl's law [56], specifically:

$$\text{Actual speedup}, s_c(n) = \frac{1}{(1-p) + \frac{p}{n}} \tag{2.5}$$

- n: the implemented speedup, here consisting of the number of processor cores
- p: the fraction of time during which the speedup is active—i.e. during which all n cores are actively (not idle—i.e. not waiting for input, for resources, or to output) processing packets

For most comparisons between the NFV scenario and the BAU scenario, we use $p = 1$. Although this is optimistic, we do evaluate performance for other values. It must be added that it is not excessively optimistic. When considering use of Amdahl's law, interest focuses upon the ***fraction*** (p) of a task that is executed at the '***enhanced***' rate (n). This, in turn, sharpens the focus upon the ***task***. There are several tasks performed by the SGW. In the case of data plane tasks, these consist of chains of functions that fulfil the forwarding of packets between the eNodeB and the PGW. These chains are parallelable to the extent that ***service requests*** are independent and that ***service request processing*** is independent. Since each service request is self-contained, there is no request interdependence. Service request processing interdependence arises if it creates contention for resources, such as the memory bus. On an architectural level, scope for service-request-processing-contention lies in the load balancer that serves as the common interface to the eNodeB. Further scope for contention lies in the output head-of-line, which is also common to all chains. At the cost of some delay jitter, input buffering resolves the contention for the load balancer. Similarly, also at the cost of some delay jitter, head-of-line blocking is resolved through output buffers. These conditions, most notably the intrinsic service request independence, facilitate actual performance where parallelism is effective throughout packet processing time. We may now write an expression for the server throughput:

$$\lambda_S = 2 \cdot \lambda_c \cdot \frac{f_a}{f_c} \cdot s_c(n) \tag{2.6}$$

- λ_c: Reference processor's throughput
- f_c: Reference processor's frequency
- f_a: Selected processor's frequency
- $s_c(n)$: Speedup, according to Amdahl's law, given that the selected processor has n cores.

This estimate for maximum throughput can now be used to tabulate packet processing capacity with maximum power consumption and facilitate the comparison with Nokia's publications of the 7750-family's throughput-power use figures. We use the value for λ_S to derive a set of figures for N_h, according to the time of the day and compare the results with the worst-case power use for the 7750 family. The results are shown in Fig. 2.10. The curves labelled 'a', 'f' and 'h' regard the vSRs equipped with the best power-performance ratio processor in respectively, the number-of-cores-optimized, power-use-optimized and frequency-optimized groups of the E5-2600 v4 family. The power use, of course, follows the traffic. The lowest power use is in the core-optimized group, but this characteristic is dependent upon the degree of parallelism in the packet processing (i.e. dependent upon the percentage of time during which the speedup is effective). More details can be found elsewhere [55]. A succinct comparison of the power use and OCF in the two scenarios is shown in Fig. 2.11. This shows all three cases of

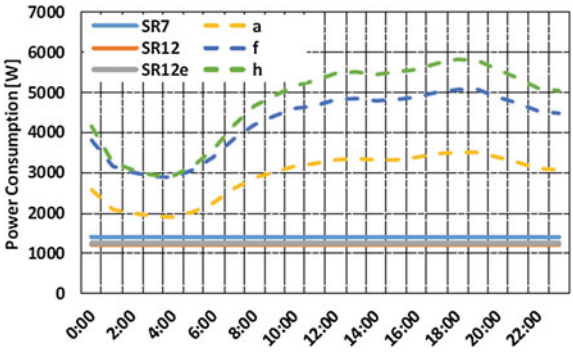


Fig. 2.10 Server consumption in a day, by the hour, for BAU and NFV scenarios [55]

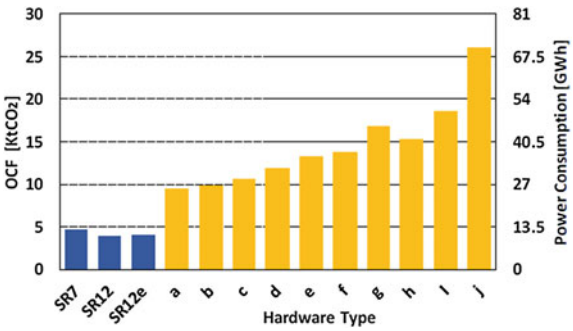


Fig. 2.11 OCF and power consumption per year in the BAU and NFV case obtained for p = 1 [55]

the 7750 SRs and all ten cases of the vSRs. The operating conditions of the vSRs include full exploitation of parallelism (but without exploitation of switching hardware power control). Considering the worst case of BAU infrastructure energy use (the SR7) and the best case of NFV infrastructure energy use, then NFV's energy use is minimally 106% larger than BAU's.

2.5 Conclusion

We have addressed VDNs from the perspective of two relevant evolutionary trends that generally characterize the current scenario of communication networks: the growing introduction of virtualization techniques that leads towards increasing the network 'softwarization', and the constant attention to energy consumption issues. In order to better focus VDN aspects in this context, we have first introduced a classification of them into different genres, based on identifiably different combinations of architectural and technological ingredients. Within this framework, we have briefly examined the main architectural components that play a relevant role, including the headend and the distribution networks, and highlighting their functionalities. Then, we have briefly described the main architectural components and

management strategies that are needed to enforce energy consumption requirements in this, as well as in other, network service deployments. We have summarized the characteristics of the Green Abstraction Layer, a recent ETSI standard that was created to convey energy management information between different control levels and platforms. The perspective of virtualization has then been taken into account, with particular reference to its relation to the VDN. Finally, we have examined possible ways for an evolution of the GAL philosophy in this more challenging and dynamic environment, and we have pointed out possible problems related with the mere introduction of general purpose processors in the virtualized scenario.

Acknowledegement This work was partially supported by the INPUT (In-Network Programmability for next-generation personal cloud service support) project, funded by the European Commission under the Horizon 2020 Programme (Grant no. 644672).

References

1. Bolla R, Bruschi R, Davoli F, Cucchietti F (2011) Energy efficiency in the Future Internet: A survey of existing approaches and trends in energy-aware fixed net-work infrastructures. IEEE Commun Surv Tutor 13(2):223–244
2. Idzikowski F, Chiaraviglio FL, Cianfrani López Vizcaíno AJ, Polverini Ye MY (2016) A survey on energy-aware design and operation of core networks. IEEE Commun Surv Tutor 18(2):1453–1499
3. Maaloul R, Chaari L, Cousin B (2017) Energy saving in carrier-grade networks: a survey. Comput Std Interfaces. https://doi.org/10.1016/j.csi.2017.04.001
4. Cisco visual networking index (2015) Forecast and methodology, 2014–2019. Cisco Systems, San Jose, CA
5. Cisco visual networking index (2016) Forecast and methodology, 2015–2020. Cisco Systems, San Jose, CA
6. Cisco visual networking index (2017) Forecast and methodology, 2015–2020. Cisco Systems, San Jose, CA
7. Ishii K, Kurumida J, Sato KI, Kudoh T, Namiki S (2015) Unifying top-down and bottom-up approaches to evaluate network energy consumption. IEEE J Lightwave Technol 33(21): 4395–4405. Accessed 1 Nov 2015
8. Metro network traffic growth: an architecture impact study, 1st ed. Alcatel-Lucent, 2013. http://resources.alcatel-lucent.com/asset/171568. Accessed 25 April 2016
9. Norton W (2014) Internet Transit. In: The 2014 Internet peering playbook
10. Sandvine (2013) https://www.sandvine.com/downloads/general/global-internet-phenomena/2013/2h-2013-global-internet-phenomena-report.pdf
11. Sandvine (2015) https://www.sandvine.com/downloads/general/global-internet-phenomena/2015/global-internet-phenomena-africa-middle-east-and-north-america.pdf
12. Metro network traffic growth: an architecture impact study, 1st ed. Alcatel-Lucent, 2013. http://resources.alcatel-lucent.com/asset/171568. Accessed 25 April 2016
13. Lee U, Rimac I, Hilt V (2010, April) Greening the internet with content-centric networking. In: Proceedings of the 1st international conference on energy-efficient computing and networking. ACM, pp 179–182
14. Guan K, Atkinson G, Kilper DC, Gulsen E (2011, June) On the energy efficiency of content delivery architectures. In: 2011 IEEE international conference on communications workshops (ICC). IEEE, pp 1–6

15. Choi N, Guan K, Kilper DC, Atkinson G (2012, June) In-network caching effect on optimal energy consumption in content-centric networking. In: 2012 IEEE international conference on communications (ICC). IEEE, pp 2889–2894
16. Osman N, El-Gorashi T, Elmirghani JM (2011, May) Reduction of energy consumption of video-on-demand services using cache size optimization." In: 2011 Eighth international conference on wireless and optical communications networks (WOCN). IEEE, pp 1–5
17. Llorca J, Tulino AM, Guan K, Esteban J, Varvello M, Choi N, Kilper D (2013, April) Dynamic in-network caching for energy efficient content delivery. In: 2013 Proceedings IEEE INFOCOM. IEEE, pp 245–249
18. Mandal U, Chowdhury P, Lange C, Gladisch A, Mukherjee B (2013) Energy-efficient networking for content distribution over telecom network infrastructure. Opt Switch Netw 10 (4):393–405
19. Modrzejewski R, Chiaraviglio L, Tahiri I, Giroire F, Le Rouzic E, Bonetto E et al (2013, December) Energy efficient content distribution in an ISP network. In: 2013 IEEE Global communications conference (GLOBECOM). IEEE, pp 2859–2865
20. Abji N, Tizghadam A, Leon-Garcia A (2015, November) Energy efficient content delivery in service provider networks with content caching. In: 2015 IEEE Online conference on green communications (OnlineGreenComm), pp 23–29
21. Savi M, Ayoub O, Musumeci F, Zhe L, Verticale G, Tornatore M (2015, November) Energy-efficient caching for video-on-demand in fixed-mobile convergent networks. In: 2015 IEEE online conference on green communications (OnlineGreenComm), pp 17–22
22. Jayasundara C, Nirmalathas A, Wong E, Chan C (2011) Improving energy efficiency of video on demand services. IEEE/OSA J Opt Commun Netw 3(11):870–880
23. Fratini R, Savi M, Verticale G, Tornatore M (2014) Using replicated video servers for VoD traffic offloading in integrated metro/access networks. In: 2014 IEEE international conference on communications (ICC), Sydney, NSW, 2014, pp 3438–3443
24. Di Pascale E, Payne DB, Ruffini M (2012) Bandwidth and energy savings of locality-aware P2P Content Distribution in next-generation PONs. In: 2012 16th international conference on optical network design and modeling (ONDM), Colchester, 2012, pp 1–6
25. ITU-T Y.3001 (2011) Future networks: Objectives and design goals. Technical report, ITU-T
26. Pervasive Mobile Virtual Services, Strategic Research and Innovation Agenda, Expert Advisory Group of the European Technology platform Networld 2020, July 2016. https://www.networld2020.eu/wp-content/uploads/2014/02/SRIA_final.pdf
27. Ma KJ, Bartoš R, Bhatia S (2011) A survey of schemes for Internet-based video delivery. J Netw Comput Appl 34(5):1572–1586
28. Popescu A (2014, May) Greening of IP-based video distribution networks: developments and challenges. In: Proceedings of the 10th international conference on communications (COMM 2014), Bucharest, Romania
29. Bolla R, Bruschi R, Davoli F, Donadio P, Fialho L, Collier M, Lombardo A, Reforgiato D, Riccobene V, Szemethy T (2014) A northbound interface for power management in next generation network devices. IEEE Commun Mag 52(1):149–157
30. Green Abstraction Layer (GAL) Power management capabilities of the future energy telecommunication fixed network nodes, ETSI Std. 203 237 version 1.1.1 (2013, March) http://www.etsi.org/deliver/etsi_es/203200_203299/203237/01.01.01_60/es_203237v010101p.pdf
31. RFC1122 Requirements for Internet Hosts - Communication Layers. R. Braden, Ed.. October 1989. (Format: TXT = 295992 bytes) (Updates RFC0793) (Updated by RFC1349, RFC4379, RFC5884, RFC6093, RFC6298, RFC6633, RFC6864, RFC8029) (Also STD0003) (Status: INTERNET STANDARD) (10.17487/RFC1122)
32. Moustafa H, Zeadally S (2012) Media networks. CRC Press, New York
33. Lynch J (2017) Advertisers beware: audiences are taking longer than ever to watch TV shows. Adweek.com, 2016. http://www.adweek.com/tv-video/advertisers-beware-audiences-are-taking-longer-ever-watch-tv-shows-172781/. Accessed 20 Aug 2017
34. Live linear streaming. Comcast Technology Solutions, 2017

35. Arnason B (2017) NCTC OTT Deals include Sony PlayStation Vue and fuboTV—Telecompetitor. Telecompetitor.com, 2017. http://www.telecompetitor.com/nctc-ott-deals-include-sony-playstation-vue-and-fubotv/. Accessed 20 Aug 2017
36. Perfecting the Media Experience, Elemental.com, 2016. https://www.elemental.com/resources/case-studies/bt. Accessed: 21 Aug 2017
37. Sklar B (2013) Digital communications. Pearson, Harlow
38. Proakis J, Salehi M (2008) Digital communications. McGraw-Hill, Boston
39. ITU-T J.83 (2007, December) Digital multi-programme systems for television, sound and data services for cable distribution
40. European Telecommunications Standards Institute (ETSI) EN 300 429 V1.2.1 (1998-04), Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for cable systems
41. DVB-S2 Implementation Guidelines (2015, November) Technical report TR 102 376-1 V1.2.1, ETSI
42. Implementation guidelines for a second generation digital terrestrial television broadcasting system (DVB-T2) (2012, August) TS 102 831 V1. 2.1, ETSI
43. A/53: ATSC Digital Television Standard (2007, January) ATSC
44. "Broadcast-Only Tv Homes Grow 41% in Last Five Year…", ION Media, 2017. https://ionmedia.com/press/imn/broadcast-only-tv-homes-grow-41-in-last-five-years. Accessed 14 Sept 2017
45. Gwatt.net, (2015) G.W.A.T.T.: Global 'What if' Analyzer of neTwork energy consumpTion. http://gwatt.net/intro/5. Accessed 23 Jan 2016
46. Perfecting the media experience, Elemental.com, 2016. https://www.elemental.com/resources/case-studies/bt. Accessed 23 Aug 2017
47. Smith J, Nair R (2005) Virtual machines. Morgan Kaufmann, San Francisco, CA
48. Stavdas A (2010) Core and metro networks, 1st edn. Wiley, Chichester, West Sussex
49. Lefevre L (2015) Towards energy proportional clouds, data centers ad networks: The holy grail of energy efficiency? In: IEEE 2015 Online GreenComm (OGC)
50. Bolla R et al (2013) The green abstraction layer: a standard power-management interface for next-generation network devices. IEEE Internet Comput 17(2):82–86. https://doi.org/10.1109/MIC.2013.39
51. Popek GJ, Goldberg RP (1974) Formal requirements for virtualizable third generation architectures. Commun ACM 17(7):412–421
52. Intel® 64 and IA-32 Architectures Software Developer's Manual, 1st ed., Intel Corp., Santa Clara, CA, 2014
53. Jarschel M, Hoßfeld T, Davoli F, Bolla R, Bruschi R, Carrega A (2015) SDN-enabled energy-efficient network management. In: Samdanis K, Rost P, Maeder A, Meo M, Verikoukis C (eds) Green Communications: principles, concepts, and practice. Wiley, pp 323–338
54. Bolla R, Bruschi R, Carrega A, Davoli F (2014) Green networking with packet processing engines: modeling and optimization. IEEE/ACM Trans Networking 22(1):110–123
55. Bolla R, Bruschi R, Davoli F, Lombardo C, Pajo JF, Sanchez OR (2017) The dark side of network functions virtualization: a perspective on the technological sustainability. In: 2017 IEEE international conference on communications (ICC), Paris, 2017, pp 1–7. doi:10.1109/ICC.2017.7997129
56. Hill MD, Marty MR (2008) Amdhal's Law in the multicore era. IEEE Comput 41(7):33–38
57. ETSI, "NFV Use Cases." http://www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_NFV001v010101p.pdf
58. Nokia 7750 Service Router. http://networks.nokia.com/portfolio/products/7750-service-router
59. Intel, "Introducing the Intel Xeon Processor E5-2600 v4 Product Family. https://newsroom.intel.com/wpcontent/uploads/sites/11/2016/04/intel-xeon-processor-e5-2600-v4-factsheet-x.pdf
60. Alcatel-Lucent, "Global "What if" Analyzer of NeTwork Energy ConsumpTion (G.W.A.T.T.)." http://gwatt.net/intro/1

61. Cisco Systems, Inc (2014, June) The Zettabyte Era: trends and analysis. Technical report
62. Feknous M, Houdoin T, Le Guyader B, De Biasio J, Gravey A, Gijón JAT (2014) Internet traffic analysis: a case study from two major European operators. In: 2014 IEEE symposium on computers and communications (ISCC), Funchal, 2014, pp 1–7. https://doi.org/10.1109/ISCC.2014.6912519
63. Al-Fares M, Loukissas A, Vahdat A (2008, August) A scalable, commodity data center network architecture. In: Proceedings of the ACM SIGCOMM 2008 conference on data communication, Seattle, WA, USA, pp 63–74
64. Rossenhövel C et al (2016) Validating Nokia's IP Routing & Mobile Gateway VNFs. LightReading, 2016. http://www.lightreading.com/nfv/validating-nokias-ip-routing-and-mobile-gateway-vnfs/d/d-id/720902?_mc=RSS_LR_EDT

Chapter 3
Video Distribution Networks: Models and Performance

Adrian Popescu, Yong Yao, Markus Fiedler and Xavier Ducloux

3.1 Introduction

The creation of video content and the distribution over IP networks is a sophisticated process that follows a chain model from the acquisition at the video source, through the production and packaging of the content to the final element of distribution to viewers. A video distribution system contains therefore several parts, which are the contribution, the primary distribution, the secondary distribution and delivery to end users.

Video contribution refers to functions like capturing, initial processing of the video content and initial transport prior to distribution. The video streams may be compressed or uncompressed, depending upon the particular existing conditions. The distribution video is sent from the central broadcast facility to other broadcast facilities for ultimate transmission to end users. Distribution configurations of type one to one (unicast) and one to many (multicast) can be used in this case. There are two categories of distribution systems, the primary distribution and the secondary distribution [1]. The primary video distribution providers are responsible for the transport of video content from the production entity to the secondary distribution entity for further transmission to end users. Primary distribution services are normally using compressed video formats, e.g., MPEG-2 or MPEG-4 or JPEG 2000. The secondary video distribution is about the transport of video content from the primary video distribution providers to end consumers. Examples of secondary distribution services are IPTv and cable TV. Secondary distribution services are normally compressed with MPEG-2 or MPEG-4 standards, with rates of 2–4 Mbps for Standard Definition (SD) systems and 8–20 Mbps for High Definition (HD)

A. Popescu (✉) · Y. Yao · M. Fiedler
Faculty of Computing, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden
e-mail: adrian.popescu@bth.se

X. Ducloux
Harmonic Inc, Cesson-Sevigne, France

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0_3

systems. The distribution configurations are of type one to one (unicast) and one to many (multicast).

Basic operations done on the video signals along the video distribution chain are video coding and compression, encapsulation, forward error correction, transmission, reception and decapsulation, error correction and decompression. Like other categories of service provider networks, video over IP distribution systems are expected to provide services with a good mix of simplicity, scalability, security, manageability and cost-effectiveness. Service Level Agreement (SLA) requirements for video are about parameters like network delay, network jitter, packet loss, availability and loss recovery.

The requirements for the greening of IP-based video distribution networks further complicate the picture. That means greening elements and mechanisms are used. The major system components are the access networks (with different categories of technologies of type wired and wireless), the core networks, the Data Centers (DCs) and the storage networks used to provide Web-based services like IPTv, content distribution networks and cloud-based services.

The chapter is organized in such a way to present some of the main components involved in video distribution networks. Section 3.2 presents elements of video distribution, Sects. 3.3 and 3.4 are about video coding and compression. Sections 3.5, 3.6 and 3.7 provide an overview of video streaming. Section 3.8 presents modern streaming solutions. Section 3.9 presents a short overview of the main video traffic models. Section 3.10 is about energy consumption models for video flows. Sections 3.11 and 3.12 report on the problem of performance optimization in video distribution networks. Section 3.13 is about QoE- and energy-optimal video streaming. Finally, Sect. 3.14 concludes the chapter.

3.2 Video Distribution

The creation, distribution and delivery of video content is a sophisticated process with elements like video acquisition, preprocessing and encoding, content production and packaging as well as distribution to customers. IP networks are usually used for the transfer of video signals.

The treatment of video content is also very complex, and we have a multidimensional process with elements like content acquisition, content exchange and content distribution [2].

The Internet is today undergoing an adaptation process to provide large demands for bandwidth increase, where one of the most important contributors is the video traffic. The appearance of new bandwidth-demanding Over-The-Top (OTT) video streaming services like Netflix and Skype-like video communications in combination with the growing number of multimedia users has accelerated this process. Another complication is because the Internet has democratized the process of creation, distribution and sharing of video like for instance in the case of YouTube. The adoption of more advanced video formats like the Ultra High Definition

(UHD), defined and approved by the International Telecommunication Union (ITU), means that even more bandwidth is needed. UHD is intended to be used for displaying with an aspect ratio of 16:9 and at least one digital input is capable of carrying and presenting native video at a minimum resolution of 3840 × 2160 pixels [2].

Video distribution networks are therefore distribution networks with elements like encoders, transcoders, multiplexers and decoders as well as networking elements to provide support for professional video distribution and delivery. The basic operations are capturing and initial processing of the video content, initial transportation prior to distribution, primary and secondary distribution and finally delivery to end users.

These networks can be of category terrestrial or satellite or cable television networks. The first category, also known as broadband networks, uses the IP technology. These networks are targeting individual users although multicast and broadcast mechanisms can turn them into multicast networks as well.

Today we have a multidimensional process for the treatment of video content, where the most important elements are content acquisition, content exchange and content distribution. Furthermore, basic operations done on the video signals along the video distribution chain are video coding and compression, encapsulation, forward error correction, transmission, reception and decapsulation and decompression. Like in the case of other categories of service provider networks, video over IP distribution systems are expected to provide services with a good mix of simplicity, scalability, security, manageability and cost-effectiveness. Service Level Agreement (SLA) requirements for video are used to define the service requirements. These requirements refer to parameters like network delay, network jitter, packet loss, availability and loss recovery.

3.3 Video Coding and Compression

Image and video coding and compression have been the area of intensive research activity and development of coding standards to achieve low bit rate for data storage and transmission, while maintaining acceptable distortion. The general structure of an image coding process contains several elements: image partitioning, transformation (to decorrelate the signal), quantization (to reduce the amount of information required to be stored or transmitted) and entropy encoding. Also, movie files of videos are usually combined with associated audio information and encapsulated in the so-called video containers [3]. The problem here is to keep the average percentage increase low.

Major coding predecessors are H.262/MPEG-2, and H.263 [4]. Today, some of the most popular standards that support efficient video delivery storage and provide higher coding efficiency than legacy standards are H.264/MPEG-4 AVC (Advanced Video Coding) used by nearly all services, VC-1 and VP8 as well as the latest international video standard High Efficiency Video Coding (HEVC) [4, 5].

Table 3.1 Features main standards for video coding and compression [5]

Standard	Key features
MPEG-1	Developed for audio/video storage on CD-ROM; motion vectors coded losslessly
MPEG-2	Supports video on DVDs, standard/high definition TV; supports scalable extensions
MPEG-4	Supports video on low-bit rate multimedia on mobile platforms and Internet and also object-based coding
H.261	Developed for video conferencing over ISDN; support for CIF and QCIF resolutions
H.262	Standardized as MPEG-2 Part-2
H.263	Improved quality compared to H.261 at lower bit rate
H.264 AVC	Significantly improved picture compared to H.263, at low bit rates, but increased computational complexity
H.265/ HEVC	Support for ultra HD video; greater flexibility in prediction modes; support for parallel processing
VC-1	Adaptive block transform for easier implementation in hardware devices
VP8	Developed to operate in low bandwidth environment such as Web video
VP9	Basic structure simpler than VP8

H.264 is standardized by the International Telecommunications Union (ITU), whereas VC-1 is standardized by SMPTE 421 M and it was initially implemented by Microsoft as Windows Media Video (WMV) 9 for supporting online video streaming. Finally, VP8 is an open source video codec formerly owned by Google but released later as part of the WebM project. Detailed description of key features of the MPEG, H.26x, VC-1, HEVC and other popular video coding standards is provided in [4, 5]. Among others, it has for instance been observed that HEVC encoders can achieve equivalent subjective reproduction quality like H.264/ MPEG-4 AVC encoders using 50% less bit rate on average [4].

A detailed description of standards for video coding and compression is provided in [3–5]. Main characteristics of these standards are provided in Table 3.1 [5].

3.4 Recent Developments

There has been a phenomenal growth over the last years in video applications and transport, which is because of, e.g., an ever-increasing number of users upload and download video signals using sites like YouTube. Applications like HD DVD and video streaming (Netflix, Hulu) are increasing in popularity as well. Video calling over the Internet (by using, e.g., Skype, Facetime) is also increasing in popularity. Video conferencing systems like Cisco TelePresence and WebEx are popular today

in different business and organizations. The consequence of this growth in combination with the growing number of multimedia users is that recent forecasting studies done by Cisco predict that the video traffic is expected to increase to 80–90% of global consumer traffic by 2021 [6].

However, the design of robust and reliable networks and services to provide expected performance in terms of best end user QoE and minimum e2e energy consumption is becoming increasingly difficult. For doing this, one needs to first have a detailed understanding of the network and traffic characteristics. Furthermore, from the viewpoint of a network service provider, the demands on the network are not easily predictable. This aspect must be also correlated to the demands to accurately estimate the e2e network performance. This in turns demands for accurate traffic models able to capture the statistical characteristics of traffic in the network as well as good understanding of the process of video streaming.

3.5 Video Streaming

Basically, streaming means that people are listening to music or watching video in real time, instead of first downloading a file to the computer and listening or watching it later. The client starts the content playback a few moments after it begins receiving the content from the server. This method is different from the classical method of normal file download where the entire file is first downloaded before starting up the playback.

Streaming media refers to multimedia content (video or audio) sent in compressed form over the Internet and played in real time instead of first being saved to the hard drive. This means that a Web user does not need to wait to download a file and to play it, but it plays data stream as it arrives.

There are several content delivery methods [7]:

- Traditional streaming
- Progressive download
- HTTP-based adaptive streaming

A good example of traditional streaming is the Real Time Streaming Protocol (RTSP), which is a stateful protocol where the users interact with the media content provider. RTSP is used together with the Real-time Transport Protocol (RTP), which is designed for use with real-time traffic on the Internet.

Today the move is, however, towards HTTP-based streaming where the media delivery is adapted to the Internet instead of trying to adapt the entire Internet to streaming protocols.

The progressive download streaming method is a method based on downloading from a HTTP Web server. The player client plays the media back while the file download is still in progress. This means the user can, e.g., pause the streaming,

waiting for the download to finish, allowing so a smooth playback when the user decides to play the media. The drawback is that this is not perfect and not very flexible in selecting the parameters for downloading.

A better method is the HTTP-based adaptive video streaming, which is a hybrid method of progressive download and streaming. Here the video is cut into short segments and encoded with the specific delivery format and rate. The segment length may vary depending upon the implementation, but they are typically several seconds long. The main advantage of this method is given by the adaptive properties of the streaming process. Adaptive facilities like available network resources (e.g., bandwidth), device facilities (e.g., display resolution, available CPU) and streaming conditions (e.g., playback buffer size) may be used in the streaming process. The video streaming client can also adjust its selection of encoded segments based, e.g., on the communication link throughput. A good example of HTTP-based adaptive video streaming standard is MPEG-DASH, which stands for MPEG Dynamic Adaptive Streaming over HTTP, also known as ISO/IEC 23009-1 [8]. The International Organization for Standardization (ISO) published the MPEG-DASH standard in April 2012.

Basically, live streaming requires some form of source media (e.g., video camera, audio interface), an encoder to digitize the content, a media publisher and a CDN to distribute and deliver the content. Furthermore, a user needs a player entity, i.e., software that uncompresses and sends video data to display and audio data to speaker. The storage size is calculated from the product of streaming bandwidth and media length. Examples of streaming technologies are Adobe Flash, Apple QuickTime, Microsoft Windows Media and Silverlight. Streaming technologies typically use compression to shrink the sizes of audio and video files so they can be retrieved and played by remote viewers in real time. The codecs are usually independent of the streaming technology that implements them.

There are several transport protocols that can be used in video streaming. These protocols are Hyper Text Transfer Protocol (HTTP, this is, however, not a streaming protocol, but used to distribute small files), Real Time Streaming Protocol (RTSP), Real-time Transport Protocol (RTP), Real-time Transport Control Protocol (RTCP), Real-Time Messaging Protocol (RTMP) and Real Data Transport (RDT). Details about these protocols as well as other relevant information are provided in different books on computer networking.

A broadband speed of at least 2 Mbps is recommended for streaming standard definition video without experiencing buffering, especially live video, e.g., to Apple TV, Google TV or Sony TV Blu-ray Disc Player. Under such circumstances, streaming media storage sizes of, e.g., about 128 MB is needed for one hour of video encoded at 300 kbps [9]. In the case of live video with 3000 viewers, then the media storage size increases to almost 2×10^6 MB [9].

3.6 Streaming Process

Figure 3.1 shows an example of video streaming process encountered at video download [10]. As observed in the figure, there are two phases in a video streaming process, namely the buffering phase and the steady state phase [10]. In the first phase, the data transfer rate (indicated by slope) is decided by the end-to-end bandwidth at the particular time moment. The video playback starts up as soon as sufficient data is available in the buffer, which means that the video playback process does not need to wait till the end of the buffering phase.

The steady state phase is a state where the average download rate is larger than the video encoding rate. The so-called accumulation ratio is used to express this [10]. An accumulation ratio of at least one is necessary to avoid the video playback getting interrupted because of empty buffers [10].

Data downloading in the steady state phase is done by transferring blocks of video content, which creates the ON–OFF cycles observed in the figure. During the ON period, a block of data is transferred under the condition of the end-to-end bandwidth limited by TCP. On the other hand, the TCP connection is closed during the OFF periods. This means that the slope of data downloading during the steady state is an indication of the end-to-end bandwidth.

A sufficient large amount of data is needed to compensate for the variance in the end-to-end bandwidth during video playback. The reduced transfer data in the steady state phase ensures that the video player is not overwhelmed.

The buffer size needed in the steaming process for a single user and file is given by the product of streaming bandwidth (in bit/s) and media length (in seconds) [11].

$$\text{Storage\_size (MBytes)} = \frac{\text{Media\_length (s)} * \text{Bit\_rate (bit/s)}}{8 * 1024 * 1024}$$

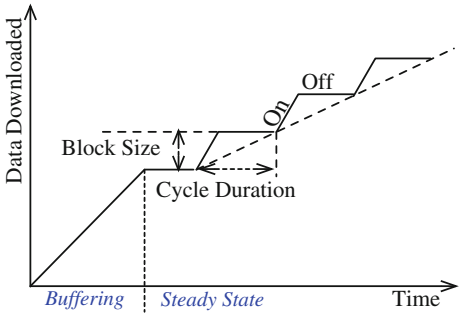


Fig. 3.1 Typical process of video download [10]

In the case of using unicast transport protocol, an increased number of receivers for the same document increases accordingly the storage size.

3.7 Streaming Strategies

As mentioned above, data streaming process involves two phases: buffering phase and steady-state phase. There are three strategies that can be used, e.g., in Netflix and YouTube, as indicated in [10]. These are as follows:

- No ON-OFF cycles: all data is transferred during the buffering phase, which means there is no steady state phase. In other words, we simply have a file transfer session in this case, but the risk exists for overwhelming the player. The advantage is that this method is simple, but measures must be taken to avoid the risk of overwhelming the player.
- Short ON–OFF cycles: this is a simple strategy, where measures must be taken that the client is not overwhelmed by the data sent by the server. The goal, in this case, is to maintain an accumulation ratio slightly larger than one.
- Long ON–OFF cycles: large amounts of data are transferred in a cycle, which demands for careful buffer dimensioning.

A general observation is that the three streaming strategies may have different impacts on the loss rate. This observation must be considered in the development of streaming strategies. This is because the wasted bandwidth associated with a particular streaming strategy may be different, which depends upon the particular streaming and networking conditions.

3.8 Transcoding

HTTP-based adaptive video streaming solutions have been widely deployed in the last decade due to the facility of simple deployment on the existing HTTP infrastructure combined with simplicity for end user device. Instead of building a complex scalable stream that requires a sophisticated decoder, video streaming solutions store in the video server as many representations (in terms of resolutions and bit rates) as required by the existent network bandwidth and the end user devices (Fig. 3.2). The drawback of these solutions is related to the storage in the origin server, which may become large. Moreover, the coexistence of many video streaming protocols forces the operators to duplicate the encoded content in various formats. For instance, to stream everywhere, Netflix encodes each movie more than 120 times.

The first phase in the evolution of streaming solutions was to implement the Just-In-Time packaging in the requested format. Just-In-Time Transcoding is a

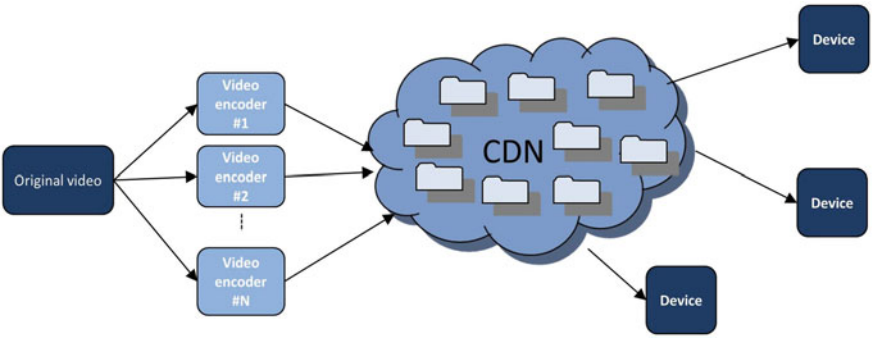


Fig. 3.2 HTTP-based video streaming—Just-In-Time Transcoding

natural evolution of the architecture, which provides storage savings. Only the nominal representation with the highest bit rate is stored in the server and the other representations, with lower bit rates and possibly lower resolutions, are produced on the fly from the nominal representation (Fig. 3.2).

Today it is observed a clear move towards Just-In-Time Transcoding, which is because of several reasons:

- Video processing is more and more off-loaded in the cloud. Ingest of video content in servers is simplified in the case of Just-In-Time Transcoding as it is limited to the nominal stream
- Streaming solutions are more and more flexible in terms of resolution, frame rates and bit rates
- Operators are more and more sensitive to rising costs for storage; this is in close relationship with energy savings as the storage cost includes energy cost.

The penalty of this approach is in form of more processing required in the network to transcode the nominal content. This creates a big pressure on servers at peak viewing times.

The legacy Just-In-Time Transcoders are based on a very simple architecture, aggregating a decoder and a full encoder, which is not optimal in terms of computational cost. More and more developments are therefore presented in the 'Guided Transcoding' solutions [12], which use assistance metadata to do the transcoding process with lower processing and latency (Fig. 3.3). The metadata takes advantage of the encoding correlation that exists among the various coded representations of a given video content, at least in the motion field, which is a very expensive part of a classical encoder.

Using the assistance metadata, a Guided Transcoder complexity can be reduced to twice a decoder complexity, which means a huge reduction of complexity compared to the legacy Transcoder (−99%). At the same time, the storage size of metadata in the server can be limited to a few percent of the nominal stream size.

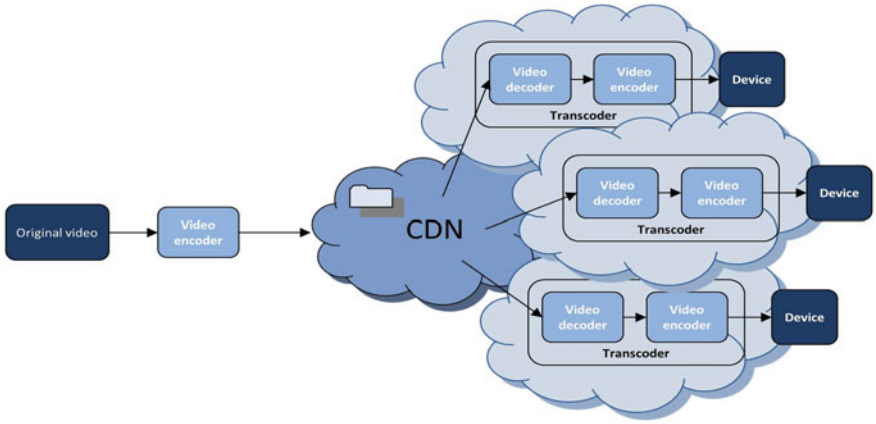


Fig. 3.3 HTTP-based adaptive video streaming—Guided Transcoding

The low complexity of the Guided Transcoder does not limit it to centralized transcoding functions in the Cloud and gives a lot of opportunities for deployment in edge nodes or home gateways.

ISO/IEC MPEG standardization body is currently foreseeing to standardize interfaces for Guided Transcoding applications to ease interoperability and boost their deployment and has recently launched a Call for Evidence at its 119th meeting [12].

3.9 Video Traffic Models

A good video traffic model is expected to be able to capture characteristics of video sequences and predict the network performance. A large number of video traffic models have been suggested to capture and to describe different statistical properties of video traffic like encoding formats, creating synthetic loads and others. A general comment is that the standards have been designed such as to provide the developers of encoders and decoders as much freedom as possible to customize their implementations. This is essential in order to let the particular standards be adapted to a wide variety of platform architectures, resource constraints and application environments.

There are two main characteristics of video traffic that are relevant for predicting the network performance for video transfer [13, 14]. These are the distribution of frame sizes and the Autocorrelation Function (ACF) that captures common dependencies among frame sizes in VBR video traffic. The problem of capturing

the ACF structure of VBR video traffic is challenging because this traffic has both Short-Range Dependent (SRD) and Long-Range Dependent (LRD) statistical properties. A detailed description of these statistical properties is presented in [15].

Mathematically, the difference between SRD and LRD processes can be stated as follows [16].

In the case of SRD process with the degree of aggregation m the following properties are representative:

- The mean value $E(X^m)$ approaches second order pure noise as $m \to \infty$
- $\mathrm{Var}(X^m)$ is asymptotically of form $\frac{\mathrm{Var}(X)}{m}$ for large m
- $\sum \mathrm{Cov}(X_n, X_{n+k})$ is convergent
- Spectrum $S(w)$ is finite at $w = 0$

On the contrary, LRD processes with the degree of aggregation m are characterized by the following properties:

- The mean value $E(X^m)$ does not approach second order pure noise as $m \to \infty$
- $\mathrm{Var}(X^m)$ is asymptotically of form $m^{-\beta}$ for large m
- $\sum \mathrm{Cov}(X_n, X_{n+k})$ is divergent
- Spectrum $S(w)$ is singular at $w = 0$

Good video traffic models for real video sequences should satisfy several criteria [17]. First, the particular model should match basic statistical characteristics like, e.g., probability distribution function (pdf), mean value, variance, peak, autocorrelation and coefficient of variation of the bit rate [18]. Second, the synthetic video sequence must be similar to the real video sequence. The third criterion is that the model must be simple and the synthetic trace generation must have low computational complexity. Finally, the last criterion is that the model must be able to characterize a wide range of video sources with low to high motion activity.

As a general comment, the existent video traffic models have been developed independently of each other, and today there exist six general classes of Variable Bitrate (VBR) traffic models [15, 17]:

- Autoregressive (AR) models
- Markov process (MP) models
- Self-similar (SS) models
- Wavelet (WV) models
- Fluid flow models
- Other models

The traffic models are basically focused on capturing different types of video and encoding formats by using different modelling techniques. The models are further focused on different attributes of the video and encoding standard as well as other important characteristics.

3.10 Energy Consumption Models

Typically, a video distribution system is composed of various network components in terms of hardware, software and network solutions. Examples of hardware are Headend, routers, switches, WiFi access points and mobile terminals. The diversity of network components indicates the complexity in describing the video flow through the system. The video flow is defined to be an end-to-end (e2e) data transmission from the video streaming source side (e.g., Headend) to the video consumer side (e.g., mobile terminals).

Moreover, a video flow is usually associated with a particular application scenario (e.g., mobile television, mobile cloud gaming), which means a set of specific requirements for the practical implementation, defined with respect to different parameters like, e.g., QoE, QoS and power consumption.

As shown in the Fig. 3.4, there are four categories of entities traversed by a video flow:

- Video Data Center (VDC): responsible for streaming the video data to video consumers. For instance, for both OTT and IPTV models, a key component of VDC is the Headend
- Transport: network elements between the Headend and the delivery part of the network
- Delivery: it is about the particular network solution to deliver the video data from VDC to video consumers. For example, a suitable network solution for delivering the video data in the OTT model is the Content Distribution Network (CDN)
- Video consumers: they are connected to various terminal devices (e.g., mobiles, Set-top box) that consume the video data

The traffic-related power consumption model for N different traffic flows traversing an individual networking device d_m during a given time slot $[t_k, t_{k+1}]$ is given by [19]:

$$P_{\text{traffic}}(m, t_k) = \sum_{n=1}^{N} P_{\text{traffic}}^{(n)}(m, t_k) = \sum_{n=1}^{N} \left[P_{S\&F}^{(n)}(m, t_k) + P_{\text{Pro}}^{(n)}(m, t_k) \right]$$

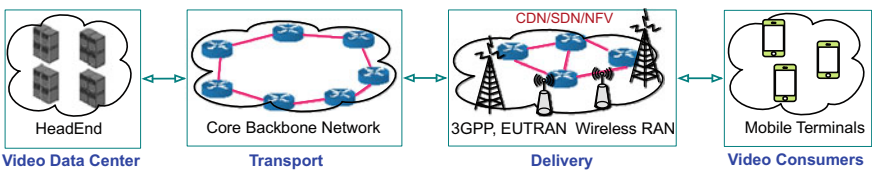


Fig. 3.4 Typical e2e video flow through mobile multimedia network

For the above equation at the time moment t_k, $P^{(n)}_{\text{traffic}}(m, t_k)$ means the traffic-related power consumption by the device d_m. Further, $P^{(n)}_{S\&F}(m, t_k)$ means the power used for storing and forwarding the data (i.e., in terms of packets) by the device d_m, while $P^{(n)}_{Pro}(m, t_k)$ means the power used for processing the data by the device d_m.

The total energy consumption used by M networking entities to serve a given traffic flow f_n is given by [19]:

$$\widetilde{E}^{(n)}_{\text{traffic}}(\Delta t) = \sum_{m=1}^{m} \left\{ \overline{E}^{(n,pkt)}_{\text{incre}}(m) \int_{\tau=t}^{t+\Delta t} \beta^{(n)}(m, \tau) d\tau \right\}$$

It is assumed that, for traffic flow f_n, the *incremental energy per packet* $E^{(n,pkt)}_{\text{incre}}(m, \tau)$ at device d_m does not change during the time interval $[t, t + \Delta t]$. In other words:

$$E^{(n,pkt)}_{\text{incre}}(m, \tau) = \overline{E}^{(n,pkt)}_{\text{incre}}(m)$$

This assumption is applicable under different cases like, e.g., when there is no traffic congestion experienced by the device d_m.

Furthermore, this refers to the variable part in the power consumption profile for an e2e chain of equipments traversed by a traffic flow. This is also known as the *energy consumption due to packet forwarding* [19]. The total e2e energy consumption is therefore given by the sum, taken over all M devices in the e2e route, of individual energy consumptions. The energy consumption of an individual device is given by the product of incremental energy per packet multiplied by the packet rate.

3.11 Performance Elements

When discussing performance modelling, evaluation and monitoring of networking solutions, three concepts emerge: Quality of Service (QoS), Quality of Experience (QoE) and energy consumption.

QoS is normally defining the performance at lower layers in the TCP/IP protocol stack, i.e., from the physical layer up to network layer, and even transport layer. QoS typically refers to individual networking domains. Examples of relevant QoS parameters are throughput (bit rate), frame or packet loss, delay, delay variations (PDV) and out-of-order packets. For instance, the throughput $B(T_i)$ is defined to be the number of data units (bit, bytes, packets) $DU(T_i)$ observed during a given time interval $T_i = t_i - t_{i-1}$

$$B(T_i) = \frac{DU(T_i)}{T_i}$$

On the other hand, QoE reflects the end user subjectively perceived experience of the quality of the particular service and it is a subjective measure. QoE is, in fact, an e2e parameter, which captures the effects of the complete e2e system. This means the network between source (Headend) and destination (Terminal) can be treated as a black box.

QoE is dependent on the QoS parameters characterizing the black box. The impact of each QoS parameter on QoE differs depending upon the application type. For instance, in real-time video streaming the irregularities of QoS parameters can effectively be considered as packet loss. The multidimensional relationships existing between QoE and QoS parameters and the definition of the so-called Key Performance Indicators (KPIs) are very difficult questions to be answered [20]. Definitions of QoE include keywords like 'quality of perception', 'user behaviour' and 'psychological measure', whereas QoS refers to parameters relevant for network like throughput, e2e delay, jitter and error rates. User-centric measurements are normally considered in combination with parameterization methods like Mean Opinion Score (MOS), Degradation Opinion Score (DOS) and Media Delivery Index (MDI) [20].

The standardization activity for video QoE and QoS has been very intense. Most active standardization organizations are ITU, ETSI, IETF and IEEE. Table 3.2 shows, e.g., part of the ITU activity [21].

Furthermore, it is also important to mention that today little work is available on end user perception of offered services and their dependency on network quality, especially for Internet-based applications and networks. There are simply still gaps between the network measured QoS and the user-perceived QoE [20, 21]. This indicates the strong need existing today for developing of mechanisms, procedures and tools to continuously monitor, operate and report QoE indicators function of QoS parameters. Of particular importance is the development of mechanisms for QoE monitoring for network optimization, supervision and operation purposes for video distribution networks, under the condition of minimum e2e energy consumption.

Figure 3.5 shows the case of a real network of type inter-provider model, where two or more network providers share the end-to-end (e2e) path and, associated with

Table 3.2 ITU-T standardization activity on video QoE [21]

Image resolution	Subjective estimation	Full reference
SDTV/HDTV	ITU-R BT.500; ITU-T J.140; ITU-T J.245	ITU-T J.144; ITU-T BT.1683
VGA/CIF/QCIF	ITU-T P.910; ITU-T P.911	ITU-T J.247

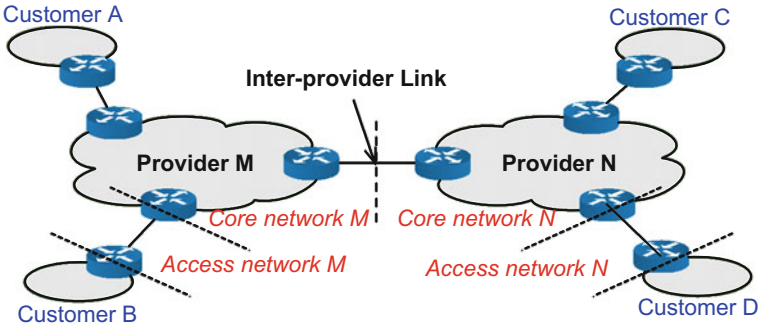


Fig. 3.5 Real networking scenario [22]

this, the existent resources between the two edges [22]. By resources, it is meant both physical resources and other categories of resources (e.g., bandwidth).

With reference to the example shown in the above figure, an important aspect is that the two service providers M and N are expected to offer services, which are concatenated with the services from the other provider, and provide so the end-to-end (e2e) service to end user. For instance, in the case of a given maximum e2e delay, the individual allowable delay may need to be negotiated among multiple carriers. In a similar way, the minimum e2e channel bandwidth can be negotiated and agreed among the participating teleoperators. The conclusion, therefore, is that there is need for monitoring the e2e performance experienced by a customer, which means that all service providers are involved. This also means that a number of interconnection methods need to be available to provide the e2e service. This does not refer to IP interconnection only, but also to other layers like Multi-Protocol Label Switching (MPLS) and layer 2 interconnection solutions (e.g., Ethernet) [22]. Other important elements are regarding the interfaces, of type Customer Edge (CE), Provider Edge (PE) and Customer to Provider Edge (CPE), Service Class (SC) and, related to this, the associated Service Level Agreements (SLA) [22–24].

Another important question is regarding the definition of Quality of Service (QoS) constraints for an e2e packet flow. The constraints for a packet flow are generally specified for the entire e2e path, which may traverse several networking domains owned by different Internet Service Providers (ISP). The problem is that each ISP allocates bandwidth and provides particular QoS guarantees independently from other providers [24]. This means there is need for cooperation to combine and to coordinate the QoS guarantees for the multi-domain routing specified by the entire e2e path shown, e.g., in Fig. 3.5.

Using of average values as performance indicators for performance metrics like, e.g., the response time of a Web service or for controlling the power budget in green Data Centers (DCs) may not provide the best e2e solution. This is because average values can be misleading, they do not represent the range of values the metric under study may take. A better performance indicator, in this case, is the percentile of the particular metric, which statistically bounds the behaviour of the system [24]. The

q-th percentile (e.g., the 95th percentile) of a variable X is defined to be the value below which a given percentage of observations in a group of observations fall.

Furthermore, another difficult problem that must be solved is regarding how to add percentiles and how to partition them into individual components. Several interesting solutions are reported in [24] on the calculation of the weight function for the cases of Gaussian individual distributions, exponential individual components with identical rate parameters, exponential components with different rate parameters and two-stage Coxian distribution. Other interesting solutions are reported in [22], where the authors addressed the issue of how to allocate the e2e response time, packet loss and jitter across multiple networks owned by different operators. The authors of [22] suggest using mean values for response time and percentile for inter-arrival time at destination for jitter in combination with a particular method suggested for adding the jitter. It is reminded that the jitter is the one-way IP packet delay variation (IPDV).

Another important question related to the evaluation of performance on an e2e basis is regarding the particular traffic model that can be used for the video flow. As described above, there are several categories of models that can be used for this purpose, each of them with own advantages and drawbacks. Given that such models must be also related to the e2e performance in terms of QoS parameters and energy consumption as well as the end user performance in terms of QoE, the concept of decomposition of general tandem queueing networks with Markov-Modulated Poisson Process (MMPP) input [25] can be used as well as fluid flow models. The main advantage, in this case, is simplicity in combination with precise analysis.

The advantage, in this case, is that the network can be partitioned into individual elements/nodes that are analyzed in isolation. Further, when using decomposition algorithms for continuous-time analysis of networks, the output of a queueing system (particular network) is usually approximated by a renewal process, which serves as the arrival process to the next queueing system. By doing this, one can easily obtain models based on MMPP, which are simple and easy to use for doing sophisticated analysis studies [25]. The drawback is that this model (tandem queueing networks) does not lend themselves to an exact analysis, especially because of concurrent non-exponential activities.

3.12 Performance Optimization

As shown in Table 3.3, the main QoS parameters that influence the experienced service satisfaction (i.e., the QoE) at video services are the throughput and the jitter. Given that the e2e throughput performance is limited by the networking element with the minimum throughput performance, this means important efforts must be laid on the provision of good e2e performance in terms of maximum e2e throughput and minimum e2e energy consumption and not only individually at the different networking elements.

Table 3.3 Examples of service satisfaction criteria [21]

Web service	Web paging loading time AND Active session throughput
Video streaming	Streaming media active throughput AND Streaming media active transport jitter AND Streaming media active loss
Video over HTTP	Active session throughput
VoIP	Call setup success ratio AND Call setup time AND Call cut-off ratio AND Speech quality

Given a particular networking configuration composed of a number of connected networking elements like, e.g.,

Headend—Edge Cloud—Wide Area Network—Radio Access Network—Terminal, every networking element can be characterized by a set of four performance indicators, as follows:

Energy Consumption E—Throughput T—Quality of Service QoS (expressed in terms of other parameters than Throughput, specific for the particular networking element like, e.g., error rate for RAN, execution time/delay for Edge Cloud)—Quality of Experience (at Terminal only).

The goal of the performance optimization algorithm is to provide the best performance with regard to, e.g., best possible QoE at the end user and minimum e2e energy consumption.

There are two optimization algorithms involved in this process.

- Optimization algorithm #1: best end user QoE performance AND minimum e2e acceptable energy consumption

Accordingly, this involves the following

- Optimization algorithm #2: best QoS performance for the individual networking entities with reference to the above-mentioned optimization algorithm; examples of QoS parameters are throughput, error rate, jitter

The QoS parameters for individual networking entities can be different from networking element to networking element, and they depend upon the specific particularities of the individual networking entity.

For instance, 3GPP suggests the following metrics (so-called Key Performance Indicators KPIs) for variable traffic in mobile networks [26]:

- Mean, 5th, 50th, 95th percentile user throughput
- Served (cell) throughput
- Harmonic mean normalized cell throughput
- Normalized cell throughput
- Resource utilization

In particular, the following metrics suggested by 3GPP to trade-off the quality versus energy consumption are relevant [26]:

- Mean user throughput
- Throughput Complementary Distribution Function (CDF)
- Median and 5th percentile (worst) user throughput

The throughput is a measure of the rate at which data is successfully transmitted through the network or through the network entity. This corresponds to the amount of useful data transfers, and it is generally less than the amount of data injected into the network. This is because of eventually lost, corrupted, retransmitted or mis-delivered packets. Sharing the network resources also lowers the throughput for every competing user. Finally, diverse protocol overheads contribute to reducing the throughput as well. Maximum obtainable throughput is therefore defined to be equivalent to the system's capacity.

Typical units of measure for throughput are bit/s, byte/s and packets/s. Recommended values for the time interval are one to five minutes interval [26].

Besides throughput, other relevant QoS parameters are error rate, jitter and e2e delay. The point is that these parameters can be translated into the throughput performance for the particular networking entity, as it is done for instance at Transport Control Protocol (TCP) [27].

The fundamental law governing the behaviour of networking nodes must be respected in every node as well. This is about the so-called 'flow-conservation law' [28], which states that:

[Flow-out-of-a-node]—[Flow-into-a-node] = [Network-supply-at-the-node]

where by 'network supply' we mean auxiliary flow(s) that enter the particular node, with the purpose of maintenance and control.

Given these elements, the system for performance optimization has two parts:

- Optimization of low-layers networking elements, which refers to layers 1–3 (PHY/DLL/NL) in the TCP/IP protocol stack.
- Optimization of the protocols in the layers above (MPEG-DASH/HTTP/TCP)

This concept is valid for all categories of architectures and scenarios.

The general procedure for performance optimization is as follows [19]:

- Define the block scheme and the component elements like, e.g., access (eNodeB, HGW, …), backhaul and transport (switching …), core network (element1…) and servers (Data Center …).
- Measure and model the power consumption of each element in steady state and also when serving video flows.
- Measure and model the throughput provided by each networking element.
- Do performance optimization for the individual networking elements in terms of best trade-off between the minimum energy consumption and highest possible throughput.
- On an e2e perspective, the throughput of individual networking elements is also related to the throughput of the other networking elements in the sense that one should avoid, as much as possible, low values for any of them. In other words, the goal is to provide as high as possible e2e throughput, which means high

throughput in all networking elements, under the condition of minimum energy consumption in all networking elements. This concept is also depending upon the particular testbed. It is well known that the minimum individual throughput in the e2e chain of networking elements, also known as the bottleneck bandwidth [29], determines the throughput experienced by the end user, which may deteriorate the QoE performance in the case of low values.

- Do performance optimization at low layers NL/DLL/PHY for the whole e2e networking chain with regard to highest possible e2e throughput, best (terminal) QoE performance and minimum possible e2e energy consumption.
- Do performance optimization at high layers MPEG-DASH/HTTP/TCP for the e2e networking chain with regard to maximum e2e throughput and minimum error rates.
- Do system performance optimization for the whole e2e system with regard to the best trade-off: minimum e2e energy consumption AND highest e2e system throughput AND best QoE performance; this refers to the whole protocol stack MPEG-DASH/HTTP/TCP/NL/DLL/PHY.

The modelling and evaluation activities take into consideration the architectural blocks and the temporal behaviour of component elements as well. It is also assumed that the total traffic coming into the network is equal to the total traffic leaving out of network. A unit of work is defined to be the work associated with the transport of traffic accepted from one external interface to another external interface.

3.13 QoE- and Energy-Optimal Streaming

An important goal in providing the performance for streaming services is in form of QoE- and energy-optimal streaming. In order to yield the highest possible QoE in terms of delight while avoiding annoyance [30], video streaming services should provide highest possible resolution while avoiding freezes. The image freezes may appear due to a temporary or permanent mismatch between the required and the available throughput. The maximal throughput that yields a freeze-free operation is denoted by sustainable throughput [31], which is a measure for QoE-optimal streaming. The higher the sustainable throughput, the better quality can be expected. However, once the sustainable throughput is bypassed, disturbances (in form of, e.g., image freezes) turn up.

For instance, the popular standard Dynamic Adaptive Streaming over HTTP (DASH) [32] is based on the concept of allocating the sustainable throughput. Basically, DASH is a standard for optimizing the multimedia delivery over the open Internet. It reacts to varying network quality (e.g., low throughput, extra delays and packet losses), yielding a drain of the de-jitter buffer, by lowering the throughput until stable operations without freezes can be ensured.

To better understand the DASH dynamic control actions towards the QoE-optimal streaming, a static scenario with preselected quality levels is

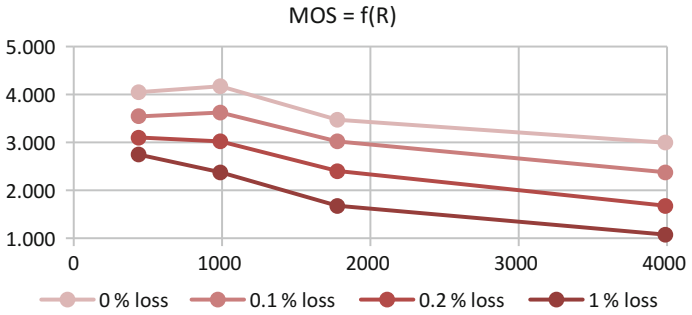


Fig. 3.6 Mean Opinion Scores as a function of rates (in kbps) for different additional loss levels

considered as an example. A set of QoE estimations are reported, as obtained in December 2016 by asking 40 users of the BTH Eduroam WiFi network about their opinion scores (OS) of Youtube videos with different resolutions (240, 360, 480 and 720p). Eduroam is the Swedish national academic WiFi network. The QoE is defined to be the mean opinion score (MOS) of all 40 users as a function of video resolution R.

Figure 3.6 shows the $MOS(R)$ for each resolution R, on the scale: 1 = bad; 2 = poor; 3 = fair; 4 = good; and 5 = excellent. The confidence intervals are rather small ($\pm$ 0.2…0.25 on the MOS scale) and are omitted for the sake of brevity.

The QoE estimations are combined with an extrapolation of the end-to-end energy measurement results obtained in the CONVINcE project [19, 31]. This way, one can obtain the trade-off between the QoE (described by the mean opinion score MOS of all 40 users as described above) and the energy as a function of the video resolution R. The trade-off is captured by the parameter Quality of Experience per Joule $QoEJ(R)$, which expresses the gain on the MOS scale per extra Joule, and it is a function of the resolution R:

$$QoEJ(R) = (MOS(R) - 1)/E(R)$$

Obviously, the higher the $QoEJ(R)$, the better the quality-energy trade-off becomes.

The function $E(R)$ is extrapolated from the CONVINcE end-to-end energy consumption results [19, 31], focusing on HEVC FHD (R = 1080p) and HD (R = 720p), respectively. Two models are obtained:

1. Linear extrapolation: $E_{\text{lin}}(R) \approx 40\text{ J} + R/\text{p} \times 0.22\text{ J}$
2. Exponential extrapolation: $E_{\text{exp}}(R) \approx \exp(0.0009\ R/\text{p}) \times 102\text{ J}$

While the linear extrapolation is a default choice, the exponential extrapolation inherently assumes a higher idle consumption (102 J instead of 40 J).

Table 3.4 illustrates the measured MOS (serving as QoE estimations) and QoEJ measures obtained through the above linear and exponential models. The optimal values (‘sweet spots’) are typeset in bold.

Table 3.4 QoE estimations and QoEJ measures for different resolutions

Resolution R	MOS(R)	$QoEJ_{\text{lin}}(R)$	$QoEJ_{\text{exp}}(R)$
240p	4.05	**0.033/J**	**0.024/J**
360p	**4.18**	0.026/J	0.022/J
480p	3.48	0.017/J	0.015/J
720p	3.00	0.010/J	0.010/J

It may be a bit surprising to observe that the QoE 'sweet spot' is obtained at a resolution of 360p. Indeed, for 480p and 720p, increasing the amounts of jerkiness and freezes were disturbing the users. The latter disturbances are due to heavy traffic conditions on the university Eduroam WiFi network that is shared by many students—a rather typical case for a quasi-public wireless network. As soon as disturbances due to resource overload show up, the QoE gets affected quite significantly.

However, taking into account the rise of the end-to-end energy with growing resolution, the QoEJ 'sweet spot' is now found at 240p, i.e., the higher resolutions (360p and onwards) lead to a suboptimal quality–energy trade-off. These insights go hand in hand with the indications on end-to-end energy savings by reducing the resource consumption (through reduced resolution, SNR or frame rate). Moreover, QoEJ decreases even faster as network-induced disturbances appear for 480p and 720p, which means that conditions of resource over-utilization imply suboptimal quality-energy trade-offs.

3.14 Conclusions

The book chapter has presented an overview of some of the most important models for video distribution networks. Different models are presented, which are about video coding and compression, video streaming, transcoding, video traffic models, energy consumption models, elements of system performance, performance optimization concepts as well as QoE- and energy-optimal streaming. Associated with this, the concept of sustainable throughput is introduced as a measure for QoE-optimal video streaming. A practical example for QoE- and energy-optimal streaming is provided as well.

Acknowledgements This research was partially supported by the European Celtic-Plus project CONVINcE and funded by Finland, France, Sweden and Turkey.

References

1. CISCO (2012) IP/MPLS networks: optimize video transport for broadcasters. CISCO
2. Monnier R, Popescu A, Ljung R (2016) CONVINcE: towards power-optimized video distribution networks. In: 19th International ICIN conference, Paris, France
3. Bing B (2010) 3D and HD broadband video networking. Artech House, USA

4. Ohm JR. Sullivan GJ, TK Tan (2012) Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC). IEEE Trans Circuits Syst Video Technol 22(12):1669–1684
5. VCODEX (2016) Historical timeline of video coding standards and formats. https://www.vcodex.com/historical-timeline-of-video-coding-standards-and-formats
6. CISCO (2017) Cisco visual networking index: VNI Forecast 2021. https://www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/index.html?dtid=osscdc000283
7. Zambelli A (2009) IIS smooth streaming technical overview, Microsoft Corporation, 2009
8. Sodagar I (2011) The MPEG-DASH Standard for multimedia streaming over the internet. IEEE Multim (18):62–67
9. Storage B (2016) Bandwidth and storage. http://www.broadbandchoices.co.uk/guides/internet/watching-tv-online, 2016
10. Rao A, Lim Y, Barakat K, Legout A, Towsley D, Dabbous W (2011) Network characteristics of video streaming traffic. ACM SIGCOMM
11. Gelman AD, Halfin S (1991) On buffer requirements for store-and-forward video on demand service circuits. IEEE GLOBECOM
12. Call for Evidence on Transcoding for Network Distributed Video Coding, Press release from MPEG 119—Torino. http://mpeg.chiariglione.org/
13. Tanwir S, Perros H (2013) A survey of VBR video traffic models. IEEE Commun Surv Tutor 15(4)
14. Tanwir S, Perros H (2014) VBR video traffic models. Wiley
15. Popescu A (2008) Traffic analysis and control in computer communications networks. Blekinge Institute of Technology, Sweden
16. Cox DR (1984) Long-range dependence: a review. Iowa State University Press, Iowa, USA
17. Tanwir S, Perros H (2013) A survey of VBR video traffic models. IEEE Commun Surv Tutor 15 (4)
18. Alheraish AA (2004) Autoregressive video conference models. Int J Netw Manage 14 (5):329–337
19. Popescu A, e. a. (2016) CONVINcE D1.1.4 Theoretical models, report project CONVINcE, Celtic-Plus 2016. https://www.celticplus.eu/project-convince/
20. COMBO (2014) Monitoring parameters relation to QoS/QoE and KPIs
21. Soldani D (2010) Bridging QoE and QoS for mobile broadband networks. ETSI document. http://www.etsi.org/WebSite/NewsandEvents/QoSQoEUserExperience.aspx
22. Group, Q. O. (2006) Inter-Provider Quality of Service, Quality of Service Working Group
23. RFC, 4, (2006) IETF—Internet Engineering Task Force, BGP/MPLS IP Virtual Private Networks (VPNs). http://www.rfc-editor.org
24. Anjum B, Perros H (2015) Bandwidth allocation for video under quality of service constraints. ISTE Ltd and John Wiley & Sons. ISBN 978-1-84821-746-1
25. Heindl A (2001) Decomposition of general tandem queueing networks with MMPP input. Perform Eval 44(4):5–23
26. ETSI-2011 (2011) Environment Engineering (EE) Energy Efficiency of Wireless Access Network Equipment, ETSI
27. Padhye J (1998) Modeling TCP throughput: a simple model and its empirical validation. In: ACM SIGCOMM 1998, New York, USA
28. Ravindra K, Ahuja TL (1993) Network flows: theory, algorithms and applications. Prentice-Hall, USA
29. Constantine B (2011, August) Framework for TCP throughput testing. Internet Engineering Task Force (IETF) Request for Comments (RFC) 6349, IETF
30. Qualinet White Paper on Definitions of Quality of Experience. European Network on Quality of Experience in Multimedia Systems and Services (COST Action IC 1003). In: Le Callet P, Möller S, Perkis A (eds) Lausanne, Version 1.2, Novi Sad, March 2013
31. Fiedler M, Popescu A, Yao Y (2016) QoE-aware sustainable throughput for energy-efficient video streaming. In: IEEE SustainCom conference, Atlanta, Georgia, October 2016
32. DASH Industry Forum. http://dashif.org/about/

Chapter 4
Energy-Aware Software Video Encoding in Head-End

Mikko Uitto

4.1 Introduction

The era of high-speed mobile and fixed network connections has increased especially the video traffic on the Internet and it is expected to continue growing in future. Internet video streaming and downloads will concern over 80% of all the consumer traffic by 2021 [1]. Furthermore, Internet video viewed from PC screen or TV is also expected to form a larger share from the total video traffic by growing its amount from the current over 30% [1]. This means that needed energy and power for online video will also increase requiring greener solutions not only to save mobile battery charge but also to reduce carbon footprint.

As the smartphones are one's best friends today suitable for communication, as well as mobile pay, it is essential to save battery every time it is possible. Naturally, heavy video processing can consume battery notably, which means that its optimisation can lead to best results when resolving ways for increasing the battery life especially for those who use their mobiles in every fallow period, possibly, for watching online videos, news, etc. Although this chapter focuses on investigating the head-end video encoders, it is essential to know that the choices in head-end affect to terminals.

Video encoding/transcoding can be extremely heavy and time-consuming task for the head-end server and still quite relevant in order to minimise the transmitted bits. Traditionally, the input video is encoded only once and streamed point-to-point/multipoint suffering from transmission errors caused by a bad network connection. One of today's phenomena on the Internet favours for adaptive HTTP streaming, where the input feed is encoded into multiple representations in order to allow client adaptation against bandwidth fluctuations and therefore, ensuring improved Quality of Experience (QoE). However, this leads to producing

M. Uitto (✉)
VTT Technical Research Centre of Finland, 90570 Oulu, Finland
e-mail: mikko.uitto@vtt.fi

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0_4

several versions of the same video suitable for different bandwidths, display resolutions and stream characteristics that are dedicated to specific devices. Naturally, this can result in dozens of encoded representations of the same video and therefore, causing expenses to the service providers. On the other hand, social video streaming has become very popular, in which the smartphone can operate as the service provider unloading the battery charge. Therefore, energy-aware encoding methods for the video head-end are in key position.

In this chapter, selected software encoders and formats are evaluated in our testbed for finding ways to reduce head-end energy/power consumption in conjunction with obtained video quality and bitrate. First, used hardware and codecs are compared. Second, encoding algorithms in terms of complexity and encoding speed are evaluated. Third, the effect of video output resolution, frame rate and end quality are investigated in proportion to energy usage. In addition to the evaluation, the overall description of video transmission, nowadays, is depicted focusing especially on one of the hot topics, adaptive HTTP streaming. In the end, the conclusion is drawn.

4.2 Modern Video Transmission

Technology today enables high-speed video streaming to different client terminals that are capable of processing the stream in real-time and displaying it in high-resolution, even ultra-high-definition (UHD) displays. For example, one frame of 4 K (3840 × 2160) resolution raw (YUV 4:2:0) video would require enormous 12.4 MB size, which means that 1 s of video with 25 frames per second (fps) requires 25*12.4 = 310 MB. Therefore, video compression plays an important role as the starting point in the media transmission path. In video coding, especially term compression efficiency plays an important role, which means how bitrate and quality are in relation between each other.

Figure 4.1 presents a timeline of video coding standards. The H.264 (H.264/AVC), HEVC (H.265) and VP9 are the ones under scope in this work. Recently, Alliance for Open Media (AOMedia) announced a new royalty-free coding format AV1 that basically follows the outline of VP10 and the work after VP9. It is predicted to compete with HEVC in compression efficiency. AOMedia is founded by the big companies Amazon, Cisco, Google, Intel, Microsoft, Mozilla and Netflix [2].

4.2.1 Media Preparation: Video Compression

As was presented, video compression has evolved and needs to evolve in future for adjusting the compression efficiency towards future extremely high-resolution media streaming. The encoder plays an essential role in creating energy savings in

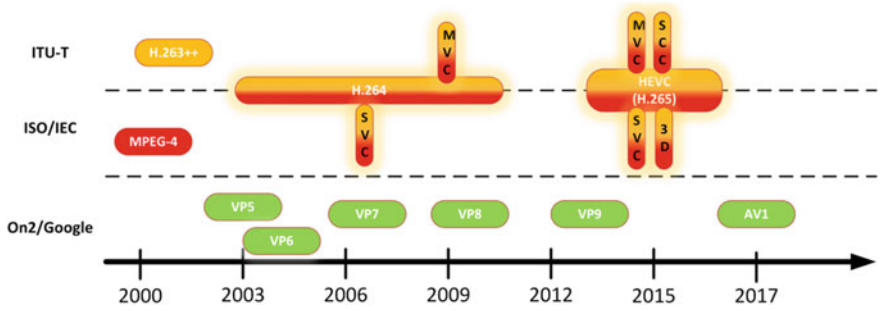


Fig. 4.1 Timeline of video coding standards

the head-end, especially if the encoder should produce multiple output streams for a variety of client terminals and of course as fast as possible.

The current leader and widely applied video compression standard is H.264 or MPEG-4 Part 10, Advanced Video Coding (often also named as H.264/AVC) [3]. It was founded after fusion of ITU-T Video Coding and ISO/IEC Moving Pictures Experts Groups rapidly superseding its successor MPEG-2 with up to 50% bit rate reduction. Currently, H.264 is still widely used in terrestrial, cable, satellite and IPTV broadcasts. In addition, it is also utilised in Blu-ray discs, videoconferencing and mobile video streaming. However, the main shortcomings in H.264 standard is in its limited support, up to 4K resolution and 60 fps and naturally the compression efficiency is not enough to tackle current (4K) and beyond (8K) resolution requirements. Today's mobile phones can already support and process 4K video.

Some of the key features in H.264 include hybrid spatio-temporal prediction, flexible partition of macroblocks (MB) and sub-macroblocks, context-adaptive binary arithmetic coding (CABAC) and context-adaptive variable-length coding (CAVLC) [3]. The standard holds over 20 approved profiles with 17 levels [4]. In addition, the standard also holds Multiview Video Coding (MVC) and Scalable Video Coding (SVC) amendments that focus on enhancing the 3D video delivery and videoconferencing via scalable representations.

As the compression efficiency of H.264 was insufficient for increased resolution, High-Efficiency Video Coding (HEVC) standard [5] was derived offering 50% bit rate reduction against same quality as in H.264. As a result of this, UHD content can be compressed with decent bit rate up to 8K resolution and amazing 300 fps. One of the massive improvements in HEVC was its ability to use parallel processing architectures [6], which allow fast and efficient frame processing by utilising the processor cores and threads.

HEVC uses enhanced spatio-temporal prediction where MBs are replaced by so-called Coding Tree Units (CTUs) and Coding Tree Blocks (CTBs) that are designed to be larger than traditional (H.264) 8×8 or 16×16 macroblock sizes. The standard comprises 15 recommended profiles with 13 levels including MVC and SVC amendments (MV-HEVC and SHVC) [5]. One of the interesting divergences compared to H.264 scalable amendments is that HEVC base layer is not

restricted to be HEVC compliant, which expands HEVC usage in several use-case scenarios. One of the new amendments includes Screen Content Coding (SCC), which enables improved compression efficiency for animation, graphics and text [7] targeting to real-time desktop sharing.

One of the HEVC exploitation issues is its licence. Therefore, Google has also generated its own royalty-free video codec named as VP9, which aims at improved compression efficiency targeting to the same level with HEVC. VP9 is designed especially suitable for web-based streaming on top of HTML5. As HEVC brought CTUs for replacing MBs, VP9 uses so-called superblocks [8].

4.2.2 Media Transmission: Video Streaming

Many of the current video streaming applications, such as YouTube and Netflix, use HTTP-based streaming, idea is to cut the encoded video into small, usually 1–10 second-long segments of which order is media presentation description (MPD). As traditional video streaming services utilised point-to-point streaming using UDP/RTP/RTSP protocols initialized by the streaming server, HTTP-based streaming is initialised by the client on top of TCP. Naturally, already TCP protocol causes a delay from the guaranteed packet transmission, which means it is not extremely suitable for real-time video conferencing. However, the latency can be optimised by using, for example, smart client buffering.

The advantages of HTTP-based streaming include flawless video delivery (TCP), non-firewall and NAT-free access by clients, no dedicated streaming servers and maybe the most important issue, adaptation possibility against network fluctuations. Especially the adaptation in modern mobile networks is an important issue since operators cannot guarantee the full-speed network coverage everywhere caused by congestion or weak signal power. Traditionally, this situation would lead to low video quality, transmission errors and stalls, because the streaming server is sending the data without the knowledge of client capabilities. However, adaptive HTTP streaming allows client-side adaptation depending on the available network bandwidth, which can improve QoE significantly.

Adaptation possibility generally means that multiple encoded versions aka representations of the same video are produced in the head-end with alternate e.g. bitrate or resolution. All the representations are depicted in the MPD. Usually, the service providers generate at least three representations, but there can be more for covering the majority of the client devices and enabling smooth adaptation to unpredictable network conditions. As it is known, even the target device display resolution can range from old VGA resolution into modern 4K resolution. This means that the production process concerning video encoding for all these representations can be very time-and energy-consuming tasks causing costs for service providers.

One of the favoured technologies in adaptive HTTP streaming is Dynamic Adaptive Streaming over HTTP (MPEG-DASH) [9]. It differs from its rivals

Apple's HTTP Live Streaming (HLS), Microsoft Smooth Streaming and Adobe's HTTP Dynamic Streaming (HDS) mostly from the structure of MPD and video segment types. Also, MPEG-DASH has currently better support for HEVC, a variety of client terminals and also for CDN type solutions supported by, for example, Akamai. Naturally, HLS is targeted primarily on Apple's devices.

In this work, videos are encoded into structure supporting MPEG-DASH for partial client-side energy measurements depicted in the later chapter of this book. The MPEG-DASH encapsulation process was measured also for the encoding side and determined to be less than 1 W when performing it to VoD type video sequences.

4.2.3 Energy Efficiency in Head-End

It is essential and important to identify energy consuming phases in video transmission towards green multimedia streaming. The head-end selections can also have an effect on the network part as well as terminals. For example, the high data rate for the video will increase the network-side load and require more terminal-side processing performance. In some situations, it is not wise to encode high-resolution streams if the clients don't have enough processing power or network bandwidth [10].

The earlier research has slightly investigated green media transmission. It is not well covered how head-end encoding can bring energy savings. Authors in [11, 12] resolve ways and focus on energy-efficient HTTP streaming, but neglect head-end and especially encoding energy consumption. Many of the recent studies discuss techniques that can reduce terminal power and energy consumption, especially on mobile devices [13–15].Video compression parameters for reducing energy are investigated in [16, 17]. It is confirmed that encoding phase is the most energy-hungry phase in the video transmission chain and optimising the encoding parameters can provide good energy savings.

This work focuses on resolving ways to reduce head-end energy consumption during software encoding when powerful PCs are used. Energy savings from software encoding is quite little-investigated issue so far and needs to be studied more. Thus, energy consumption and sufficiency in the world is an overarching issue.

4.3 Evaluation

In this chapter, evaluation cases and setup for energy consumption under software video encoding are described. Basically, we focus on five different aspects: compression efficiency via encoding presets, resolution, frame rate, objective quality/bitrate and physical encoding hardware. In this work, these presets are also referred

as profiles, which should not be mixed with the profiles introduced by the standard (e.g. Main profile, High profile). Furthermore, HW acceleration (GPU) is neglected and disabled in order to focus more deeply on pure software encoding. However, GPU usage would probably have a favourable energy consumption reduction in video encoding.

4.3.1 Evaluation Setup and Tools

The evaluation setup for measuring the power consumption of software video encoding holds basically two primary elements that are visualised in Fig. 4.2. The first component includes the actual encoding machine that is responsible for taking raw (YUV) video feed as an input and producing wanted output format. Second, an energy/power measurement device was connected to the encoding machine providing reliable measurements of the consumption.

The video test sequences are widely known clips from open-source material '*Tears of Steel*' and '*Big Buck Bunny*'. The extraction points and encoding parameters can be found from Table 4.1. The first test sequence contains only a little motion and should be, therefore, considered as easy for encoding. The second clip is opposite to the first, containing lots of motion and action resulting in difficult complexity. The last test sequence contains animation with medium difficulty and colourful content.

The structure of encoded sequences was aligned so that the outcome would be MPEG-DASH compliant when using one-second segment size. Therefore, the intra-decoding refresh (IDR) period was determined to 24 when using frame rate 24 fps, which was used in the majority of the evaluation runs. This enabled proper MPEG-DASH segmentation, whose effect was mainly investigated only in the terminal side presented in the later chapter of this book. The VoD style MPEG-DASH encapsulation in the head-end had insignificant energy consumption since the 60 s encapsulation process took less than 0.5 s for the encoded sequences. The variable bitrate was applied during the encoding because it provides slightly better end quality versus file size and no padding is needed for slow-motion scenes.

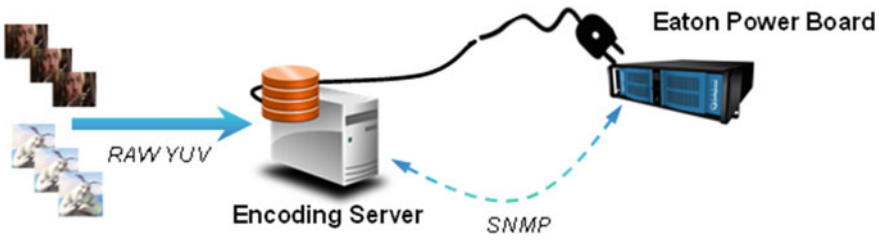


Fig. 4.2 Evaluation setup

Table 4.1 Encoding parameters

Parameter	Value
Sequences	Tears of Steel Big Buck Bunny
Extraction points for sequences	TOS1: 103 s TOS2: 448 s BBB: 0 s
Sequences length	60 s
IDR period	24
GOP length and type	8, closed-GOP
Bitrate type	Variable
Objective target quality PSNR-Y (global average)	TOS1: 43,7 dB TOS2: 42,9 dB BBB: 43,7 dB
Codecs and their versions	×264: 0.148x ×265: 1.9 Kvazaar: 0.8.2 Vpxenc: 1.5.0

Primarily Intel Core i7-5820 K@ 3.3 GHz desktop PC (2014) was utilised for the video encoding. The machine is a powerful 64-bit desktop PC running with Ubuntu 14.04 LTS and 32 GB internal memory. The idea for the evaluation is to measure the total energy consumption of the machine without splitting it into individual processes. As software encoding is the key interest, no GPU or HW acceleration was utilised. Also, restricted lab environment prevented possible incoming/outgoing network connections for influencing the energy consumption. In addition, no display was attached to the machine. The idle consumption over a long-term average was concluded to be 50 W.

Furthermore, Intel PowerEdge 510 rack server (2009) with two Intel Xeon E5606 @ 2.13 GHz processors and 16 GB internal memory were utilised for comparing the effect of older hardware to the newer hardware. The software part for the encoding and power measurement was identical as in the newer machine. The idle power consumption for this machine was also 50 W.

4.3.2 Energy Measurement

In this work, the total energy consumption was measured instead of, for example, the power consumption of the software encoding process. This originated from the issue where the 60 s test sequences were not possible to encode in real-time, which would cause distorted results. For example, power consumption for encoding the 60 s sequence in real-time could be 180 W following exactly the same value with increased encoding time. Therefore, total energy consumption for the encoding process was measured without excluding the idle power consumption which was

measured to be approximately 50 W. In addition, as was stated earlier, the total machine consumption was measured instead of evaluating only the encoding software performance. This originated mainly from using a reliable and designated power consumption tool that is connected between the wall socket and encoding machine.

The actual power measurement device in use was Eaton Managed ePDU device. This device can collect several values from the target machine related to matters such as voltage, CPU temp and power. It uses Simple Network Management Protocol (SNMP) for providing the ePDU device information to the requesting client that then parses the wanted information from the return message. In this work, *outletActivePower* (P_{out}) was the key interest. It was measured that single SNMP call takes less than 1 W and 100–200 ms to operate, whose influence was then neglected from the final measurements. The SNMP requests in the tests were performed at 1 s intervals.

From the collected P_{out} values, the average power consumption for n measured points is gained by

$$P_{\text{avg}} = \frac{1}{n}\sum_{t=0}^{n} P_{\text{out}}(\text{i}) \tag{4.1}$$

Furthermore, total energy consumption E is derived from average power consumption by

$$E = P_{\text{avg}} * t = P_{\text{avg}} * (t_{\text{end}} - t_{\text{start}}), \tag{4.2}$$

where t_{end} and t_{start} represent the encoding start and end times, respectively.

4.3.3 Evaluated Codecs

One of the earlier phases of this work consisted of selecting proper open-source codecs to be evaluated. The main evaluative work has been done in creating a basis of this work [18]. The ×264 open-source library [19] producing H.264 compliant bitstream is one of the leading and most widely used codecs in the world, which means its selection as a baseline was a natural choice. It is, nowadays, utilised in many popular web services such as in Youtube. Its well-optimised behaviour enables creating multiple full HD (FHD) streams in single market-type PC. However, its usage in future high-resolution streaming is quite improbable since compression efficiency is not enough for compressing the streams into reasonable bitrate.

As introduced earlier, HEVC standard promises a 50% better compression efficiency than H.264 and is, therefore, a considerable alternative for next-generation coding format that can compress future 4 and 8 K resolution with a

decent bitrate. Therefore, we selected ×265 open-source library [20], which aims to deliver the world's fastest and most computationally efficient HEVC encoder. The ×265 HEVC encoder was judged as the best overall HEVC encoder by the Moscow State University [21]. The second selected HEVC encoder was Kvazaar [22], which was mainly chosen because having the similar encoding presets that are also available in ×264 and ×265.

Finally, as HEVC standard is not totally royalty-free, Google's rival VP9 was selected as a part of the base evaluation. The selected open-source codec producing VP9 stream compressed into WebM format was Vpxenc [23].

4.3.4 Use Cases

This section presents the evaluation use cases from which the first two select the equipment used in the evaluation. The latter four cases comprehend the optional parameters that can be used during the encoding.

(a) ENCODERS
The first evaluation case forms basically a baseline and selection for the further evaluated encoders. In this evaluation, four encoders are compared: ×264, ×265, Vpxenc and Kvazaar.

(b) HARDWARE
The second evaluation case compares two encoding PCs in order to see the effect of more novel hardware in relation to the encoding process.

(c) COMPRESSION EFFICIENCY (PROFILES)
The third evaluation case concerns compression efficiency that is pre-handled via different presets available in ×264 and ×265 encoders, namely as *ultrafast, superfast, veryfast, faster, fast, medium, slow, slower, veryslow* and *placebo* [24, 25]. Basically, these encoding profiles affect the encoding speed and compression efficiency; ultrafast provides the fastest software encoding, with the highest bitrate while as a placebo is the slowest providing the lowest bitrate when targeting to identical average quality.

(d) RESOLUTION
The fourth evaluation case compares how video target resolution affects to energy consumption.

(e) FRAME RATE
The fourth evaluation case resolves how video frame rate affects to encoding energy consumption.

(f) QUALITY
The fifth evaluation case shows how objective video quality target influences on energy consumption. Basically, this has an effect also on video bitrate when using the same encoding presets.

4.3.5 *Quality Assessment*

The evaluation use cases of this work contained lot of runs and comparisons between different encoding algorithms, parameters and codecs. Due to this, average objective quality was set similar (inside the margin of 0.2 dB) in order to provide a fair comparison. For example, when using different encoding presets, average objective quality was not identical against H.264 and HEVC, which could favour the results in one direction or another. This adjustment was executed by setting the average target bitrate to the needed level.

Objective quality was measured using Peak Signal-to-Noise Ratio (PSNR), which is a widely used metric in video technology [26, 27]. It is not the best metric because it disregards the human visual perception and viewing conditions and therefore, does not correlate very well with the characteristics of Human Visual System (HVS). More sophisticated indicators such as Structural Similarity Index (SSIM) [28] and Video Quality Model (VQM) [29] were not utilised in this work, mainly because of their implemented complexity for measuring the outcome. Thus, PSNR indicator is quite lightweight allowing fast processing.

Usually, PSNR is referred to the calculation of luminance Y component because the human eye is more sensitive to changes in video sharpness than in saturation, namely as chrominance (C_r, C_b) [26]. The PSNR-Y formula is derived in the following way providing the result as decibels (dB):

$$PSNR = \frac{1}{T}\sum_{t=1}^{T} 10 * \log\left(\frac{L2}{MSE(t)}\right), \tag{4.3}$$

where L is the maximum value of a pixel and T represents frames per second in temporal time t. Furthermore, mean square error (MSE) is calculated as follows:

$$MSE(t) = \frac{1}{m * n}\sum_{t=1}^{m}\sum_{j=1}^{n}[I(i,j,t) - K(i,j,t)]^2, \tag{4.4}$$

where I is the luminance pixels of the original image, K is the reconstructed pixels in the compressed picture with picture size m x n.

4.4 Results

The following section presents the results according to the evaluation. First, the energy consumption of different open-source encoders is evaluated following the comparison between two encoding machines depicted earlier. Next, the extensive comparison between H.264 (×264) and HEVC (×265) is performed in order to

illustrate energy consumption in terms of compression efficiency and other encoding related parameters for these current and future video coding technologies.

4.4.1 *Effect of Different Codecs*

Figure 4.3 shows the comparison between different evaluated codecs when measuring the total energy consumption of the encoding process. These results are continuation for the real-time encoding measurements introduced in [18]. It is worthwhile mentioning that these results are completed by encoding the sequences real time, according to the frame rate, whereas, the current work uses as fast as possible encoding. As it is observed, ×264 (H.264) provides the lowest energy consumption from the evaluated encoders. Vpxenc (VP9) also performs well regarding the energy consumption and it is only slightly higher than ×264. HEVC encoders ×265 and Kvazaar are the most energy-hungry. However, ×265 provides the best compression efficiency. According to these results, ×264 and ×265 are selected for further evaluation as the aim is to resolve energy-saving techniques for decreasing HEVC head-end energy consumption. However, VP9 shows great potential for future evaluation although it lacks support for a variety of client terminals.

4.4.2 *Effect of Hardware*

Figures 4.4, 4.5 and 4.6 present energy consumption comparison between selected Intel PowerEdge Rack server and Intel Core i7, described above. The left picture in these figures is the result of H.264/AVC while right one presents the results of HEVC. The comparison setup utilised different target quality levels for the encoder. The detailed results are presented in Sect. 4.4.5. It is notable that energy consumption is strongly in relation to the source media, meaning that high-motion

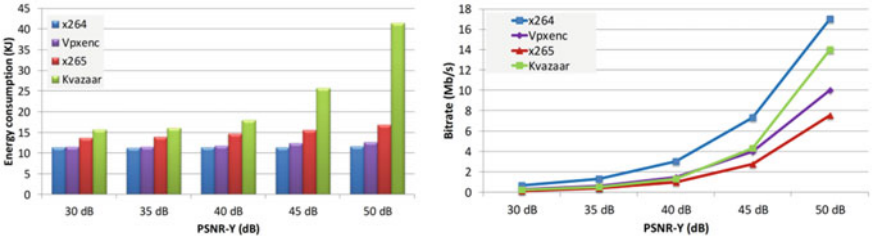


Fig. 4.3 Encoding energy consumption for different codecs against increasing objective quality [18]

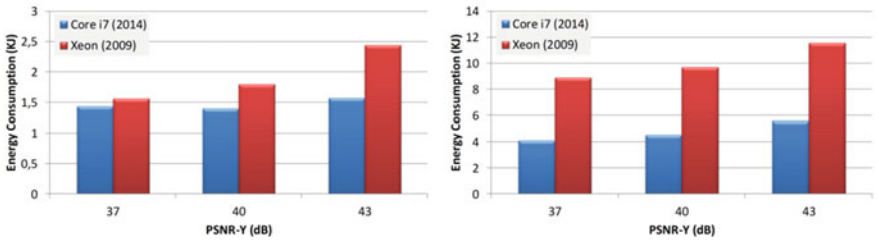


Fig. 4.4 Encoding energy consumption comparison for H.264/AVC (left) and HEVC (right) between Intel PowerEdge Rack server and Intel Core i7 desktop PC for the test sequence 1

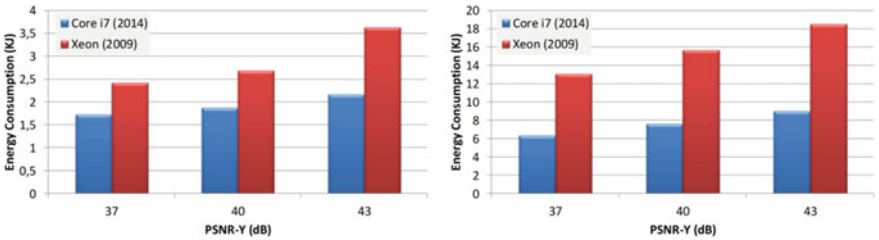


Fig. 4.5 Encoding energy consumption comparison for H.264/AVC (left) and HEVC (right) between Intel PowerEdge Rack server and Intel Core i7 desktop PC for the test sequence 2

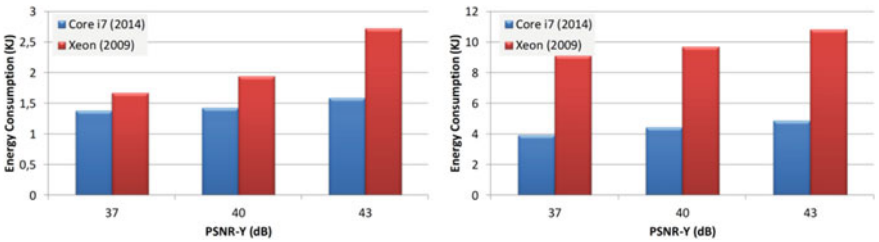


Fig. 4.6 Encoding energy consumption comparison for H.264/AVC (left) and HEVC (right) between Intel PowerEdge Rack server and Intel Core i7 desktop PC for the test sequence 3

clip (Sequence 2) requires more effort from the encoding machines. This probably originates from the difficulties in using predictive motion estimation between frames. It is also evident that newer Core i7 consumes less energy than PowerEdge server holding Intel Xeon processor. Due to this, Core i7 is selected as the encoding machine for further evaluation.

4.4.3 *Effect of Compression Efficiency*

The target average objective quality for the test sequences 1, 2 and 3 were set to 43.7, 42.9 and 43.7(dB) when testing the compression efficiency. The target values were derived from the outcome of *medium* ×264 preset when encoding the sequence without any bitrate restrictions.

Figures 4.7, 4.8 and 4.9 illustrate the software encoding energy consumption diagrams for the three test sequences when using different encoding profiles. Energy consumption for the second test sequence is slightly higher since high-motion clip requires more efforts from the encoder. Correspondingly, the bitrate curves for these sequences are also presented. As it can be observed, the fastest encoding profiles (*ultrafast*, *superfast*) have the lowest energy consumption regardless of the codec, but resulting also to highest output bitrate. It is also notable that encoding energy consumption is not directly in proportion to output video bitrate: Big Buck Bunny clip (Sequence 3) consumes less energy than Sequence 1 but ends up with a higher bitrate. Naturally, the average bitrate for the high-motion clip is also higher.

It is notable that *faster* and *fast* presets already provide basically quite a good compression efficiency for all the test sequences. It is also observed that bitrate saturates only little after this point when using more complex presets. The average energy consumption difference between ×264 and ×265 encoders is approximately

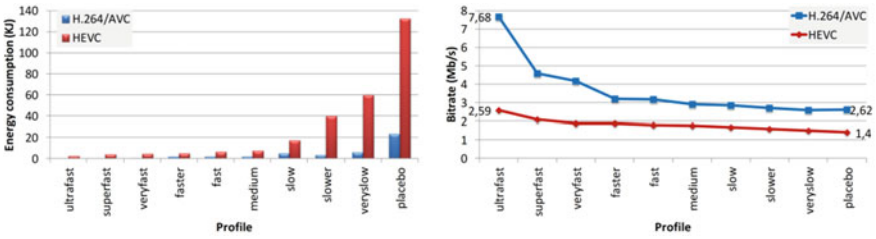


Fig. 4.7 Energy consumption and bitrate for test sequence 1 using different encoding profiles

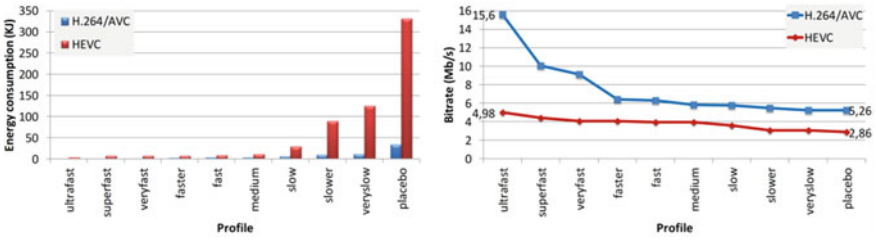


Fig. 4.8 Energy consumption and bitrate for test sequence 2 using different encoding profiles

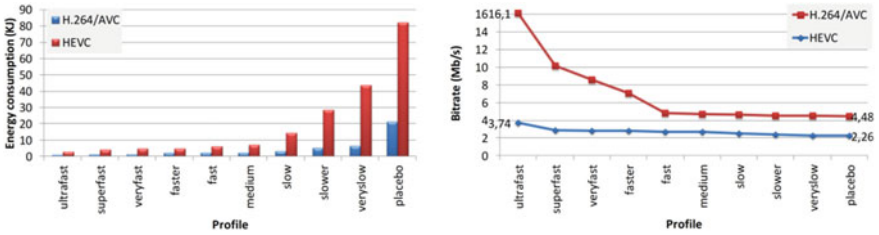


Fig. 4.9 Energy consumption and bitrate for test sequence 3 using different encoding profiles

3 kJ/3 × for the *faster* preset in favour of ×264. The encoding time for ×265 is approximately three times longer, which is in direct relation to energy usage.

Although ×265 and HEVC consumes energy 3 times more, compared to ×264 and H.264 it saves network bandwidth 40–50% and naturally consumes less energy in the rest of the transmission chain, such as in base stations. Furthermore, HEVC playback energy consumption can be even reduced compared to H.264. Although the utilised HEVC encoder is still in development stage without final optimisations, its usage is respectable especially when only few representations of the same video need to be produced.

As a conclusion of evaluating the encoding profiles, it can be seen that energy can be saved in head-end by using less complex encoding profiles. However, it will also lead to higher video bitrate, which can cause more costs for the rest of transmission.

4.4.4 Effect of Frame Rate and Resolution

Figures 4.10, 4.11 and 4.12 present the energy consumption curves when using varying frame rate or resolution for the three test sequences. The encoding profile was set to *veryfast* and target average objective quality for each sequence as shown in Table 4.1. It can be seen that encoding energy consumption increases significantly when increasing the video resolution. For example, energy consumption is nearly double when the resolution is raised from HD-ready (720p) into full HD (1080p), of which both are widely used resolutions in today's video streaming and playback devices. The same phenomenon can be seen also when doubling the frame rate from 15 to 30 fps when using 1080p resolution. Naturally, in both cases, less bits and bytes are needed for producing lower resolution and/or frame rate, which affects reductively to energy consumption. Thus, the reasonability of using lower frame rate should be well considered, since it creates a non-smooth effect to video playback.

Figure 4.13 illustrates the achieved bitrates for this evaluation case. As expected, the output bitrate grows quite linear when increasing resolution or frame rate. For

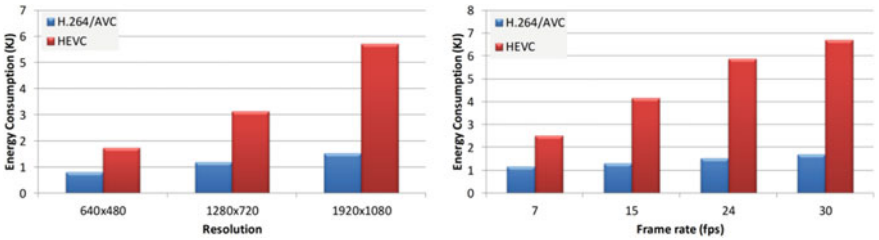


Fig. 4.10 Encoding energy consumption when using different resolution and frame rate for test sequence 1

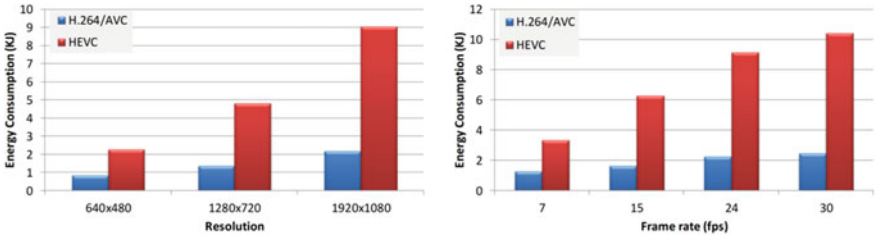


Fig. 4.11 Encoding energy consumption when using different resolution and frame rate for test sequence 2

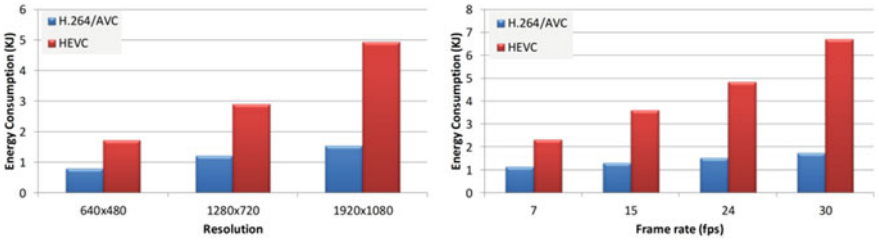


Fig. 4.12 Encoding energy consumption when using different resolution and frame rate for test sequence 3

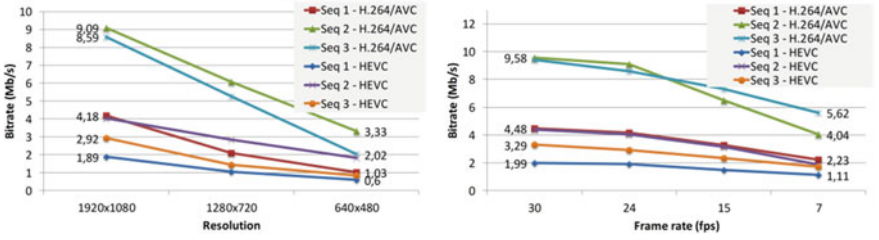


Fig. 4.13 Encoding bitrates when using different resolution and frame rate for test sequences

small display devices, it is wiser to stream with lower resolution, because small, sharp artefacts are harder to see in large resolution displays such as TVs.

4.4.5 *Effect of Quality*

Figures 4.14 and 4.15 illustrate encoding energy consumption curves against different objective quality. Similarly as earlier, increasing average quality with higher bitrate also leads to increased energy consumption. However, the improvement regarding energy savings is not that significant as in earlier cases. Still, bitrate reduction is several Mb/s, which can have further savings in the transmission chain.

4.4.6 *Summary*

This book chapter presented the results regarding the use cases for evaluating software-based video encoding. The results consisted of use cases related to the aspects of utilised hardware, encoders, encoding profiles, resolution, frame rate and quality. During the first steps, newer and more powerful encoding machine was

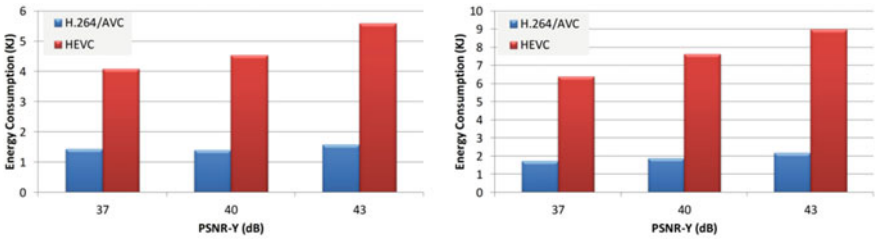


Fig. 4.14 Encoding energy consumption against different objective quality for test sequences 1 (left) and 2 (right)

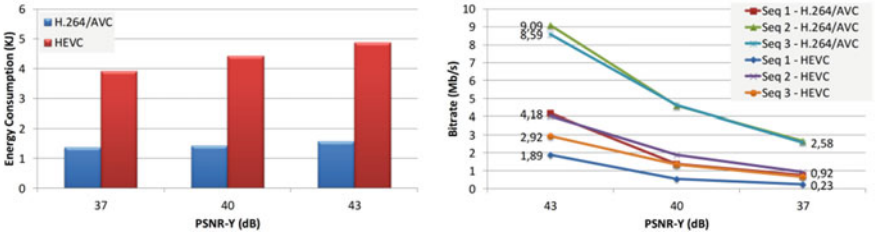


Fig. 4.15 Encoding energy consumption (sequence 3) and bitrates against different objective quality for the test sequences

selected running on ×264 and ×265 open-source encoders that were outperformed as the most efficient regarding

Energy consumption: ×264

Compression efficiency: ×265

After this, ×264 was selected as a baseline to be compared with ×265. The next step was to compare encoding performance between old Intel PowerEdge Rack server with Xeon processors and Intel Core i7. The results indicate that over 50% energy savings can be retrieved with the newer machine holding more processor cores. Also, ×265 encoding to HEVC uses 2–4 times more energy than ×264 encoding to H.264 but requires 50% less bandwidth.

The choice of encoding profile can have a significant effect on energy consumption. Most complex profiles provide only slightly better compression efficiency, but consume large amount of energy and have slow encoding speed. The best trade-off for the energy-efficient encoder is to use quite fast encoding preset that has small energy consumption and relatively satisfying bitrate versus quality.

The selection of encoding resolution and frame rate can bring energy savings. Approximately 50% savings are achievable for HEVC when dropping resolution from full HD to HD, which also decreases the video bitrate 50%. Furthermore, 25% savings are gained when dropping frame rate into half of its origin.

Finally, dropping the quality via bitrate can influence also positively to energy consumption. Approximately 20% can be saved when dropping HEVC quality 3 dB, which also decreases the video bitrate 100%.

4.5 Conclusion

This book chapter presented a preliminary investigation of energy consumption in software video coding focusing on current and future video encoding formats. The chapter provided an insight into how software-based video encoding uses energy and which elements have an effect on the consumption. The ultimate target was to see how energy consumption could be reduced in head-end, which can be considered as the production site of media delivery. A small comparison between different hardware was performed. Furthermore, a few open-source encoders were compared with each other from which one H.264 and HEVC encoder were selected. The further evaluation included how compression efficiency via different encoding profiles, resolution, frame rate and end quality/bitrate effect on energy consumption. Due to several runs with the three test sequences, it was concluded that all these elements have an influence on the head-end energy consumption and should be, therefore, taken into account when considering savings and ways for supporting a green ideology.

Acknowledgements This research was partially supported by the European Celtic-Plus project CONVINcE and partially funded by Finland, France, Sweden and Turkey. The author would like to thank for their support.

References

1. Cisco (2017) Cisco visual networking index: forecast and methodology 2016–2021. Technical report
2. Alliance for Open Media (2015) Alliance for Open Media established to deliver next-generation open media formats. Press release
3. Wiegand T, Sullivan GJ, Bjontegaard G, Luthra A (2003) Overview of the H.264/AVC video coding standard. IEEE Trans Circuits Syst Video Technol 13(7):560–576
4. Information Technology (2017) Advanced video coding for audio-visual services: recommendation ITU-T H.264 Std, H.260–H.279:2016
5. ITU-T (2017) Advanced video coding for audio-visual services: recommendation ITU-T H.265 Std. H.260–H.279:2016
6. Sullivan GJ, Ohm JR, Han WJ, Wiegand T (2012) Overview of the high efficiency video coding (HEVC) standard. IEEE Trans Circuits Syst Video Technol 22(12):1649–1668
7. Xu J, Joshi R, Cohen RA (2015) Overview of the emerging HEVC screen content coding extension. IEEE Trans Circuits Syst Video Technol 26(1):50–62
8. Mukherjee D, Bankoski J, Grange A, Han J (2013) The latest open-source video codec VP9—an overview and preliminary results. Paper presented at the picture coding symposium (PCS), San Jose, California, Dec 2013
9. Sodagar I (2011) The MPEG-DASH standard for multimedia streaming over the internet. IEEE Multimed 14(4):62–67
10. Uitto M, Heikkinen A (2016) SAND-assisted encoding control for energy-aware MPEG-DASH live streaming. Paper presented at the 2016 24th international conference on software, telecommunications and computer networks (SoftCOM), Split, Croatia, Sept 2016
11. Yunmin G, Kwon OC, Song H (2015) An energy-efficient http adaptive video streaming with networking cost constraint over heterogeneous wireless networks. IEEE Trans Multimed 19 (9):1646–1657
12. Khan S, Schroeder D, Essaili AE, Steinbach E (2014) Energy-efficient and QOE-driven adaptive HTTP streaming over LTE. Paper presented at the wireless communications and networking conference (WCNC), Istanbul, Turkey, Apr 2014
13. Trestian R, Moldovan A-N, Ormond O, Muntean G-M (2012) Energy consumption analysis of video streaming to android mobile devices. Paper presented at the network operations and management symposium (NOMS), Maui, Hawai, USA, Apr 2012
14. Hu W, Cao G (2015) Energy-aware video streaming on smartphones. Paper presented at the IEEE conference on computer communication (INFOCOM), Hong Kong, May 2015
15. Hoque MA, Siekkinen M, Nurminen JK (2015) Energy-efficient multimedia streaming to mobile devices—a survey. IEEE Commun Surv Tutor 16(9):579–597
16. Ahmad JJ, Khan HA, Khayam SA (2009) Energy efficient video compression for wireless sensor networks. Paper presented at the 43rd annual conference on information sciences and systems (CISS), Baltimore, Maryland, USA, Mar 2009
17. Sharrab YO, Sarhan NJ (2013) Aggregate power consumption modeling of live video streaming systems. Paper presented in 4th ACM multimedia systems conference (MMSys), Oslo, Norway, Feb 2013
18. Uitto M (2016) Energy consumption evaluation of H.264 and HEVC video encoders in high-resolution live streaming. Paper presented at the 2016 12th IEEE international conference on wireless and mobile computing, networking and communications (WiMob), New York, USA, Oct 2016
19. VideoLan (2016) x264 project. http://www.videolan.org/developers/x264.html
20. Multicoreware (2017) x265 HEVC encoder/H.265 video codec. http://x265.org/
21. MSU Graphics & Media Lab (Video Group) (2016) HEVC video codecs comparison
22. Ultra Video Group (2016) Kvazaar HEVC encoder. http://ultravideo.cs.tut.fi/#encoder
23. WebM (2016) VP9 video codec. http://www.webmproject.org/vp9/
24. Dipp S (2013) Encoding presets for x264. http://dev.beandog.org/x264_preset_reference.html

25. Multicoreware (2014) x265 documentation. https://x265.readthedocs.io/en/default/
26. Olsson S, Stroppiana M, Baina J (1997) Video objective methods for assessment of video quality: state of the art. IEEE Trans Broadcast 43(4):487–495
27. Chikkerur S, Sundaram V, Reisslein M, Karam LJ (2011) Objective video quality assessment methods: a classification, review and performance comparison. IEEE Trans Broadcast 57(2):165–182
28. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612
29. Pinson MH, Wolf S (2004) A new standardized method for objectively measuring video quality. IEEE Trans Broadcast 50(3):312–322

Chapter 5
Reducing Power Consumption in Mobile Terminals—Video Computing Perspective

Martti Forsell

5.1 Introduction

Watching videos from a variety of fixed and mobile terminals is one of the most popular applications of the current ICT era. However, both the terminals and the delivery systems behind them consume a considerable amount of energy if the global scale and forecast of the growth in this area are taken into account [1, 2]. High power consumption in video delivery and playback has a number of negative consequences. First, production of energy creates negative environmental effects, e.g., carbon dioxide emissions and pollution, while with more power-saving solutions it could be possible to eliminate at least some of them. Second, high consumption heats up and limits the available operating time of increasing popular battery-powered mobile terminals deteriorating the obtained user experience and mobility. Third, the power consumption sets a limit to performance and thus functionality that can be fitted to a silicon chip, logic board, and highly portable mobile terminal.

Known solutions to reduce the power consumption in video delivery include increasing the power efficiency of the components of used devices and altering the quality and/or user experience. Additional possibilities to trade power consumption between different parts of the video delivery chain can be found from *video computing* that takes care of compression, transportation, and playback of videos. Increasing the compression rate takes more computational power and thus energy in the headend where videos are produced and encoded as well as in the terminals where videos are decoded. In the delivery network, however, it decreases consumption by reducing traffic needed to transfer the videos. In terminals, the components having major effect to the total energy consumption include video processing machinery, decoder, playback software, and display. Although power

M. Forsell (✉)
Computing Platforms, VTT, Oulu, Finland
e-mail: Martti.Forsell@VTT.Fi

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0_5

saving at the component level is obvious, there is a need for better understanding how the minimum consumption without sacrificing mobility and quality could be achieved by applying right tradeoffs and combining them with an optimal stack of available power-saving techniques.

The contents of this chapter reflect the research done in VTT for the Celtic-Plus project CONVINcE that aimed at power consumption reduction in video delivery networks and was operational in the years 2014–2017.

5.1.1 Contribution

The focus of this chapter is on reducing the power consumption of mobile terminals as a part of video delivery system. For that a wide variety of video computing-related energy-saving techniques is presented and evaluated on a line of Apple laptop mobile terminals running publicly available video playback software. The tests are done with video clips representing different kinds of video content and the effect of terminal hardware, video coding, video quality, player software, execution-environment/parameters, and streaming to power consumption and data rate is measured. According to the measurements, the video computing induced power consumption, total consumption and data rate can be reduced with respect to the state-of-the-art situation in the beginning of the research by 83%, 65%, and 43%, respectively. Additional reductions can be achieved by decreasing the quality of videos, switching to smaller footprint devices, and replacing the current processors with more advanced ones.

5.1.2 Previous Results

Despite of the importance and popularity of the topic, there exists a relatively low number of studies focusing on the power consumption of current mobile terminals and none of them takes a wide systematic approach as this chapter.

Power consumption of mobile terminals is covered by works focusing mostly on smartphones: Balasubramanian et al. compare the energy consumption characteristics of 3G, GSM and WiFi and measure the fraction of energy consumed for data transfer versus overhead [3]. They also present a simple energy model to quantify the energy consumption but the focus is on smartphones only. Carroll et al. propose a power model of the Freerunner device and analyze the energy usage and battery lifetime under a number of usage patterns of the HTC Dream and Google Nexus One [4]. Choi models and analyzes the power and performance of smartphones with the help of waiting time and usage scenarios [5]. Chen et al. evaluate the power consumption of several applications on a series of Samsung smartphones and take a deep look into AMOLED's power consumption and its relative contributions for multimedia apps [6]. Li et al. proposes an analytical power consumption model for H.264/AVC video decoding using hardware accelerator on popular mobile platforms [7].

This work was at the foundation of a new ISO/IEC MPEG standard "Energy-Efficient Media Consumption (Green Metadata)" published in 2015 [8, 9]. This standard specifies Metadata:

- which helps the mobile device build a precise prediction of picture decoding complexity to better govern *Dynamic Voltage Frequency Scaling* (DVFS) in processors, and
- which helps the mobile device apply the appropriate scaling of pixels luminance and backlight for a significant power reduction without any flickering.

Video computing inspired works focus mostly on decoding and power consumption: Ostermann et al. introduce the H.264/AVC standard and related tools as well as analyze their behavior [10]. Sullivan et al. provide an overview of the technical features and characteristics of the H.265/HEVC standard including parallel computing techniques supported by it [11]. Viitanen et al. provides a complexity analysis of H.265/HEVC reference decoder (HM 3.1) and compares the results with H.264/AVC reference decoder (JM 18.0) [12]. Mesa et al. propose and evaluate a parallelization strategy for the H.265/HEVC video codec [13]. The strategy requires that each *large coding unit* (LCU) row constitutes an entropy slice. The LCU rows are processed in a wave front parallel fashion by several line decoder threads using a ring synchronization. Bossen et al. analyze the complexity of a H.265/HEVC decoder implementation, which appears not to be significantly more complex than a H.264/AVC decoder [14]. Han et al. propose a hybrid parallel decoding strategy for H.265/HEVC, combining task-level parallelism and data-level parallelism based on coding tree units [15]. Hamidouche et al. investigate a pipeline and parallel software architecture for the scalable H.265/HEVC decoder [16].

5.1.3 Organization of the Chapter

The rest of the chapter is organized so that in Sect. 5.2 we take a look at video computing hardware and functionality in a mobile terminal, introduce terminals used in our tests and discuss measuring power consumption, in Sect. 5.3 we describe power reduction methods, evaluate them on our test terminals, and finally in Sect. 5.4 we give our conclusions.

5.2 Video Computing and Power Measuring in Mobile Terminals

Video computing is used in various points of video delivery to realize required functionalities. In the following, we introduce the machinery needed for video computing, standard video computing functionality, and methods for measuring the power consumption in mobile terminals.

5.2.1 Video Computing Machinery

A modern mobile laptop terminal is a compact, light-weight, highly efficient computing device with a very high definition display, integral keyboard, and versatile wired/wireless connectivity mechanisms. Video computing is handled by a processor connected to the memory and I/O devices.

A processor chip contains typically a *central processing unit* (CPU) and *graphics processing unit* (GPU). The CPU is aimed for general purpose processing and GPU for graphics-related duties. High-end laptops typically feature also an external GPU since the integrated ones have limited performance. Modern CPUs and GPUs feature multiple processor cores to speed up execution that are connected together and to the shared memory system via a communication network. Individual cores are accelerated by using pipelining, out-of-order execution, branch prediction, various speculations, multithreading, and *single instruction stream multiple data stream* (SIMD) units [17]. A typical memory system consists of small fast associative memories, caches, coupled with the processor cores and a hierarchy of larger but slower memories [17].

To validate the effect of the power-saving techniques in modern real-life terminals, we selected a line of popular Apple laptop mobile terminals including MacBook Pro 15" (mid-2012), MacBook Air 13" (early 2014) and MacBook Pro 15" (late 2016) for our tests (see Table 5.1).

5.2.2 Video Computing Functionality

The main video computing functionalities in mobile terminals are *video decoding*, decompressing the coded video streamed from the video delivery network or fetched from the local storage and *playing back* the obtained decompressed video on the display of the terminal.

Video encoding is used to compress a huge amount of data required to present high-resolution videos. E.g., a 4K video in 24-bit RGB format with a premium 60 Hz frame rate takes almost $3840 \times 2160 \times 3 \times 60 = 1.49$ GB/s. That makes 10.7 TB for a full 2-hour movie. While the best displays and internal memory systems are designed to process this amount of data per time unit, efficient compression of video data is crucially needed for delivery in the network and saving limited storage in terminals and headend devices to save resources and energy. The high-level idea is to compress the video content at the head end, move the compressed data via the network to the terminals, and decompress/expand it there for satisfactory playback quality. The most popular coding is nowadays *advanced video coding* (H.264/AVC) format [10], which is also considered as the baseline format in the beginning of the research. Current state-of-the-art coding format is *high efficiency video coding* (H.265/HEVC), which provides more precise/flexible control of coding, better possibilities for parallel implementation of the encoder and

Table 5.1 Terminals used in the tests. TDP = *thermal design* point reflecting the typical maximum power consumption in sustained use cases, Acc = Hardware decoder acceleration

Terminal	CPU/GPU/external GPU	Display/ TDP/Acc.	Role
Apple MacBook Pro 15" RETINA (mid 2012)	22 nm 4-core Intel Core i7 2.6 GHz 128-bit AVX SIMD units Intel HD Graphics 4000 Nvidia GeForce GT 650 M	2880 × 1800 45 W AVC	Baseline terminal
Apple MacBook Air 13" (early 2014)	22 nm 2-core Intel Core i7 1.7 GHz 128-bit AVX SIMD units Intel HD Graphics 5000 (no external GPU)	1440 × 900 15 W AVC	Low-power, small-footprint alternative
Apple MacBook Pro 15 (late 2016)	14 nm 4-core Intel Core i7 2.7 GHz 256-bit AVX-2 SIMD units Intel HD Graphics 530 Radeon Pro 460	2880 × 1800 45 W AVC, HEVC	New state-of-the-art terminal

decoder functionalities, and considerably higher compression rate with the same video quality as in H.264/AVC [11]. As a side effect of compression, extra energy is used in terminals and in the headend while significant amount of energy is saved in the network that we estimate to be by far the largest contributor to energy consumption in end-to-end video delivery. Video coding-related computations are typically fixed enough for dedicated logic-based accelerators that provide better performance/power and performance/area factors but their availability is limited. Since the Sandy Bridge microarchitecture released in 2011 Intel CPUs have contained video decoding accelerator for H.264/AVC and since the Skylake microarchitecture released in 2015 also for H.265/HEVC (see Table 5.1). Taking advantage of the accelerators requires player and driver software designed for acceleration. However, our focus is on software decoding.

The playback software receives the decompressed video from the decoder and shows it on the display. Besides just displaying video, players contain a lot of playback logic, video scaling, subtitle, audiorelated functionalities that contribute to the power consumption of terminals. For decoding and playback in our tests we use the following software:

- **DivX Player**. A popular video playback software from Neulion. Features 3 DivX codecs: the original MPEG-4 Part 2 DivX, the H.264/MPEG-4 AVC DivX Plus HD and the High Efficiency Video Coding DivX HEVC Ultra HD. DivX 10.7.2 was used as the baseline player software.
- **VLC Player**. A popular open source cross-platform multimedia player and framework that plays most multimedia files, DVDs, audio CDs, VCDs, and

various streaming protocols. VLC is provided by the VideoLAN organization. We used prerelease version 3.0 in some of our tests as an alternative to DivX.
- **openHEVC**. A fork of Libav (set of cross-platform tools and libraries to convert, manipulate and stream many multimedia formats and protocols.) with only the files needed to decode H.265/HEVC content. While it can also display videos, we used it for studying the effect of different parallel executions to power consumption and providing adjustable decoding to GPAC Osmo.
- **GPAC Osmo**. A parametric, open source, cross-platform video playback software that plays a wide variety of video files and streams. It was considered as the improved player for the research. We used version 4.0 in our tests.

5.2.3 Methods for Measuring the Power Consumption in Terminals

Measuring the power consumption of a terminal can be done with power meters directly from the power source. In mobile terminals, this is difficult since they are operated with batteries, difficult to open and there is very little space inside them. This applies to especially measuring the video computing power that should be measured directly from the processor attached to the main board with hundreds of miniature pins. While there are solutions for doing that, for the tested terminals we selected a software approach relying on the internal power measurement of the processor chip installed by the manufacturer and battery level inspection:

- **Intel Power Gadget**. A software-based power usage monitoring tool for Intel processors. Includes an application, driver, and libraries to monitor and estimate real-time processor package power information in watts for both the processor cores (IA measurement) and the whole processor chip including on-chip GPU using the energy counters in the processor. We used version 3.0.1 with default 10 Hz frequency and 0.005 W resolution.
- **FIPLAB Battery Health**. A software tool for gaining information on Apple MacBook batteries, e.g., current charge level, battery capacity, power usage, temperature, charge cycles. We used version 4.9 to determine the total power consumption although the accuracy weaker than that of Intel Power Gadget.

In order to evaluate the effect of using more advanced parallel processor architectures, we estimated the performance, silicon area, and power consumption of a processor chip needed to handle the video computing functionality with detailed analytical modeling of the architecture and state-of-the-art silicon technology using the methodology described by Pamunuwa et al. [18] and Forsell [19] since there are no silicon implementations available.

5.3 Methodology for Reducing the Power Consumption and Measured Improvements

This section describes a spectrum of methods for reducing the power consumption of video playback in mobile terminals and presents the results of our evaluations to show how much the power consumption could be reduced by using them in practice. In order to be able to quantify the improvements and compare the results with each other, the baseline configuration reflecting the situation in the beginning of this research is defined. To determine the overall cumulative reduction of the methods applied together (or minimal power consumption), a number of techniques are stacked on a top of each other. The ultimate aim is to find the best combination of techniques, tools, software, and hardware to show the full potential for power reduction in mobile terminals.

5.3.1 Measurements and the Baseline

The power consumption measurements are made with the line of Apple laptop computers introduced in Sect. 5.2.1 using the tools and methods described in Sect. 5.2.3. To test the power reduction techniques with different kinds of video content, we extracted three 60s video clips from the *Tears of Steel* short movie. The clips reflect typical slow motion, action, and end text/animation sequences and use the full HD 1080p 24 fps format (see Table 5.2).

For the tests, we encoded the original uncompressed video clips with 10 H.264/AVC and 10 H.265/HEVC profiles (that are in increasing efficiency order ultrafast, superfast, very fast, faster, fast, medium, slow, slower, very slow, and placebo) featured by ×264 and ×265 encoders. In order to obtain repeatable and fair video clips for measurements, we carefully set the average PSNR of H.264/AVC and H.265/HEVC counterparts to the same value.

Since the emphasis of this chapter is on video computing, we focus on the power consumption of the processor of the terminal rather than the total power consumption in most of our measurements. To identify potential for additional power saving with the end-to-end approach from the headend—where videos are produced

Table 5.2 Video clips used in systematic measurements

Name of clip	Short name	Type	Source video	Extraction position/duration
Slow motion clip	Seq1	1080p 24 fps	Tears of Steel	103 s/60 s
Action clip	Seq2	1080p 24 fps	Tears of Steel	448 s/60 s
End text/animation clip	Seq3	1080p 24 fps	Tears of Steel	650 s/60 s

Table 5.3 The average processor-only and total power consumption of the state-of-the-art Apple laptop in H.264/AVC HD video playback as recorded in the beginning of the research (September 2014) with DivX player using software decoder. We use these as the baseline for the evaluations

Terminal	Processor power consumption (AVC SW decoder)	Total power consumption (AVC SW decoder)
Apple MacBook Pro 15" 2012	19.8 W	27.0 W

and encoded—via the distribution network to the terminals, we measure the effects of different coding formats to the required data rate.

The state of the art of mobile terminals in the beginning of the research (September 2014) within the Apple product line was Apple MacBook Pro 15" RETINA laptop computer (mid-2012). The table 5.3 shows our measurements on the total power consumption and processor-only power consumption of them in video playback with a popular DivX player employing H.264/AVC software decoding. We will use these results as the *baseline* of the evaluation.

We are aware that these power consumption figures are higher than those that Apple announces in its technical specifications of MacBook Pro due to not using hardware accelerators but these results reflect a likely real-life situation.

In the following subsections, we study the effect of terminal hardware, video encoding, video quality, player software, execution environment-related issues/parameters, and streaming to the power consumption and datarate where applicable. The two final subsections are dedicated to overall reduction and estimating possibilities to further improve the processor hardware.

5.3.2 *Effect of Terminal Hardware*

The first and most obvious method for saving power comes from the evolution of the computer hardware. Using the latest computers employing latest processor technology is expected to make a big contribution to power saving. The main driver for this is the progress of semiconductor technology enabling smaller, faster, and less power hungry processor, and development of more power-savvy input/output devices and storage of data. The power saving of new processors come from smaller feature size involving less electrons to fill the wires and components, lower voltage, more efficient architecture, and new power-saving techniques employing power and clock gating as well as the big-little technique where "big" hardware is used in the case of heavy and medium workload and "little" less power consuming hardware in the case of low workload. To evaluate the effect of the new hardware, we measured the total power consumption and processor-only power consumption for more recent high-performance Apple MacBook Pro 15" (late 2016) (see Table 5.4).

Table 5.4 The average processor-only and total power consumption of new Apple MacBook Pro 15" (late 2016) laptop computer in H.264/AVC HD video playback on DivX Player using software decoder

Terminal	Processor power consumption (AVC SW decoder)	Total power consumption (AVC SW decoder)
Apple MacBook Pro 15" 2016	9.9 W	16.0 W

We observe that the total power consumptions of the 4 years newer MacBook Pro is 40.7% lower and the processor-only power consumption is even 50.0% lower than the baseline with the same videos and player software.

Another hardware-related power-saving method is to move to smaller footprint devices. The reductions are due to smaller displays and scaled-down processors and logic boards consuming less power. For that we measured the average power consumption of a smaller low-power Apple MacBook Air 13" (late 2014), Apple iPad Air 2 and Apple iPhone 6 Plus of which the two latter can natively display 1080p videos (see Table 5.5).

We observe that the switching to smaller footprint laptop, tablet computer or smartphone drops the total power consumption by 63.7%, 87.4% or 95.4%, respectively. The processor-only power consumption drops in the case of the laptop by 70.2% with respect to the baseline. We, however, acknowledge that the video quality (resolution 1440 × 900) in the MacBook Air 13" is clearly weaker than that of the MacBook Pro 15" due to its resolution utilizing only one-fourth of the pixels of MacBook Pro (2880 × 1800).

In the following subsections, we use the Apple MacBook Pro 15" (mid-2012) for defining the baseline representing a state-of-the-art terminal in the beginning of research. To exploit the low-power technology and advances in the product line, we use Apple MacBook Air 13" (late 2014) as a simple low-power small footprint alternative and Apple MacBook Pro 15" (late 2016) as a representative of the latest technology.

5.3.3 *Effect of Video Coding*

Coding of videos is an essential part of video delivery since without coding videos consume huge amount of storage and network resources. The downside of this is increased power consumption due to extra computation for encoding and decoding of videos and hardware resources needed for them. An interesting trade-off is that the more efficient the coding is, the less video delivery consumes energy in the network due to reduced data rate but the more energy in the headend due to more complex algorithms. In terminals, the situation is mixed, since more complex coding does not automatically imply more complex decoding but a smaller amount of data to be fetched to the decoder.

Table 5.5 The average processor-only and total power consumption of new Apple MacBook Air 13" (late 2014) laptop computer, Apple iPad Air 2, and Apple iPhone 6 Plus in H.264/AVC HD video playback on DivX, VLC, and VLC players with software decoders, respectively

Terminal	Processor power consumption (AVC SW decoder)	Total power consumption (AVC SW decoder) (W)
Apple MacBook Air 13" 2014	5.9 W	9.8
Apple iPad Air 2	<NA>	3.4
Apple iPhone 6 Plus	<NA>	1.25

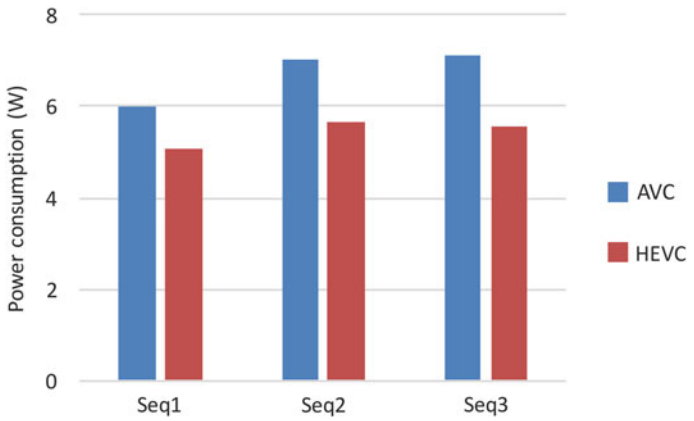


Fig. 5.1 The power consumption of the test video clips in DivX Player on Apple MacBook Pro 15" (late 2016)

Let us first have a look at the effect of coding standards. For that, we measured the power consumption of decoding and playing back the three test video clips encoded the "superfast" preset of ×264 and ×265 encoders in DivX player for H.264/AVC and H.265/HEVC coding standards, respectively. The results are shown in the (Fig. 5.1).

We observe that for the DivX player with these settings it is actually H.265/HEVC that leads to 19.1% lower power consumption although H.265/HEVC has a reputation of having higher computational requirements than H.264/AVC. The reasons for this unexpected behavior are likely that the decoder makes use of available parallelism in greater degree in H.265/HEVC processing so that the total amount of certain computational resources, e.g., caches, in use is actually higher, and that for some reason implementation of H.265/HEVC decoder seems better optimized. We will also see later that while this is the case with some other player software too, this does not hold, e.g., for the VLC player in which the power consumption is higher for H.265/HEVC than for H.264/AVC assuming the same quality.

The H.264/AVC and H.265/HEVC standards give a spectrum of choices for adjusting the packing, quality, and inclusion of different techniques and filters

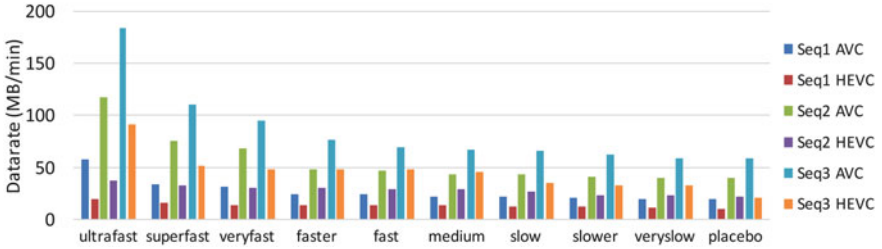


Fig. 5.2 The data rate of the Seq1 "slow motion", Seq2 "action", and Seq3 "end texts/animation" obtained with ten ×264 profiles for AVC and ten ×265 profiles assuming the same video quality

leading to different results. Since the number of parameters controlling these things is high, the × 264 and × 265 encoders provide a variety of *presets* featuring good-known values of the parameters for different compression factors. In order to have an overall understanding whether the usage of these parameters have effect on power consumption of video playback in terminals, we compressed the test videos using ten ×264 and ten ×265 presets for H.264/AVC and H.265/HEVC so that the obtained video quality in terms of average PSNR remains the same. The obtained data rates of the videos packed with these different settings are shown in (Fig. 5.2)

We observe that videos packed with slower presets lead to better compression and thus smaller data rate than those packed with faster ones. This is caused by the inclusion of more advanced packing tools and more optimal set of parameters in H.264/AVC and H.265/HEVC standards. In order to determine whether the data rate and selection of packing methods have an effect on the energy consumption also in terminals, we played back all the videos with these ten presets in the DivX player. The power consumption of playing back the clips is shown in (Fig. 5.3)

We observe that for H.264/AVC the lowest power consumption is obtained from the clips coded with preset "ultrafast" but with H.265/HEVC there is no single winner profile. Furthermore, packing videos more densely appears not to have clear effect on power consumption although they need to process less input data than sparsely packing presets.

All in all, according to these tests H.256/HEVC consumes less power than H.264/AVC, reduces the data rate by over 40% and the "very slow" preset gives almost as good and sometimes even better results than "placebo" but the encoding time and power consumption are significantly smaller than in "placebo". We will use these selections as the settings for most of our tests.

5.3.4 Player Software

An essential part for video playback in terminals is the player software that obtains the coded video from a local file or streams it from the delivery network, decodes it and shows the video on the display. Like with all software, there are differences in

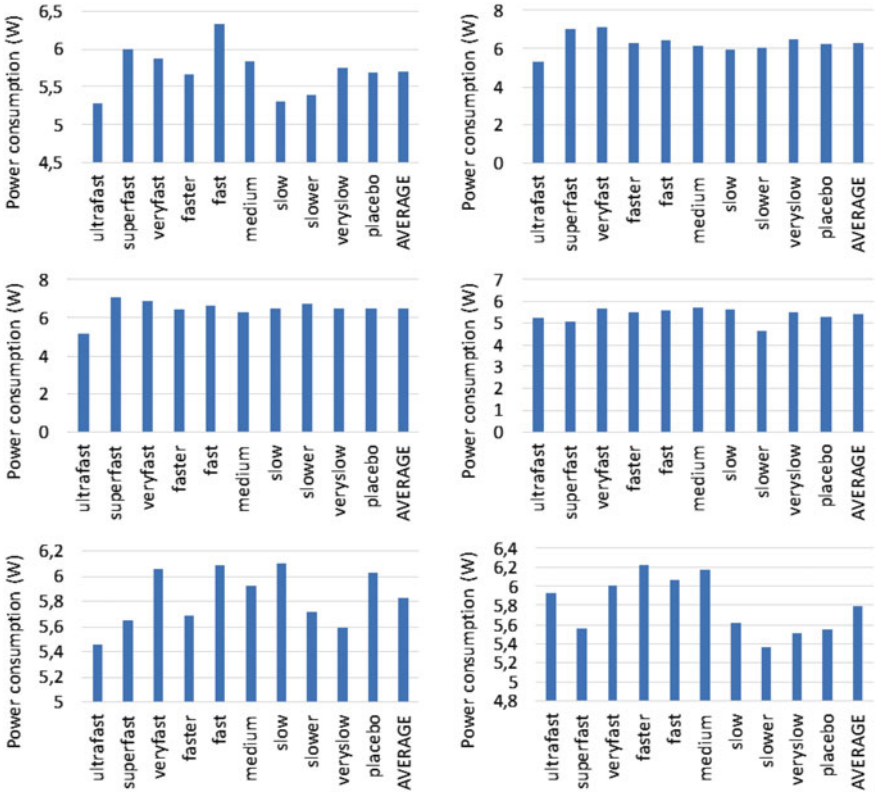


Fig. 5.3 The power consumption of the x264 profiles for AVC video clips Seq1 "slow motion" (top left), Seq2 "action" (top right), and Seq3 "end texts/animation" (middle left), as well as x265 profiles for HEVC video clips Seq1 "slow motion" (middle right), Seq2 "action" (bottom left), and Seq3 "end texts/animation" in DivX Player on MacBook Pro 15" (late 2016)

performance and power consumption of them. These differences are caused by the usage of different algorithms, optimization levels, and settings.

In order to evaluate the effect of player software to the energy usage in mobile terminals, we measured the power consumption of playing back the video clips with different player software, studied the effect of their parameters to the power consumption, and investigated how recompiling the player software specifically for the terminal at hand effects the power and finally explored the effect of optimization parameters to the power consumption.

5.3.4.1 Effect of Player Software

Out of a relatively high number of commercial and publicly available players, we selected VLC, which is another popular player software, and GPAC Osmo, which

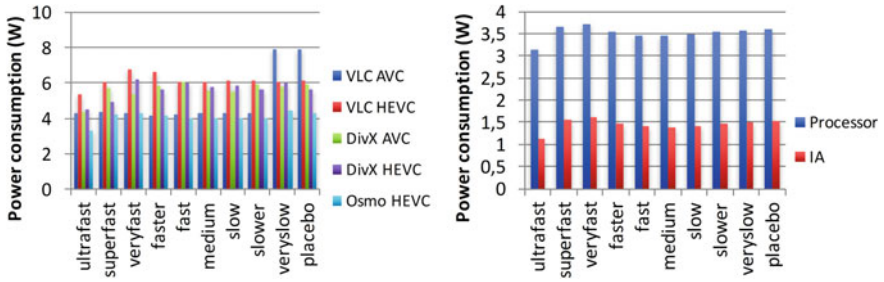


Fig. 5.4 The power consumption of selected players (left) and the power consumption of Osmo player by taking advance of frame and slice parallelism, FS-P4 (right) in Low-Power laptop MacBook Air 13”

makes use of the openHEVC decoder software and provides a number of user-adjustable parameters, for closer inspection and comparison to DivX.

We measured the effect of player software by displaying the video clips with DivX 10.7.2 player, VLC 3.0 prerelease, and GPAC Osmo 4 players using the 10 ×264 and 10 ×265 presets for H.264/AVC and H.265/HEVC, respectively. The results are shown in Fig. 5.4. Our observations are:

- The lowest power consumption is given by GPAC Osmo H.265/HEVC. The second best is VLC H.264/AVC except for “very slow” and “placebo” presets.
- The worst power consumption is VLC H.265/HEVC except in the case of “very slow” and “placebo” presets.
- The power consumption of DivX H.265/HEVC is lower than H.264/AVC with half of the presets.
- The high consumption of VLC H.265/HEVC with “very slow” and “placebo” is caused by nonfinalized optimizations of the prerelease version of VLC.

The averaged results for different players are summarized in Table 5.6 for MacBook Air 13”. The relative results for MacBook Pro 15” (late 2016) are similar but for the next subsections, we will use the former to minimize the effects of default parallelism setting before optimizing it in Sect. 5.3.7.3 (Table 5.6).

Based on these results and excellent configurability of GPAC Osmo we select it for further inspection along with H.265/HEVC coding.

Table 5.6 The average power consumption over presets and coding standards on DivX, VLC, and GPAC Osmo players executed in Apple MacBook Air 13”

Player SW	Power consumption, laptop LP (W)
DivX H.264/AVC	5.61
VLC H.264/AVC	5.02
DivX H.265/HEVC	5.61
VLC H.265/HEVC	6.15
GPAC Osmo H.265/HEVC	4.09

5.3.4.2 Effect of Player Parameters

GPAC Osmo has over 200 parameters adjustable via the configuration file, command line, and user interface. Since it is likely that the default configuration of parameters gives not the lowest power consumption, we browsed through the parameters and selected frame and slice parallelism, length of the *time slice* specifying the target maximum time of one cycle of the media manager and number of CB units, specifying the number of decoded frames in memory before display, for closer inspection. The results of adjusting *time slice* and CB units settings, as well as the effect of configurations, are shown in Fig. 5.5. The effect of parallelism is discussed in Sect. 5.3.7.3.

We observe reductions of roughly 6% and 3% of optimizing CB units and *time slice* parameters, respectively. The cumulative average power consumption effect as the improvements obtained by the optimal values of the parameters stacked on a top of each other in Apple MacBook Air 13" running GPAC Osmo are shown in Table 5.7. The "FS-P4" refers to the optimal parallelism setting for MacBook Air 13". See Sect. 5.3.7.3 for more details.

5.3.4.3 Recompiling the Player for the Terminal

Yet another player software-based power reduction technique is to recompile the software and related libraries with more optimal parameters. Like in the case of GPAC Osmo configuration, the parameter space of compiling is way too large to be

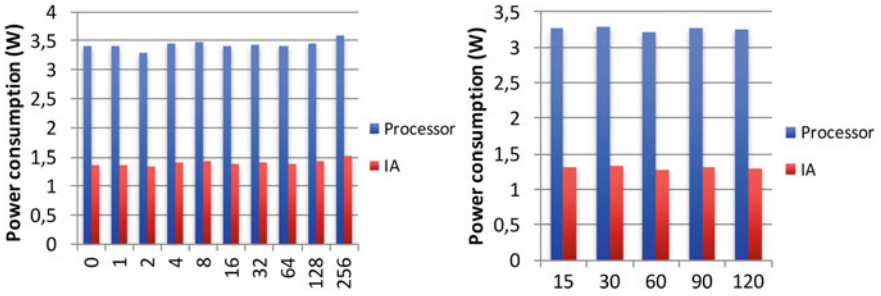


Fig. 5.5 The power consumption of GPAC Osmo player with different CBUnits (left) and Timeslice (right) values in Apple MacBook Air 13"

Table 5.7 The average power consumption of parameter optimizations shown cumulatively for Apple MacBook Air 13" executing GPAC Osmo Player

Configuration parameter	Power consumption (W)
Baseline	4.09
FS-P4	3.52
CBU units	3.30
Time slice	3.21

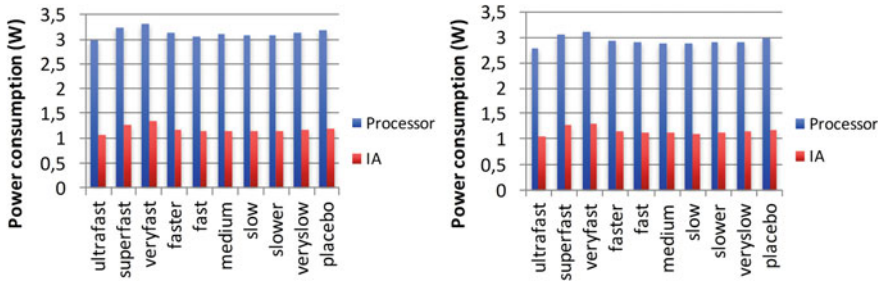


Fig. 5.6 The measurement results of power consumption for GPAC Osmo player after recompilation (optimization level O3) (left) and after recompilation of openHEVC (with optimization level O3) and taking advance of SDL2 library and AVX2 (right) in Apple MacBook Air 13”

Table 5.8 The average power consumption of recompiling GPAC Osmo player only and recompiling both player and the openHEVC decoder for Apple MacBook Air 13” (late 2014)

Optimization	Power consumption (W)
Baseline (player parameters reconfigured)	3.21
Recompilation w/O3	3.13
Recompilation of openHevc w/O3 and enabling SDL2 and AVX2	2.93

tested thoroughly. Again, we analyzed the parameters and selected the most obvious ones for closer inspection since they have known effect on the performance and, therefore, likely also on the power consumption.

In order to see the effect of this, we recompiled GPAC Osmo player and the underlying openHEVC decoder library with a number of different settings for the terminals. The summary of the obtained results for Apple MacBook Air 13” is shown in (Fig. 5.6).

We observe that recompiling Osmo drops the power consumption roughly by 2.5% and recompiling both GPAC Osmo and openHEVC decoder leads roughly 8.7% power reduction.

The average power consumption of the baseline obtained from the previous subsection, the consumption after recompiling GPAC Osmo player with the best combination of parameters and consumption after recompiling both player GPAC Osmo player and openHEVC decoder for Apple MacBook Air 13” are shown in (Table 5.8).

5.3.5 Effect of Quality

The quality of video clips being played back in a terminal is expected to have a direct effect on its power consumption because the worse the quality, the less

information one has to decode and process in the terminal. Let us study this with respect to resolution, quality settings, and frame rate.

5.3.5.1 Resolution

Resolution of the video is directly proportional to the data rate needed for moving it through the network but also the quality. In terms of power consumption, this should have a linear effect on the dynamic processing component of the power consumption. To evaluate this in practice, we measured the power consumption of DivX player with H.264/AVC coded videos and GPAC Osmo player with H.265/HEVC coded videos over the range 1080p…480p of resolution. The results for Apple MacBook Air 13" are shown in Fig. 5.7.

We observe that by decreasing resolution, the power consumption is also decreased for the both players. This is because the number of pixels processed in 720p and 480p are 55.6% and 80.2% is lower than in 1080p, respectively.

5.3.5.2 Effect of Quality Settings as PSNR

The *Peak Signal to Noise Ratio* (PSNR) means the mean squared error of the compressed signal with respect to the original signal. Despite of its limited correlation with the subjective human observed video quality measures, it is widely used as a metric for compressed video quality. In this subsection, we explore the effect of encoder quality settings (data rate) interpreted as PSNR to the decoding and playback of the resulting video. For that, we measured the power consumption of DivX player with H.264/AVC coded videos and GPAC Osmo player with H.265/HEVC coded videos over a range 43..37 dB of PSNR values. Figure 5.7 shows the results for Apple MacBook Air 13".

We observe that decreasing the PSNR decrease in power consumption. DivX looks to benefit more from PSNR dropping than GPAC Osmo, however, relatively speaking the effect is almost the same in both players.

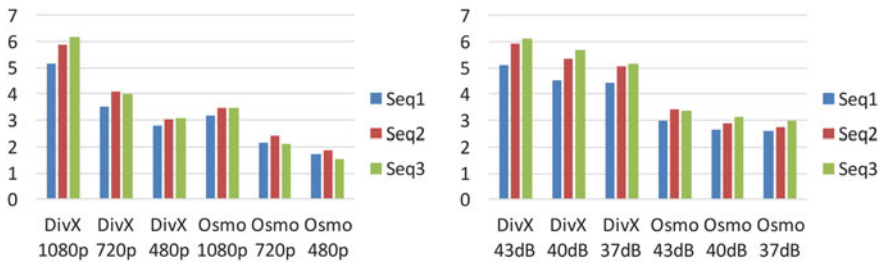


Fig. 5.7 The average power consumption of videos coded with H.264/AVC and played back in DivX player as well as with H.265/HEVC and played back in GPAC Osmo with different resolution (left) and PNSR (right) in Apple MacBook Air 13"

5.3.5.3 Effect of Frame Rate

Our final quality-related power-saving technique is to drop the frame rate of the video having a linear effect on the data rate. To evaluate this in practice, we measured the power consumption of DivX player with H.264/AVC coded videos and GPAC Osmo player with H.265/HEVC coded videos over a range 24.7 frames per second. The results are shown in (Fig. 5.8).

We observe that the power consumption decreases remarkably for both selected players when the framerate is decreased. The behavior is quite similar to that seen in Sect. 5.3.5.1 with video resolution.

5.3.6 Execution Environment-Related Issues

The execution environment of a laptop mobile terminal that may have an effect on the power consumption in video playback includes other programs and processes running in parallel with the player as well as the brightness and resolution settings of the display.

5.3.6.1 Effect of Other Programs

In modern multitasking operating systems, there are typically a number of user applications running in parallel. We study here how much these other applications have an effect on the power consumption if the video playback is running as the front application. To see the practical effect of other programs, we measured the processor power consumption of Apple MacBook Pro 15" with macOS Sierra

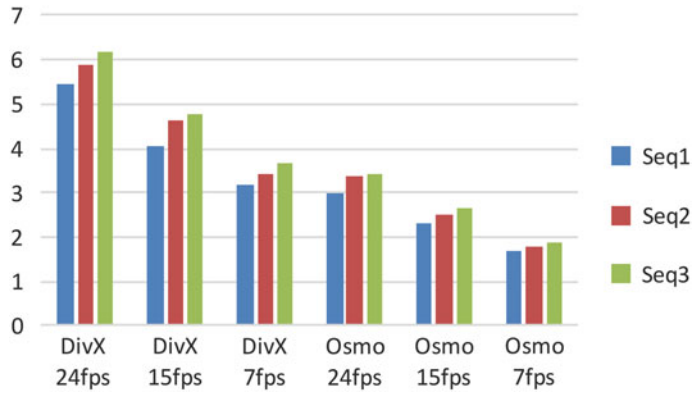


Fig. 5.8 The average power consumption of videos coded with H.264/AVC and played back in DivX player as well as with H.265/HEVC and played back in GPAC Osmo with different framerates for Apple MacBook Air 13"

Table 5.9 The power consumption of GPAC Osmo video playback along with a different number of applications running in parallel with the player in Apple MacBook Pro 15" with macOS Sierra 10.12

Load	Applications running in parallel	Power consumption (W)
None	GPAC Osmo only	3.3
Low	GPAC Osmo, Mail, Xcode	3.3
Medium	GPAC Osmo, Mail, Xcode, Safari (2 windows), Apple Maps, TextEdit, App Store	3.4
High	GPAC Osmo, Mail, Xcode, Safari (2 windows + Google Maps street view, YouTube video playback), Apple Maps, TextEdit, App Store, MS Word, MS Excel, MS PowerPoint	4.0

10.12 running GPAC Osmo player and a different number of application programs at the same time. The results are shown in the (Table 5.9).

We observe that the other programs running in parallel with video playback have an amazingly small effect on the power consumption unless they are intentionally performing heavy computation.

5.3.6.2 Effect of Display Brightness

Displays consume power for forming the pixels but also lighting them in the case of active matrix technology like employed in the line of tested Apple mobile terminals. This consumption can be substantial due to small size and overall power efficiency of terminals. In order to see, how much this has effect in practice, we measured the total power consumption of Apple MacBook Air 13" with three different brightness settings (minimum, medium and maximum) while the machine was idle, running a video with DivX and running a video with GPAC Osmo. The results are shown in (Fig. 5.9).

We observe that the difference between minimum display brightness and medium brightness is small but that the maximum brightness gives a substantial contribution to the power consumption.

5.3.6.3 Effect of Display Resolution Settings

Laptop displays can be set to lower resolution modes for compatibility of old programs and external displays if mirroring is used. There may be an opportunity to reduce energy consumption depending on the realization of the displays if lower display resolutions are used. To evaluate the effect this in practice we measured the power consumption of Apple MacBook Air 13" playing back full HD videos in DivX and GPAC Osmo players with different display resolutions ranging from 1440×900 down to 1024×640 pixels. The results are shown in the (Fig. 5.10).

From the results, we can observe that there can be some benefits by decreasing the display resolution. However, the benefits are quite minor when compared to the

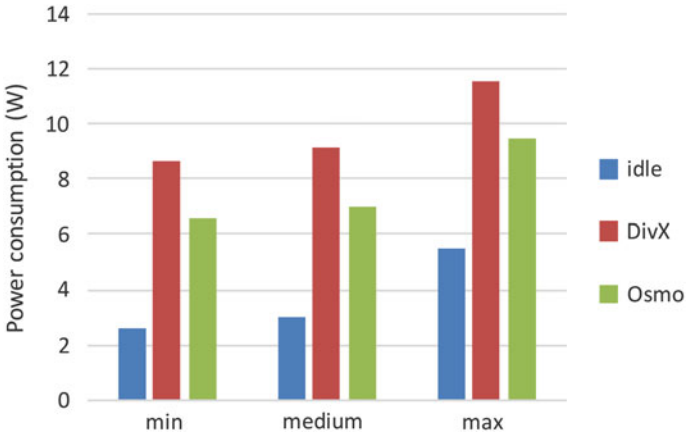


Fig. 5.9 The effect of display brightness for DivX and GPAC Osmo players measured in Apple MacBook Air 13”

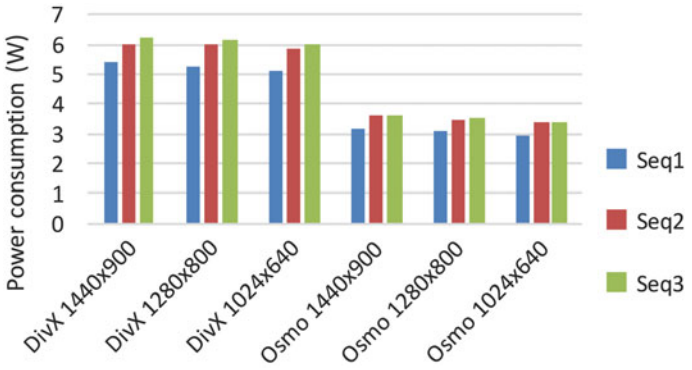


Fig. 5.10 The average power consumption of Apple MacBook Air 13” with different display resolution while running DivX and GPAC Osmo players

results obtained by lowering the resolution already in the video encoding, like seen in Sect. 5.3.5.1.

5.3.7 Execution Parameters

5.3.7.1 GPU Settings

The baseline MacBook Pro 15” RETINA (mid 2012) and MacBook Pro 15” (late 2016) have two GPUs—an Intel GPU that is built into the processor chip and a high-performance external one provided by NVIDIA and ATI, respectively. Since

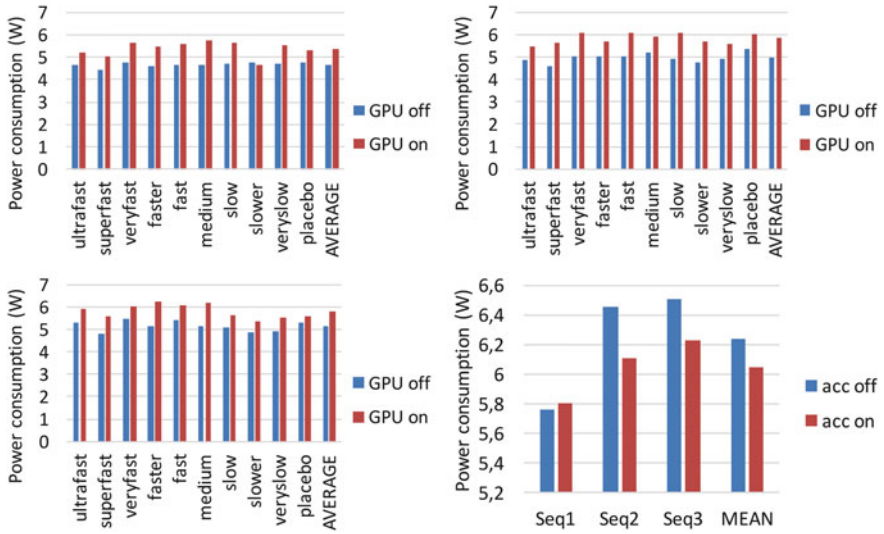


Fig. 5.11 The power consumption of the x265 profiles for the HEVC video clips Seq1 "slow motion" (top left), Seq2 "action" (top right) and Seq3 "end texts/animation" (bottom left) in DivX Player on High-Performance laptop MacBook Pro 15" (late 2016) with external GPU off/on. The power consumption of the "very slow" preset H.264/AVC video clips in DivX Player on High-Performance laptop MacBook Pro 15" (late 2016) with accelerator off/on

there is difference in integration and performance of these systems, we expect to see some difference also in power consumption. We measured the power consumption of clips both with internal GPU and external GPU in High-Performance laptop MacBook Pro 15" (late 2016) in order to see the effect of power consumption (see Fig. 5.11).

According to the measurements, the power consumption drops by switching external GPU off by 0.74 W in average that makes 13.1% relative drop. This is not the total difference between the GPUs since our Intel Power Gadget-based power measurement methodology does not include the power consumption of external GPU and we do not have a direct way to measure it. However, we can estimate it by measuring the total power consumption of the terminal with both GPU settings and by comparing the obtained results to above measurements. As the result of this, we observe that the power consumption by forcing external GPU off drops the power consumption by 3.7 W which is corresponds 37.4% reduction in Apple MacBook Pro 15" (late 2016) without other power-saving techniques.

5.3.7.2 HW Acceleration

In this chapter, we have deliberately ruled focus out of hardware decoder-based solutions. For curiosity, however, we measured the effect of hardware acceleration

in DivX for H.264/AVC since the acceleration is not available in GPAC Osmo nor for H.265/HEVC even though there is a HW accelerator for both of the standards in the Intel Core i7 "Skylake" version present in Apple MacBook Pro 15" (late 2016). The results are shown in Fig. 5.11.

We can observe that the speedup provided by the accelerator is only 3.1% in average for the most relevant preset "very slow". The speedup is higher for profiles that packing the video more sparsely and, therefore, less relevant for the video-on-demand use case.

5.3.7.3 Parallelism Settings

Exploitation of available parallelism increases the performance of parallel computation-aware architectures by executing multiple instructions, threads or processes in parallel, which increases the power consumption due to increased utilization of hardware elements and overheads needed to handle intercommunication and synchronization. With certain conditions, it can also be used to decrease the power consumption by letting the processor run at lower speed but also at lower voltage and by replicating availability of certain key resources like caches and registers that are present in each processor core.

To figure out the effect of parallelism in our case, we investigated the impact of frame parallelism, slice parallelism, and their combination on video processing with openHEVC decoder and GPAC Osmo player utilizing openHEVC. Since the number of processor cores in Apple MacBook Pro 15" is larger than in Apple MacBook Air 13" we ran the tests for both terminals. The results for Apple MacBook Air 13" are shown in (Fig. 5.12).

From the results, we observe that the most efficient setting of parallelism is different for openHEVC decoder and GPAC Osmo player. It also was noted in investigations that changing configuration or recompilation setting, or hardware (e.g., Apple MacBook Pro 15" late 2016) will lead to different results.

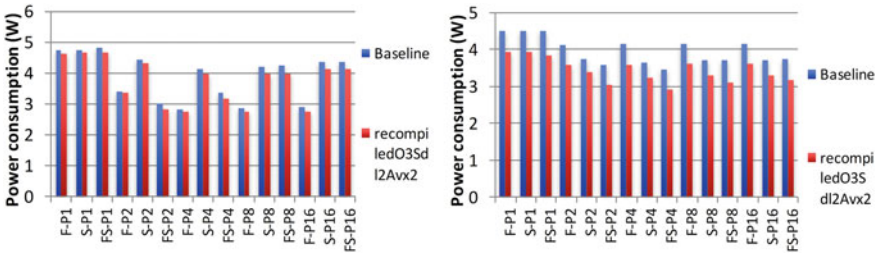


Fig. 5.12 The average power consumption of Apple MacBook Air 13" playing back videos with different parallelism setting for openHEVC decoder. These results are obtained by stretching the energy usage evenly for the duration of the video in order to make comparisons possible (left). The average power consumption with different parallelism setting for GPAC Osmo player (right)

The most efficient parallelism settings for executing openHEVC and GPAC Osmo in Apple MacBook Air 13" with the used configurations and recompilation settings were frame parallelism only, four processes (F-P4) and frame and slice parallelism, four processes (FS-P4), respectively. For Apple MacBook Pro 15" the most efficient settings were F-P16 and FS-P8, respectively.

One would expect that the decoding power of openHEVC would be smaller than the total playback power of GPAC Osmo but this is not the case. As anticipated above, this behavior is caused by the DVFS of the processor that has to increase the voltage of the processor to be able to compute decoding as fast as possible. We recall that power consumption is a quadratic function of voltage (cubic function if we take frequency into account).

5.3.8 *Streaming Versus Local Playback*

Local playback of videos after downloading them along with the first watching instead of streaming them repeatedly from the service provider may save power in both video delivery network and terminal. To have a practical understanding of this, we set up a streaming server containing the encoded MPEG DASH segments and streamed videos to the terminal through a 20 Mb/s WiFi connection. The results are seen in (Fig. 5.13).

From the measurement results, we can see that for GPAC Osmo player the streaming brought some additional power consumption when compared to the local playback.

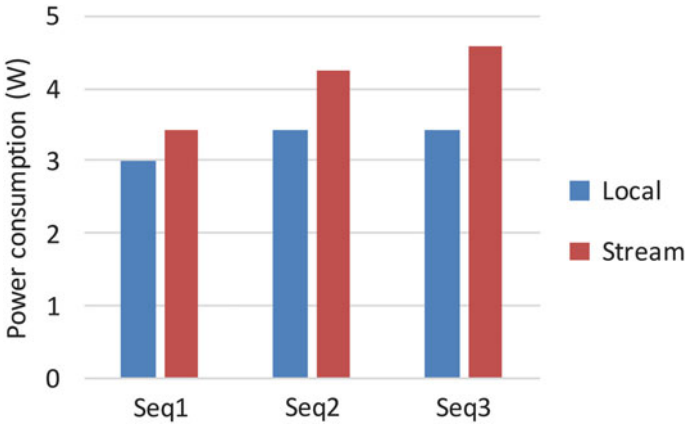


Fig. 5.13 The average power consumption of Apple MacBook Air 13" with local playback vs. MPEG DASH streaming from a headend video server

5.3.9 Overall Improvements

The overall improvement using the best combination of the techniques presented in previous sections and retaining the video quality is almost 83% lower processor power consumption and 43% smaller data rate (Table 5.10).

The power consumption could be further pushed down by reducing the quality and/or switching the hardware to low-power laptop or even smaller footprint tablet computer and smartphone. In these cases, mobility increases with the cost of quality.

5.3.10 End-to-End Considerations

The overall power reductions obtained with the techniques described above assume that coding is switched from H.264/AVC to H.265/HEVC. This means that encoding of the videos in the headend will consume more power as projected in Chap. 4. Let us figure out how many streaming playbacks of a video clip are needed in average to so that the savings in terminals cover the extra power used in the headend for coding it. Our assumptions are that the videos are streamed from the headend in MPEG DASH format to the terminals and that the same DivX H.264/AVC baseline setup and GPAC Osmo setup using H.265/HEVC and all the improvements described above. We measured the energy consumption and data rate of video Seq2 encoded with preset "superfast" that allows real-time encoding in a normal PC. The results are shown in (Fig. 5.14).

We observe that encoding H.265/HEVC version consumes 144% more energy than encoding H.264/AVC version. This extra power can be reduced by up to 39% by decreasing the quality to 720p. On the terminal side, the situation is quite opposite. Playback of the H.265/HEVC coded video consumes 81.5% less energy than H.264/AVC and the difference gets even bigger if we decrease the quality of the video. This is caused by using optimized GPAC Osmo instead of DivX that handles streaming very inefficiently.

For curiosity, we determined how many playbacks per encoding are needed to compensate the extra energy used by switching from H.264/AVC to H.265/HEVC coding. The results are illustrated in Fig. 5.14.

Table 5.10 The average processor-only and total power consumption and datarate in the baseline Apple MacBook Pro RETINA (mid 2012) and in the new Apple MacBook Pro 15" (late 2016) laptop computers running H.264/AVC HD video playback on DivX using the baseline settings and H.265/HEVC playback on GPAC Osmo with all the above improvements on

Terminal	Processor power usage (W)	Gain	Total power usage (W)	Gain	Datarate (MB)	Gain
Baseline	19.8		27.0		32.9	
Our proposal	3.4	82.8%	9.5	64.8%	22.4	42.9%

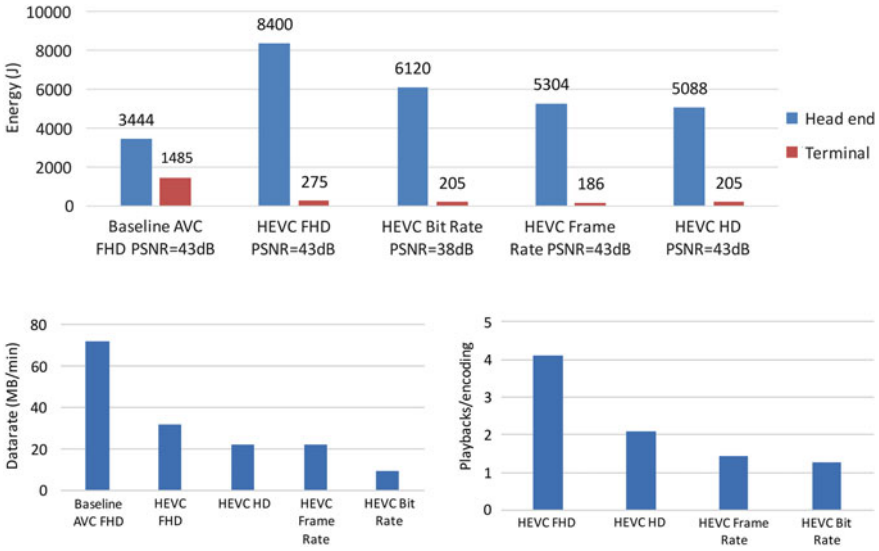


Fig. 5.14 The average energy consumption of encoding in our headend server and playback in Apple MacBook Pro 15" terminal (top), data rate of resulting MPEG DASH streams (bottom left) and playbacks/encoding needed to compensate the extra energy consumed by switching from H.264/AVC to H.265/HEVC (bottom right) for video clip Seq2. ×264 encoder was used for H.264/AVC and ×265 encoder was used for H.265/HEVC coding. The data rates of streamed videos were Full HD (FHD) and HD-ready (HD). (Bit rate = reduced bit rate, Frame Rate = reduced frame rate)

5.3.11 Additional Processor-Level Considerations

Yet another power reduction technique is to use theoretically more advanced/optimized parallel processor for video computing. To evaluate the effect of using REPLICA chip multiprocessor being developed at VTT [20] with the threading system and interconnection network improvements presented by Forsell et al. [21, 22] instead of Intel Core i7 present in the MacBook Pro 15" (late 2017), we modeled the architecture and video computing functionality estimated the maximum clock rate, silicon area, and power consumption of the processor assuming the state-of-the-art 11 nm silicon implementation using detailed analytical modeling of the architecture [18, 19] for the video computing functionality. The lowest power consumption configuration still capable of taking care of the functionality (among the evaluated ones) featured 4 processor cores with 9 functional units each and estimated to run at 723 MHz. The power consumption estimates for that are shown in Table 5.11.

We observe that with this speculative technique, the overall processor-only reductions jump to 93.8% and total power reductions to 72.9%. However, the accuracy of these estimations is limited due to multiple modeling levels.

Table 5.11 The average processor-only and total power consumption estimates for Apple MacBook Pro 15" (late 2016) laptop computer running H.265/HEVC playback on GPAC Osmo with all the above improvement on but assuming its Intel Core i7 processor would be substituted by an optimized REPLICA processor with the threading system and interconnect improvements

Terminal	Processor power usage (W)	Gain	Total power usage (W)	Gain
Baseline	19.8		27.0	
Our proposal with REPLICA	1.23	93.8%	7.3	72.9%

5.4 Conclusions

We have described a variety of techniques for reducing the power consumption of mobile terminals in video playback with software decoder as a part of video delivery network. The techniques include using the latest hardware for more power efficient operation, usage of more power-savvy and configurable video players, e.g., GPAC Osmo, optimization of video player and underlying libraries by finding more optimal set of parameter values and simple recompilation, usage of latest video coding format, e.g., H.265/HEVC and finding more optimal parameters for them, decreasing the quality of the video, and optimizing the terminal execution parameters and environment. We measured systematically the practical effect these techniques to the power consumption on a line of Apple laptop terminals. The results are surprisingly good with respect to the baseline of the terminal hardware, coding and popular DivX video player in the beginning of research. The overall improvements of the techniques applied to latest terminal hardware retaining the video quality in video computing power and total power consumption were as high as 82.8% and 64.8%, respectively. At the same time, the needed data rate dropped by 42.9% leading significant savings also in the video delivery network. Additional power reductions can be achieved by dimming the display, by closing down the other software running in the terminal while playing back videos, dropping the quality of videos, by switching to smaller footprint devices, by replacing the current processors with more advanced ones, and by using hardware accelerators.

Acknowledgements This research was supported by the European Celtic-Plus project CONVINcE and funded by the Finnish Funding Agency for Innovation (TEKES) and VTT. I would like to thank Tuomas Nissilä and Jussi Roivainen who helped in the measurements.

References

1. Mills M (2013) The cloud begins with coal: big data, big networks, big infrastructure, and big power—an overview of the electricity used by the global digital ecosystem. Digital Power Group
2. Ericsson Mobility Report (2013) Ericsson June 2013
3. Balasubramanian N, Balasubramanian A, Venkataramani A (2009) Energy consumption in mobile phones: a measurement study and implications for network applications. Paper presented at IMC'09, Chicago, Illinois, USA, 4–6 Nov 2009

4. Carroll A, Heiser G (2010) An analysis of power consumption in a smartphone. Paper presented at the 2010 USENIX annual technical conference, Boston, MA, 23–25 June 2010
5. Choi M (2013) Power performance analysis of smart device. Int J Smart Home 7:57–66
6. Chen X, Chen Y, Ma Z, Fernandes F (2013) How is energy consumed in smartphone display applications? Paper presented at ACM HotMobile'13, Jekyll Island, Georgia, USA, 26–27 Feb 2013
7. Li X, Ma Z, Fernandes F (2012) Modeling power consumption for video decoding on mobile platform and its application to power-rate constrained streaming. Paper presented at 2012 IEEE visual communications and image processing, San Diego, CA, USA, 27–30 Nov 2012
8. Information technology-MPEG Systems Technologies-Part 11: Energy-Efficient Media Consumption (Green Metadata) (2015) ISO/IEC JTC1/SC29/WG11 International Standard. 23 001-11, July 2015
9. Fernandes F, Ducloux X, Ma Z, Faramarzi E, Gendron P, Wen J (2015) The Green Metadata standard for energy-efficient video consumption. IEEE Multimed Mag 22:80–87
10. Ostermann J et al (2004) Video coding with H.264/AVC: tools, performance, and complexity. IEEE Circuits Syst Mag 4:7–28
11. Sullivan G, Ohm J-R, Han W-J, Wiegand T (2012) Overview of the high efficiency video coding (HEVC) standard. IEEE Trans Circuits Syst Video Technol 22:1649–1668
12. Viitanen M, Vanne J, Hämäläinen T, Gabbouj M, Lainema J (2012) Complexity analysis of next-generation HEVC decoder. Paper presented at the 2012 IEEE international symposium on circuits and systems (ISCAS), Seoul, South Korea, 20–23 May 2012
13. Mesa M, Chi C, Schierl T, Juurlink B (2012) Evaluation of parallelization strategies for the emerging HEVC standard. Paper presented at the 2012 IEEE international conference on acoustics, speech and signal processing
14. Bossen F, Bross B, Sühring K, Flynn D (2012) HEVC complexity and implementation analysis. IEEE Trans Circuits Syst Video Technol 22:1685–1696
15. Han B, Wang R, Wang Z, Dong S, Wang W, Gao W (2014) HEVC decoder acceleration on multi-core X86 platform. 2014 IEEE international conference on acoustic, speech and signal processing (ICASSP), Florence, Italy, 4–9 May 2014
16. Hamidouche W, Raulet M, Déforges O (2014) Multi-core software architecture for the scalable HEVC decoder. Paper presented at acoustics, speech and signal processing (ICASSP), 2014 IEEE international conference on, Florence, Italy, 4–9 May 2014
17. Hennessy J, Patterson D (2011) Computer architecture: a quantitative approach, 5th edn. Morgan Kaufmann
18. Pamunuwa D, Zheng L-R, Tenhunen H (2003) Maximizing throughput over parallel wire structures in the deep submicrometer regime. IEEE Trans VLSI Syst 11:224–243
19. Forsell M (2010) On the performance and cost of some PRAM models on CMP hardware. Int J Found Comput Sci 21:387–404
20. Forsell M, Roivainen J (2015) REPLICA T7-16-128—A 2048-threaded 16-core 7-FU chained VLIW chip multiprocessor. Presented at the special session on multicore, manycore and distributed systems at the 48th asilomar conference on signals, systems, and computers, Pacific Grove, USA, 2–5 Nov 2014
21. Forsell M, Roivainen J, Leppänen V (2016) Outline of a thick control flow architecture. Presented at the 5th workshop on parallel programming models special edition on task parallelism, Marina del Rey Marriott, Los Angeles, USA, 26–28 Oct 2016
22. Forsell M, Roivainen J, Leppänen V (2016) The REPLICA on-chip network. Presented at the 2016 IEEE nordic circuits and systems conference, Copenhagen, Denmark, 1–2 Nov 2016

Chapter 6
Software Measurement of Energy Consumption on Smartphones

Thomas Vincent and Olivier Philippot

6.1 Introduction

Energy consumption is critical in the domain of smartphones; yet, the usual techniques to measure such consumption may not apply for handheld devices. In fact, energy measurement has always been performed in hardware-related fields. A lot of systems are thus available to measure the consumption of several platforms: industrial infrastructure, household appliances, batteries, and others. Measurement instruments are consequently adapted to these platforms: higher tension and intensity, low frequency of measure, and instant measure.

The involved software has several different requirements. It is indeed necessary to have a higher measure frequency, OS being cadenced at 10 ms. The currents used to measure can be high in data centers and really low in Internet of Things (IoT). Similarly, the software has varying behaviors in terms of resource consumption; energy consumption must then not be performed instantly but during a certain time. These constraints are closer to profiling tools than measurement tools.

The software system has different needs which brought a variety of tools, from IDEs to classical measurement devices. They differentiate on their level of integration in the developer toolbox, on the use of physical or modeled measurement, and others. Selecting a tool depends on the environment and also on the constraints related to each platform (accuracy, flexibility, and ease of installation).

In the following, three methodologies to measure energy consumption are presented. Section 2 describes the measurement using external hardware, Sect. 3 focuses on modeling tools allowing one to estimate the power consumption,

T. Vincent (✉) · O. Philippot
GREENSPECTOR, Nantes, France
e-mail: tvincent@greenspector.com

O. Philippot
e-mail: ophilippot@greenspector.com

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0_6

and Sect. 4 describes the embedded hardware measurement. Finally, Sect. 5 concludes the benefits and drawbacks of each methodology in the context of smartphones.

6.2 External Hardware Measurements

When considering energy measurement, the first idea is to use an external device, such as a wattmeter, to perform the measurements.

External hardware measurement is the most common method for energy measurement. Indeed, devices do not have any embedded connector or instrument. It is thus necessary to connect a "black box" to the system to be measured. This solution is the less intrusive one in most cases because it does not require to install software on the device or to modify it. However, this does not hold for certain types of platforms such as smartphones where accessing the battery can be complicated, if not impossible. In any case, having access to the power source is mandatory in order to connect such kind of hardware.

To measure the energy consumption, these devices rely on both current measurement and voltage measurement, which have to be measured in parallel. These probes can be classified depending on the type of integration to the power source:

- Probes requiring a definitive cut of the power supply chain. It can for instance consist of a shunt between a phone battery and the phone electrical input. This setup is difficult because it does not require to unplug and plug back a power plug but also to add clips or to solder the probe between the power source and the system.
- Probes requiring a temporary cut of the power supply chain. This system is the most widespread. The probe is located between the power plug and the system. A simple unplug/plug is necessary without other modification of the system.
- Probes not requiring cutting the power supply chain. This is the less invasive system where a loop is put on the power cable to perform the measurement. Using such a probe can be mandatory in some contexts where cutting the power is not possible, such as production servers.

Where to Measure

The energy consumption can be physically measured at different levels, each one of them having advantages and drawbacks:

- At the board input: the consumption for the whole device is measured. It has the advantage of providing a global and simple vision of the power consumption, as no part of the hardware is ignored. However, it is difficult to understand the behaviors or to identify the consumption of a given component.
- At each rail of power supply: consumption is measured after each regulator (e.g., 3.3, 5 V, etc.). This provides a more fine-grained vision but it can still be difficult to interpret the measurements if a specific rail powers many

components. Another limit is that it requires instrumenting a lot more hardware than to simply measure the global consumption: more connections, more connection-related problems, noise processing, and others.
- At each component (CPU, GPU, etc.): this approach is interesting because it allows to better understand the behavior of the hardware depending on the application. However, the required complexity of the setup is of course important.

It is worth to note that dedicated wattmeters exist for smartphone, such as the Monsoon Power Monitor.

6.3 Modeling Tools

Energy probes are not necessarily integrated in platforms and it can be cumbersome to set up a measurement device. In that case, estimating the energy consumed can become relevant. Power estimation relies on mathematical models taking the system activity as input.

Several approaches exist to model energy consumption. They can be classified as follows [1]:

- Deterministic—this approach maps the system's operations to stimulations to parts of the hardware (CPU, memory). Knowing the power consumption of each hardware component allows one to compute the consumption of the software.
- Statistic—this approach finds a model allowing it to map the system consumption to certain variables such as the network frames using a statistical model.

These approaches have the advantage of having a finer granularity on the energy consumption: rather than having a global overview of the consumption of the whole platform, it gives us an estimation at the process level or at a subsystem level (radio system and CPU). Moreover, it removes the constraint of acquisition frequency to be only constrained by the data acquisition frequency (CPU and memory) and processing time.

The limit of those modeling approaches is given by the necessity to do a significant modeling and analysis work during the algorithm design. If these models are not maintained by designers, they will not remain viable. In fact, they have to take into account every new architecture, evolutions in the hardware behavior, and others. Another limit is the possible incompleteness of the model. Not taking a parameter (e.g., GPS) into account may result in a nonrepresentative consumption estimation in some particular use cases. It is for instance the case of ARO, which focuses on network only or power API on the CPU.

Power API
Power API is a middleware toolkit for building software-defined power meters. Software-defined power meters are configurable software libraries that can estimate the power consumption of software in real time.

Intel RAPL
Starting from SandyBridge, Intel CPUs integrate an energy model: Running Average Power Limit (RAPL). The model is based on hardware counters, temperature, and leakage. RAPL provides a set of counters providing energy and power consumption information.

PowerTOP
PowerTOP is a Linux tool maintained by Intel. It aims at diagnosing issues related to power consumption and power management. PowerTOP has an interactive mode where the user can experiment with various power management settings for cases where the Linux distribution has not enabled these power saving settings.

ARO
AT&T Application Resource Optimizer (ARO) is a free, open-source tool for developers and testers that provides recommendations to optimize the performance of mobile web applications, make battery usage more efficient, and reduce data usage.

The model is based on the behavior of radio cells (2G, 3G, etc.) and their power consumption depending on their state.

Summary
The following table sums up the models presented and their approach. It is worth noting that only ARO can be directly used in current smartphones, with the limitation of only estimating the energy of the radio cells.

Project	Description	Approach
ARO	Estimation of the energy consumption of radio cells	Statistic
Power API	Estimation using data from CPU, memory, etc.	Deterministic
RAPL	Estimation from CPU data (Intel only)	Deterministic
Power TOP	RAPL application with CPU data such as CState	Deterministic

6.4 Embedded Hardware Measurements

Strong energy constraints made manufacturers integrate probes directly in the hardware. Continuously monitoring energy is indeed necessary for users. The most visible application is battery gages. Without energy or current probes, the battery level cannot be displayed. Similarly, data center managers can monitor precisely the consumption of all their servers using embedded probes in power supplies or energy distribution systems.

Such solutions present the advantage of freeing us from adding hardware such as an external wattmeter. Moreover, they are fit to the platform. However, they are constrained by the electronic design (mainstream electronics with low-cost circuits, small factor card). Accuracy depends in this case on platform.

The complexity of implementation depends strongly on openness and documentation of the APIs allowing one to retrieve information. In fact, in certain cases, the embedded probes can be used for the manufacturer's algorithms. For instance, for a phone, the probe allows it to compute precisely the battery load. Consequently, the API provided is not always documented.

How the Battery Load is Computed

The battery load of smartphones is computed from data gathered by the phone. The algorithm relies on several elements, generally current, tension, and temperature. Integrated circuits allow this computation to be done, for instance, the Maxim MAX17047 chip. This chip provides several possible estimation techniques:

- Tension-based computation (Open-Circuit Voltage OCV)—the estimation relies on a complex model, which takes into account the behavior of the tension of battery through time.
- Current-based computation—it consists in a coulomb counter measuring the battery discharge through time.
- Impedance-based computation—the algorithm tries to measure the impedance of the battery in order to infer the battery level.

The accuracy of the estimation of the SOC can be increased by combining techniques. This is for instance done in the Maxim chip.

However, these algorithms present some limits when used in real life. A diminution of the total battery capacity (Full charge capacity) can be observed. Numerous studies show this behavior, such as [2], where it is stated that "Among 9560 user devices, 5311 devices of 333 models had reduced battery capacity. They show that more than 50% of the devices of nine popular models had reduced battery capacity."

6.5 Conclusion

We introduced three methodologies to perform energy consumption measurements: external hardware measurement, measurement using modeling tools, and embedded hardware measurement. External hardware measurement provides reliable results but it requires to physically access the power source or the board of the device to be measured, which can be a deal breaker in the context of smartphones. Modeling tools are potentially the most accurate method of the three considered as they rely on a deep understanding of the hardware. However, the segmented nature of the smartphone market makes it nearly impossible to provide a solution that would fit every device. Finally, using embedded hardware measurement, although being

heavily dependent on the tools provided by device manufacturers in terms of reliability and accuracy, is nonetheless widespread and easy to use. Of these three methodologies, the last one is consequently the more promising as it allows the measurement to meet the criteria of flexibility, ease of use, and widespread usage.

Acknowledgements This work was partially supported by the European Celtic-Plus project CONVINcE.

References

1. Xiao Y (2011) Modeling and managing energy consumption of mobile devices, Aalto University publication series DOCTORAL DISSERTATIONS, 139/2011, Aalto, pp 1–74. ISBN 978-952-60-4430-9, ISSN 1799-4942
2. Hoque MA, Siekkinen M, Koo J, Tarkoma S (2016) Accurate Online Full Charge Capacity Modeling of smartphone batteries, arXiv:1604.05689v2

Chapter 7
Energy Efficiency in Wireless Multimedia Sensor Networking: Architecture, Management and Security

Erkki Harjula, Tenager Mekonnen, Miika Komu, Pawani Porambage, Tero Kauppinen, Jimmy Kjällman and Mika Ylianttila

7.1 Introduction

The Internet of things (IoT) brings computation everywhere around us by connecting the surrounding smart objects to the Internet. IoT enables rapid development of applications that are sensing and modifying our environment in a way that was not possible with traditional technologies. Wireless sensor networks (WSN) is an application area within IoT, which enables collaborative communication among low-cost, low-capacity and power-restricted devices with sensing and computing capabilities [1]. A WSN node consists of a transceiver, an independent energy supply and a sensing unit capable of collecting data such as location, acceleration, temperature, humidity and pressure.

E. Harjula (✉) · T. Mekonnen · P. Porambage · M. Ylianttila
Centre for Wireless Communication (CWC), University of Oulu, Oulu, Finland
e-mail: erkki.harjula@oulu.fi

T. Mekonnen
e-mail: tenager.mekonnen@oulu.fi

P. Porambage
e-mail: pawani.porambage@oulu.fi

M. Ylianttila
e-mail: mika.ylianttila@oulu.fi

M. Komu · T. Kauppinen · J. Kjällman
Ericsson Research, NomadicLab, Jorvas, Finland
e-mail: miika.komu@ericsson.com

T. Kauppinen
e-mail: tero.kauppinen@ericsson.com

J. Kjällman
e-mail: jimmy.kjallman@ericsson.com

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0_7

Wireless multimedia sensor networking (WMSN) extends the WSN technology by introducing more powerful sensor nodes capable of capturing and delivering multimedia data from multimedia sensors, such as still cameras, video cameras, infrared cameras and microphones [2]. The current use cases for WMSN include different surveillance and monitoring systems, such as intruder detection for property owners [3], monitoring large open areas in airports [4], monitoring crops and farm equipment in agricultural plots [5], monitoring conditions of affected people during an accident [6] and monitoring elderly people in assisted living scenarios [7]. Furthermore, WMSN is—together with emerging edge computing (EC) paradigm—a centric area of the future remote and self-controlled systems, including autonomous ground, sea and air traffic, where reliable, low-latency and high-quality video feed is a core requirement [8, 9]. Overall, video networking is continuously growing its share of Internet traffic. Recent forecasts show that the IP video traffic will be 82% of all the global IP traffic by 2020, of which video surveillance contributes 3.9%, up from 1.5% in 2015 [10].

Low energy consumption is among the first-priority design principles for WSNs and WMSNs, since most of the wireless sensor nodes are battery-powered, and battery life requirements are high [11, 12]. In a typical WSN node, the wireless interface dominates the total energy consumption [13]. During the recent years, the industry has focused on developing ultra-low-power short-range radio technologies, such as Bluetooth Low Energy (BLE), ZigBee, Z-Wave and ANT [14], and reducing the idle power consumption of nodes by introducing different levels of sleep modes [15]. In general, the methods for improving energy efficiency in IoT networking include (1) network topology and routing optimization, (2) advanced sleep/wake-up schemes, (3) data reduction, (4) parallel data transfers and (5) computational offloading [14].

In WMSN, the sensing and processing of multimedia data (sound, video) require higher performance computing and communication hardware that in turn consumes more energy [2]. Thus, suitable off-the-shelf hardware for WMSN are difficult to find. Traditional WSN nodes are too restricted with regard to their hardware capacity, and traditional visual monitoring systems such as closed-circuit television (CCTV) are either bound to fixed power supply or require high battery capacity or external power source. The fast development of cheap and compact embedded computers has resulted in popular commercial products, such as Raspberry Pi. Raspberry Pi can be powered from battery, and it is capable of capturing and streaming high-definition video, but it lacks sufficient power management for fulfilling the battery life requirements of sensor networking use cases. A promising method for improving the energy efficiency of WMSN systems is to use multi-tier network architectures [2, 16]. This allows low-power sensor nodes to wake up the high-power high-performance nodes when needed. For example, in video surveillance systems, a low-power motion sensor node can send a trigger message, e.g. using BLE radio, to wake up a video sensor node which then starts streaming video to the Internet and shuts down if no movement has been detected for a while.

In IoT scenarios, hundreds or even thousands of nodes may be deployed, and they may occasionally need software upgrades to improve their functionality,

reliability or security, or to adapt to changes in the infrastructure. Managing large deployments may turn out to be a challenging task, especially in multimedia sensor scenarios, such as video surveillance, where the cameras are often off when no activity is detected. Thanks to the rapid development of IoT node hardware capabilities, virtualization has become a viable option for node management for the most powerful IoT nodes, such as video sensor nodes in video surveillance scenarios. Lightweight virtualization technologies have revolutionized the world of software development by introducing flexibility and innovation to this domain [17, 18]. Virtualization can be utilized for upgrading the firmware and managing the configuration of camera nodes in generic, hardware-agnostic way. Previous studies have demonstrated the feasibility of using container technologies on IoT resource-constrained devices, but the number of devices tested has so far been extremely limited [17]. Therefore, further empirical evaluation on using container technologies in IoT devices, particularly in the area of WMSN, is needed.

Security against various attacks is one of the basic requirements for IoT systems [19]. Also, when every object in our daily life is connected to the Internet, they must speak the same secure protocols [20]. Furthermore, since IoT systems invade personal spaces in many applications, such as home automation and surveillance, privacy is a significant concern in these systems [21]. Many of the security and privacy challenges are consequences of adding new device types to IoT systems that have not originally been designed to be networked. This has opened the potential for new vulnerabilities of which we have seen examples during the previous years, such as hacking into Internet-connected light bulbs, smart home automation systems and connected car systems. Therefore, IoT systems should ensure that the information exchanged in the network must be protected end-to-end [22]. The basic security elements such as confidentiality, authentication and freshness of secret keys between two communicating entities should be carefully maintained [23]. One of the major challenges in secure IoT systems is related to resource efficiency: the available hardware resources do not allow heavy cryptography on many IoT devices [1, 24]. This makes it necessary to develop lightweight security mechanisms to overcome the problem.

CONVINcE (Consumption OptimizatioN in VIdeo NEtworks) project has focused on optimizing end-to-end energy consumption of video delivery from the video content source to the video content consumer, including the nodes and networks in between. The authors of this chapter have focused on optimizing the energy efficiency of capturing video content in a video surveillance scenario utilizing WMSN networks. The rest of this chapter is organized as follows: Sects. 7.2 and 7.3 discuss the optimized hardware architectures and multi-tier network architectures optimized for WMSNs. Section 7.4 is devoted to energy-efficient management of WMSNs based on lightweight virtualization. Section 7.5 describes a lightweight key management scheme designed for WMSNs and analyses the differences in energy consumption between secure and non-secure video delivery. Finally, Sect. 7.6 concludes the chapter by summarizing the contributions, discussing their limitations, generalizability and significance, and explores the avenues for future work.

7.2 Optimized Hardware Architectures

Optimized hardware architectures are among the basic building blocks of energy-efficient IoT networking. By advanced power management, the consumption of today's WSN nodes (scalar sensors, actuators) has been reduced to levels enabling battery life counted in years. However, in off-the-shelf nodes capable of performing WMSN tasks, particularly capturing and streaming high-definition video, the power management technology is not on a good level.

7.2.1 Background

The literature on WMSN shows a variety of proposals on energy-efficient smart camera hardware architectures. Tavli et al. [25] present an overview of popular visual sensor platforms, including most popular ones Cyclops [26], MeshEye [27] and Panoptes [28].

The main design goal for Cyclops is low-power object detection in multimedia sensor nodes. To reduce the power consumption, Cyclops follows an on-demand clocking of resources. Cyclops uses ATMEL ATmega128L MCU. The camera module has a resolution of 352 × 288. For minimizing energy consumption, Cyclops has separate boards for sensing and networking. Cyclops uses CC1000 radio module with a throughput of 38.4 Kbps. The main design goal of MeshEye is to enhance in-node processing of a smart camera for distributed intelligence surveillance. MeshEye is a fully integrated node based on 32-bit Atmel AT91SAM7S microcontroller. MeshEye uses three image sensors. One of these image sensors has the resolution of 30 × 30 and it is used to continuously poll for moving objects in its proximity. When an object is detected, the region of interest is determined from stereo vision of the other two image sensors that have a resolution of 640 × 480. MeshEye has the CC2420 IEEE802.15.4 standard chip for communicating with other MeshEye nodes with a maximum throughput of 250 kbps. The main design goal of Panoptes is high-quality video surveillance over IEEE 802.11 network. It is based on a 32-bit StrongARM MCU and a camera module that can be set to resolutions 160 × 120, 320 × 240 and 640 × 480. The article also reviews other WSN platforms that can potentially be used as multimedia sensor platforms. These include: Mica2, Mica2Dot, MicaZ, Telos, Imote, Yale XYZ and Stargate.

7.2.2 sleepyCAM Prototype

Despite the several hardware proposals, including those revised above, to the best of our knowledge, not much has been done to capture and to deliver higher resolution video (above 640 × 480) at a reasonable power. Thus, we wanted to see if

we can achieve high-resolution video (full HD, 1920 × 1080) in a wireless multimedia sensor node and yet maintain a feasible battery life. To address that, we developed a low-power video camera prototype for WMSN, called sleepyCAM [18]. sleepyCAM was developed with following design principles:

- Use a single board computer, such as Raspberry Pi, as camera node.
- Turn it off when there is no motion (i.e. during the waiting time), and
- Find means to monitor the area under surveillance with the minimum power possible and wake up the camera node when motion is detected.

Raspberry Pi (RPi) is a popular choice for many computationally intensive IoT projects. The latest model RPi 3 B has BCM2837 chip (Quad core @1.2 GHz ARMv8 Cortex-A5), 1 GB RAM, Broadcom VideoCore IV GPU, 4xUSB 2.0, 40 I/O pins, Ethernet, Wi-Fi and BLE. Furthermore, the price is only 35$, making it an ideal platform for video-related tasks.

Despite all of these advanced features, RPi does not support Advanced Configuration and Power Interface (ACPI), and thus does not provide sleep modes. Hence, implementing low-power RPi-based camera nodes for energy-saving is challenging without having a means of controlling the power of RPi. Thus, as a first step, we design a power management mechanism for RPi. The objective is to shut down the RPi when there is no motion and wake it up when activity is detected in the area under surveillance. We did this by attaching a dedicated external low-power sensor node, *controller node*, to detect motion and to control the powering of an RPi. The resulting hybrid device—named *sleepyCAM*—can operate at power levels of either the controller or the camera node (RPi).

The controller node is implemented using highly energy-efficient Libelium Waspmote sensor platform. For movement detection and switching the RPi ON when needed, we use Parallax Rev. B PIR sensor and 10A 30VDC Songle relay. By disabling all unused ports, the controller is programmed to operate in low-power mode. Using an interrupt signal from the PIR sensor, the controller is alerted to activate the relay connected to one of its general-purpose digital I/O pins. The relay turns ON the RPi to record a video. RPi decides when to stop recording and eventually shuts down. An important step in the power management process is to detect a graceful shutdown of the RPi before deactivating the relay, in order to avoid memory card damage. For that, the RPi can be configured to signal the completion of shutdown process using one of its general purpose pins. The state change of this pin can be read by the controller so that it can deactivate the relay. Figure 7.1 presents the hardware architecture of the proposed power management system, and Fig. 7.2 presents the power management state machine.

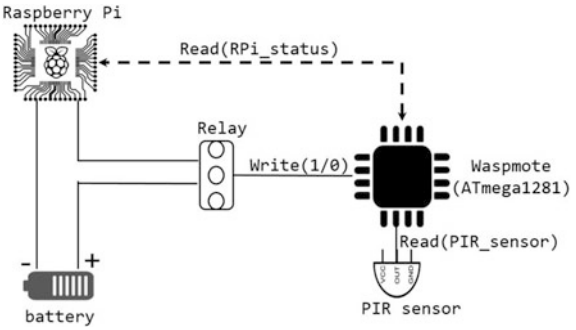


Fig. 7.1 sleepyCAM hardware architecture [29]

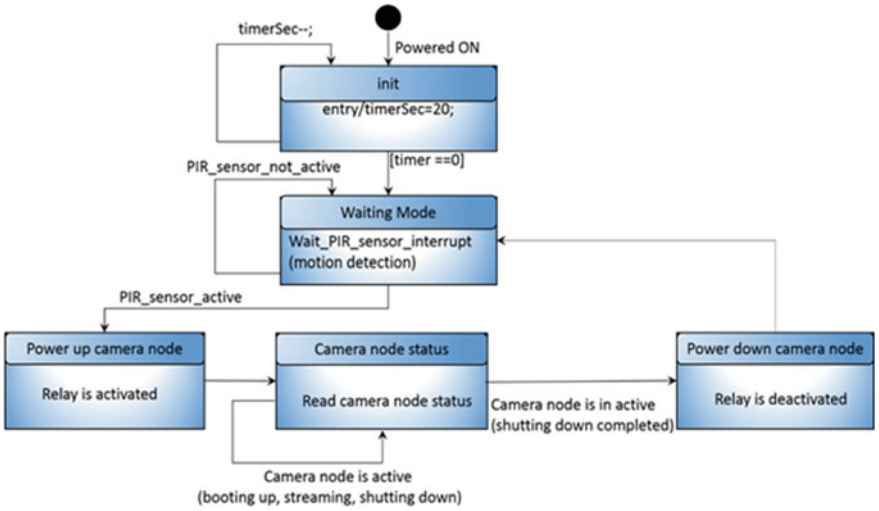


Fig. 7.2 State machine diagram of the power management

7.2.3 Power Consumption Analysis

7.2.3.1 Evaluation Setup

To evaluate the performance of sleepyCAM in video surveillance, we have built a real-life testbed and set up a baseline video surveillance system in addition to sleepyCAM prototype [30]. In the baseline system, we also use an RPi-based camera sensor node equipped with PIR sensor and a camera module. Figure 7.3 shows the sleepyCAM and the baseline systems side by side.

In the evaluation scenario, both setups detect motion using the PIR sensor, and when motion is detected, start capturing and streaming a video of 10 s to a remote laptop. The video data are transferred using netcat utility over Transport Control

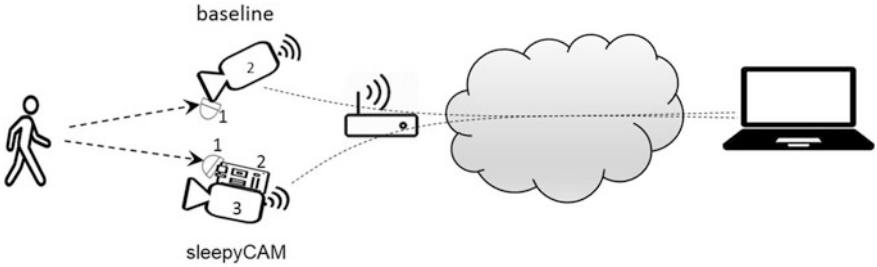


Fig. 7.3 Network architecture: baseline versus sleepyCAM [30]

Protocol (TCP). On the terminal side, the incoming video data are piped to mplayer and displayed with full HD (1920 × 1080) resolution.

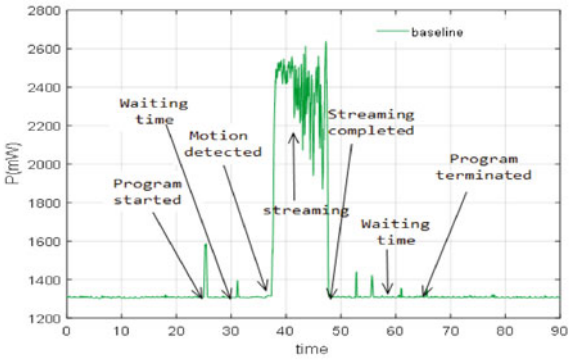
In the baseline setup, when motion is detected, the camera module is initialized and video streaming to a remote laptop is started. RPi is dedicated to both sensing and streaming tasks. Thus, in order to maintain motion detection capability, the camera node (RPi) needs to be continuously powered and active.

With sleepyCAM, the camera node (RPi) is powered off during idle (waiting) time. The controller node performs motion detection during idle time. When motion is detected, controller node wakes up the RPi, which then initializes the camera module and starts streaming video to a remote laptop.

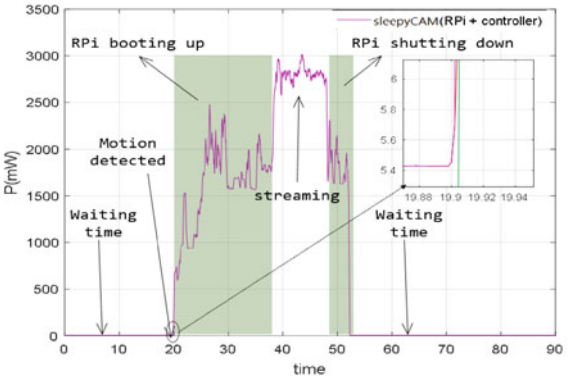
7.2.3.2 Power Consumption Analysis

Figure 7.4 presents the total power consumption of the baseline and sleepyCAM video surveillance setups. The idle (waiting time) power consumption of the baseline setup is 1 309 mW, whereas with sleepyCAM, it is only 5.43 mW, representing a decrease of around 99.6%. During video streaming, the power consumption of the baseline approach jumps to 2430 mW and that of sleepyCAM to 2682 mW, representing around 10.4% increase. This 10.4% increase is due to the overhead inflicted by the controller. When motion is detected, the controller has to keep the relay switch latched as long as the RPi is up and running.

In most surveillance scenarios, the time interval between events is in an order of magnitude longer than the duration of the event. In other words, the surveillance system spends most of its time waiting for an event and the video stream length when events occur is usually short. With the assumption, made in [24], that a multimedia sensor node spends more than 99% of its time in idle mode, over the time the 10.4% increase in power consumption during active state is relatively insignificant compared to the 99.6% power decrease during idle time. Table 7.1 compares the power consumption and the resolution of sleepyCAM node with existing video nodes suitable for video surveillance in WMSN networks. As it can be seen, sleepyCAM can provide full HD video with roughly similar power consumption profile as the existing systems providing VGA or lower resolution video.



(a) Baseline camera



(b) sleepyCAM

Fig. 7.4 Power consumption [30]

Table 7.1 Features of existing camera sensor systems and sleepyCAM

Sensor node	Active power	Idle power	Average power (99% idle time)	Resolution
Cyclops	110.1 mW	0.8 mW	1.9 mW	352 × 288
MeshEye	175.9 mW	1.8 mW	3.5 mW	640 × 480
Panoptes	5300 mW	58 mW	110.4 mW	640 × 480
sleepyCAM	2682 mW	5.4 mW	32.2 mW	1920 × 1080

7.3 Multi-tier Network Architectures

Multi-tier network architectures are among the most powerful energy-saving mechanisms for multimedia sensor networks that usually consist of heterogeneous devices. In these architectures, traditional scalar sensor nodes (i.e. sensor nodes capturing scalar data such as temperature) are typically located on lower tier, establishing an energy-efficient, always-on sensing environment. Higher capacity, more energy-consuming, multimedia sensor nodes (i.e. sensor nodes capable of capturing multimedia data such as video) are typically located on higher tiers,

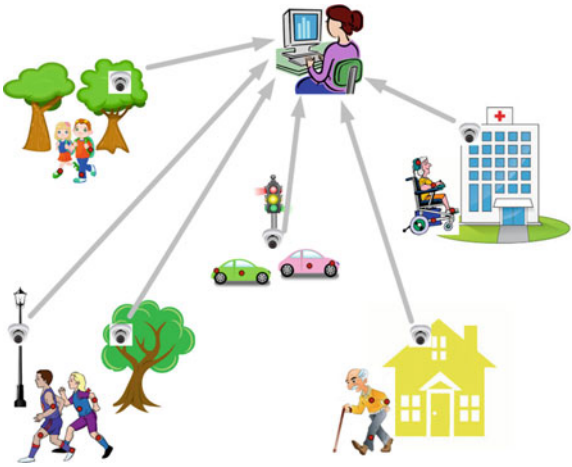


Fig. 7.5 WMSN application areas

where nodes are active only when needed. Figure 7.5 illustrates several application areas where traditional scalar and multimedia sensors work collaboratively for different purposes.

7.3.1 Background

During the past decade, several proposals for utilizing multi-tier network architectures for WMSN have been presented. Akyildiz et al. [2] have proposed a reference architecture for WMSNs with three feasible options: (1) single-tier network with homogeneous multimedia sensors, (2) single-tier clustered architecture of heterogeneous sensor nodes and (3) multi-tier network with heterogeneous sensor nodes. Among these, the multi-tier approach is seen as the most appropriate architecture for video surveillance applications. Furthermore, several other proposals [26, 31, 32, 33, 34] have suggested the use of heterogeneous sensor nodes in multi-tier architectures to provide benefits over homogeneous deployment of camera nodes, in terms of overall cost and energy consumption.

7.3.2 Multi-tier sleepyCAM Prototype

We have extended the sleepyCAM prototype to implement a multi-tier network architecture in order to realize an even more energy-efficient video surveillance scenario [35]. For that, we modularized the surveillance scenario into motion detection and video streaming tasks. The motion detection task is implemented on separate low-power Waspmote sensor nodes equipped with PIR sensors at the front end of the network (tier-1) and the video surveillance task is implemented on

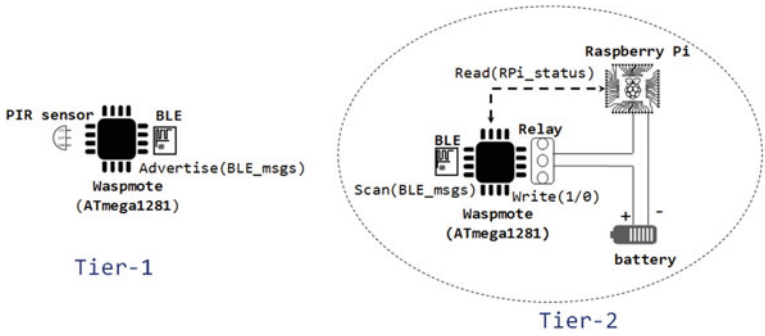


Fig. 7.6 Multi-tier sleepyCAM hardware architecture [35]

sleepyCAM (tier-2) node. As described in Sect. 7.2.2, sleepyCAM includes the high-performance RPi-based video sensor node, combined with Waspmote-based controller node. The difference in this setup is that the PIR sensor, previously mounted on the controller of sleepyCAM, is now mounted on the RPi itself for detecting the continuity of movement during the video capture. The controller node in the multi-tier architecture is used to manage the powering of the RPi and to receive message from tier-1 nodes about detection of motion. The message exchange between tier-1 motion sensors and tier-2 controller node is conducted over Bluetooth Low Energy (BLE) wireless link. For this we mounted BLE radio modules on tier-1 motion sensors and tier-2 controller node of sleepyCAM. When motion is detected by a tier-1 motion sensor (master Waspmote), it starts advertising BLE message to tier-2 controller node (slave Waspmote). The hardware architecture of the Multi-tier sleepyCAM is presented in Fig. 7.6.

7.3.3 Power Consumption Analysis

To measure the performance of the multi-tier architecture, we used the single-tier setup described in Sect. 7.2.3 as a baseline benchmark.

Figure 7.7 presents the power consumption transient of the motion detection nodes at tier-1. These nodes enter the waiting (idle) mode soon after they are turned on and initialization is completed. In idle mode, most of the circuitries including the BLE module are off and the MCU of this node is in deep sleep mode until triggered by an external interrupt from the PIR sensor that is mounted on its digital pins. When motion is detected, the device activates its BLE module and advertises alarm messages to tier-2 controller Waspmote for about 3 s. The idle (waiting time) power consumption of master Waspmote is 0.5 mW. The mean power consumption during this period is 76.4 mW.

In Fig. 7.8, we present the power consumption transients of the controller node and the camera node. In addition, the figure shows the combined power

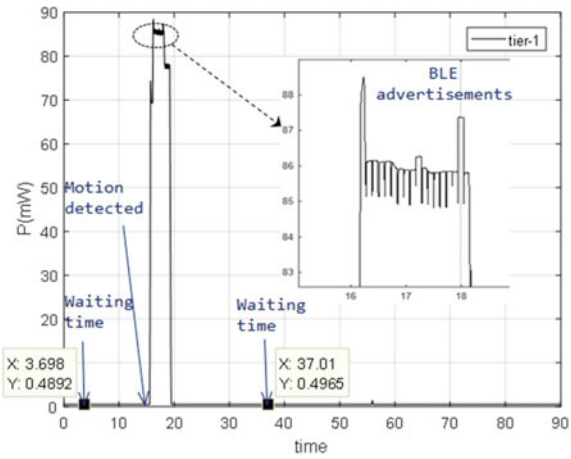


Fig. 7.7 Power consumption of a tier-1 motion detection node

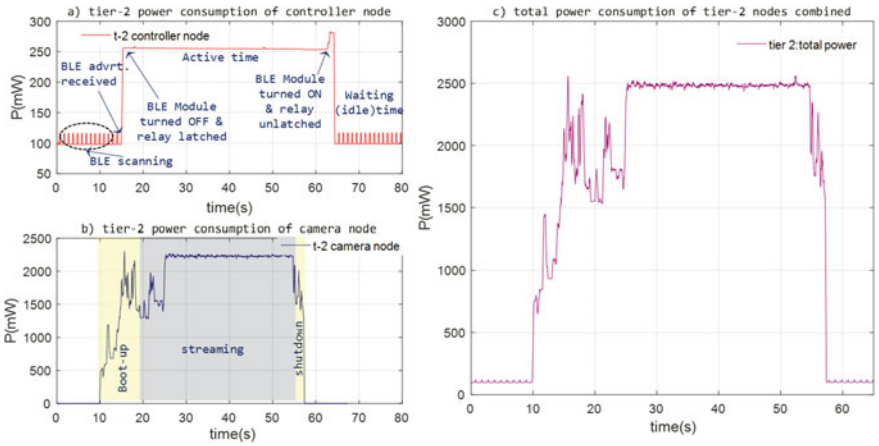


Fig. 7.8 Power consumption of tier-2 controller and camera nodes

consumption of the tier-2 node, including both the controller and the camera node. In idle mode, the controller node is scanning BLE advertisements transmitted from known tier-1 Waspmote nodes. When BLE messages are received, this node turns off its BLE module and latches relay switch to power-up the camera node. During idle time, the RPi camera node is completely powered off and thus, the mean power of tier-2 as a system equals to the mean power of the controller Waspmote during its idle state. During the active state, the mean power of tier-2 as a system is the sum of the active mean power of RPi and slave Waspmote.

The idle power consumption of single tier is 1309 mW, and for the multi-tier, it is 101.2 mW (92.3% reduction). During active (streaming) state, the power

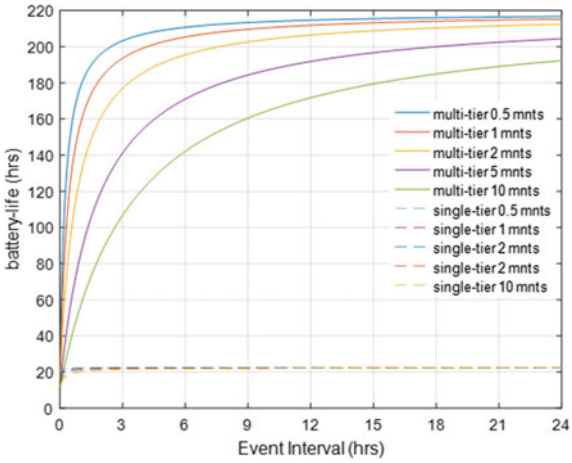


Fig. 7.9 Battery life of multi-tier versus single tier for different durations of video streaming [35]

consumption of single tier is 2251 mW and that of multi-tier is 2484.2 mW (10.4% power increase).

We also estimated the battery life of multi-tier WMSN based on the empirical measurement results on the basis of one event cycle. Accordingly, the simulation results for the battery life, as illustrated in Fig. 7.9, indicate that multi-tier WMSN can provide up to 194 h longer battery life (about 870% more) than single-tier architecture.

7.4 Lightweight Virtualization

In a real-life environment, hundreds or even thousands of cameras may be deployed in scenarios such as smart traffic or large surveillance systems. The camera nodes may occasionally need software upgrades due to software bugs, security vulnerabilities or changes in the infrastructure. Managing a large number of devices this way may turn out to be a challenging task, especially since the cameras are often offline in order to save energy.

7.4.1 Background

One way of resolving the management issues is to implement the software of the camera nodes to support, e.g., remote firmware upgrades and configuration management. This is a reasonable approach with resource-constrained IoT devices. However, in our case, we are using a relatively powerful platform, Raspberry Pi, as the camera node. Such devices are capable of running software virtualization that can be utilized for upgrading the firmware of camera nodes and managing their

configuration in a more generic and hardware agnostic way. If the support for some other platforms besides RPi is needed, the firmware can be reused and recompiled for the other platform. The management, or *orchestration*, of the virtualized camera software can be handled in a centralized way. The orchestrator managing the deployment of virtualized software can deploy the software in a flexible way to the camera nodes. When a camera node is turned on, it can request the orchestrator for the latest camera software with the latest updates. If needed, multiple software instances can be run on a single camera node. Furthermore, since the reprogramming of the camera software is easier, it is beneficial to offload computationally intensive tasks from the RPi to a cloud.

Another challenge related to IoT scenarios involving large numbers of devices is the addressing [8]. While private address realms can be employed with NATs in order to mitigate the shortage of IPv4 addresses, they constrain connectivity towards the devices. Using IPv6 solves many connectivity issues associated with IPv4 NATs, but current virtualization platforms have some limitations with IPv6 [8].

Power consumption is yet another challenge. Utilizing a platform such as Waspmote is very power efficient, but computational and storage capabilities are quite limited. A platform such as Raspberry Pi has better resource capacity than, e.g. WaspMote, consumes less energy than PCs and supports lightweight virtualization such as Docker.

A number of different virtualization platforms exist. Below we present a few open-source solutions. KVM virtual machine offers hypervisor-based virtualization, supporting, e.g. multi-tenancy. The drawback of hypervised virtual machines is that they tend to be rather big especially in terms of memory and their boot-up tend to be slow. Linux containers, for example, are a popular alternative for implementing virtualizing software, with Docker as one of the most used implementations. In contrast to hypervised virtual machines, Docker containers usually consume less memory than hypervised virtual machines and their boot-up time is considerably reduced [36]. Currently, Docker containers offer the best trade-offs as the virtualization platform. In addition, Docker has, at the time of writing, the best support for the ARM-based Raspberry platform. The major drawback of Docker is the unfinished multi-tenancy support, which is still a work-in-progress issue for the open-source version of Docker. However, projects such as Hypernetes are trying to fill in this gap by combining the best from the hypervisor and container technologies. The energy overhead of Docker, compared to Linux without virtualization is almost insignificant on the Raspberry Pi platform [17]. A promising emerging technology called Unikernels (aka library operating systems) provides the fastest boot-up time and smallest memory overhead. The technology is, at the moment, immature and porting of many existing libraries and available software would require a lot of effort. However, Unikernels may be worth considering in the future.

Docker has a number of available tools for container orchestration, including Docker Swarm and Kubernetes. Docker Swarm is available in two flavours: standalone and swarm mode. Unfortunately, swarm mode supports only Docker's built-in network modes, where IPv6 requires administrative privileges to the underlying network (i.e. routable IPv6 prefix), which is not always possible.

Swarm mode does not support custom Docker network plug-ins, so this mode cannot be used in our use case. The standalone mode is implemented in a component called Docker Swarm standalone, which serves the standard Docker API, and thus everything that works with Docker works also with the standalone component. Therefore, standalone mode supports also custom Docker network plug-ins. In contrast to the swarm mode, standalone mode is no longer actively maintained and is therefore somewhat outdated in functionality compared to the swarm mode. In addition, the standalone version has some further limitations compared to the swarm mode. For example, the standalone version does not have an option to leave the cluster.

Our implementation, described in the next section, utilizes the Docker Swarm standalone, for which have implemented our own IPv6 support. Our IPv6 module eliminates the need to have a dedicated IPv6 prefix for the underlying host (unlike in the swarm mode) by using a sub-prefix assigned for the host. The module supports IPv6 in two modes: NAT and proxy. In the proxy mode, the underlying host uses neighbour discovery proxy to enable IPv6 connectivity for the cluster (i.e. camera) nodes. Since only the proxy option accepts inbound connections to the cluster nodes, we chose to use it for the prototype.

7.4.2 *Virtualized Multi-tier sleepyCAM Prototype*

The orchestrated (or managed) version of the sleepyCAM-based video surveillance system that was introduced in Sects. 7.2 and 7.3 contains a new node called 'orchestrator' implemented on the RPi platform. Figure 7.10 illustrates the setup. The purpose of the orchestrator is to support dynamic reconfigurations of camera nodes. The camera software is running inside a Linux container instance that can be easily replaced with another, and it is possible to run multiple separate

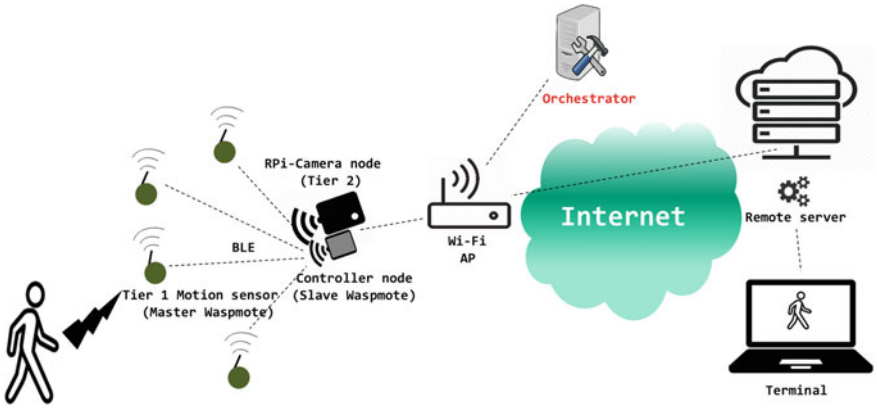


Fig. 7.10 Orchestrated sleepyCAM

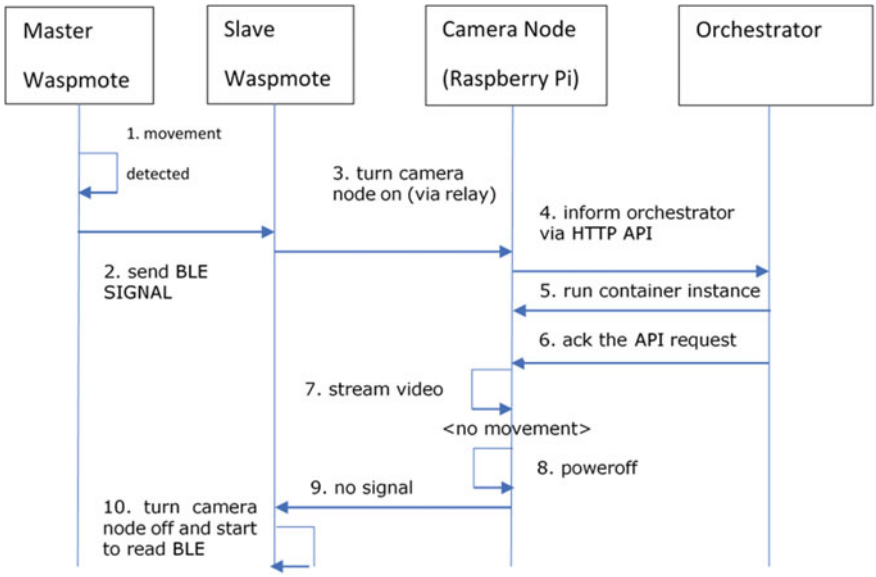


Fig. 7.11 Orchestrated sleepyCAM—sequence diagram

containers in the camera node if needed for extension purposes. The prototype uses Docker container technology, and the orchestrator uses Docker swarm APIs to control the Docker instances running in the camera node. The orchestrator is crafted specifically for CONVINcE project and includes a special feature typically missing from many orchestrator implementations. The camera node is sleeping most of the time, which means that the orchestrator cannot manage it periodically. Instead, the camera node informs the orchestrator when it is available for management.

The flow diagram of Fig. 7.11 illustrates the orchestrated video surveillance implementation based on multi-tier sleepyCAM. The scenario begins with the low-power master Waspmote (tier-1 motion sensor node) detecting some movement with its PIR sensor in step 1. The master Waspmote sends a BLE signal to slave Waspmote (sleepyCAM controller node) in step 2. In step 3, the controller node turns on the camera node (RPi) using an electronic relay.

In step 4, the camera node informs the orchestrator that it is now available. The orchestrator node consists of two components: orchestrator manager and orchestrator engine. The purpose of the orchestrator manager is to select a suitable service for video streaming and provide the necessary configuration for this service to operate in the node in which the orchestrator manager selects to deploy the camera service to. In the current prototype, the service is always deployed to the camera node itself due to lack of other nodes. The orchestrator engine implements the required functionality to request nodes to create new camera service instances.

In step 5, the orchestrator engine instructs the camera node to start a Docker instance suitable for video streaming. When the orchestrator engine has completed

the request, the orchestrator manager sends a response to the HTTP request the camera node sent in step 6. The Docker instance in the camera node then starts streaming video to a server specified by the orchestrator in step 7. The camera node is equipped also with a PIR sensor which is available to the Docker instance using the GPIO ports of the RPi. When the Docker instance detects no motion for certain period of time (as specified by the orchestrator), it stops streaming the video and instructs the host system (using an HTTP API call) to power off in step 8. In step 9, the controller node detects that the camera node has turned itself off, so it turns off the electronic relay in order to achieve further power savings in step 10. The controller node also starts to read BLE signals again from the motion detection node and then the operation cycle repeats.

As part of the power off cycle in step 8, the camera node could notify the orchestrator that the node is being shut down. The purpose of this notification would be to remove created services. However, because in the current implementation, the service is always created in the camera node itself, there is no need to remove instances from other nodes. In addition, Docker Swarm standalone has no functionality to leave a cluster and, therefore, the camera node does not inform the Swarm manager when it shuts down. The Swarm manager does not, therefore, detect that the node is shut down until the registration of the node in the cluster times out.

7.4.3 Power Consumption Analysis

Based on our initial evaluation, Docker containers mainly add fixed overhead to the energy consumption due to start-up delay of roughly 20 s and the shutdown delay of roughly 15 s before the camera node can be powered off. Based on our rough estimate, the start-up and shutdown procedures double the fixed costs. However, once Docker services are started, the power consumption overhead of Docker-based virtualization is negligible. According to our measurements, the difference in power consumption is below 2 percent in favour of Docker version. It should be noted that the prototype has not been optimized, and it is possible to optimize the start-up and boot procedures by fine-tuning the underlying Linux system. Our interpretation of the results is that Docker containers are suitable for virtualizing video surveillance applications where the video is being streamed for longer time periods (in the range of several minutes or hours). For shorter videos (minute or less) or just for capturing individual images, other virtualization technologies (such as ‘Unikernels’) should be considered once they become more mature. The measurement results are visualized in Fig. 7.12.

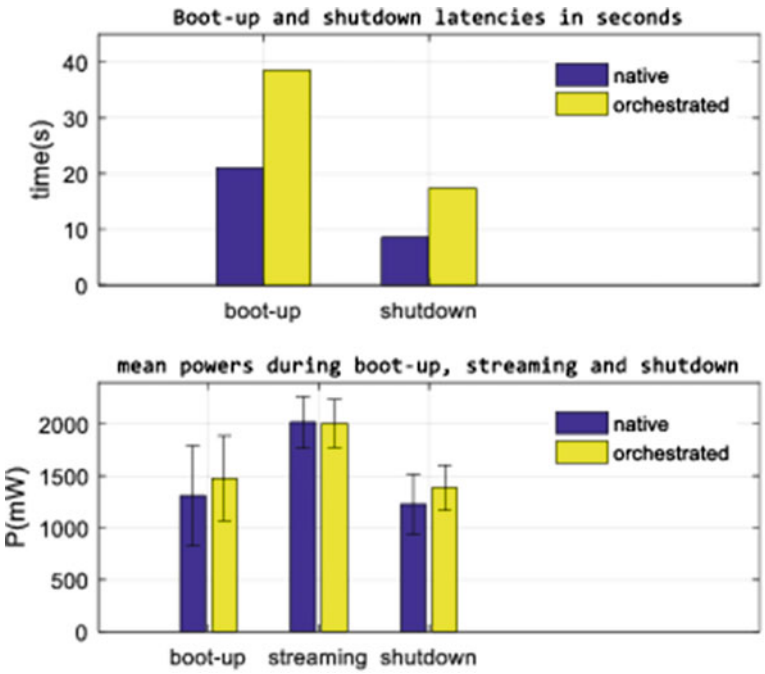


Fig. 7.12 Latencies and power consumption between native and orchestrated sleepyCAM during different phases of camera node operation

7.5 Lightweight Security

Security solutions, particularly cryptography operations, are computationally intensive, and thus, energy-hungry. This is emphasized with sensor and multimedia sensor nodes that have less computational and energy capacity compared to other device types, such as personal computers, tablets and smartphones. Therefore, it is important to exploit the most energy-efficient security solutions at the terminal and sensor node side.

7.5.1 Motivation

Lightweight key management and secure group communications are centric components in establishing secure connections for internode communication in WMSNs [37, 38]. Key management is a major prerequisite to construct secure communication channels among network devices for both unicast and multicast multimedia sensor networking scenarios. The resource-consuming cryptographic operations of the key establishment, revocation and distribution can be too

computationally expensive to perform by most devices in WMSNs. This is problematic due to the constraints of these devices concerning computational and battery capacity. The key establishment occurs only at the initialization phase of a secure communication channel. Later on, the key can be reused until there is need for rekeying. Therefore, a lengthy key establishment process, such as few seconds, is still acceptable as long as it does not occur too frequently (e.g. applications where reliability is more critical than latency).

In [38], a real-time dynamic key management (RDKM), a self-adjusting binary search tree-based rekey management is presented. It can efficiently enhance the network security and survivability in low-energy adaptive clustering hierarchy (LEACH) type protocol. Compared to other LEACH security solutions, this approach offers many advantages, such as real-time rekey mechanism and dynamic architecture for generating new keys and making the dynamic key management feasible without any overhead. The mechanism can efficiently protect the network against attacks from eavesdropping or captured nodes compromise and address challenging security issues of runtime in WMSNs.

The most consistent candidates that have been currently proposed to establish E2E secured communications among the IoT devices are datagram transport layer security (DTLS) handshake, Internet key exchange (IKEv2) scheme and host identity protocol diet exchange (HIPDEX) protocol [20]. All of these require key agreements with the asymmetric cryptographic primitives that have the variants of Diffie–Hellman (DH) protocol.

According to the lightweight collaborative key establishment scheme presented in [39], a constrained device may delegate its heavy cryptographic load to less constrained nodes in the neighbourhood exploiting the spatial heterogeneity of IoT environment. While initiating a secure E2E connection between two unknown nodes in distinctive networks, they exploit one set of intermediary nodes as proxies in order to support the key establishment process. However, this would not be practical in the actual scenarios, since both end nodes (that are completely unknown to each other) may not have securely pre-established communication links with the common proxies.

7.5.2 Proxy-Based Key Management

7.5.2.1 Collaborative HIP (CHIP) Prototype

In our experiment, we consider an application, presented in Fig. 7.13, where scalar sensors are used to monitor an elderly person's health conditions [40]. The sensors can be integrated into mobile personal devices (smartphones or smartwatches). Whenever a scalar medical sensor receives health critical data, they invoke the visual

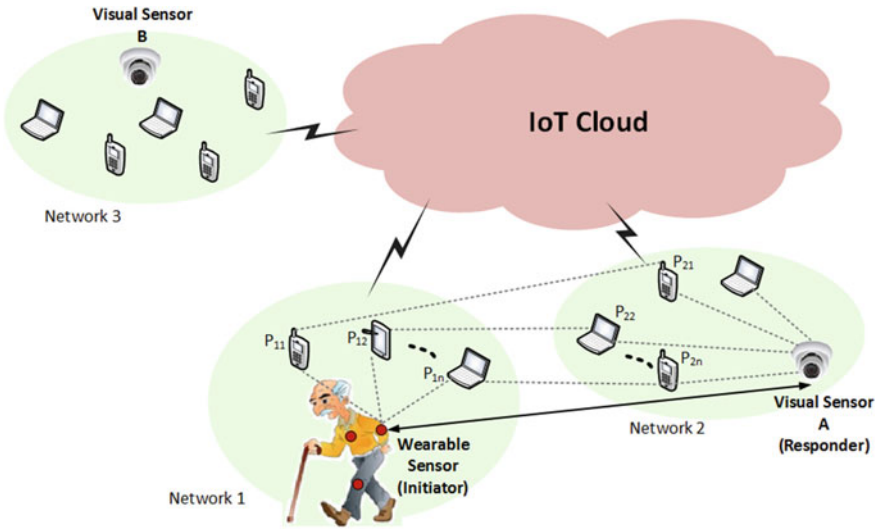


Fig. 7.13 Network system architecture [40]

sensors (camera nodes) in their closest proximity. Therefore, as shown in Fig. 7.13, we consider E2E key establishment between two highly resource-constrained devices (e.g. a scalar sensor and visual sensor) located in separate local networks [40]. Each node is capable of forming logical networks with a set of resource-rich devices (e.g. smartphones, PDAs and laptops) in their proximity, performing as proxies. This creates two logical local networks (networks 1 and 2) and the proxies collaboratively support the two nodes for computing the shared secret key. The proxies in networks 1 and 2 can securely communicate with each other since they have enough resources and well-known techniques can be used. When an elderly person moves beyond the visual sensor A, the wearable sensor has to initiate a new connection with another visual sensor B, which is located in network 3.

Host identity protocol (HIP) [41] is an IETF standard that establishes secure signalling channels, inherently supporting node mobility and multihoming. The lighter version HIP diet exchange (HIP DEX) [42] is introduced for WMSNs, due to resource-consuming cryptographic operations in the original HIP base exchange (HIP BEX) [41]. However, the exploitation of static DH keying materials in HIP DEX may not be very pragmatic for the mobility and scalability requirements of IoT because they cannot always exchange the keys for every secure link establishment. Therefore, we propose a collaborative HIP (CHIP) solution with efficient key establishment to provide E2E secure connectivity among resource constrained devices in IoT. In CHIP, a resource-constrained device delegates demanding cryptographic operations of HIP protocol to devices in the neighbourhood with more computational resources, which act as proxies.

7.5.2.2 Power Consumption Analysis

In order to evaluate the performance and quantify the energy efficiency of the key establishment component of the suggested CHIP protocol at the responder's side, we implemented the corresponding cryptographic operations on Libelium Waspmote platform using Waspmote cryptographic libraries. Waspmote has an Atmega1281 microcontroller running at 8 MHz with 8 KB SRAM, 4 KB EEPROM and 128 KB Flash memory. We measure the execution time (t) for individual cryptographic operations and calculate the computation energy cost using formula $U \times I \times t$ based on the execution time (t), the nominal voltage (U) and the current in active mode (I) on Waspmote sensors. Based on the data sheet, we select I = 9 mA and U = 3 V. The results are shown in Fig. 7.14.

As shown in Fig. 7.14, while increasing the number of involved proxies, the total energy cost for the key establishment component of CHIP grows exponentially. The suggested key establishment phase outperforms that of HIP DEX and HIP BEX when the cooperative proxies on one side are kept below 40. However, the involvement of a very low number of proxies may also create a higher probability for them to communicate among the group of proxies and reconstruct the secret key. Therefore, it is important to maintain the balance between the number of involved proxies and the risk they tend to cooperate.

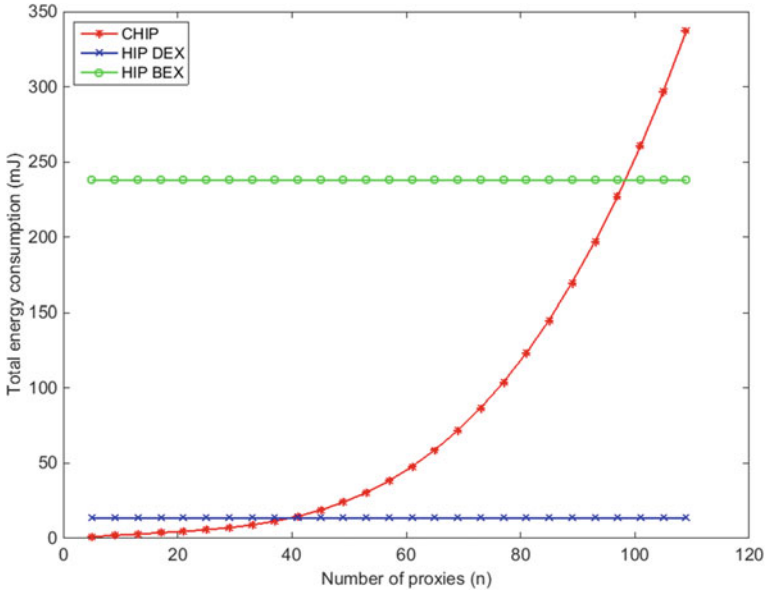


Fig. 7.14 Variation of total energy cost for key establishment with collaborating proxies (n) [40]

7.5.3 *Power Consumption Analysis of Secure Media Transport*

Furthermore, in order to see how using secure transport protocols affect the energy efficiency of video surveillance networks, we did two sets of measurements. The first one focused on comparing encrypted FTPS protocol versus unencrypted FTP protocol when sending still images from Waspmote-based camera sensor nodes to a server. In the second measurement, we compared encrypted HTTPS protocol versus unencrypted HTTP protocol when streaming security camera video from Raspberry Pi-based camera sensor nodes to a server. For measurement purposes, we used the multi-tier WMSN architecture similar to that presented in Sect. 7.3, as illustrated in Fig. 7.15. The multi-tier operation enables keeping the camera sensor nodes in a sleep mode for most of the time and waking them up only when activity is detected.

7.5.3.1 FTPS Versus FTP in Transferring Still Images

In the first stage, we measured the power consumption for secure and non-secure video data transmission over 3G network by Waspmote multimedia sensor in multi-tier WMSN architecture [24]. The BLE scalar sensor is used to wake up the multimedia sensor. This particular WMSN system is used for comparing encrypted FTPS protocol (secure) against unencrypted FTP protocol (non-secure) when sending still images from Waspmote-based camera sensor nodes to the server. The measurement results are illustrated in Fig. 7.16. The main observation is that using secure protocols in transferring still images does not inflict significant increase in the total energy consumption. The gain in energy consumption is around 2%, and the total time for video upload is slightly higher (around 17 versus 15 s).

7.5.3.2 HTTPS Versus HTTP in Streaming Video

In the second stage, we compared the energy consumption of the encrypted HTTPS protocol versus unencrypted HTTP protocol when streaming surveillance camera

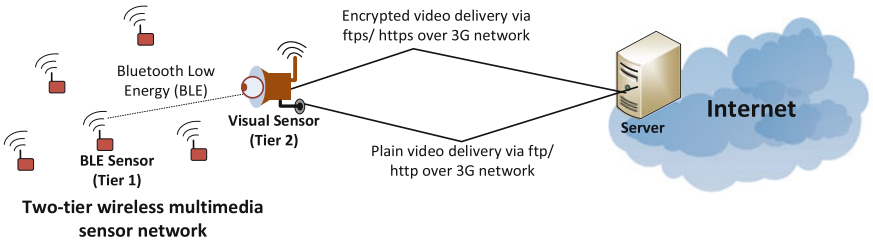


Fig. 7.15 Network architecture with multi-tier WMSN

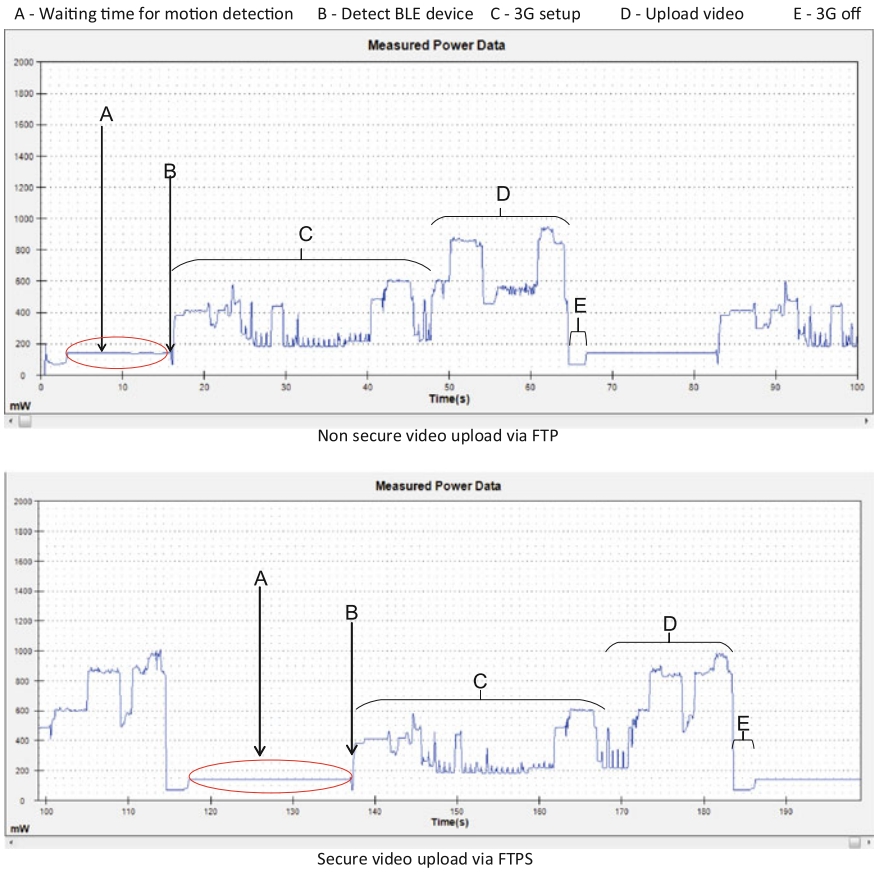


Fig. 7.16 Energy consumption of secure and non-secure video upload

video from Raspberry Pi-based camera sensor nodes to a server. The measurement setup is almost similar as in stage 1 except the replacement of Raspberry Pi platform for tier-2 visual sensor. The Raspberry Pi 2, model B V1.1, was used along with the Pi camera. Real-time HTTP and HTTPS streaming was performed with the native uv4 l-server module. The average power consumptions for HTTP and HTTPS streaming were measured for different video formats such as MJPEG, JPEG still, H264 and YUV streaming.

Figure 7.17 shows the power consumptions for HTTP and HTTPS streaming for different video formats and resolutions. The average power consumption for all four video formats decreases when the resolution decreases. YUV video exhibits the lowest average power consumption for both HTTP and HTTPS streaming. Compared with the other three video formats, YUV video power consumption is relatively low. In all four video formats, using HTTPS instead of HTTP in

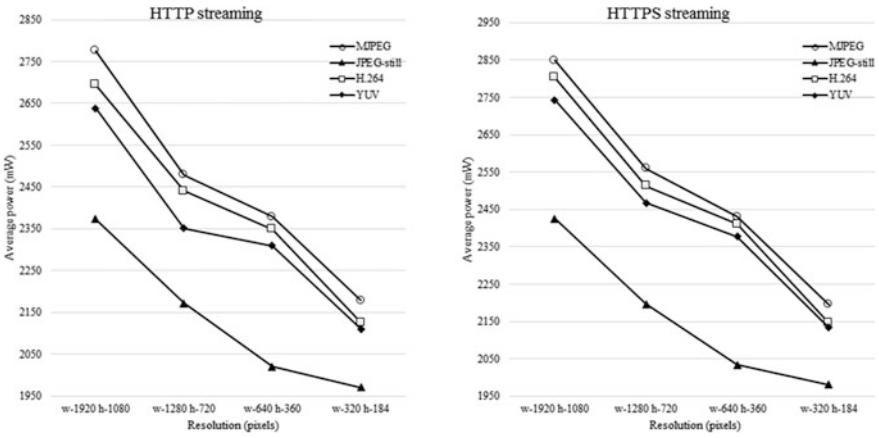


Fig. 7.17 Average power consumption for HTTP/HTTPS streaming

streaming video increased the total energy consumption by around 1–5%. Again, the main observation is that use of secure protocol does not inflict significant increase in the total energy consumption.

7.6 Conclusion

The book chapter has presented methods for improving the energy efficiency of wireless multimedia sensor networks. These methods included energy-efficient hardware architectures, energy-optimized management of network topologies as well as lightweight virtualization technologies and lightweight security solutions for preserving system manageability and security. The optimization methods have been evaluated using real-life prototype implementations. The presented methods provide a valuable toolbox for designing and implementing energy-efficient, remotely manageable and secure wireless multimedia sensor networks, particularly from the viewpoint of video surveillance systems.

Acknowledgements This work was supported by TEKES and by the European Celtic-Plus Project CONVINcE, which was partially funded by Finland, France, Sweden and Turkey.

References

1. Porambage P, Schmitt C, Kumar P, Gurtov A, Ylianttila M (2014) PAuthKey: a pervasive authentication protocol and key establishment scheme for wireless sensor networks in distributed IoT applications. Int J Distrib Sens Netw 10(7)
2. Akyildiz IF, Melodia T, Chowdhury KR (2007) A survey on wireless multimedia sensor networks. Comput Netw 51(4):921–960

3. Aasha Nandhini S, Radha S, Kishore R (2017) Efficient compressed sensing based object detection system for video surveillance application in WMSN. Multimed Tools Appl 60(C): 175–192
4. Margi CB, Petkov V, Obraczka K, Manduchi R (2006) Characterizing energy consumption in a visual sensor network testbed. In: 2nd international conference Testbeds and research infrastructures for the development of networks and communities, Barcelona, Spain, 1–3 March 2006
5. Garcia-Sanchez AJ, Garcia-Sanchez F, Garcia-Haro J (2011) Wireless sensor network deployment for integrating video-surveillance and data-monitoring in precision agriculture over distributed crops. Comput Electron Agric 75(2):288–303
6. Misra S, Mali G, Mondal A (2015) Distributed topology management for wireless multimedia sensor networks: exploiting connectivity and cooperation. Int J Commun Syst 28(7):1367–1386
7. Hossain M, Ahmed D (2012) Virtual Caregiver: an ambient-aware elderly monitoring system. IEEE Trans Inf Technol Biomed 16(6):1024–1031
8. Al-Fuqaha A, Guizani M, Mohammadi M, Aledhari M, Ayyash M (2015) Internet of Things: a survey on enabling technologies, protocols, and applications. IEEE Commun Surv Tutor 17(4):2347–2376
9. Prabhu B, Gajendran E (2016) An investigation on remote controlled tank using sensors for defense applications. Int J Innov Sci Eng 3:6p
10. CISCO (2017) Cisco Visual Networking Index: VNI Forecast 2021. https://www.cisco.com/c/en/us/solutions/service-provider/visual-networking-index-vni/index.html?dtid=osscdc000283
11. Chiasserini CF, Magli E (2012) Energy consumption and image quality in wireless video-surveillance networks. In: 13th IEEE international symposium on personal, indoor and mobile radio communications, Lisbon, Portugal, 15–18 September 2002
12. Yang M, Wang D, Bourbakis N (2013) Optimization of power allocation in multimedia wireless sensor networks. Int J Monit Surveill Technol Res 1(1):13
13. Zhang Y, Shakhsheer Y, Barth A, Powell HC, Ridenour SA, Hanson MA, Lach J, Calhoun BH (2011) Energy efficient design for body sensor nodes. Low Power Electron Appl 1(1):109–130
14. Harjula E (2016) Energy-efficient peer-to-peer networking for constrained-capacity mobile environments. Doctoral dissertation, University of Oulu, Acta Universitatis Ouluensis
15. Rault T, Bouabdallah A, Challal Y (2014) energy efficiency in wireless sensor networks: a top-down survey. Comput Netw 67:104–122
16. Bhatt R, Datta R (2016) A two-tier strategy for priority based critical event surveillance with wireless multimedia sensors. Wirel Netw 22(1):267–284
17. Morabito R (2017) Virtualization on internet of things edge devices with container technologies: a performance evaluation. IEEE Access 5:8835–8850
18. Celesti D, Mulfari M, Fazio M, Villari M, Puliafito A (2016) Exploring container virtualization in IoT clouds. In: IEEE international conference on smart computing, St. Louis, MO, USA, 18–20 May 2017
19. Roman R, Najera P, Lopez J (2011) Securing the internet of things. Computer 44(9):51–58
20. Keoh SL, Kumar S, Tschofenig H (2014) Securing The Internet of Things: a standardization perspective. IEEE Internet of Things J 1(3):265–275
21. Sicari S, Rizzardi A, Grieco LA, Coen-Porisini A (2015) Security, Privacy and Trust in Internet of Things: The Road Ahead. Comput Netw 76:146–164
22. Walters JP, Liang Z, Shi W, Chaudhary V (2007) Wireless sensor network security: a survey. In: Security in distributed, grid, mobile, and pervasive computing, vol 1, p 367
23. Brachmann M, Keoh SL, Morchon O, Kumar S (2012) End-to-End Transport Security in the IP-based Internet of Things. In: 21st international conference on computer communications and networks, Munich, Germany, 30 July–2 August 2012
24. Porambage P, Heikkinen A, Harjula E, Gurtov A, Ylianttila M (2016) Quantitative power consumption analysis of a multi-tier wireless multimedia sensor network. In: 22th European wireless conference, Oulu, Finland, 18–20 May 2016

25. Tavli B, Bicakci K, Zilan R, Barcelo-Ordinas JM (2012) A survey of visual sensor network platforms. Multimed Tools Appl 60(3):689–726
26. Rahimi M, Baer R, Iroezi OI, Garcia JC, Warrior J, Estrin D, Srivastava M (2005) Cyclops: in Situ image sensing and interpretation in wireless sensor networks. In: 3rd international conference on embedded networked sensor systems, San Diego, CA, USA, 2–4 November 2005
27. Hengstler S, Prashanth D, Fong S, Aghajan H (2007) MeshEye: a hybrid-resolution smart camera mote for applications in distributed intelligent surveillance. In: 6th international conference on information processing in sensor networks, Cambridge, MA, USA, 25–27 April 2007
28. Feng WC, Kaiser E, Feng WC, Baillif ML (2005) Panoptes: scalable low-power video sensor networking technologies. ACM Trans Multimed Comput Commun Appl 1(2):151–167
29. Mekonnen T, Harjula E, Koskela T, Ylianttila M (2017) sleepyCAM: power management mechanism for wireless video-surveillance cameras. In: Workshops in IEEE international conference on communications, Paris, France, 21–25 May 2017
30. Mekonnen T, Harjula E, Heikkinen A, Koskela T, Ylianttila M (2017) Energy efficient event driven video streaming surveillance using sleepyCAM. In: 17th IEEE international conference on computer and information technology, Helsinki, Finland, 21–23 August 2017
31. Jelicic V, Magno M, Brunelli D, Bilas V, Benini L (2014) Benefits of wake-up radio in energy-efficient multimodal surveillance wireless sensor network. IEEE Sens J 14(9):3210–3220
32. Kulkarni P, Ganesan D, Shenoy P, Lu Q (2005) SensEye: a multi-tier camera sensor network. In: 13th annual ACM international conference on Multimedia, Singapore, 6–12 November 2005
33. Lee JJ, Krishnamachari B, Kuo CC (2004) Impact of heterogeneous deployment on lifetime sensing coverage in sensor networks. In: 1st IEEE communications society conference on sensor and Ad Hoc communications and networks, Santa Clara, CA, USA, 4–7 October 2004
34. He T, Krishnamurthy S, Luo L, Yan T, Gu L, Stoleru R, Ab-delzaher TF (2006) VigilNet: an integrated sensor network system for energy-efficient surveillance. ACM Trans Sens Netw 2(1): 1–38
35. Mekonnen T, Porambage P, Harjula E, Ylianttila M (2017) Energy consumption analysis of high quality multi-tier wireless multimedia sensor network. IEEE Access 5:15848–15858
36. Kjällman J, Komu M, Kauppinen T (2016) Power aware media delivery platform based on containers. In: 19th international ICIN conference—innovations in clouds, internet and networks, Paris, France, 1–3 March 2016
37. Roman R, Alcaraz C, Lopez J, Sklavos N (2011) Key management systems for sensor networks in the context of the Internet of Things. Comput Electron Eng 37(2):147–159
38. Zhang Y, Li X, Yang J, Liu Y, Xiong N Vasilakos AV (2013) A real-time dynamic key management for hierarchical wireless multimedia sensor network. Multimed Tools Appl 67(1): 97–117
39. Winkler T. Rinner B (2014) security and privacy protection in visual sensor networks: a survey. ACM Comput Surv 47(1):2:1–2:42
40. Porambage P, Braeken A, Kumar P, Gurtov A, Ylianttila M (2017) CHIP: collaborative host identity protocol with efficient key establishment for constrained devices in internet of things. Wirel Pers Commun 96(1):421–440
41. Host Identity Protocol (HIP) (2008) RFC 5201, IETF, 2008
42. Moskowitz R, Hummen R (2016) HIP Diet EXchange (DEX), Expired Internet-Draft (individual). https://datatracker.ietf.org/doc/draft-moskowitz-hip-dex/

Chapter 8
Energy-Efficient Routing in SDN-Based Access Networks

Siwar Ben Hadj Said and Alexandre Petrescu

8.1 Introduction

Recently, energy efficiency become a key factor in the design of access network driven by the desire to reduce the communication carbon footprint and network operator energy bills as well as to extend terminal battery life [1–3]. Particularly, this is a requirement of paramount importance for network operators who are constantly expanding their network infrastructure with new access technologies and a significant number of access points to satisfy the increasing number of customers.

Several studies forecast a dramatic growth of the Internet traffic due to streaming media services such as IPTV, Video-on-Demand (VoD), and videoconferencing [4, 5]. Indeed, these services require network delivery paths with enough bandwidth (i.e., to ensure little delay variation and little packet loss). Therefore, one of the major challenges in access networks is to find routes that reduce the power consumption without compromising other performance indicators such as user throughput, mean packet delay, or percentage of packet loss.

With this recent dimension, the routing algorithms become highly complex with multi-objectives and multi-constraints.

Energy efficiency in access networks could be simply realized by implementing a bi-objective routing algorithm that considers both the available bandwidth and the link power consumption metrics instead of considering only the traditional bandwidth metric. In fact, the European Commission prepared a Code of Conduct (CoC) on energy consumption of broadband equipment [6]. This report specifies the maximum electricity consumption allowed for each link technology. According to

S. Ben Hadj Said (✉) · A. Petrescu
CEA List, CEA Saclay, Gif-sur-Yvette, Ile-de-France 91190, France
e-mail: siwar.benhadjsaid@cea.fr

A. Petrescu
e-mail: alexandre.petrescu@cea.fr

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0_8

this report, each technology has its proper power consumption independent of the related capacity. For instance, a fiber point-to-point with a capacity of 10 Gbps consumes around 8 W, whereas a fiber EPON with the same capacity consumes around 13.4 W. A simple heuristic is suggested to compute routing paths that are energy efficient and at the same time satisfy the user traffic requirements in terms of bit rate. This heuristic is based on the computation of the k paths according to the first metric and the selection of the best one according to a second metric. For this, a modified version of the Yen's algorithm is used as it is the most pertinent algorithm for determining k paths without loops.

To this end, we advocate that Software-Defined Networking (SDN) is an essential tool to achieve our objective. In fact, SDN is a recent trend in communications networking, whereby the behavior of networking devices is controlled by a logically centralized controller. This trend is reshaping the way networks are designed, managed, and secured. In fact, SDN replaces manual and specific interfaces of networking devices with a programmable and open interface. This enables the automation of tasks such as networking devices configuration and traffic policy management [7]. Having a global view of network topology including the link available bandwidth and power consumption, the SDN controller is able to calculate the optimal routing path while considering different objectives and constraints. In this sense, networking devices in the data plane (e.g., routers, switches, radio cells, etc.) just need to be equipped with the OpenFlow (OF) protocol in order to apply rules coming from the SDN controller.

The holistic control functions such as routing path computation are done in the SDN controller (e.g., in a data center or high-performance server). Moreover, due to the flexibility offered by SDN, it is possible to implement two different routing approaches in the SDN controller and to apply these approaches depending on the context and network operator goals. In the first approach (i.e., GoGreen approach 1), the network operator may consider that ensuring the adequate bandwidth for the user traffic (e.g., video streaming) is prior to reducing network energy consumption. In this case, the algorithm determines the k paths with the best available bandwidth and then selects among them the one with the least energy consumption. In the second approach (i.e., GoGreen approach 2), the network operator may consider that minimizing energy consumption in network is more important than offering paths with high bandwidth for the user traffic (e.g., web browsing, sensor messages). For that, the algorithm determines the k paths with the least power consumption and then selects among them the path with the highest bandwidth.

The remainder of the chapter is organized as follows. Section 2 presents the context of our work and summarizes the related work. Section 3 provides a formal definition of bi-objective routing problem. Section 4 gives an overview of the proposed k-shortest path algorithms and describes the GoGreen routing algorithm. Section 5 evaluates the performances of the proposed algorithm through extensive simulations. Section 6 concludes the chapter.

8.2 Background

8.2.1 Future Access Networks

Figure 8.1 shows an example of future access network architectures design [2]. It is divided into three parts, namely access, backhaul, and core networks. The access network represents the first contact of the user's terminal with 5G network and ensures the data traffic delivery between the user and the backhaul network. It includes heterogeneous access technologies such as LTE, DSL, fiber, and Wi-Fi. The aggregation network typically aggregates traffic coming from multiple access technologies and ensures the data traffic delivery to the core network. It includes heterogeneous technologies such as fiber and ethernet. The core network provides access to other packet data networks such as Internet. The future access network provides the following functions: (i) removing boundaries between mobile and fixed aggregation networks, (ii) having SDN-capable networking devices (routers, switches, optical termination units, etc.) and data forwarding in data plane, and (iii) setting up an SDN controller that has global network visibility.

8.2.2 Related Work

Energy-efficient routing in networks has been largely addressed in the literature. Here, we limit our discussion to the studies that are relevant to backhaul networks. The studies reported in [8–10] were among the first studies concerned about the

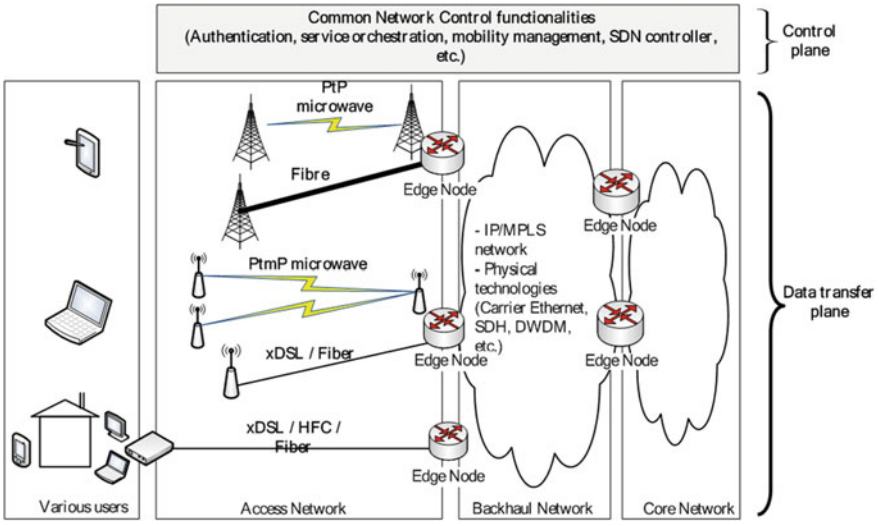


Fig. 8.1 Future access networks architecture

energy efficiency of nodes and links management in backbone networks. The basic idea of [8, 9] consists in routing the data packets over a given subset of network links during low traffic periods by means of coordinated strategy between routers. In this way, the links not included in routing paths can be powered off without causing issues in network availability. In [8], authors estimated the power consumption of nodes and links and suggested a new routing algorithm able to exploit the traffic information to minimize network consumption. In [9], authors proposed an Energy-Aware Routing (EAR) strategy to save energy in IP network during low traffic hours by allowing a subset of IP router interfaces be put in sleep mode. They propose to elect a set of routers (Exporter routers) that calculate the Shortest Path Trees (SPTs) used to fix the routing paths. The rest of routers (Importer routers) take as a reference these pre-calculated SPTs, modify their SPTs accordingly, and determine the links that have to be switched off. In [10], the authors proposed an algorithm that aims at reducing the network power consumption by adapting the network capacity to the current traffic demand. In their algorithm, the links are switched off when they are underutilized or in IDLE state. However, the idle links are switched on when they are required to guarantee a proper reaction to faults or changes in the traffic pattern. Recently, [11, 12] proposed a heuristic approach for the same problem and used the SDN concept in order to achieve their goal. Their algorithms turn on the top of an SDN controller and aim at turning off idle nodes and concentrate traffic on the smallest possible set of links. In all these works, a homogeneous topology (i.e., routers are connected via the same type of link technology, typically fiber links) assumption is taken. In our context, the topology is more realistic where heterogeneous link technologies (e.g., Fiber, Ethernet, 4G, Wi-Fi, WiMAX, etc.) between heterogeneous networking devices (e.g., smartphone, server, Wi-Fi access point, DSLAM, eNodeB, router, etc.) are deployed. In such realistic topology, the power consumption differs from one link to another. In our work, we try to reduce the network power consumption by suggesting an energy-aware routing algorithm. We consider that our solution is complimentary to the approach of turning off equipment. Together, the network can realize a large energy saving.

8.3 Mathematical Formulation

8.3.1 Network Model

The network is modeled as a directed weighted graph $G = (V,E)$, where $V = \{1, 2, \ldots, n\}$ is the set of nodes and E is the set of links. $n = |V|$ and $m = |E|$ represent the number of nodes and edges in graph G, respectively. Two positive weights $w(i,j) = \left(w_{ij}^b,\ w_{ij}^p\right) \in \mathbb{N} \times \mathbb{N}$ are associated with each edge (i,j). The weights w_{ij}^b and w_{ij}^p denote the available bit rate and power consumption of link (i,j). These metrics belong to different categories. The power consumption is an

additive metric (i.e., the path cost is the sum of the power consumption of individual links along the path), whereas the available bit rate metric is a concave one (i.e., the path cost is the minimum of the available bit rate metric of individual links along the path).

Each edge $(i,j) \in E$ has a capacity c_{ij} in Mbps. On edge (i,j) at a particular time index k, there is a traffic load l_{ij}^k (in Mbps). Thus, the available bit rate b_{ij}^k on the link (i,j) is calculated as follows:

$$\boldsymbol{b_{ij}^k = c_{ij} - l_{ij}^k}$$

A path in G from node $i_1 \in V$ to node $i_l \in V$ is a sequence $\{(i_1, i_2), (i_2, i_3), (i_3, i_4), \ldots, (i_{l-1}, i_l)\}$ of links in E.

8.3.2 Link Energy Consumption Model

A network node in the access network has a number of interfaces. Each interface can have one or multiple ports. The power consumption of one port depends on the link technology in use.

8.3.3 Traffic Model

We model the traffic as a set of L flows. Each flow $f \in F$ expects a specific bit rate b_f and has a specific delay constraint d_f.

8.3.4 Problem Formulation

The aim is to minimize the power consumption of network nodes (i.e., switches, routers, etc.) over time, while guaranteeing QoS for the different traffic flows in aggregation network.

The problem can be formulated as follows:

GIVEN:

- A physical topology represented by the graph G(V, E)
- A set L of flows, each flow is associated with a couple of source–destination (s,d), is expecting a specific bit rate b_f, and has a specific delay constraint d_f.

FIND:

- The optimal routing path for each flow that minimizes the network power consumption and at the same time ensures the adequate bit rate.

The goal is to jointly minimize the total network power consumption and ensure a good bandwidth for each video flow. In the following, we present in detail each term of the formulation.

8.3.4.1 Decision Variable

We introduce the binary variable x_{ij}^{f} to indicate whether the flow f is routed on link (i,j) or not, as follows:

$$x_{ij}^{f} = \begin{cases} 1, & \text{if the flow } f \text{ is routed on link}(i,j) \\ 0, & \text{Otherwise} \end{cases}$$

8.3.4.2 Objective Functions

In our problem, we have two objectives: finding the path that ensures the highest bit rate for each flow and reducing the power consumption in the network.

The objective function related to maximizing the bit rate is given by the following equation:

$$f\left(x_{ij}\right) = \max\left(w_{\text{path}}^{b}\right) \quad \text{where} \quad w_{\text{path}}^{b} = \min_{(i,j)\in E}\left(w_{ij}^{b} x_{ij}^{f}\right)$$

w_{path}^{b} represents the bit rate that a candidate path could offer for a given flow (i.e., path bit rate). Therefore, $f\left(x_{ij}\right)$ selects the candidate path that offers the highest bit rate. Recall that the path bit rate w_{path}^{b} is the minimum of the available bit rate metric of individual links along the same path.

The objective function related to minimizing the power consumption at the network is

$$g\left(x_{ij}\right) = \min \sum_{(i,j)\in E} {}^{p}_{ij}\, x_{ij}^{f}$$

8.3.4.3 Flow Conservation Constraints

We denote by $A_j^{-} = \{j \in V \mid (j,i) \in E\}$ and by $A_j^{+} = \{j \in V \mid (i,j) \in E\}$ the set of predecessor and successor nodes for any node $j \in V$. The constraints specified in the following equation guarantee that each flow is transported from its origin to its destination.

$$\sum_{j \in A_j^-} x_{ij}^f - \sum_{j \in A_j^+} x_{ij}^f = \begin{cases} 1 & \text{if } j = s \\ -1 & \text{if } j = d \\ 0 & \text{Otherwise} \end{cases} \quad \forall j \in V \quad \forall f \in F$$

8.3.4.4 Node-Link Capacity Constraints

The following equation states that the total bit rate of flows affected to the same link (i,j) must be lower than the maximum link capacity to avoid congestion situation.

$$\sum_{f \in F} b_f \, x_{ij}^f \leq \mu \, w_{ij}^f \quad \forall (i,j) \in V$$

8.3.4.5 Decision Variable Domain

$$x_{ij}^f \in \{0,1\} \quad \forall f \in F \quad \forall (i,j) \in E$$

In our problem, the selected routing path should provide the following: (i) presents the least power consumption and (ii) ensures the adequate bit rate. Our problem belongs to the class of Multiple Objective Combinatorial Optimization (MOCO) problems. It is known that MOCO problems are NP-hard and may be intractable (i.e., the number of efficient solutions may increase exponentially with the number of nodes) [13]. Thus, obtaining the optimal problem solution in large-scale scenarios cannot be viable. Therefore, to address our problem, we suggest a heuristic algorithm, called GoGreen routing algorithm, that has less computational complexity and enables us to find near-optimal solutions.

8.4 GoGreen Routing Algorithm

In order to design an efficient algorithm to solve the above problem, we first observe that an approach based on the use of k-shortest path algorithms would be an efficient heuristic for the solution of our problem. In the following, we give a small overview about the k-shortest path algorithms. Then, we present the suggested algorithm.

8.4.1 K-Shortest Path Algorithms

Routing in networks is usually associated with the computation of shortest paths. For instance, in the most widely used protocol Open Shortest Path First (OSPF), each router runs the Dijkstra algorithm [14] to compute the shortest path from one router to other destinations using a given routing metric [15]. In Cisco equipments,

the routing metric is defined to be the reference bandwidth (e.g., 10 Mbps) divided by the link bandwidth. However, Dijkstra is a single-shortest path algorithm (i.e., provides only one solution according to a given metric). Finding the path that minimizes the power consumption in network and offers the adequate bandwidth for the video delivery is a bi-objective problem. In this case, it is more relevant to compute a set of k-shortest paths between source and destination nodes to find the adequate path. The k-shortest path problem is a generalization of the single-shortest path problem in which k paths are determined in an increasing order of length. Epstein [16] focused on k-shortest path problem in which paths can contain loops. He proposed an algorithm that achieves the optimal time complexity $O(|E| + |V| \cdot \log|V| + k)$. Finding k loop-less shortest path, such as in our context, has proved to be more challenging. Several studies have examined this problem [17–20]. Yen's algorithm [19] is the most pertinent and fast k-shortest path algorithm without loops. It has a time complexity in $O(k \cdot |V|(|E| + |V| \cdot \log|V|))$. Therefore, we decided to use the Yen's algorithm in our solution. However, as we do not compute paths based on the hop count metric, we modified the Yen's algorithm in order to adapt to our case. Then, instead of looking for the k-shortest paths using the hop count metric, we compute the k paths that have the best available bandwidth or the least power consuming.

8.4.2 Algorithm Description

The GoGreen routing algorithm aims at reducing the network power consumption by considering both link available bandwidth and power consumption weights while computing the routing path. It is based on the computation of the k paths having the best weights using a modified version of Yen's algorithm.

The proposed routing algorithm is summarized in Fig. 8.2. Some parameters are required to execute this algorithm, namely a graph of node G where each link is characterized with two weights $(w_1; w_2)$, a source–destination pair $(\text{src}; \text{dst})$ and the maximum number of the calculated shortest path k.

The algorithm starts by running Yen's algorithm on the graph G. In the first step, the algorithm considers only the first weight w1 and calculates k candidate paths accordingly. In the case more than one feasible path is provided, the GoGreen algorithm sorts the candidate paths by the second weight w_2 so that the first path has the best value of w_2. After that, it selects the first path and returns it.

We identified two approaches to execute the proposed algorithm. In the first approach, we give priority to available bandwidth metric (i.e., $w_1 = w_{ij}^b$ and $w_2 = w_{ij}^p$). In the second approach, the priority is given to power consumption metric (i.e., $w_1 = w_{ij}^p$ and $w_2 = w_{ij}^b$).

(1) ***Approach 1 (priority for available bandwidth metric)***: The GoGreen routing algorithm computes the k feasible shortest paths according to available bandwidth metric. In other words, the algorithm determines k paths that have the

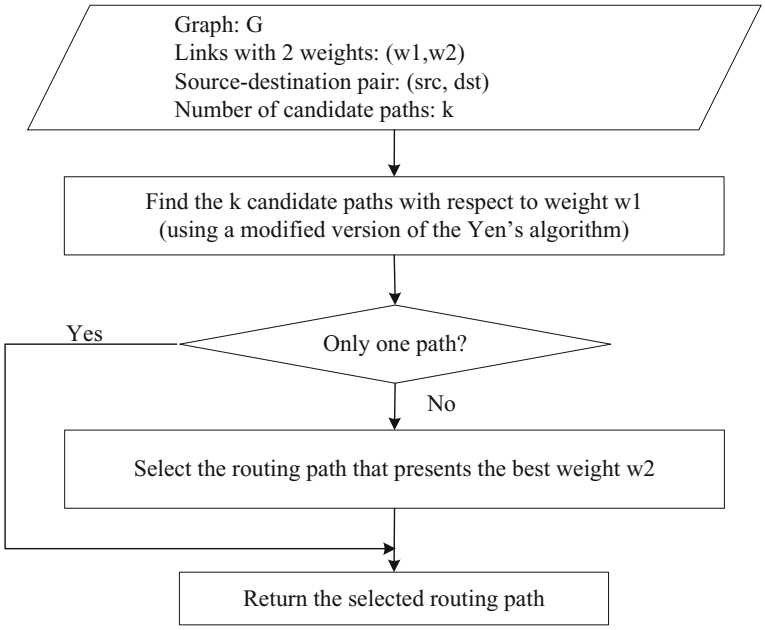


Fig. 8.2 GoGreen routing algorithm

highest available bandwidth value. Recall that the bandwidth metric represents the minimum of the available bandwidth of individual links along this path. Then, among these candidate paths, it selects the path that presents the least end-to-end power consumption.

(2) ***Approach 2 (priority for link power consumption metric)***: The GoGreen routing algorithm determines the k feasible shortest paths according to power consumption metric. After that, among the candidate paths, it selects the path with the highest available bandwidth. It is obvious, in this approach, that the priority is given to the power consumption of the path over its available bandwidth.

8.5 GoGreen Routing in SDN

GoGreen routing algorithm should be implemented in the SDN controller. One of the SDN controller features is his ability to have global view of the network topology including links properties (i.e., type, capacity, power consumption, etc.). As the SDN controller is hosted in a high-performance server or a data centre, processing and memory resources are no more constraints for using a k-shortest path algorithm. Figure 8.3 presents the exchanges between networking devices in the data plane with the SDN controller to establish the data path. When the edge node receives a packet data that is not associated with a specific flow entry, it

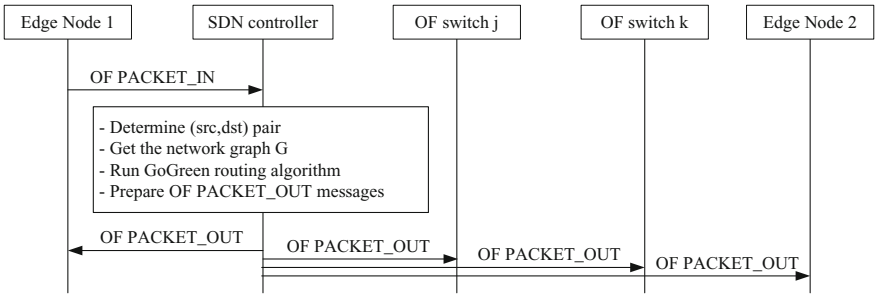


Fig. 8.3 Data path establishment flow chart

encapsulates the packet header in an OF_PACKET_IN message and sends it to the SDN controller. This latter determines the source and destination associated with this packet and gets the network graph from its database. Then, it runs the GoGreen routing algorithm in order to compute a feasible path. After that, it prepares the OF_PACKET_OUT messages that are needed to prepare the routing path in the data plane. Finally, the SDN controller sends the OF_PACKET_OUT messages to networking devices belonging to the calculated path.

The different steps related to the GoGreen routing algorithm execution in the SDN controller are depicted in Fig. 8.4. When the SDN controller receives an OF_PACKET_IN message, it analyzes the packet header. If the flow needs a specific bandwidth (e.g., the source IP address is subscribed in QoS service in the controller), the SDN controller determines the value of the parameter k (e.g., it can vary depending on network load) and run the GoGreen routing algorithm in mode approach 1. If the SDN controller decides that the new flow has no specific requirements in terms of QoS, then it determines the value of the parameter k and runs the GoGreen routing algorithm in mode approach 2. In both cases, the GoGreen routing algorithm execution results in a calculated routing path. After that, the SDN controller identifies the OF switches as well as their network interfaces that participate in this routing path. Then, it prepares and sends the OF_PACKET_OUT messages to the identified OF switches in order to install and configure the routing path in the data plane. If the routing path is installed successfully, the SDN controller updates its database with the new values of the "available bandwidth" metric associated with the links participating in this routing path.

8.6 Performances Evaluation

8.6.1 Simulation Setting

The performance of GoGreen routing algorithm is investigated through extensive simulations. We randomly generate a graph of 50 nodes with 300 links. For each link, we randomly associate two nodes. Each link is associated with two different

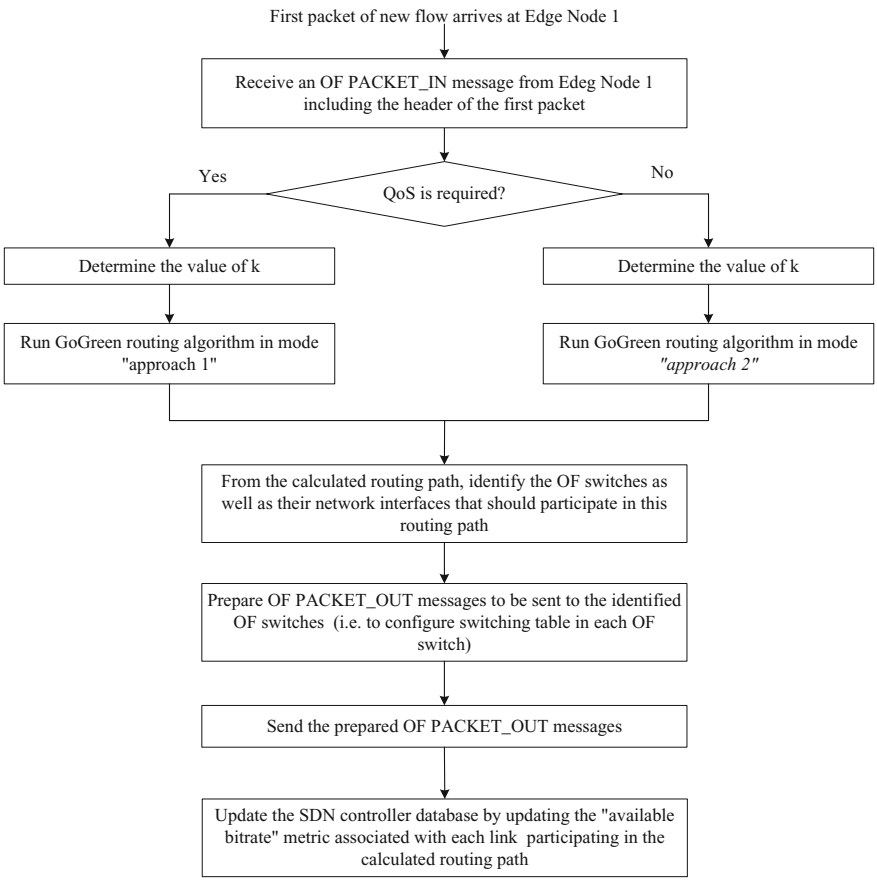


Fig. 8.4 GoGreen routing in SDN controller

weights: additive weight (i.e., power consumption) and concave weight (i.e., available bandwidth). Table 8.1 shows examples of link technologies used in access/aggregation networks, their maximal download bit rate (i.e., link capacity) as well as the maximum power consumption as specified in the CoC [6]. The link power consumption and capacity are randomly selected from this table. The load and available bandwidth on each link are initially set to zero and link capacity, respectively. After each path assignment, the load and available bandwidths of all links within the path are accordingly updated.

The routing requests are generated according to Poisson distribution with the rate of λ. There are several types of multimedia traffic with different needs in terms of bandwidth utilization. Table 8.2 provides some examples of multimedia services and their requirement in terms of bit rates [21]. Therefore, for each routing request, we randomly assign a bit rate using a uniform distribution in the interval [500 kbps; 10 Mbps].

Table 8.1 Link technologies and their power consumption

Link technology	Capacity (Mbps)	Link power consumption (W)
Fiber point-to-point	1000	1.7
Fiber point-to-point	10,000	8
Fiber GPON	2500	6
Fiber EPON	1000	6
Fiber EPON	10,000	13.4
ADSL2plus	24	0.4
VDSL2	75	1.4
DOCSIS3.0	304	6
WiMAX	219	~25
LTE	100	~30
Wi-Fi	54	8

Table 8.2 Link technologies and their power consumption

Multimedia type	Bitrates
Broadcast IPTV	1−4 Mbps (SD) or 6−10 Mbps (HD)
Live event video streaming	1− 4 Mbps
IP surveillance video	500 kbs to 2 Mbps
Interactive video conferencing	1 Mbps
On-demand video	1−4 Mbps (SD) or 6−10 Mbps (HD)

8.6.2 Performance Metrics

We focus on the following metrics:

- ***Success rate***: it represents the percentage of routing requests where the calculated path ensures the required bit rate.
- ***Total network power consumption***: it is calculated as the sum of all path power consumption after assigning paths for all routing requests. The path power consumption is defined as the total power consumption of all links of the same path.

8.6.3 Simulation Results

In our simulations, Dijkstra bandwidth (i.e., Dijkstra algorithm using link available bit rate as a metric), Dijkstra energy (i.e., Dijkstra using link power consumption as a metric), and GoGreen routing are executed independently to find a feasible path. For each algorithm, we calculate total power consumption, energy gain, and success

rate. Recall that Dijkstra bandwidth is the widely used algorithms in current routing protocols.

First, we take a look at the *k* parameter impact on the performance of the Dijkstra and GoGreen routing algorithms. For that, we vary the k value from 1 to 4 and we calculate the total power consumption and energy gain. The latter parameter is defined as the energy saving that our algorithm can realize compared to Dijkstra bandwidth. It is calculated as follows:

$$\text{energy gain} = \frac{\text{power}_{\text{GoGreen}} - \text{power}_{\text{Dijkstra}}}{\text{power}_{\text{Dijkstra}}}$$

For the routing requests generation, we generated 1000 sampling events. At each event, we use the Poisson distribution ($\lambda = 10$ requests/s) to generate a number of routing requests.

Figures 8.5 and 8.6 show the total power consumption and the energy gain, respectively, as a function of the parameter k for the four simulated algorithms. The total power consumption of Dijkstra bandwidth and Dijkstra energy is the same for all values of k. This can be explained by the fact that Dijkstra provides only one path and does not depend on the parameter k. Dijkstra bandwidth presents the highest power consumption compared to the rest of algorithms as it considers only the bandwidth as a metric. As expected, Dijkstra energy presents the least total power consumption as it searches, for each routing request, the path with the least power consumption. GoGreen routing algorithm considers both metrics: available bandwidth and power consumption.

In GoGreen (approach 1), k-shortest path is determined by using the bandwidth metric. Among these k paths, the path with the least power consumption is selected. The total power consumption of GoGreen (approach 1) decreases when the parameter k increases. This is expected because when the k-shortest path algorithm produces more candidate paths, a path with a relatively good bandwidth and a lower

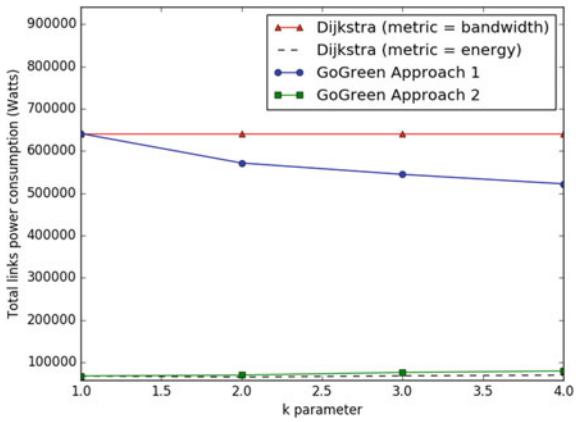


Fig. 8.5 Total power consumption

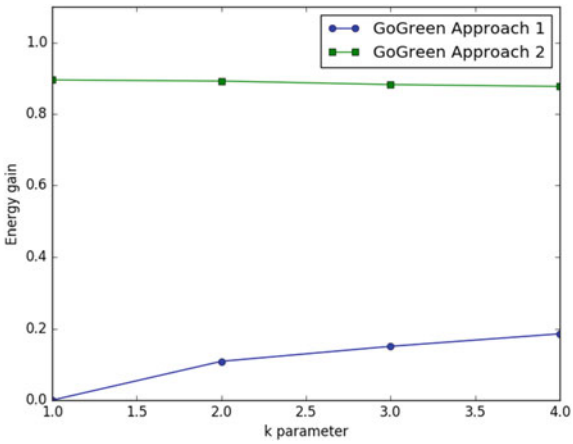


Fig. 8.6 Energy gain versus k parameter

power consumption will be selected. As we can see in Fig. 8.5, GoGreen (approach 1) can realize from 10% to of 20% energy saving when k varies from 2 to 4.

In GoGreen (approach 2), k-shortest path is determined by using the link power consumption metric. Among these k paths, the path with the highest bandwidth is selected. The total power consumption of GoGreen (approach 2) increases when the parameter k increases. In fact, when the k-shortest paths produce more candidate paths, a path with a better bandwidth and a higher power consumption will be selected. GoGreen (approach 2) can realize more than 50% of energy saving.

Figure 8.6 presents the energy saving that GoGreen routing in both approaches could achieve compared to Dijkstra bandwidth. As it is observed, GoGreen (approach 1) routing allows around 10% energy saving. Further, GoGreen (approach 2) routing allows more than 80% of energy saving.

In the second simulation set, we set the parameter k to 2 for GoGreen routing algorithms. Then, we investigate the λ parameter (i.e., flow arrival rate) impact on the performance of the Dijkstra bandwidth, Dijkstra energy, and GoGreen routing algorithm. For doing that, we vary the λ value from 0.01 flows/s to 1 flows/s and we calculate the total power consumption, energy gain, and the success rate.

Figure 8.7 shows the total power consumption as a function of parameter λ. We note that the gap between the total power consumption of GoGreen (approach 1) and that of Dijkstra bandwidth increases as λ increases. Total power consumption of GoGreen (approach 2) is almost equivalent to that of Dijkstra energy.

For each traffic flow, we check if the bandwidth advanced by the computed path can satisfy the bandwidth required by the arriving flow. Therefore, the success rate is calculated as the sum of satisfied flow divided by the total number of flows. The success rate as a function of lambda is depicted in Fig. 8.8. As expected, the success rate decreases when lambda increases. Dijkstra bandwidth and GoGreen routing present a better success rate. Dijkstra energy presents the lowest success rate. Although this algorithm achieves the highest energy saving (80%), more than 90% of flows are not satisfied in terms of bandwidth. In fact, Dijkstra energy

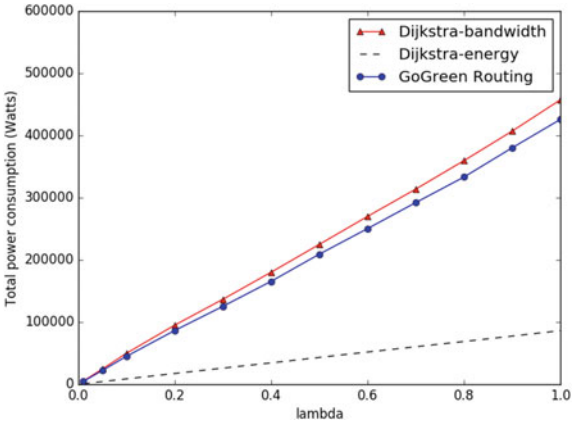


Fig. 8.7 Total power consumption as a function of λ (k = 2)

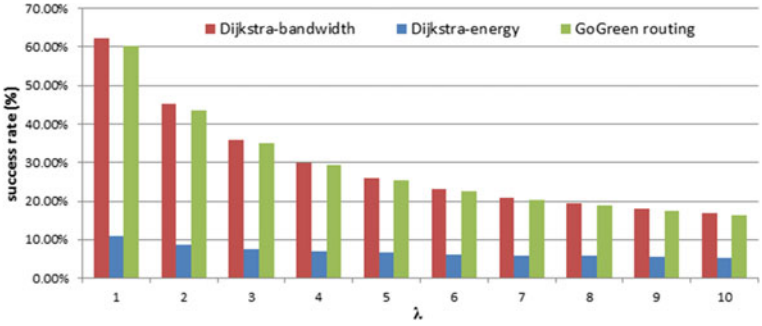


Fig. 8.8 Success rate as a function of λ (k = 2)

algorithm considers the link power consumption as the only routing metric while computing paths for each flow, without considering available bandwidth. As observed, GoGreen routing can save energy (around 10% compared to Dijkstra bandwidth) while having the same success rate as Dijkstra bandwidth. As GoGreen routing algorithm considers both the link power consumption and available bandwidth while computing paths, the flows requirement in terms of bandwidth are more likely to be satisfied.

8.7 Conclusion

In this book chapter, we suggested a novel routing algorithm, called GoGreen routing, where the computed path respects the traffic requirement in terms of QoS and at the same time consumes less energy. It considers two metrics, namely link

available bandwidth and power consumption. We also suggested two approaches depending on metric priority for network operators. In GoGreen (approach 1), the priority is given to the available bandwidth metric over power consumption. In GoGreen (approach 2), the link power consumption is prioritized. Through extensive simulations, we showed that GoGreen (approach 1) can reduce network power consumption and at the same time, ensures a good QoS comparable to what is achieved by Dijkstra algorithm when using a bandwidth-based metric.

Acknowledgements This research was partially supported by the European Celtic-Plus project CONVINcE and it was partially funded by Finland, France, Sweden, and Turkey.

References

1. Ericsson White Paper (2015) 5G Energy Performance
2. Jaber M, Imran MA, Tafazolli R, Tukmanov A (2016) 5G backhaul challenges and emerging research directions: A survey. IEEE Access 4:1743–1766
3. Bolla R, Bruschi R, Davoli F, Cucchietti F (2011) Energy efficiency in the future internet: a survey of existing approaches and trends in energy-aware fixed network infrastructures. IEEE Commun Surv Tutor 13(2):223–244
4. Cisco White Paper (2015) Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 20152020
5. Soldani D, Manzalini A (2015) Horizon 2020 and beyond: on the 5G operating system for a true digital society. IEEE Veh Technol Mag 10(1):32–42
6. European Commission (2013) Code of Conduct on Energy Consumption of Broadband Equipment, Version 5.0
7. Open Networking Foundation (ONF) (2012) Software-defined networking: the new norm for networks. Technical report
8. Chiaraviglio L, Mellia M, Neri F (2009) Energy-aware backbone networks: a case study. In: IEEE international conference on communications workshops (ICC Workshops 2009). pp 1–5
9. Cianfrani A, Eramo V, Listanti M, Marazza M, Vittorini E (2010) An energy saving routing algorithm for a green OSPF protocol. In: INFOCOM IEEE conference on computer communications workshops 2010. IEEE, 2010, pp 1–5
10. Bianzino AP, Chiaraviglio L, Mellia M, Rougier J-L (2012) Grida: green distributed algorithm for energy-efficient IP backbone networks. Comput Netw 56(14):3219–3232
11. Amokrane A, Langar R, Boutaba R, Pujolle G (2015) Flow-based management for energy efficient campus networks. IEEE Trans Netw Serv Manage 12(4):565–579
12. Tadesse SS, Casetti C, Chiasserini C (2016) Energy-efficient traffic allocation in SDN-based backhaul networks: theory and implementation. CoRR. Accessed http://arxiv.org/abs/1609.04844
13. Serafini P (1987) Some considerations about computational complexity for multi objective combinatorial problems. In: Recent advances and historical development of vector optimization. Springer, Berlin, pp 222–232
14. Dijkstra EW (1959) A note on two problems in connexion with graphs. Numer Math 1 (1):269–271
15. Moy J (1998) RFC 2328: OSPF version 2. Technical report
16. Eppstein D (1998) Finding the k shortest paths. SIAM J Comput 28(2):652–673
17. Hoffman W, Pavley R (1959) A method for the solution of the n th best path problem. J. ACM (JACM) 6(4):506–514

18. Katoh N, Ibaraki T, Mine H (1982) An efficient algorithm for k shortest simple paths. Networks 12(4):411–427
19. Yen JY (1971) Finding the k shortest loopless paths in a network. Manage Sci 17(11):712–716
20. Lawler EL (1972) A procedure for computing the k best solutions to discrete optimization problems and its application to the shortest path problem. Manage Sci 18(7):401–405
21. Lisa P (2012) WLAN design for optimized Wi-Fi video delivery, http://searchnetworking.techtarget.com/tip/WLAN-design-for-optimized-Wi-Fi-video-delivery

Chapter 9
Toward Green SDN Networks

Yannick Carlinet and Nancy Perrot

9.1 Introduction

In network architectures based on Software-Defined Networking (SDN), the control plane is decoupled from the data plane while traditional network routers integrate both planes. The control of network elements in network is centralized in external devices called SDN controllers that communicate with the network devices through Application Programming Interfaces (API).

SDN brings flexibility and offers huge opportunities for network programmability thanks to open interfaces. Thus, the migration of the networks towards SDN-based networks is a key solution to manage and efficiently configure the network in order to cope with the huge increase of traffic and transport of new heterogeneous services with multiple Service-Level Agreements (SLA). Furthermore, SDN could allow the managing of on/off functionality on router interfaces that is a promising way to drastically reduce the energy consumption of the networks.

This chapter presents our contributions to an initial design of SDN/NFV. In Sect. 9.2, we investigate the so-called Controller Placement Problem (CPP) in the context of a Wide Area Network (WAN). The purpose is to find the best placement of controllers that allows managing a SDN-based WAN. Section 9.3 is dedicated to the topic of geographical load balancing, and investigates the achievable gain of providing a sleep mode mechanism in DCs and the gain that can be achieved by using a geographical load balancing algorithm.

Y. Carlinet (✉) · N. Perrot
Orange Labs, Chatillon, France
e-mail: Yannick.Carlinet@orange.com

N. Perrot
e-mail: Nancy.Perrot@orange.com

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0_9

9.2 SDN Controller Placement

The SDN controller of a wide network area is logically centralized, but it must be physically deployed on several distributed devices to satisfy different constraints. First, for scalability purpose: the capacity of a controller in terms of processing, memory, and in/out bandwidth, limits the number of nodes (e.g., switch, routers) that could be managed by a single controller. Second, for performance purpose: the communication delay between a node and its controller must be lower than an admissible delay bound to meet QoS (Quality of Service) requirements. Third, for reliability reasons: to prevent a disruption of the network in case of controller failure.

In this section, the so-called Controller Placement Problem (CPP) is investigated in the context of a Wide Area Network. The objective of this problem is to find the optimal placement/deployment of controllers to efficiently manage a SDN-based WAN.

This problem has been extensively studied in the past few years. Several solutions have been proposed depending on different optimization criteria and additional constraints: minimum number of controllers, minimization of the worst-case latency between nodes and controllers, minimization of the inter-controller latency, optimization of the balancing of clusters of routers for each controller, taking into account or not the controllers capacities, and finally considering failures of controllers.

In its simpler form, the Controller Placement Problem reduces to a Facility Location Problem and it is proved to be NP Hard [6]. This study presents formulations proposed in the paper [12] for the optimization of the placement of controllers in a resilient SDN architecture. Several optimization criteria are embedded in the formulations (in the objective function or as a constraint). Our formulations can be used to help network architects to find the best deployment of controllers in operational networks. The main objective is to minimize the number of active resources (energy) that translates into minimizing the number of controllers which has an increased impact in case of over provisioning of SDN controllers for resiliency purpose.

9.2.1 Problem Statement

The main idea driving the development of SDN is to separate the data plane from the control plane and, thus, to provide an abstraction between the controller (control plane) and the network elements. This controller acts as the “brain” of the network: it centralizes the control plane policies, and instantiates the routing rules for each network elements through an API. This network architecture brings huge flexibility to the network management. However, to maintain a tight latency between the

controller and its network elements and to ensure the reliability of the network, the controller redundancy could lead to solutions with multiple under-used controllers. Thus, our purpose in this work is to find the optimal trade-off between several (sometimes opposed) optimization criteria thanks to integer linear programming.

A "good" controller placement in a network depends on the considered use case: for instance, in a data center the latency is not a fundamental criterion while the reliability is critical.

The network is represented by an undirected graph $G = (V, E)$, where $V = \{1, \ldots, n\}$ is the set of nodes (routers), and E is the set of edges. An edge uv corresponds to a bidirectional link between two nodes u and v. The weight, d_{uv}, of an edge, represents the latency of the link.

Assuming that a controller could be placed at any node, the length of the shortest path from a router r to its controller c corresponds to the propagation latency and it is denoted by $d(r, c)$ Assuming k controllers that can be distributed throughout the network at any of the possible locations $C = (C_1, C_2, \ldots)$, several usual metrics are defined by:

- The average latency between a node and its controller:

$$L_{avg}(C_i) = \frac{1}{n} \sum_{v \in V} min_{c \in C_i} d(v, c) \tag{9.1}$$

- The maximal or worst-case latency between a node and its controller:

$$L_{max}(C_i) = max_{v \in V} min_{c \in C_i} d(v, c) \tag{9.2}$$

- The maximal latency among all the couples of controllers:

$$L_{cc}(C_i) = max_{C_1, C_2 \in C_i} d(c_1, c_2) \tag{9.3}$$

- The load balancing of the controllers:

$$LB(C_i) = max_{c \in C_i} n_c - min_{c \in C_i} n_c \tag{9.4}$$

where n_C is the number of nodes managed by the controller c.

A good controller's placement solution is a placement in which one or several of these criteria are minimized.

9.2.1.1 Related Work

In [6], Heller et al. addressed first the Controller Placement Problem in WAN [6], taking into account the control plane propagation latency. They study three placement metrics: the minimum average latency (9.1) is obtained by solving a minimum k-median problem, the worst-case latency (9.2) by solving the related minimum k-center problem and finally the number of nodes within a latency bound, by solving then a maximum k set-covering problem. This paper motivates the

relevance of the controller's placement problem and quantifies the impact of this placement on real topologies with up to 50 nodes.

However, some structuring metrics like inter-controllers latencies (9.3) or the load balancing (9.4) are not taken into account.

Several works, [7, 10], deal with a multi-objective approach. They include the inter-controllers latency (9.3), load balancing (9.4), and resilience among the decision criteria. They developed a MATLAB framework named POCO for Pareto-based Optimal Controller Placement. This framework allows displaying the Pareto frontier enumerating all the feasible placements and visualizing the different Pareto optimal placements. They also developed heuristics to deal with larger networks. But their purpose is to find the best placement of a set of fixed controllers, not to minimize their quantity.

In [16], the authors consider a capacity on the load of controllers and suggested a capacitated K-center algorithm, taking into account the average latency and the controller capacity as a constraint.

In [8], the authors suggest an algorithm to find the minimum number of controllers within a delay limit in the worst-case scenario of a single controller failure. In [13], the authors consider a failure probability of each component and suggest a heuristic to determine the best placements to achieve fine reliability between controllers and nodes. Their study shows that the reliability is strongly dependent on the network topology structure, particularly the graph density.

More generally, the controller placement problem is a variant of the well-known facility location problem [3]. As explained in [6], minimizing (9.1) and (9.2) corresponds respectively to the k-median problem and to the k-center problem. While minimizing the number of controllers refers to the set-covering problem. Several studies [5, 14, 15] introduced failure in the delivery network, referring then to the fault-tolerant facility location problem.

The increasing amount of literature on the Controller Placement Problem in the past 3 years shows the growing interest of the SDN community for providing a good controllers placement. For small instances, it is possible to enumerate all feasible solutions. However, enumerating all feasible solutions is obviously not realistic when working with a real network.

In the following, linear programming tools are used to provide both the optimal number of controllers and their optimal placement within the network. Our approach allows considering most of the criteria.

9.2.2 The Controller Placement Problem

When the objective is to minimize the number of active controllers, the network performance criteria could be shifted in the constraints, so that the QoS requirements are still guaranteed. The routers are assumed to be automatically assigned to

their nearest available controller; and thus to ensure a good QoS performance a constraint on the allowed maximal latency is introduced. The set of nodes from which a router could be reached within this maximal latency is computed in a preprocessing procedure.

Let C be the set of candidate nodes for placing the controllers indexed by i and R the set of routers nodes j (C is a subset of R). D is the latency matrix, $d_{(i,j)}$ being the length of the shortest path between i and j. The maximal allowed latency between a router and its controller is denoted l_{max} and the maximal latency between two controllers is noted l_{cc-max}. The maximal difference between loads of two controllers is δ. The cover matrix A of graph G is defined as

$$a_{ij} = \begin{cases} 1 & \text{if} \quad d_{ij} \leq l_{max} \\ 0 & \quad otherwise \end{cases}$$

To ensure that the router j will be assigned to one of its nearest controllers, we sort all the candidate sites i by increasing the order of d_{ij} and we note σ_{jq} the index of candidate site in position q.

The shortest paths matrix, D, can be computed with Djikstra algorithm [2] for each couple of nodes (complexity $O(mn + n^2 log(n))$) or using Floyd–Warshall algorithm [4] (complexity $O(n^3)$)

The three binary decision variables $\forall i, i' \in C, \forall j \in R$ are:

- $x_{ij} \in \{0, 1\}$, that takes value 1 if controller i is assigned to the router j, 0 otherwise.
- $y_i \in \{0, 1\}$, that takes value 1 if node i is active (it has a controller), 0 otherwise.
- $t_{ii'} \in \{0, 1\}$, that takes value 1 if $y_i = y_{i'} = 1$, i.e., if both nodes i and i' are active, 0 otherwise.

The following Controller Placement Problem formulation aims at finding the minimal number of required controllers, taking into account all the described metrics as side constraints.

$$\min \sum_{i \in C} y_i$$

Such that

$$\sum\nolimits_{i \in C} a_{ij} y_i \geq 1 \quad \forall j \in R \tag{9.5}$$

$$\sum\nolimits_{i \in C} x_{ij} = 1 \quad \forall j \in R \tag{9.6}$$

$$x_{ij} \leq y_i \quad \forall i \in C, \forall j \in R \tag{9.7}$$

$$y_{\sigma_{jq}} \leq \sum_{m=1}^{q} x_{j\sigma_{jm}} \quad \forall j \in R, \forall q \in [1, |C| - 1] \tag{9.8}$$

$$t_{ii'} d_{ii'} \leq l_{cc-max} \quad \forall i, i' \in C \tag{9.9}$$

$$-\delta - |R|(1 - t_{ii'}) \leq \sum_{j \in R} (x_{ij} - x_{i'j}) \, \forall i, i' \in C \tag{9.10}$$

$$\sum_{j \in R} (x_{ij} - x_{i'j}) \leq \delta + |R|(1 - t_{ii'}) \, \forall i, i' \in C \tag{9.11}$$

$$t_{ii'} \geq y_i + y_{i'} - 1 \quad \forall i, i' \in C \tag{9.12}$$

$$t_{ii'} \leq y_i \quad \forall i, i' \in C \tag{9.13}$$

$$t_{ii'} \leq y_{i'} \quad \forall i, i' \in C \tag{9.14}$$

$$x_{ij}, y_i, z_j, t_{ii'} \in \{0, 1\} \quad \forall i, i' \in C, \forall j \in R \tag{9.15}$$

Constraints (9.5) ensure that each router must be covered by at least one controller within the latency bound. Constraints (9.6) and (9.7) assign each router to exactly one controller, while constraints (9.8) ensure that the routers are assigned to their nearest available controller. (9.9) is the constraint on the maximal allowed inter-controller latency. (9.10) and (9.11) are the load balancing constraints in terms of the number of assigned routers is the load difference between two controllers cannot be more than a fixed threshold δ. (9.12), (9.13), and (9.14) are used to define the variables $t_{ii'}$ that are necessary to make the formulation linear ($t_{ii'} = y_i y_{i'}$ is required to measure the difference of load between each couple of clusters of routers). Finally, (9.15) defines the binary variables.

This formulation has been solved with the commercial solver Cplex (CPLEX), on several instances from the SND Lib [11]. Some solutions of the Cost Instance are represented below. This network instance is composed of 37 nodes and 57 edges, and the optimal solution represented in Fig. 9.1 was obtained with parameters l_{max} = 30% and l_{cc-max} = 70% of the graph diameter, and δ = 2.

The relaxation of the load balancing constraint is illustrated in Fig. 9.2. The optimal number of controllers is still 4, while the controller's clusters are very unbalanced: from 3 to 18 nodes.

The sensitivity analyses of the three main parameters of the problem are summarized in Table 9.1. First, the number of controllers increases when the maximal latency l_{max} decreases: from 2 controllers when l_{max} = 0.4 to 8 controllers when l_{max} = 0.25. Second, we note that the latency between each couple of controllers l_{cc-max} is very restrictive to obtain a feasible solution. In addition, when its value increased, it changes the localization of the controllers, yielding to solutions where controllers are closer to the graph border.

We used the framework POCO [7] to evaluate the solutions on the three main performance criteria (Figs. 9.3 and 9.4).

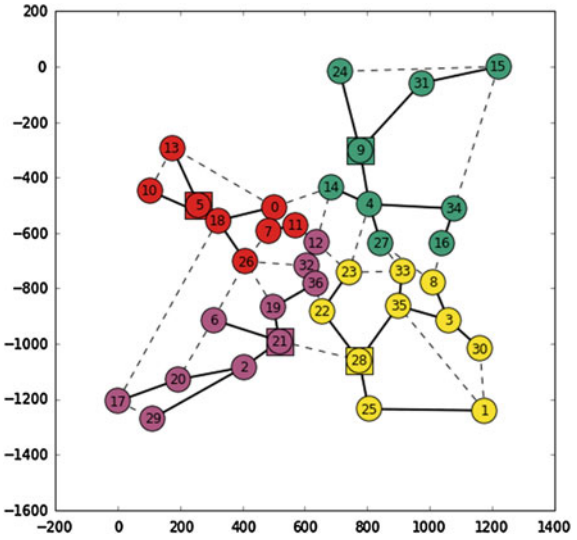


Fig. 9.1 Optimal solution with lmax = 30%, lcc-max = 70%, and δ = 2

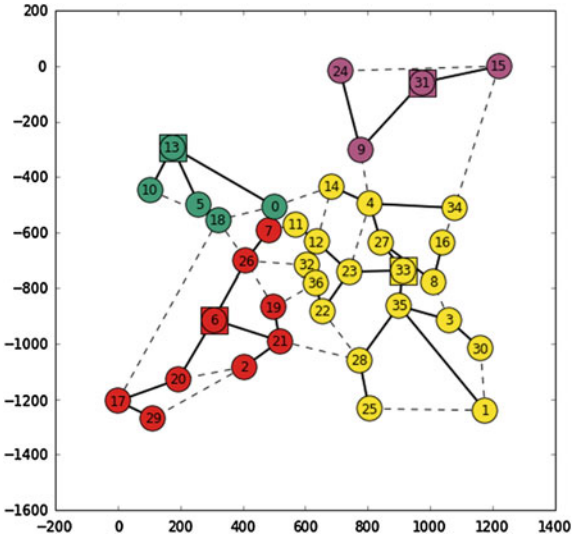


Fig. 9.2 Optimal solution with lmax = 30%, lcc-max = 70%, and δ = 37

In the above figures, the green points correspond to the optimal placement with routers assignment to their nearest controller. The red points correspond to the same placement but taking into account the additional constraints (load balancing, latency). The solutions of the linear programming formulation are a good trade-off between the maximal latency (2), the balancing of clusters (4) and the inter-controllers latency (3). Indeed, the green point solution is near the Pareto frontier for these three metrics. Nevertheless, the assignment to the nearest

Table 9.1 Sensitivity analysis

l_{max}	l_{cc-max}	δ	Placement
0.4	0.7	3	4 21
0.3	0.7	3	9 18 21 28
0.27	0.7	3	2 4 11 26 28 34
0.25	0.7	3	2 9 18 22 23 24 26 30
0.35	0.4	3	ϕ
0.35	0.7	3	10 19 22 29
0.35	0.9	3	14 16 19 32
0.35	1.0	3	2 5 8 9
0.35	0.7	1	2 3 5 9
0.35	0.7	7	6 9 28
0.35	0.7	15	6 31 35

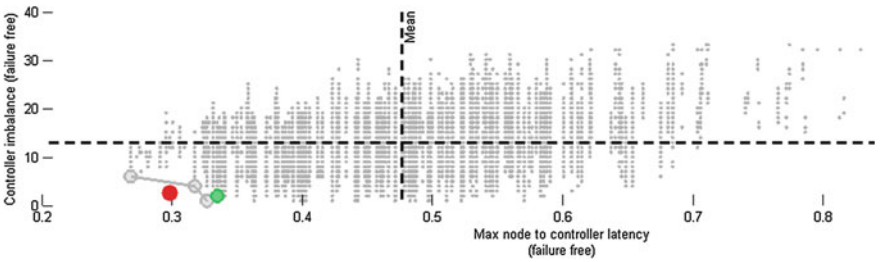


Fig. 9.3 Cluster balancing on maximal latency

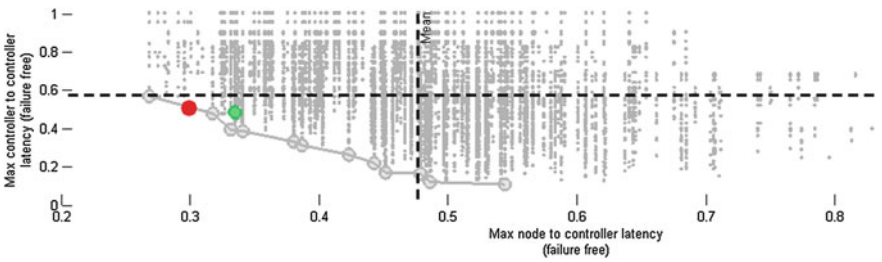


Fig. 9.4 Inter-controller latency on maximal latency

controller could increase the maximal latency if it allows minimizing the average latency. The red colored solution is out of the Pareto front in Fig. 9.3, as routers are assigned to their second controller to satisfy load balancing constraints.

In Figs. 9.5 and 9.6, the optimal controller placement of instance ZIB (54 nodes) from SNDlib is represented. It gives an illustration of the gain obtained when relaxing the latency constraints, from 30 to 50% of the graph diameter.

All instances from SND lib were solved to optimality in a few seconds.

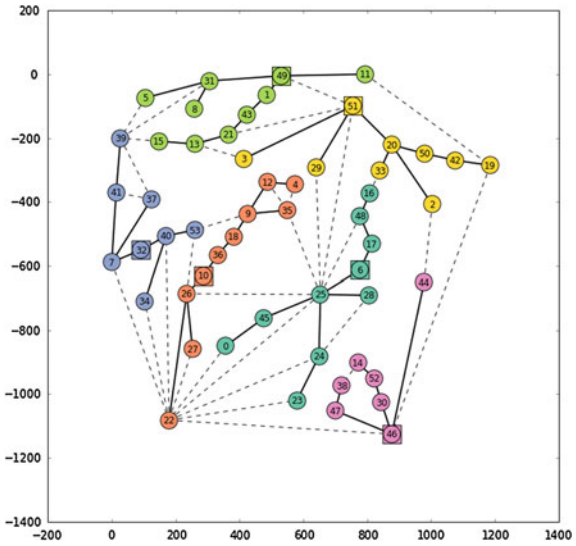


Fig. 9.5 Optimal solution with l_{max} = 30%, l_{cc-max} = 70%, and δ = 3

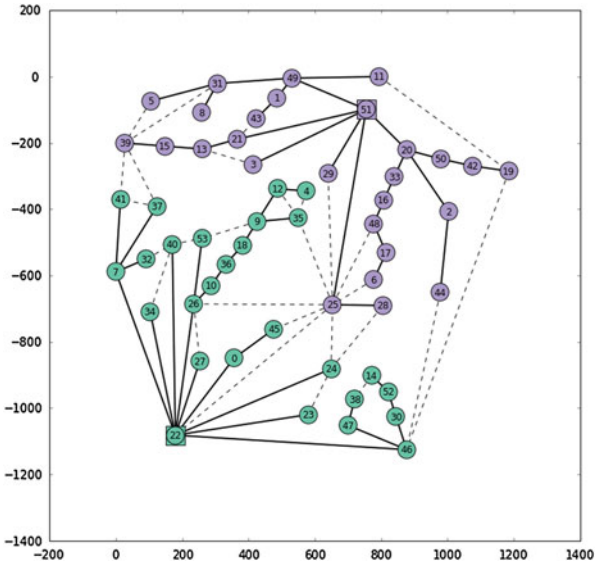


Fig. 9.6 Optimal solution with l_{max} = 50%, l_{cc-max} = 70%, and δ = 3

9.2.3 The Resilient Controller Placement Problem

If a controller fails, its routers must be assigned to another one, increasing then the latency between routers and controller, and giving rise to unbalanced domains, particularly if the secondary controller takes the management of all the routers of the failed controller.

In this section, we consider a failure probability of the controller, assuming that these failure probabilities are inherent to the controller and independent from each other.

We denote p the failure probability of a controller. The new variables are the following:

- $x_{ij}^k \in \{0, 1\}$, that takes value 1 if controller i is the kth backup controller of router j, 0 otherwise.
- $z_j^k \in \{0, 1\}$, that takes value 1 if router j has a $(k-1)$th backup controller, but not a kth backup controller, 0 otherwise.

The router j is assigned to its kth controller if and only if all the $(k-1)$th primary controllers fail.

The probability that all the controllers of a router j fail until level $(k-1)$ is p^{k-1}. In that case, if j is assigned to controller i at level k then it is operational with probability $(1-p)$.

The delay of communication between the kth controller i and j is:

$$\sum_{i \in C} d_{ij} x_{ij}^k (1-p) p^{k-1}$$

When there is no controller at level k for router j, it induces a penalty cost defined by

$$l_{max} z_j^k p^{k-1}$$

The objective is still to minimize the number of active controllers, to which we add the total delay. Then, a formulation of the resilient controller placement problem is

$$\min \sum_{i \in C} y_i + \sum_{j \in R} \sum_{k=1}^{|C|} \sum_{i \in C} d_{ij} x_{ij}^k (1-p) p^{k-1} + l_{\max} \sum_{j \in R} \sum_{K=1}^{|C|+1} p^{k-1} z_j^k$$

where

$$\sum\nolimits_{i \in C} x_{ij}^k + \sum\nolimits_{s=1}^{k} z_j^s = 1 \quad \forall j \in R, \forall k \in [1, |C|+1] \tag{9.16}$$

$$\sum_{i \in C} x_{ij}^{k} = 1 \quad \forall j \in R, \forall k \in [1, \gamma] \tag{9.17}$$

$$\sum_{k=1}^{|C|} x_{ij}^{k} \leq y_i \quad \forall i \in C, \forall j \in R \tag{9.18}$$

$$-\delta - |R|(1 - t_{ii'})) \leq \sum_{j \in R} (x_{ij}^{k} - x_{i'j}^{k'}) \quad \forall i, i' \in C, \forall k, k' \in [1, \eta] \tag{9.19}$$

$$\sum_{j \in R} (x_{ij}^{k} - x_{i'j}^{k'}) \leq \delta + |R|(1 - t_{ii'}) \quad \forall i, i' \in C, \forall k, k' \in [1, \eta] \tag{9.20}$$

$$d_{ii'} t_{ii'} \leq l_{cc-max} \quad \forall i, i' \in C \tag{9.21}$$

$$t_{ii'} \geq y_i + y_{i'} - 1 \quad \forall i, i' \in C \tag{9.22}$$

$$t_{ii'} \leq y_i \quad \forall i, i' \in C \tag{9.23}$$

$$t_{ii'} \leq y_{i'} \quad \forall i, i' \in C \tag{9.24}$$

$$x_{ij}^{k}, y_i, z_j^{k}, t_{ii'} \in \{0, 1\} \quad \forall i, i' \in C, \forall j \in R, \forall k \in [1, |C| + 1] \tag{9.25}$$

Constraints (9.16) guarantee that, at each level k, a router j is assigned to a controller or receives a penalty cost. Constraints (9.17) ensure that each router has a minimum of γ levels of backup controllers. Constraints (9.18) prevent a router from being assigned to the same controller at two different levels.

Constraints (9.19) and (9.20) are the load balancing constraints, from level 1 to level η. They mean that, until level η, each time a controller fails, its routers are reassigned to some controllers such that the difference between the most loaded and the less loaded controllers is, at most, δ. Constraints (9.21) are the constraints on the maximal latency between each pair of controllers. Finally, constraints (9.22) and (9.23) are required to define the variables $t_{ii'} \quad \forall i, i' \in C$.

Let $\varepsilon \in [0, 1]$, the objective function can be written again as

$$\min \ (1 - \varepsilon) N(y_i) + \varepsilon L(x_{ij}^{k}, z_j^{k})$$

This is a bi-objective problem that is solved in two steps:

- First, the problem restricted to our first objective $\min \sum_{i \in C} y_i$ is solved with an additional constraint on the maximal latency between a router and its controller. It provides the optimal number of controllers N^*.
- Then, the problem with the secondary objective

$$\sum_{j \in R} \sum_{k=1}^{C} \sum_{i \in C} d_{ij} x_{ij}^{k} (1 - p) p^{k-1} + l_{max} z_j^{k} p^{k-1}$$

Table 9.2 Parameters used for simulations

Parameters	Value
Number of nodes	[10, 80]
l_{max}	{3 000, 5 000, 7 000}
l_{cc-max}	7 000
δ	3
γ	2
η	1

- is solved, and, instead of the constraint on the delay (which is embedded in the objective function) a constraint to ensure that the number of controllers must be less or equal than the minimal one, N^* is added.

For numerical simulation, we have used instances from SNDlib, and for sensitivity analyses experiments, we have made around 3.000 simulations on random graphs. The parameters used for the simulations are summarized in Table 9.2. The failure probability has been fixed to a high value, $p = 0.95$, in order to obtain architectures with a lot of backup levels. Indeed, the lower p is, the fewer backup levels there are.

The evolution of the number of controllers on the number of nodes is plotted in Fig. 9.7 and the number of controllers on the number of arcs in Fig. 9.8. Obviously, the graph diameter increases with the number of nodes, resulting in the need for more controllers. At the opposite when graphs are denser (i.e., more arcs) there is a more feasible assignment between routers and controllers, leading then to fewer active controllers.

The optimal number of controllers depending on the length of the average shortest path is illustrated in Fig. 9.9. Each of the three colors corresponds to a set of solutions obtained with a specific value for the parameter l_{max} (3.000 in blue, 5.000 in green and 7000 in red). For high values of l_{max}, most of the solutions

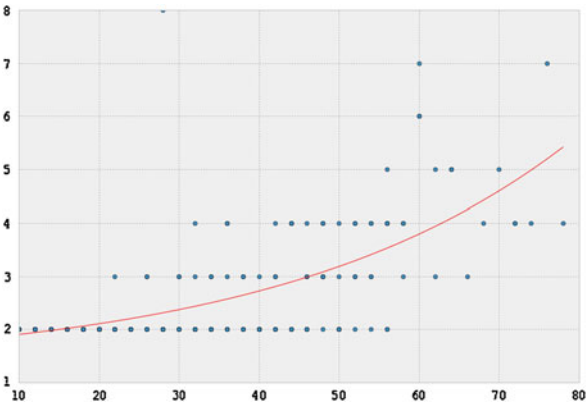


Fig. 9.7 Number of controllers on the number of nodes

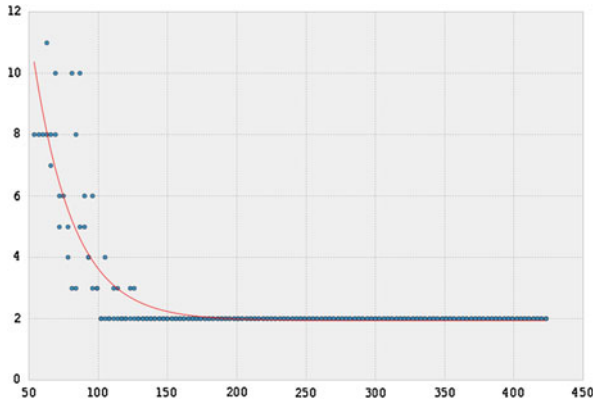


Fig. 9.8 Number of controllers on the number of arcs

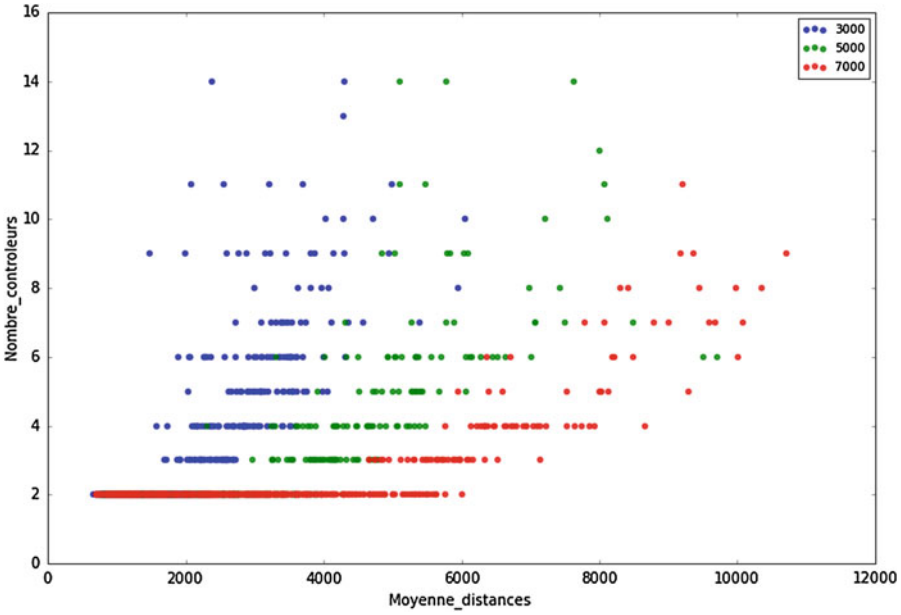


Fig. 9.9 Number of controllers on the average length of the shortest path in the graph and on maximal latency

required few controllers. In addition, the number of required controllers increases with the average length, until there is no more solution.

When the maximal latency is 3.000, up to 14 controllers could be required; while high maximal latencies give rise to a decrease of the number of controllers, up to the minimum threshold 2 (one backup controller).

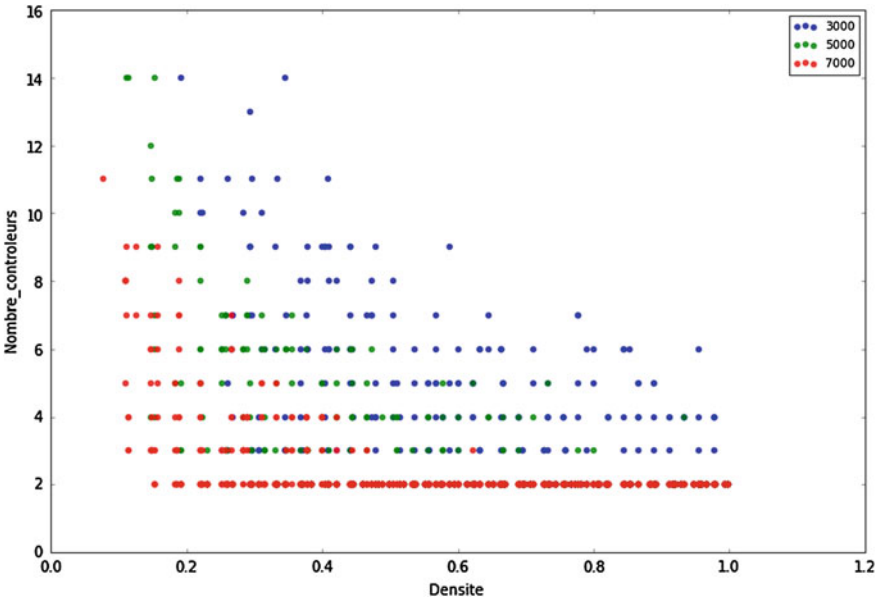


Fig. 9.10 Number of controllers on the graph density and on maximum latency

The increase of required controllers when the graph density decreases is illustrated in Fig. 9.10. For low values of maximal latency, there is no solution under a threshold (around 0.2) of graph density.

9.2.4 Concluding Remarks

In this work, new integer linear programming formulations are suggested for the Controller Placement Problem. The first suggested formulation is to provide the minimum number of controllers in a SDN-based network architecture. It takes into account the maximal bound on latency between routers and controllers, the maximal bound on latency between each pair of controllers, and a load balancing of the controller's sub-domains. This formulation has been extended to provide resilient SDN architecture. The solutions of the second formulation provide the optimal number of controllers, their placement within the network, their assigned routers and, finally, several levels of backup controllers in case of failures of the primary controllers.

Both random and realistic network instances, with up to 80 nodes, have been solved to optimality to test the formulations.

9.3 Geographical Load Balancing in SDN-Based Data Centers

The aim of this chapter is to demonstrate how SDN technologies can save energy, in particular in a network of Data Centers (DCs). More specifically, we want to answer the question of what are the benefits of a geographical load balancing algorithm between Data Centers. The SDN controller is a centralized point of the entire system, thus it is an enabler for implementing algorithms that provide an optimized solution over the whole data center network [1].

We consider a set of interconnected DCs with characteristics as follows:

- The DCs are located in different geographical zones, and particularly in different time zones.
- The DCs produce locally renewable energy, thanks to photovoltaic panels for instance. The power generated depends on the time of the day. Renewable energy will also be referred to as green energy in the following.
- Requests for videos are sent to each DS, and the latter can either process the request, or assign it to another DC. Requests can only be reassigned once.

Figure 9.11 is an example of a DC network, for a connectivity service, offered by Orange Business. The DCs are connected with high bandwidth Virtual Private Networks (VPN).

The two main leverages for energy savings are: the sleep mode of the servers inside the DCs and the green energy usage, where and when it is available. Our load balancing algorithm tries to makes the most of these two leverages. To evaluate the performance of the algorithm, we have developed and solved an explicit

Fig. 9.11 Example of network of DCs (from Orange Business-Together-as-a-Service platform architecture)

formulation of the global optimization problem. Thanks to this, we are able to know the energy consumption of the optimum load balancing.

This chapter is organized as follows: the problem statement and its formulation as a multi-period optimization problem are given in Sect. 9.3.1. Section 9.3.2 then describes the online algorithms. The numerical results are then presented and discussed in Sect. 9.3.3.

9.3.1 The Multi-Period Geographical Load Balancing Problem

In this subsection, a formal description of the Multi-Period Geographical Load Balancing (MP-GLB) problem is given.

9.3.1.1 Characteristics of the Problem and Notations

We model the data center network as a digraph G = (V, E), where the set of nodes V represents the DCs and the set of edges E the links that interconnect them. A link $(i, j) \in E$ corresponds to a physical path between two data centers i and j and its capacity, denoted cij, corresponds to the minimum of all residual capacities along the physical path. We express it in terms of number of requests. Let Pij be the set of links used along the path to connect DC i to DC j. As we consider only video services the delay is not as critical as for interactive services. We assume that, as long as the delay is under an acceptable value for the end-user who requested a video, it is not necessary to minimize it. Furthermore, we assume that all generated paths Pij respect the required delays, thus the delay constraints are relaxed from the model.

We assume a periodical evolution of the traffic and of green energy: typically a daily period. This period is composed of homogeneous time slots $t \in \{0, \ldots, T\}$, and $\boldsymbol{o_i}$ denotes the time offset of DC i. The video requests, that we consider in this study, are characterized by their default DC i, and by their duration (expressed in the number of time slots). Several classes $c \in 1, \ldots, C$ of video requests are defined, sorted by increasing order of their duration D = {d1, …, dc, …, dmax}, where dmax is the duration of the longest video. Then, the number of requests for class c sent by users attached to DC i during time slot t is denoted $\boldsymbol{S_{ic}^t}$.

Regarding the hardware equipment in each Data Center, it is assumed that the characteristics of the physical server are homogeneous, with a unique capacity M, which corresponds to the number of Virtual Machines (VM) that can be run simultaneously on a given server.

We denote the number of servers in DC i as $\boldsymbol{Mi}$.

In order to define the energy consumptions, the following notations are used:

- $\boldsymbol{E_i(t)}$ energy used by the non-IT infrastructure of DC i during time slot t (in W.h),
- $\boldsymbol{G_i(t)}$ green energy generated by DC i in the time slot t (in W.h),

- c_{VM} energy used for the creation of a VM (in W.h),
- e_{VM} energy used for the execution of a VM during a time slot (in W.h),
- e_{PR} energy used by a running physical server during a time slot (in W.h),
- e_{PS} energy used by a physical server in sleep mode, during a time slot (in W.h),
- e_P energy used by a physical server to go from sleep mode to running mode (in W.h),
- e_{ij} energy used for sending a request from DC i to DC j (in W.h).

We assume that the energy used by the non-IT infrastructure of DCs is made of a fixed part (for instance light and fans) and a part that depends linearly on the number of requests processed (for instance the air cooling). In consequence, we define $E_i(t)$ as

$$E_i(t) = E_i^{\min} + (E_i^{\max} - E_i^{\min})\left(\frac{R_i^t}{M_i M}\right)$$

where R_i^t denotes the number of requests to process, E_i^{min} the energy used when there is no request to process, and E_i^{max} the energy used at full load.

The green energy generated is based on a solar energy model and therefore depends on the local time of the day. We define it as

$$G_i(t) = G_i G[(t - o_i) \bmod 24]$$

where G_i denotes the total daily solar energy generated by DC i, and G(t) a function that is normalized, so that $\sum G(t) = 1$.

The number of requests that the users send is based on a daily traffic profile given by a function L(t), that is normalized so that $\sum L(t) = 1$. In consequence, we can write

$$S_{ic}^t = \alpha_c S_i L[(t - o_i) \bmod 24]$$

where αc is the part of requests for class c, and Si is the total daily number of requests sent to DC i by end-users.

9.3.1.2 Mixed Integer Linear Programming Formulation

We consider the following sets of variables: $\forall t \in \{0, \ldots T\}, \forall i \in \{0, \ldots |V|\}$,

- x_i^t number of VMs in DC I, at time t,
- $x_i'^t$ number of VMs created in DC I, at time t,
- y_i^t number of running physical servers in DC i, at time t,
- $y_i'^t$ number of physical servers leaving sleep mode in DC i, at time t,
- λ_{ij}^t number of requests of class c sent from DC i to DC j, at time t,
- R_{ic}^t number of requests for class c processed locally at time t at DC i,

- $\boldsymbol{R_i^t}$ number of requests processed locally at time t, at DC i,
- Fi(t) the non-green energy used by DC i at time t (these variables are introduced so that the problem stays linear)

With these notations, we can formulate the Multi-period Geographical Load Balancing Problem (MP-GLBP) as

$$min \sum_{i=1}^{|V|} \sum_{t=1}^{T} F_i(t) + \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} \sum_{c=1}^{C} \sum_{t=1}^{T} \lambda_{ijc}^t e_{ij} \quad \text{(OBJ)}$$

such that

$$R_{ic}^t = S_{ic}^t + \sum_j \lambda_{jic}^t - \sum_j \lambda_{ijc}^t \quad \forall i, \forall c, \forall t \tag{9.26}$$

$$R_i^t = \sum_{a=0}^{min(d_{max},t)-1} \sum_{d_c > a} R_{ic}^{t-a} \quad \forall i, \forall t \tag{9.27}$$

$$\sum_j \lambda_{ijc}^t \leq S_{ic}^t \quad \forall i, \forall c, \forall t \tag{9.28}$$

$$R_i^t \leq x_i^t \leq y_i^t M \leq M_i M \quad \forall i, \forall t \tag{9.29}$$

$$x_i^t - x_i^{t-1} \leq x_i'^t \quad \forall i, \forall t > 1 \tag{9.30}$$

$$y_i^t - y_i^{t-1} \leq y_i'^t \quad \forall i, \forall t > 1 \tag{9.31}$$

$$x_i^t e_{\text{VM}} + x_i'^t c_{\text{VM}} + y_i^t e_{\text{PR}} + y_i'^t e_P + (M_i - y_i^t) e_{PS} + E_i(t) - G_i(t) \leq F_i(t) \quad \forall i, \forall t \tag{9.32}$$

$$0 \leq F_i(t) \quad \forall i, \forall t \tag{9.33}$$

$$\sum_{a=0}^{min(d_{max},t)-1} \sum_{d_c > a} \sum_{I,J}^{st \cdot (ij) \in P_{IJ}} \lambda_{ijc}^{t-a} \leq c_{ij} \quad \forall (ij) \in E, \forall t \tag{9.34}$$

Constraints (9.26) express the fact that the number of requests that are processed in a DC at each time, must be equal to the number of requests that are sent by end-users, plus the number of requests sent from other DCs, minus the requests that are sent to other DCs, and this for each class of requests. Constraints (9.27) express the fact that the number of requests that are processed must be equal to the number of requests that are processed at time t, plus the number of requests that were processed at the previous timeslots, and still running. Please note that requests cannot move once they have started to be processed by a VM. The set of constraints (9.28) ensure that requests are migrated only once. The set (9.29) ensure that each

request is processed, and that each VM is hosted on a physical machine. The sets (9.30) and (9.31) make sure that VMs are created when necessary, and physical servers are turned on when necessary. The sets of constraints (9.32) and (9.33) make sure that green energy can only be used locally, where it is generated, and it cannot be stored. Finally, the set (9.34) are capacity constraints that make sure that the capacity limit of each link is enforced. In (9.34), the sum is over the requests migrated at the current time, but also from the previous times (for the requests that span more than one timeslot), therefore, the sum is over the index a. Moreover, we have to sum on all the paths that contain the edge (ij), that were pre-computed.

9.3.1.3 Geographical Load Balancing Period by Period (GLBPP)

By decomposing the MP-GLB problem for each period, we obtain T subproblems, that we call GLBPP problems. Each one solves the optimal mapping requests-DCs with no knowledge of the future requests. This variant is more realistic and it is used to estimate the performance of the results obtained with the online algorithms, described in the next subsection.

9.3.2 Online Algorithms

In this subsection, we present two online algorithms, that can be implemented in a real-world setting. The online algorithm is implemented in a Load Balancer (LB), which is itself a component of the SDN controller. The LB is, therefore, located in a logical central point of decision.

We consider that time is continuous, and that requests arrive according to a Poisson law (i.e., with exponential inter-arrival times). Each request is defined by a time of arrival and a duration (which is actually the duration of the requested video).

The online algorithm, upon the arrival of each request, decides by which DC it is going to be processed. The first algorithm is called LLB (Local Load Balancing) because it performs the local assignment, i.e., each request is always assigned to the local DC (recall that each request is received to a default DC). LLB is used as a reference in order to estimate the benefits of request migration. The second online algorithm is called GGLB (Green Geographical Load Balancing). It is based on the computation of the migration reward for each request.

We now introduce the following additional notations

- PVM power consumption of a VM (in Watt (W)),
- PPR power consumption of a running physical server (in W),
- PPS power consumption of a physical server in sleep mode (in W),
- Pi(R) power consumption of all the non-IT infrastructure of DC i, with R requests to process (in W),

- E1i (d; t) energy used by DC i for the treatment of one request of duration d, at time t (in W.h),
- C1i (d; t) non-green energy used by DC i for the processing of one request of duration d, at time t (in W.h),
- PGi (t) green power available at time t, at DC i,
- MR (d; t; i; j) (for Migration Reward), the non-green energy saved if a request of duration d is moved from DC i to DC j, at time t.

9.3.2.1 Online Algorithm GGLB

When a request of duration d arrives at the DC i at time t, the load balancer computes the MR (d; t; i; j) for all the DCs j $\in \{1, \ldots, |V|\}$. Note that for j = i, MR (d; t; i; i) = 0. MR (d; t; i; j) represents the potential gain of migrating the request from DC i to DC j, therefore it is written as

$$\mathrm{MR}(d,t,i,j) = C_i^1(d,t) - \left[e_{ij} + C_j^1(d,t)\right],$$

i.e., the cost of local processing minus the cost of processing at remote DC j plus the cost of transit (costs are expressed in terms of non-green energy consumption). The cost of processing a request can be computed thanks to the algorithm below. In a first step, the algorithm computes the energy consumption E1i (d; t), and then it computes the non-green energy consumption.

The load balancer assigns the request to DC j with the highest MR (d; t; i; j). In order to evaluate the performance of online algorithms, we have designed and implemented a simulation tool that is described below.

Algorithm 1 Computation of $C_i^1(d,t)$, incremental energy cost of processing a request

1: Computation of $E_i^1(d,\ t)$

(1) If at least one physical server is running and not fully loaded:

$$E_i^1(d,t) = P_{\mathrm{VM}}d + C_{\mathrm{VM}} + P_i(1)d$$

(2) If all the running servers are fully loaded:

(a) if there is a server in sleep mode, then E_i^1 is the same as above, augmented by the following amount: $C_P + (P_{\mathrm{PR}} - P_{\mathrm{PS}})d$

(b) If there is no more available server, then we set the cost as infinite, since this DC cannot process the request $E_i^1(d,t) = +\infty$

2: Computation of $C_i^1(d,t)$, knowing $P_i^G(t)$

(1) If the green power is sufficient to process the request, then $C_i^1(d,t) = 0$

(2) If the green power is not enough, then $C_i^1(d,t) = E_i^1(d,t) - P_i^G(t)d$

9.3.3 Results and Discussion

9.3.3.1 Input Values

In order to obtain numerical results for large instances, random instances were generated. The random values of the parameters indicated in the table below are generated with a Gaussian distribution. Some of the parameters are kept as fixed values. We have used two classes of requests, one class of duration 1 h (80% of the requests) and one class of duration 2 h (20%). The parameters oi (time zone offset) are determined randomly with a uniform distribution in {0, …, 23}. Parameters Si have random values according to a Gaussian distribution, and, by varying its average and its variance, we define four types of instance, as shown in the table below. Finally, the parameters Mi are set so that the DC i would reach a load of 80% at the traffic peak. For each type of instance, we have generated random instances by varying numbers of DC, from 10 to 70.

Figure 9.12 shows the traffic load functions L(t) and the solar energy production G(t), as defined in section II-A. The function L(t) was measured on an operational

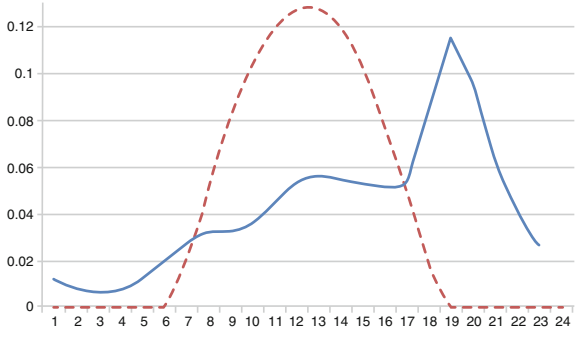


Fig. 9.12 Normalized functions L(t) (plain) and G(t) (dotted)

VoD (Video on Demand) service, over a 1-year period. We can observe that the peak of traffic occurs in the early evening, while the peak of solar energy production obviously occurs around noon.

9.3.3.2 Implementation Details

The MP-GLB and GLBPP problems have been implemented, respectively, in Java and in Python, thanks to the Pyomo library. Both problems are solved to optimality using the solver CPLEX.

The simulation tool in which algorithms LLB and GGLB run were implemented by us in Python. The simulation generates requests with exponential inter-arrival times, and the rate is determined such that the average daily number of requests is equal to Si. The simulation tool manages and keeps track of the state of each DC all the time, with the request assignment determined by the load balancing algorithm considered.

9.3.3.3 Results

The computational studies are made on a computer with 16 Xeons at 2.13 GHz and 32 MB of RAM. The largest instance, with 70 data centers, is solved within less than 20 s for the MP-GLB problem and less than 6 s for the GLBPP problem.

Figure 9.13 shows the gain of Geographical Load Balancing (in Y-axis) versus γ, the relative green energy total production (in X-axis). The gain of GGLB is computed as the relative difference of GGLB minus LLB, for each instance. The green

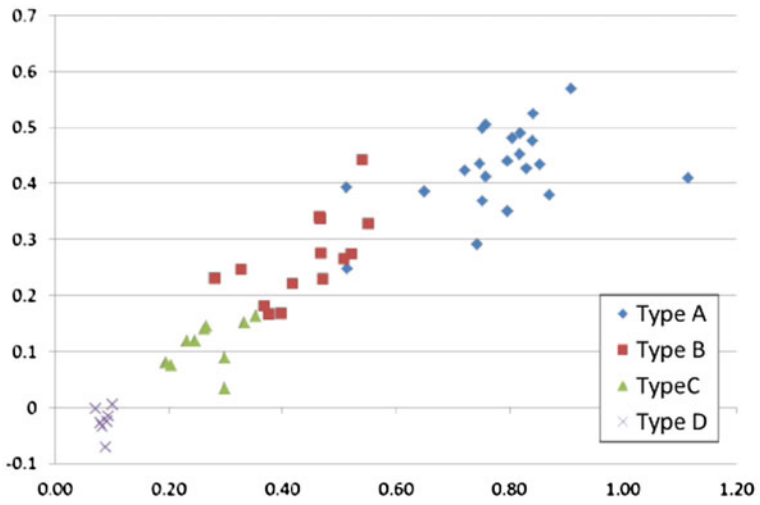


Fig. 9.13 Gain of GGLB (x-axis) verses γ (rel. green energy production)

Table 9.3 Random value inputs (in KW.h)

Input	$E_i^{\min}$	$E_i^{\max}$	e_{ij}	G_i
Average	1.5	3	2.10^{-5}	500
Variance	0.3	0.6	10^{-5}	300

energy is expressed as the ratio of the total green energy produced by all DCs, by the total energy consumed by the DCs. Each point on the figure represents a random instance, for which the number of DCs varies on a range from 10 to 70. Please note that when it is equal to 1 (or greater), then green energy is necessarily sufficient to process all the requests locally, because green energy is not necessarily available at the time requests arrive, thus resulting in lost green energy. It is interesting to notice the linear trend in the figure. It shows that the more green energy is available, the more beneficial is geographical load balancing. Another striking fact is that for lower values of green energy, the gain becomes negative, which means that it costs more to assign requests to remote DCs than locally (Tables 9.3, 9.4, 9.5, and 9.6).

The Table above sums up the results obtained for each instance type. The second column gives the values of the average. The third column gives the average relative gap of LLB to GGLB. It represents the relative gain of assigning requests to any DC, as opposed to only locally. The fourth column is the average relative gap of GGLB to MP-GLB. It represents the gap of our online algorithm to the optimal request assignment. The last column is the average relative gap of GLPPP to MP-GLB. It represents the gain of having a good knowledge of the future requests, as opposed to assigning the requests period by period.

Table 9.4 Fixed value inputs

Input	$\|V\|$	C_{VM}	e_{VM}	e_{PR}	e_{PS}	c_p	α_1	α_2
Value	100	10	6	300	50	100	80	20
Unit	nb	W.h	W.h	W.h	W.h	W.h	%	%

Table 9.5 Random inputs

Type of instance	A	B	C	D
Average $\boldsymbol{S_i}$	15,000	30,000	60,000	150,000
Variance S_i	7,000	15,000	30,000	70,000

Table 9.6 Summary of results

Type of instance	γ	LLB gap to GGLB (rel.)	(GLB gap to MP-GLB (rel.)	GLBPP gap to MP-GLB (rel.)
A	0.783	0.426	0.385	0.025
B	0.440	0.265	0.352	0.052
C	0.269	0.112	0.255	0.038
D	0.087	-0.025	0.043	0.017

When the green energy production is close to zero, the algorithm becomes counterproductive. This is due to the fact that there is a cost in assigning a request to a distant DC. The GGLB algorithm sometimes chooses a remote DC in order to exploit the green energy produced there, but the production is so small that it would have been used anyway by the local requests. In that particular case, assignment to a distant DC is not desirable. It must also be noted that, in that case, the gap to the optimal assignment is 4.3%, which means that meaningful gains are not possible anyway. However when it is high, the gain of our online algorithm is clear, and it goes up to 42%, when the green energy covers 78% of the energy needs. In that case, the gap of our algorithm to optimality is around 38%. This means that, even with the high gains we obtained, the assignment algorithm still can be improved.

Finally, the results show that knowing the future does not bring great benefits; the results are only improved by 2–5%, depending on the instance type. This means that using an online with prediction capabilities should not improve significantly the performance.

9.3.4 Concluding Remarks and Perspectives

In this chapter, we have described an efficient online algorithm to load balance the video requests between data centers, so that the carbon emissions are minimized. We have also proposed an integer linear programming formulation for the multi-period geographical load balancing problem. These mathematical formulations, and the one obtained by decomposing the problem for each period, have been solved to optimality, in order to evaluate the performance of our online algorithm. We have made simulations based on real traffic traces and on realistic instance parameters. These numerical results show that we could obtain a gain up to 42% of brown energy reduction thanks to smart load balancing in data center networks. This gain is achieved when the green energy production is high, which may not be the case as of today, but in the future, we expect that renewable energy will be more prevalent.

Since turning on and off servers too frequently can lead to premature wear and tear, we will design an online algorithm that requires waking each physical server a limited number of times per day. We will evaluate how this new constraint reduces the performance of GGLB.

In addition, while the performance of the proposed online algorithm is very good, it is still quite far from the optimal assignment. We will, therefore, investigate different assignment strategies that can lead to improved performance. Finally, this work could be easily extended to other services by restricting the delay bound while generating the paths related to the service.

9.4 Conclusion

In this work, we have suggested new formulations based on Integer Linear Programming for the Controller Placement Problem and the problem of geographic load balancing in a data center network.

We have solved the optimal and resilient controller placement problem with both random and realistic network instances with up to 80 nodes.

Besides, this work has shown that we can obtain a gain up to 42% of brown energy reduction thanks to smart load balancing in data center networks. This gain is achieved when the green energy production is high, which may not be the case as of today, but in the future, we expect that renewable energy will be more prevalent.

References

1. Carlinet Y, Perrot N (2016) Energy-efficient load balancing in a sdn-based data-center network. In 17th International Telecommunications Network Strategy and Planning Symposium (Networks), pp 138–143 (Sept 2016)
2. Dijkstra EW (1971) A short introduction to the art of programming. Circulated privately
3. Fallah H, Sadigh A, Aslanzadeh M (2009) Covering problem. In Hekmatfar RZ (ed) Facility location, ser. contributions to management science, pp 145–176
4. Floyd RW (1962) Algorithm 97: shortest path. Commun ACM 5(6):345
5. Guha S, Meyerson A, Munagala K (2001) Improved algorithms for fault tolerant facility location. In Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms, pp 636–641. Society for Industrial and Applied Mathematics
6. Heller B, Sherwood R, McKeown N (2012) The controller placement problem. In Proceedings of the first workshop on hottopics in software defined networks, ser. HotSDN'12, pp 7–12). ACM
7. Hock D, Hartmann M, Gebert S, Jarschel M, Zinner T, Tran-Gia P (2013) Pareto-optimal resilient controller placement in sdn-based core networks. International Teletraffic Congress (ITC), pp. 1–9. IEEE
8. Jimenez Y, Cervello-Pastor C, Garcia A (2014) On the controller placement for designing a distributed SDN control layer. In IFIP networking conference, pp 1–9
9. Lange S, Gebert S, Spoerhase J, Rygielski P, Zinner T, Kounev S, Tran-Gia P (2015) Specialized heuristics for the controller placement problem in large scale SDN networks. In International Teletraffic Congress, pp 210–218
10. Lange S, Gebert S, Zinner T, Tran-Gia P, Hock D, Jarschel M, Hoffmann M (2015) Heuristic approaches to the controller placement problem in large scale SDN networks. IEEE Trans Netw Serv Manage 12(1):4–17
11. Orlowski S, Pioro M, Tomaszewski A, Wessaly R (2007) SNDlib 1.0–survivable network design library. In Proceedings of the 3rd international network optimization conference (INOC 2007). Récupéré sur http://sndlib.zib.de
12. Perrot N, Reynaud T (2016) Optimal placement of controllers in a resilient SDN architecture. In 12th International Conference on the Design of Reliable Communication Networks (DRCN), Paris, pp 145-151
13. Ros J, Ruiz PM (2014) Five nines of southbound reliability in software-defined networks. In Proceedings of the third workshop on Hot topics in software defined networking, pp 31–36. ACM

14. Shen Z-JM, Zhan RL, Zhang J (2011) The reliable facility location problem: Formulations, heuristics, and approximation algorithms. INFORMS J Comput 23(3):470–482
15. Swamy C, Shmoys DB (2008) Fault-tolerant facility location. ACM Trans Algorithms (TALG) 4(4):51
16. Yao G, Bi J, Li Y, Guo L (2014) On the capacitated controller placement problem in software defined networks. IEEE Commun Lett 18(8):1339–1342

Chapter 10
Estimates of the Economic Impact of Energy Savings in the End-to-End Chain for Video Services

Kinga Pilarska, Bernard Liau and Nancy Perrot

10.1 Introduction and Presentation of the Approach

The objective of the European cooperative project CONVINcE (Consumption OptimizatioN in VIdeo Networks) was to reduce the energy consumption of video services with an End-to-End (E2E) approach, from the headend where videos are encoded to the terminals where they are consumed, while considering the core and access networks. The purpose of this analysis is to provide an evaluation method of the energy savings obtained by the project research results by comparing State-Of-The-Art (SOTA) reference scenario with scenarios including technical power savings proposals developed and tested by CONVINcE project [1–5]. Comparison of the scenarios allows positioning possible savings with the total energy consumption and give relative savings per component of the E2E chain. In this study, the analysis focuses on the Internet Protocol Television (e.g., Orange Video on Demand) and on the Over The Top (e.g., Netflix, YouTube).

IPTV is a service provided by a video provider with a guaranteed quality of service (QoS). The image definition is generally of high quality. Transport to the end-user is made through a network that guarantees quality rate. The service is usually directly paid by the end-user. The video is downloaded, encoded in a single, terminal independent HQ format .

On the contrary OTT is a video downloaded from the Internet, without QoS guarantee. Transport to the end-user is made through the Internet network of the

K. Pilarska (✉)
Orange Labs Polska, Aleje Jerozolimskie 160, 02-326 Warsaw, Poland
e-mail: kinga.pilarska@orange.com

B. Liau · N. Perrot
Orange Labs Networks, 44 avenue de la République, 92 326 Chatillon, France
e-mail: bernard.liau@orange.com

N. Perrot
e-mail: nancy.perrot@orange.com

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0_10

service provider. In the headend multiple versions are encoded according to the type of terminal. This is a mass service, which involves a lot more traffic than IPTV.

The methodology presented in the chapter explains the way we calculate power consumption for one service. Based on CONVINcE project results the analysis provides an example of the achieved savings. This chapter does not explicit in detail what these proposals are, but analyses globally their possible impact. It gives an assessment of the energy savings they can obtain in relative value for the video services and gives an order of magnitude of their economic stake at the level of European countries. Convince project suggested following scenarios:

- terminals and headend—the new coding modes [5, 6];
- network—to store the most frequently viewed videos close to the final user with the aim of saving transportation resources (edge cloud deployment), and to turn off unused transport systems outside busy periods of network using the routing functions of the IP network (SDN/NFV deployment) [4, 7].

We use, for the network calculation, Next Generation Point of Presence (NGPoP) solution which hosts, in particular, the IP edge for all access types (fixed, Wi-Fi, and mobile) and distribute content closer to the user at the boundary between aggregation and core networks. We take into account also Geographical Load Balancing (GLB) solution, which provides more savings [7].

The final objective of the outlined methodology is thus to give an order of magnitude of the possible savings for energy consumption.

To reach this objective, we suggest to analyze the impacts of the project on a unit of video service (for example, an average downloaded video for IPTV and OTT. The idea is to incrementally compare the results of the project with the current scenario (SOTA).

By analogy with studies on costs, we aim at using a top-down model. Starting from the observed energy consumption of a unit of video, we rebuilt the basic elements of energy consumption. The main steps are:

(1) From real figures of energy consumption of a video unit: share it on the network elements of the chain, including terminals, network (typically, for the network: core network, backhaul and access), headend and cloud;
(2) analyze globally the impact of the project results on each of these elements;
(3) decrease or increase the energy consumption for each one;
(4) sum again on the whole chain.

As energy cost deeply depends on the energy tariffs within the particular country, transformation of kWh into Euro is done at the end.

The analysis is done for an architecture already in place (SOTA or proposed scenarios). Details on transition between the existing scenario (SOTA) toward proposed ones are not part of this chapter.

One of the difficulties of the exercise is to define the perimeter of the analysis:

- What size for the network: country wide, European wide, or worldwide?
- What topology?

- Supporting which services (network elements are always shared among a lot of services)?

To avoid this difficulty, we reduce the problem to a unit of service supposed to support the consumption of energy. From the experience at Orange in calculating cost of one gigabyte of data in the European affiliates of the Orange group, which have certain stability despite the differences in scale, network architecture, or customer usages, we assumed the same stability applies to energy consumption in the network supporting the same units of traffic, at least in the same proportions. Thus, we suppose that the resulting consumption of energy is quite stable for European countries (we attempt to give bounds). The calculations are based on energy consumption of basic elements including implicitly network topology, engineering rules, sharing with other services (for example, by using an average energy consumption to download a gigabyte of data in the core network). The assumption is that these basic consumption figures have the same order of magnitude, independently of the size and topology of the network. As the considered figures correspond to the network of a specific operator, we aim to have a precision lower than a ratio of 10.

10.2 Definition of a Service Unit to Support the Costs

For IPTV and OTT, we define an average video (denoted DurationVideo) by its duration in hours of a video, and by its average volume (denoted VolumeVideo) in terms of downloaded data (in Giga Byte). The calculations take the form below, where items in italic characters are assumed to be data observed from measurements.

$$\text{VolumeVideo[GB]} = \mathit{VolumeVideoTV}[\mathit{GB}]*\mathit{percentTV} + \mathit{VolumeVideoPC}[\mathit{GB}]*\mathit{percentPC} + \mathit{VolumeVideoSM}[\mathit{GB}]*\mathit{percentSM}$$

Where:

- *VolumeVideoXX* [*GB*] represent the volume in Gigabyte of an average video on terminal *XX* [XX: TV set (*TV*), personal computer (*PC*), or smartphone or tablet (*SM*)],
- *percentXX* [*GB*] is the frequency of usage by customers.

Note that: $\mathit{percentTV} + \mathit{percentPC} + \mathit{percentSM} = 100\%$

The average duration of a video may also depend on the terminal:

- *DurationVideoTV[hour]* for TV sets,
- *DurationVideoPC[hour]* for personal computers,
- *DurationVideoSM[hour]* for smartphones and tablets.

$$\text{DurationVideo[hour]} = \mathit{DurationVideoTV}[\mathit{hour}]*\mathit{percentTV} + \mathit{DurationVideoPC}[\mathit{hour}]*\mathit{percentPC} + \mathit{DurationVideoSM}[\mathit{hour}]*\mathit{percentSM}$$

We suppose VolumeVideoXX, BitrateVideoXX, and percentXX are available input data.

In the following calculations, we make the following assumptions for IPTV and OTT. It reflects practical usage experience.

$$VolumeVideoTV[GB] = 1$$
$$VolumeVideoPC[GB] = 0,5$$
$$VolumeVideoSM[GB] = 0,25$$

$$DurationVideoTV[hour] = 1$$
$$DurationVideoPC[hour] = 0,5$$
$$DurationVideoSM[hour] = 0,25$$

For OTT we take into account way of watching free YouTube videos, that is why for IPTV and OTT we take a different percentage of usage.

For IPTV:

$$percentTV = 50\%$$
$$percentPC = 25\%$$
$$percentSM = 25\%$$

For OTT:

$$percentTV = 20\%$$
$$percentPC = 45\%$$
$$percentSM = 35\%$$

We obtain then the characteristics of an average video: (Table 10.1)

Table 10.1 Characteristic of an average video for IPTV and OTT video

	DurationVideo (hour)	VolumeVideo (GB)
IPTV	0,7	0,7
OTT	0,5	0,5

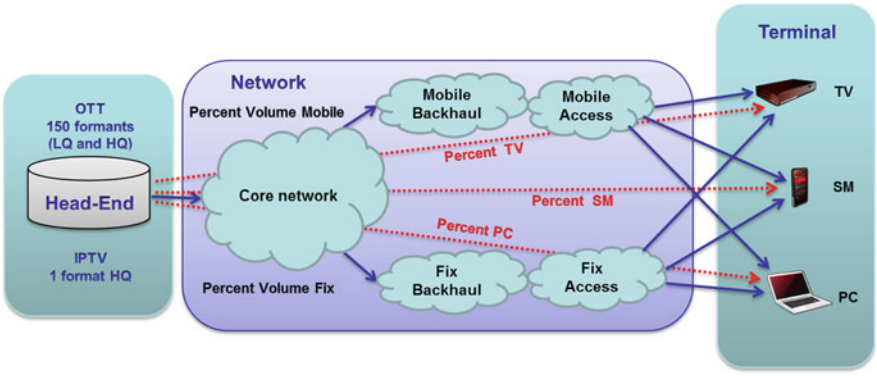


Fig. 10.1 Architecture model for IPTV and OTT service

10.3 Energy Consumption Models in State-of-the-Art Scenario

10.3.1 Simplified End-to-End Architecture for IPTV and OTT

The analysis is based on the simplified reference architecture (Fig. 10.1) [3], mainly:

- Headend located within the video data center,
- Network composed of an IP-based national core network,
- Fiber IP/GE aggregation network,
- Wireless and wired accesses,
- Terminals.

At headend, videos for IPTV service are generally only coded once in a format HQ. For OTT videos, on the contrary, many formats are encoded (until 150) adapted to each kind of terminals.

In practice, videos are downloaded via two access networks, fixed and mobile, on different kinds of terminals (some of them reached by both types of accesses).

For OTT video, the IP core network is the Internet network carrying the data traffic from the Internet (unmanaged IP network). For IPTV, the core network is a specific network (dedicated to traffic in connected mode) that provides a level of traffic protection and a level of quality of service higher than in the IP transport network of the Internet. This network generally overlaps the IP transport network (directly used to carry Internet traffic) over which it is created; it is based on VPN dedicated to the connected mode traffic involving VPN routers, it is operated with traffic engineering different from those operating the IP network in background.

10.3.2 Methodology to Calculate the Energy Consumption of an Average Video in SOTA Scenario

The average consumption of a video is calculated from the average consumption in the different elements of the end-to-end chain (headend, network, and terminals). The calculation method differs according to these elements. For the network elements, it depends on consumption units related to volume of downloaded data (in networks, costs are typically supported by gigabytes of data and minutes of voice). The calculation is thus carried out from the average consumption to download a gigabyte of data, weighted by the usage to download a video (which may differ for IPTV and OTT video). For terminals and headends it depends on consumption units related to durations of use (1 h of video seems the most natural unit to compare different technologies of terminals). For the various network elements, the calculation is thus carried out from the average consumption to download a gigabyte of

data, weighted by the usage to download a video. For the terminals and headends, the calculation is carried out from the average consumption of processing 1 h of video.

$$\begin{aligned}\text{ConsoVideo[Wh]} = \text{VolumeVideo[GB]} * \text{Conso1GBNetwork}\left[\frac{\text{Wh}}{\text{GB}}\right] + \text{Conso1hourTerminal[Wh]} \\ * \text{DurationVideo[hour]} + \text{Conso1hourHeadEnd[Wh]} * \text{DurationVideo[hour]}\end{aligned}$$

$$\begin{aligned}\text{Conso1hourTerminal[Wh]} = \text{Conso1hourTV[Wh/hour]} * \textit{durationVideoTV}[\textit{hour}] * \textit{percentTV} \\ + \text{Conso1hourPC[Wh/hour]} * \textit{durationVideoPC}[\textit{hour}] * \textit{percentPC} \\ + \text{Conso1hourSM[Wh/hour]} * \textit{durationVideoSM}[\textit{hour}] * \textit{percentSM}\end{aligned}$$

Where *VolumeVideo* is the average volume of a video downloaded by all kinds of terminals given in Sect. 10.2.

Basic energy consumptions of the elements of the chain are made explicit in Sects. 10.3.3, 10.3.4, and 10.3.5.

10.3.3 Terminals Including Sharing with Other Services

10.3.3.1 Smartphones and Tablets

A smartphone or a tablet is usually used for a large amount of use cases [5, 6, 8]. Hence, such terminals have an average power consumption even when they are not used for video services. We denote this power consumption as "background consumption". This background consumption can be the result of using the terminal for voice calls, messaging, social networking, gaming, web browsing, etc. However, on top of the background consumption, we consider here, specifically, the usage of smartphones and tablets for video usage.

For downloading a video on mobile terminals we consider the data to be delivered either over a cellular network (e.g., LTE) or local area network (Wi-Fi). The available data rates over the radio interface for these network types are typically in the same order of magnitude, e.g., $\sim$100Mbps. However, the available data rate of a single mobile phone at a given time slot could be different, depending, among others, on the radio environment and on the radio access load. The video download can utilize a dynamic rate selection protocol, e.g., MPEG-DASH to adapt the selected media data rate/quality to the available link data rate. Codec H265 and H264 are typically used for video, where the H265 codec needs in the order of 50% less data than H264 for the same video quality.

Within CONVINcE state of the art, we consider different power consumptions for smartphones and tablets respectively, mainly due to larger average display sizes

of tablets. We can consider the consumptions of energy to incrementally process 1 h video, on top of the background consumption for a smartphone or a tablet, are:

$$\boxed{\begin{array}{c}\text{Conso1hourSmartPhone[Wh]} = 1.23\\ \text{Conso1hourTablet[Wh]} = 3.5\end{array}}$$

Assuming a penetration ratio between smartphones and tablets of 25% for video we obtain, for the sake of simplicity we assumed the same ratio for IPTV and OTT:

$$\boxed{\text{Conso1hourSM[Wh]} = 1.8}$$

Note that the capacity of a Smartphone battery is about 11 Wh.[1] This means that a smartphone when watching video works during 9 h (simply by dividing 11 Wh by 1.23 W), which corresponds to the experience of everyone.

10.3.3.2 Personal Computers

The PCs power consumption differs a lot, based on hardware specifications such as CPU, GPU, screen technology and size, hard disk drive, etc. However, we can here typically focus on a state-of-the-art laptop, targeting mainstream performance. Similar as for smartphones and tablets, the average usage is widespread in many different use cases besides video usage. We here consider the additional power consumption on top of such background usage. Basic consumption of energy to incrementally process 1 h video on a PC supposed to be already connected is assumed to be:

$$\boxed{\text{Conso1hourPC[Wh]} = 27}$$

10.3.3.3 Set-Top Boxes

A TV set is connected to the network through a Set-Top Box (STB) . Just like other terminals, the power consumption of the STB varies depending on the CPU, GPU and other parameters of the hardware used. In this project, a specific terminal is used as reference STB which is equipped with a Broadcom 7250 chipset [5]. The average power consumption can be assumed to be 30 Wh.

Analyses made by Orange reveals that a given number of customers (15%) do not switch off their STB after watching TV. The STB is set in sleeping mode. The consumption is then reduced to 3 Wh (10% of the consumption in activity). For these STB, we only count the incremental cost of activity, let say 27 Wh. The final figure is hence given by:

$$\boxed{\text{Conso1hourSTP[Wh]} = 29.6}$$

[1]A typical smartphone battery size is centered around $\sim$ 3Ah which corresponds to $\sim$ 11Wh.

10.3.3.4 TV Sets

The consumption of TV sets (the terminal is supposed to be on at the beginning of the video and off at the end) to process 1 h video is [5]:

$$\boxed{\begin{array}{c}\text{Conso1hourTV}[\text{Wh}] = 63 \text{ for } 40'' \text{ TV Sets} \\ \text{Conso1hourTV}[\text{Wh}] = 93 \text{ for 4K } 40'' \text{ TV Sets}\end{array}}$$

We suggest the following sharing: 67% for HD 40″ TV Sets and 33% for 4 K HEVC 40″ TV Sets. The average energy consumption is thus:

$$\boxed{\text{Conso1hourTV } [\text{Wh}] = 72.9}$$

10.3.3.5 Average Consumption Per Terminal

The average consumption of terminals can be easily calculated from the average usage in time (see formula in Sect. 10.3.2):

$$\boxed{\begin{array}{l}\text{IPTV:} \\ \qquad \text{Conso1hourTerminal}[\text{Wh}] = 77 \\ \text{OTT :} \\ \qquad \text{Conso1hourTerminal}[\text{Wh}] = 52.1\end{array}}$$

And for an average video:

$$\boxed{\begin{array}{l}\text{IPTV:} \\ \qquad \text{ConsoVideoTerminal}[\text{Wh}] = 52.9 \\ \text{OTT :} \\ \qquad \text{ConsoVideoTerminal}[\text{Wh}] = 26.7\end{array}}$$

10.3.4 Network

For the analysis of energy consumption, the network has been divided into macro elements compatible with the block diagram of Fig. 10.1.

For mobile networks:

- Radio access network (including 3GPP macro and micro cells, WiFi),
- Mobile backhaul,
- Mobile core platforms (EPC).

For the wired networks:

- Copper and optical local loop,
- DSLAM, MSAN access equipment for data services,
- Backhaul (based on GE routers and fiber).

IP core transport network is common to both wired and wireless networks.

$$\begin{aligned}\text{Conso1GBNetwork}\left[\frac{\text{Wh}}{\text{GB}}\right] &= \text{Conso1GBCoreIP}\left[\frac{\text{Wh}}{\text{GB}}\right] \\ &+ \left(\text{Conso1GBMobileBackhaul}\left[\frac{\text{Wh}}{\text{GB}}\right] + \text{Conso1GBMobileAcces}\left[\frac{\text{Wh}}{\text{GB}}\right]\right) * \text{percentVolumeMobile} \\ &+ \left(\text{Conso1GBFixBackhaul}\left[\frac{\text{Wh}}{\text{GB}}\right] + \text{Conso1GBFixAccess}\left[\frac{\text{Wh}}{\text{GB}}\right]\right) * \text{percentVolumeFix}\end{aligned}$$

Where:

- *Conso1GBXX* defines the energy consumption to download one Gbyte of data in the network element XX (XX = Core IP, Mobile access and backhaul, Fixed [wired] access and backhaul), values are given in column 4 "energy consumption to download 1 Gbyte" of Table 10.2.
- *percentVolumeMobile* and *percentVolumeFix* are respectively the percentage of data traffic downloaded by wireless and wired users, respectively 20% and 80%, as it appears in column 3 of Table 10.2.

Energy to carry 1 Gbyte of data was calculated using real measurements in Orange France Network, extrapolated for the year 2020. The scope is data networks, including 3G and 4G technology for mobile access network and DSL copper or optical access for the wired network. The equipment dedicated to voice services or legacy data services is not involved. To give an idea of the distribution of energy consumption in a network among its main elements, the consumptions are given in

Table 10.2 Network basic energy consumption for 1 GByte downloaded (2020)

1	2	3	4
Network element	Energy consumption forecast for 2020 on a basis 100	Percent of concerned network traffic (%)	Energy consumption to download 1 Gbyte (Wh)
Radio access network 3G and 4G	33	20	**122**
Mobile packet core and backhaul	8	20	**32**
Fixed access (copper and fiber) and backhaul	39	80	**37**
IP core network	17	90 unmanaged	**13**
		10 managed	**16**
Cloud	3		**2**

column 2 of Table 10.2 as a percentage of the total consumption. The energy consumption is quite stable during the year. The conclusion drawn from these assessments of energy consumption is a trend to stability, regardless of traffic growth, new technologies consume less, which counterbalances the traffic growth.

For every part of the network, the consumption has then been divided by the traffic forecasted during the period. Of course, the traffic volume involved is different for each of the network segments as expressed in the column "percent of concerned network traffic." Traffic towards wireless users is forecast to represent 20% of the total data traffic (about 10% in 2016, meaning a yearly increase of about 60% for mobile users).

Traffic for IPTV is routed through a VPN overlay network. This network requires additional routers. The traffic engineering and dimensioning rules induce an average usage of the VPN 20% lower than in the IP network systems. We concluded that the consumption for carrying a gigabyte of IPTV requires 20% more energy than the equivalent in the IP network.

To summarize trends, energy consumption of the involved network elements between 2016 and 2020 is forecasted to be relatively constant, including changes in technology and evolution of usage. However, data traffic is sharply increasing during the same period with an overall growth of 40% per year. This means that a large part of the energy consumption is quite independent of the usage and the consumption of energy of a video directly decreases as the number of videos increases.

Consumption for 1 Gbyte is given in Table 10.2, column 4 "energy consumption to download 1 Gbyte."

To summarize trends, energy consumption of the involved network elements between 2016 and 2020 is forecasted to be relatively constant; including changes in technology and evolution of usage (new technologies consume less, which counterbalances part of the traffic growth). However, data traffic is sharply increasing during the same period with an overall growth of 40% per year. This means that a large part of energy consumption is quite independent of the usage and it directly decreases as the number of video increases.

It should be noted that energy is rather used in the access, particularly for wireless access. The core network represents only 17% of the full network consumption and for cloud (NGPoP solution) 3% (see column 2 "energy consumption forecast for 2020 on a basis 100" Table 10.2).

Note that for an average Gigabyte of downloaded video network usage is different for IPTV and OTT (IPTV is most often viewed on television sets and, therefore, relatively uses more fixed access and less mobile access than OTT).

With the given assumptions, using the model above, the energy consumption for one Gigabyte of video is:

$$\begin{array}{l} \text{IPTV}: \\ \qquad \text{Conso1GBNetwork} = 76.5\text{Wh} \\ \text{OTT}: \\ \qquad \text{Conso1GBNetwork} = 79.6\text{Wh} \end{array}$$

The energy consumption in the network for an average video is:

$$\begin{array}{l} \text{IPTV}: \\ \qquad \text{ConsoVideoNetwork} = 52.6\text{Wh} \\ \text{OTT}: \\ \qquad \text{ConsoVideoNetwork} = 40.8\text{Wh} \end{array}$$

10.3.5 Headend

Basic energy consumption models for video headends have been introduced in the CONVINcE project. For IPTV and OTT services, we supposed that videos are mainly delivered by IPTV headend.

The model calculates the energy consumption per transcoding unit. Fixed costs related to redundant units, management systems, scramblers, etc. represent about 15% of the total headend consumption.

For IPTV:

Taking into account the nature of video channels, a more accurate model is:

$$HEConsumption_{(Wh)} = FixedPartCons_{(Wh)} + \left(SDChannelCons_{(Wh)} \times NumberOfSDChannels\right) + \left(HDChannelCons_{(Wh)} \times NumberOfHDChannels\right)$$

The energy consumption is supposed to be quite linear with the time of encoding. For IPTV, only one high quality format is encoded. For OTT video, the same video is transcoded several times to have versions corresponding to the various kinds of terminals (typically 150 versions for the same video). Considering that a video is watched by a certain number of users, we obtain the average energy consumption related to a video usage by customer Table 10.3.

We note that this consumption is negligible compared to network and terminal consumptions.

Table 10.3 Average consumption of headend

	IPTV	OTT
Consumption for 1 average video [Wh]	372	197
Number of versions coded for a single video [unit]	1	150
Average popularity [user]	100 000	500 000
Consumption for 1 h video per user [Wh]	0,005	0,162
Consumption for 1 video per user [Wh]	0,004	0,059

10.4 Dedicated Architecture for Video—Suggested Improvements

This section briefly describes the main impact of the suggested architectures of CONVINcE project for the energy consumption for IPTV and OTT [2].

10.4.1 Terminals

Studies on potential savings for terminals include more efficient utilization of the wireless radio interface in the communication between network and terminals.

First analysis shows that the reduction of power consumption for video usage over a cellular radio access network such as LTE could be achieved by a collaborative architecture sharing information between terminals and the mobile access network [5, 8]. Other studies refer to the more efficient usage of the Wi-Fi interface, utilizing a power-saving mode [9]. The savings on mobile terminal consumption are typically in the order of 5%.

For the fixed terminals, namely the STBs and TV Sets, the decoders on the hardware and the screens are the two major components responsible for power consumption. Furthermore, the effective usage of the operating modes is one of the factors of reducing the average power consumption. Overall, the expected power consumption reductions for a video are about 10% in STBs and 15% in TV Sets, considering the total terminal average power consumption during active video usage. We estimate that the potential savings due to PCs are from 43% up to 65%. Then, the terminal energy consumptions, for an average hour of video and for an average video, are given (column 5) in Table 10.4 for IPTV and in Table 10.5 for OTT.

Table 10.4 Energy consumption for units of IPTV

1	2	3	4	5	6
	SOTA (Wh)	Network improvements (Wh)		Terminal improvements (Wh)	All (Wh)
		Including SDN/NFV	Including CDN	~17% on terminals	
Conso_ lhour	**156**	**151**	**153**	**143**	**136**
Conso_1hour_Terminal	79,6	79,6	79,6	66,3	66,3
Conso_1hour_Network	76,5	71,7	73,4	76,5	70,0
Conso_1hour_HeadEnd	0,01	0,01	0,01	0,0	0,01
Conso_Video	**107,3**	**104,0**	**105,1**	**98,2**	**93,7**
Conso_Video_Terminal	54,7	54,7	54,7	45,6	45,6
Conso_Video_Network	52,6	49,3	50,4	52,6	48,1
Conso_Video_HeadEnd	0,00	0,00	0,00	0,00	0,00

10.4.2 Network

CONVINcE project considered two scenarios of network evolution [7].

- Edge cloud;
- Software Defined Networking, in addition to the use of Network Function Virtualization (SDN/NFV) technology.

Note that the impact of both solutions edge cloud and SDN/NFV on the new network architectures is cumulative, by introducing simultaneously an edge cloud architecture and an SDN/NFV architecture on top of a content distribution architecture.

10.4.2.1 Edge Cloud

The edge cloud architecture allows storing contents in the edge of the network, nearby the user. As a consequence, the data traffic in the IP/MPLS core IP network decreases.

The decrease of traffic volume depends on several factors, among them the popularity of the requested contents, the possibility to catch and store them, and the performance of the caching algorithms. In practice, today, the performance of transparent cache observed in Orange affiliate networks is about 30% of traffic saved upstream of the cache, for OTT video; and it rises to 40% for IPTV.

This scenario should multiply the number of storage sites and the total volume of stored data. Today, based on NGPoP solution improved by Orange, we have only estimation of the required energy consumption for storing data in a cloud. However, most of the energy used while delivering contents is due to the video downloading and, in both scenarios (SOTA and Edge Cloud). Assuming that the number of downloads will be the same, in this analysis, we neglect the difference between both scenarios taking into account power consumption in the cloud.

Supposing that globally 30% of the traffic is saved (all services), this may have an impact on the dimensioning and design of the core packet network. Due to the modularity of transport systems, the impact on the network dimensioning is not linear. However, due to the high volume of traffic impacted, it should have roughly a linear trend. As we are looking for an order of magnitude in energy saving, we suppose that decreasing by 30% the traffic volume in the core network induces a decrease of 30% in network dimensioning and consequently a decrease of 30% in the energy consumption of the core IP network. It corresponds to a decrease of 30% for OTT videos and to a specific decrease of 40% for IPTV that is more impacted by caching [7].

The respective network energy consumptions for an average Gigabyte of video and for an average video are given in Table 10.4 (column 4) for IPTV, and Table 10.5 for OTT video.

Table 10.5 Energy consumption for units of OTT video

1	2	3	4	5	6
	SOTA (Wh)	Network improvements (Wh)		Terminal improvements (Wh)	All (Wh)
		Including SDN/NFV	Including CDN	~25% on terminals	
Conso_1hour	**132**	**128**	**130**	**119**	**114**
Conso_1hour_Terminal	52,1	52,1	52,1	39,0	39,0
Conso_1hour_Network	79,6	75,6	77,2	79,6	74,5
Conso_1hour_HeadEnd	0,16	0,16	0,16	0,2	0,16
Conso_Video	**67,6**	**65,5**	**66,4**	**60,9**	**58,2**
Conso_ Video_Terminal	26,7	26,7	26,7	20,0	20,0
Conso_Video_Network	40,8	38,7	39,6	40,8	38,2
Conso_Video_HeadEnd	0,08	0,08	0,08	0,08	0,08

10.4.2.2 SDN/NFV

SDN/NFV architecture allows controlling the routers thanks to a centralized control plane. Then, the network transport resources could be optimized, switching off the routers when the network is under loaded, typically during off-peak periods [7].

Figure 10.2 gives the trend of the daily data traffic profile in the IP core network of Orange France. During a day period, the carried volume of traffic is under two-third of the volume reached at the busy period at least during 12 h, half of this volume during 10 h and one-third of this volume during 5 h. Thus, there is a potential energy saving in the IP/MPLS core network.

We suppose that during this period of time, it will be possible to switch off equipment up to two-third of the core IP systems (not above this threshold due to problems of connectivity within the network), linearly to the decrease of traffic. As a consequence, it leads to 30% of saving of the energy consumed in IP transport network during the day.

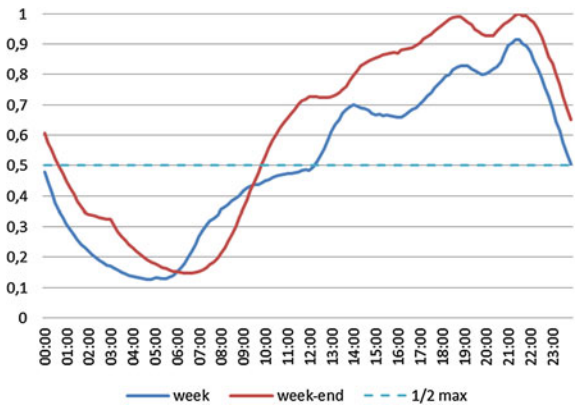


Fig. 10.2 Daily profile of traffic in the IP core network

The respective network energy consumptions for an average Gigabyte of video and for an average video are given (column 3) in Table 10.4 for IPTV and Table 10.5 for OTT video.

10.4.3 Headend

We have not analyzed possible improvement for headend because the energy consumption was 0,004Wh for IPTV and 0,08Wh for OTT. Compared to the other elements of the chain it is very small.

10.4.4 Global Evaluation of Potential Savings

The average energy consumption for the SOTA scenario and for the various proposed improvements on the E2E chain is illustrated in Table 10.4 for an average IPTV and in Table 10.5 for OTT video. Results are given with components for each one of the end-to-end segments (terminals, network and headend), for 1 h of video (Conso_1hour) and for an average video (Conso_Video) as defined in Sect. 10.3.2.

Applying all CONVINcE solutions together for network and for terminals would lead to a 13% decrease of energy consumption in the E2E chain for IPTV and to a 14% decrease for OTT video. Note that most of this energy consumption comes approximately equally from the terminals (50%), 49% comes from the network; the energy consumption in headends is relatively insignificant (less than 1%).

The difference in energy consumption between 1 h of IPTV and 1 h of OTT (higher for IPTV) is mainly due to the difference in volume of data needed to encode the video (significantly higher for IPTV). IPTV more often uses television sets that consume more energy than mobile devices. However, OTT videos more often use the mobile access network that consumes more than the wireline access.

Relative savings from terminals are eventually limited (17% for IPTV and 25% for OTT). Most of the savings in the network (8% for IPTV and 6% for OTT) comes from the core network that remains 2nd order compared to wired and wireless accesses.

10.5 Economic Model: How to Pass from Energy to Euros

To quantify the economic gain an European scale has been chosen. From an estimation of IPTV and OTT downloaded per country, we aim at estimating the corresponding savings in terms of energy. Considering the average cost of energy per country, we deduce savings in term of cost, per country, and globally.

For the number of videos considered, the approach was the following. Assumptions about the number of IPTV and OTT were made for France, as Orange had the best statistics for this country. Figures were then extrapolated to other European countries, using demographics and standard of living indicators.

Average earnings in energy by video are assumed to have the same order of magnitude for all the concerned countries. Cost of energy was derived from public sources accessible on the Internet.

10.5.1 Calculation of Video Volume Forecast and Savings for IPTV and OTT Per Country

Based on Orange France experience we presume that in 2020 IPTV will represent 10%, and OTT 80%, of the total traffic downloaded by users. The rest of 10% is reserved for other video traffic. Supposing that Orange France has 40% of the market share, the volume of IPTV calculated for France population is in the order of scale of 4.4 GB x 10^9 videos in 2020 and the volume of OTT videos 47.3 GB x 10^9 videos. The total quantity of watching videos for other European countries is supposed to be proportional to population and to the "purchasing power parities" factor representing the standard of living within the country (from OECD [10]).

The volume of videos per country is presented in Table 10.6 (columns 4 and 6). Energy savings obtained by CONVINcE solutions are for IPTV in column 5, and for OTT in column 7 of Table 10.6.

10.5.2 Energy Consumption in the Edge Cloud in France

Orange uses Next Generation Point of Presence (NGPoP) cloud solution, that hosts, in particular, the IP edge for all access types (fixed, Wi-Fi, and mobile) and distributes content closer to the user at the boundary between aggregation and core networks (Table 10.7).

Based on the experience of Orange affiliates (Orange Poland) and forecasted launching of NGPoP solutions in Orange France, we measure the volume of video downloaded in the Network and we estimate the consumption of energy for 1 Gbyte downloaded in the cloud for IPTV and OTT.

10.5.3 Cost of Energy Per Country

The objective is to consider the different costs of energy in the European countries. For these economic assessments, we evaluated two aspects of the problem:

Table 10.6 Energy savings for videos at the scope of Europe

1	2	3	4	5	6	7
Country	Population	Purchasing Power Parities	Number of video IPTV (Giga)	Savings energy convince IPTV (MWh)	Number of video OTT (Giga)	Savings energy convince OTT (MWh)
Germany	81 174 000	115	6,2	*84 889*	66,5	*621 077*
France	**66 352 469**	**100**	**4,4**	*60 338*	**47,3**	*441 456*
United Kingdom	64 767 115	103	4,4	*60 664*	47,5	*443 836*
Italy	60 795 612	89	3,6	*49 204*	38,5	*359 992*
Spain	46 439 864	87	2,7	*36 741*	28,8	*268 807*
Poland	38 005 614	63	1,6	*21 773*	17,1	*159 301*
Netherlands	16 900 726	122	1,4	*18 750*	14,7	*137 182*
Belgium	11 258 434	109	0,8	*11 159*	8,7	*81 646*
Greece	10 812 467	66	0,5	*6 489*	5,1	*47 479*
Czech Republic	10 538 275	76	0,5	*7 283*	5,7	*53 286*
Portugal	10 374 822	71	0,5	*6 698*	5,2	*49 008*
Hungary	9 849 000	63	0,4	*5 642*	4,4	*41 282*
Sweden	9 747 355	118	0,8	*10 459*	8,2	*76 524*
Austria	8 584 926	117	0,7	*9 134*	7,2	*66 827*
Denmark	5 659 715	114	0,4	*5 867*	4,6	*42 927*
Finland	5 471 753	103	0,4	*5 125*	4,0	*37 497*
Slovakia	5 421 349	72	0,3	*3 550*	2,8	*25 970*
Ireland	4 625 885	129	0,4	*5 427*	4,3	*39 702*
Total	466 779 381		30	*409 193*	321	*2 993 800*

Table 10.7 Volume and power consumption of the IPTV and OTT stored in the cloud

Consumption of 1Gbyte downloaded OTT in the cloud [Wh]	1,84
Consumption of 1Gbyte downloaded IPTV in the cloud [Wh]	1,636

- The economic savings in terms of costs of production and supply of energy (column 3 and 4 of Table 10.8)
- The total savings in terms of fee for energy (to pay as an invoice) for all actors within the chain: enterprises for content delivery and network, final consumer for terminals with different tariffs corresponding to the different market segments (columns 7 to 10 of Table 10.8).

Values used in these evaluations are extracted from European statistics [11] The price of energy excluding VAT and other recoverable taxes and levies include several taxes.

Table 10.8 Economic assessment of Energy savings for IPTV and OTT at the scope of Europe

1	2	3	4	5	6	7	8	9	10
Country	Cost of energy estimation 2020 (€/kWh)	Cost of energy economic savings IPTV (M €)	Cost of energy economic savings OTT (M€)	Price of energy households estimation 2020 (€/kWh)	Price of energy enterprises estimation 2020 (€/kWh)	Price of energy savings network IPTV (M €)	Price of energy savings terminals IPTV (M €)	Price of energy savings network OTT (M €)	Price of energy savings terminals OTT (M€)
Germany	0,0368	*3,13*	*22,88*	0,3002	0,1790	*5,06*	*17,00*	*31,15*	*134,24*
France	0,0502	*3,03*	*22,14*	0,1751	0,1072	*2,15*	*7,05*	*13,25*	*55,64*
United Kingdom	0,0774	*4,70*	*34,35*	0,2093	0,1534	*3,10*	*8,47*	*19,07*	*66,87*
Italy	0,0782	*3,85*	*28,17*	0,2554	0,1867	*3,06*	*8,38*	*18,83*	*66,18*
Spain	0,0792	*2,91*	*21,29*	0,2264	0,1235	*1,51*	*5,55*	*9,30*	*43,82*
Poland	0,0526	*1,14*	*8,37*	0,1319	0,0978	*0,71*	*1,92*	*4,36*	*15,12*
Netherlands	0,0506	*0,95*	*6,95*	0,1579	0,0966	*0,60*	*1,98*	*3,71*	*15,60*
Belgium	0,0532	*0,59*	*4,34*	0,2764	0,1390	*0,52*	*2,06*	*3,18*	*16,24*
Greece	0,0736	*0,48*	*3,49*	0,1834	0,1338	*0,29*	*0,79*	*1,78*	*6,27*
Czech Republic	0,0397	*0,29*	*2,12*	0,1408	0,0878	*0,21*	*0,68*	*1,31*	*5,40*
Portugal	0,0601	*0,40*	*2,95*	0,2242	0,1358	*0,30*	*1,00*	*1,86*	*7,91*
Hungary	0,0533	*0,30*	*2,20*	0,1063	0,0955	*0,18*	*0,40*	*1,10*	*3,16*
Sweden	0,0454	*0,47*	*3,47*	0,1884	0,0787	*0,27*	*1,31*	*1,69*	*10,38*
Austria	0,0455	*0,42*	*3,04*	0,2010	0,1205	*0,37*	*1,22*	*2,26*	*9,67*
Denmark	0,0412	*0,24*	*1,77*	0,2960	0,1123	*0,22*	*1,16*	*1,35*	*9,15*

(continued)

Table 10.8 (continued)

1	2	3	4	5	6	7	8	9	10
Country	Cost of energy estimation 2020 (€/kWh)	Cost of energy economic savings IPTV (M €)	Cost of energy economic savings OTT (M€)	Price of energy households estimation 2020 (€/kWh)	Price of energy enterprises estimation 2020 (€/kWh)	Price of energy savings network IPTV (M €)	Price of energy savings terminals IPTV (M €)	Price of energy savings network OTT (M €)	Price of energy savings terminals OTT (M€)
Finland	0,0487	*0,25*	*1,83*	0,1495	0,0833	*0,14*	*0,51*	*0,87*	*4,04*
Slovakia	0,0436	*0,15*	*1,13*	0,1537	0,1334	*0,16*	*0,36*	*0,97*	*2,87*
Ireland	0,0716	*0,39*	*2,84*	0,2472	0,1494	*0,27*	*0,89*	*1,66*	*7,07*
Total		*24*	*173*			*19*	*61*	*118*	*480*
						80		*597*	

10.5.4 Economic Savings in Terms of Costs (Purchase, Production, and Distribution)

Without access to data from producers and suppliers of energy it was difficult to define the cost of energy. At official Eurostat statistics [11] only the prices are available. These prices are collected by a different class of customers. Costs of production and distribution are not available. We suppose that wholesale prices of energy for the highest volume without taxes are near production cost. Evaluated basic costs of energy can be found in column 2 of Table 10.8. Except some specific countries, the value of a kWh ranges from 4 cents of euro to 8 cents.

10.5.5 Cumulated Savings in Terms of Fee for Energy

Based on official Eurostat statistics [11] we used the prices for households and for enterprises. To simplify comparison between price for private person and entrepreneur we used price excluding VAT and other recoverable taxes and levies include several taxes. The amount of money paid by the consumer include the prices of energy and delivery. It corresponds to what the consumer pays for its energy. When we look at the end-to-end chain we find that energy will be consumed by various actors:

- entrepreneurs like operators and content suppliers,
- domestic customers,

The players have a different tariff corresponding to the respective market segments. Values used in this evaluation refer to electricity prices for:

- industrial customers—network and headend
- domestic customers for terminals.

Corresponding values can be found respectively in columns 5 and 6 of Table 10.8. Prices of energy are more than two times the estimated costs of production.

10.5.6 Economic Assessment

Taking into account these energy costs and the forecast volume of videos per country for 2020, we obtain an order of magnitude of the yearly energy savings in production and delivery of energy in column 3 for IPTV and in column 4 for OTT video of Table 10.8. The cumulative savings in energy price, for the energy consumption in the whole end-to-end chain within the cloud, for network operators and the cumulative savings for terminal users are given respectively in columns 7 and 8

for IPTV and in columns 9 and 10 for OTT of Table 10.8. The total cumulative savings for all actors is given at the bottom side of the corresponding columns.

In both cases, the total energy savings at the scope of the main European countries is some tens of millions of euros for IPTV and in a few hundred million euros for OTT, which is little compared to the energy consumption of network operators in Europe. What eventually matters is the relative saving for these video services, globally about 10%, which is significant, even if it is not a true break with the current energy consumption.

The assumptions taken for this analysis (IPTV and OTT in the global traffic, consumption of energy of the elements of the chain) have been chosen to give upper bounds to the energy savings.

In our methodology, results and measurements obtained for one operator have been generalized to other operators in other countries. The objective of these calculations is to give an order of scale about possible savings and not a precise evaluation. We developed our methodology after comparing the evaluation of pricing costs of data gigabyte in various European affiliate of the Orange group. The range of value is in a ratio less than 5, mainly depending on the usage of traffic. We made the assumption that the trend should be similar for the consumption of energy per gigabyte as for the cost.

This methodology may thus induce errors of evaluation in a ratio until five times more. Suppose thus the energy savings is five times higher, the final trend of the evaluation remains unchanged: the economy of energy related to the service IPTV and OTT is small compared to the total energy consumed by networks and terminals.

10.6 Conclusion

Savings that can be obtained by the CONVINcE project for IPTV and OTT service remain small at an European level. We took into account cloud (NGPoP solution) where we defined and included cost of energy of the cloud. We can only obtain savings in the order of magnitude of some hundreds of millions of Euros per year in Europe. With regard to the energy expenses of all participants (video service sellers and users) we obtained 80 M€ for IPTV and 597 M€ for OTT (Table 10.8) savings. However, at the level of the service, this represents about 13% for IPTV and 14% for OTT savings for the delivery of the service, which is not negligible for the stakeholders.

The energy consumption for IPTV and OTT service is a small portion of the total energy that operators use in their networks. But nowadays is very hard to reach spectacular savings. Operators are looking for even small cost-cutting opportunities.

Acknowledgements This research was partially supported by the European Celtic-Plus project CONVINcE and funded by Finland, France, Sweden and Turkey.

References

1. Convince Project website. https://www.celticplus.eu/project-convince/
2. Monnier R, Popescu A, Ljung R (2016) CONVINcE: towards power-optimized video distribution networks. In: Proceedings of the 19th international ICIN conference, March 2016, Paris
3. Pilarska K, Liau B, Perrot N, Bardon J, Ljung R, Bakanoglu K, Forsell M (2017) CONVINcE D1.2.3 Business Case IPTV and OTT with NGPoP solution. CONVINcE project public document, Aug 2017
4. Carlinet Y, Le Rouzic E (2016) CONVINcE D3.1.2 Specification of energy saving mechanisms in the core/metro network. CONVINcE project public document, Nov 2016
5. Forsell M, Nissilä T, Tiensyrjä K, Nilsson Plymoth A, Zhang Z, Erol OI (2017) CONVINcE D4.1.2 Description of energy-aware-video processing architecture on terminals. CONVINcE project public document, April 2017
6. Nilsson Plymoth A, Zhang Z (2016) Energy savings for video streaming using fountain coding. In: Proceedings of the 19th international ICIN conference, March 2016, Paris
7. Bastani S, Carlinet Y, Naghmouchi Y, Perrot N, Heikkinen A, Farahbakhsh R, Mohammadi S, Crespi N, Komu M (2017) CONVINcE D3.3.2 Design of energy efficient CDN including data centers and cloud. CONVINcE project public document, July 2017
8. Ickin S, Fiedler M (2016) Quality of experience on smartphones: network, application, and energy perspectives In: Proceedings 19th international ICIN conference, March 2016, Paris
9. Zhang Z, Karaca M, Moradi F, Bastani S, Nilsson Plymoth A, Ljung R, Landfeldt B (2016) Cross-layer energy optimization for dynamic video streaming over Wi-Fi. In: Proceedings of the 19th international ICIN conference, March 2016, Paris
10. http://www.oecd.org/std/prices-ppp/purchasingpowerparitiespppsdata.htm
11. Eurostat. http://ec.europa.eu/eurostat/web/energy/data/database

Index

A. Popescu (ed.), *Greening Video Distribution Networks*,
Computer Communications and Networks,
https://doi.org/10.1007/978-3-319-71718-0