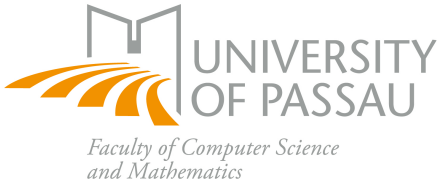


Power-Adaptive Computing in Future Energy Networks

Florian Niedermeier

A dissertation submitted to the faculty of computer science and mathematics in partial fulfillment of the requirements for the degree of doctor of natural sciences

Advisor: Prof. Dr.-Ing. Hermann de Meer

August 2021

1. Reviewer: Prof. Dr.-Ing. Hermann de Meer
   Computer Networks and Communications Group
   University of Passau
   Innstraße 43
   94032 Passau, Germany
   Email: hermann.demeer@uni-passau.de
   Web: http://www.net.fim.uni-passau.de

2. Reviewer: Prof. Dr. Michael Sonnenschein
   Environmental Informatics Group
   University of Oldenburg
   26111 Oldenburg, Germany
   Email: michael.sonnenschein@uni-oldenburg.de
   Web: http://www.uni-oldenburg.de/ui/

Abstract

The current electricity grid is undergoing major changes. There is increasing pressure to move away from power generation from fossil fuels, both due to ecological concerns and fear of dependencies on scarce natural resources. Increasing the share of decentralized generation from renewable sources is a widely accepted way to a more sustainable power infrastructure. However, this comes at the price of new challenges: generation from solar or wind power is not controllable and only forecastable with limited accuracy. To compensate for the increasing volatility in power generation, exerting control on the demand side is a promising approach. By providing flexibility on demand side, imbalances between power generation and demand may be mitigated.

This work is concerned with developing methods to provide grid support on demand side while limiting the associated costs. This is done in four major steps: first, the target power curve to follow is derived taking both goals of a grid authority and costs of the respective load into account. In the following, the special case of data centers as an instance of significant loads inside a power grid are focused on more closely. Data center services are adapted in a way such as to achieve the previously derived power curve. By means of hardware power demand models, the required adaptation of hardware utilization can be derived. The possibilities of adapting software services are investigated for the special use case of live video encoding. A method to minimize quality of experience loss while reducing power demand is presented. Finally, the possibility of applying probabilistic model checking to a continuous demand-response scenario is demonstrated.

Acknowledgements

First, I would like to express my sincere gratitude to my supervisor, Prof. Hermann de Meer, for his continuous support and guidance throughout my time with his research group. He entrusted me with interesting and equally challenging tasks, many of them inspired parts of this thesis.

I also want to cordially thank Prof. Michael Sonnenschein for kindly agreeing to serve as second reviewer for my thesis.

My special thanks go to all of my colleagues at the University of Passau for being both the best coworkers I could wish for and true friends, even outside the office. Specifically, I want to thank Andreas Fischer and Gergő Lovász for their support, especially during the first years of my academic career, the insightful discussions, and reviews of my thesis.

To my parents, family, and friends for their support, love, and their unquestioning believe in me—thank you for everything.

Finally, I want to thank my beloved girlfriend Carola for her support during the last years, her patience with me and continuous encouragement to not stray from this path.

Contents

List of Figures

LIST OF FIGURES

LIST OF FIGURES

List of Tables

LIST OF TABLES

List of Acronyms

AC Alternating Current 4
ACPI Advanced Configuration and Power Interface 116
AMB Advanced Memory Buffer 74
ASHRAE American Society of Heating, Refrigerating and Air-Conditioning Engineers . . 93
ASIC Application-Specific Integrated Circuit 107
AS Ancillary Service 12
BIOS Basic Input Output System 111
CCX CPU Complex 70
CMDB Configuration Management Database 68
CMOS Complementary Metal-Oxide-Semiconductor 53
CPU Central Processing Unit 16
DC Data Center 4
DCT Discrete Cosine Transform 22
DSO Distribution System Operator 11
DVFS Dynamic Voltage and Frequency Scaling 61
DVS Dynamic Voltage Scaling 60
D/R Demand/Response 4
ELB Energy Layer Boundary 54
FET Field-Effect Transistor 55
GA Grid Authority 5
GHG Greenhouse Gas 29
GPU Graphics Processing Unit 21
HAS HTTP Adaptive Streaming 21
HDD Hard Disk Drive 17

LIST OF ACRONYMS

HPC	High-Performance Computing 13
HVAC	Heating, Ventilation & Air Conditioning 14
IC	Integrated Circuit 16
ICT	Information and Communications Technology 2
IQR	Interquantile Range 88
JEDEC	Joint Electron Device Engineering Council 74
MC	Markov Chain 153
MDP	Markov Decision Process 25
MOS	Mean Opinion Score 132
MSE	Mean Squared Error 19
MTBM	Mean Time Between Maintenance 96
NIC	Network Interface Card 17
OAS	Optimal Adaptation Sequence 104
OS	Operating System 61
PCB	Printed Circuit Board 80
PF	Power Factor 46
PMC	Probabilistic Model Checking 6
P2P	Peer-to-peer 21
PP	Power Planner 32
PSU	Power Supply Unit 81
PSNR	Peak Signal-to-Noise Ratio 132
PUE	Power Usage Effectiveness 18
QoE	Quality of Experience 6
QoS	Quality of Service 116
RAM	Random Access Memory 17
RC	Resistor-Capacitor 61
RPP	Residual Power Plan 148
SDN	Software-defined Networking 21
SG	Synchronous Generator 3
SIL	Safety Integrity Level 148
SLA	Service Level Agreement 14

SoC System on Chip 111

SSD Solid-State Drive 17

SSIM Structural Similarity Index 128

TCO Total Cost of Ownership 12

TDP Thermal Design Power 63

TSO Transmission System Operator 6

UEFI Universal Extensible Firmware Interface 111

UML Unified Modelling Language 64

UPS Uninterruptible Power Supply 14

VM Virtual Machine 13

VoIP Voice over IP 147

VPP Virtual Power Plant 12

VRM Voltage Regulator Module 87

LIST OF ACRONYMS

The beginning is the most important part of the work.

Plato

1 Introduction

In modern society, electric power has become an indispensable good. Over the last decades, mankind's reliance on a highly available power supply has grown from gaining mere comfort to a substantial dependency touching every aspect of life. In the early 20th century—in the advent of publicly available electric power—services enabled by electricity were expendable. Today however, a disruption in the electric power system may cause enormous economic or even personal damage. The adverse consequences of power system failures were visible in independent incidents in the U.S. and Sweden in 2003, causing an economic damage of more than 6.8 billion dollars and 145 million euros, respectively. Based on these past incidents, a report to the German Federal Parliament was released, investigating the consequences of a long term, wide area failure of the public power grid [7]. The economic damage is estimated at between 0.6 and 1.3 billion euros per hour. Additionally, several critical infrastructures are identified as directly dependent on the availability of the public power grid, e.g., transport, water supply, health and financial sector. Due to the grave consequences predicted, the report identifies a long term, wide-area power outage as possible cause for a *national catastrophe*.

1.1 Future Energy Systems

While the dependency on electric power systems is ever increasing, the efforts to strengthen the reliability and resilience of these systems are lacking behind by far [136]. This is even worsened by the fact that the electrical power system is undergoing the arguably largest change since its advent more than 100 years ago: the evolution into a smart grid. While the exact definition of the term *smart grid* differs between stakeholders, several key aspects remain in common. Among others, the following statements are generally agreed upon, as outlined by Rohjans et al. [157].

- The smart grid will be enabled by augmenting the current electricity network with interconnected information and communications technology (ICT) devices.
- Distributed generation from renewable sources will contribute with an increasing share to the total energy mix.
- To cope with volatile generation from renewable sources, energy management systems or electrical energy storage devices will be a necessary part of a smart grid.

The integration of distributed generation from renewable sources is also a key point in worldwide efforts to slow climate change. However, it also leads to a paradigm shift in power grid operation.

In the past, the power grid had a clear hierarchical structure and unidirectional power flow. Large bulk generation power plants generated electrical energy by burning fossil fuels and fed electricity to the high voltage grid from which it traveled to consumer side. Figure 1.1 shows a simplified conceptual overview of a traditional power grid.

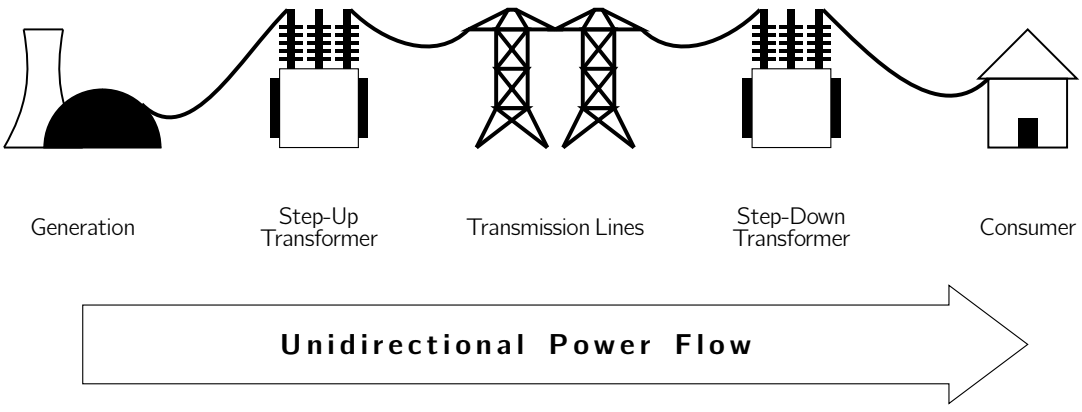


Figure 1.1: Traditional power system overview.

In future smart grids, this comparatively simple power transmission concept is invalidated. Electricity generation is increasingly decentralized as more renewable energy sources are connected to the distribution system of the power grid. This changes the power flow throughout the grid and in some cases may even lead to a reversed power flow from consumer side towards higher voltage grid levels.

Among renewable energy sources, wind and solar power generation play a special role for two reasons. On the one hand, they are responsible for more than half of the total electricity generation from renewable sources in Germany[1]. On the other hand, these energy sources are specifically volatile, i.e., controllability is often limited to capping feed-in to the grid [50]. This adds a new challenge to the provisioning of a stable and sustainable power grid: the previously controllable and hierarchical architecture of power generation is faced with an increasing uncertainty and non-controllability created by a rising share of solar and wind power sources. What once was a single-sided volatility in electrical power demand, compensated for by a continuously adapted generation to meet the needs of consumers, is now a two-sided problem, with volatility in both power generation and demand. Figure 1.2 shows a conceptual view of a possible future smart grid.

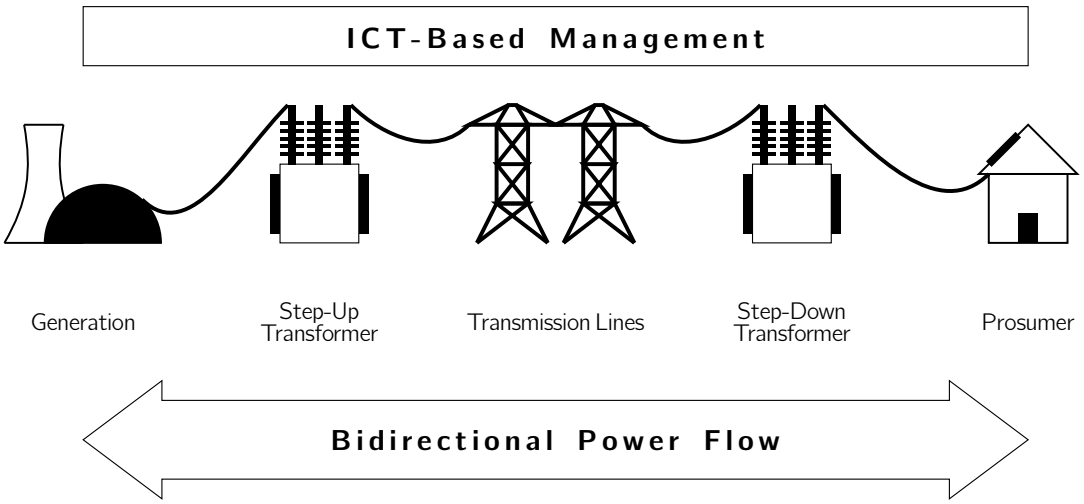


Figure 1.2: Possible smart grid power system overview.

Apart from the limited controllability, renewable sources are often interfaced with the power grid using inverters. This creates further challenges, as inverters—in contrast to synchronous generators (SGs)—can provide only a very limited amount of *inertia* [45, 168]. This inertia is an essential system parameter required for the synchronized operation of the current power grid. The kinetic energy stored in the rotating masses of synchronous generators attenuates deviations from nominal

[1] `https://www.umweltbundesamt.de/themen/klima-energie/erneuerbare-energien/erneuerbare-energien-in-zahlen`, accessed 20.03.2019.

grid frequency by either injecting (i.e., slowing down) or absorbing (i.e., speeding up) energy into or from the power grid [168]. With a continuously dropping amount of available inertia, grid stability margins decrease while potential challenges to power quality increase. The reduced margins for error further increase the need for a way to accurately balance power generation and demand.

In summary, operating a future smart grid will encompass several new challenges, which will not be solvable by traditional means alone. The assumptions required for the successful enactment of current grid control strategies, like controllable generation in large power plants by means of SGs, lose their validity in a future smart grid. Therefore, new options towards grid control need to be evaluated. One approach to regain flexibility lost to the intermittent nature of renewable sources is to employ the ICT components present in a smart grid to create a control channel between generation and demand side. As it is now also the generation side that is facing volatility problems, the demand side may offer to adjust its power draw in a way such as to meet the needs of the generation side. This concept is called demand/response (D/R). Although rudimentary control channels are already present in a traditional power grid (e.g., ripple control), this implementation has some major drawbacks.

- As ripple control signals are superimposed on the standard 50 Hz grid alternating current (AC) line, it has negative effects on power quality [25].
- Ripple control offers low transmission speeds, thus limiting control on power use to simple switching actions instead of fine-grained control [183].
- Ripple control is flooded through a large area of the power grid, introducing additional challenges in privacy and addressing of individual loads.

To preserve grid stability in spite of the massive changes the power grid is currently facing, new paradigms for demand-side management are necessary. By employing the ICT augmentations to the power grid, it is possible to communicate power demand adaptation requests from generation to demand side in a more elaborate way. This is an enabler for a more sophisticated cooperation between generation and demand side towards the common goal of a more reliable and sustainable electricity grid operation. Still, D/R comes at a price: as outlined at the very beginning of this chapter, there is a significant dependency on a permanent availability of almost any requested amount of electricity. Imposing limitations on this currently mostly unconstrained power draw may cause undesirable effects for a load. Although a number of D/R programs exist in Europe [169] and the U.S. [35], they are often centered around dynamic pricing to indirectly influence power demand. Also, the reaction delay of loads may be significant due to manual processing of power adaptation requests by human operators.

Data centers (DCs) have been identified as potentially viable candidates as participants in D/R programs [181]. They exhibit a significant power demand and have highly automated infrastructures in place. However, there is ongoing debate regarding the concrete power adaptation point inside a DC. The question of how to integrate a DC in a D/R program while simultaneously keeping the impact on its service quality to a minimum remains. Additionally, even if a solution is proposed which claims to provide both power demand flexibility and an acceptable service quality, a DC operator may be hesitant to adopt it without guarantees regarding its success. A further question is therefore currently unanswered: are formal methods available which can verify certain properties of a DC in a D/R scenario?

1.2 Solution Approach

This work proposes to tackle the challenges currently faced by the electricity grid by creating a system that provides the possibility to adapt the power demand of DCs according to guidelines provided by a grid authority (GA). These guidelines need to respect both grid requirements and demand-side limitations in power adaptation. In contrast to most existing D/R systems, it is envisioned that grid load is controlled by influencing DC power demand indirectly by adapting utilization caused by software execution. This provides several advantages over current approaches: adapting power demand in software is an autonomous *solid-state* operation, thus no mechanical components are involved. This allows for quick reaction times and wear-free switching. Additionally, no human action is required to perform switching, which eliminates possible human errors in operation and minimizes delays in decision making. As the approach is software-based, updates can be easily deployed throughout the system lifecycle. Finally, the amount of power demand adaptation can often be adjusted in several steps. Adaptation is therefore more fine-granular and not limited to on/off switching. However, adapting ICT power demand via changes in software execution is generally not possible without certain side effects, i.e., impacts on software service quality. This negative influence on experienced quality needs to be minimized for each available setpoint in power demand. Also, as the proposed system is intended to increase power grid stability, it has to be designed for operation within critical infrastructures. The main research hypotheses derived and investigated throughout this thesis are listed in the following.

1. Inside a DC, adapting the power demand of ICT devices is the most favorable point of action, as further DC power flexibilities are gained by a reduced ICT power demand.
2. A significant advantage in the tradeoff between power reduction and loss of experienced service quality can be achieved by adapting resource demand from within the application via a configuration change, as opposed to imposing resource limits externally by limiting resource availability.

3. The scalability dimensions offered by a live video-encoding service are well suited to provide a graceful quality of experience (QoE) degradation when faced with a reduced power budget.
4. The method of probabilistic model checking (PMC) is an appropriate way to verify certain properties of the proposed system when deployed in a critical infrastructure such as the public electricity grid.

The specific use case focused on in the course of this work is that of a DC specializing in live video encoding. To be feasible in achieving the stated goal of integrating these DCs as flexible loads in a future smart gird, several assumptions need to be taken. General assumptions required for the viability of the proposed system are discussed in the following.

First, it is assumed that the amount of power demand flexibility gained by influencing a DC's load is of relevant magnitude, i.e., has a measurable effect on the (local) electricity grid. The available flexibility in a single DC is predominantly influenced by two factors: the total power demand and the fraction of this power demand which is dynamic, i.e., depends on the utilization of hardware components. To participate in the German market for instantly and quickly curtailable loads, a minimum power flexibility of 5 MW is required[2], which can be seen as a lower bound for relevance to transmission system operators (TSOs). For distribution system operators, this bound is expected to be significantly lower. Regarding a DC's ability to provide this flexibility, there are examples of DCs using dedicated substations with power demands of up to 1000 MW[3]. Even with only very limited flexibilities these exceptionally large DCs could provide considerable services to TSOs. For more average-sized DCs, total power demand is estimated in a range from 105 kW to 5700 kW [68]. For DCs of this size, power adaptations may either be offered to a TSO by means of an aggregator (marketing the cumulative flexibilities of multiple DCs), or for either voltage or power generation optimization to a lower voltage grid operator.

Second, an ICT-based communication is required for flexible loads to dynamically react to remote commands. As the use case discussed in the following focuses on DCs as flexible loads, it is safe to assume that an IP-based communication infrastructure is available. On the side of the grid authority, communications with all flexible loads are managed. It is assumed that no bottlenecks in communication need to be accounted for, as most arising scalability problems may be tackled by, e.g., cloud deployments and horizontal scaling.

Finally, the specific class of DCs specializing in live video encoding is assumed to present a relevant use case. According to a white paper published by Cisco [76] the total amount of video traffic is

[2] `https://www.regelleistung.net/ext/static/abla`, accessed 10.08.2018.
[3] `https://data-economy.com/norway-lands-worlds-largest-data-centre-1000-mw-6-5-million-sqf/`, accessed 12.08.2018.

estimated to "[...] continue to be in the range of 80 to 90 percent of total IP traffic. Globally, IP video traffic will account for 82 percent of traffic by 2021". Although not all video traffic is created by live encoding, the rise of online services like Twitch[4] shows a clear trend of live video broadcasting popularity. According to statistics gathered by TwitchTracker[5], the number of unique monthly broadcasters on Twitch increased from 300 000 in 2012 to 3 149 211 in 2018. Additionally, IP-based high-resolution live broadcasting of sports events (e.g., the 8K resolution broadcast of the 2018 Winter Olympic Games in South Korea[6]) require high computational power as provided by DCs. Several further examples regarding the increasing importance of live video processing could be given.

There are further, more specific assumptions required for certain methodological approaches used in the upcoming chapters. To enhance readability, these assumptions are given and their validity is discussed in the respective chapter.

1.3 Contributions

This work suggests the use of DCs as flexible participants in the electricity grid. The methods proposed build on existing research and extend or adapt it to realize this approach. The subsequently listed contributions are part of this work.

1. **A formulation of an optimization problem to derive a load-tailored plan on future power demand based on the availability of renewable energy.** The two major concerns addressed while deriving this plan on future energy draw are hiding specific information regarding process flexibilities and not overburdening the load by demanding excessive amounts of power demand adaptation.
2. **A component-based power demand model applicable from individual hardware components to entire DCs.** Based on an analysis of power demand on circuit level, the demand of different hardware components is modeled with the capability for reuse of component models in other hardware configurations in mind.
3. **An investigation regarding the best point of power adaptation inside a data center.** By analyzing the impact of limiting resource availability/utilization on different levels of a server, conclusions regarding a favorable point of power adaptation can be drawn. This point is selected by minimizing the number of unintentionally affected server levels.

[4] `https://www.twitch.tv/`, accessed 12.08.2018.
[5] `https://twitchtracker.com/`, accessed 12.08.2018.
[6] `https://www.ibc.org/delivery/winter-olympics-innovates-with-8k-hdr-and-live-5g-production-firsts/2648.article`, accessed 12.08.2018.

4. **A method to derive an optimal way of adapting a software service in the joint power/QoE plane.** The proposed algorithm ensures a minimum degradation in service quality while traversing a sequence of configurations with decreasing power demands.
5. **An optimal service adaptation for the specific use case of live video encoding.** The proposed approach of software configuration adaptations for power flexibility is applied to a live video encoding use case. The results are compared to an external resource limitation with regard to the power/QoE tradeoff.
6. **A conceptual formal safety analysis based on probabilistic model checking.** Using a model based on Markov decision processes, the possibility of applying probabilistic model checking to a continuous demand response scenario is demonstrated.

1.4 Structure

This thesis is structured following a top-down approach and subdivided into three interrelated areas: guidance, control and verification. Chapter 2 introduces related work that has connections to this thesis. In Chapter 3 a guideline on how much power to consume in the future is derived, considering both requirements from generation and demand side. Chapter 4 discusses how ICT power demand can be estimated based on analytical models and readily available monitoring data. In the following, Chapter 5 discusses possibilities to influence the resource demand of software. Chapter 6 demonstrates the applicability of PMC to the verification of DC operation in continuous D/R programs. Chapter 7 concludes the thesis and gives an outlook on possible future work. The structure is depicted in Figure 1.3.

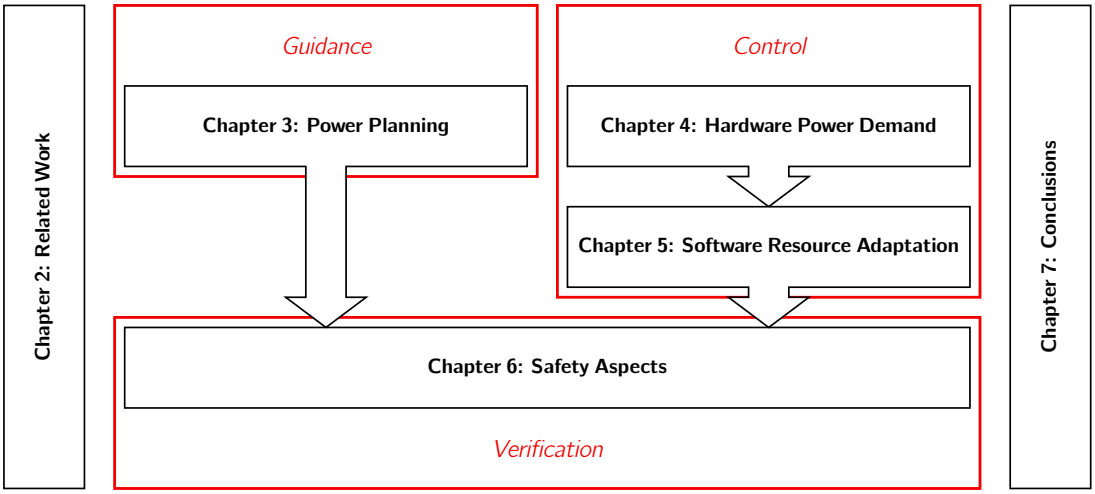


Figure 1.3: Thesis structure.

If I have seen further it is by standing
on the shoulders of giants.

Isaac Newton

2

Background and Related Work

This chapter contains material previously published in [126], [127], and [15].

This thesis has connections to four major research areas: demand response, power demand modeling, video encoding, and model checking. While related work in these areas can mostly be clearly separated, some publications fit equally well into several subfields. In this case, the work is included where context seems most fitting while avoiding duplicate mentioning. The following chapter introduces relevant background information for the major research fields and gives an overview on existing research directions. The idea of *demand response* is to provide flexibility in the power draw of electricity consumers. There are several approaches to D/R, some realized in practice and far more proposed in scientific literature. *Power demand modeling* is concerned with building models to estimate the electrical power demand of a load. In the course of this work, models are built for ICT components of a DC. To simulate the resulting power demand of different workload assignments to computing devices, power demand models are indispensable. The field of *video encoding* researches possibilities of finding an efficient digital representation of video data. In the context of this work, video encoding is used as an example of a flexible software service to react to

bounds regarding power demand. *Model checking* is a formal method to verify certain properties of a model. To realize the system proposed in this thesis, it would have to be implemented in a safety critical infrastructure—the power grid. Therefore, a formal proof of its properties is highly recommended according to IEC 61508 [75]. Figure 2.1 shows the interaction of the different research fields within this work, including relevant related work in the respective field. Contributions by the author, i.e., (co-)authored publications and connections of research fields, are marked in red color.

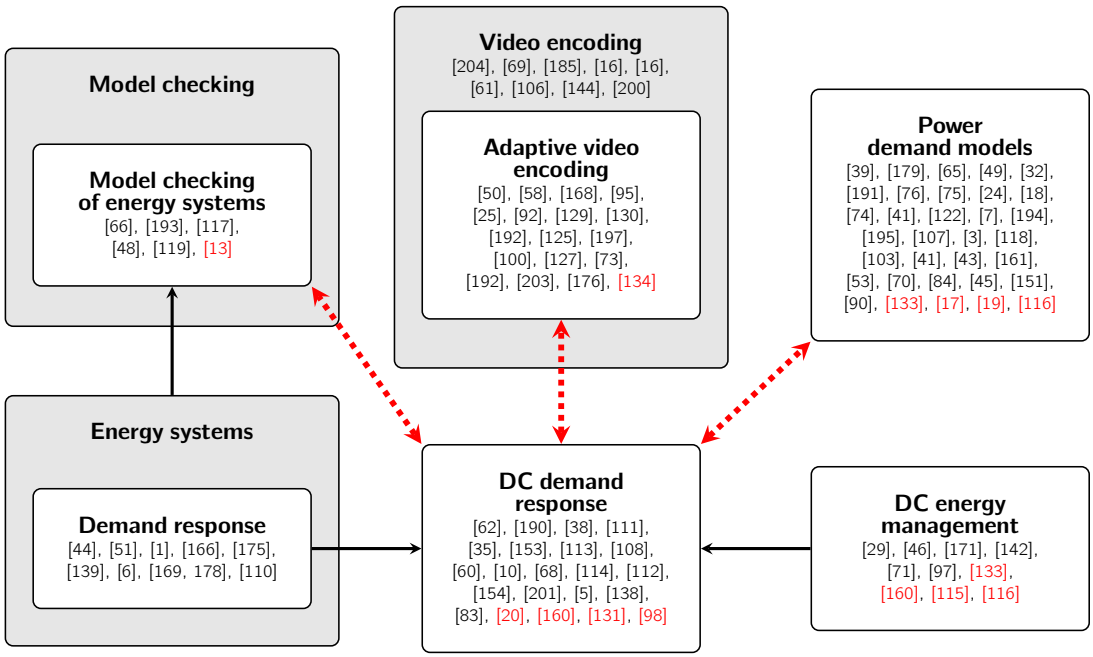


Figure 2.1: Research field interaction in this thesis.

In the following, background information and previously conducted work in the context of these research fields is summarized and categorized in relation to the presented thesis.

2.1 Demand Response

The basic concept of D/R is to increase flexibility of electrical power demand by incentivizing adaptive behavior of customers. The Federal Energy Regulatory Commission[7] defines demand response as

[7] https://www.ferc.gov/industries/electric/indus-act/demand-response/dem-res-adv-metering.asp, accessed 18.06.2018

> "[c]hanges in electric usage by demand-side resources from their normal consumption patterns in response to changes in the price of electricity over time, or to incentive payments designed to induce lower electricity use at times of high wholesale market prices or when system reliability is jeopardized."

By successfully employing a D/R program, distribution system operators (DSOs) may save significant amounts of investments in grid reinforcement, as peak demands threatening to overburden the power grid infrastructure may be avoided. However, consumers need to be provided with incentives to participate in D/R programs. Deng et al. [42] categorize D/R programs as listed in the following.

1. **Incentive-based programs:** In this type of program, participants receive incentive payments independent from electricity pricing. Incentive-based programs can be further subdivided into four categories.
   (a) **Direct load control:** In this most invasive version of D/R, an energy utility receives direct control over a device's operation. If required, a power utility may switch off the load temporarily. Participating consumers are offered incentive payments, as outlined by Ericson [49].
   (b) **Curtailable load:** This program works similar to direct load control, however final control over a device is enacted by a user. Again, certain incentive payments for participating consumers are offered in return [1].
   (c) **Demand bidding:** Similar to an auction, demand bidding allows users to only reduce their demand if a certain incentive payment amount is offered. Failing to reduce load after a payment has been accepted will result in a penalty for the respective user [161].
   (d) **Emergency demand reduction:** An emergency D/R program is designed to incentivize users to react to sudden requests required in the case of, e.g., generator failure. Users participating in the program are again offered compensation payments [170].
2. **Price-based programs:** In contrast to incentive-based programs, participants are offered a reduction in electricity pricing. It is therefore a more indirect form of D/R.
   (a) **Time-of-use pricing:** Time-of-use pricing assigns different electricity prices based on seasons, days of the week or even time of day. The effectiveness of the approach relies on the assumption that power system challenges arise at similar points in time repeatedly.
   (b) **Critical peak pricing:** Critical Peak Pricing is a more pronounced form of time-of-use pricing, assigning very high costs to times of high stress in the power infrastructure.

(c) **Real-time pricing:** Real-time pricing is aligned with electricity market pricing, therefore varying continuously.

In an academic context, research in D/R is diverse and has been directed into multiple subfields. Several surveys on existing approaches do exist, outlining the wider context of demand side management [134], general deregulated electricity markets [5], or with special focus on D/R in smart grids [164, 173]. However, the information exchange to enable D/R cooperation also introduces new challenges: the privacy threats of participation in D/R programs are covered by, e.g., Lisovich et al. [104]. They argue that mining of data gathered in the course of a D/R program can reveal sensitive information. This aspect is briefly covered in Chapter 3 of this work.

Due to the vertical unbundling in the European electricity sector [141], price-based D/R programs are comparatively hard to implement (as D/R services are in fact provided to the DSO or TSO, however, electricity accounting is part of the sales department). Despite the legal challenges, some practical realizations of D/R programs do exist. Torriti et al. [169] give an overview of existing D/R programs in Europe. Additionally, they provide in-depth case studies of the UK, Italy and Spain and analyze factors that have promoted or impeded advances in D/R. A similar overview of D/R in the U.S. is presented by Cappers et al. [35]. Regarding a technical realization, Open Automated Demand Response (OpenADR)[8] is an open and standardized way for electricity providers and system operators to communicate D/R signals with each other and with their customers over existing IP-based communications networks. Virtual power plants (VPPs), in contrast, aim to group several distributed generators in a way such as to create a reliable overall power supply. Prototype implementations exist in several locations. Research in VPP management is presented by, e.g., Nieße [130].

Other research focuses on the broader context of providing ancillary services (ASs) to the grid by, e.g., using electrical power storage devices to re-introduce inertia into the power grid. As outlined in Chapter 1, an increasing share of inverter-coupled renewable energy in the public power grid leads to a loss of inertia, previously provided by a large number of SGs. Although possible, emulating the behavior of SGs by inverter-coupled electrical storage devices is challenging, as outlined by multiple publications (e.g., [52, 142]). In this work, similar ideas are only briefly discussed in Chapter 5.

The special use case of DCs as participants in D/R programs creates an overlap with a previously mostly independent research field: DC energy efficiency. DC energy efficiency is concerned with improving the performance/energy demand ratio of DCs, primarily to reduce total cost of ownership (TCO) (i.e., the cost for keeping the DC operational after initial purchase of components). With the overall increase in power grid volatility, combined with a trend of ever-increasing power demand

[8] `http://www.openadr.org/`, accessed 07.10.2018.

of DCs, the favorable properties of DCs for the use in D/R programs (e.g., large power draw, flexibility, and automation [181]) become apparent. Before focusing on DC D/R, a limited overview on DC energy efficiency research is given.

There is an extensive body of research in the area of DC energy efficiency. There are approaches both on hard- and software-side: Bramfitt et al. [28] give a high-level overview on possibilities to increase DC energy efficiency. Several different points of action are focused on, among others cooling system efficiency, variable frequency operation, or central AC/DC conversion. In addition to hardware-based approaches, technologies like virtualization and consolidation are considered. Work by Diouri et al. [44] introduces the SESAMES architecture, which focuses on energy-efficiency in the specific scenario of high-performance computing (HPC). A purely infrastructure-based approach to raise energy efficiency is hot and cold aisle containment, increasing air conditioning efficiency [166] or overall operating temperature modification [138]. Other publications propose a network-centric approach to realize power demand savings [67, 92]. Publications contributed to by the author ranged from overall efforts to quantify ICT energy efficiency [128], over DC infrastructure management [155] to optimizing placement of virtual machines (VMs) in virtualized DC environments [109, 110]. In summary, approaches towards increased DC energy efficiency are numerous and some date back over a decade.

This overview is far from exhaustive. However, research in DC energy efficiency is only partly related to this work. It also tries to influence DC power demand, does however generally not address flexibility. Instead, the goal is a permanent maximum saving in power demand. Although this is desirable from a TCO perspective, offering power demand flexibility instead may create additional benefits regarding power grid stability, while also making the DC eligible for incentive payments.

The special use case of DCs in D/R programs combines the two previously covered research directions: influencing DC power demand in a way such as to provide the necessary flexibility to enhance power grid operation. Feasible power management techniques are actively investigated, an overview may be gained using the survey by Wierman et al. [181]. Three research directions are specifically related to the approach proposed in this work.

1. Temporal load shifting: Data centers may process both jobs that are delay-tolerant and -intolerant. These flexibilities may be exploited by rescheduling jobs which are insensitive to limited time delays. Several approaches in this field exist in literature, optimizing towards different goals (e.g., [36, 59, 105, 184]).
2. Spatial load shifting: In case a data center is federated with other data centers capable of taking over computing tasks, a redistribution of load between locations may be feasible.

Although there may be technical and legal constraints in reassigning processing locations, multiple approaches have been proposed [34, 101, 106, 149].

3. Quality degradation: Finally—and most closely related to the solution approach of this work—there is a possibility of power adaptation in DCs via quality adaptation of services. Some previous work exists in this area [9, 57, 64], however, none of these approaches is focused on the multi-dimensional scalability exhibited by modern video encoders.

Further research closely related to the approach presented henceforth is conducted by, e.g., Liu et al. [107], who investigate the overall feasibility of DCs replacing electricity storage systems as providers of power flexibility and analyze the effectiveness of prediction-based pricing as D/R mechanism for DCs. In [105], Liu et al. present an optimization based on linear programming for virtualized DCs to maximize the intake of renewables. Similarly, Rao et al. [149] define a mixed-integer programming problem, however with the goal of minimizing energy cost. Zhu et al. [195] propose a rolling-horizon scheduling architecture, closely related to the work performed in the following Chapter 3 of this thesis. However, their approach considers an integrated scheduling and power planning, whereas the approach in this work argues for a separation of these steps.

The special case for continuous D/R is presented by O'Connell et al. [133]. They argue that in a power grid with a high penetration of renewable sources, a continuous D/R approach is required to handle intermittent generation. Irwin et al. [78] focus on the more specific case of continuous D/R in DCs, however their work deals with the availability of data replicas in face of server power cycling due to power adaptation.

The EU FP7 research project All4Green[9] also focused on D/R with DCs acting as power grid demand side. In the course of the project, a protocol to communicate power adaptation requests to participating DCs was developed. Additionally, two contractual frameworks were co-developed by the author: GreenSDAs [18] and GreenSLAs [155]. While the former formalizes an agreement between utility and DC on power flexibilities, the latter captures an agreement between a DC and its clients on flexibilities in their service level agreements (SLAs). The targeted ICT systems did however not employ software reconfiguration but rather exploited uninterruptible power supply (UPS), heating, ventilation & air conditioning (HVAC) and VM consolidation flexibilities. This approach was extended to include the cooperation between a DC and a smart city authority in the EU FP7 project DC4Cities. This specific use case is considered in two of the author's publications [93, 126], which contributed to the research idea of this thesis.

[9] `http://www.all4green-project.eu/`, accessed 16.05.2016.

The author is not aware of any other rolling horizon-based approach that separates power target calculation and schedule implementation while encoding flexibility preferences in cost function parameters, apart from his own publication [127].

2.2 Power Demand Modeling

Power demand modeling is concerned with deriving estimation methods for the electrical power demand of a load. The specific sub-field relevant to this work is power demand modeling of ICT devices and DCs. Creating such a power demand model means to derive an algorithmic way of estimating the power demand without performing actual measurements. A sufficiently accurate power demand model provides two major advantages in comparison to measurement-based approaches.

1. Collection of input parameters: parameters required by the power model may be easier and/or cheaper to obtain as compared to direct power measurement of all relevant devices. Even more, some system components may be hardly measurable during runtime.
2. Power demand simulation: the input parameters to a power demand model may be chosen freely to simulate hypothetical scenarios. In comparison to measurements, this allows for a power demand simulation of future system situations (e.g., component utilization by workload) without actually stressing the system.

Power demand modeling may be performed at different levels, depending on the required accuracy and available system information. System information in this case refers to both constant (i.e., time invariant) and dynamic (i.e., time variant) system parameters. For many levels of modeling granularity there are power estimation approaches available, as shown in Figure 2.2.

Some representative models for each level are presented in the following. It is noteworthy that additional models exist for auxiliary infrastructure in a DC, especially cooling equipment. These components are however out of scope of this work. For several levels, especially on component and full-system level, the survey of Dayarathna et al. [41] gives a wide overview on existing approaches. Those more closely related to this work are discussed in the following subsections.

2.2.1 Transistor Level

On the smallest scale relevant in the following work, there are power demand models that estimate the power dissipation in transistors (e.g., [37, 61, 174]). Although modeling on transistor level

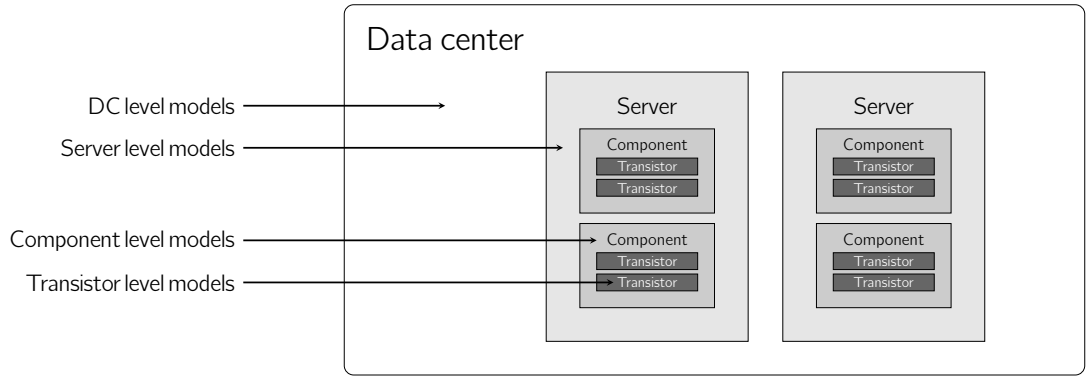


Figure 2.2: Abstraction levels of power demand models.

can reach significant accuracy, applying these models on the ICT devices of an entire DC may be infeasible: the sheer number of transistors in a single CPU may reach up to several billion (cf. Table 4.1). Thus, efficient ICT device modeling needs to be performed on a higher level of abstraction. Transistor level models are not investigated further at this point.

2.2.2 Component Level

There are several models available that focus on individual ICT components. Many component models are concerned with central processing unit (CPU) power demand, as it is the primary cause of dynamic power demand in servers [47]. Multiple CPU power models are available for single-core [31, 72, 185] and multi-core [23, 71] processors in literature. More recent work reflects the hardware architecture of modern multi-core processors more accurately by separating a CPU into exclusive and shared parts, yielding more accurate results. Several approaches towards estimating newer shared-component CPU power demand have been proposed in literature. Among others, Basmadjian et al. [16] employ power measurements to deduce the impact of shared uncore (i.e., not belonging to the individual CPU core itself, e.g., higher level cache) components and inter-core communication. They stress that actual measured power demand is strictly lower than the sum of estimated independent individual core powers, due to shared components. The model proposed by Hofmann et al. [70] discerns between core and uncore components of a CPU, with each having idle and dynamic power demand components. The dynamic power demand is in turn modeled in dependency to the CPU frequency, therefore reflecting integrated circuit (IC) voltage indirectly. This model does however not consider turbo P-state frequencies. Further models are based on CPU-specific performance counters (e.g., [39, 116]). The generality of these approaches is however limited, as performance counters are CPU specific and may therefore be unavailable on certain systems.

Some systems, e.g., file servers, are dominated by storage device power demand. Among others, Allalouf et al. [6] and Zedlewski et al. [188] model the power demand of storage devices. The former introduce STAMP (STorage modeling for Power), a method to estimate the power demand of a storage system on different levels. Power estimations range from storage controllers, arrays, or single disks. In the latter, the disk simulation environment Dempsey is proposed, aiming to accurately model disk power consumption. It does however require physical metering. solid-state drives (SSDs) are a more recent trend in storage device technology. Their power demand is modeled by, e.g., Zhang et al. [189].

A mandatory component in server operation is random access memory (RAM), providing an intermediate step in memory hierarchy between CPU registers and permanent storage. Power models for memory build on different methods, e.g., additive models [3, 100, 113] or based on performance counters [39, 97]. Many of these models require detailed knowledge on weighting parameters or energy demand of atomic operations in RAM.

Further models are available for the listed and many other components. An exhaustive presentation of component models would exceed the required brevity in the context of this chapter. For a more elaborate overview, the reader is again referred to, e.g., the survey of Dayarathna et al. [41].

2.2.3 System Level

On the next higher level of abstraction, full-system power models estimate the power demand of complete ICT devices. Rivoire et al. [156] propose a model that predicts a constant average power demand regardless of server utilization. Of course, this model has the benefit of not requiring any additional information. However, as idle power and maximum power demand are pushed further apart (e.g., by exploiting component sleep states), the model error increases.

Many other models base their estimations on the monitoring of a low number of components as a proxy for full-system utilization. Fan et al. [51] estimate system power using a linear model based on CPU utilization. The assumption of a linear relation with CPU load is strong, and is not valid for at least some modern multi-core CPUs [16]. Additionally, it is not applicable to systems whose power demand is not CPU-dominated (e.g., file servers) or workloads that are not CPU intensive. The model of Heath et al. [66] employs a similar linear model, however augments it with an additional consideration of HDD utilization. This way, the model is able to be applicable to a wider set of scenarios and reach higher accuracies [156]. Economou et al. [47] introduce the Mantis system, a full-system power estimation building on an individual CPU, RAM, Hard disk drive (HDD), and network interface card (NIC) models, augmented with additional monitoring of performance counters of the system, as far as available (e.g., the amount of instruction-level par-

allelism, the activity of the cache hierarchy, or the utilization of the floating-point unit). However, performance counters are may not be available homogeneously across different CPU generations or architectures, limiting the general applicability of this model. Although this model is similar to the one proposed in this work, it does not consider power beyond that of a server.

Some more recent approaches apply machine learning techniques to estimate server power demand [43, 79]. However, challenges inherent to machine learning approaches apply to these models: depending on the used algorithms, artificial neural networks require high numbers of training data and/or supervision during learning. Additionally, the models are often susceptible to overfitting, limiting reusability [41].

2.2.4 DC Level

Abstracting from individual devices, complete DC power models estimate the combined power of ICT equipment and the required auxiliary infrastructure. One of the most prominent ways of estimating DC facility power is the use of the PUE measure, as introduced by the Green Grid. It estimates a DC's total facility power by applying a scalar factor to the cumulative ICT device power. Although valid criticism of power usage effectiveness (PUE) exists (as it, e.g., does not include any notion of achieved ICT performance), it has become the de-facto standard in industry [87, 159]. Rahmani et al. [147] propose a modular model for full DC power demand estimation. While the general methodology is similar to the model presented in this work, it focuses on the auxiliary infrastructure of a DC (e.g., cooling) on a similar granularity as servers. Janacek [85] describes a DC-wide power model, based on models for UPS, servers and cooling. The overall goal of this work is the optimization of DC energy demand while assuming workload to be non-controllable. Also, specifically the power models of servers are considered on a higher abstraction level.

Research contributions by the author ranged from an overall overview on ICT energy demand [128], over analytical models to estimate the full-system power demand of servers [15] and models for the specific calculation of server idle power [17] to the estimation of power demand in the face of software migration between servers [110]. The model introduced in Chapter 4 is an extension of the work performed in [15], which in contrast to most of the previously presented models is not focused on a specific level of granularity. The approach of this work is bottom-up cross-level, starting at hardware components up to entire DC level, with a decreasing granularity. The rationale behind this decrease in granularity is that servers contribute more than 50 % to overall DC power demand [139].

2.3 Video Encoding

In the course of this work, digital video encoding is used as an example of an adaptive service. Adaptivity focuses on the ability of services to react to changes in the requirements towards them. This includes, e.g., changes in service utilization, resulting in a higher/lower demand for the underlying substrate performance or a failure of a substrate network device, requiring a modification to routing decisions. As these examples show, reactions may be required to ensure functional or non-functional software properties. There is a large body of related work in the direction of adaptive software available, both in academic literature and in applied practice. To keep a concise view on the specific topic, related work is focused on the subfield of adaptive video encoding.

To understand why digital video encoding is a well-suited service for further investigation, the basic working principle of modern video CODECs is explained in the following. Subsequently, related work in the field of video encoding and the subfield of adaptive video encoding is discussed. Video encoders exploit the strong temporal image redundancy in a series of video frames, i.e., try to reuse information that was already encoded to predict future video frames. To this end, the image is broken up into smaller segments, which are shifted in a way such as to compensate for the video motion between two adjacent frames. This is done by comparing an error measure (e.g., mean squared error (MSE)) among different candidate motion vectors $\overrightarrow{mv} = \begin{pmatrix} \text{Offset}_x \\ \text{Offset}_y \end{pmatrix}$ for block shifting. Figure 2.3 depicts this process.

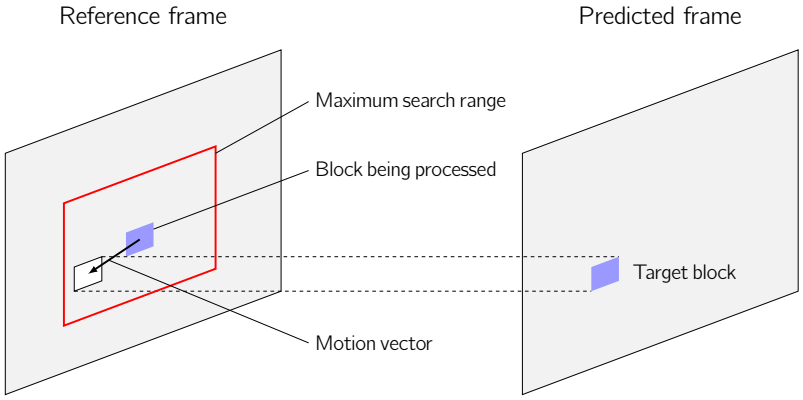


Figure 2.3: Motion estimation in a video encoding process.

After repeating this process of estimating video frame movement for all video frame segments, a residual image containing the remaining error information is stored. By combining the reference image, motion vectors and residual error image, the next video frame can be restored. This process is known as motion compensation. Calculating the offset vector $\overrightarrow{mv}$ (i.e., *motion estimation*) is the most computationally demanding sub-algorithm in the video encoding process. Zrida et al. [197]

quantify the fraction of motion estimation complexity in the encoding process to be higher than 77 % in the H.264 reference configuration. He et al. [65] give a more conservative estimate of around 40 %, however even in this case, it remains the most computationally complex subtask of video encoding.

The resource demand of video encoding is therefore also drastically increasing with higher video resolutions. While for a long time the Phase-Alternating-Line (PAL) and National Television Systems Committee (NTSC) video standards prevailed, in the last decades the subsequent introduction of the HD and UHD standards demanded for increasingly high video resolutions. Table 2.1 shows the specifications of several common video formats.

Standard	Width [px]	Height [px]	Pixel count
NTSC	720	480	345 600
PAL	702	576	404 352
1080p HD	1920	1080	2 073 600
4K UHD	3840	2160	8 294 400
8K UHD	7680	4320	33 177 600

Table 2.1: Video format specifications.

The 4K UHD standard has around 20 times more pixels in each video frame than the now obsolete PAL standard. To encode these high-resolution videos, a far higher computational complexity arises, as simple motion compensation techniques scale proportionally with the pixel count. According to a white paper published by Cisco [76], the percentage of connected flat-panel TV sets supporting 4K UHD will exceed 50 % by 2021. Video data is already a significant contributor to global data traffic and its importance is expected to increase even further in the future.

Extensive related work in the domain of video encoding is available in several subfields. Regarding video CODEC quality evaluations, some comparisons of subjective and objective assessment methods have been conducted, especially the CS MSU Graphics & Media Lab Video Group ran several evaluations concerning CODEC competitions featuring various MPEG-4 ASP & AVC implementations[10,11]. MPEG-4 SVC was analyzed in both objective and subjective tests [14, 180]. The results of the subjective evaluation of the SVC reference CODEC [14] are limited to the quality change if temporal levels are reduced in exchange for higher image quality. Many proposals for fast motion estimation algorithms are available [58, 99, 140, 194], significantly decreasing computational complexity compared to an exhaustive full search approach. Most of them limit the maximum motion vector magnitude and reduce candidate motion vector count by iteratively

[10] `http://www.compression.ru/video/codec_comparison/pdf/msu_subjective_codecs_comparison_en.pdf`, accessed 28.06.2018

[11] `http://www.compression.ru/video/quality_measure/subjective_metrics_info.html`, accessed 28.06.2018

searching narrower areas around the best candidate motion vector of the previous iteration. This does not guarantee an optimal solution, as the error gradient cannot generally be assumed to be without local minima. The author's own work combined subjective and objective evaluation of multi-dimensional scalability in MPEG-4 SVC [129].

There are several examples of adaptive software related to video streaming and graphics processing in industry today. One of the most prominent examples is the adaptive video resolution used on the YouTube video streaming platform. The adaptiveness is embedded in the video transmission and playback software: a feedback control is used to adjust video resolution according to available streaming bandwidth. More specifically, the video client continuously monitors video buffer filling (i.e., video content already downloaded but not yet played back). In case buffer filling falls below a threshold, the video delivered automatically switches to a lower resolution such as to avoid video interruptions due to buffer underruns. A second example for currently available adaptive applications are graphics engines (e.g., CRYENGINE[12]) capable of dynamic resolution rendering. Dynamic resolution changes the size of a rendered image dynamically such as to achieve a certain minimum number of images rendered per second. As Binks [26] shows, the computational complexity of rendering larger resolution images far outweighs that of rendering a smaller resolution image, even considering overhead for additional upscaling and filtering. Another related technique of dynamic resolution adaptation is NVIDIA Multi-Res Shading[13]. In this case, different rendering resolutions are applied to different areas of an image to account for peculiarities of virtual reality applications, i.e., distortions in image geometry. All of these techniques have a common goal: to reduce stress on an overburdened resource at runtime while delivering content to a human user. While adaptive video resolution on the YouTube platform is used to reduce the required transmission bandwidth to avoid buffer underruns, the two latter approaches are used to reduce load on a graphics processing unit (GPU).

Several papers cover the optimization of adaptive video streaming for specific network architectures (e.g., software-defined networking (SDN) [48, 55], wireless [24, 90, 163] or peer-to-peer (P2P) [86, 124, 125]) by taking the peculiarities of the underlying networks into account. Further publications are centered on the employed control method (e.g., control-theoretic [119, 186, 191] or game-theoretic [94]). Other, more closely related work in the direction of optimizing adaptive video quality was performed by, e.g., Moldovan et al. [122]. The work is optimizing YouTube video quality adaptation decisions based on two different approaches, outperforming the standard quality adaptation algorithm. In [69], Hoßfeld et al. present a subjective evaluation of video content streamed using the HTTP adaptive streaming (HAS) technology. Based on the results, a method to compute the QoE-optimal adaptation strategy for HAS on a per user basis with mixed-integer linear programming is presented.

[12] `https://www.cryengine.com/`, accessed 12.12.2018.
[13] `https://developer.nvidia.com/vrworks/graphics/multiresshading`, accessed 12.12.2018.

Specifically, regarding the impact of changes to frame rate and resolution in video streams on objective QoE metrics, research has been performed by, e.g., Zinner et al. [196]. The paper evaluates the influence of spatial and temporal decimation using different input video sequences. Results are given in both SSIM and VQM objective video quality metrics. The paper does cover both spatial and temporal scalability, however the resulting power demand during encoding is not considered. The author of [171] investigates the power demand of a live video encoding scenario using different software encoders (`x264`, `Vpxenc`, `x265` and `Kvazaar`). Although the scenario is similar to the one used in this work, the only scalability dimension considered in [171] is video bit rate and encoding quality preset.

Most of the presented approaches are concerned with adapting video to bandwidth or buffer limitations while optimizing QoE. However, only very few publications consider the power demand of different video encoder configurations. To the best of the author's knowledge, there is currently no optimization of power/QoE tradeoff that considers three scalability dimensions (resolution, frame rate and bit rate) for a `x264` live-encoding scenario. In previous own work [129], the author shows that an optimal choice regarding adaptation may be content-dependent. However, similar to the former approaches, the work focuses on a video CODEC's capabilities to adapt to resource limitations like bandwidth scarcity.

The case for extending the consideration of electrical power as a limited resource, specifically in the context of video processing, is made in this thesis. More concretely, the specific use case of an adaptive live video encoding service is investigated. Some previous work is available in this narrow subfield: Sachs et al. [160] use an adaptive motion compensation accuracy and a modified discrete cosine transform (DCT) to create an energy adaptive video encoder. The goal of their work is to minimize encoder energy demand. This is in contrast to this work, which optimizes QoE under power constraints using multi-dimensional scaling. Work by Kim et al. [91] proposes a variable frame rate encoder to adapt temporal video resolution to available bandwidth. Their approach is based on the ITU-T H.263 video compression standard[14] and does consider only one scalability dimension. He et al. [65] use a similar approach to optimize quality of mobile video streaming. As the use case is mobile video, in addition to only considering video encoder complexity, a wireless data transmission is taken into account. While their work in addition to changes in motion compensation considers temporal scalability, it is focused on mobile environments with wireless links and it does not consider spatial scalability.

In contrast to existing approaches, this work focuses on adaptive video encoding in DC environments. Three independent dimensions of scalability are covered: temporal, spatial and bit rate

[14] `https://www.itu.int/rec/T-REC-H.263/`, accessed 16.07.2018.

scalability. Additionally, the tradeoff between encoder power demand and resulting QoE is investigated and optimized.

2.4 Model Checking

Model checking is a formal method for verification of certain system properties. A model checker investigates all possible system states in a brute force way: by traversing all states and checking for the compliance to a certain objective, guarantees regarding the fulfillment or possible violation of a property can be given. As input to the system, a model checker requires a formalized description of both the system under investigation (in the form of a system model) and the requirements (in the form of property specifications). This is depicted—adapted from the work of Baier et al. [10]—in Figure 2.4. However, if a system is subject to stochastic influences from the system's environment (e.g., weather, component reliability, etc.), the Boolean output (i.e., hard guarantees) of a model checker may not be able to adequately represent the system model evaluation anymore. In these situations, PMC offers the possibility to quantify the probability of achieving a certain objective in the system.

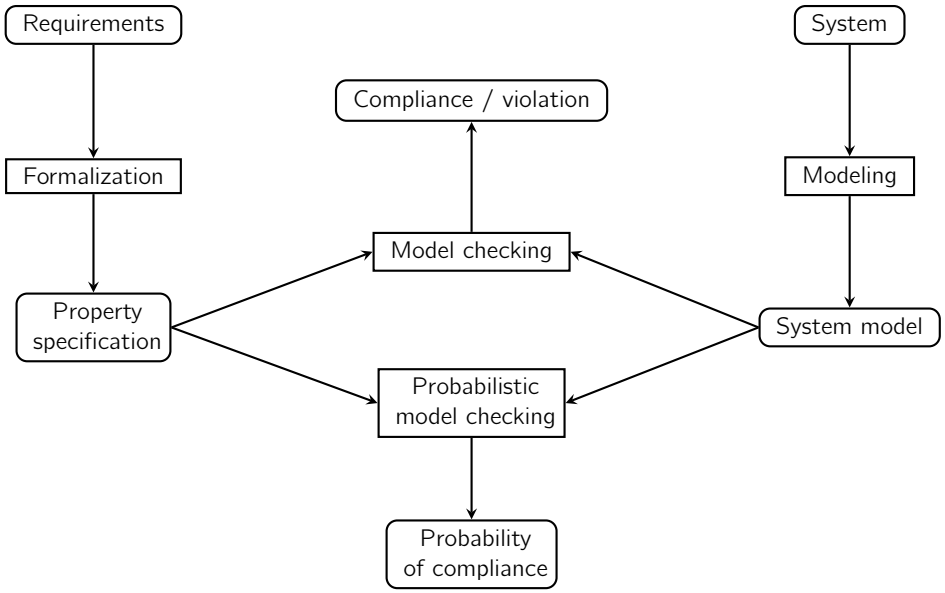


Figure 2.4: Conceptual steps in (probabilistic) model checking (adapted from [10]).

In the course of this work, PMC is used to analyze the safety of DCs participating in a continuous D/R program in a future power grid. DCs are expected to host safety-critical services for an increasing number of sectors in the future (e.g., communication, finance, etc.). Therefore, a

formal assessment of the system may be required to achieve the required safety certification (cf. Chapter 6). In practice, utilities currently often rely on employing power grid simulation tools to plan and verify changes to power grid operation (e.g.DIgSILENT PowerFactory[15]), however formal guarantees regarding the success of D/R actions can hardly be given.

There are several research initiatives to formalize methods applied to the power system: Hackenberg et al. [62] analyze possibilities of applying formal software engineering methods to smart grids. The work of Yüksel et al. [187] focuses on statistical model checking in a broader smart grid context. The authors evaluate a specific use case of a Chinese smart grid and use the PRISM model checking software to verify performance and energy demand properties. Mahmood et al. [112] also employ PRISM to formally analyze a smart grid. Their work—similarly to this thesis—uses PMC based on Markovian models. However, it focuses specifically on protection systems. Drozdov et al. [46] also investigate protection functions, however instead of PRISM base their analysis on the NuSMV model checker. Mancini et al. [114] use statistical model checking to verify the safety of employing time-dependent tariffs to steer peak demand. However, the authors check the aggregated power demand of multiple users which are comprised of private households only, in contrast to this work. Other publications deal with model checking in power systems focused on optimizing maintenance schedules [137] or distributed fault localization, isolation, and supply restoration [103], for example.

The author is however not aware of any work using PMC to analyze D/R in the context of DCs except for his own co-authored publication [11]. It deals with a traditional D/R approach in which a DC aims at keeping its power demand inside a certain range during a limited time duration.

2.5 Positioning of This Thesis

This thesis is positioned at the intersection of several specific subfields of the research areas covered in this chapter. Regarding D/R, this work focuses on a continuous D/R scheme, i.e., a permanent, rolling horizon guidance on power demand setpoints. Furthermore, it explicitly distinguishes between deriving the guidance values and implementing them with the help of flexibilities on customer side. This helps protecting sensitive information concerning used equipment and employed processes on consumer side, which might be inferred from sharing explicit values regarding flexibilities. This work is therefore simultaneously positioned in the fields of DC D/R, continuous D/R, privacy and optimization.

With respect to power demand modeling, the method employed in this thesis encompasses a larger scope than most power models in literature. The model is based on the investigation of independent

[15] `https://www.digsilent.de/en/`, accessed 03.01.2018.

components, while also considering the influence of additional DC infrastructure. The work thus has connections to analytical modeling on component level while considering the additional impact of auxiliary DC infrastructure. While models exist for many different scopes, e.g., from transistor level to entire DCs, the author is not aware of any model that features a reusable decomposition that spans from component level to entire DCs.

Software service adaptivity, in contrast to most related work, is considered in a non-mobile context while keeping power limitations as primary constraint. Previous work on adaptive/scalable applications, especially in the multimedia domain, mostly considers bandwidth limitations. Energy conservation is a broadly covered concern in mobile computing scenarios. The specific use case of live video encoding in a non-mobile, power constrained environment focused on in this work has connections to research in the areas of self-adaptive systems, digital video encoding, and energy efficiency.

The model checking aspect of this work is a new application of PMC. It presents a new way of modeling a DC and its service and power properties by means of Markov decision processes (MDPs). Both objectives towards the power grid and towards the DC side are captured by standard temporal operators. In contrast to most other work, it therefore has connections to the specific subfields of model checking of power and ICT systems.

This thesis investigates the entire D/R system encompassing the deriving of a reasonable power use guidance, hardware power demand models, software adaptivity and model checking. To the best of the author's knowledge, this specific combination of research fields is not found in other publications.

A goal without a plan is just a wish.

Antoine de Saint-Exupéry

3

Power Planning

This chapter contains material previously published in [127] and [126].

The work presented in this thesis aims to assist in a reliable and efficient operation of a public power grid. To this end, the power demand of large consumers of electrical energy is influenced in a way such as to reach a certain target value. This process requires two steps: the regarded consumer needs to be provided with *guidance*, i.e., the power demand that should be exhibited by the consumer needs to be derived and communicated. After this guidance is obtained, *control* over the power demand of the consumer needs to be exerted to match the numbers given by the guidance. The following chapter discusses the former requirement: how to derive the target value of power demand while keeping both the costs of participating loads and goals of a grid authority (GA) in mind. In the following, the process of deriving a short-term plan on the provisioning of power to a consumer is referred to as *power planning*. This term is derived from the term *energy planning*, which Prasad et al. [144] define as "[...] a roadmap for meeting the energy needs of a nation and is accomplished by considering multiple factors such as technology, economy, environment, and the

society that impact the national energy issues". In contrast to energy planning, power planning is concerned with short term plans on power demand targets.

Section 3.1 introduces the scenario focused throughout this thesis. A brief outline of changes in power generation in future energy systems is given. Section 3.2 defines the challenge investigated in this chapter more closely. A hypothetical electrical load should adapt its power demand in a way such as to maximize the uptake of renewable energy while not overburdening its own flexibility capabilities. The following Section 3.3 introduces a running example employed throughout this work. The running example is intentionally kept abstract, it is defined more concretely in the subsequent chapters as required. In Section 3.4 the assumptions taken in this chapter are stated. Section 3.5 argues for the separation of power plan generation and scheduling of flexibilities on load side. In a first step, two cases for different placements of actual power plan calculations are investigated. Second, desirable properties (i.e., easily implementable by the load) of a power plan are discussed. The following Section 3.6 introduces the chosen optimization method to derive a power plan. The objective function is presented, with special focus on the cost function. Section 3.7 evaluates the proposed optimization function. To this end, a sensitivity analysis regarding the four parameters of the cost function is performed in a one-at-a-time manner. Finally, Section 3.8 concludes this chapter and gives an overview on possible future research directions.

3.1 Power Demand in Future Energy Systems

Aims regarding the target value of power demand largely depend on the point of view of the stakeholder. For a private household, power demand should be kept to a minimum, as cost increases linearly with demand. Of course, energy resellers may have the opposite goal, as selling more electrical energy increases their profits. However, these are both monetary goals largely unrelated to actual technical, ecological and other aspects of grid operation.

With regard to technical grid operation, an adjustment of power demand on consumer side may support a power grid in two major cases: first, in emergency situations requiring cooperation to keep grid stability, also known as *red phase* or *network phase* in the smart grid traffic light concept[16]. These cases are characterized by a deviation from normal operation that threatens the security of supply and requires the enactment of non-market-based regulation. In these situations, power grid load is shed without prior notification to an extent required to restore a stable grid operation. Second, power demand adjustments may be feasible to optimize grid operation regarding ecological and economical operation or power quality, known also as *amber phase*. In the following, the latter is focused.

[16] `https://www.bdew.de/internet.nsf/id/816417E68269AECEC1257A1E0045E51C/$file/Smart%20Grid%20Traffic%20Light%20Concept.pdf`, accessed 26.07.2018.

Regarding ecological aspects, according to the U.S. Energy Information Administration, electricity generation is accountable for around 35 % of greenhouse gas (GHG) emissions in the U.S.[17]. In the face of climate change and limited supply of fossil fuels, it is a desirable goal to decrease these emissions and/or reduce the consumption of fossil fuels. While an overall reduction in energy demand may generally be achieved by reducing the operation of energy-intensive processes or increasing their energy efficiency, the effectiveness of these approaches may be limited. In the former case, productivity or consumer comfort may be reduced, which is often not acceptable. In the latter, reluctance of investment in more energy-efficient equipment (or the required research and development) limit wide-spread reduction of energy requirements. Additionally, the rebound effect [21] (i.e., an increased use of an appliance after more energy-efficiency has been achieved) may show detrimental effects.

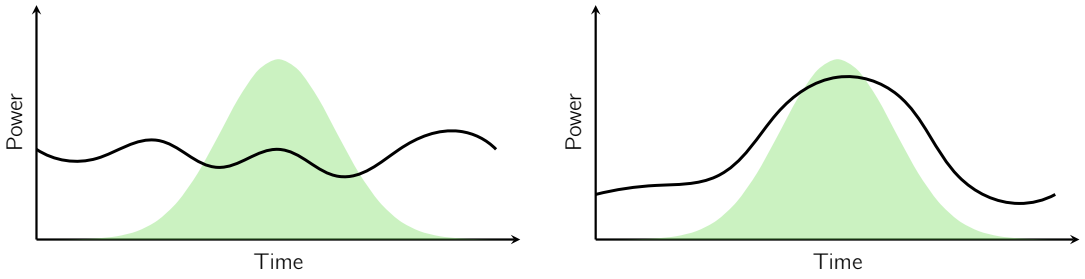
Another generally accepted way of action is to increase the coverage of electricity demand by means of renewable energy. An increased fraction of renewable energy in the energy mix poses new challenges to power grid management, as outlined in Chapter 1. Depending on the type of power quality impairment, different consumer flexibilities may be requested:

- To tackle a challenge in a local phenomenon like voltage, power demand may either be shifted temporally or spatially to relieve local grid stress. Additionally, a change in reactive power contribution may support voltage regulation.
- Grid-wide phenomena, e.g., AC frequency, require a large change on either generation or demand side to remediate a grid power imbalance. Therefore, a spatial power demand shift would need to be of very high distance (e.g., to outside the interconnected European grid). A temporal shift is equally feasible as in the local scenario.

In contrast to traditional power generation, power from volatile renewable sources can only be reduced by wasting free generation capabilities (i.e., lowering generator output will not result in a lower investment of fuel). Therefore, it is a viable goal to use all power generated by renewable sources to their full extent to reduce power requirements from generators powered by fossil fuels. To avoid curtailment of renewable energy feed-in, new ways of adaptation are required. The approach focused in the following is to reduce electricity curtailment from renewable sources by temporal power shifting. The concept of making power draw coincide with times of high renewable power generation is illustrated in Figures 3.1a and 3.1b.

Another way of increasing renewable feed-in efficiency—minimizing transport losses—is covered in the author's previous work [127] and is not considered further in this thesis. In summary, both

[17] `https://www.eia.gov/tools/faqs/faq.php?id=77&t=11`, accessed 26.04.2018.



(a) Renewable generation oblivious power draw.

(b) Suggested adapted power demand.

Figure 3.1: High level concept of power demand adaptation.

for power quality management and improved renewable energy uptake, temporal power demand shifting capabilities on demand side are desirable. This is investigated in more detail in the following sections.

3.2 Problem Definition

The challenge indicated in Section 3.1 is now focused more closely. To adapt the amount of power demanded by a load, first a general concept of an electrical load is derived. The amount of power delivered to systems powered by electricity usually changes depending on the amount of output currently demanded. Examples include, e.g., electric heating (more heat → more power) or motors (more speed/torque → more power). However, in case a varying amount of output generated by a system can be accepted, other inputs or outputs of this system may be used to provide *guidance* to the system. A well-known example for such systems is combined heat and power plants (CHP), which may be operated in either power-driven or heat-driven mode [63]. In the former case, power to be generated is set to a certain value, while heat is treated as a "byproduct", available to an extent that is dissipated during power generation. In heat-driven operation, roles of heat and power are interchanged. However, for any system input or output to act as guidance value, the dependency between system operating point and this input or output needs to be quantifiable. Figure 3.2 shows a schematic view of an open system, i.e., a system which interacts with its surrounding environment (for details the reader is referred to literature by, e.g., von Bertalanffy et al. [175]).

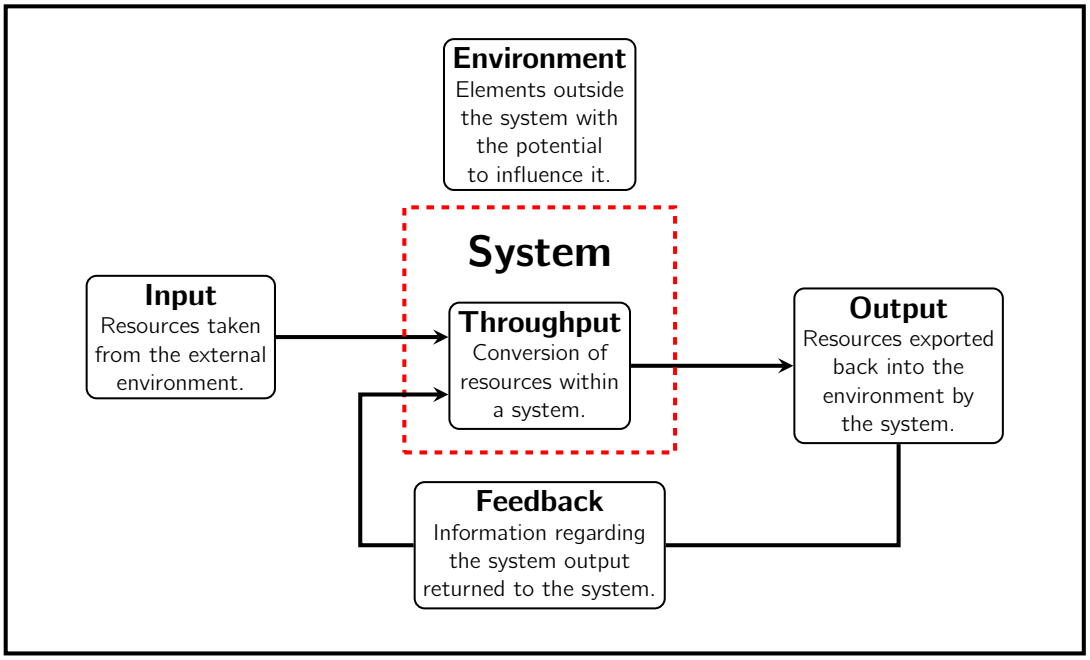


Figure 3.2: Conceptual model of an open system (adapted from [88]).

In this work, the system input of electrical power acts as guidance value. For this to be feasible, the following conditions have to be met:

1. A target electrical power input to the system has to be derived. This value should benefit the progress towards an overall higher-level goal.
2. Measures to enact control on the power demanding components operated inside the system and their potential side effects have to be identified.
3. For certain systems (e.g., those contributing to critical infrastructures), detrimental effects on system safety need to be considered.
4. The influence of operating point modifications on electrical power draw need to be quantified.

The time series of guidance values is henceforth referred to as *power plan*, which is formally specified in Definition 3.1.

Defintion 3.1

A power plan is a sequence of 2-tuples (t, p), with $t > 0 \in \mathbb{N}$ denoting a relative point in time in the future and $p \geq 0 \in \mathbb{N}$ an associated target power demand value.

Two stakeholders in the system's environment are involved in the process of defining the target power plan: first, the load adapting to the power targets, if applicable also in representation of its business goals and its potential clients. It has to ensure that its goals are not interfered with by adapting to power targets deviating too strongly from baseline operation. Second, a GA representing interests regarding the power grid state in certain areas, and thus also regarding power demand of consumers in these areas. This role may be taken by different entities, e.g., a DSO that may want to influence consumers in its grid to optimize power plant scheduling and increase power quality. Virtual power plants may gain additional flexibility in power generation/draw by influencing its prosumers. Another example is future smart grid scenarios, in which a more general energy management authority may have goals regarding the renewable energy utilization of consumers inside its grid. Several additional scenarios are imaginable. Regardless of the concrete entity behind a GA, a power plan can only be considered successful if properly implemented. Thus, it has to balance interests of both GA and load. To this end, an objective function is defined to model the consumer's required effort to adapt to a requested power plan in Section 3.6. This serves as algorithmic foundation of a component henceforth referred to as power planner (PP).

3.3 Running Example

Throughout the course of this work, a running example is employed to enhance overall coherence. This example is specified with increasing detail in the upcoming chapters. At this point, it is assumed that the consumer is a load exhibiting a baseline power demand that can be approximated by the G3 BDEW standard load profile, i.e., a power demand typically exhibited by a continuously operating commercial consumer. Figure 3.3 shows a 24 hour example of the G3 profile[18].

In Chapter 4, the further peculiarities of this consumer are elaborated on. The consumer operates special equipment and due to its business strategy does not want to disclose the exact type of devices operated. Therefore, it is also unwilling to disclose its exact per-device flexibilities regarding power demand. As shown by, e.g., Reinhardt et al. [153], having information on power demand details is often sufficient to identify a device. To account for the consumer's privacy concerns while at the same time enabling the generation of a power plan tailored towards the flexibilities of the

[18] http://www.kommenergie.de/fileadmin/media/Hauptmenue/03_Unser_Netz/06_Standardlastprofil/SLP_2011.zip, accessed 18.08.2017

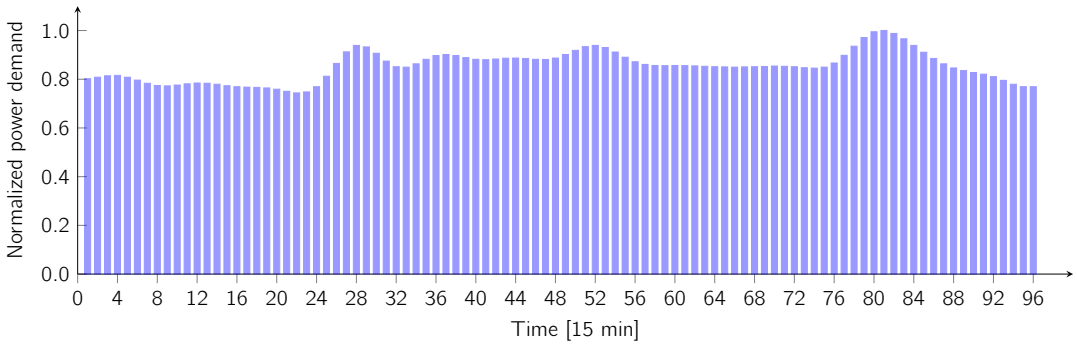


Figure 3.3: Example 24 hour trace extracted from G3 standard load profile.

consumer, Section 3.5 describes the interactions required by the stakeholders inside the regarded ecosystem.

3.4 Assumptions

In the subsequent sections, several assumptions are taken to limit the scope of this chapter. These are given in the following.

1. In this work, it is assumed that a PP is used as means to increase the uptake of renewable energy while allowing a consumer to express its aggregated flexibility to change their planned consumption. This is formally expressed in the objective function given in Section 3.6.1. This optimization goal could be exchanged by any other preference, as long as it can be represented by a value assigned to each time slot.

2. It is assumed that the load has the capability to shift some or all of its electrical power demand in time to a certain extent. These flexibilities are non-depleting, i.e., their availability is not impacted by previous adaptation actions. Although not generally applicable, this assumption fits the specific use case focused in the subsequent chapters of this thesis. The additional implications of considering depleting flexibilities are briefly discussed in the following however.

3. The load is assumed to be connected to a power grid with an energy mix containing a time-varying amount of renewable energy. Grid connections with a constant share of renewable energy can be handled by the approach, however changing the baseline power demand is infeasible as the uptake of renewable energy cannot be influenced by shifting demand. Power plans are devised centrally by a GA and subsequently communicated to the load. As a further simplification, the G3 load profile is approximated by a constant baseline power demand. This

choice is not limiting generality, as the approach supports any baseline power demand within the flexibility range of a load.

4. It is assumed that the grid infrastructure is sufficiently capable to deliver any amount of power within the limits of the load at any point in time. In other words, the grid infrastructure does not impose any limitations on the power plan. This assumption can be relaxed by adding time-dependent hard upper limits to the power plan, representing constraints imposed by the power grid infrastructure.

5. The approach discussed in this chapter focuses the power planning for a single load. It is assumed that a GA is capable of coordinating the power draw of several loads inside its managed grid and influencing the power plans accordingly. The means to achieve this coordination is not part of this work.

3.5 Separation of Concerns: A Dedicated Power Planner

The PP utilizes information on renewable power availability and grid energy mix. Upon completion, the power plan is communicated to a load to implement the according measures to adapt its power demand to match this power plan. While there are already several approaches to combined power planning and its implementation (cf. Chapter 2), there are strong arguments for why it is useful to decouple power planning from the actual implementation of power adaptation mechanisms on consumer side.

Generally, it has to be assumed that separate entities are responsible for power planning and technical implementation of measures to reach the desired power demand. On the one hand, a GA is responsible for managing its power distribution area, including power quality, voltage stability and power plant scheduling. These tasks require working with confidential or at least sensitive information, which a GA is unlikely to share with consumers inside its grid [121]. On the other hand, large energy consumers inside the aforementioned grid area have a clear view of their flexibilities in scheduling work, but this information may also be confidential. These two entities would have to either exchange sensitive information or use a trusted third party to use an integrated power planning tool, neither of which are attractive options. Additionally, it may not be possible to express all scheduling constraints in a way that is compatible with a specific optimization algorithm used for power planning, which limits the applicability of an integrated planning and scheduling approach that uses this tool. Splitting planning from scheduling enables a dedicated first computation of an appropriate power plan using mathematical optimization, and in a second step an independent implementation using a different set of scheduling tools that can deal with complex requirements. Figure 3.4 shows the proposed information exchange between load and GA.

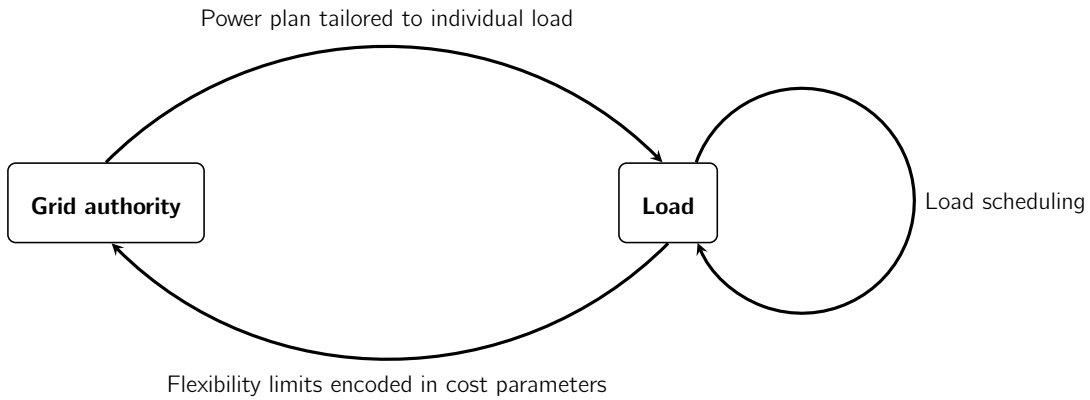


Figure 3.4: Data exchange between grid authority and load in the proposed scheme.

It is assumed that the goal of a power plan from the GA's point of view is to suggest a power plan to a flexible load to increase its uptake of renewable energy. Assuming a load is capable of shifting its power demand to a certain extent without affecting overall energy demand, this also implies that power demand will drop below baseline demand at times of comparatively low renewable generation. This way, it is possible to reduce the magnitude of required adaptations in conventional (mostly fossil fuel-based) electricity generation. At the same time, a power plan is only useful if it is followed by the consumer. It therefore has to consider limitations regarding consumer load and flexibility. Examples for consumer limitations include, e.g., different levels of flexibility with respect to changes in the power plan over time, oscillations in the power plan, and power ramping speed. Thus, the *power planning problem* is formulated as follows: given information on renewable energy availability and the cost function parameters of the consumer, derive a time series of suggested power values that maximizes the uptake of renewable energy while taking into account costs associated with deviating from a consumer's baseline power demand.

Dividing an integrated approach into two parts (planning and scheduling) requires accounting for potential difficulties in realizing a power plan. To this end, certain qualities of a power plan need to be identified which help to ensure that with high confidence a consumer is capable of implementing it with sufficient accuracy. A combined PP and scheduler is able to avoid this challenge, as it cannot propose a plan without the respective schedule. A potential scenario is a power plan that maximizes the objective of increased renewable uptake, but proposes a highly oscillating power plan due to fluctuations in renewable generation (e.g., due to intermittent cloud cover). This may create challenges when deriving a feasible schedule that matches the plan. For example, sharp fluctuations in a power plan would cause frequent interruptions and subsequent restarts of processes in order to decrease (or increase, respectively) power consumption, which may be harmful to lifetime of certain equipment [102]. The power planning algorithm thus has to include a notion

of properties which make a power plan specifically hard to implement. To this end, consumer costs are incorporated into an objective function, including weighting parameters that can be adjusted to match the specific requirements of different loads.

3.5.1 Power Planner Location

Power plans may be generated in two different ways: distributed or centralized. Distributed power plan generation is performed autonomously by each load participating in the system, adhering to objectives set a priori by a GA. It is assumed that a GA follows one or more high level *goals*, i.e., aims which are generally too abstract to directly implement in a power plan (e.g., minimize renewable energy curtailment in its grid area). To implement measures to fulfill these goals, they have to be translated into more concrete *objectives* (e.g., shift power demand to times with high shares of renewable energy). Objectives, in turn, can be further subdivided into constraints and optimization targets to be implemented by an optimizer. The author proposed a similar concept in [126].

Given regularly updated information on grid power properties is available to a load, this approach allows for a continuous adaptation towards moving targets, e.g., maximizing the amount of renewable energy used. However, it is limited to statically defined objectives which need to be communicated before a new power plan is calculated or must be translated to traditional D/R requests (and be accounted for in contracts). Although a GA may tailor objectives given to each load inside the program, uncertainty regarding the derived power plans (and therefore also their superposition) remains on GA side. Figure 3.5 shows a schematic view of the distributed power planning approach.

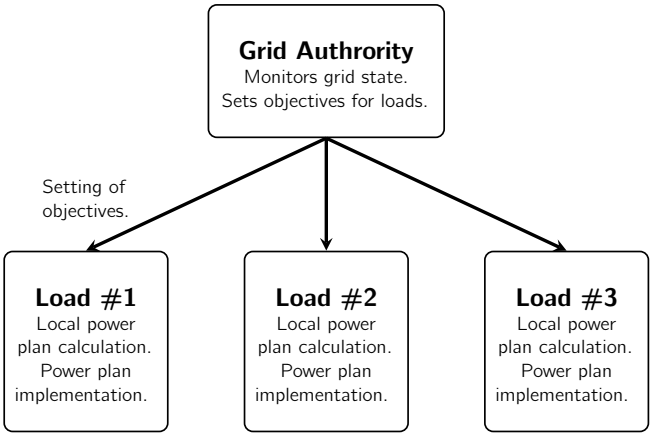


Figure 3.5: Distributed power plan generation scheme.

Centralized power plan generation reverses the point of power planning result uncertainty. In this setting, a GA has complete control over power planning and may include any desired properties in a power plan. However, each load participating in the program has specific constraints regarding its business operations and scheduling of its flexibilities. Ultimately, a power plan is only beneficial to the overall goal of a GA if correctly implemented, which strongly depends on not overburdening a load. Thus, a GA needs to be informed of these restrictions regarding power plan properties. Using this knowledge, a power plan specifically tailored to the flexibilities of each load may be set by a GA. Figure 3.6 shows a schematic view of the centralized power planning approach.

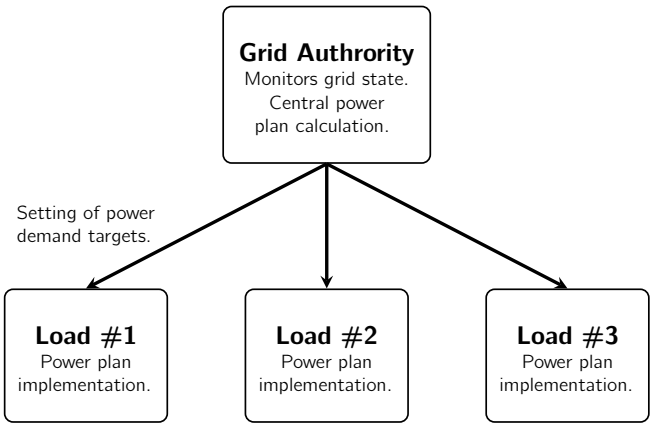


Figure 3.6: Centralized power plan generation scheme.

The power plan provides a guideline on how much power to consume at which time. This plan has to be provided in a continuous manner, updating while time progresses. To this end, a rolling horizon approach is employed: from the present time, a power plan provides a guide on power demand for a certain time into the future. The amount of time into the future is henceforth referred to as *planning horizon* (PH). A rolling horizon approach provides two major advantages: first, updating the power plan as time progresses provides more accurate forecasts on renewable power availability. Second, a regular refresh of the power plan allows to compensate for unexpected deviations from requested power demand. In the following, the option of centrally calculating power plans is investigated further. The objective is assumed to consist of maximizing the share of renewable energy in the total electrical energy used by a load.

3.5.2 Consumer Concerns

To increase applicability of the PP to a wide variety of loads, the cost of a power plan is modeled in a parameterized form, penalizing possibly challenging characteristics of a power plan. The characteristics considered in the following are:

- deviation from unaltered baseline power,
- high frequency power changes, and
- low lead time power changes.

To reflect the peculiarities of a specific load, each of these terms may be assigned an additional weighting factor. More details regarding the choice of weights are given in Section 3.7. An important additional concern is the—despite performing power adaptation—continued capability of a load to provide a service quality similar to that of a baseline scenario. To realize this, the overall energy amount suggested by a new power plan should remain unchanged from the baseline scenario for a certain time period. Overall energy demand may not always properly reflect service quality, however, for certain types of loads (e.g., electric cooling/heating or overnight EV battery charging) it is well suited as a measure of service delivered.

3.6 Methodology and Formulation

To provide a continuous guidance in form of a target power demand, a rolling horizon approach requires the PP to generate an updated power plan at each time slot over a sliding window. In the following examples, a window length of 24 hours is assumed, a period in which accurate forecasts on renewable energy generation can be expected [77]. Additionally, this window length is well suited to capture the diurnal pattern of, e.g., solar power generation.

3.6.1 Formulation

The creation of a power plan is formulated as an optimization problem regarding the average power demand planned in each time slot $i \in \mathbb{N}$ in $[1, \mathrm{PH}]$. A description of the parameters and variables is given in Table 3.1.

Name	*Description [unit]*
Parameters	
$P^{old}_{PP,i}$	Power allocated to time slot i in previous power plan [W]
$RenPercent_{PP,i}$	Forecast of renewable energy share in the grid in time slot i
$P^{min}_{PP,i}$	Minimum power required by the consumer in time slot i [W]
$P^{max}_{PP,i}$	Maximum power the consumer is able to consume in time slot i [W]
$\alpha, \beta, \gamma, \delta$	Weight parameters for cost function
PH	Number of continuous time slots considered
Variables	
$P^{new}_{PP,i}$	Power allocated to time slot i [W]

Table 3.1: Notation used in the optimization problem formulation.

Given $P^{old}_{PP,i}$, $RenPercent_{PP,i}$, $P^{min}_{PP,i}$, $P^{max}_{PP,i}$, and PH, the optimization problem is formulated in Expression (3.1).

$$\max_{P^{new}_{PP,i}} \sum_{i=1}^{PH} \left(P^{new}_{PP,i} RenPercent_{PP,i} - Cost_i \right), \tag{3.1}$$

subject to the constraints given in Inequality (3.2) and Equation (3.3).

$$P^{min}_{PP,i} \leq P^{new}_{PP,i} \leq P^{max}_{PP,i} \qquad \forall i \in [1, PH], \text{ and} \tag{3.2}$$

$$\sum_{i=1}^{PH} P^{old}_{PP,i} = \sum_{i=1}^{PH} P^{new}_{PP,i}. \tag{3.3}$$

The objective function maximizes the amount of renewable energy drawn from the grid, while adhering to the given constraints. It is additionally affected by a cost function that is discussed more elaborately in Section 3.6.2.

Constraint (3.2) specifies upper and lower bounds to the values assigned to each time slot. These are subject to technical limitations of the load or minimum service requirements that need to be achieved and require a certain amount of power. In case more detailed information on a load's ramp speeds, i.e., maximum speed at which a load is capable to change its power demand, is available, minimum and maximum power may be set dynamically depending on the power assigned to the current time slot. This may be achieved by applying the concept of power envelopes similar to the work of Nosair et al. [131]. Figure 3.7 shows the concept of power envelopes adapted to the calculation of power plans.

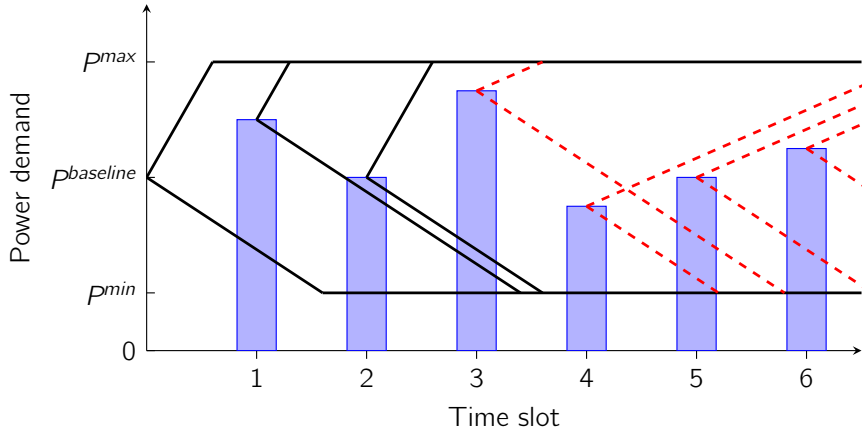


Figure 3.7: Power envelopes applied to a power plan.

By applying a power envelope to each time slot, placing its root at the current power demand level, more precise upper and lower bounds on power demand can be set for upcoming time slots. It is noteworthy that if there are flexibilities which cannot be applied continuously (e.g., discharging a battery), the power envelope's shape may change throughout the power planning horizon. In the example depicted in Figure 3.7, in time slot 2 a high ramp speed flexibility is applied which is not available starting from the following time slot 3. This is reflected in a modified power envelope (dashed red lines in Figure 3.7). In the example, the power assigned to time slot 4 is invalid, as it violates the lower bound of the power envelope (i.e., is outside the maximum rate of power decrease of the load).

Constraint (3.3) ensures that the total amount of energy delivered in a period of PH time slots of the new power plan is the same as in the baseline power plan. This constraint is used as a proxy for service quality, assuming that it captures the relationship between energy demand and service quality delivered with sufficient accuracy. If required, this constraint may be relaxed to allow for an overall increase or decrease of energy used to cope with a grid energy surplus/scarcity. A modified version of constraint (3.3) that allows for a change in total energy supply is given in Equation (3.4).

$$\sum_{i=1}^{PH} P_{PP,i}^{old} = \varepsilon \sum_{i=1}^{PH} P_{PP,i}^{new}, \tag{3.4}$$

with $\varepsilon \in \mathbb{R}^+$ denoting a scalar factor determining the overall energy increase/decrease over the planning horizon.

3.6.2 Cost Function

Deriving a power plan solely limited by Constraints (3.2) and (3.3) results in a probably infeasible output: the time slots exhibiting a maximum share of renewable energy are assigned their respective maximum power $P^{max}_{PP,i}$, while less favorable time slots are assigned the respective minimum power allowed $P^{min}_{PP,i}$. Although a power plan with these properties is effectively optimizing towards a maximum use of renewable energy, it is likely to be hard to implement. To realize it, a load would need to repeatedly engage and disengage all of its flexibility potential (e.g., start and stop machines in operation). Therefore, apart from the constraints given in Section 3.6.1, additional costs need to be considered when creating a power plan. The cost function should capture the consumer concerns outlined in Section 3.5.2. For the formulation of the cost function, it is assumed that the following statements capture the cost of implementing a power plan.

1. The power plan is harder to implement the further it deviates from the unmodified baseline power demand of the considered load.
2. Highly fluctuating power plans that include high frequency changes in power demand may be challenging to implement and potentially disturb the operation of the load.
3. Deviations and ramps often have a cost that is non-linear with respect to their magnitude.
4. Adapting to power plan changes in upcoming time slots is harder than in temporally distant time slots.

These items need to be captured in a mathematical way to enable an algorithmic optimization. Equation (3.5) gives a possible formulation.

$$\text{Cost}_i = \alpha \left(P^{baseline}_{PP,i} - P^{new}_{PP,i}\right)^2 + \beta \left(P^{new}_{PP,i} - P^{new}_{P,i-1}\right)^2 + \frac{\gamma \left(P^{new}_i - P^{old}_i\right)^2}{1 + \delta\, i} \tag{3.5}$$

with $\alpha, \beta, \gamma, \delta \in \mathbb{R}^+_0$ denoting weighting parameters used to reflect a load's peculiarities. α is a weighting factor signifying the overall capability of a load to deviate from its baseline power demand. Higher values of α indicate a higher cost of strong power adaptations. β is the weighting factor attributed to having sharp changes in power demand in the power plan. The second term is set to 0 for $i = 1$ (i.e., $P^{new}_0 = P^{new}_1$), as the first time slot has no predecessors from which

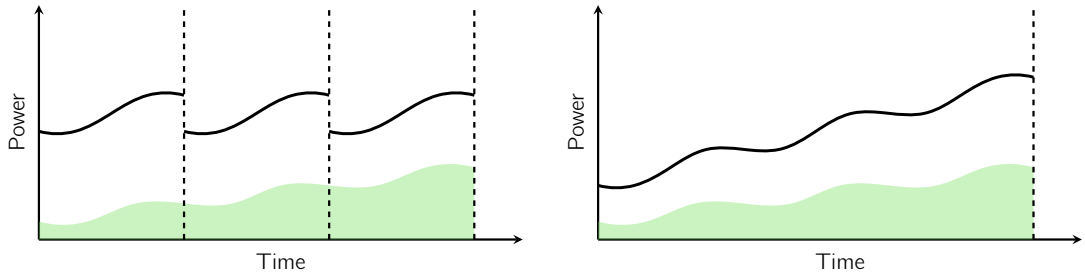
a difference could be calculated. γ is used to weight costs for subsequent changes—due to the rolling horizon approach—to an existing power plan. δ is a discount factor to promote changes to the power plan in temporally more distant time slots.

In addition to the mentioned objective and cost functions, further constraints may have to be incorporated in the optimization process. On the one hand, this is the minimum desired fixed planning horizon of the load. It is denoted as $\mathrm{PH}_{\mathrm{fixed}} \in \mathbb{N}$ and is used to fix the first $\mathrm{PH}_{\mathrm{fixed}}$ calculated time slots to be unchangeable in follow-up power planning iterations, increasing scheduling security for the load. This is a constraint required for loads that cannot reschedule their flexibilities on short notice (e.g., flexibilities with long ramp times). In the optimization formulation, $\mathrm{PH}_{\mathrm{fixed}}$ can be incorporated by creating linear constraints of form $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$. To fix the first n time slots, the respective number of rows from the top of the unit matrix $\mathbf{1}_{\mathbf{PH}}$ have to be added to matrix $\mathbf{A}$ and the first n slots of $\mathrm{PH}_{\mathrm{fixed}}$ calculated in the current power planning step have to be added to the solution vector $\mathbf{b}$. It is noteworthy that sudden changes to a power plan (due to, e.g., corrections of previous $RenPercent_i$ forecasting errors) may not be possible if a high value of $\mathrm{PH}_{\mathrm{fixed}}$ is used. Advantages gained by increasing forecasting accuracy over time may thus be limited.

Another parameter already mentioned before is the planning horizon PH. As the total energy amount delivered over the period of PH is equal to the baseline power demand over an equal amount of time, PH also acts as a limiter regarding the maximum temporal delay of energy provisioning. Lower values of PH ensure that the energy available in a power plan reaches equilibrium with baseline energy availability after fewer time slots. A high value of PH could result in a power plan assigning most power to temporally distant time slots, leaving only little power for other time slots. Generating power plans for the same time horizon with a low value of PH results in a subsequent calculation of independent power plans. A low value of PH also limits the ability to exploit long term changes in renewable energy availability: in case there are favorable conditions forecasted for temporally distant time slots beyond PH, power assigned to time slots until PH cannot be shifted to contribute to an increased uptake of renewable energy. Optimization of power distribution of the load is therefore limited to the variations of renewable power availability within PH. This is visualized in Figure 3.8a and 3.8b.

3.7 Evaluation

In the subsequent evaluation, a central power plan calculation on GA side and a cooperative consumer is assumed. Cases of consumers trying to exploit the system or even active malicious behavior are not covered in this work. In the following, the impact of changing the cost function parameters α, β, γ, and δ on the calculation of a power plan is analyzed. For this evaluation, a



(a) Low PH value. **(b)** High PH value.

Figure 3.8: Approximate power plans for different values of PH.

trace of power generation data from Germany over 72 hours is used[19]. Forecasting is assumed to be accurate. Further settings for the evaluation are listed in Table 3.2. The baseline scenario parameter choices are given in bold font.

Parameter	Value(s)
PH_{fixed}	1
PH	24
$P^{min}_{PP,i}$	50 kW
$P^{max}_{PP,i}$	500 kW
$P^{baseline}_{PP,i}$	275 kW
α	0, 0.1, **0.2**, . . . , 2.0
β	0, 0.5, **1.0**, . . . , 10.0
γ	0, 0.01, . . . , **0.20**
δ	0, **10**, . . . , 200

Table 3.2: Parameters used in the evaluation.

The MATLAB® software is used to evaluate the optimization using its internal solver `fmincon` (constrained nonlinear multivariable function solver). Power plan calculations are performed every hour, ending with time slot 48. The primary goal of the PP is to increase the uptake of renewable energy. However, subsequent plans in the rolling horizon calculation cannot be made from scratch, as this could lead to unstable plans inducing large changes between two rounds of calculations. To keep parameter values comparable between different loads, the dependence on concrete amounts of power demand needs to be eliminated. To this end, the power demand is normalized to the upper and lower bounds of power flexibility, allowing for a meaningful comparison between values of the cost function parameters. Figure 3.9 shows power plans calculated for the same scenario using different values for α.

[19] `https://transparency.entsoe.eu/generation/r2/actualGenerationPerProductionType/show`, accessed 13.04.2018

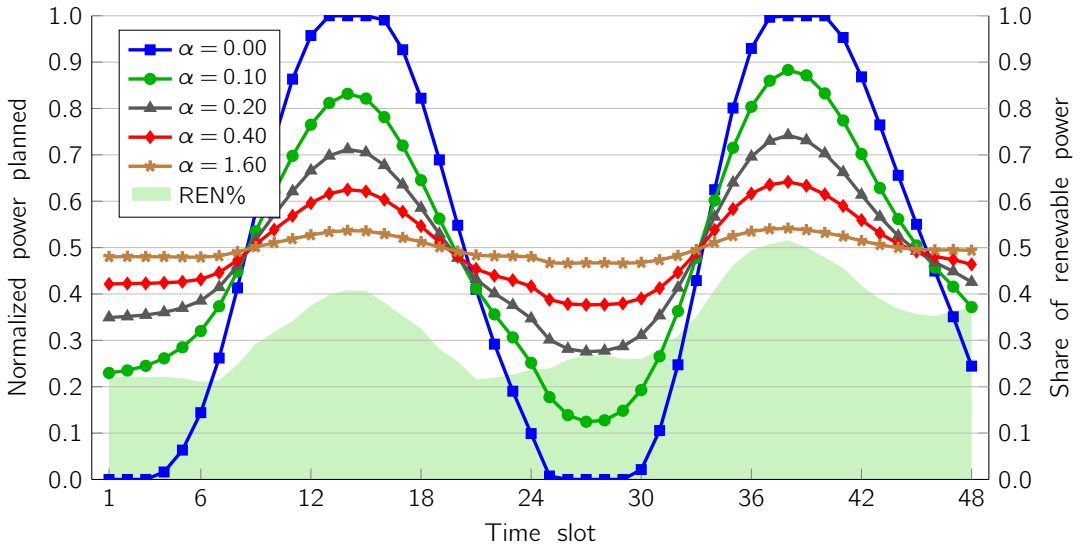


Figure 3.9: Resulting power plans for different values of α.

As expected, the results show an increasing reluctance of the optimizer to deviate from the baseline power demand. The parameter α is the cost weight assigned to the difference between the baseline power demand and new power plan. The numerical results are given in Table 3.3.

A power plan may be easier to follow if it does not have steep ramps. Oscillations in the renewable energy generation forecasts (e.g., due to intermittent cloud cover) may cause a power plan that tries to increase renewable energy use by following these oscillations. The parameter β is used to weight the reluctance of a load to quickly adapt its power demand. Figure 3.10 shows power plans calculated for the reference scenario using different values of β. The results show very abrupt changes in the power plan if the associated cost ($P^{new}_{PP,i} - P^{new}_{PP,i-1}$) is completely neglected ($\beta = 0$). Increasing values of β show decreasing differences in adjacent time slots. The capability of reducing power fluctuations between temporally adjacent time slots by increasing the value of β is additionally evaluated in a frequency domain analysis. It shows that parameter β effectively acts as a type of *lowpass filter*, attenuating high frequency contents in the power plan. Figure 3.11 shows the power spectral density of power plans obtained using $\beta = 0$, $\beta = 1.0$, and $\beta = 10.0$ respectively (DC signal content was removed prior to FFT analysis). The numerical results are given in Table 3.4.

α	β	γ	δ	Maximum deviation from normalized baseline	Achieved REN%
0.00				0.4999	37.54
0.10				0.3831	35.86
0.20				0.2422	34.70
0.30				0.1785	34.18
0.40				0.1418	33.89
0.50				0.1178	33.70
0.60				0.1008	33.57
0.70				0.0881	33.47
0.80				0.0783	33.39
0.90	1.00	10.00	0.20	0.0704	33.33
1.00				0.0640	33.28
1.10				0.0587	33.24
1.20				0.0542	33.21
1.30				0.0503	33.18
1.40				0.0470	33.15
1.50				0.0441	33.13
1.60				0.0415	33.11
1.70				0.0392	33.10
1.80				0.0371	33.08
1.90				0.0353	33.07
2.00				0.0336	33.05

Table 3.3: Detailed results for different values of α.

The parameter γ weights the cost assigned to the subsequent power plan refinements. γ limits scheduling uncertainty for the load, as it reduces the likelihood of significant changes to the power plan in subsequent calculations. Figure 3.12 shows the results of varying values of γ. As can be seen, a high value of γ inhibits the subsequent reduction of power allocated to time slots 24 to 30 to exploit the high renewable share in time slots 34 to 40. Increasing the value of γ is similar to the constraints of a high value of PH_{fixed}, although—in contrast to PH_{fixed}—it presents a form of *soft constraint*. This means that if there are significantly favorable conditions, subsequent changes to the power plan still occur. The numerical results are given in Table 3.5.

The weighting parameter δ represents a discount factor, reducing the cost of subsequent power plan changes in temporally more distant time slots. It incentivizes the power planning process to create optimizations in time slots further into the future, increasing scheduling security especially in directly upcoming time slots. Similarly to γ, high values of δ (in combination with a non-zero value of γ) result in a soft constraint version of a low PH_{fixed} value: while subsequent changes to upcoming time slots are assigned a high cost, the discount factor δ reduces cost in time slots

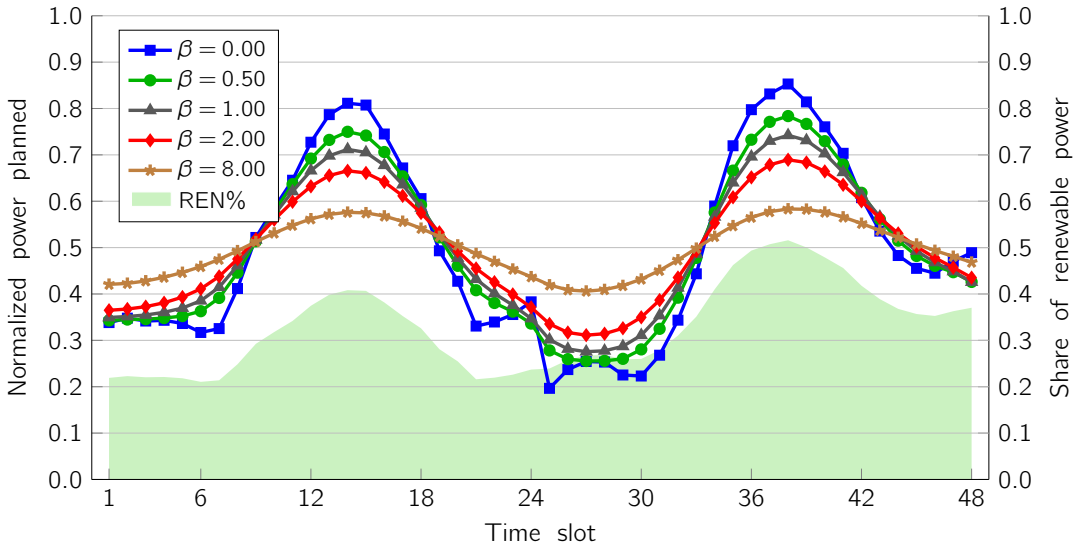


Figure 3.10: Resulting power plans for different values of β.

further in the future. Figure 3.13 shows power plans calculated for the example scenario using different values for δ. The numerical results are given in Table 3.6.

3.8 Summary and Future Work

In this chapter, the idea of power planning for flexible loads in a power grid was presented. It provides a GA with a possibility to create custom tailored power plans while not overburdening the respective electricity consumers. To this end, an objective function captures both optimization goals of a GA and corresponding costs if the respective consumer were to implement a power plan. The described approach considers the planning of active power demand of a single consumer. Although this is useful in a variety of situations, the methodology used to derive power plans may be applied to further related problems.

3.8.1 Reactive Power Planning

A challenge inherent to AC environments is the potential occurrence of phase shifting between current and voltage. The magnitude of this phase shift is quantified in a dimensionless value called power factor (PF), defined as given in Formula (3.6).

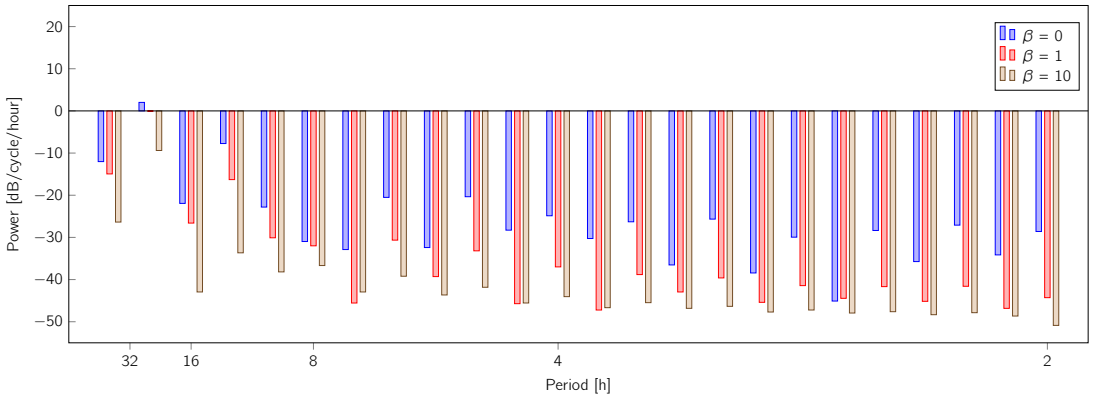


Figure 3.11: Power spectral density of power plans obtained with different settings of β.

$$\mathrm{PF} = \cos\varphi, \tag{3.6}$$

with φ representing the phase angle between current and voltage. Decreasing values of PF indicate a rising phase angle magnitude and have undesirable effects on the public power grid, e.g., increasing generation costs and higher stress on power lines. Depending on the components used for power provisioning, changes to reactive power demand on consumer side may propagate to the public grid infrastructure. Depending on the type of reactive load—inductive or capacitive—there are different effects on the AC power system.

1. Inductive loads cause a lagging power factor, meaning the relative angle φ between current and voltage increases. Inductive loads decrease local voltage.
2. Capacitive loads cause a leading power factor, meaning the relative angle φ between current and voltage decreases. Capacitive loads increase local voltage.

However, a low PF also poses a challenge for local emergency backup systems supplied by one or several dedicated generators. Rasmussen et al. [152] showed that a leading PF of more than 20 % of a generator's rated power output may lead to unstable operation.

Thus, in addition to planning active power demand, an additional reactive power planning allows for influencing distribution grid voltage to a certain extent. Reactive power planning must take into account both effects from grid and local device side, as listed in the following.

α	β	γ	δ	Maximum deviation from normalized baseline	Achieved REN%
	0.00			0.3529	35.40
	0.50			0.2837	34.97
	1.00			0.2423	34.70
	1.50			0.2124	34.49
	2.00			0.1895	34.33
	2.50			0.1745	34.19
	3.00			0.1620	34.08
	3.50			0.1512	33.99
	4.00			0.1417	33.91
0.20	4.50	10.00	0.20	0.1333	33.84
	5.00			0.1258	33.77
	5.50			0.1191	33.72
	6.00			0.1130	33.67
	6.50			0.1076	33.63
	7.00			0.1026	33.59
	7.50			0.0981	33.55
	8.00			0.0939	33.52
	8.50			0.0901	33.49
	9.00			0.0866	33.46
	9.50			0.0833	33.44
	10.00			0.0803	33.41

Table 3.4: Detailed results for different values of β.

- Grid line capacity: Reducing power factor magnitude will (in absence of compensation of some form) cause a higher reactive power amount to be transmitted. This in turn results in higher currents which have to be accounted for in line capacity.
- Local device peculiarities: As shown in Section 4.6.2, even power factor corrected switching power supplies typically exhibit a load-dependent power factor and conversion efficiency. Therefore, in case the active power demand may be split among several units powered by switching power supplies, different power factors may be realized depending on the distribution of power demand among those units.

Enabling a power plan to consider both active and reactive power targets would require a load to share more information than discussed up to this point. This includes, e.g., the minimum and maximum reactive power contribution achievable by a load at different operating points. Additionally, provision of reactive power may increase component stress, as higher currents need to be accounted for. Accordingly, the planning of active and reactive power generally cannot be performed independently in order to not exceed component current ratings.

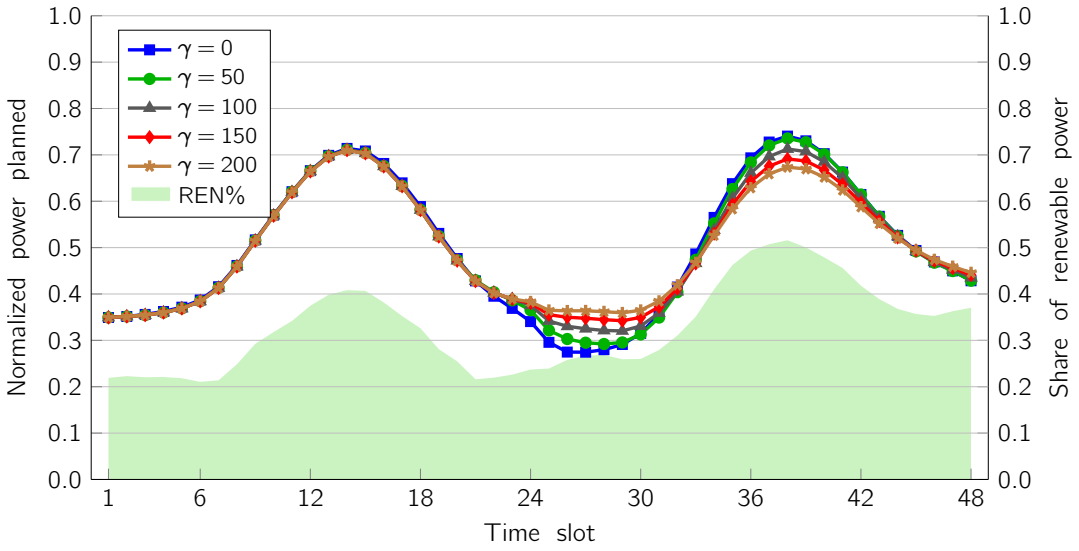


Figure 3.12: Resulting power plans for different values of γ.

3.8.2 Use of Local Renewable Generation

In certain cases, local generation of renewable power may provide a significant fraction of a consumer's power demand. In this case, it is desirable to shift power demand to times of high local generation as this is both economically and ecologically reasonable. This problem was focused by the author in a previous publication [127]. Summarized, a substantial local generation may be reflected in the objective function by subtracting the local generation from demand prior to optimization. This formulation is given in Expression (3.7).

$$\max_{P^{new}_{PP,i}} \sum_{i=1}^{PH} \left[(P^{new}_{PP,i} - \text{LocalRen}_{PP,i}) \text{RenPercent}_{PP,i} - \text{Cost}_{PP,i} \right], \tag{3.7}$$

with $\text{LocalRen}_{PP,i}$ denoting the average locally generated renewable power in time slot i. This way, a complete utilization of local generation can be ensured, avoiding transport losses and reducing energy cost. It is however noteworthy that in power-demanding industrial environments, local renewable generation is often insignificant compared to demand. As an example, a DC with a surface power density of $1200\,\mathrm{W\,m^{-2}}$ is assumed. This represents a common DC according to Rasmussen [150]. In comparison, the average surface power density of rooftop solar installations in Germany is around $5\,\mathrm{W\,m^{-2}}$ according to Wilson[20].

[20] `https://www.energycentral.com/c/ec/future-energy-why-power-density-matters`, accessed 27.11.2018.

α	β	γ	δ	Maximum deviation from normalized baseline	Achieved REN%
		0.00		0.2405	34.70
		10.00		0.2423	34.70
		20.00		0.2431	34.69
		30.00		0.2418	34.68
		40.00		0.2391	34.66
		50.00		0.2354	34.64
		60.00		0.2312	34.62
		70.00		0.2267	34.59
		80.00		0.2220	34.57
0.20	1.00	90.00	0.20	0.2173	34.54
		100.00		0.2127	34.52
		110.00		0.2105	34.49
		120.00		0.2104	34.47
		130.00		0.2103	34.45
		140.00		0.2102	34.43
		150.00		0.2101	34.41
		160.00		0.2100	34.39
		170.00		0.2100	34.37
		180.00		0.2099	34.36
		190.00		0.2099	34.34
		200.00		0.2098	34.33

Table 3.5: Detailed results for different values of γ.

3.8.3 Adaptation Recovery

Some adaptation measures may require a recovery phase before being used again. This includes, e.g., HVAC or UPS equipment. After a UPS was discharged to temporally shift power demand from the public grid, internal batteries are depleted and cannot provide power until a minimum state of charge is restored. To this end, a recovery phase has to be planned which accounts for additional power demand needed to restore a system to the state before an adaptation was performed. Different systems may however exhibit different recovery behaviors. Certain devices, like UPSs, will enter recovery immediately after stopping their adaptation phase. Others, like cooling equipment, may be able to delay recovery until a suitable time slot is found. To model recovery phases in general, they have to be decoupled from the initial adaptation phase. Modeling work on power flexibilities that reflect recovery phases was published by, e.g., Berl et al. [22]. The adaptation of HVAC and UPS equipment is covered in more detail in Section 4.8.

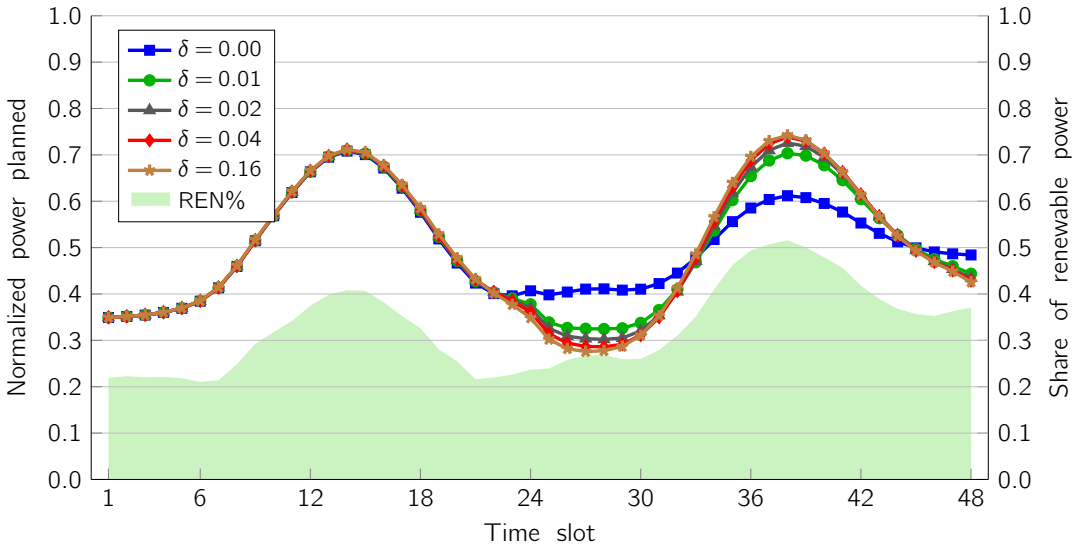


Figure 3.13: Resulting power plans for different values of δ.

3.8.4 Spatial Power Shifting

Apart from the temporal power shifting possibilities discussed in this chapter, an addition to the PP might be a spatial power shifting. If load can be migrated to other parts of the electricity grid, new options of grid optimization become available. A relocation of load within the same transmission grid may be employed to minimize renewable energy transport loss, to mitigate violations of allowable voltage ranges or to reduce load on overloaded parts of the grid infrastructure. To support transmission grid-wide characteristics, primarily AC frequency, the load would need to be both shifted over high distances and be significant in amount. Independent of the distance, only very specific consumers are capable of exploiting this kind of flexibility: the services they provide need to be quickly movable across high distances. Together with the previously discussed temporal power shifting, a combined spatio-temporal power planning could be realized. This idea is not further investigated throughout this thesis. It would however be a valuable direction of future research.

α	β	γ	δ	Maximum deviation from normalized baseline	Achieved REN%
			0.00	0.2082	34.04
			0.02	0.2098	34.49
			0.02	0.2255	34.59
			0.03	0.2344	34.64
			0.04	0.2385	34.66
			0.05	0.2407	34.68
			0.06	0.2418	34.68
			0.07	0.2424	34.69
			0.08	0.2427	34.69
0.20	1.00	10.00	0.09	0.2429	34.69
			0.10	0.2429	34.69
			0.11	0.2429	34.69
			0.12	0.2429	34.70
			0.13	0.2428	34.70
			0.14	0.2428	34.70
			0.15	0.2427	34.70
			0.16	0.2426	34.70
			0.17	0.2425	34.70
			0.18	0.2424	34.70
			0.19	0.2424	34.70
			0.20	0.2423	34.70

Table 3.6: Detailed results for different values of δ.

If you want to find the secrets of the universe, think in terms of energy, frequency and vibration.

Nikola Tesla

4

Hardware Power Demand

This chapter contains material previously published in [108], [111], and [15].

Chapter 3 describes how a *guidance* value, i.e., a target power demand, may be derived for a load inside a considered power grid. Applying the methods and techniques described, the hypothetical load is presented with a plan on how much power to draw from the grid at which point in time. The following chapter is concerned with estimating an operating point of the load that exhibits the desired power demand. This is a prerequisite towards enacting *control* on the load such as to arrive at the power demand specified in the power plan.

Section 4.1 specifies the running example used throughout this work in more detail. The abstract electrical load introduced in Chapter 3 is defined more concretely as a medium-sized DC. Section 4.2 examines the foundations of ICT power demand on hardware level. The observations start with losses on transistor level. The power demand equation for complementary metal-oxide-semiconductor (CMOS) circuits is used to derive suitable points of action to effectively influence ICT power demand. In the following, Section 4.3 presents a power demand model based on common ICT hardware components. The model is based on a subdivision of an ICT system into

independent components. The components are subsequently populated with power demand estimation formulas, presented in 4.4. Section 4.5 is dedicated to an evaluation of the power demand models introduced in Section 4.3 and Section 4.4. Component power demand is modeled primarily using analytical models employing parameters which are commonly available in monitoring systems. The evaluation shows relative errors between 6 % and 9 %. In the following, Section 4.6 introduces the concept of energy layer boundaries (ELBs), which enable an application of the previously introduced power models for entire DCs. Section 4.7 explores further possible ways of raising flexibilities available in hardware devices of DCs. Specifically, the benefits of employing water cooling on available thermal inertia is focused. The following Section 4.8 briefly introduces a multi-modal flexibility model. It divides a DC environment into different domains, each with a respective amount of inertia and flexibility. The total flexibility may be derived by considering the interdependency between the described domains. Based on these observations, ICT power demand is identified as the most reasonable point of action to exploit DC flexibilities. Finally, Section 4.9 concludes this chapter and gives an overview on possible future work.

4.1 Running Example, Second Iteration

The running example introduced in Section 3.3 is now specified in more detail. It is assumed that the load is a DC hosting a large number of ICT equipment. The specifications of the assumed DC are chosen according to the findings of Hintemann et al. [68] regarding medium-sized DCs, i.e., mostly hosting rack servers with a total ICT power demand of 250 kW. Participation of DCs within D/R systems is highly attractive, as they usually have automation frameworks already in place, making automated processing of D/R requests possible. Additionally, DCs consume high amounts of electrical power and therefore are well suited to create a significant impact when adapting their power demand. As discussed throughout the upcoming chapters of this work, an adaptation of power demand in DCs may be achieved by exploiting flexibilities present in both soft- and hardware domain. However, for a DC to be able to participate in a D/R program, it needs to establish a correlation between the utilization of its hosted equipment and the resulting power demand. The following sections of this chapter are concerned with devising a method to provide this correlation.

4.2 ICT Power Demand

Hardware power demand in the context of this work refers to two major components: power directly required for providing computing capability and power to maintain a stable operation over a wide range of environmental and utilization conditions. Power directly associated to computing is characterized by being required for switching activity of electronic circuits. It is noteworthy

that this power is not inherently required by the process of computation but is instead a result of non-ideal properties of electronic components (e.g., parasitic capacitances or resistance of transistor channels), which is discussed in more detail in Section 4.2.1. On a physical level, this translates to power demand of solid-state circuitry. This electric power is not lost but converted into heat (as per the law of energy conservation). As components of ICT devices have a limited tolerance regarding their maximum operating temperature, additional power is required (e.g., for cooling equipment) to keep environmental conditions within acceptable ranges, which is elaborated on further in Section 4.7. In a first step, the power demand required by ICT devices providing computational capabilities is investigated closer. This is structured in a bottom-up approach: starting from transistors—the very fundamental building blocks of ICs—an increasing abstraction is performed in the subsequent sections.

4.2.1 Electronic Circuit Power Demand

Current electronic computing devices are made from ICs, i.e., very densely packed solid-state electronic components. The most common of these components and actual enablers of high performance state-of-the-art computation are field-effect transistors (FETs), which act as electronically actuated switches. Figure 4.1 shows a schematic view of a FET. In the following, the high level operation of a FET is explained briefly[21]. A FET has a minimum of three electric connections to its environment: *source*, *drain*, and *gate*. Source and drain contacts are insulated against each other by default. Applying a positive voltage V_{GS} between the gate and source contacts leads to charge accumulating on the gate electrode, which generates an electric field. Once V_{GS} exceeds a certain threshold voltage $V_{GS(th)}$, this field becomes strong enough for a conducting channel to form in the silicon substrate, ultimately resulting in a conducting connection of drain and source. Discharging the gate electrode reverses the effect.

In contrast to current-driven bipolar transistors, FETs are voltage-driven. As almost no current is required for FET switching, a single FET requires very little power during operation. However, two major factors contribute to a non-negligible overall power demand of ICs: on the one hand, the sheer number of transistors contained in a single IC may reach several billion (cf. examples listed in Table 4.1). On the other hand, the ever-increasing need for higher computational performance causes manufacturers to aim for higher transistor switching speeds, at the time of writing up to about 5 GHz. However, these high switching speeds dramatically increase power demand, details are given in Section 4.2.1.2.

[21] In the example, an n-channel enrichment type FET is assumed.
[22] `https://www.realworldtech.com/arms-race/`, accessed 28.01.2019.
[23] `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.85.988&rep=rep1&type=pdf`, accessed 28.01.2019.
[24] `https://www.intel.com/pressroom/kits/itanium2/`, accessed 28.01.2019.

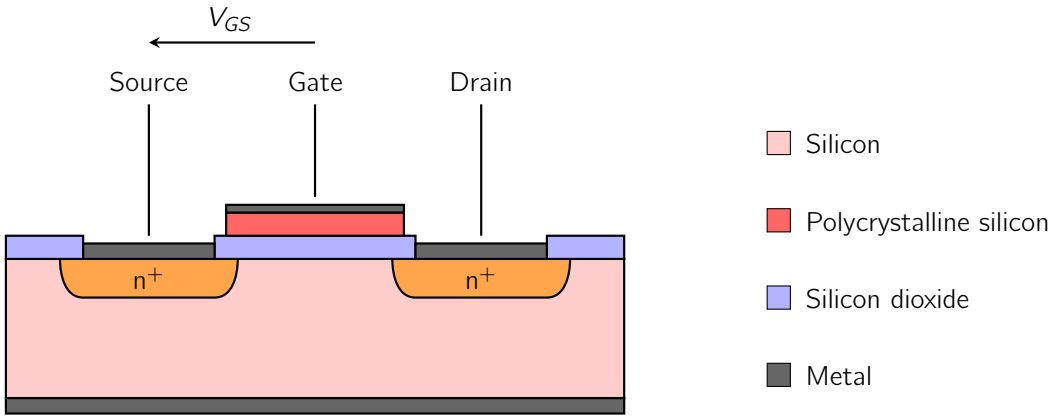


Figure 4.1: Schematic view of a FET.

Processor	Year released	Structure size [nm]	Transistor count
ARM 1[22]	1985	3000	25 000
DEC WRL MultiTitan[23]	1988	1500	180 000
Intel 2-core Itanium 2[24]	2006	90	1 700 000 000
Intel 8-core Xeon Nehalem-EX[25]	2010	45	2 300 000 000
Qualcomm Centriq 2400[26]	2017	10	18 000 000 000

Table 4.1: Technical specifications of different processors.

The actual power dissipation originates on the physical level, namely in the components and circuits of an IC. The actual amount of power dissipated during IC operation depends on several factors, some of them can be manipulated during operation while others are fixed at design time. Power dissipation in a circuit consists of two main components: dynamic and static power. The total power of a circuit is the sum of these two as given in Formula (4.1).

$$P_{total} = P_{dynamic} + P_{static}. \tag{4.1}$$

Static and dynamic power are investigated more closely in the following.

[25] https://www.intel.com/pressroom/archive/releases/2009/20090526comp.htm, accessed 28.01.2019.
[26] https://www.qualcomm.com/news/releases/2017/11/08/qualcomm-datacenter-technologies-announces-commercial-shipment-qualcomm, accessed 28.01.2019.

4.2.1.1 Static Power

When a chip is not performing active work, power is still being consumed, namely static power. Static power originates in leakage currents and in circuits with an intentional path from supply voltage (V_{DD}) to voltage ground (GND). Formula (4.2) shows the detailed components of static power as outlined by Weste et al. [179].

$$P_{static} = P_{subthreshold} + P_{gate} + P_{junction} + P_{contention}, \tag{4.2}$$

with $P_{subthreshold}$ denoting the subthreshold leakage, P_{gate} the gate leakage, $P_{junction}$ the junction leakage, and $P_{contention}$ the contention current. However, as the name implies this power is static as long as an IC is powered on. To eliminate static power dissipation of an IC, it has to be powered down entirely. To this end, a deep sleep technique named *power gating* may be applied, cutting power to idle IC components. Figure 4.2 shows a possible implementation of power gating for hardware components.

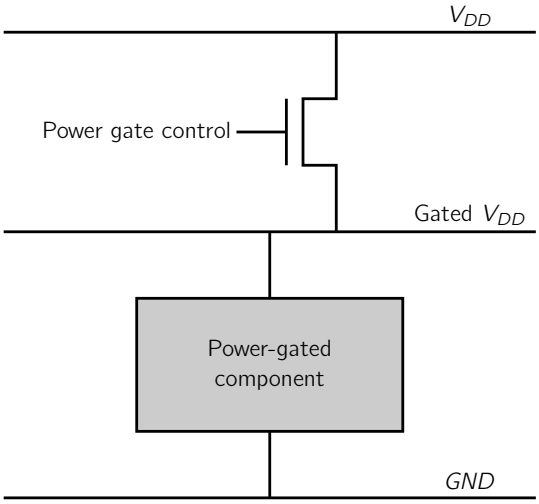


Figure 4.2: Schematic view of a power gated component.

If either V_{DD} or GND are disconnected, no current can flow through the power-gated components, eliminating most static power dissipation[27]. It is noteworthy that the amount of deactivated components may vary, depending on the sleep state. At the time of writing, available sleep states are ordered by the amount of power savings in *C-states*, from C0 (active) to C7 (maximum savings). However, to re-enable power-gated components, a non-negligible wakeup delay needs to be ac-

[27] As the power gate transistor also suffers from static power dissipation, even power gated components do dissipate trace amounts of power.

counted for. As no influence on static power can be exerted during component operation, it is not considered further in the course of this work.

4.2.1.2 Dynamic Power

Since the amount of static power in the total power of a circuit is almost constant during operation, dynamic power is the only remaining point of action to actively influence power demand. It depends on the activities that occur in an IC. There are two major components that contribute to dynamic power: switching power and short-circuit power, as indicated in Formula (4.3).

$$P_{dynamic} = P_{switching} + P_{short\text{-}circuit}, \tag{4.3}$$

Switching power originates in IC internal transistor switching operations, i.e., changing a transistor state from conducting to insulating or vice versa. As explained in Section 4.2.1, this requires charging or discharging a transistor's gate. Each of these gates has a certain capacitance, which causes it to accumulate charge once a voltage is applied to it. As stated in Formula (4.4),

$$Q_{gate} = C_{gate} \cdot V_{DD}, \tag{4.4}$$

with Q_{gate} denoting the gate charge, C_{gate} the gate capacitance, and V_{DD} the supply voltage. At the same time Formula (4.5),

$$I = \frac{dQ}{dt}, \tag{4.5}$$

holds. Combined with Formula (4.6),

$$P = VI, \tag{4.6}$$

this leads to Formula (4.7):

$$P = V\frac{dQ}{dt}. \tag{4.7}$$

Combining Formulas (4.4) and (4.7) yields Formula (4.8):

$$P = V\frac{d(CV)}{dt}. \tag{4.8}$$

Averaging Q over t and assuming a constant supply voltage allows for simplifying to Formula (4.9).

$$P = V^2 \cdot \frac{C}{t}. \tag{4.9}$$

Further, it has to be considered that not all transistor gates are switched at each IC clock cycle. To reflect this, the activity factor α_{IC} is introduced, reflecting the average probability of a transistor gate switching its state. Also introducing IC clock frequency $f = \frac{1}{t}$ into Formula (4.9) leads to the final equation for calculating switching power, shown in Formula (4.10):

$$P_{switching} = V_{DD}^2 C \alpha_{IC}^{avg} f. \tag{4.10}$$

While a transistor switches, short-circuit power is caused temporarily by current flowing from V_{DD} to *GND*. According to Nose et al. [132], short circuit power can be estimated to amount to 2 % to 10 % of the switching power in current applications. Other sources even state that "short-circuit current has become almost negligible" [179]. Thus, dynamic power is henceforth approximated by considering switching power only, which is calculated as shown in Approximation (4.11).

$$P_{dynamic} \approx P_{switching} = V_{DD}^2 C \alpha_{IC}^{avg} f. \tag{4.11}$$

The total switching capacitance C of an IC is determined by its physical layout, specifically wires and transistor properties [167]. It is a property inherent to an IC's design, manufacturing process and manufacturing tolerances of components.

The supply voltage V_{DD} has a quadratic effect on dynamic power. Power consumption can be considerably reduced by applying a lower supply voltage. Therefore, from an energy saving perspective, the best option is choosing the minimal value of V_{DD} that is required for the chosen clock frequency. By scaling down the voltage together with the frequency, a cubic reduction of power demand can be achieved. This is further discussed in Section 4.2.1.3. It is noteworthy that lowering V_{DD} even inside stability limits of a circuit may have a negative effect on resilience against

fluctuations in supply voltage or voltage droop during sudden demand spikes. However, the discussion on reliability implications is not followed further at this point. For more information, the reader is referred to, e.g., Zhu et al. [193].

The clock frequency f and dynamic power are linearly related. Therefore, the frequency should not be higher than required. Reducing frequency can have a great impact on power requirements, since using a lower frequency means it is possible to decrease supply voltage as well. The relation between voltage and clock speed is covered in more detail in Section 4.2.1.3.

In summary, the power demand of modern ICs can be broken down to Approximation (4.12):

$$P \approx P_{static} + \alpha_{IC} C V_{DD}^2 f. \tag{4.12}$$

Considering only those parts of Approximation (4.12) which can be influenced during runtime, the remaining parameters are α_{IC}, V_{DD}, and f. In the following, the details on how to influence these parameters are discussed.

Switching Activity α_{IC}:
Parameter α_{IC} denotes the switching activity, i.e., the probability of transistors switching their state from insulating to conducting. To reflect an entire IC, the average activity factor α_{IC}^{avg} has to be taken into account. Powell et al. state in [143] that "[...] activity factor is fundamentally a function of utilization". How to influence device utilization is elaborated on in more detail in Chapter 5.

Supply Voltage V_{DD}:
The supply voltage of ICs may be influenced during runtime, provided it supports dynamic voltage scaling (DVS). To this end, most high-power ICs employ multi-phase buck converters coupled with a respective controller (e.g., International Rectifier IR35201[28]) that allows to change its output voltage on demand. The supply voltage is required to enable the charging of FET gate capacitances. Therefore, its setpoint depends on multiple factors, e.g., transistor size [192] and operating frequency (cf. Section 4.2.1.3).

Frequency f:
An IC's operating frequency f denotes its internal switching speed. It is generated by an electronic

[28] `http://www.irf.com/product-info/datasheets/data/pb-ir35201.pdf`, accessed 20.02.2018

oscillator and is required to synchronize IC subcomponent operation. According to Approximation (4.12), f exhibits a linear influence on circuit power demand. In CPU-bound workloads, it is not feasible to only reduce f: workload execution slows down proportionally with frequency decrease. At times of partial load, it may however increase power efficiency, especially if combined with reducing voltage as outlined in Section 4.2.1.3.

4.2.1.3 Voltage/Frequency Dependency

Dynamic voltage and frequency scaling (DVFS) is a hardware-based technology invokable through software (e.g., the operating system (OS)). DVFS simultaneously adjusts the operating voltage V_{DD} and frequency f of an IC. These parameters cannot be chosen independently: a stable IC operation requires a certain minimum voltage for each operating frequency. Voltage influences the time needed to switch a transistor from conducting to non-conducting state (and vice versa). To change the conductivity of a transistor, a small amount of charge has to be transported to or from its gate. The speed of this charge transport depends on the voltage applied to the gate. Formula (4.13) describes how the voltage across the gate (whose electrical behavior can in this context be described similarly to a capacitor) evolves over time.

$$V_C(t) = V_{DD}(1 - e^{-\frac{t}{\tau}}), \tag{4.13}$$

with V_C denoting the voltage across the capacitor, t the elapsed time since the application of the supply voltage, V_{DD} the supply voltage, and τ the time constant of the resistor-capacitor (RC) charging circuit.

While τ represents characteristics intrinsic to a specific component and is not adjustable during runtime, both V_{DD} and f can be adjusted. For a complex IC like a CPU to work correctly, it has to be in a well-defined state at the end of each clock cycle. This means all transient operations stemming from switching activities in the transistors (e.g., switching from a logical LOW to HIGH voltage level) have to be propagated to the next stage of the CPU. Put more briefly, the signal at each input to transistors active during the next clock cycle has to be in a well-defined state: either below the upper voltage bound recognized as LOW state or above the lower voltage bound recognized as HIGH state. To change the input to a transistor, the output of the previous transistor has to be changed. This output change requires a change of the transistors gate-to-source voltage V_{GS}. This in turn is achieved by moving electric charge from or to the gate. Let Q_{gate} denote the electric charge of a transistor gate. To charge or discharge Q_{gate}, a current $I = \frac{dQ}{dt}$ is required, in this case $I \geq \frac{Q_{gate}}{t_{cycle}}$. As $I = \frac{V}{R}$, a change in voltage is increasing charging speed. The author is aware that these relations are oversimplified (e.g., with regard to velocity saturation and mobility

degradation, as outlined in [179]), however accurate enough to base further investigations on. Although transistor gate charge is a fixed property, knowledge on the relation of IC operating frequency and required voltage enables the design of a circuit that is able to operate at different voltage/frequency operating points. The lower the IC frequency, the less voltage has to be applied to its transistors in order to achieve a stable operation. In case an IC is operated under partial utilization conditions, exploiting the possibility of reducing voltage and frequency simultaneously leads to significant savings in dynamic power. Figure 4.3 shows a qualitative comparison of an IC operated at two different voltage/frequency settings.

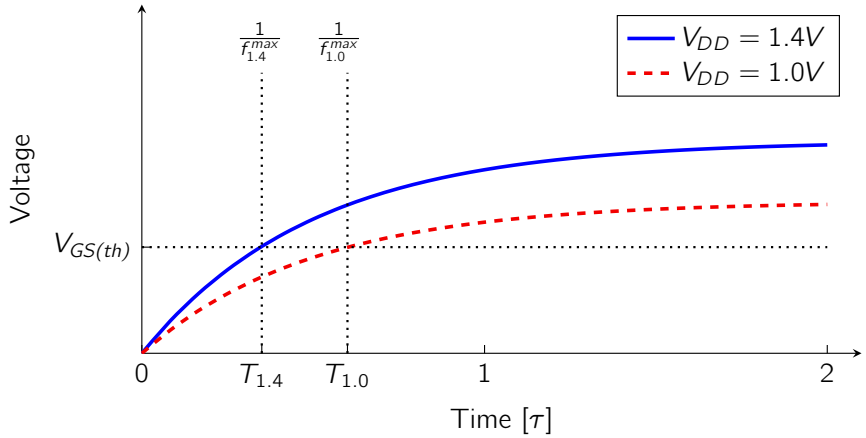


Figure 4.3: Influence of V_{DD} on transistor switching speed.

The time instances $T_{1.4}$ and $T_{1.0}$ in Figure 4.3 illustrate when the threshold voltage $V_{GS(th)}$ is reached. As can be seen, the upper bound of frequency f at which the IC can be operated is $\frac{1}{f_{1.4}^{max}}$ and $\frac{1}{f_{1.0}^{max}}$, respectively. A lower supply voltage is only possible while operating at a lower frequency setting. For the high voltage setting of $V_{DD} = 1.4\,\mathrm{V}$ the threshold voltage $V_{GS(th)}$ is reached after approximately 0.35 τ ($T_{1.4}$), while the low voltage setting of $V_{DD} = 1.0\,\mathrm{V}$ takes approximately 0.6 τ ($T_{1.0}$). Of course, this calculation is only an approximation, as it omits many details of current transistor technology, signal propagation and driver circuits, however, basic relations between voltage and switching speed are visible. DVFS has been widely adopted by manufacturers of ICT equipment and is in use in many major server and consumer CPUs (e.g., Intel® SpeedStep® or AMD® PowerNow!™).

Example 4.1 (Intel SpeedStep operating points)
An Intel white paper[29] on the Intel Pentium M processor gives the stable DVFS operating points of the CPU, as depicted in Table 4.2.

[29] `http://download.intel.com/design/network/papers/30117401.pdf`, accessed 27.02.2019.

Frequency [MHz]	Voltage [V]
1600	1.484
1400	1.420
1200	1.276
1000	1.164
800	1.036
600	0.956

Table 4.2: Example Intel® DVFS table.

As can be seen, there is a wide spectrum of possible voltage/frequency setpoints. This enables an adaptation offering enough computing capabilities while at the same time achieving high savings in power demand.

While DVFS is used to reduce the power demand of ICs during periods of partial utilization, increasing clock speed beyond nominal frequency in certain load situations is a more recent development. As discussed in this section, different clock speeds require a respective minimum voltage to be operated in a stable way. In multi-core CPUs, thermal design power (TDP) (i.e., the power a cooling system designed for this CPU has to be able to handle) is based on power dissipation exhibited by all cores. As the primary constraint for maximum CPU voltage is temperature limitations, this allows for a higher frequency/voltage combination to be used in case only a subset of cores is being utilized.

The findings of this section enable the selection of a point of action to influence runtime power demand. As a voltage reduction by means of DVFS can only be realized after a decrease of operating frequency, a purely voltage-centric power management would result in a possible selection of unstable operating points (i.e., low voltage in combination with high frequency). Adapting only frequency or both frequency and voltage (i.e., selecting different stable operating points of DVFS), would result in capping available CPU performance indiscriminately for all hosted software. However, switching activity α_{IC} is influenced by the cumulative CPU utilization of all software hosted on the same substrate hardware. By selectively reducing α_{IC} attributed to specific software services, the required overall CPU frequency to satisfy software demands may be decreased. This in turn allows for a reduction in supply voltage V_{DD}. A further investigation of this cascading effect is given in Chapter 5.

After identifying all major components contributing to power demand on circuit level, in a next step hardware power demand models on a higher abstraction level need to be derived. This is required due to several reasons: on the one hand, power demand models used *in the field* need to rely on information available to all equipment operators, not only IC designers (e.g., the total

circuit capacitance of a CPU is usually not publicly disclosed). On the other hand, in the context of this work a large number of ICT devices need power demand estimation. To not overburden devices, power demand models should require only limited computational effort. Additionally, a power demand model enables the evaluation of hypothetical resource utilization scenarios.

4.3 Power Demand Model

Depending on the desired accuracy, power demand models may range from simple machine level on/off models to detailed analysis on transistor level. As previously stated, ICT power demand originates in electrical circuits being operated and therefore only indirectly from software load. To translate from software load to power demand, it is necessary to establish power demand models which correlate relative information on hardware utilization to absolute information on power demand. In the following, a power demand model is derived that—by employing a bottom-up approach—estimates the total power demand exhibited by a site hosting computing equipment. Due to space limitations and to not stray too far from the main topic of this work, the most basic structures considered in the following are individual hardware components. An even more fundamental explanation regarding power demand of electronic components was given in Section 4.2.1, however, the following section is focused on creating a model that can operate on monitoring data readily available during runtime without additionally installed measurement equipment.

4.3.1 Server Structure Model

As the author showed in previous work [17], it is feasible to calculate ICT device power demand by regarding its individual components independently. Although certain application specific ICT devices do exist, the overall architecture of many ICT systems, in terms of power demand, can be described in a generic ontology. A well-suited method of description is employing unified modelling language (UML), which enables a description of architectural dependencies between server components. At the same time, it is capable of specifying subclasses to describe certain components more elaborately.

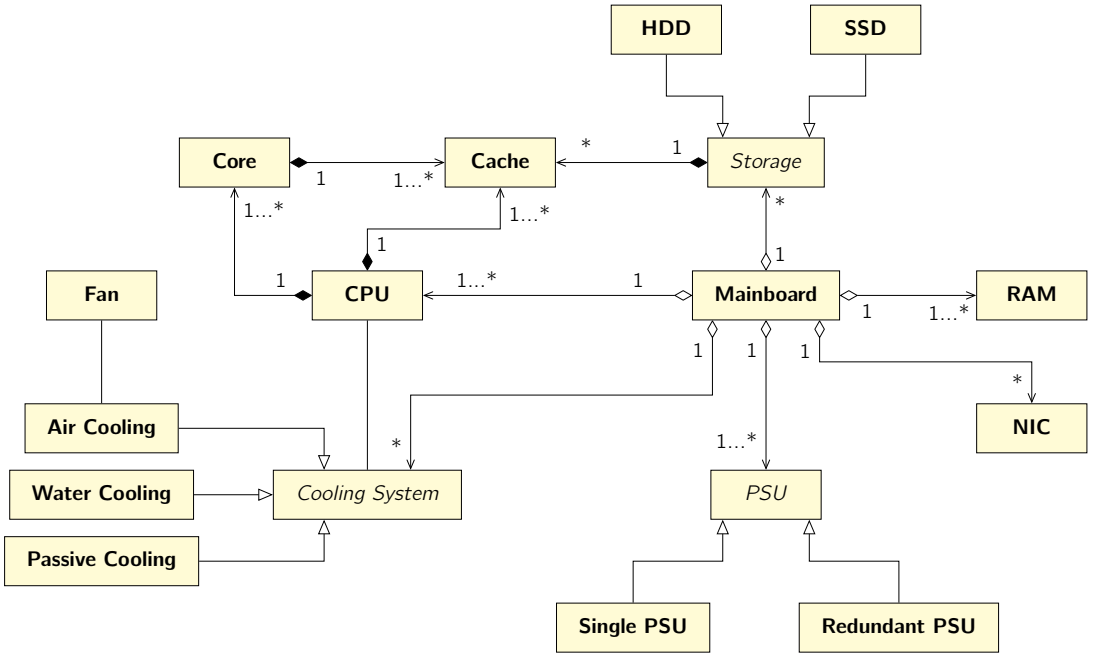


Figure 4.4: Decomposition of a standard rack server into a UML diagram of its components.

Figure 4.4 shows a UML diagram of a single server's hardware components in a well human readable form. It depicts certain components of an ICT device's hardware configuration as given in Definition 4.1.

Defintion 4.1: Hardware Configuration

A computer system's hardware configuration denotes all hardware components (and their hardcoded programming) it is comprised of. It is runtime-invariant, i.e., it remains constant during the system's runtime.

While information as in Definition 4.1 gives a complete description of the physical components of an ICT system, it is insufficient to completely capture hardware properties. In addition to their time-invariant properties, hardware components traverse several states during runtime. Notable examples of runtime-variant hardware properties are, e.g., CPU registers or RAM contents. This information is henceforth referred to as hardware state, as given in Definition 4.2.

Defintion 4.2: Hardware State

A computer system's hardware state denotes the internal state of all elements in its hardware configuration, i.e., all potentially runtime-variant information of hardware components.

From the definition of hardware configuration and state directly follows Corollary 4.1.

Corollary 4.1

The combination of hardware configuration and hardware state describes a computer system's hardware completely, as these convey all time-invariant and time-variant hardware information.

To enable the algorithmic calculation of power demand from the UML model presented in Figure 4.4, it is necessary to define a method that estimates the power demand of each individual component. Thus, each component contributing to overall power demand is required to implement the interface depicted in Figure 4.5.

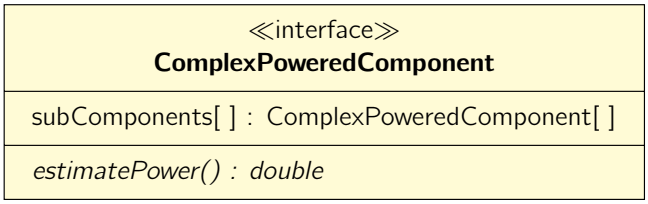


Figure 4.5: Interface implemented for power demand calculation.

The interface «ComplexPoweredComponent» ensures that each component may contain an arbitrary number of subcomponents that in turn contribute to overall power demand and that this contribution to power demand may be estimated for each component. Depending on the respective equation used in the device-specific *estimatePower()* method, a component may need to provide information regarding its *hardware configuration* (cf. Definition 4.1) and current *hardware state* (cf. Definition 4.2).

To calculate power demand for a specific instance of a server from a UML diagram, an *instantiation* of the model needs to be performed. As power demand calculation follows a hierarchical pattern, it is necessary to establish an order among the component instances. This order is chosen in accordance to electrical power flow inside the device, i.e., the top component(s) of the order

is of the type *PSU*[30]. By recursively descending through its *subComponents[ ]*, all components contributing to overall power demand are visited. The UML ontology is realized into an ordered instance by using the following rules:

1. The top element **T** is selected such that it reflects the component whose power demand should be estimated.
2. Child nodes are added to **T** for each directed association of **T** in the UML diagram. Abstract classes are not added, only their respective realization. In case the UML diagram allows for higher multiplicity, several nodes of the same type may be added as child nodes.
3. Step 2 is applied recursively to all child nodes created in the previous step. This process is repeated until none of the child nodes have any further associations in the UML graph.
4. Power demand is calculated by calling *estimatePower()* of **T**.

To calculate power demand from this graph, each vertex has to be assigned a function *estimatePower()*, as indicated in Step 4. This is ensured by implementing the interface «ComplexPoweredComponent» depicted in Figure 4.5. Depending on the type of node, this function may take several forms, typically:

- For any node with at least one child node, *estimatePower()* recursively calls *estimatePower()* for all its child nodes and sums their power demands, optionally adding power demand of its own, or
- for leaf nodes, *estimatePower()* returns a constant or a value depending on the hardware configuration/state of the currently observed leaf node only. It is noteworthy that the hardware state (cf. Definition 4.2) may vary in time (e.g., the utilization of a CPU core).

This way, all nodes are being visited in the course of a power estimation process. The required features for an implementation of power estimation in the described form (interfaces, abstract classes, etc.) are readily available in almost all common object-oriented programming languages, e.g., JAVA, C++ or Python.

Another factor required for a runtime power estimation as proposed in this work is knowledge on the hardware configuration and hardware state. For both of these requirements, tools for automation

[30] In fact, not all power is necessarily routed through the mainboard, some components may instead be directly connected to the PSU. To keep the model generic and to not create unnecessary complexity by splitting the power flow, power flow is assumed to entirely pass through the mainboard.

are available and in common use in DC environments. On the one hand, hardware configuration (i.e., time-invariant hardware information) may be captured in a configuration management database (CMDB). A CMDB stores information on hardware and software assets and their relations, enabling an automated extraction of device components and connections. Examples include OneCMDB[31] or Device42[32]. On the other hand, capturing the entire hardware state (as given in Definition 4.2) has to be considered impractical for several reasons, among others the amount of monitoring traffic and security challenges. However, as shown in Section 4.5, hardware power demand may be estimated relying on limited, high-level monitoring information only. These data are often already monitored or may be gathered by monitoring utilities like Cacti[33] or Nagios[34].

4.4 Power Demand Estimation Models

In the following section, the power demand estimation equations of individual ICT components are populated. In previous co-authored work [15, 17], analytical models for the respective ICT components are derived and evaluated in a full system power model. Most of the models presented in the following are based on these publications. However, the validity of some of these models has been limited by recent advancements in ICT hardware manufacturing and management. As an example, an updated CPU power model and the hardware changes that require model adaptation are presented, including a discussion on how the original analytical model error may develop if applied to more recent hardware. The applicability of some models to newer hardware is therefore limited, the general methodology may however be re-applied. This includes the possibility of populating the power estimation equations with models derived with methods not considered in the following.

4.4.1 CPU

A CPU's power demand depends, among other parameters, on its utilization, voltage, and frequency (cf. Section 4.2.1.2). These information need to be provided in an up-to-date manner to the power estimation model. Fan et al. [51] showed through measurements that the power demand of an ICT system with a single-core processor not employing power saving techniques may be estimated with high accuracy by assuming a linear increase of its power demand with its utilization, as shown in Equation (4.14).

$$P = P_{idle} + (P_{max} - P_{idle})u, \tag{4.14}$$

[31] https://sourceforge.net/projects/onecmdb/, accessed 05.02.2019.
[32] https://www.device42.com/features/cmdb/, accessed 05.02.2019.
[33] https://www.cacti.net/, accessed 08.02.2019.
[34] https://www.nagios.org/, accessed 08.02.2019.

with P_{max} denoting the maximum and P_{idle} the idle power demand of the investigated system, and u the utilization of the processor. This is in line with the findings of Section 4.2.1: as no advanced power saving techniques (e.g., DVFS) are applied, CPU power demand depends on its utilization and accordingly, on its switching activity α_{IC}. Figure 4.6 depicts the model described in Equation (4.14).

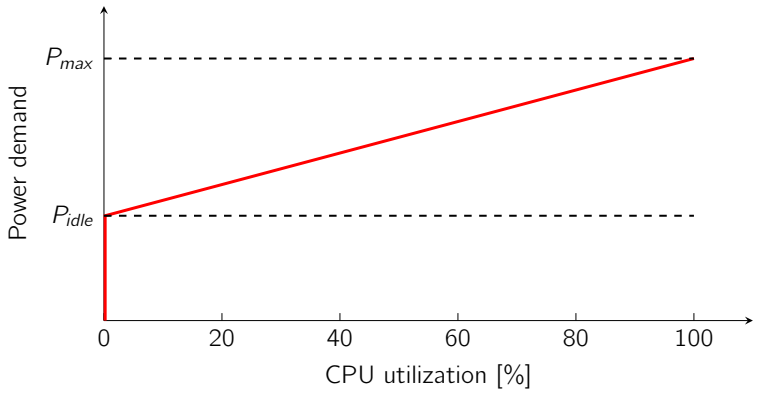


Figure 4.6: Visualization of the single core system power demand model.

In early multi-core processors (i.e., employing independent cores with linked clock speeds and shared voltage supply), individual cores of multi-core processors may be modeled similarly to a single-core system: the power consumption increases linearly with utilization [15]. Therefore, similarly to Equation (4.14), the power demand of individual cores may be estimated as given in Equation (4.15).

$$P_{core} = P_{core_{max}} u_{core}, \tag{4.15}$$

with $P_{core_{max}}$ denoting the maximum power consumption of the CPU core, and u_{core} the utilization of the core. Based on Equation (4.15), the power consumption of a processor consisting of at least one core is given by Equation (4.16):

$$P_{CPU} = P_{idle} + \sum_{i=1}^{N} P_{core_i}, \tag{4.16}$$

with N denoting the total number of cores for a given processor, P_{core_i} the power demand of core i, and P_{idle} the idle power demand of the processor. For early multi-core systems, previous work contributed to by the author shows this modeling provides high accuracy with relative errors of

less than 10 % for the full system [15]. This model is also used in the evaluation presented in Section 4.5. However, the author is aware that newer generation multi-core systems include more sophisticated architectures and per core power management techniques. Accounting for these more recent developments requires a refined power modeling. Several reasons for why this linear model is not applicable to newer multicore architectures are discussed in the following.

At the time of writing state-of-the-art multi-core architectures include components that are shared among several cores (e.g., higher level caches and memory controllers). As an example, the architecture of an AMD EPYC CPU—as presented by Brookwood [32]—is discussed. In this architecture, CPU cores cannot be assumed to work independently: a single CPU may offer up to 32 cores, grouped in CPU complexes (CCXs). Each CCX is comprised of four CPU cores, individual L1 & L2 caches and an 8 MB shared L3 cache. Each two CCXs together create a die and share a common peripheral infrastructure ("uncore" elements), including memory controllers and interconnects to other dies. This is depicted in Figure 4.7.

Figure 4.7: Architecture of an AMD CCX.

Shared elements—both on CCX and die level—influence CPU power demand, as they are active if at least one core of the CCX (or die, respectively) are active. Similar architectural concepts with an increasing number of shared components are used by other major CPU manufacturers (e.g., Intel® Xeon® CPUs). Thus, in a more general model, CPUs can be modeled by subdividing them into three distinct zones, as pointed out by, e.g., Basmadjian et al. [19].

1. **Core-exclusive components:**
   These components are exclusively used by individual CPU cores. Other CPU components have no access to these resources. Examples include core ALUs, L1 and L2-caches.

2. **Core-shared components:**
   Core-shared components are utilized by several (but not necessarily all) server-wide available CPU cores. These components are commonly higher level caches.

3. **Die-shared components:**
   These components are shared among an entire die. It is noteworthy that a CPU may consist

of several dies, interconnected by a communication fabric. Examples of die-shared components include memory controllers and voltage regulator modules.

Per core power management techniques include, e.g., individual voltages and boost clock speeds similar to DVFS (cf. Section 4.2.1.3). Table 4.3 shows an example of advanced per-core power management: the maximum core frequency of an Intel® Six-Core i7-4960X[35] processor is set dynamically, based on the total number of active cores of the CPU.

Active cores	6	5	4	3	2	1
Maximum frequency [GHz]	3.7	3.7	3.8	3.9	3.9	4.0

Table 4.3: Example Intel® turbo boost frequency table.

Modern CPU P-state control gains performance by exploiting TDP limits under partial load conditions: in case only a subset of CPU cores are active, voltage and clock speed of the remaining active cores can be increased. Referring back to Approximation (4.11), limits on power draw may be realized despite an increase of core voltage and clock speed by almost zero switched capacitance in inactive cores. To estimate CPU power demand while employing advanced power management techniques, it is required to model each core individually to account for their respective runtime state.

Recent developments in CPU power management infrastructure may lead to an increasing obsolescence of analytical models for real-time CPUs power estimation. Referring to the whitepaper of Brookwood [32], recent generation CPUs allow operation in power- or performance-deterministic way, i.e., user configurable power bounds on CPU level. Also, CPU-internal sensors continuously monitor operational parameters (e.g., power draw, voltage, temperatures) and may replace estimations with actual monitoring data. However, it is noteworthy that power demand models are still useful for the simulation of hypothetical load scenarios.

For a more recent CPU, a new analytical model is devised that considers bipartite processor domains. As the CPU does only contain one core cluster, the *core-shared* domain can be omitted. The model is fitted to Expression (4.17)

$$P_{CPU} \sim \mathrm{c} + \sum_{n=1}^{N} P_{core}(n) + (\omega^{max}\mathrm{a}) - \sum_{n=1}^{N} idle(n)\mathrm{e}, \tag{4.17}$$

[35] `https://www.intel.com/content/www/us/en/support/articles/000006652/processors.html`, accessed 02.01.2018

with c, a, and e denoting fitting parameters, $N \in \mathbb{N}$ the number of CPU cores, $P_{core}(n)$ the power demand of the n^{th} CPU core as defined in Equation (4.18), ω^{max} the maximum utilization among all CPU cores of one core cluster, and $idle(n)$ the indicator function of idle CPU cores as defined in Equation (4.19).

$$P_{core}(n) = \omega_n(\mathtt{b}f_n^2 + \mathtt{d}f_n), \tag{4.18}$$

with ω_n denoting the utilization of the n^{th} CPU core, b and d fitting parameters, and f_n the frequency of the n^{th} CPU core.

$$idle(n) = \begin{cases} 1, & \text{core } n \text{ is idle} \\ 0, & \text{core } n \text{ is active} \end{cases} . \tag{4.19}$$

The components of the fitting formula are now discussed in more detail. In Expression (4.17), c represents the static idle power demand of a CPU, which is estimated to approximately 45 W by Hofmann et al. [70]. As no idle power measurement is available in the measurement setup, this value is assumed to be sufficiently accurate. The power demand of each CPU core $P_{core}(n)$ is comprised of $\mathtt{b}f_n^2$, reflecting power demand components with quadratic dependency on voltage, while $\mathtt{d}f_n$ reflects those with a linear relationship, respectively. Both these components are multiplied by the respective core's utilization, similar to Approximation (4.11). $\omega^{max}\mathtt{a}$ is used to account for the shared CPU components. ω^{max} reflects the maximum core utilization. It is noteworthy that for the specific case of all CPU cores being idle, this term also amounts to 0. $\sum_{n=1}^{N} idle(n)\mathtt{e}$ is used to account for additional power savings in case a CPU core is idle. Idle CPU cores are powered down further, using deep sleep techniques like power gating (cf. Section 4.2.1.1) to further reduce power demand.

Table 4.4 lists the results of a least squares fitting of Expression (4.17) to the obtained measurement data.

a	b	c	d	e
3.639×10^{1}	-1.267×10^{-13}	5.609×10^{1}	1.304×10^{-6}	1.317

Table 4.4: Final fitting parameters obtained for Expression (4.17).

Figure 4.8 shows a heatmap depicting the mean relative power model errors for different numbers of active cores and per-core utilizations.

The overall mean relative error of the model is 3.07 %. As can be seen in Figure 4.8, the model exhibits higher accuracy in high utilization scenarios: in several cases, the relative model error is

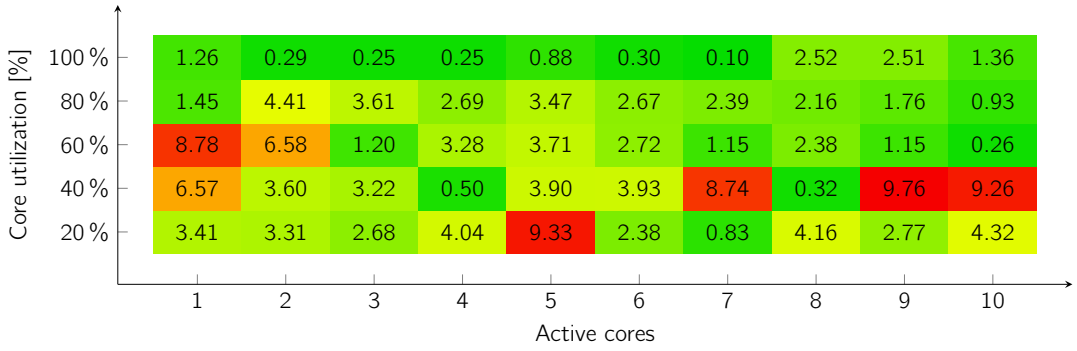
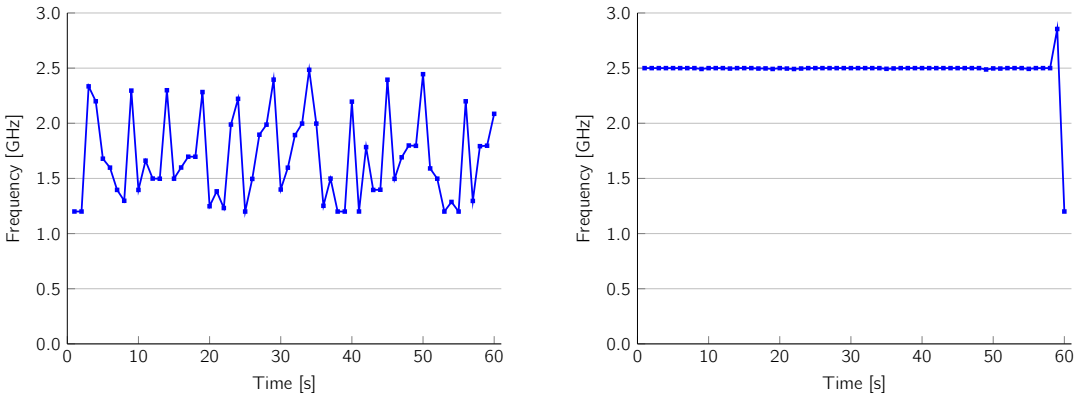


Figure 4.8: CPU power model relative error percentage.

below 1 %. At lower utilizations frequency is more volatile (i.e., DVFS is applied more frequently), as visualized in Figures 4.9a and 4.9b. This has a negative effect on estimation accuracy, as the average per-core frequency throughout each test is used for power estimation. By averaging over shorter periods of time, a further increase in model accuracy is expected.



(a) Core frequency at 40 % utilization.

(b) Core frequency at 100 % utilization.

Figure 4.9: Monitored core frequency at different utilizations.

In comparison, the linear CPU power model given in Equation (4.16) gives a mean relative error of 9.48 %, with maximum relative errors exceeding 24 %.

4.4.2 Random Access Memory

RAM is used to temporarily store data currently in processing. In contrast to permanent storage devices like HDDs and SSDs, data stored in RAM is lost if not continuously refreshed. At the time

of writing, the DDR4 standard (as defined in Joint Electron Device Engineering Council (JEDEC) JESD79-4B) of RAM modules is the most recent technology in both home and DC computing equipment. Besides different technology generations (e.g., DDR3, DDR4, etc.), RAM modules of 2nd and 3rd generation are additionally divided into two categories: buffered and unbuffered. Buffered modules are not connected to the memory controller directly but instead interfaced via an advanced memory buffer (AMB), which provides higher performance and error correction capabilities.

For the evaluation, an analytical model is used and presented in the following. It was partly generated by employing the Eureqa[36] software, which derives a model based on evolutionary algorithms and artificial intelligence techniques. The model is based on a memory size-centric estimation of idle power demand of DDR2 and DDR3 memory, as given in Equation (4.20).

$$P_{RAM}^{idle} = \sum_{i=1}^{N} s_i p \tag{4.20}$$

with N denoting the total number of installed memory modules and s_i the capacity of the i^{th} memory module. The value of p varies based on the specific type and vendor[37] of the memory module. For an unbuffered DDR2 SDRAM, Table 4.5 gives the values of p for different vendors, with f denoting the frequency of the memory module.

Vendor	p	Scaling factor for buffered module
Kingston[38]	$\frac{f}{1000}$	2.2
Samsung[39]	$0.95 \cdot \frac{f}{1000}$	4.26
Hynix[40]	$1.9 \cdot \frac{f}{1000}$	1.65

Table 4.5: p value for different DDR2 SDRAMs.

Buffered DDR2 SDRAMs require additional power for operating the AMB, which is reflected in an additional scaling factor in Table 4.5. For the given systems under investigation, the power consumption of an unbuffered DDR2 SDRAM is estimated by Equation (4.21).

[36] https://www.nutonian.com/products/eureqa/, accessed 20.03.2015
[37] In fact, the manufacturing technology affects the power demand, similarly to CPUs. As this information is hardly available during runtime, the manufacturer is used as proxy for this information in this analytical model.
[38] http://www.kingston.com/hyperx/products/khx_ddr2.asp, accessed 10.10.2010
[39] http://www.samsung.com/global/business/semiconductor/productList.do?fmly_id=696&xFmly_id=695, accessed 10.10.2010
[40] http://www.hynix.com/products/consumer/consumer_sub.jsp?menuNo=1&m=2&s=1&menu3=01&RK=03&RAM_NAME=DDR2%20SDRAM&SUB_RAM=1Gb, accessed 10.10.2010

$$P_{RAM} = P_{RAM}^{idle} + 7.347\gamma, \tag{4.21}$$

with $\gamma \in [0, 1]$ denoting a parameter defined later. For a buffered DDR2 SDRAM, the power demand estimation is given in Equation (4.22).

$$P_{RAM} = P_{RAM}^{idle} + 16.898\gamma, \tag{4.22}$$

with $\gamma \in [0, 1]$ denoting a parameter defined later. For an unbuffered DDR3 SDRAM, the value of p is calculated by employing Equation (4.23).

$$p = \frac{f}{1000} + 0.000026\sqrt{f}(f_c - f), \tag{4.23}$$

with $f_c = 1600\,\text{MHz}$ denoting the rated frequency, and f the actual operating frequency in MHz. With reference to Equation (4.20), Equation (4.24) gives the estimate power demand of unbuffered DDR3 SDRAM.

$$P_{RAM} = P_{RAM}^{idle} + 9.551\gamma, \tag{4.24}$$

with $\gamma \in [0, 1]$ denoting a parameter defined later. For a buffered DDR3 SDRAM, the idle power demand was found to amount to approximately double the idle power consumption of an unbuffered memory. For a buffered DDR3 SDRAM, the power consumption is given by Equation (4.25).

$$P_{RAM} = 2 \cdot P_{RAM}^{idle} + 13.959\gamma, \tag{4.25}$$

with $\gamma \in [0, 1]$ denoting a parameter defined in the following. In line with the power demand models derived in this section, the RAM power demand estimation relies only on runtime-obtainable monitoring information. In this analytical model, this information is captured in parameter γ, signifying an estimate on RAM activity. The value of γ is derived employing the following rationale:

1. In case all CPU cores are in idle state, all memory modules are assumed to be in idle state ($\gamma = 0$).

2. In case any CPU core is not in idle state, a probabilistic approach is used: γ is set to the total relative memory usage. This reflects the assumption that a higher utilization of available RAM capacity increases the probability that a memory access is performed.

4.4.3 Hard Disk Drive

HDDs are widely used for mass data storage, due to their superior price/storage space ratio compared to SSDs. To derive a HDD's power demand, several samples are measured using the test setup depicted in Figure 4.10.

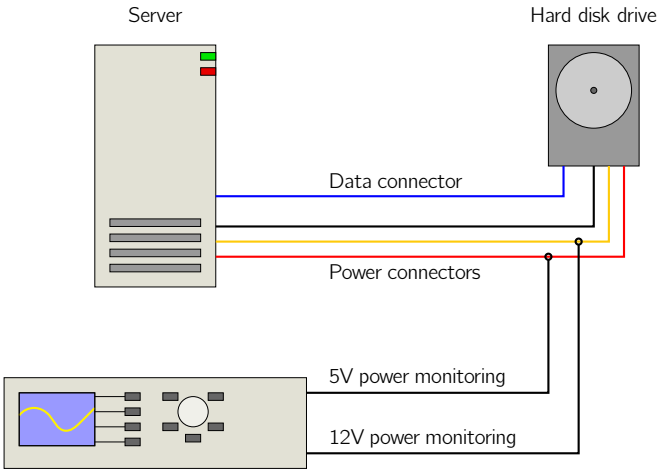


Figure 4.10: HDD power demand measurement setup.

Across a sample size of eight different HDDs, a common pattern is visible[41]. An HDD's power demand can be subdivided into four different phases: idle, active, sleep and startup. The HDD is in idle mode when no activity (read or write) is carried out. If additionally the HDD's platters are spun down, but its controllers remain active, it is in sleep state. It is in active mode in case read or write operations are performed on the disk. Finally, an HDD is in startup mode when its platters are in the process of spinning up to their operating speed. The four identified power demand phases are visible in the power measurement depicted in Figure 4.11. The time between 0 and 280 seconds represents the *idle* state, 280 to 570 seconds the *sleep* state, and starting at 570 seconds the *active* state. The peak in 12 V power demand coinciding with the beginning of the active state represents the drive startup.

It can be seen that in the sleep state, power demand on the 12 V supply drops to zero, as the mechanical parts of the drive are shut down. Restarting the mechanical components causes a

[41] In the following, a representative example in form of a Seagate Savvio 10K.3 SAS HDD is considered.

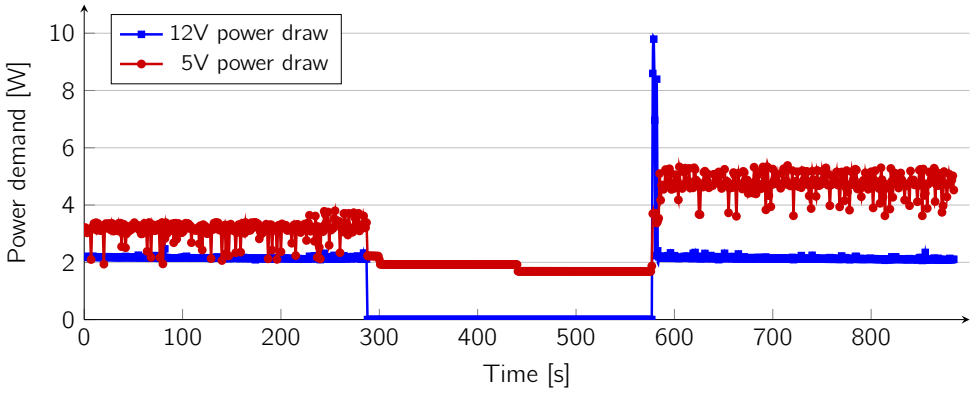


Figure 4.11: HDD power demand in different device states.

significant peak current to overcome platter inertia. However, as the startup power is required only during a very short time period, it only needs to be considered if there are frequent start/stop cycles. Using this assumption, HDD power demand may be modeled as given in Equation (4.26).

$$P_{HDD} = \alpha P_{HDD}^{sleep} + (1 - \alpha) P_{HDD}^{wake}, \tag{4.26}$$

with $\alpha \in [0, 1]$ denoting the probability of the HDD to be in sleep state and P_{HDD}^{wake} the average power demand during non-sleep states. P_{HDD}^{sleep} is provided by the manufacturer's data sheet. P_{HDD}^{wake} consists of the power demand in idle and active states and depends on the actual HDD utilization at runtime. Therefore, the power demand of the HDD is estimated as given in Equation (4.27).

$$P_{HDD} = \alpha P_{HDD}^{sleep} + \beta P_{HDD}^{idle} + (1 - (\alpha + \beta)) P_{HDD}^{active}, \tag{4.27}$$

with $\beta \in [0, 1]$ denoting the probability of the HDD to be in idle state. As the HDD states are mutually exclusive, the probabilities of the three states have to add up to 1. Again, P_{HDD}^{idle} and P_{HDD}^{active} can be obtained from the manufacturer's data sheet. Monitoring information on disk activity may be extracted using several utility programs, e.g., `iotop`[42] or `iostat`[43]. It is noteworthy that in practice, $\alpha = 0$ presents the common case of continuous HDD operation without invoking sleep states.

[42] `https://linux.die.net/man/1/iotop`, accessed 29.01.2019.
[43] `https://linux.die.net/man/1/iostat`, accessed 29.01.2019.

For the evaluation conducted in Section 4.5, the following assumptions are taken to derive the values of the required parameters:

- HDD startup power is considered in the evaluation, exhibited with probability γ.
- If there are neither read nor write operations performed, the HDD is assumed to be in idle state ($\alpha = 0$, $\beta = 1$).
- If either read and/or write operations are non-zero, a probabilistic approach is taken to derive state changes such that:
  1. If both HDD read and write rate is non-zero, Equation (4.28) is assumed to hold.

$$1 - (\alpha + \beta + \gamma) = \frac{\mathit{readRate} + \mathit{writeRate}}{\mathit{maxReadRate} + \mathit{maxWriteRate}} \tag{4.28}$$

     with *readRate* denoting the current HDD read rate, *writeRate* the current HDD write rate, *maxReadRate* the maximum HDD read rate, and *maxWriteRate* the maximum HDD write rate.

  2. If only HDD read rate is non-zero, Equation (4.29) is assumed to hold.

$$1 - (\alpha + \beta + \gamma) = \frac{\mathit{readRate}}{\mathit{maxReadRate}} \tag{4.29}$$

  3. If only HDD write rate is non-zero, Equation (4.30) is assumed to hold.

$$1 - (\alpha + \beta + \gamma) = \frac{\mathit{writeRate}}{\mathit{maxWriteRate}} \tag{4.30}$$

- Sleep power is assumed to amount to 20 % of idle power.
- Considering Equations (4.28) to (4.30), the following additional assumptions given in the set of Equations (4.31) to (4.32) regarding device state are taken.

$$\alpha + \beta = 0.9(\alpha + \beta + \gamma). \tag{4.31}$$

$$\gamma = 0.1(\alpha + \beta + \gamma). \tag{4.32}$$

  Finally, in order to derive the values of α and $\beta \in [0, 1]$ for the idle and startup state power demand, a probabilistic approach is adopted, given in the set of Equations (4.33) to (4.35).

$$\alpha = 0.1(\alpha+\beta) \wedge \beta = 0.9(\alpha+\beta), \qquad \text{for } 0.0 < (\alpha+\beta) \leq 0.3. \tag{4.33}$$

$$\alpha = 0.5(\alpha+\beta) \wedge \beta = 0.5(\alpha+\beta), \qquad \text{for } 0.3 < (\alpha+\beta) \leq 0.6. \tag{4.34}$$

$$\alpha = 0.9(\alpha+\beta) \wedge \beta = 0.1(\alpha+\beta), \qquad \text{for } 0.6 < (\alpha+\beta) \leq 1. \tag{4.35}$$

4.4.4 Solid-State Drive

SSDs are increasingly replacing traditional HDDs due to their superior performance, energy-efficiency and storage density. As SSDs do not rely on any moving parts, they lack a specific power state that relates to an HDD's startup power. Similarly, the idle power of SSDs is comparatively low. Research by Zhang et al. [189] shows that an increase in SSD performance can be achieved by internal resource parallelization at the expense of an increased power demand. However, this increased power demand is accompanied by an increase in performance and therefore approaches the goal of an energy-proportional ICT system much more closely than a traditional HDD. As an SSD exhibits the same basic device states as an HDD, a power demand model is assumed to be similar to that of an HDD.

4.4.5 Fan

Fans are indispensable parts in modern servers to force convective cooling of high power components. Depending on their frame size, fans can reach high rotational speeds of over 10 000 revolutions per minute (RPM). Using data available from data sheets, power measurements and runtime monitoring, a power demand model is derived. The model shown in Equation (4.36) is used as basis for a non-linear least squares fitting.

$$P_{\text{fan}} \sim (a(dw))(\pi w \frac{rpm}{60})^b) + c, \tag{4.36}$$

with a, b and c denoting fitting parameters, d the fan's depth, w the fan's width, and rpm the fan's RPM. The fitting model used builds on known physical properties, correlating the power demand to rotor blade area and velocity. Blade area is captured in the term $a(dw)$, a multiple of fan width times depth. $(\pi w \frac{rpm}{60})$ is the fan blades speed. Its exponent is fitted, as it is the primary influence for both kinetic energy and friction. The value of b is therefore expected to exceed two. Table 4.6 lists the measurement data collected from a fan used in the fitting process. In total, the data from five fans with different specifications is used.

Width [m]	Depth [m]	RPM [min^{-1}]	Power demand [W]
0.07	0.02	3000	0.714
		3500	0.916
		4000	1.188
		4500	1.509
		5000	1.885
		5500	2.319
		6000	2.850
		6500	3.462
		7000	4.198

Table 4.6: Measurement data collected from a fan used in the fitting process.

The calculation is performed using the statistical computing software R[44] version 3.5.1. The final fitting parameters converged to the values denoted in Table 4.7.

a	b	c
1.696 95	2.272 08	0.273 13

Table 4.7: Final fitting parameters obtained for Equation (4.36).

Using the values of Table 4.7 for the fitting parameters, the Pearson correlation between measured and modeled power demand is 0.9839. Although average relative error is improvable at 18.12 %, the average absolute error of 0.12 W is low compared to total server demand.

4.4.6 Mainboard

Mainboards are the central printed circuit board (PCB) in each server, interconnecting the different components constituting the complete system. The main tasks of a mainboard include voltage regulation, physical interconnection of components and provision of peripheral communication. The cumulative power drawn by a mainboard and its subcomponents is given in Equation (4.37).

$$P_{mainboard} = \sum_{h=1}^{L} P_{CPU_h} + P_{RAM} + \sum_{i=1}^{M} P_{NIC_i} + \sum_{j=1}^{N} P_{HDD_j} + \sum_{k=1}^{O} P_{fan_k} + c, \tag{4.37}$$

[44] `https://www.r-project.org/`, accessed 13.02.2017.

with L denoting the total number of installed CPUs, each with a power demand P_{CPU} as given in Section 4.4.1, P_{RAM} the power demand of the memory modules as given in Section 4.4.2, M the total number of installed NICs, each with a power demand P_{NIC}. The details on how to derive P_{NIC} are out of the scope of this work. Let further N denote the total number of installed HDDs, each with a power demand P_{HDD} as given in Section 4.4.3, O the total number of installed fans, each with a power demand of P_{fan} as given in Section 4.4.5, and c a constant representing the power demand of the mainboard itself. As it is impossible to measure mainboard power independently from other ICT components in the test setup, its power demand is assumed constant and estimated as the difference of its connected components to the measured power demand, resulting in a value of 40 W in case of tower and rackable servers, and a value of 55 W in case of blade servers.

4.4.7 Power Supply Unit

A power supply unit (PSU) is a required component to perform AC/DC power conversion[45]. Power dissipation due to non-ideal efficiency of PSUs (AC/DC and voltage conversion) has to be accounted for in the cumulative power draw of an ICT device. Details on how a load-dependent estimate on PSU efficiency can be derived is given in Section 4.6.2. Several additional considerations are required for the practical applicability in a DC environment. Servers, especially in mission-critical environments, are usually equipped with $n+1$ redundancy of their PSUs. Considering this, the following methodology is applied to derive the power demand of a PSU as exhibited towards the AC domain.

1. If a monitoring system provides the delivered DC power of each installed PSU individually, the dissipated power of each PSU can be derived by using Equation (4.38).

$$P_{PSU\ AC}^{estimated} = P_{PSU\ DC}^{monitored} \frac{1}{\eta(P_{mainboard}^{monitored})}, \tag{4.38}$$

   with $\eta(L)$ denoting the PSU's efficiency at a power draw L, and $P_{PSU\ DC}^{monitored}$ the measured DC power output of the regarded PSU.

2. In case only the total DC server power is available, an equal split of power demand among the installed PSUs is assumed. Therefore, the power draw of each PSU is calculated using Equation (4.39).

$$P_{PSU\ AC}^{estimated} = \frac{P_{PSU\ DC}^{monitored}}{N} \frac{1}{\eta(\frac{P_{mainboard}^{monitored}}{N})}, \tag{4.39}$$

   with N denoting the number of installed PSUs.

[45] There are in fact scenarios with a central AC/DC conversion in DCs, feeding direct current to servers. These are however rare and disregarded in the following.

3. If there is no power monitoring available, power output of the PSU has to be estimated by summing the server component power demands. The multiplicity of components is determined by the server specific structure model according to Section 4.3.1. The power demand of components is estimated by populating the structure model with equations as exemplified in Sections 4.4.1 to 4.4.5. Given a power estimation for server power is available, Equation (4.40) may be used to approximate per PSU power dissipation.

$$P_{PSU\ AC}^{estimated} = \frac{P_{mainboard}^{estimated}}{N} \frac{1}{\eta(\frac{P_{mainboard}^{estimated}}{N})}, \tag{4.40}$$

with $P_{mainboard}^{estimated}$ denoting the estimated power output of the PSU.

Further details on PSUs—specifically regarding their load-dependent properties—are covered in Section 4.6.2.

4.4.8 Server Power

Server power is estimated depending on the server type. The power demand of a rack or tower server is estimated as the sum of its components with the additional overhead of PSU AC/DC conversion. This is captured in the sum of the power demands assigned to the installed PSUs, as given in Equation (4.41).

$$P_{server\ AC}^{estimated} = \sum_{i=1}^{N} P_{PSU\ AC_i}^{estimated}, \tag{4.41}$$

with $P_{server\ AC}^{estimated}$ denoting the estimated power demand of a server as exhibited towards the AC domain, N the total number of installed PSUs, and $P_{PSU\ AC_i}^{estimated}$ the estimated power demand of the i^{th} PSU as given in Section 4.4.7.

In case the power demand of a blade server is estimated, it has to be noted that reference measurements regarding power demand are taken from the blade enclosure's internal power sensors. As the common enclosure includes fans and PSUs, which are not covered by the power sensors. Thus, blade server power demand is calculated inside the direct current domain, as given in Equation 4.42.

$$P_{blade\ DC}^{estimated} = \sum_{i=1}^{N} P_{mainboard_i}^{estimated}, \tag{4.42}$$

with $P^{estimated}_{blade\ DC}$ denoting the estimated power demand of a blade as exhibited towards the blade enclosure, N the total number of installed blades, and $P^{estimated}_{mainboard_i}$ the estimated power demand of the i^{th} mainboard as given in Section 4.4.6.

4.5 Model Evaluation

In previous shared work [15], the author evaluated a component-wise power demand model similar to the one presented in Section 4.4. The evaluation is performed on three different hardware substrates: one tower (with two different CPU configurations) and one blade server. The detailed configurations are given in Table 4.8.

	Server 1	Server 2	Server 3
CPU	Intel Xeon L5420 (2.50 GHz)	Intel Xeon L5420 (2.00 GHz)	2 × Intel Xeon E5520 (2.27 GHz)
RAM	2 × 4 GB Hynix DDR2-667 PC2-5300F FB-DIMM ECC		6 × 4 GB HP 2Rx4 PC3-10600R-9
HDD	Maxtor 120 GB DiamondMax Plus 9		2 × HP 300GB SAS 10K
PSU	AcBel API3FS43		6 × HP Common-Slot PSU
Fans	1 × 70mm fan & 4 × 120mm fans		Blade chassis internal fans

Table 4.8: Detailed evaluation server configuration.

The synthetic workloads are generated using the `lookbusy`[46] utility, which induces load on CPU, HDD and RAM simultaneously. Four different load scenarios are investigated, from 20 % to 90 % of CPU usage. The details of each load scenario for Server 1 and 2 are given in Table 4.9.

	Scenario 1	Scenario 2	Scenario 3	Scenario 4
CPU utilization	20 %	40 %	60 %	90 %
RAM use	1 GB	1 GB	1 GB	1 GB
HDD operations	52	42	32	20
70 mm fan speed	1183	1183	1278	1249
120 mm fan speeds	708	708	796	783

Table 4.9: Detailed evaluation settings for Server 1 and 2.

Table 4.10 gives the detailed settings for each load scenario of Server 3.

4.5.1 Obtained Results

Figures 4.12, 4.13 and 4.14 depict the measured and estimated power demands for the different servers in the previously described scenarios. Additionally, a linear interpolation line of the relative

[46] `http://www.devin.com/lookbusy/`, accessed 18.02.2019

	Scenario 1	Scenario 2	Scenario 3	Scenario 4
CPU utilization	20 %	40 %	60 %	90 %
RAM use	100 MB	250 MB	500 MB	750 MB
HDD operations	80	80	80	80

Table 4.10: Detailed evaluation settings for Server 3.

error is given. The results are averaged over a time of 50 minutes for each test. Due to the long measurement time and stable power demand, confidence intervals are omitted in the following figures. The observations are carried out based on different RAM, HDD as well as CPU utilizations (cf. Table 4.8).

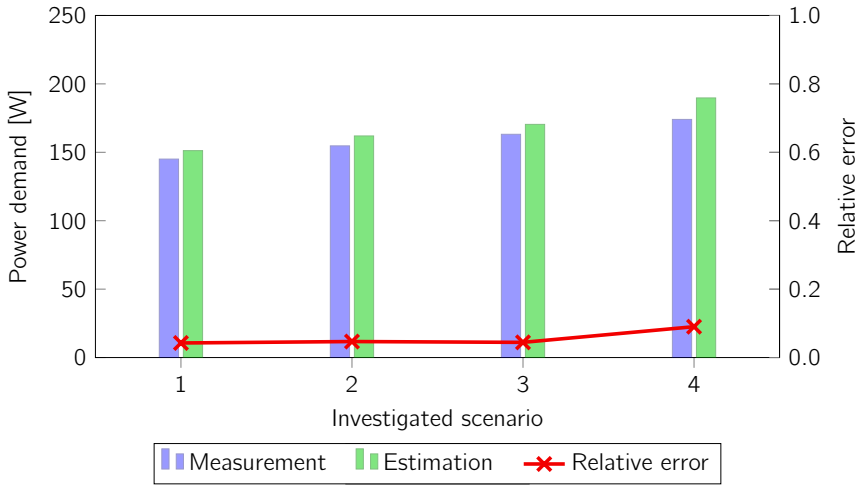


Figure 4.12: Evaluation results of Server 1.

Figure 4.12 visualizes the results obtained for Server 1. It shows that the difference between the computed and measured power consumptions is less than 8 W for Scenarios 1, 2, and 3 (a CPU utilization between 20 % and 60 %. The relative error is below 5 % for the aforementioned scenarios. Scenario 4 shows a worse performance: the absolute error increases to approximately 15 W, a relative error of 9 %.

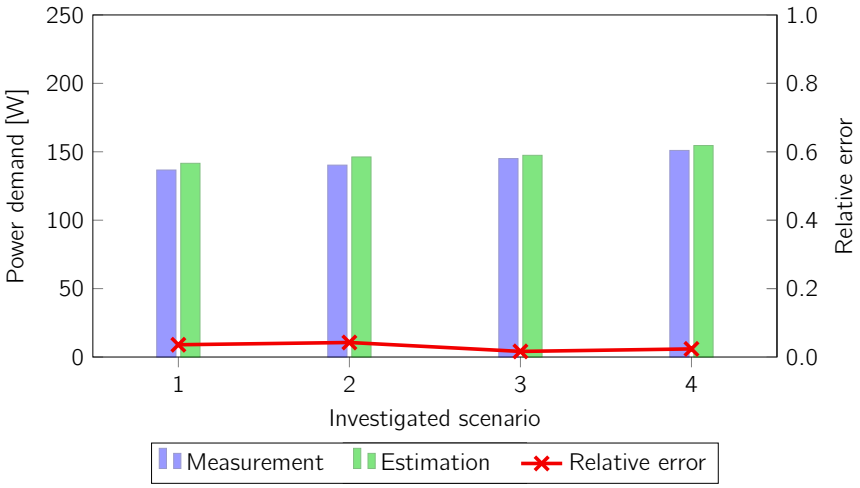


Figure 4.13: Evaluation results of Server 2.

A mostly similar result can be observed for Server 2. Figure 4.12 shows that the absolute error of the power demand estimation is less than 6 W for all scenarios (a relative error of less than 5 %). There are no significant outliers in terms of power demand estimation with this server.

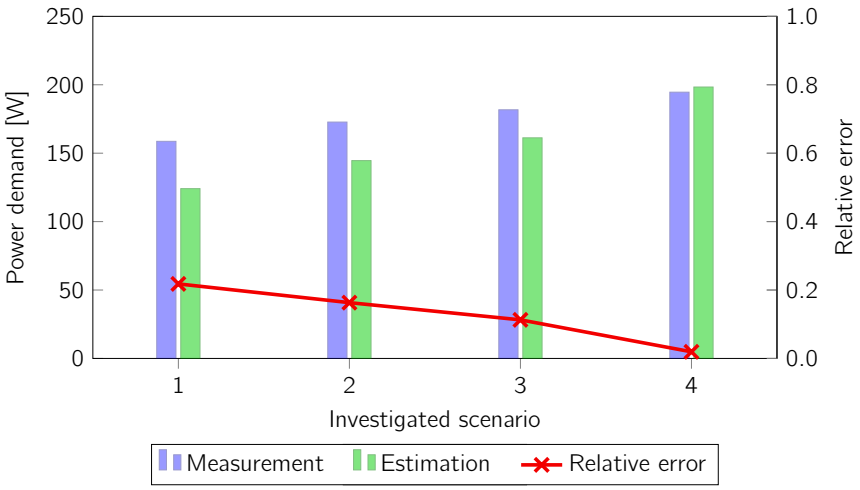


Figure 4.14: Evaluation results of Server 3.

Figure 4.14 depicts the measured and estimated power demand for Server 3 at different CPU and RAM utilizations. For the blade server, the relative error decreases with an increased resource utilization. While the relative error is just below 9 % for CPU utilizations between 10 % and 50 %,

it decreases to around 2 % for a CPU utilization of 90 %. It is noteworthy that mainboard power demand differs significantly between rack and blade servers.

The evaluation of the component-based power demand model shows that an estimation of power demand based on runtime monitoring information is feasible. However, a DC's power demand–as exhibited towards the public power grid—encompasses more than what is provided to servers: ICT components need to be supplied with direct current and be kept inside a well-defined temperature range. Both of these tasks require additional infrastructure powered by electricity. The following Section 4.6 discusses how this overhead may be formalized and estimated.

4.6 Energy Layer Boundaries

To calculate the overall effect of the proposed system on the exhibited power demand, summing up the power demand of all ICT components is insufficient for two major reasons: on the one hand, ICT components are powered by direct current, however electricity from the power grid is delivered as AC. Therefore, a PSU has to provide the necessary AC/DC conversion. Non-ideal component properties make this conversion a lossy process which affects the cumulated power draw of all components from the AC domain. In a next step, in addition to power delivered to ICT devices of a DC, the power required for additional non-ICT components like HVAC needs to be considered. The effects of both support infrastructure and AC/DC conversion is similar as it both affects the entire power demand once a PSU (or DC, respectively) is passed by the power delivered. These components thus represent what is being referred to in the course of this work as *ELB*. Definition 4.3 defines the term ELB as understood in the course of this work.

Defintion 4.3: Energy Layer Boundary

An energy layer boundary is a threshold which, upon passing, exhibits a multiplicative effect on the cumulated power demand of its sublayer(s).

In other words, the power demand inside an ELB is summable, those separated by an ELB are not. Figure 4.15 shows ELBs in a broader DC ecosystem.

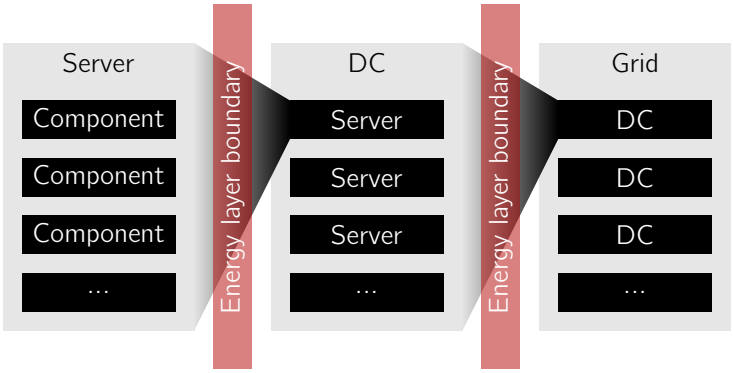


Figure 4.15: Energy layer boundaries in a data center ecosystem.

Figure 4.16 shows the ELBs relevant in the course of this work. In the following, each of these ELBs is discussed in more detail.

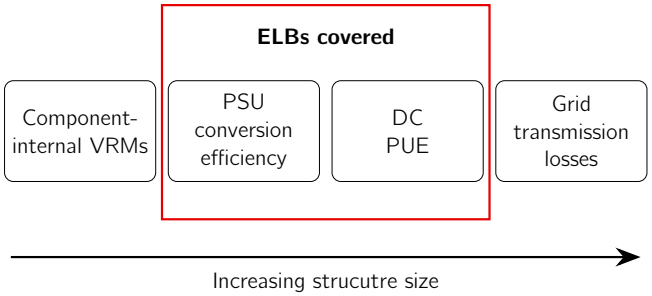


Figure 4.16: Extended view on energy layer boundaries in a wider DC environment.

4.6.1 Layer Boundaries not Covered in This Work

Two energy layer boundaries are not covered in this work: on the one hand, energy layer boundaries inside individual ICT components. These are, e.g., voltage regulator module (VRM) which are responsible for reducing supply voltage to a level appropriate for the respective circuit and its operating point. VRMs for high-power components like CPUs or GPUs usually employ a multi-phase buck converter layout to distribute load and heat. The inefficiencies associated with VRMs are not covered separately but instead included in component specific power demand. The rationale of this decision is that VRM operation is beyond user control and usually hardcoded in hardware components.

On the other hand, the components used in a power distribution grid have non-ideal properties, causing them to incur transport losses during power provision to consumers. To evaluate the impact a change in DC power demand has on generation side, these transport losses have to be accounted for. It is noteworthy that these losses are highly dependent on the amount of current delivered, the power factor of attached loads and transformer efficiency, to name just a few. In the following, each of the covered layers is elaborated on in more detail.

4.6.2 Power Supply Energy Layer Boundary

PSU efficiency was briefly covered in Section 4.4.7, however a more in-depth analysis is provided in the following. Moving from the demand of single components to total server power, the load-dependent efficiency of PSUs has to be taken into account. Efficiency in this case refers to the electrical conversion efficiency between PSU input (AC) and output (single or multiple voltage level direct current). To derive a model applicable to arbitrary PSU load values, a large dataset on PSU efficiency was analyzed. The data was obtained from the 80 PLUS website[47], an institution awarding efficiency certificates to PSUs depending on their efficiency. The data is publicly available and at the time of writing contains measurement values of more than 6000 different PSUs. Although only the values of certified PSUs are listed, the data may still be considered representative, as "80 PLUS market penetration is approaching 85 %" [54]. Figure 4.17 shows box plots for the load levels evaluated in the data set (10 %, 20 %, 50 % and 100 %). The horizontal line inside the boxes shows the median value, the lower and upper edge of the boxes denote the 25 % and 75 % quantile (the difference of these values shall be referred to as the interquantile range (IQR)). The whiskers at the top and bottom of the boxes cover all data points that are inside the range of $1.5 \times$ IQR from the top and bottom of the boxes. Any data points outside this range are considered outliers and marked separately. A quadratic regression is performed to derive an approximation over all load values (dashed red line in Figure 4.17).

The resulting second order regression polynomial is shown in Equation (4.43).

$$\eta_{\mathrm{PSU}}(u) = -18.90u^2 + 22.69u + 81.95, \tag{4.43}$$

with $\eta_{\mathrm{PSU}}(u)$ denoting the efficiency of the PSU in percent, and u the PSU utilization in percent. Equation (4.43) is used when extending the system boundary from the DC domain inside a server to the AC domain, as it is necessary to account for limited PSU conversion efficiency.

[47] http://www.plugloadsolutions.com/80pluspowersupplies.aspx

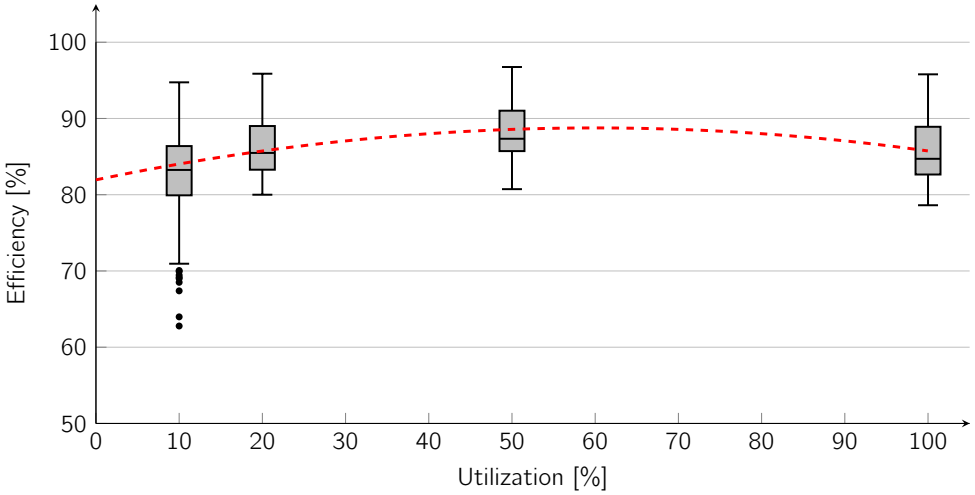
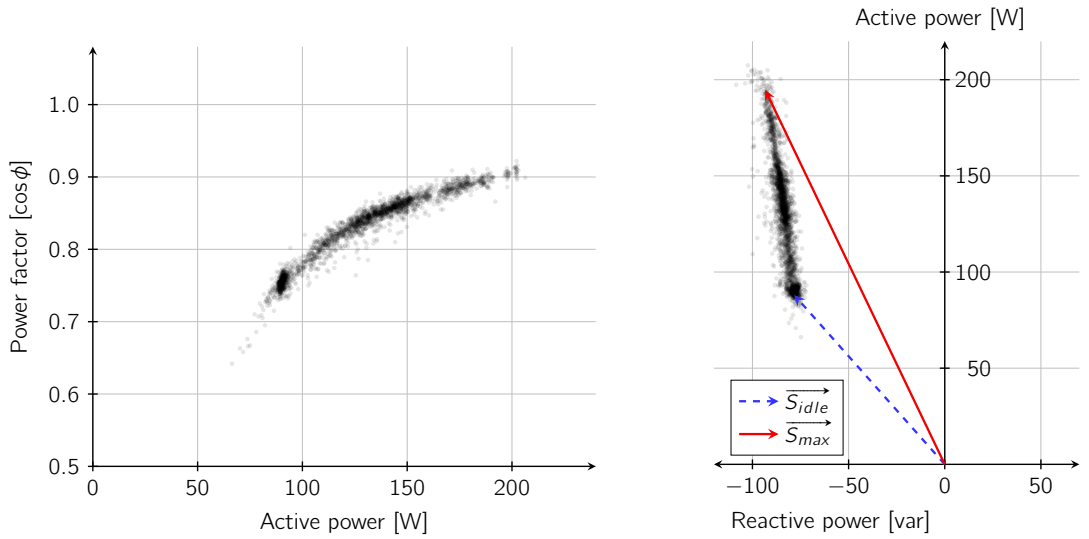


Figure 4.17: Power supply unit efficiency under different load conditions.

Example 4.2 *The power demand of a server may be estimated by summing up the power demand of its subcomponents (cf. Section 4.3.1). Examples for these subcomponents include, e.g., CPU, RAM and fans [15]. However, the aggregated power of its subcomponents is subject to the limited AC/DC (and voltage level) conversion efficiency of its power supply, affecting the exhibited total power demand towards the AC domain. Assuming a power supply efficiency of 83 %, and a server subcomponent power demand of 384 W, the power demand (viewed from the AC domain), is* $\frac{1}{0.83} \sum P_{sub} = 1.2048 \cdot 384\,\mathrm{W} \approx 463\,\mathrm{W}$.

Apart from a utilization-dependent power conversion efficiency, PSUs also exhibit a utilization-dependent capacitive PF. Although generally a high PF is desirable to minimize reactive current flow, certain situations may benefit from an intentional deviation from unity PF. A potential scenario may be devices that exhibit an inductive PF, operated in the same facility as the discussed ICT devices. In fact, DCs require HVAC equipment to maintain the required conditions for the operated ICT devices. This HVAC equipment includes compressors and fans, both include electric motors which exhibit an inductive PF. Figure 4.18a depicts PSU PF measurements at different utilizations. It is noteworthy that there are temporary dips in PF to below 0.8. These are of temporary nature and occur in situations of sudden drops in processor utilization.

Figure 4.18b identifies the active and reactive power components in both idle and maximum utilization conditions. From this, the amount of continuously sustainable P-Q demands for a higher number of servers can be extrapolated by employing Equation (4.44).



(a) Power factor at different utilizations.

(b) Complex power demand vectors.

Figure 4.18: Reactive power analysis of a server PSU.

$$\overrightarrow{S_{total}} = N \cdot \overrightarrow{S_{idle}} + \sum_{i=1}^{N} \varepsilon_i (\overrightarrow{S_{max}} - \overrightarrow{S_{idle}}), \tag{4.44}$$

with $\overrightarrow{S_{total}}$ denoting the total complex power demand, N the number of ICT devices, $\overrightarrow{S_{idle}}$ the idle complex power demand, $\overrightarrow{S_{max}}$ the maximum complex power demand, and $\varepsilon_i \in [0, 1]$ a scaling factor depending on the PSU utilization of the i^{th} ICT device. It is noteworthy that for Equation (4.44) to hold, the ICT devices need to be homogeneous in the sense that they all exhibit similar P-Q patterns.

The ideas regarding reactive power planning presented in Section 3.8.1 are supported by these findings. In case capacitive reactive power is requested by a GA, a DC may redistribute load among its ICT equipment[48] such that its PF is minimized. From Figure 4.18a, it can be deduced that load should be distributed such as to minimize active power demand of each ICT device. In contrast, a high power factor is achievable by consolidating load to maximize PSU utilization while shutting down idle servers to eliminate low power factor operation. Approximating complex power demand at different PSU utilizations by $\varepsilon(\overrightarrow{S_{max}} - \overrightarrow{S_{idle}})$ is feasible, as reactive power demand Q can be derived from active power demand P with high accuracy. Equation (4.45) gives the P-Q

[48] It is assumed that services may move between different hardware substrates by means of virtualization and migration capabilities.

relation. The linear regression result given in Equation (4.45) yields a low mean relative error of 2.55 %.

$$Q = -0.1358P - 66.0617. \tag{4.45}$$

However, although ICT device's PSUs exhibit a capacitive PF, it is noteworthy that additional DC equipment (e.g., cooling) requires electric motors, which exhibit an inductive PF. Depending on the peculiarities of the regarded DC, the total PF towards the public grid may vary. By adjusting ICT utilization through workload allocation and selective power down of servers, limited control on total PF can be exerted.

4.6.3 Data Center Energy Layer Boundary

When broadening the view from single ICT devices to complete DCs, several additional effects have to be considered. The energy used in computing systems is being transformed into heat, which has to be managed in order to keep the DC environment inside manufacturer specifications. In case ICT equipment is hosted in a dense way, in addition to its own consumption, several additional sources of power demand have to be considered. This additional demand is caused primarily by heat exchangers that are required to keep the equipment's environment inside its operational specifications. Furthermore, other equipment contributes to power overhead, such as humidity control equipment, lighting, etc. This overhead of power demand in DCs is summed up in a factor known as PUE, formally given in Definition 4.4. By multiplying the ICT equipment power with the DC's PUE value, an estimation of the entire facility power demand can be achieved. PUE was introduced in 2006 by a non-profit organization named *The Green Grid*[49] and has become the de-facto standard to measure DC energy efficiency [87, 159].

Defintion 4.4: Power Usage Effectiveness

PUE is defined as the ratio of the total power required to run a DC facility to the total power drawn by all its ICT equipment as shown in Formula (4.46).

$$\text{PUE} = \frac{\text{Total facility power}}{\text{ICT equipment power}}. \tag{4.46}$$

[49] `http://www.thegreengrid.org`, accessed 22.01.2017.

Applying the knowledge on relevant ELBs in a DC ecosystem, the power demand of a DC exhibited at its connection point to the public grid infrastructure may be expressed as in Equation (4.47).

$$P_{DC} = PUE_{DC}(\sum_{j=1}^{M} \eta_{PSU_j}(\sum_{i_j=1}^{N_j} P_{Component_{i_j}})), \tag{4.47}$$

with M denoting the total number of server PSUs in a DC, η_{PSU_j} the efficiency of the PSU of server j, M_j the total number of components in server j, and $P_{Component_{i_j}}$ the power demand of the i^{th} component of server j.

Example 4.3 *Assuming a PUE of 1.42 and an ICT power of 600 kW, the total power drawn from the grid amounts to* $1.42 \cdot 600\,\text{kW} = 852\,\text{kW}$.

It should be noted that PUE may generally not be assumed to be constant: diurnal and seasonal changes in PUE occur due to changes in environmental temperature. Additionally, different overall ICT device utilization may lead to changes in DC power efficiency, as outlined by Avelar et al. [8]. This can be observed in PUE data published by Google[50]. It is thus advisable to update PUE values in the proposed model periodically.

4.7 Non-ICT Power Flexibilities

Apart from equipment directly responsible for the provisioning of computing capabilities, other hardware commonly found in DCs may be employed to provide power demand flexibility. The flexibility that can be harnessed from these devices is often of limited duration and usually requires some time before being ready to be used again. In the following, two major contributors to power demand in a DC's auxiliary infrastructure are investigated in more detail: cooling equipment and UPSs.

4.7.1 Cooling

As stated in Section 4.2, computing equipment dissipates its power draw in the form of heat. At the same time, there are strict limits to maximum IC operating temperature to ensure reliability over extended periods of time. To keep IC temperature inside a specified range, cooling of ICT devices is necessary.

[50] `https://www.google.com/about/datacenters/efficiency/internal/`, accessed 19.02.2019.

In a first step, generated heat needs to be distributed across a surface area larger than that of the actual IC, as power density in ICs can be extremely high (up to around $1\,\mathrm{W\,mm^{-2}}$ [40]). To distribute the heat across a larger area, heat spreaders are used, usually made of materials with high thermal conductivity (e.g., aluminum or copper). In low power applications, this larger surface area allows for natural convection to provide cooling. Higher loads or high-density environments require a faster removal of heat from the heat spreader. Two different types of heat transfer techniques are of special significance in DC applications: air cooling via forced convection and water cooling.

Air cooling: In air cooling, heat transfer to the surrounding air is improved by forcing a high air volume across the heat spreader. Figure 4.19 shows the working principle of air cooling.

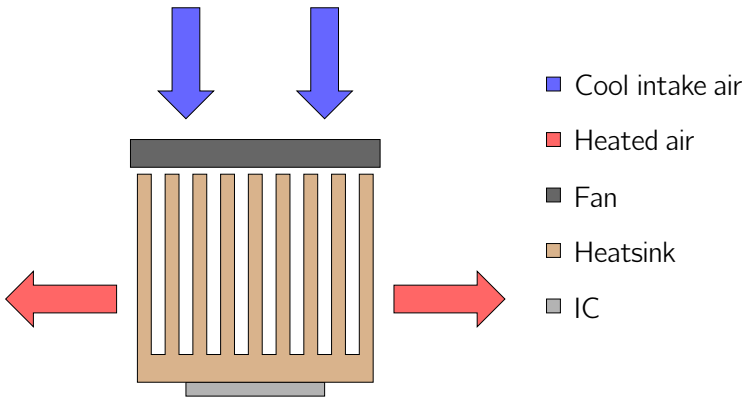


Figure 4.19: Conceptual model of air cooling.

This way, the heat difference between heat spreader and air moving across it is increased, therefore increasing heat transfer according to Formula (4.48).

$$q = h_c A \Delta T, \tag{4.48}$$

with q denoting the heat transferred per unit time, h_c the convective heat transfer coefficient of the process, A the heat transfer area of the surface, and ΔT the temperature difference between the surface and the ambient air.

Although specific details on operating environments are given by hardware vendors, general guidelines for DCs have been published by the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) [165]. The environmental guidelines specify, among others, recommendations regarding operating temperatures for DCs, namely 64.4 °F to 80.6 °F (18 °C to 27 °C). Providing electrical power demand flexibility from HVACs devices relies on the thermal

inertia present in a DC. Assuming an air-cooled DC, the duration over which HVAC devices can be switched off largely depends on baseline temperature, available air volume and ICT device power dissipation.

Water cooling: Water cooling employs water as heat transport medium instead of air. Figure 4.20 shows the working principle of water cooling.

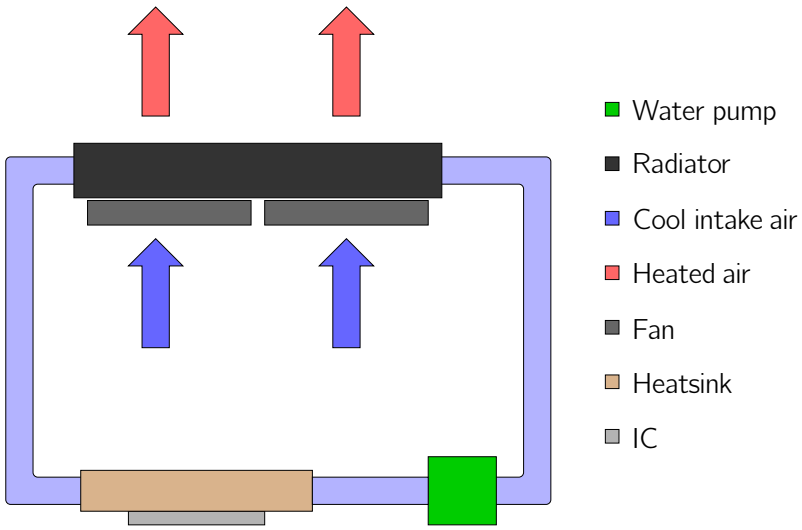


Figure 4.20: Conceptual model of water cooling.

In several aspects, water shows favorable properties as a cooling medium when compared to air, as detailed below. However, water cooling adoption is still rare: many DC operators are hesitant of the required initial investment and are reluctant to install water carrying equipment close to electronics due to possible leakage. Accordingly, most DC equipment is sold pre-configured for air cooled environments.

It is also noteworthy that hybrid cooling approaches exist, encompassing some components cooled with water (mostly high-power ICs like CPUs) and others with air cooling. These scenarios are not considered further at his point.

Comparing the two cooling techniques regarding their respective fitness to be used in power adaptivity, the primary qualification criterion is the ability to provide thermal inertia. Thermal inertia enables the cooling system to trade a swift change in power demand for a gradual change in component temperature. To quantify thermal inertia, the properties of the cooling medium are essential. More concretely, three values define thermal inertia: thermal conductivity k, density ρ,

and specific heat capacity c. Formula (4.49) gives the equation for calculating the thermal inertia I.

$$I = \sqrt{k\rho c} \tag{4.49}$$

Using Formula (4.49), air and water are compared regarding their thermal inertia.

Medium	Thermal conductivity k	Density ρ	Specific heat capacity c	Thermal inertia
Air	0.0262	1.184	1.012	0.177
Water	0.5562	997.048	4.181	48.152

Table 4.11: Comparison of thermal inertia of air and water.

Table 4.11 shows that the thermal inertia of water is more than two orders of magnitude higher than that of air. Two points have to be considered with respect to thermal inertia: on the one hand, the value of thermal conductivity k is only relevant in so far as it is sufficient to remove heat from ICs fast enough to stay within heat tolerance limits. However, even disregarding this factor, the thermal properties of water are still far superior to air. On the other hand, a DC does generally hold a much higher volume of air than water in its respective cooling system (primarily as it does not require any containment due to its inherent insulating properties).

4.7.2 UPS

UPSs are designed to ensure DC operation if security of supply cannot be maintained from power grid side. Power is then switched to be provided from the UPS-internal energy storage[51]. Power is fed from the internal storage to an inverter and supplied to the DC's equipment. As the name implies, switching from grid power to internal supply is fast enough to supply power in an uninterrupted way. After grid power is restored, the internal energy storage is recharged. Grid outages of longer duration can only be coped with using dedicated generators as UPS autonomy time is limited by its internal energy storage capacity.

Using UPSs as suppliers of electrical power demand flexibility requires a voluntary switch from power grid supply to UPS-internal energy storage. Similar to operation in emergency cases, the UPS is capable of relieving the grid of all load created by the DC's equipment for a limited period of time.

[51] Most commonly stored in batteries, other forms of energy storage (e.g., flywheels) do exist.

4.7.3 Challenges

Harnessing electrical power flexibility from hardware devices as described in Sections 4.7.1 and 4.7.2 poses several challenges. These are discussed in the following.

Recovery: Reducing cooling results in temporary power savings, however this adaptation cannot be sustained indefinitely: as components approach the limits of their safe operating temperatures, cooling needs to be resumed to avoid damage to the equipment. The same holds true for a UPS: if its internal batteries are discharged to lower DC power draw from the public grid, additional power is drawn later to recharge. The flexibility is thus limited to shifting demand in time, in other words, it refers to flexibility in power demand, energy flexibility is not achieved[52].

Granularity: Another downside of hardware-based power flexibility is a lack of intermediate power adaptation states, i.e., often only two states—on or off—are supported. While some cooling devices may support different operating points, in UPSs power either is provided by the public grid or internal storage. Large UPS devices rated for several kVA may therefore create comparatively large impacts on the power demand from grid side.

Availability: Harnessing power flexibility from temporarily powering down cooling or switching to UPS devices may conflict with the goal of maximum availability of a DC. Increasing temperature of solid-state devices is known to substantially affect their expected lifetime [30]. Especially the repeated occurrence of quick temperature changes severely degrades IC life expectancy, as shown by Huang et al. [73]. A similar challenge arises if UPSs are regularly used to shift DC power demand: on the one hand, batteries can only tolerate a limited number of charge/discharge cycles [4]. This means that repeated use of a UPS for power adaptation effectively decreases its mean time between maintenance (MTBM) (i.e., replacing internal batteries). On the other hand, discharging a UPS system in non-critical situations may leave the autonomy time degraded in case the adaptation time coincides with a power grid outage, creating the risk of complete loss of power supply to DC equipment[53]. Due to the very high importance of availability in DC ecosystems [115], there is significant reluctance to use UPS systems in this way.

[52] In case a cooling device exhibits varying levels of efficiency depending on inlet temperature, actual energy demand may vary. This is however outside the scope of this work.

[53] Arguably, a grid outage is even more likely to occur in adaptation scenarios as this very adaptation request may be caused by a grid imbalance.

4.8 Multi-Modal Flexibility Model

D/R is concerned with exploiting flexibilities present in the electricity demand of consumers. It is important to consider that electricity is an almost universal energy carrier. In a DC environment, electricity is converted to other forms of energy for use in different domains. These domains may be interrelated, i.e., available inertia in one domain may be affected by changes to another domain. This becomes especially apparent in a DC context. Three major subsystems of a DC are supplied with electrical power: ICT systems, cooling, and UPS. These subsystems perform different energy conversions on their input, as described in the following.

- ICT: electricity → heat,
- Cooling: electricity → cold[54], and
- UPS: electricity → chemical energy[55].

As previously discussed, these domains do not operate independently: cooling is only provided to the extent that is required to keep ICT component temperature inside a well-defined range; UPS systems only deliver power to the extent that is required by ICT systems and cooling (if no grid power is available). In this way, dependencies between electricity conversion domains may be established. More formally, let D_x denote an energy conversion domain. $I(D_x)$ is the amount of inertia available in domain D_x, i.e., the time the domain can supply its dependent domains without external power input. $P(D_x, D_y)$ describes the inter-domain power requirement of D_x from D_y. $F(D_x)$ denotes the flexibility available in domain D_x, i.e., the maximum possible power shift regarding external power input to the domain. Figure 4.21 shows a conceptual system of two interdependent domains.

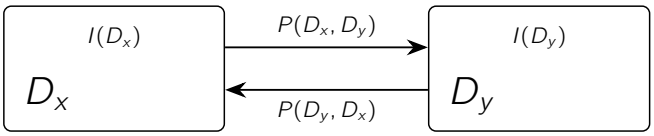


Figure 4.21: Generic concept of domain power interdependencies.

The generic concept shown in Figure 4.21 is applied to a DC use case. In a DC environment, an example of inter-domain power requirements is between the combined power demand of ICT systems and cooling to the UPS domain. This is depicted in Figure 4.22.

[54] In fact, *cold*—in contrast to *heat*—is not a form of energy, but lack thereof. However, cooling equipment requires cooling media to be moved and, in case of air conditioning, heat exchange via compression and vaporization of a cooling fluid. This is achieved via electric motors, which convert electricity to kinetic energy.

[55] Assuming a battery based UPS. UPS systems based on, e.g., flywheels would convert electricity into kinetic energy and vice versa.

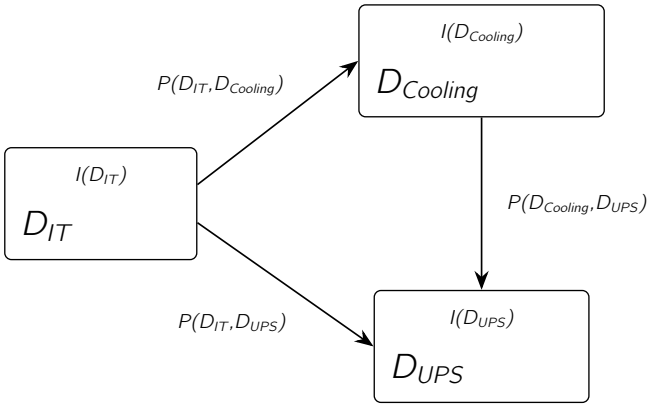


Figure 4.22: Domain power interdependencies in a DC environment.

While in normal operation, the UPS domain does not draw any power[56]. In case of a power outage, the UPS system has to supply power to the ICT and cooling domains. As there is neither power being fed to the UPS domain nor operating point adjustments possible, the power demand of ICT and cooling has to be served by the inertia present in the domain's energy reserves. The UPS inertia, which is equal to its autonomy time, can be estimated using Equation (4.50).

$$I(D_{UPS}) = \frac{(SoC_{bat} - (1 - DoD_{max})) \cdot C_{bat} \cdot \eta_{inv}}{P_{ICT} + P_{cooling}} \tag{4.50}$$

with SoC_{bat} denoting the state of charge of the battery, DoD_{max} the maximum allowed depth of discharge, C_{bat} the capacity of the battery, η_{inv} the inverter efficiency, P_{ICT} the ICT power demand, and $P_{cooling}$ the cooling system power demand. The available flexibility of the UPS domain is the entire DC power demand, in this simplified model $P_{ICT} + P_{cooling}$.

Cooling system inertia is determined by the capability of keeping acceptable environmental conditions for ICT devices in the absence of cooling medium refrigeration. This time depends on the type and volume of the cooling medium, the baseline temperature and power from the devices to be cooled. This is shown in Equation (4.51).

$$I(D_C) = \frac{\Delta K_{max} \cdot c_V \cdot V}{P_{ICT}} \tag{4.51}$$

[56] Assuming an offline UPS and neglecting self discharge. Online UPS systems incur a loss of approximately 5 % to 10 % (see, e.g., [118]).

with ΔK_{max} denoting the maximum allowed temperature increase, c_V the isobaric volumetric heat capacity, V the available volume of the cooling medium, and P_{ICT} the ICT power demand. It is noteworthy that Equation (4.51) is only valid if a perfect mixing of the cooling medium is assumed, i.e., a uniform heating of the cooling medium. To legitimize this assumption, it is assumed that only the actual refrigeration of the cooling medium is switched off during power adaptation, not the cooling medium flow.

Cooling flexibility, regarding its magnitude in power, is thus limited to the power of the chiller (the device responsible for cooling medium refrigeration). Pelley et al. [139] estimate that "[...] chiller power dominates overall cooling system power, and its requirements grow with both the data center thermal load and outside air temperature". Rasmussen [151] gives a more detailed estimation of power flows inside a DC, estimating chiller power at 23 % and CRAC/CRAH power at 15 % of total DC power demand (assuming a DC PUE of 2.13). In addition, a breakdown of cooling equipment inefficiencies, as given in [151], is shown in Table 4.12.

	No Load Loss	Proportional Loss
CRAC/Water pumps	9 %	0 %
Chiller plant	6 %	26 %

Table 4.12: Power losses in cooling equipment.

The power flexibility harnessed from the cooling system is limited to the chiller plant power, as the cooling medium still needs to be circulated by fans or water pumps. Depending on whether the chiller plant can be switched off entirely or needs to be kept running in idle mode, the power demand flexibility is either $P_{ICT} \cdot PUE \cdot 0.23$ or $P_{ICT} \cdot PUE \cdot 0.23 \cdot 0.94$, respectively.

Example 4.4 *Regarding a medium-sized DC as according to Section 4.1 with an assumed ICT load of 250 kW and PUE of 2.13 (as in [151]), the inertia and flexibility of the UPS and cooling system can be estimated as follows:*

UPS:

Total DC power $= P_{ICT} \cdot PUE = 250\,\text{kW} \cdot 2.13 = 532.5\,\text{kW}$.
ICT and cooling power $= P_{ICT} + P_{cooling} = 250\,\text{kW} + 532.5\,\text{kW} \cdot (0.23 + 0.15) = 452.35\,\text{kW}$.

Assuming $SoC_{bat} = 100\,\%$, $DoD_{max} = 80\,\%$, $\eta_{inv} = 98\,\%$, $C_{bat} = 350\,\text{kW h}$, *and that only the ICT and cooling domains need to be provided with UPS power:*

$I(D_{UPS}) = \frac{(1-(1-0.8))\cdot 350\,\text{kW h}\cdot 0.98}{452.35\,\text{kW}} \approx 36\,\text{min}$.
$F(D_{UPS}) = 532.5\,\text{kW}$, *i.e., the entire DC power demand.*

Cooling:
Chiller power $= 532.5\,\text{kW} \cdot 0.23 = 122.475\,\text{kW}$.
CRAC/CRAH power $= 532.5\,\text{kW} \cdot 0.15 = 79.875\,\text{kW}$.
$F(D_{Cooling}) = 122.475\,\text{kW}$, *i.e., switching off the chiller.*

Assuming $T_{baseline} = 20\,°\text{C}$*:*

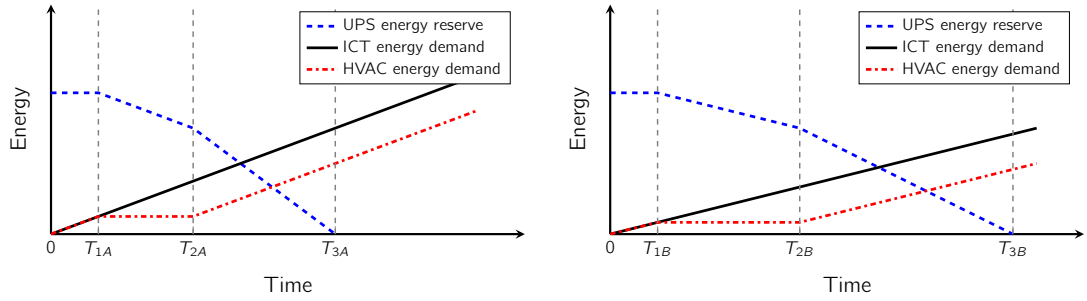
Maximum allowable temperature increase (according to ASHRAE guidelines, assuming air cooling and a DC of class A1): $\Delta K = 32\,°\text{C} - 20\,°\text{C} = 12\,\text{K}$. *For water cooling, due to its superior thermal conductivity, cooling medium temperature may increase significantly higher. Product developments by, e.g., CLOUD&HEAT Technologies GmbH*[57]*, have shown the feasibility of liquid cooling ICT devices with coolant temperatures of more than 45°C*[58]*.*

$$I(D^{Air}_{Cooling}) = \frac{\Delta K_{max} \cdot c_V \cdot V}{P_{ICT}} = \frac{12\,\text{K} \cdot 1.2106\,\text{kJ}\,\text{K}^{-1}\,\text{m}^{-3} \cdot V}{250\,\text{kW}} = 0.058\,11\,\text{s}\,\text{m}^{-3} \cdot V$$
$$I(D^{Water}_{Cooling}) = \frac{\Delta K_{max} \cdot c_V \cdot V}{P_{ICT}} = \frac{15\,\text{K} \cdot 4179.6\,\text{kJ}\,\text{K}^{-1}\,\text{m}^{-3} \cdot V}{250\,\text{kW}} = 250.776\,\text{s}\,\text{m}^{-3} \cdot V$$

The superior thermal properties of water increase cooling system inertia by a factor of more than 4×10^3 *per volume. However, it is noteworthy that the cooling medium volume is generally much higher in air cooled environments, as air does not need any containment measures (in contrast to, e.g., water).*

Coming back to Figure 4.22, another important point regarding total DC inertia and therefore, flexibility towards the power grid, becomes apparent: power drawn by ICT devices affects UPS inertia time not only directly, but also via the cooling domain. The lower the power demand of ICT devices, the less power is required from the cooling domain and in turn, autonomy time of the cooling domain is increased. An example power adaptation in the ICT domain is now analyzed qualitatively regarding its influence on DC adaptation potential. Figures 4.23a and 4.23b show the influence of decreasing the power demand of the ICT domain. In Figure 4.23b, the ICT power demand is reduced by 33 % in comparison to Figure 4.23a. This results in an increased time until the thermal inertia of the cooling domain is used up, i.e., the time at which the cooling system has to be switched on ($T_{2B} > T_{2A}$). In turn, the autonomy time of the UPS system increases ($T_{3B} > T_{3A}$). Both scenarios assume a 1 : 1 ratio of ICT and cooling system energy demand.

[57] `https://www.cloudandheat.com/`
[58] For the following calculations, a conservative estimation of 35 °C maximum coolant temperature is used.



(a) Energy under high ICT power demand.

(b) Energy under low ICT power demand.

Figure 4.23: Maximum UPS autonomy time assuming different ICT power demand.

As can be seen, adapting the power demand of ICT devices causes increased autonomy time through add-on power savings which make it a favorable domain to increase flexibility. After establishing the most feasible domain of power adaptation, the following Chapter 5 is concerned with identifying the exact point of action *within* the ICT domain.

4.9 Summary and Future Work

This chapter provided an overview of ICT power demand, its sources and possibilities of estimation based on high-level hardware information. Starting from a low-level analysis of ICT power demand, the possible parameters to influence IC power demand during operation were identified. It was concluded that influencing the activity factor α_{IC} enables further savings by reducing IC voltage through DVFS, and is therefore the most feasible point of action. In the following, a power demand model based on a server decomposition into independent components was presented. It enables an estimation of a server's power demand across a wide variety of utilization conditions using only information readily available in common monitoring systems. Applying the knowledge of ELBs, it is possible to estimate power demand of an entire DC based on hypothetical future hardware utilization. Finally, different domains of hardware power demand were analyzed regarding their flexibilities, inertia and interdependencies. From the results of Sections 4.7.3 and 4.8, a significant conclusion can be drawn: if high power flexibility should be achieved, power should be reduced in the ICT domain. The reasons are threefold: first, ICT power flexibilities do not suffer from the inherent flaws outlined in Section 4.7.3. As will be discussed in Chapter 5, especially the granularity of ICT-based power adaptation is usually superior to a binary power state hardware device. Second, power adaptations in ICT cause a reduced heat dissipation of equipment. This in turn reduces the required effort—and thus power demand—in the cooling domain. As both ICT and cooling domain are supplied with power by the UPS domain, its autonomy time increases linearly with the power

reduction[59]. Third, adapting hardware utilization is a wear-free process, as opposed to, e.g., cycling UPS batteries. Especially if used on a regular basis (i.e., not only in emergency situations), this is important to not cause reliability challenges in hardware devices.

There are several possible directions for future work: some of the power estimation models employed in Section 4.3 were derived from components that can at the time of writing not be considered state of the art. A repeated evaluation with newer components may reveal models which need to be updated, as was shown for the CPU model. Related work is available in this direction, as outlined in Chapter 2. Another possible future research direction may consider improving power flexibility via increased cooling system inertia: several studies, e.g., by Greenberg et al. [60] demonstrated the overall superior energy efficiency of water cooling systems in DC environments. Further increase of thermal inertia requires either decreasing baseline water temperature or increasing coolant volume. Decreasing water temperature to less than environmental temperature requires special equipment to be installed. If an adequate lead time is given, the adaptation duration can be increased by investing more power to create a thermal buffer in the cooling medium. Work by, e.g., Zheng et al. [190] discuss the possibility of exploiting thermal energy buffers in DCs in more detail.

[59] Autonomy time may actually increase more than linearly, as apparent battery capacity increases as discharge speed is reduced [148].

Intelligence is the ability to adapt to change.

Stephen Hawking

5

Software Resource Demand

This chapter contains material previously published in [129] and [128].

Chapter 4 established a relation between ICT power demand and the utilization of its hardware components. Building on this knowledge, the following chapter discusses possibilities to influence hardware utilization (i.e., as previously established, circuit activity factor α_{IC}) by means of software resource utilization control. This is the second step towards exerting *control* on a DC's power demand and a requirement to realize a software-based power adaptation system.

Section 5.1 states the assumptions taken in this chapter. Section 5.2 introduces the concept of *supervenience*. It serves as basis to conceptualize a software-driven approach to power demand adaptation in ICT systems. Section 5.3 further refines the *running example* used throughout this thesis. It is narrowed down to the specific use case of a DC specializing in live video encoding/transcoding. In Section 5.4, *power adaptation layers* are investigated. More specifically, the effects of changing ICT resource availability and utilization via different techniques are evaluated with regard to their (possibly unintended) side effects on other parts of the ICT system.

A measure to quantify these side effects is presented. Section 5.5 analyzes the specific *flexibilities* present in a live video encoding/transcoding scenario. The considered flexibility dimensions are presented and evaluation methodologies are explained. An evaluation of a multi-dimensional adaptation is performed. Additionally, the concept of an optimal adaptation sequence (OAS) is introduced. In the following, Section 5.6 compares the results achieved by performing a *software-extrinsic* and *software-intrinsic* power adaptation. The results show that in the evaluated scenario, a software-intrinsic adaptation clearly outperforms a software-extrinsic one. Subsequently, Section 5.7 presents a possible integration of the proposed adaptive live video encoding into the DASH standard. Finally, Section 5.8 concludes this chapter and gives an outlook on possible future research directions.

5.1 Assumptions

In the following sections, several assumptions are taken to limit the scope of this chapter. These are discussed in the following.

1. It is assumed that there is *no contention for server resources* among software. This assumption is backed by low average CPU utilization measured in DCs, typically below 50 % [12, 13]. Some studies estimate it even lower at around 10 % [29]. In case there is resource contention on individual servers, this can be mitigated via software migration or load balancing.

2. It is assumed that the software regarded in the following sections is *mutually independent*, i.e., no software depends on services or input provided by other software. In case there are software dependencies, the interdependent software components may be considered as single software in the following, as long as being hosted on the same substrate hardware. Interdependent software hosted on different hardware substrates is excluded from the following considerations.

3. It is assumed that *no nested virtualization*, i.e., further virtualization inside an already virtualized environment, is employed. This assumption is considered valid as nested virtualization is not widely used in practice [135]. In case this assumption does not hold, the model derived in the following would need to be modified to allow for an arbitrary number of virtualization layers nested within each other.

4. The term *virtualization* is used to describe both virtualization and containerization, as both allow for imposing limits on resource utilization. This is also the rationale behind not further differentiating virtualized software into OS and non-OS, as containers do not need an OS separate from the host OS [117].

5. Unless stated otherwise, computing devices are assumed to operate according to the *energy proportional computing* paradigm described by Barroso et al. [13]. This simplification improves understandability and could be replaced by more sophisticated power demand models as introduced in Chapter 4. Specifically, this assumption allows bypassing the additional challenge of virtual machine embedding in a hardware substrate, which is outside the scope of this work, but has been widely covered in literature (e.g., [120, 146]).

5.2 Supervenience

In the further course of this thesis, software is used to influence the power demand of ICT components. However, software itself does not require any power. Still, it may be used to influence the power demand of its hosting hardware substrate. Any two different software states (cf. Definition 5.2) exist if and only if their hardware substrate is in different hardware states. This relationship is known as *supervenience*. Although originally introduced as a concept in philosophy by Moore [123], in this thesis it is adapted to fit into a computer science context. Lewis [98] gives the following example for supervenience:

> "A dot-matrix picture has global properties – it is symmetrical, it is cluttered, and whatnot – and yet all there is to the picture is dots and non-dots at each point of the matrix. The global properties are nothing but patterns in the dots. They supervene: no two pictures could differ in their global properties without differing, somewhere, in whether there is or isn't a dot."

While easy to understand, this is merely an example and not a clear definition. In the context of hard- and software, supervenience means that each change in software has to be preceded by a change in its hardware substrate. Definition 5.1 gives a formal definition of supervenience in the context of soft- and hardware. Adapted from the definition of strong supervenience by Wilson [182], Expression (5.1) formally defines software-hardware supervenience.

Defintion 5.1: Software-Hardware Supervenience

Software supervenes on its substrate hardware, i.e., any change in the software state is preceded by a change in its substrate hardware state.

$$(\forall x \in D)(\forall s \in S)(SS(x) = s \rightarrow (\exists h \in H)(HS(x) = h) \wedge (\forall y \in D)(HS(y) = h \rightarrow (SS(y) = s))), \quad (5.1)$$

with D denoting the set of computing devices, H the set of substrate hardware states, S the set of hosted software states, $HS(x \in D)$ the current hardware state of x, and $SS(x \in D)$ the current software state of x.

In other words, there exists a surjective function f from the domain of HS to the codomain SS. For every element s in the codomain SS of f there is at least one element h in the domain HS of f such that $f(h) = s$. In the following, the elements s are referred to as software states, as defined in Definition 5.2.

Defintion 5.2: Software State

A computer system's software state denotes the state of all memory which may be accessed by a software either reading or writing.

It is noteworthy that the converse conclusion to Expression (5.1) does not hold: there are generally several hardware states representing the same software state, e.g., those that only differ with regard to contents of non-allocated RAM or hard drive redundancy. With Definition 5.1, Corollary 4.1 can be extended as given in Corollary 5.1.

Corollary 5.1

The combination of hardware configuration and hardware state describes a computer system's hardware and software completely, as these convey all time-invariant and time-variant hardware information. The complete hardware information also carries the complete software information.

By applying the principle of supervenience to ICT systems, several design principles for power adaptivity become apparent. First, there may be possibilities to change the power demand of a hardware substrate without impacting software execution. This is indeed a feasible way and exploited by employing, e.g., DVFS (cf. Section 4.2.1.3). Reducing CPU frequency is functionally

transparent and, given CPU utilization is low enough, also mostly performance-neutral [176]. If functional requirements of hardware are restricted to very specific tasks, it may increase performance and power efficiency to create an application-specific integrated circuit (ASIC), i.e., a computing chip tailored to perform a limited set of specific tasks only and thus eliminating irrelevant parts of a general-purpose CPU. A noteworthy example is the Intel AES-NI implementation, utilizing a hardware supported instruction set to speed up encryption tasks [158].

Knowing that any software state change is preceded by a hardware state change enables dynamic power demand management on software level: referring back to Approximation (4.11), changing software operation allows for influencing the circuit's switching activity α_{IC}, as outlined in Section 4.2.1.2.

5.3 Running Example, Third Iteration

The running example introduced in Section 3.3 and specified more concretely in Section 4.1 is now narrowed down further. It is assumed that the DC participating in the D/R program specializes in video data processing. More concretely, the regarded DC serves two different types of load: interactive and batch load. Both types have respective representative tasks in video processing.

Offline transcoding (i.e., batch load) is an important use case for streaming video service providers like Netflix. In a Netflix technology blog[60] and a video linked in the blog entry[61], it is mentioned that Netflix creates 120 different versions of a video file employing different encoder settings to suit various end user devices. With several thousand video files—in a continuously renewed catalog—available on Netflix and similar platforms, a high number of transcoding processes has to be performed on a repeating basis.

In contrast, live video encoding is an interactive-type load. A recent example in DC environments is the NVIDIA Geforce Now program, which centralizes GPU-intensive rendering from user machines into a DC while delivering results in a live video stream[62]. In the following, service-type load is considered more closely. A mixed scenario of service- and job-type load is not excluded, however the scheduling of jobs is outside the scope of this work. Another prominent example of live video encoding is video conferencing. Although an initial encoding is usually performed on client side, different target devices may require an additional transcoding on server side.

[60] `https://medium.com/netflix-techblog/complexity-in-the-digital-supply-chain-958384cbd70b`, accessed 14.12.2017.
[61] `https://vimeo.com/52637219`, accessed 14.12.2017.
[62] `https://www.nvidia.com/en-us/geforce/products/geforce-now/mac-pc/`, accessed 01.02.2018.

5.4 Power Adaptation Layers

Modern computer systems allow for power adaptation on several different layers. Figure 5.1 shows a graph that orders system layers in a *is host of* relation. This order relation between system layers is required to formalize the interdependence of power adaptation on different layers.

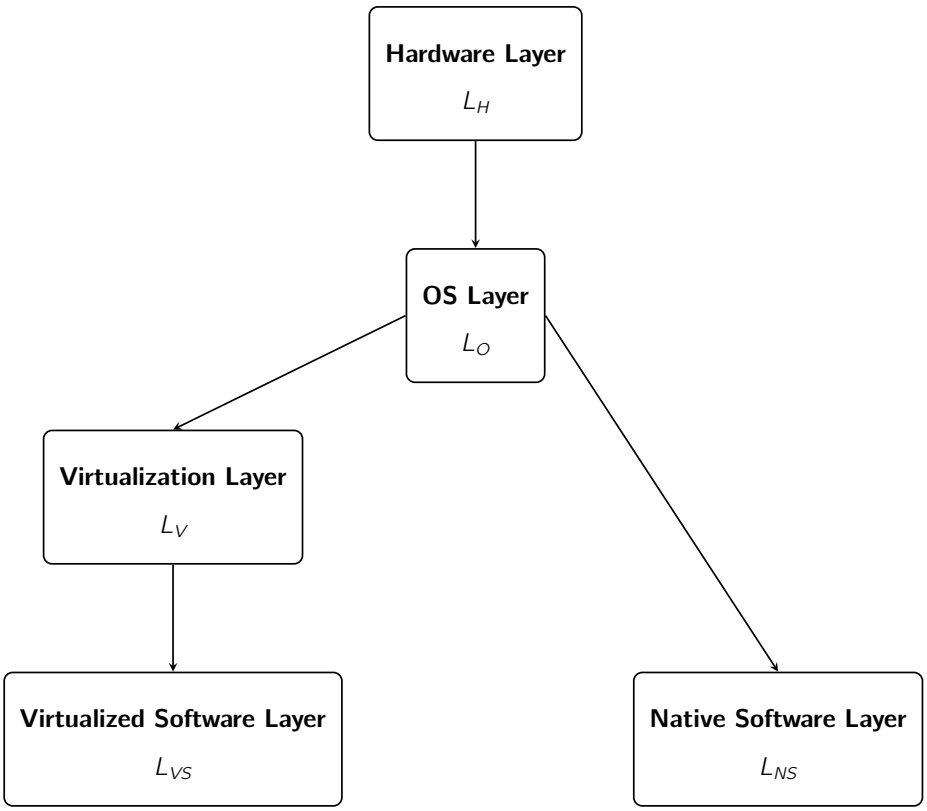


Figure 5.1: Power adaptation layers.

Depending on the layer the adaptation is performed on, other layers may be affected. Let $\Phi(L_X)$ denote the set of layers directly influenced by performing adaptations on layer L_X. Let $P = \{L_H, L_O, L_V, L_{NS}, L_{VS}\}$ denote the set of all power adaptation layers. Then the following equations hold true:

$$\Phi(L_H) = \{L_O\}, \tag{5.2}$$
$$\Phi(L_O) = \{L_V, L_{NS}\}, \tag{5.3}$$
$$\Phi(L_V) = \{L_{VS}\}, \tag{5.4}$$
$$\Phi(L_{NS}) = \emptyset, \tag{5.5}$$
$$\Phi(L_{VS}) = \emptyset, \tag{5.6}$$
$$\Phi(\emptyset) = \emptyset, \tag{5.7}$$

with $\Phi : P \rightarrow 2^P$. However, resource limits imposed on a layer indirectly also influence available resources on all layers hosted by subsequent layers. To capture this relationship, Equation (5.8) is defined such that it represents the transitive closure of the set of Equations (5.2) to (5.7).

$$\Psi(L_X) = \Phi(L_X) \cup \bigcup_{y \in \Phi(L_X)} \Psi(y), \tag{5.8}$$

with $L_X \in P$ and $\Psi : P \rightarrow 2^P$. In a similar manner, changes in resource utilization on a (virtualized) software layer propagate to all layers involved in hosting the respective software. Let $\Phi^{-1}(L_X)$ denote the set of layers directly influenced by changes in resource utilization on layer L_X. Then the following equations hold true:

$$\Phi^{-1}(L_H) = \emptyset, \tag{5.9}$$
$$\Phi^{-1}(L_O) = \{L_H\}, \tag{5.10}$$
$$\Phi^{-1}(L_V) = \{L_O\}, \tag{5.11}$$
$$\Phi^{-1}(L_{NS}) = \{L_O\}, \tag{5.12}$$
$$\Phi^{-1}(L_{VS}) = \{L_V\}, \tag{5.13}$$
$$\Phi^{-1}(\emptyset) = \emptyset, \tag{5.14}$$

with $\Phi^{-1} : P \rightarrow 2^P$. Equation (5.15) captures all layers influenced by a resource utilization change on a layer L_X.

$$\Psi^{-1}(L_X) = \Phi^{-1}(L_X) \cup \bigcup_{y \in \Phi^{-1}(L_X)} \Psi^{-1}(y), \tag{5.15}$$

with $L_X \in P$ and $\Psi^{-1} : P \rightarrow 2^P$. Figure 5.2 visualizes the overall concept of resource limitation and utilization propagation over layers.

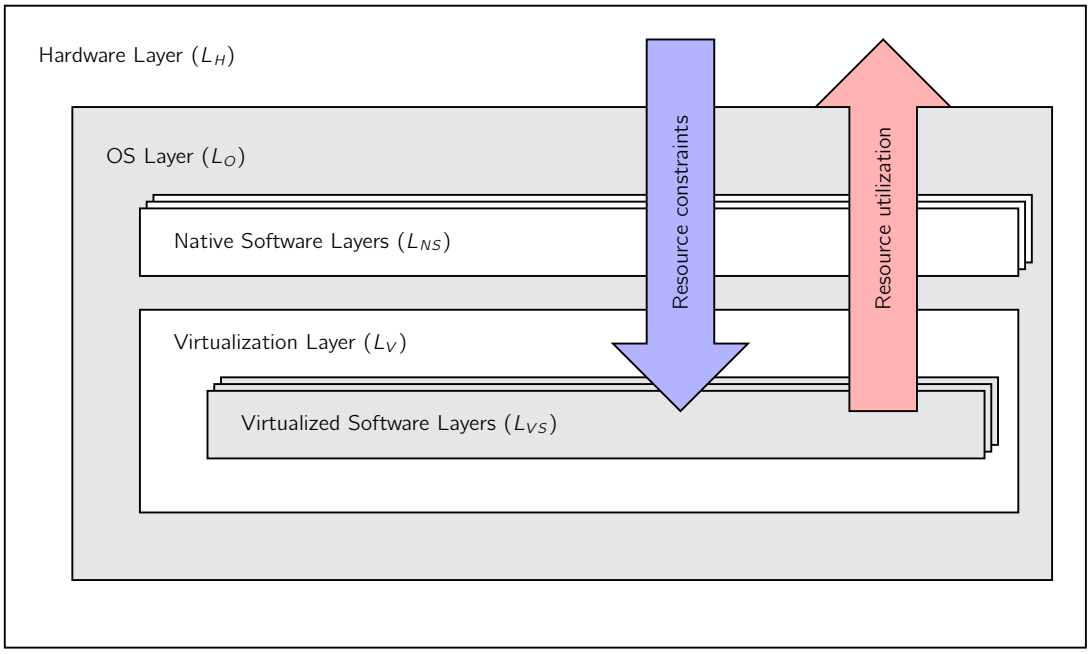


Figure 5.2: Interdependence of adaptation layers.

Restricting the resource utilization of a layer L_X therefore limits this layer's maximum power demand to a certain value, however influence on utility propagates to all layers which are part of $\Psi(L_X)$. This way, it is possible to define a quantitative measure of the resource limitation and utilization scope, as outlined in Definition 5.3 and 5.4.

Defintion 5.3: Resource Limitation Scope

The scope of limiting resource availability on a layer L_X can be quantified by the cardinality of the respective set $|\Psi(L_X)|$.

Specifically, it is noteworthy that adaptations on software layer have a cardinality of 0, i.e., no potentially unintended effects on other software are to be expected[63]. On the other hand, fixing utility of an element located on a software layer L_S causes effects on resource utilization to propagate to all layers L_X in the set $\Psi^{-1}(L_S)$.

[63] Assuming that there are no dependencies between software components.

Defintion 5.4: Resource Utilization Scope

The scope of adapting resource utilization on a layer L_X can be quantified by the cardinality of the respective set $|\Psi^{-1}(L_X)|$.

Due to the properties of adaptations performed on layers L_{NS} and L_{VS}, in the following an argument in favor of power management via changes in software execution is made: a restriction of available resources on layers that are hosting software may be non-transparent to hosted software, i.e., it can generally not be inferred whether the resource limitation is intentionally imposed or there is resource contention among hosted software. Much in the same way, hardware is oblivious to causes of component utilization. While the latter is acceptable in terms of power management, the former may result in a non-optimal tradeoff between power demand and service utility. A detailed observation for a specific service example is presented in the upcoming sections. In the following, an overview on technologies for power adaptation on different layers is given.

5.4.1 Technologies for Power Adaptation on Different Layers

In the following, available technologies for power adaptation on each of the layers are introduced.

Hardware layer: Limiting power on hardware layer effectively imposes limits on maximum resources available for computation. This can be configured either specifically per hardware resource or on an even higher level of abstraction for an entire computing device. The former may be achieved by low-level hardware configuration in a system's basic input output system (BIOS) or its more recent successor universal extensible firmware interface (UEFI). Both allow setting hardware parameters which may not be exposed to changes at runtime. These configuration options usually include setting frequency and voltage of CPU, RAM and mainboard system on chip (SoC) as well as CPU sleep and performance states to be used, among others. It is noteworthy that the manual configuration of these settings may allow for users to operate components outside of vendor specifications. By increasing frequency beyond specification (referred to as *overclocking*), additional performance may be gained at the cost of increasing power demand, while a reduction of supply voltage (referred to as *undervolting*), may decrease power demand at the expense of stability margins. Machine-wide power adaptation is often only available for modern server-class machines and implemented in, e.g., HP iLO[64] or Dell iDRAC[65]. These high-level power settings

[64] https://www.hpe.com/emea_europe/en/servers/integrated-lights-out-ilo.html, accessed 27.09.2018.
[65] https://www.dell.com/support/contents/us/en/04/article/product-support/self-support-knowledgebase/enterprise-resource-center/systemsmanagement/idrac, accessed 27.09.2018.

may influence several hardware components' settings at once and their exact behavior is often not formally documented.

OS layer: An OS may influence power demand of all hosted software by enacting control on hardware, primarily via DVFS (presented in detail in Section 4.2.1.3). These power adaptation features impact all software hosted on the same hardware substrate[66]. Potential negative effects on utility may propagate to all subsequent layers (virtualization and software) in case of resource scarcity. No discrimination among different services is possible, resource scarcity increases for all software sharing a common hardware substrate.

Virtualization layer: To raise overall hardware utilization while maintaining mutual software isolation, DCs increasingly adopt virtualization technology [33]. Hypervisors enable setting resource limits for CPU, RAM, etc. for each hosted VM. Potentially negative effects on utility may propagate to the virtualized software layer in resource contention scenarios. No discrimination among different software inside the same VM is possible. Resource scarcity increases for all software running on a common VM. Similar limitations apply to software containers (e.g., Docker containers[67]).

Software layer: Power adaptation on software level is further subdivided into two different categories: *software-extrinsic* and *software-intrinsic* power adaptation. Software-extrinsic power adaptation is based on process-level limits on resource utilization. Potentially negative effects on utility are limited to the resource-limited software only. In practice, there are several software utilities available, some are presented in the following.

- `nice`[68]: The `nice` command sets the scheduling priority of a process in a range from -20, the most favorable scheduling priority, to 20, the least favorable.
- `cpulimit`[69]: In contrast to the `nice` command, `cpulimit` is a third-party tool which limits the actual CPU usage of a process instead of scheduling priority. To this end, `cpulimit` employs the `SIGSTOP` and `SIGCONT` signals.
- `cgroups`[70]: In comparison to the two former tools, `cgroups` offers a more sophisticated approach to processor resource management. Processes may be structured in a hierarchical way to realize a more fine-grained resource management.

[66] In case both hard- and software substrate support per-core DVFS, only software running on the same CPU core are affected.
[67] `https://www.docker.com/`, accessed 27.09.2018.
[68] `https://linux.die.net/man/1/nice`, accessed 27.09.2018.
[69] `https://github.com/opsengine/cpulimit`, accessed 27.09.2018.
[70] `http://man7.org/linux/man-pages/man7/cgroups.7.html`, accessed 27.09.2018.

Apart from limiting the overall amount of CPU utilization, modern multi-core CPUs allow for another way of influencing software power demand: capping maximum thread parallelization. To this end, the tool `taskset`[71] enables a user to restrict a process to an arbitrary subset of available CPU cores. Running a software on a lower number of CPU cores affects only such software that is capable of profiting from multi-threading by limiting its utilization of additional CPU cores.

In contrast, software-intrinsic power adaptation requires a change to the computation performed by the software itself. Based on the past findings, three basic functional requirements can be defined that enable a runtime software-intrinsic power demand adaptation. These are investigated more closely in Section 5.4.2.

5.4.2 Requirements for Software-Intrinsic Resource Utilization Adaptation

To enable software-intrinsic power adaptation, several requirements need to be met. In the following, it is assumed that adaptation should be performed *at runtime*, i.e., without significant service interruption.

Requirement 1: Runtime reconfiguration
Services need to provide the means to reconfigure them during runtime, i.e., without a major disruption in service quality. This may be achieved by, e.g., adopting a new configuration without the need to restart a service, the capability of restarting a service very rapidly, or by employing a buffer that is able to mask the service's downtime resulting from a service restart.

Requirement 2: Configuration resource influence
The (re-)configuration options available during runtime need to provide at least one option that influences service resource utilization. As established throughout Chapter 4, influencing resource utilization of software is a possible way of influencing power demand. It is noteworthy that a software providing multiple independent dimensions of resource utilization influence via reconfiguration may benefit from a more graceful adaptation.

Requirement 3: Non-binary utility function
Runtime power adaptation is only feasible with software that provides at least one dimension of resource utilization scaling and at the same time has a non-binary utility function. This is now elaborated on in further detail.

[71] `https://linux.die.net/man/1/taskset`, accessed 27.09.2018.

Bentham [20] defines utility as

> "[...] that property in any object, whereby it tends to produce benefit, advantage, pleasure, good, or happiness, (all this in the present case comes to the same thing) or (what comes again to the same thing) to prevent the happening of mischief, pain, evil, or unhappiness to the party whose interest is considered [...]".

Based on this very broad definition, in the following utility of ICT systems is considered as defined in Definition 5.5.

Defintion 5.5: Utility of an ICT system

The utility of an ICT system is the degree of its output meeting the requirements of a considered consumer or group of consumers.

In ICT systems, utility is not inherent to the object itself, i.e., an ICT system that is non-operational does not deliver any utility in the sense of Definition 5.5 as it does not create any output. Applying the system view introduced in Section 3.2 to ICT systems, throughput and in turn output is created by committing computing resources. Kephart et al. [89] state that a utility function "[...] maps each possible state of a system or object under consideration to a real scalar value on a common scale [...]". Put more formally, a utility function U assigns a scalar utility value to a software state. Additionally, the time at which this software state is realized may be of importance (e.g., if a software instance is assigned a specific deadline). Thus, the term utility function is defined as given in Definition 5.6.

Defintion 5.6: Utility function

A utility function is a function mapping from the current software state s and point in time t to a scalar utility value u as shown in Expression (5.16).

$$U(s,t) \rightarrow u, \tag{5.16}$$

with s denoting a software state, t a point in time and $u \in [0,1]$ the achieved relative utility.

Two additional considerations are specifically noteworthy in the context of power adaptivity. First, the sequence of software states traversed by a software is assumed to be deterministic for a given

software configuration (i.e., there is no intentional randomness and the computation is error-free). Changing the software configuration will influence the sequence of software states however. This is formalized in state transitions (5.17) and (5.18).

$$s \xrightarrow{S_C} s^* \tag{5.17}$$

$$s \xrightarrow{S_C^*} s^{**} \tag{5.18}$$

with s denoting an initial software state, $S_C \neq S_C^*$ two distinct software configurations, and $s^* \neq s^{**}$ two distinct successor software states. Definition 5.7 defines the term software configuration as used in this work.

Defintion 5.7: Software configuration

A software configuration S_C are all settings influencing a software in a way such as to modify its traversed software states for at least one instance of input data.

Second, the point in time t at which a certain software state s is assumed also depends on the computing resources committed to processing the respective software. Committing a higher or lower amount of computing resources may influence the software utility, depending on the utilization of resources and required response time of the regarded software.

The utility received from a given software can therefore be influenced in two ways: either by changing the software states that are being traversed, or by changing the amount of resources committed to software processing. While the latter case can be performed obliviously of the actual software being executed, the former case requires application-specific knowledge. It is also noteworthy that altering the software configuration will generally lead to an output differing from the one obtained using the original software configuration. This may or may not be acceptable, depending on the utility function. This is investigated in the following.

A binary utility function is a utility function that either delivers a result perfectly matching the requirements or no useful output at all. This is formally defined in Definition 5.8.

Defintion 5.8: Binary utility function

A binary utility function is a utility function that takes only values $u \in \{0, 1\}$.

A service exhibiting a binary utility function is undesirable for adaptation purposes, as it cannot be gracefully scaled. Assuming that a service may not be shut down, it thus degrades to an unadaptable load causing a utilization which can only be forecasted probabilistically. For service resource utilization adaptation to be feasible, it is required for a service to provide intermediate *setpoints* of utility/resource usage.

Assuming **Requirement 2** (cf. Section 5.4.2) holds, for a general service an arbitrary number of adaptation dimensions need to be considered. More formally, different software configurations are compared regarding their effect on resource utilization and a target metric (i.e., a utility value u or an application specific quality of service (QoS) or QoE metric). These software configurations are henceforth referred to as E-states (cf. Definition 5.9).

Defintion 5.9: E-state

An E-state is a 3-tuple $E = (S_C, u, R)$ mapping a valid software configuration S_C to a respective utility value u and resource utilization R. Let E_s further denote the set of all E-states of a software s.

This naming is chosen in accordance to the advanced configuration and power interface (ACPI) nomenclature[72]. The ACPI standard, at the time of writing, declares the following states for various hardware components of ICT devices:

- **G-states**: global system states,
- **D-states**: device power states,
- **S-states**: sleeping and soft-off states,
- **C-states**: processor power states, and
- **P-states**: device and processor performance states.

To further enhance granularity, the adaptation of individual software may introduce a yet more fine-grained way of power control. In contrast to existing ACPI power states, any combination of software E-states can be run under any P-state. A possible integration of E-states into the existing ACPI power states is depicted in Figure 5.3.

[72] `http://uefi.org/sites/default/files/resources/ACPI_6_1.pdf`, accessed 12.03.2018

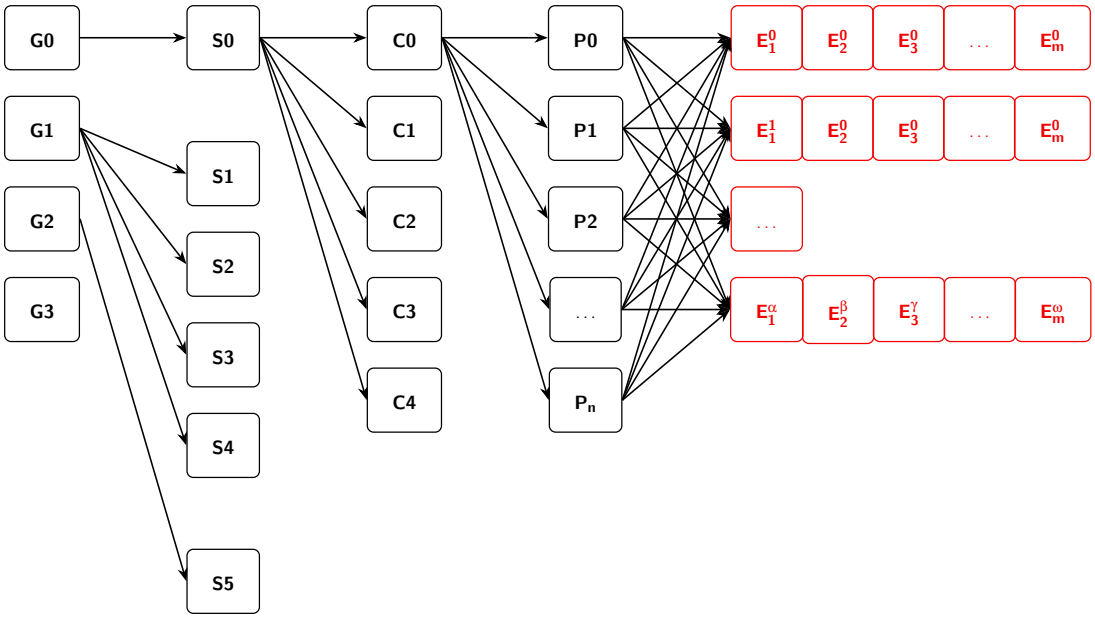


Figure 5.3: Possible integration of E-states with the existing ACPI power states.

The projection of a software's E-states onto their u dimension is generally neither surjective nor injective, i.e., not for every utility value a corresponding E-state exists, however there may be multiple E-states exhibiting the same utility value. It is assumed that an E-state capable application has information on its available operating points, i.e., the combinations of resource utilization in n dimensions and resulting utility. In Chapter 4, a possibility to map resource utilization to power demand was presented. Using this method, each application operating point can be assigned a resulting power demand on a given substrate hardware. This in turn allows for a projection of a given E-state onto two dimensions: power and utility[73]. As all software scalability dimensions can be encoded in a single dimension (S_C), the power-performance characteristics can be captured on a single power-QoE plane. This way, the E-states can be traversed in a way such as to optimize for different goals. Three of these goals are presented in the following, starting with a *power-optimal E-state sequence* described in Definition 5.10.

[73] As in the following a video encoding use case is considered, QoE will be used as a target metric. It may be converted to a utility value u by mapping it onto the range of $[0, 1]$.

Defintion 5.10: Power-optimal E-state sequence

A power-optimal E-state sequence of a software service s is a monotonically decreasing sequence of E-states by power demand $\left\{E_s^0, E_s^1, \ldots, E_s^n\right\}$. For each E-state E_s^x, $x \in \{1, 2, \ldots, n-1\}$ the following holds: $P(E_s^{x-1}) \geq P(E_s^x) \geq P(E_s^{x+1})$. A power-optimal E-state sequence contains all E-states available in the respective software service.

Ordering the E-states of a software in a power-optimal way creates a traversable sequence that guarantees a reduction in power demand in each subsequent E-state. However, accounting for QoE reveals a possible downside of this ordering: strictly going by power demand may cause strong fluctuations in QoE. Definition 5.11 describes a QoE-driven approach to E-state ordering.

Defintion 5.11: QoE-optimal E-state sequence

A QoE-optimal E-state sequence of a software service s is a monotonically decreasing sequence of E-states by QoE value $\left\{E_s^0, E_s^1, \ldots, E_s^n\right\}$. For each E-state E_s^x, $x \in \{1, 2, \ldots, n-1\}$ the following holds: $QoE(E_s^{x-1}) \geq QoE(E_s^x) \geq QoE(E_s^{x+1})$. A QoE-optimal E-state sequence contains all E-states available in the respective software service.

Similarly, Definition 5.11 leads to a traversable sequence by decreasing QoE. Ignoring power demand, the available E-states may be ordered according to a decreasing QoE value. This way, a QoE-optimal adaptation sequence can be assigned as depicted in Figure 5.4.

While a steady quality degradation is desirable, disregarding power may lead to a strong fluctuation in power demand, even containing E-states with undesirable QoE/power ratios (i.e., decreasing in QoE but increasing in terms of power demand). To arrive at a balanced sequence of E-states, an ordering that is both power- and QoE-aware is required. Definition 5.12 gives a formal description of this ordering.

Defintion 5.12: Power and QoE-aware E-state sequence

A power- and QoE-aware E-state sequence of a software service s is a sequence of E-states in descending order by QoE value $\left\{E_s^0, E_s^1, \ldots, E_s^n\right\}$. For each E-state E_s^x, $x \in \{1, 2, \ldots, n-1\}$ the following holds: $P(E_s^{x-1}) > P(E_s^x) > P(E_s^{x+1})$.

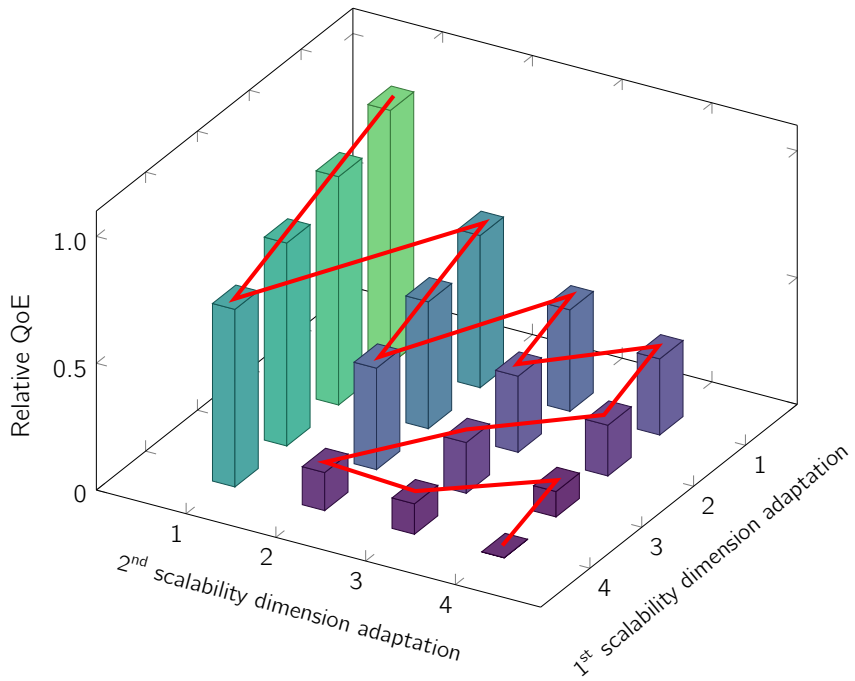


Figure 5.4: An QoE-optimal adaptation sequence.

Intuitively, Definition 5.12 says that a decreasing QoE value in a power- and QoE-aware E-state sequence always has to be accompanied by a decrease in power demand. A power- and QoE-aware E-state sequence is a (not necessarily strict) subset of the available E-states of the respective software. It is generally neither QoE- nor power-optimal, as intermediate E-states may be eliminated due to violating the additional power constraint. E-state sequences according to Definition 5.12 are henceforth referred to as optimal adaptation sequence (OAS). Algorithm 5.1 gives the pseudo-code to derive the OAS from a list of E-states of a software.

It is noteworthy that Algorithm 5.1 assumes E-states that contain both QoE and power information, different from Definition 5.9. Employing the methods described in Chapter 4, a substrate-specific power demand may be derived from the substrate-agnostic resource utilization description of each E-state.

Algorithm 5.1 Create OAS from a list of E-states

Input: An unsorted list of all E-states of a software E_s.

Output: A power and QoE-aware E-state sequence O_s.

```
function PowerQoeAware(E_s)
    //Sort list E_s descending by QoE
    E_s.sortDescending(QoE);
    while E_s ≠ ∅ do
        //Remove first element from E_s and assign it to x
        x ← E_s.remove(E_s.getFirstElement());
        if O_s = ∅ then
            //Add element x to O_s if O_s is empty
            O_s.add(x);
        else
            //Check if x has a lower power demand than the latest addition to O_s
            if (x.getPower() < O_s.getLastElement().getPower()) then
                O_s.add(x);
            end if
        end if
    end while
    Return O_s;
end function
```

Corollary 5.2

An OAS of a software service s is strictly decreasing in both QoE and power demand.

Proof In a first step, assume there exists an E-state E_s^{x+1}, with $QoE(E_s^{x+1}) > QoE(E_s^x)$. This is a contradiction to Definition 5.12 stating that in an OAS E-states are sorted by descending QoE. Second, assume there exists an E-state E_s^{x+1}, with $P(E_s^{x+1}) > P(E_s^x)$. By definition, $QoE(E_s^x) > QoE(E_s^{x+1})$. Therefore, E_s^{x+1} is eliminated from the OAS, as the condition $QoE(E_s^x) > QoE(E_s^{x+1}) \rightarrow P(E_s^x) > P(E_s^{x+1})$ does not hold. ■

Having laid out the formal considerations regarding software-intrinsic power adaptation possibilities, in the following the specific example of flexibilities gained from configuration changes in a live video encoding service are described.

5.5 Video Encoding Flexibilities

In compliance with the running example detailed in Section 5.3, an example for a service providing multi-dimensional scaling capabilities is real-time video encoding. First, the feasibility of a live video encoding service is evaluated. Regarding **Requirement 1** (cf. Section 5.4.2), the online reconfiguration is achievable by providing a certain amount of video buffer. The actual video encoding may be reconfigured with techniques as implemented in, e.g., the MPEG-4 scalable video CODEC. More details on the enabling features within MPEG-4 SVC are given later in this section. **Requirements 2** and **3** (cf. Section 5.4.2) are addressed in Section 5.5.2. The scenario of a live video encoding service is considerably different from a batch or on-demand video encoding, as its instantaneous resource utilization is limited by the rate of input material generation, i.e., the amount of video frames being fed to the encoder. This is depicted in Figure 5.5.

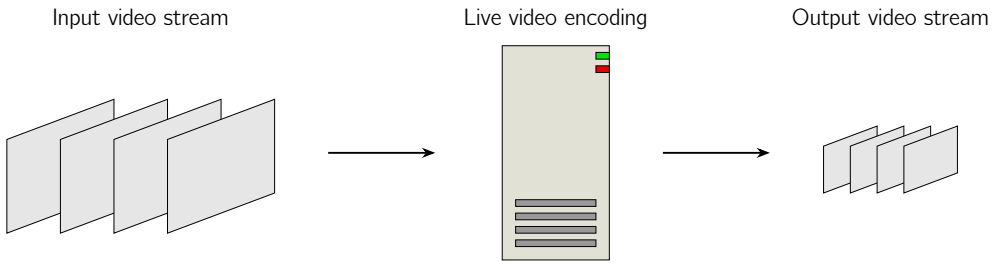


Figure 5.5: A live video encoding scenario.

More formally, this process can be described as a queuing network, featuring one input and two servers. The input queue is filled with a deterministic rate λ, representing the speed at which video material is generated and offered to the video encoder[74]. A server with deterministic service rate[75] μ_E represents the video encoder. Its speed is limited by the processing power of the available CPUs. After encoding, the video frames are forwarded to a second server with deterministic service rate[76] μ_O which represents the video consumer. This is depicted in Figure 5.6.

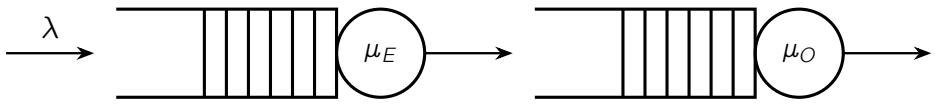


Figure 5.6: Queuing network model of live video encoding.

[74] A fixed frame rate video generation is assumed.
[75] A deterministic service rate of the encoding server assumes a video exhibiting constant encoding complexity. The buffer size is assumed to be large enough to mask intermittent changes to complexity.
[76] Similar assumptions as with the encoding server apply analogously.

The process governing μ_E is assumed to offer a basic form of control logic: in case $\mu_O < \mu_E$, μ_E reduces its service rate (and buffers additional input from λ) to not overflow μ_O's buffer. A similar result is achieved by a value of $\lambda < \mu_E$, resulting in a lowered encode speed of μ_E due to lack of input material. The queuing network parameters are given by the regarded scenario: in case the encoding works on live footage, the input rate λ is generally significantly smaller than the encode server service rate μ_E (cf. Section 5.6). The server power demand is thus limited by the input buffer filling. On the other hand, if there is on-demand video encoding serving a specific request, the output buffer emptying rate μ_O is limited due to video playback speed. Both of these scenarios are suited for a runtime power adaptation, as the amount of presented video time is limited by input or output. Another scenario—offline transcoding—is however not feasible for adaptation: in case the entire video sequence is already available and the output speed is not restricted, both λ and μ_O can be assumed to significantly outperform μ_E. This leads to a full utilization of CPU resources, independent of video encoder configuration. Although, as is shown in Section 5.5.1.1, λ can be artificially limited, offline transcoding generally has to be assumed to have strict QoE requirements (e.g., for archival purposes). Accordingly, the power adaptation capabilities as considered in this section are lost in this scenario.

The considered dimensions are in line with the MPEG4-SVC standard [162]. The most noticeable feature of SVC is, as its name implies, the ability to generate a scalable bitstream using three different scalability types. Schwarz et al. [162] define the term *scalability* in the context of video data as follows:

> "A video bit stream is called scalable when parts of the stream can be removed in a way that the resulting sub-stream forms another valid bit stream for some target decoder, and the sub-stream represents the source content with a reconstruction quality that is less than that of the complete original bit stream but is high when considering the lower quantity of remaining data."

The techniques employed to achieve this are explained in the following. It is noteworthy that the actual encoder used (`x264`) is not SVC compliant. However, the same scalability dimensions are available via an encoder configuration change.

Temporal scalability: Video may be displayed at different temporal resolutions, i.e., a different number of video frames displayed per second while retaining the represented video time. Using a higher temporal resolution increases perceived motion fluidness especially in high motion scenes. Scaling a video along this dimension requires omitting a certain number of video frames in between each frame that is encoded. Figure 5.7 shows possible levels of temporal resolution that may be

extracted from a high frame rate source material. Depending on the target frame rate, only certain frames present in the input video are encoded.

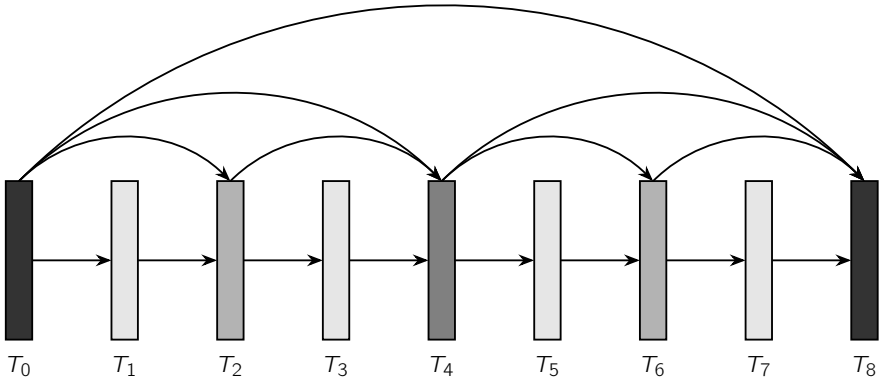


Figure 5.7: Example of video encoding at different temporal resolutions.

The temporal decimation may be chosen by only encoding video frames of number $T_n \equiv 0 \mod 2^m$, with $m, n \in \mathbb{N}_0$. For example, to encode at full temporal resolution, m is chosen equal to 0, resulting in all frames being encoded (as $2^0 = 1$, and $T_n \equiv 0 \mod 1$ for all $n \in \mathbb{N}_0$). To encode only every 4^{th} video frame, m is chosen equal to 2, and so on. Although the number of encoded video frames decreases with a lower temporal resolution, it has to be considered that generally, the average difference between temporally adjacent video frames increases. Therefore, the complexity of motion estimation may increase, leading to reduced savings in power demand. The maximum value of m depends on limitations of the chosen video CODEC. In the following, it is assumed that the requested temporal decimation does not exceed the CODEC's capabilities.

Spatial scalability: Spatial scalability refers to flexibility in choosing video resolution. Spatial scalability enables scaling the resolution of a video sequence, which can be used to match the resolution requirements of an output device or to (indirectly) decrease bandwidth requirements. Additionally, reducing video resolution severely decreases the resource utilization of video encoding, even when accounting for the overhead introduced by rescaling the video from its native resolution prior to encoding. Higher resolutions require a significantly increased motion estimation effort, as detailed in Section 2.3. To achieve a consistent experience for the viewer, the spatial dimensions need to be restored on client side, introducing another small overhead. Figure 5.8 depicts the process of encoding different spatial resolutions.

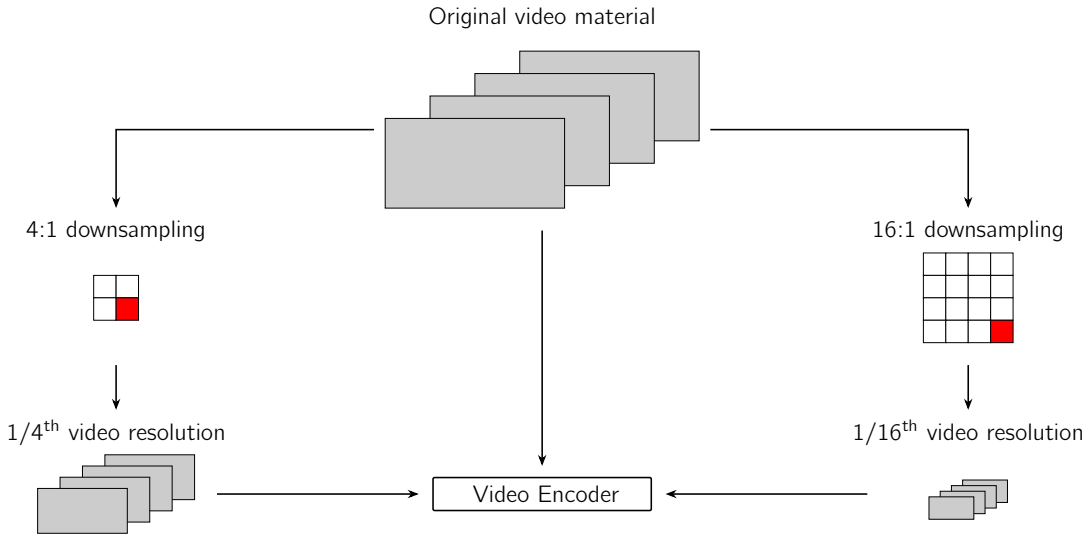


Figure 5.8: Example of video encoding at different spatial resolutions.

Quality scalability: A third scalability technique available is quality scalability, which means that a video stream may be encoded at different bit rates, i.e., quantizer levels. Higher quantizer levels result in a loss of video quality, especially in high frequency image content, i.e., omitting fine details. Changing video stream quantization is the primary option for adjusting bandwidth requirements. The quantization value is a factor Q applied to the values of a quantization matrix. Table 5.1 shows an example quantization matrix of an MPEG-2 encoder. For more detailed information on video quantization, the reader is referred to the MPEG website[77].

8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

Table 5.1: Example quantization matrix of an MPEG-2 encoder.

By increasing the value of Q, the effectively applied values of the quantization matrix are increased. This leads to a loss of high frequency data in video images. By offering video streams at varying quantizer values Q, a trade-off can be realized.

[77] `http://www.mpeg.org/MPEG/MSSG/tm5/Ch7/Ch7.html`, accessed 10.10.2018.

The different scalability options for video encoding may be applied individually or in combination. This way, a wide variety of combined scalability options is available. The total number of available configurations (assuming there are no invalid configuration combinations) is given in Expression (5.19).

$$\prod_{i=1}^{N} c_i, \tag{5.19}$$

with N denoting the total number of scalability dimensions and c_i the number of configurations available in dimension i. The utility obtained by human consumers from a live video encoding service can be captured in their QoE, which is defined by Le Callet et al. [96] as given in Definition 5.13. This definition was also adopted by Recommendation ITU-T P.10/G.100 [83].

Defintion 5.13: Quality of Experience

Quality of Experience (QoE) is the degree of delight or annoyance of the user of an application or service. It results from the fulfillment of his or her expectations with respect to the utility and/or enjoyment of the application or service in the light of the users personality and current state.

If imposing resource limitations on a video encoding process causes the encoding server to become the limiting factor of the queuing network (cf. Figure 5.6), this will lead to a drop in encoding speed. In case the encoder is not able to retain an encoding frame rate that is higher than the playback frame rate, a large decrease in QoE will occur. This is elaborated on more in Section 5.5.2. This sudden loss of QoE is especially apparent in video streaming applications. This use case is becoming increasingly relevant as video on demand (VoD) platforms are gaining momentum (e.g., Netflix and Amazon Prime Video). Clients demand a continuous video stream in high quality. Failing to deliver upcoming video frames on time will cause the video to exhibit stuttering or even complete standstill. For video streaming, technologies to avoid video standstill due to bandwidth scarcity exist since several years. However, most of these technologies are aimed at bandwidth limitations on client side. Power restrictions, especially on server side are hardly discussed. Relevant existing work in this area is covered in Section 2.3.

5.5.1 Test Setup

To test different software configurations in a use case in line with the running example presented in Section 5.3, a video file is encoded using the `x264`[78] encoder at different settings. The tests are performed on a HPe Server with specifications as listed in Table 5.2.

Component	Specification
CPU	2 × 2.20 GHz Intel® Xeon® CPU E5-2630 v4
RAM	128 GB DIMM DDR4 Synchronous Registered (Buffered) 2400 MHz
HDD	2 × Samsung SM863a 2.5" 480 GB SSD (RAID 1)
OS	Linux Ubuntu 17.04 (Kernel version 4.10.0-35-generic)

Table 5.2: Specifications of relevant server components used in test environment.

The DVFS settings are managed by the `intel_pstate` CPU performance scaling driver in active mode with HWP configuration[79] (i.e., selection of frequency/voltage operating points of each CPU core is done by the CPU's internal logic). In the following, the three phases of the conducted tests are covered in more detail.

5.5.1.1 Power Measurement & Encoding

Figure 5.9 shows the physical test setup, enabling a measurement of power demand during the encoding process.

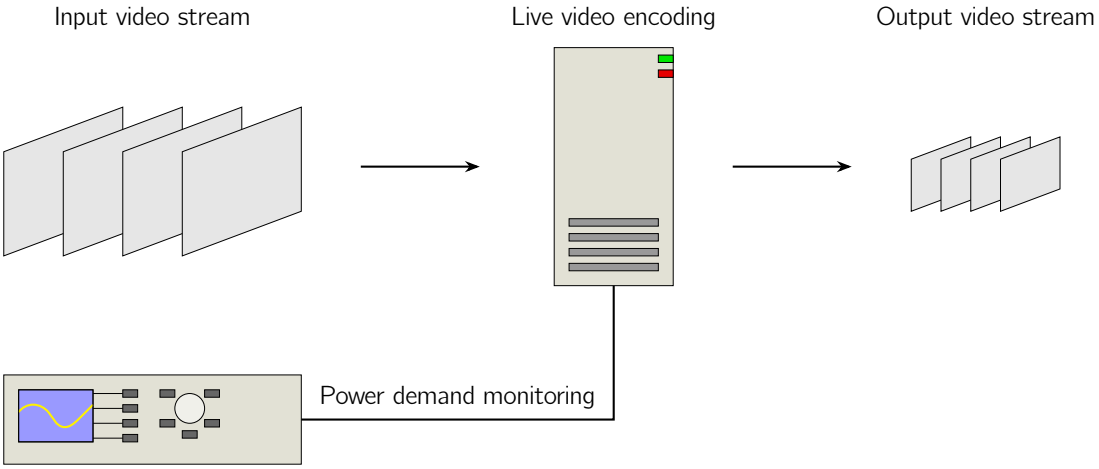


Figure 5.9: Test setup during the encoding process.

[78] `https://www.videolan.org/developers/x264.html`, accessed 05.06.2018.
[79] `https://www.kernel.org/doc/html/v4.14/admin-guide/pm/intel_pstate.html`, accessed 10.07.2018.

From a high quality input video[80], several versions with lower temporal and spatial resolutions are created and encoded at different bit rates (see Table 5.3). The exhaustive permutation of all varied encoding parameters leads to 36 different encoding configurations (three resolutions, three frame rates, and four bit rates). Command Line 5.1 shows the command used to encode the video files.

```
ffmpeg -r 30 -i %input% -an -sn -s %r_o% -filter:v "fps=%fr_o%"
-c:v libx264 -preset veryfast -pix_fmt yuv420p -tune ssim
-crf %crf% %output%
```

Command Line 5.1: Encoder command line arguments.

In Command Line 5.1, `%input%` denotes the input file name, `%r_o%` the target resolution, `%fr_o%` the target frame rate, `%crf%` the target constant rate factor (CRF), and `%output%` the output file name. In this configuration, the encoder reads the input file at its native playback speed (i.e., 30 frames per second), emulating a live encoding scenario.

Simultaneously to the video encoding, the power demand of the server under test is recorded. The measurement is conducted at the PSU AC level. Both redundant power supplies are connected to a single measurement point using a Y-adapter. The power measurement device used is a ZES Zimmer LMG500[81], featuring a measurement accuracy of 0.015 % of reading + 0.01 % of range at 45 Hz to 65 Hz. Power demand is sampled every 100 ms via an RS-232 serial connection to the power meter.

5.5.1.2 Decoding Process

To allow for a calculation of the achieved QoE in the `x264` encoded video file created in the previous encoding step, it has to match the reference file's spatial and temporal dimensions. To this end, in a first step the file has to be decoded. In the following, the video's original spatial and temporal resolutions are restored via image upscaling and frame repetition, respectively. Figure 5.10 illustrates the process.

Command Line 5.2 shows the required command to decode a `x264`-encoded file and adjust its spatial and temporal dimensions. As decoding and upscaling/filtering is in practice performed on client side, its power demand is not considered in this test.

[80] `http://distribution.bbb3d.renderfarming.net/video/mp4/bbb_sunflower_2160p_30fps_normal.mp4`, accessed 20.10.2017.
[81] `https://www.zes.com/en/Products/Discontinued-Products/Energy-and-Power-Meters/LMG500`, accessed 11.12.2018.

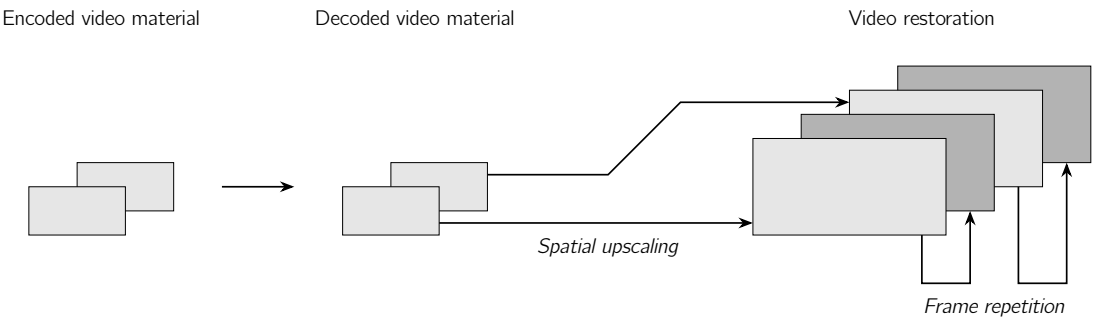


Figure 5.10: Steps in the video decoding process.

```
ffmpeg -r %fr_i% -i %input% -an -sn -s %r_r% -filter:v
"fps=%fr_r%" -c:v libx264 -pix_fmt yuv420p -crf 0 %output%
```

Command Line 5.2: Decoder command line arguments.

In Command Line 5.2, `%fr_i%` denotes the input frame rate, `%input%` the input file name, `%r_r%` the reference resolution, `%fr_r%` the reference frame rate, and `%output%` the output file name.

5.5.1.3 Quality Comparison

After decoding the video file and restoring it to the reference video's spatial and temporal dimensions, visual quality comparison is possible. To adequately reflect the expected QoE of a user, the structural similarity index (SSIM) [177] metric is chosen to evaluate performance. Wang et al. [178] have shown that SSIM scores show high correlation to perceived video quality. The `ffmpeg` software features an integrated SSIM calculation tool, invoked as depicted in Command Line 5.3.

```
ffmpeg -r %fr_i% -i %input% -r %fr_r% -i %reference% -an -sn
-lavfi ssim -f null -
```

Command Line 5.3: Y-SSIM calculation command line arguments.

In Command Line 5.3, `%fr_i%` denotes the input frame rate, `%input%` the input video file name, `%fr_r%` the reference frame rate, and `%reference%` the reference video file name.

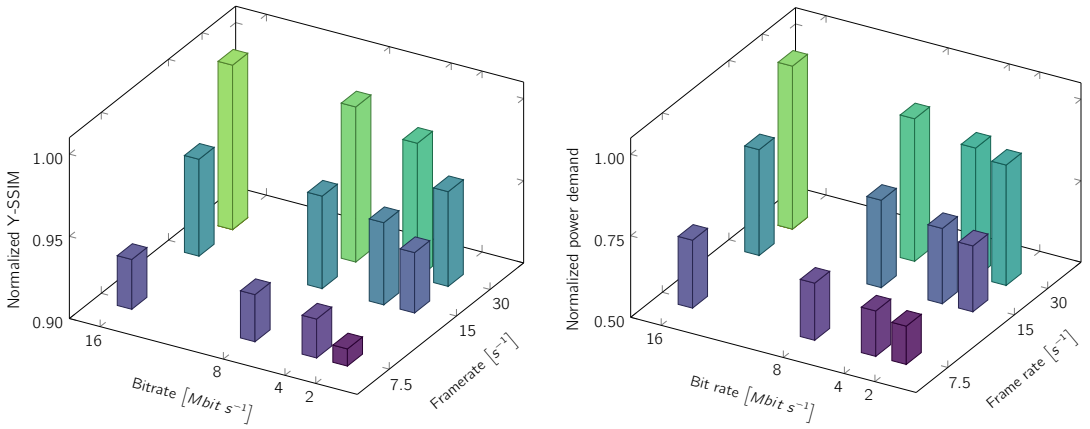
5.5.2 Evaluation

In the following, the evaluation of the tests described in Section 5.5.1 is given. To enhance readability, most results are given both in textual and graphical form. Table 5.3 shows the results of different software configurations regarding both QoE and power demand.

Resolution	Frame rate [s^{-1}]	Bit rate [MBit s^{-1}]	Y-SSIM	Avg. Power [W]
3840 × 2160	30.0	16.0	0.992 890	167.3388
3840 × 2160	30.0	8.0	0.987 267	163.1413
3840 × 2160	30.0	4.0	0.975 133	160.4277
3840 × 2160	30.0	2.0	0.950 796	158.5884
3840 × 2160	15.0	16.0	0.952 099	155.5852
3840 × 2160	15.0	8.0	0.949 442	151.8155
3840 × 2160	15.0	4.0	0.943 218	149.3283
3840 × 2160	15.0	2.0	0.930 039	147.3976
3840 × 2160	7.5	16.0	0.923 675	147.8156
3840 × 2160	7.5	8.0	0.922 056	145.6078
3840 × 2160	7.5	4.0	0.917 072	143.1963
3840 × 2160	7.5	2.0	0.903 960	141.7314
1920 × 1080	30.0	16.0	0.984 152	146.9816
1920 × 1080	30.0	8.0	0.980 764	144.5111
1920 × 1080	30.0	4.0	0.974 538	142.2500
1920 × 1080	30.0	2.0	0.963 152	139.7826
1920 × 1080	15.0	16.0	0.945 968	141.7881
1920 × 1080	15.0	8.0	0.944 043	140.5758
1920 × 1080	15.0	4.0	0.940 550	139.0529
1920 × 1080	15.0	2.0	0.934 068	137.4123
1920 × 1080	7.5	16.0	0.919 303	138.6157
1920 × 1080	7.5	8.0	0.918 161	137.6811
1920 × 1080	7.5	4.0	0.916 129	136.5994
1920 × 1080	7.5	2.0	0.912 280	135.5597
960 × 540	30.0	16.0	0.950 797	138.6451
960 × 540	30.0	8.0	0.948 810	137.8645
960 × 540	30.0	4.0	0.945 278	137.4209
960 × 540	30.0	2.0	0.939 114	136.9351
960 × 540	15.0	16.0	0.920 359	136.4688
960 × 540	15.0	8.0	0.919 434	135.8783
960 × 540	15.0	4.0	0.917 472	135.5808
960 × 540	15.0	2.0	0.913 848	135.1765
960 × 540	7.5	16.0	0.898 318	135.1908
960 × 540	7.5	8.0	0.898 253	134.9869
960 × 540	7.5	4.0	0.897 116	134.4766
960 × 540	7.5	2.0	0.894 794	134.3994

Table 5.3: Comparison of different software configurations in power and QoE.

Figures 5.11a and 5.11b show the resulting normalized Y-SSIM and power demand values for different video encoder configurations in 2160p resolution, respectively.

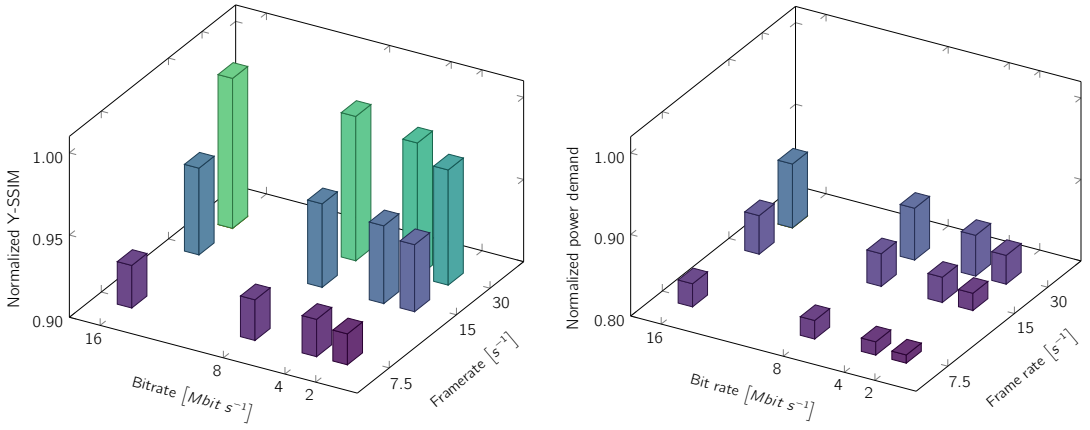


(a) Normalized Y-SSIM **(b)** Normalized power demand

Figure 5.11: Results for different video encoder configurations in 2160p resolution.

Considering the different axis scales, power demand exhibits a strong linear relationship to Y-SSIM while keeping a constant resolution of 2160p. The Pearson product-moment correlation coefficient of the two data sets is 0.9723.

Similarly, Figures 5.12a and 5.12b show the resulting normalized Y-SSIM and power demand values for different video encoder configurations in 1080p resolution, respectively.

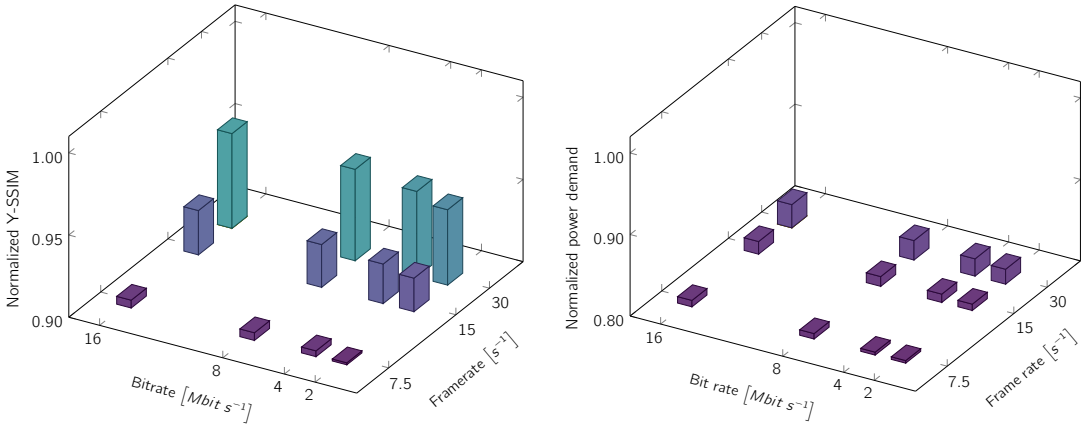


(a) Normalized Y-SSIM **(b)** Normalized power demand

Figure 5.12: Results for different video encoder configurations in 1080p resolution.

While the influence on perceived visual quality is limited, the reduction of encoding power demand is obvious. Power is on average reduced by 8.25 %, while on the same dataset, SSIM is only lowered by 0.13 %.

Figures 5.13a and 5.13b depict the respective normalized Y-SSIM and power demand values for different video encoder configurations in 540p resolution.



(a) Normalized Y-SSIM

(b) Normalized power demand

Figure 5.13: Results for different video encoder configurations in 540p resolution.

The power/QoE tradeoff is less favorable compared to the transition from 2160p to 1080p resolution. The average power decrease is 2.84 %, SSIM is however similarly decreased by 2.55 %.

By combining the three scalability dimensions (resolution, frame rate, bit rate) in one dimension (the *software configuration* S_C), all results obtained in the tests can be visualized in a double axis plot on a single plane. This is depicted in Figure 5.14.

From Figure 5.14 the OAS may now be derived. This is done in three steps:

1. Elimination of invalid software configurations.
2. Sorting the remaining software configurations in terms of descending QoE.
3. Apply Algorithm 5.1, i.e., keep only those software configurations for which the following holds: there exist no software configurations with higher or equal QoE and at the same time lower or equal power demand.

These generic steps are now formulated more precisely with regard to the running example. In a live video encoding scenario, the previously mentioned steps could be realized in the following way:

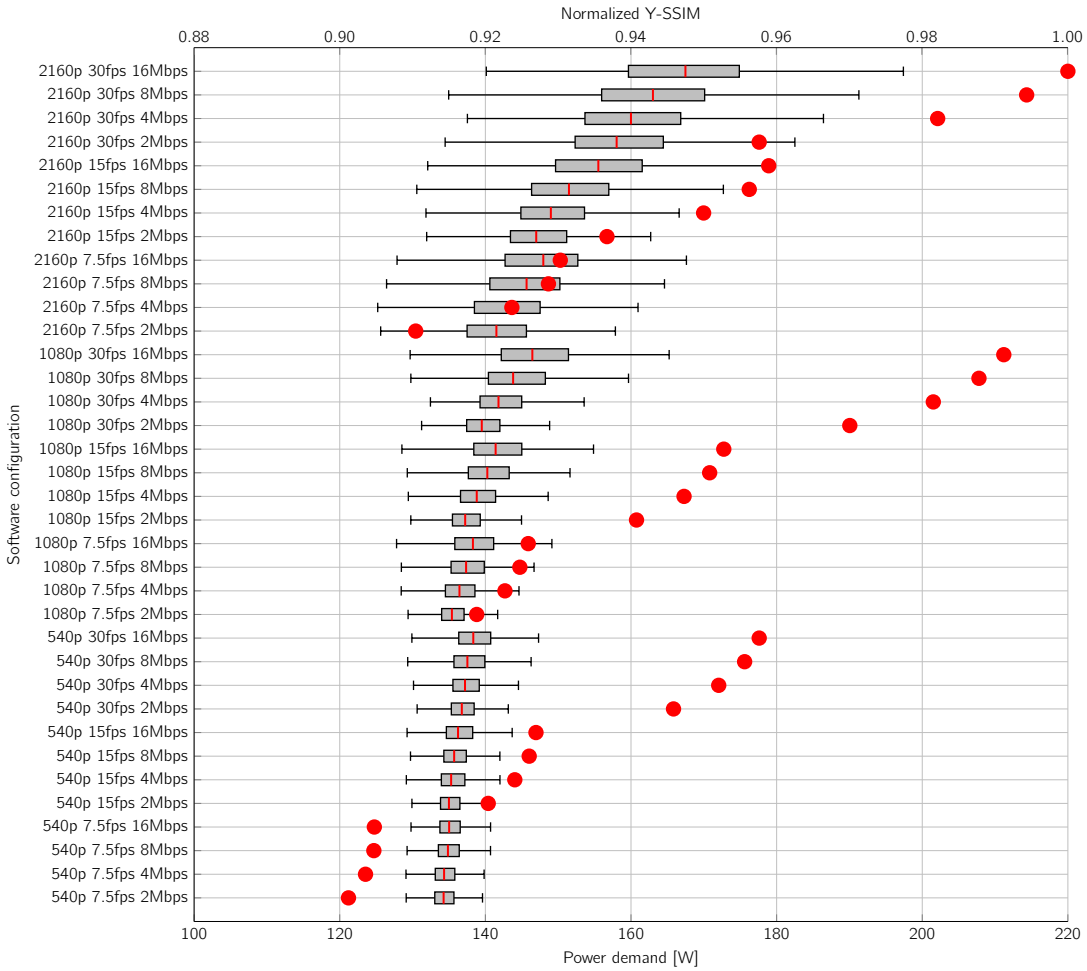


Figure 5.14: QoE and power demand results for different video encoder configurations.

1. Invalid software configurations of video encoders could, e.g., consist of configurations which create content at bit rates too high for either the data channel to transport or the client device to process properly. In this case, configurations with bit rates exceeding limitations need to be removed. In the running example, it is assumed that all bit rates up to 16 $\mathrm{Mbit\,s^{-1}}$ are within the capabilities of both the transmission channel and playback device. In case this assumption does not hold, all E-states that exceed the maximum bit rate capabilities of either the transmission channel or the playback device need to be removed prior to proceeding.

2. Perceived QoE in digital video streaming scenarios may be evaluated using subjective (e.g., mean opinion score (MOS)) or objective (e.g., peak signal-to-noise ratio (PSNR) or SSIM) measures. Subjective quality ratings obtained from trials have been shown to produce re-

peatable results in case tests are carefully designed and proper instructions are given to participants [74].

Huynh-Thu et al. [74] state that "[s]ubjective quality assessments are still today the most accurate way to measure perceived quality, but they are costly and very complex to implement". Thus, it is desirable to implement a less complex method of QoE evaluation that still reflects perceived quality with high accuracy. To this end, objective metrics like VQM or SSIM were developed which show high correlation to subjective QoE scores [196]. It is possible to map values between subjective and objective metrics: Zinner et al. [196] derived a mapping of MOS to SSIM which is shown in Table 5.4.

MOS	SSIM range
5 (excellent)	[0.99, 1.00]
4 (good)	[0.95, 0.99[
3 (fair)	[0.88, 0.95[
2 (poor)	[0.50, 0.88[
1 (bad)	[0.00, 0.50[

Table 5.4: Mapping of subjective (MOS) to objective (SSIM) quality scores.

3. Applying Algorithm 5.1 to a live video encoding scenario may be performed in the following way: first, the software E-state that causes the lowest power demand is removed from the input. This E-state is added to the OAS. For all following E-states, in ascending order with regard to their power demand, their Y-SSIM is compared to the element with highest Y-SSIM already in the OAS. If the Y-SSIM of the current element is higher, it is added to the OAS, otherwise it is omitted.

After applying the steps mentioned above to the video encodings performed in Section 5.5.1.1, 17 E-states remain in the OAS. Figure 5.15 shows all software configurations, sorted by achieved QoE. Additionally, the normalized Y-SSIM values are depicted: all E-states that are part of the OAS are marked with a red circle, the omitted ones with a black triangle.

5.5.3 Interpretation of Results

In the following, a more in-depth analysis and interpretation of the obtained results are given. The results in each considered scalability dimension are discussed separately. In the following figures,

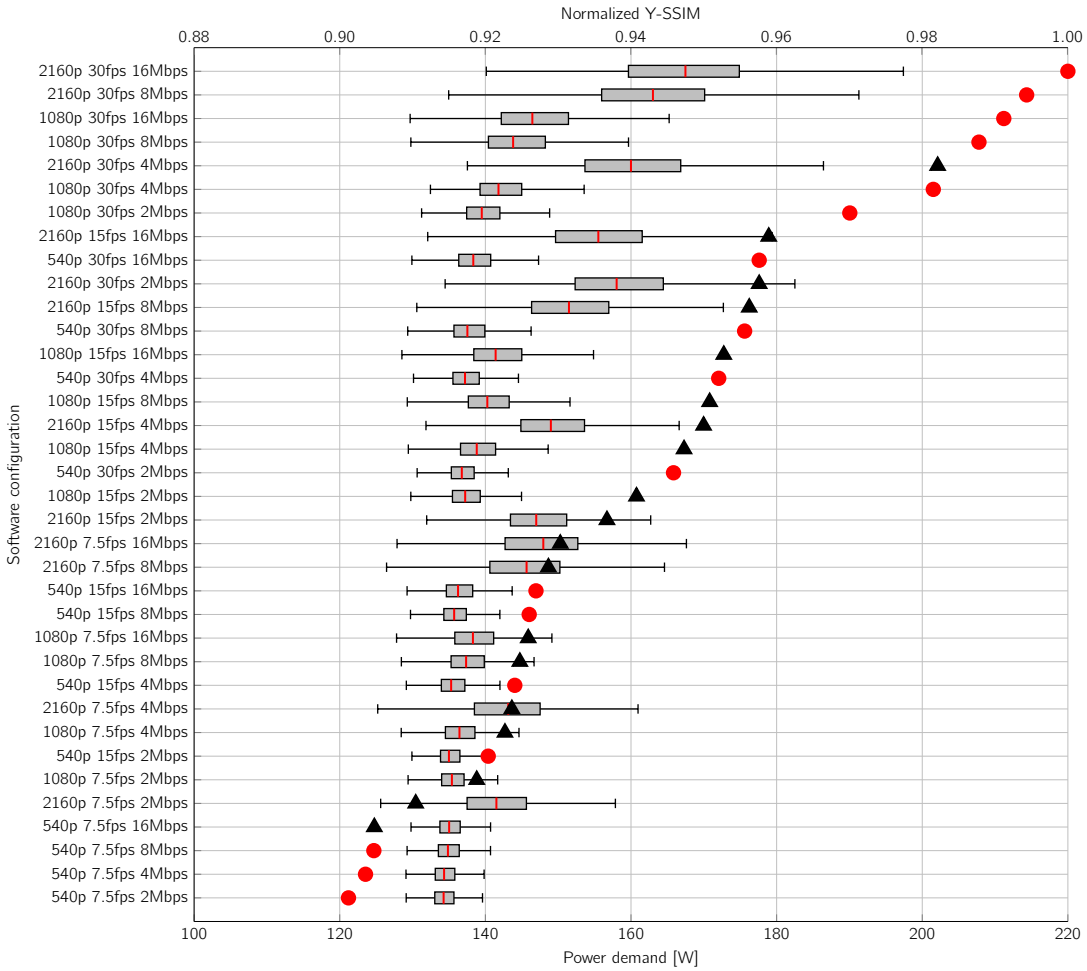


Figure 5.15: E-state OAS classification.

the markers show actual data collected in the conducted experiments, while the lines show the respective linear regression plots.

All investigated software configurations show similar intermittent minimal power demands. This is due to the chosen live encoding scenario, feeding frames to the encoder with a limited rate, resulting in periodic idle states of the CPU while waiting for new input frames. Maximum power demand, in contrast, largely depends on the chosen encoder configuration: each frame fed to the encoder causes a CPU utilization that mostly depends on chosen resolution. This is to be expected according to the findings of Zrida et al. [197], stating that motion estimation is the primary contributor to encoder CPU utilization (cf. Section 2.3). In fact, power demand changes linearly with video frame pixel count: a linear regression fitting results in a R^2 value of 0.9982. It

is noteworthy that video frame pixel count is quadrupled when rescaling from 540p to 1080p or 1080p to 2160p, respectively. Figure 5.16 shows the average normalized power demand and Y-SSIM values achieved among all software configurations sharing a common resolution. Compared to power demand, loss of QoE is much more graceful, especially between 2160p and 1080p resolutions.

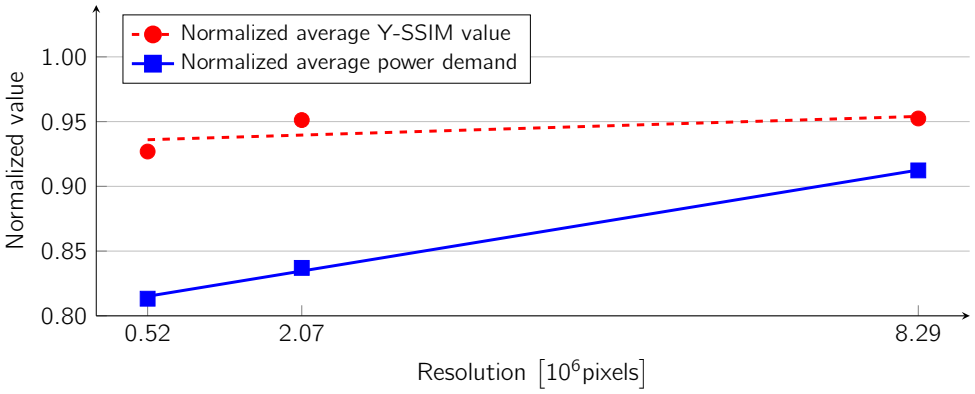


Figure 5.16: Results in the resolution dimension.

In a similar way, average results for scaling in the temporal dimension are obtained. It can be observed that power demand and QoE are decreasing at a similar pace, the Pearson product-moment correlation coefficient of the power demand and QoE data set is very high at 0.9978. Assuming the linear regression captures in-between power and QoE values correctly, even a minor reduction of frame rate significantly reduces Y-SSIM values. This is in contrast to spatial scaling and makes temporal scaling less favorable especially if only minor adaptation is required. Figure 5.17 shows the average normalized power demand and Y-SSIM values achieved among all software configurations sharing a common frame rate.

The results in the bit rate dimension show an increasingly negative impact on visual quality with lower bit rate values. In high bit rate ranges of more than 8 MBit, adapting in this dimension is favorable, as in this region the normalized Y-SSIM decrease is only 0.002164. Results obtained with 2160p resolution show an extraordinarily large QoE degradation at lower bit rates. This is due to extensive compression artifacts, as low bit rates are insufficient to retain a high image quality in high resolutions. Figure 5.18 shows the average normalized power demand and Y-SSIM values achieved among all software configurations sharing a common bit rate.

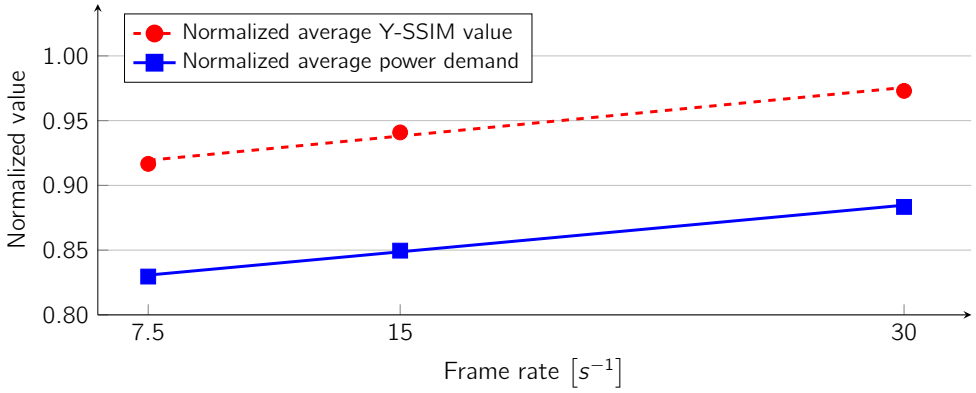


Figure 5.17: Results in the frame rate dimension.

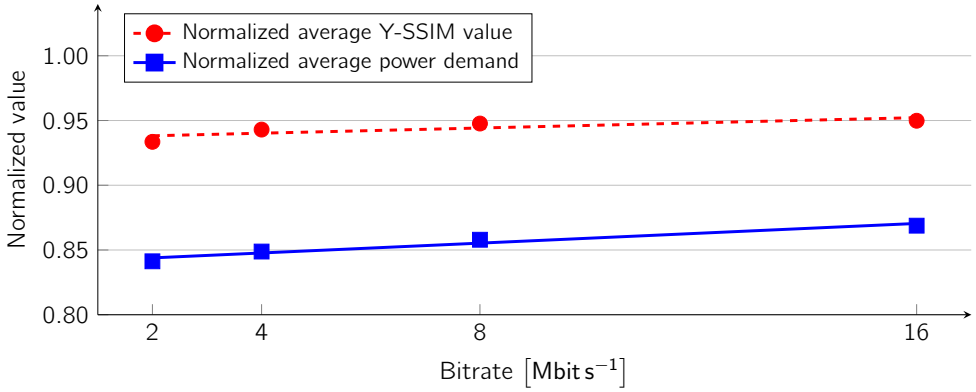


Figure 5.18: Results in the bit rate dimension.

5.6 Adaptation Quality Comparison

In the following, the quality of software-extrinsic and software-intrinsic adaptation techniques is compared. Software-extrinsic adaptations summarize all those adaptation techniques which enforce a maximum resource utilization on an application without altering its configuration. These include adaptations on all layers $\Psi^{-1}(L_{NS})$ for non-virtualized software and $\Psi^{-1}(L_{VS})$ for virtualized software, respectively (cf. Section 5.4). In contrast, software-intrinsic adaptation is a software configuration change leading to a change in power demand by reducing the application's resource utilization (while not limiting resource availability to any other application, even if hosted on the same substrate hardware). In the following, Hypothesis 5.1 is investigated.

Hypothesis 5.1

Adapting service quality in an application-intrinsic way leads to an improved power/QoE tradeoff compared to an application-extrinsic adaptation for certain service scenarios.

Limiting the CPU resources provided to a video encoding process extrinsically leads to a decreased encoding speed, reducing the average frame rate of the video output (if the encoding server becomes the limiting factor in the queuing network (cf. Figure 5.6)). It is irrelevant if this limitation is enforced on hardware, OS, VM or software layer, as long as it is imposed in an application-agnostic way. Depending on the further processing of compressed video (i.e., output of the video encoder) after encoding, two cases need to be distinguished.

1. The QoE is observed directly at the encoder's output. This results in a playback of video frames as soon as they are output by the encoder. In this case, the video would be played back at an inconsistent frame rate in case computing resourcing become too scarce.
2. The QoE is observed after video is multiplexed inside an A/V container (e.g., MP4 [80] or Matroska[82]) and played back by a dedicated software. In this case, video is usually paused if there are too little video frames buffered to continue smooth playback. The QoE degradation in such cases was investigated by, e.g., Ahmed et al [2]. They state that "[...] rate of buffering has the most negative impact on user engagement". Apart from *rate of buffering* (i.e., the number of buffering events per minute), their work also investigated the measure *rate of fluctuation* (denoting the number of changes in bit rate per minute), whose influence on user engagement is substantially lower.

Based on these considerations, it is assumed that the Y-SSIM value resulting from a decrease in average frame rate can be considered an upper bound to the actual perceived QoE in either of the two scenarios enumerated above.

In the following, it is additionally assumed that a pointwise linear interpolation of QoE values in the frame rate dimension captures user perceived QoE sufficiently well. In case this assumption does not hold, additional measurement points with a reduced frame rate would need to be captured.

This way, the QoE degradation for extrinsical power adaptation in a live video encoding scenario can be approximated. This approximation may not be exact for all subsets of encoded video frames as both the resource requirements and effect of inconsistent frame rates depend on the source

[82] `https://www.matroska.org/`, accessed 16.10.2018.

video material. Regarding the results of the intrinsic adaptation, all dimensions (frame rate, bit rate, and resolution) share a common feature: the derivatives of the pointwise linear interpolations are strictly monotonically decreasing, i.e., increasing the "effort" in any considered dimension has a decreasing return in perceived quality. It is noteworthy that especially for the frame rate dimension, this feature is less pronounced: a moderate decrease of frame rate already significantly decreases QoE (cf. Figure 5.17 to Figure 5.18). Adapting in the frame rate dimension is therefore the worst choice with respect to maintaining a high QoE (at least in the observed scenario).

In the following, the impact of imposing resource limitations on a live video encoding process is focused on more closely. The constraint is realized on OS level by forcing the video encoder to only utilize a given subset of available CPU cores. Command Line 5.4 shows the exact command used to run `ffmpeg` under resource constraints.

```
taskset -a -c %cores% ffmpeg -r 30 -i %input% -an -sn -s %r_o%
-filter:v "fps=%fr_o%" -c:v libx264 -preset veryfast
-pix_fmt yuv420p -tune ssim -crf %crf% %output%
```

Command Line 5.4: Limiting utilized CPU cores of live video encoding process.

In Command Line 5.4, `%cores%` denotes a subset of CPU cores available to the encoding process, `%input%` the input file name, `%r_o%` the target resolution, `%fr_o%` the target frame rate, `%crf%` the target CRF, and `%output%` the output file name. Command Line 5.4 consolidates all computing activity of the video encoding process on a subset of available CPU cores, limiting both resources and power demand. To evaluate the resulting power demand and Y-SSIM, a test setup similar to the one described in Section 5.5.1.1 is used. While power demand can be directly monitored, the resulting frame rate needs to be inferred from the average final encoding speed, as given in the encoder statistics. The encoding speed results are given in Table 5.5.

As previously established, the average frame rate can be assumed to be an upper bound regarding QoE. The motion estimation routines implemented in the `x264` CODEC feature *early break criteria*, i.e., ways to speed up motion estimation if there is little motion in the input video sequence. This in turn reduces CPU load. In high motion scenes, these features can only be applied less frequently, leading to higher resource utilization. However, high motion video material QoE is affected in a more negative way by reduced frame rates, as argued in Section 5.8.

Assigned CPU cores	Relative encoding speed	Avg. frame rate in test setup $[s^{-1}]$
20	1.000	30.00
19	1.000	30.00
⋮	⋮	⋮
8	1.000	30.00
7	0.982	29.46
6	0.845	25.35
5	0.845	25.35
4	0.730	21.90
3	0.560	16.80
2	0.404	12.12
1	0.213	6.39

Table 5.5: Performance impact of limiting CPU parallelism.

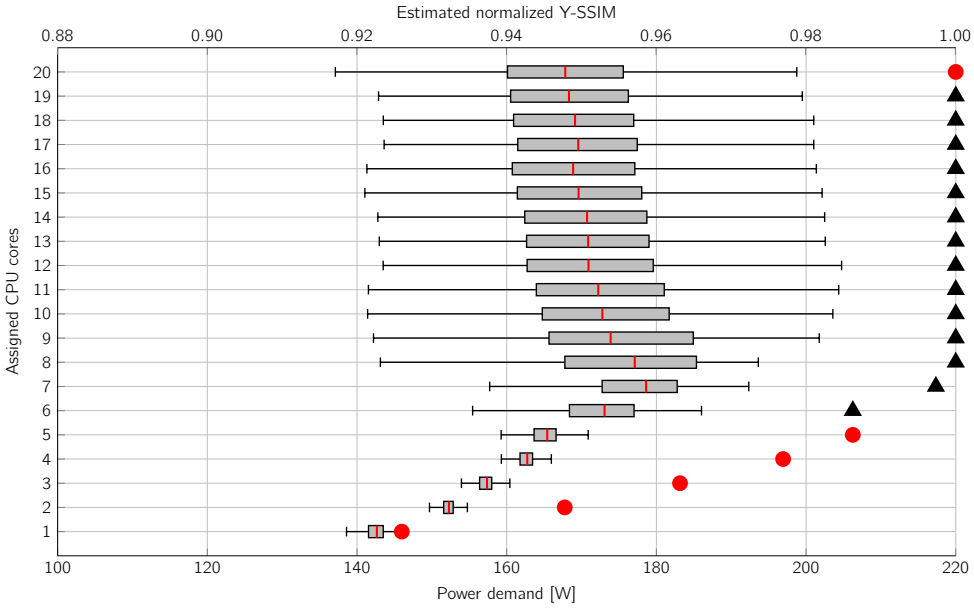


Figure 5.19: Resulting power demand and estimated Y-SSIM for extrinsic power adaptation.

Figure 5.19 shows the results of limiting resource availability to the live video encoding process on OS level. Several peculiarities are apparent in the graph: first, reducing the number of available CPU cores effectively increases power demand until the count is below six. This can be explained by referring to Approximation 4.12: a lower number of available CPU cores increases utilization on these cores. The DVFS driver reacts by increasing the frequency of the available cores. This in turn requires an increase in CPU core voltage, which influences power demand quadratically. Second, an

OAS using extrinsic adaptation only creates six E-state-like operating points, which show clearly inferior results compared to application-intrinsic adaptation. The comparison of both adaptation strategies is shown in Figure 5.20. It should be noted that the measurement data includes server idle power of 89.68 W.

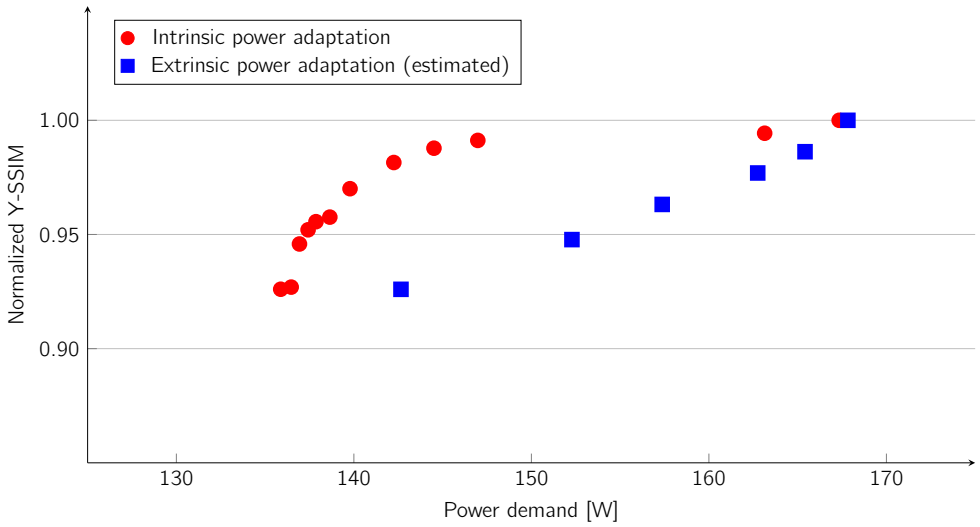


Figure 5.20: OAS of intrinsic and extrinsic power adaptation.

The results depicted in Figure 5.20 show that Hypothesis 5.1 holds for the investigated scenario: application-intrinsic adaptation of resource utilization provides a superior tradeoff of power demand and QoE in comparison to extrinsic power limitation. Considering a scenario in which the QoE is observed directly at the output of the encoder (cf. Section 5.6), the benefit of application-intrinsic adaptation is assumed to decrease with reduced motion intensity in the source video material. In the extreme case of no motion in the source video material, there is no loss of QoE in the application-extrinsic adaptation strategy. For the application-intrinsic case, the OAS would choose the temporal dimension for adaptation first, as it shows the lowest impact on QoE. In case an even lower power demand is requested, other dimensions would be considered in the OAS.

For live video encoding use cases, an interesting question is how to automatically derive an OAS based on information on video complexity in spatial and temporal dimensions. To be feasible, this information needs to be available without relying on subjective user studies, ideally also without dedicated measurements, i.e., based on information which can be provided during encoding (e.g., information such as average motion vector magnitude or high frequency image content). Promising candidate measures to use for an automatic assignment of OASs are, e.g., given in ITU-T Recommendation P.910 [82]:

Spatial perceptual information measurement: To derive spatial perceptual information, ITU-T Recommendation P.910 proposes the use of a Sobel filter [84]. The luminance plane of each video frame at time $t(f_t)$ is passed through a Sobel filter [Sobel(f_t)] in a first step. The Sobel(f_t) operator approximates the horizontal and vertical derivatives of the input images, resulting in an over-emphasis of edges in each video frame. After applying the Sobel filter to each video frame of a scene, the standard deviation over the pixels is calculated for each frame. The spatial information content of the scene is defined by Expression (5.20)

$$max(\forall t \in T : \sigma\,[\text{Sobel}(f_t)]), \tag{5.20}$$

with σ denoting the standard deviation, and $T \in \mathbb{N}$ the number of frames in the regarded video scene.

Temporal perceptual information measurement: According to ITU-T Recommendation P.910 temporal perceptual information is calculated by comparing the luminance value of pixels at the same location in two temporally adjacent frames. The temporal difference between two frames $D_t(m, n)$ may be expressed as shown in Equation (5.21).

$$D_t(m, n) = lum_t(m, n) - lum_{t-1}(m, n), \tag{5.21}$$

with m, n denoting the horizontal and vertical pixel positions respectively, $lum()$ a pixel's luminance value, and t the current frame's time. Similarly to the spatial perceptual information, in a next step the standard deviation for each difference frame D_t is calculated, with D_t defined as shown in Equation (5.22).

$$D_t = \begin{bmatrix} lum_t(1,1) - lum_{t-1}(1,1) & lum_t(2,1) - lum_{t-1}(2,1) & \dots & lum_t(x,1) - lum_{t-1}(x,1) \\ lum_t(1,2) - lum_{t-1}(1,2) & lum_t(2,2) - lum_{t-1}(2,2) & \dots & lum_t(x,2) - lum_{t-1}(x,2) \\ \vdots & \vdots & \ddots & \vdots \\ lum_t(1,y) - lum_{t-1}(1,y) & lum_t(2,y) - lum_{t-1}(2,y) & \dots & lum_t(x,y) - lum_{t-1}(x,y) \end{bmatrix}. \tag{5.22}$$

The value of temporal perceptual information is defined as the maximum over time of the standard deviations of each difference frame D_t, as depicted in Expression (5.23).

$$max(\forall t \in T : \sigma(D_t)), \tag{5.23}$$

with σ denoting the standard deviation, and $T \in \mathbb{N}$ the number of frames in the regarded video scene.

Based on the respective values of spatial and temporal perceptual information, it may be possible to derive preferences regarding the adaptation in either temporal or spatial dimension automatically. It is noteworthy however that continuous changes to the encoder configuration (and in turn fluctuations in the type of visual quality impairments) may lead to a decrease in perceived QoE, as detailed by, e.g., Ahmed et al. [2]. Further studies regarding the feasibility of the suggested approach towards an automatic assignment of an OAS are not performed in the course of this thesis and instead considered a basis for future work.

5.7 Adaptive Video Stream Representation

At the time of writing, a widely adopted standard for delivering adaptive video streams is ISO/IEC 23009 [81]. This standard defines dynamic adaptive streaming over HTTP (DASH), a technique to transfer video data by means of HTTP communication. To realize runtime video adaptation, the video stream is divided into small segments, each available as a separate file. The video can be reconstructed by concatenating the individual files. However, for each video segment, different representations may be created, showing the same time codes of the video file, however at different video encoder settings. A DASH capable video client may then switch video encoder settings by retrieving the respective files. Information on available files may be retrieved by the client in form of a manifest file complying to the XML file standard. The mapping of a power dependent video representation availability may be supported by the optional *<Representation>* subelement *@qualityRanking*, which may be employed to carry information regarding the OAS for a specific video sequence. Code 5.5 shows an example manifest file adapted from [81].

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="urn:mpeg:dash:schema:mpd:2011"
    xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 DASH-MPD.xsd"
    type="static" mediaPresentationDuration="PT3256S" minBufferTime="PT1.2S"
    profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">
<BaseURL>http://cdn1.example.com/</BaseURL>
<Period>
<AdaptationSet mimeType="video/mp4" codecs="avc1.4d0228"
    subsegmentAlignment="true" subsegmentStartsWithSAP="2">
<ContentProtection
    schemeIdUri="urn:uuid:706D6953-656C-5244-4D48-656164657221"/>
<Representation id="6" bandwidth="256000" width="320" height="240">
<BaseURL>8563456473.mp4</BaseURL>
</Representation>
```

```
<Representation id="7" bandwidth="512000" width="320" height="240">
<BaseURL>56363634.mp4</BaseURL>
</Representation>
<Representation id="8" bandwidth="1024000" width="640" height="480">
<BaseURL>562465736.mp4</BaseURL>
</Representation>
<Representation id="9" bandwidth="1536000" width="1280" height="720">
<BaseURL>89045625.mp4</BaseURL>
</Representation>
<Representation id="A" bandwidth="2048000" width="1280" height="720">
<BaseURL>23536745734.mp4</BaseURL>
</Representation>
</AdaptationSet>
</Period>
</MPD>
```

Code 5.5: Example manifest file.

It is noteworthy that the manifest file statically declares available video representations. For the case of a server-side limitation on available video configurations, these limitations need to be dynamically communicated to the DASH video client. To this end, the DASH standard defines *events* to signal aperiodic information. The ISO/IEC 23009 standard defines two possible ways to signal events: inside the manifest file or inline with the video segments. As power adaptation events generally cannot be expected to be known at manifest file creation, an inline event messaging presents the only feasible of the two options. Events can be limited to certain representations of a video stream, therefore only those encoder configurations which are available during power adaptation may be signaled. Figure 5.21 depicts the event insertion principle.

The power adaptation is encoder-centric, thus the event remains valid even for clients streaming video content with a time delay: as the respective video configurations were not created during live encoding, all subsequent views need to switch to an alternate, available representation. The minimum reaction time from the arrival of a power adaptation request to its fulfillment depends on details regarding the video encoder configuration.

1. Preemptive encoder: in case the encoder allows for aborting the currently encoded video segment, the video encoder can immediately restart creating the current segment augmented by the event that signals the reconfiguration. In case the buffer length is high enough to abort and re-encode video segments at any point, the expected delay between power adaptation request and power reduction is approximately the encoding time of one video segment.

2. Non-preemptive encoder: in case the encoder does not allow for aborting the currently encoded video segment, the video encoder has to finish processing the current video segment

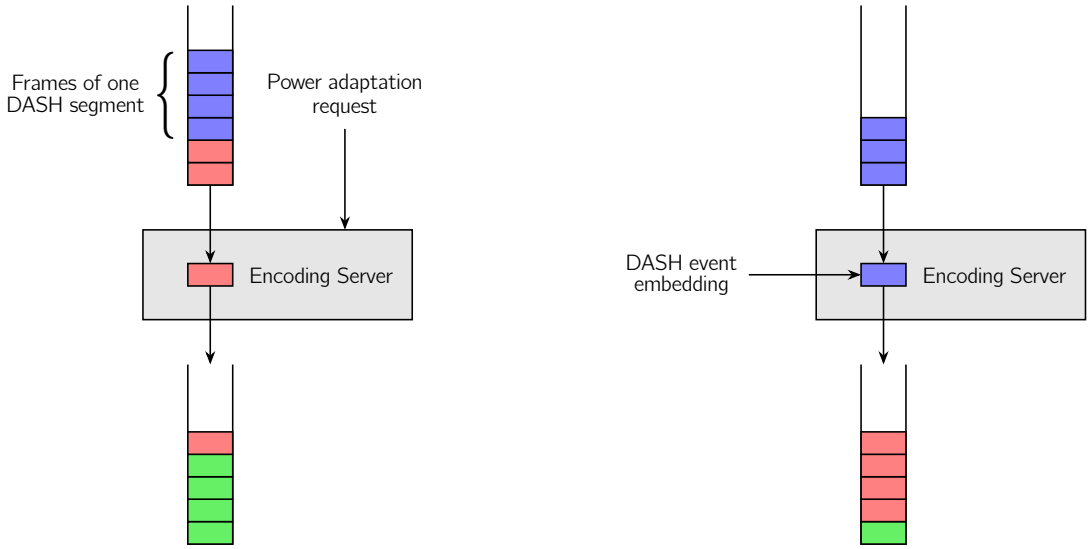


Figure 5.21: Schematic view of DASH event embedding.

before creating the event-augmented segment. Assuming a uniform distribution of power adaptation requests in time, the expected delay between power adaptation request and power reduction is 1.5× the encoding time of one video segment.

Another way of reducing available video representations might be to redirect unavailable representations via symbolic linking. This means that file names are preserved, however the contents of the delivered video stream would be provided from another file. This video stream would not comply to the specification delivered in the manifest file, thus specific support on client side would be required.

Depending on the minimum reaction time achieved by the DC, it may be capable of providing additional ancillary services to the GA. Standardization by ENTSO-E[83] states that "[...] time for starting the action of PRIMARY CONTROL is a few seconds after the incident, the deployment time for 50 % or less of the total PRIMARY CONTROL RESERVE is at most 15 seconds and from 50 % to 100 % the maximum deployment time rises linearly to 30 seconds". In case a preemptive encoder is used or DASH segment size is lower than 15 s, a DC may choose to participate in primary control. As previously mentioned, it may be required to aggregate the flexibilities of several DCs to arrive at the required minimum active power reduction magnitude.

[83] `http://www.entsoe.eu/fileadmin/user_upload/_library/publications/ce/oh/Policy1_final.pdf`, accessed 15.07.2018.

5.8 Summary and Future Work

This chapter showed the feasibility of influencing server power demand through adaptations in software configuration. First, the principle of supervenience was introduced to show the theoretical foundation of influencing hardware activity by means of changes to software resource utilization. In the following, the exact point of action was determined. To avoid side effects impacting several co-located software services, the individual management of each service on software level is desirable. Subsequently, the general requirements for a software service to be suitable for runtime power adaptation were presented. These general observations were then applied to the specific use case of a live video encoding service. The available scalability dimensions in a live video encoding setting were identified as resolution, frame rate and bit rate. In the following, comparisons of power demand and visual quality between different configurations of such a service showed that the optimal choice of encoder parameters under power limitations is non-trivial. Adaptation of software configuration proofed to be superior to application-extrinsic resource limitation. Finally, a possible integration of an adaptive video encoder with the DASH standard was briefly presented.

Of course there is a certain limit to result generality. First and foremost, it has to be considered that only some applications fulfill the requirements stated in Section 5.4.2. More specifically, only a narrow subset of software allows for outputting a quality-adapted result while fulfilling its original task. The author is aware that reducing temporal and spatial resolution has different effects depending on the source material. In theoretical edge cases, temporal filtering would be lossless (video consisting of a single repeated picture), the same holds true for spatial filtering (video showing only a single color). Apart from these examples with little practical implications, differences in QoE degradation depending on the spatial detail and amount of motion contained in the source material do exist. However, as detailed in the author's own work [129], most conclusions also hold for a variety of source materials.

Future work may include studies on the applicability of software power adaptation for other types of applications. For service adaptation to work similarly to the live video encoding scenario regarded in this chapter, fairly strict requirements need to be fulfilled (namely, the three requirements listed in Section 5.4.2). Other candidate software might be:

1. **Live audio encoding:**
   Most findings in this chapter may be applied to a live audio encoding setting as well. This use case is relevant both in combination with video encoding (as most video also includes audio content) and standalone (e.g., voice calls in IP networks). Audio content also provides different levels of acceptable quality and scalability dimensions (e.g., frequency filtering and bit rate adaptation).

2. **Webservers and databases:**
   In case the software output may be delayed for a short amount of time, reduced energy demand may be achieved by consolidating requests into bursts, as outlined by Reviriego et al. [154]. Instead of implementing a burst traffic control logic in network equipment, this could also be handled by software internally (i.e., buffer requests for a certain amount of time before forwarding them). This way, a tradeoff between request delay and power demand may be realized.

Finally, an additional video encoder setting worth investigating is the quality preset. It controls several low level encoder settings, e.g., motion estimation algorithms. By adjusting the quality preset, CPU utilization is traded off with encoder efficiency, i.e., a lower quality preset requires more bit rate to reach a similar visual quality. Future work may include an additional investigation of bit rate limitations and its implications for both required quality presets and resulting power demand.

The man of science has learned to believe in justification, not by faith, but by verification.

Thomas Huxley

6 Safety Aspects

This chapter contains material previously published in [11].

The system presented in this work impacts the state of DCs directly and, indirectly, the power grid the respective DCs are connected to. DCs are viable candidates for integration into a D/R scheme relying on software flexibilities as they are major consumers of electrical energy and usually host a large number of software services. As shown in the previous chapters, adapting DC power demand comes at the price of potential loss in QoE. While there is general consensus that the public power grid represents a critical infrastructure [27], the classification of DCs as critical infrastructure is debatable. However, considering the increasing importance of DCs for essential services like communication (e.g., voice over IP (VoIP)) and the adoption of cloud computing throughout many industries, classification of certain DCs as critical infrastructure cannot be ruled out. Church et al. [38] define the term critical infrastructure as given in Definition 6.1.

Defintion 6.1: Critical infrastructure

Critical infrastructure is "[...] those elements of infrastructure that, if lost, could pose a significant threat to needed supplies (e.g., food, energy, medicines), services (e.g., police, fire, and EMS), and communication or a significant loss of service coverage or efficiency.".

Based on this definition, both power grids and at least some DCs need to be categorized as critical infrastructure. Regarding the former, many services essential for human health and well-being depend on electrical power (e.g., provisioning of water or communication). A potential negative effect on grid stability has therefore to be ruled out. Regarding the latter, DCs are indispensable in an increasing number of industries as they provide services like storage, communication and data processing. As such, certain DCs have to be categorized as critical infrastructure. Software used in critical infrastructure must be engineered in compliance with certain safety standards (e.g., IEC 61508: Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems [75]). In IEC 61508-3, the use of formal methods is recommended to achieve functional safety for safety integrity levels (SILs) of 2 and higher. In the following, one of these methods—model checking—is applied to DCs participating in a D/R program.

In Section 6.1 arguments are given for applying probabilistic model checking to energy systems. The public power grid is identified as a critical infrastructure and accordingly, software influencing the power grid is subject to certain engineering requirements. For example, IEC-61508 outlines model checking as a recommended method to verify safety critical systems. In the following Section 6.2 an outline of the explored scenario is given, additionally stating the assumptions taken in this chapter. Section 6.3 introduces the fundamental concepts of probabilistic model checking and the underlying structures, MDPs. Additionally, the concept of reward functions is defined. The following Section 6.4 describes the MDP- or MC-based model components which constitute the overall scenario model. Individual subsections are dedicated to the service load generator, residual power plan (RPP) estimator and DC models. In Section 6.5 the model is evaluated. First, the six investigated parameter settings governing the model are introduced. The evaluated model properties and objectives are presented in increasing levels of complexity: starting from comparatively simple objectives regarding single measures, more complex queries are evaluated subsequently. Section 6.6 discusses possible challenges arising from the quickly growing state space size of the model. The model sizes for different scenarios are presented, along with the objective evaluation times. Two possibilities for model size reduction are briefly introduced: power demand quantization and service aggregation. Subsequently, Section 6.7 aligns the obtained results with the flexibilities observed in live video encoding. Finally, Section 6.8 concludes this chapter and gives an overview on possible future work.

6.1 Model Checking of Energy-Related Properties

The system proposed in this work influences both DCs and power grids. Any modification to the operation of such infrastructures should consider the influence on system reliability, safety and performance. Model checking is a formal method that provides the means to verify certain properties of the designed system. In the following, primarily two properties are investigated: first, the capability of adaptive DCs to provide reliable power adaptation capabilities to a GA. In other words, in case a DC is requested to decrease its power demand, it has to be guaranteed that with high confidence, post-adaptation power is below a certain upper bound. Second, the property of a DC participating in a D/R system of fulfilling the SLAs (in the focused case, delivering an acceptable QoE) of its services is to be verified.

In line with the use case regarded throughout this thesis, a continuous D/R scenario is examined, in which a GA is creating a plan for power demand on consumer side. Further, the scenario assumes that scheduling job type software has already been completed. The scenario focuses on the subsequent adaptation of services in a way such as to not exceed the RPP (cf. Definition 6.2) after scheduling job type software.

Defintion 6.2: Residual power plan

A residual power plan *RPP* is a power plan resulting from a per-timeslot subtraction of job power demand J from the original power plan PP. Let $J = \{(t_1, p_1^J), (t_2, p_2^J), \ldots, (t_{PH}, p_{PH}^J)\}$ denote the power demand of scheduled jobs per time slot, $PP = \{(t_1, p_1^{PP}), (t_2, p_2^{PP}), \ldots, (t_{PH}, p_{PH}^{PP})\}$ the original power plan, and $RPP = \{(t_1, p_1^{PP} - p_1^J), (t_2, p_2^{PP} - p_2^J), \ldots, (t_{PH}, p_{PH}^{PP} - p_{PH}^J)\}$ the residual power plan.

Figure 6.1 identifies the part of the D/R system which is analyzed in the following sections.

6.2 Scenario & Assumptions

In the following, it is assumed that a GA delivers a power plan with a planning horizon PH of 24 hours in a rolling horizon pattern, i.e., the target power demand in each time slot is only stochastically known to the load. Several additional assumptions are taken for two primary reasons: first, to limit the high computational demands of the investigated model checking scenario. This is further elaborated on in Section 6.6. Second, distributions are chosen to simplify the analysis and detection of sensitivity to changes.

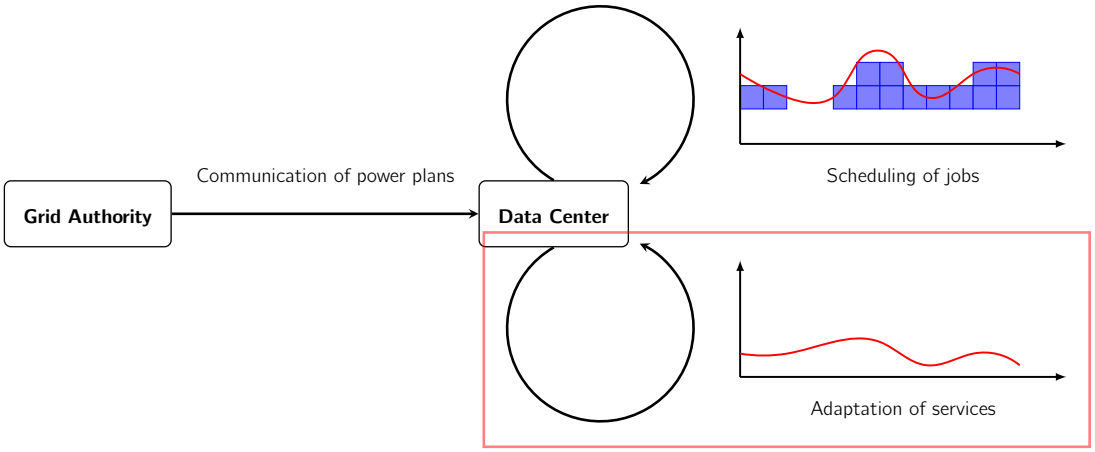


Figure 6.1: Focused subsystem identification in the D/R system.

1. It is assumed that demand side service load may be adapted within a certain range. As the scenario assumes a live video encoding service, adapting service load leads to a change both in current power draw and total energy demand: once a video sequence has been encoded, no future compensation for past quality impairments (and associated power savings) can be performed.

2. For simplicity, both DC power demand and service performance are represented as integer values. Service performance and power demand are assumed to change by equal amounts, i.e., reducing a service's performance by 1 unit in turn reduces its power demand by 1 unit. In general, the correlation between workload and respective power demand has to be assumed to be more complex as outlined in Chapter 4 and Chapter 5. However, the actual complexity of the power demand equation does not influence the size of the model as it does not impact the cardinality of the state space.

3. The amount of service load during a time slot cannot be perfectly forecasted due to the stochastical nature of user interactions. Accordingly, a distribution of the actual load around a mean value has to be assumed. This mean value could, e.g., be obtained by gathering historical data on the respective service's load. For simplicity and without loss of generality, it is assumed that service loads as well as the residual power plan follow a normal distribution around a respective mean value in each time slot. If a normal distribution does not adequately capture the actual patterns, other distributions may be chosen instead without invalidating the methodological approach presented henceforth.

4. Similarly, the RPP is modeled as normally distributed around a constant mean. This reflects a simplified version of the power planner introduced in Chapter 3. With an increasing value

of α_{PP}, the probability of deviating from the baseline power demand is decreasing. This is elaborated on in more detail in Section 6.4.1.2.

5. Both exceeding the RPP and delivering a video stream at a reduced quality results in a penalty, which is discussed in more detail in Section 6.4.1.4. Both penalties are represented by sum of squared error values. Generally, (and similar to the power demand calculations) the complexity of penalty equations does not impact the model state space.

6.3 Preliminaries

In the following, an introduction to the basic concepts of MDPs is given. An extensive explanation would exceed the the scope of this work, so the reader is referred to literature for further details (see, e.g., Puterman [145]). The following introduction of theoretical foundations and compositional models required in the further course of this chapter is based on [11] and modified to fit the focused use case.

An underlying structure which may be employed for model checking are Markov chains and—more generalized—MDPs. The latter are formally introduced in Definition 6.3.

Defintion 6.3: Markov decision process

An MDP is a tuple $\mathcal{M} = (S, \text{Act}, P, \text{AP}, \mathsf{L})$, with S denoting a finite set of states, Act a finite set of actions, $P : S \times \text{Act} \times S \to [0, 1]$ the transition probabilities, AP a finite set of atomic propositions, and $\mathsf{L} : S \to 2^{\text{AP}}$ a labeling function.

Several additional requirements and explanations are given in the following. The values $P(s, \alpha, s^*)$ have to be rational and $\sum_{s^* \in S} P(s, \alpha, s^*) \in \{0, 1\}$ for all states $s \in S$ and actions $\alpha \in \text{Act}$. The 3-tuples (s, α, s^*) with $P(s, \alpha, s^*) > 0$ are called *transitions*. An action α for which $P(s, \alpha, s^*) > 0$ holds for a state s^* is called *enabled* in state s. $\text{Act}(s)$ denotes the set of actions that are enabled in $s \in S$. It is required that $\text{Act}(s) \neq \varnothing$ holds for all states s to avoid terminal behavior. A path π in an MDP $\mathcal{M}$ is a sequence alternating between states and actions, i.e., $\pi = s_0\, \alpha_0\, s_1\, \alpha_1 \ldots \in (S \times \text{Act})^* S \cup (S \times \text{Act})^\omega$ with $\alpha_i \in \text{Act}(s_i)$ and $P(s_i, \alpha_i, s_{i+1}) > 0$ for all i. It is noteworthy that these paths may be finite or infinite, however infinite paths are not required in the further course of this chapter. In the following, it is assumed that an initial state s is given.

An MDP's actions in a state s may generally be non-deterministic. To resolve this non-determinism, a *scheduler* σ is required to choose among the possible actions of a state. A path property ϕ is

denoted as $Pr^{\sigma}_{\mathcal{M}}(\phi)$, signifying the probability of ϕ in $\mathcal{M}$ while employing a scheduler σ. $Pr^{min}_{\mathcal{M}}(\phi)$ and $Pr^{max}_{\mathcal{M}}(\phi)$ denote the minimal and maximal probabilities for ϕ, respectively, among all schedulers σ.

Additionally, reward functions may be created to assign values to state-action pairs. Formally, reward functions are described in Definition 6.4.

Defintion 6.4: Reward function

A reward function rew : $S \times \text{Act} \to \mathbb{N}$, assigns a natural number to state-action pairs.

Rewards may also be assigned to either states or actions independently. Rewards may be summarized along a finite path π. It should be noted that for a certain target state G, it is possible to calculate the expected reward until this state is reached, given a scheduler σ, by summarizing the rewards of each possible path weighted by the path's probability. This way, also the minimum and maximum expected rewards $Ex^{min}_{\mathcal{M}}$ and $Ex^{max}_{\mathcal{M}}$ may be evaluated.

6.4 Model Creation

In this section an MDP-based compositional model for continuous D/R for the use case of a DC is presented. Following the description given in Section 6.2, the scenario consists of a DC and an RPP, obtained by subtracting the power demand of scheduled jobs from an original power plan received from a GA. The DC needs to adapt its services to keep its power demand in line with power available as signaled in the RPP. This is formalized in stochastic distributions of the service loads and RPP as well as the nondeterministic choices within the DC. The MDP model created on the basis of these components is the foundation for a further formal analysis. Following the described setting, strategies (i.e., nondeterminism-resolving actions in the MDP) are evaluated regarding their capability to optimize several QoE or power-related objectives introduced later in this chapter.

The model used in the following considers two services $ser_1, ser_2 \in SER$ that are hosted inside the DC, as well as a number $T \in \mathbb{N}$ of discrete time steps into which the day is divided, similarly to the time slots concept introduced in Chapter 3. At each time slot t the DC needs to decide which service to adapt how aggressively in power demand and service quality. The service load is limited to $\{0, \ldots, 4\}$ for ser_1 and $\{0, \ldots, 10\}$ for ser_2. The maximum adaptation of the services is limited to 2 for $ser_1$and 4 for ser_2. The RPP is limited to $\{0, \ldots, 14\}$. The DC's service adaptation

decision is influenced by the RPP (which in turn is influenced by the power plan received from the GA and scheduling of jobs). Both services exhibit a power demand that for each time slot, conforms to a normal distribution with constant mean and standard deviation.

The model is equipped with reward functions to capture the number of time steps and penalties for violating the RPP or service quality constraints. For the penalty functions, standard squared error measures are used, however in general arbitrarily complex functions may be used capture penalties. In the following, the components of the MDP-based model are presented.

6.4.1 Model Components

The model needs to take into account all processes that influence the successful adaptation to the requested RPP and compliance to service quality: the processes that are being run in the DC, the power requested by the RPP, and the decisions taken inside the DC regarding adaptation of services. Each of them needs to be represented in a structure that is adequate for use in a model checking setting. Deterministic components, i.e., models that are not governed by a scheduler, may be represented by a Markov chain (MC). In the focused scenario, sub-models that can be expressed this way are services and RPP. The only component requiring a scheduler is the DC itself: it needs to decide at which point to adapt a service by which amount. In other words, in each time slot the scheduler σ must decide on an action to perform in the current state. It is noteworthy that the actual choice among the set of available actions is limited by the current service load (i.e., the amount of performed service adaptation cannot exceed the service demand).

In summary, the considered model for evaluation is a MDP consisting of four sub-models. Each of the components is modeled as either an MC or MDP, depending on whether nondeterminism is present or not: two stochastic service load estimators $\mathcal{M}_{ser}(ser_1)$ and $\mathcal{M}_{ser}(ser_2)$, an RPP forecast $\mathcal{M}_{RPP}$ and the DC $\mathcal{M}_{DC}$. It is noteworthy that additional services may be added to the model by introducing another instance of $\mathcal{M}_{ser}$ to the final MDP. The four introduced components are used as basis for the final MDP model $\mathcal{M} = \mathcal{M}_{ser}(ser_1) \otimes \mathcal{M}_{ser}(ser_2) \otimes \mathcal{M}_{RPP} \otimes \mathcal{M}_{DC}$.

6.4.1.1 Service Load Generators ($\mathcal{M}_{\mathsf{ser}}$)

Without loss of generality, a service's load is assumed to follow a normal distribution with a constant mean. A service's load may assume other, more complex distributions. It should be noted however that especially distributions with a high standard deviation will significantly increase model size, as discussed in Section 6.6. For any point in time $t \in \{ 0, 1, \ldots, T \}$ for the respective service $ser_x \in SER$, a random variable $\mathsf{service}(t) \in \{ 0, 1, \ldots, max(ser_x) \}$ is given, signifying the load caused by service ser_x at time t. From this, a set of MCs $\mathcal{M}_{ser}(ser_x)$ can be derived, with states of the form

(t, c) denoting a service load of c at time t ($\mathsf{service}(t) = c$). Each service is represented by an individual MC, which exhibits transition probabilities from (t, c) to $(t+1, c^*)$ in a way that reflects the probabilities of the underlying distribution assuming the value c^*. Specifically, the probability of a value c^* being assumed at time $t + 1$ is independent from the values assumed at previous points in time. Relaxing this constraint in a way such as to reflect high probabilities of service load remaining unchanged (i.e., placing the mean of a normal distribution at c for $t+1$) would increase the model size, as the possible values must generally be assumed to increase (unless there are hard lower and upper bounds on service power demand). Figure 6.2 shows two possible realizations of the $\mathfrak{M}_{ser}$ Markov chain.

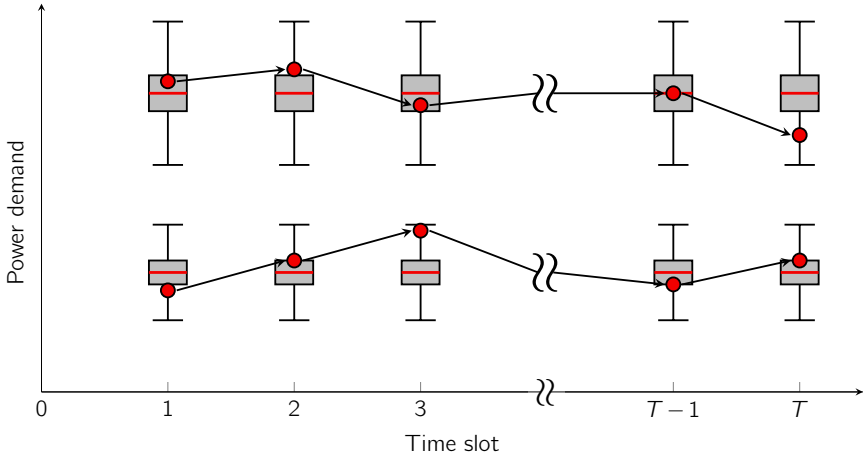


Figure 6.2: Two possible underlying distributions for $\mathfrak{M}_{ser}$, each with a possible realization.

In the regarded model instance, two services are observed: A small service (ser_1) with a power demand mean of 2 and a large service (ser_2) with a power demand mean of 6. In the baseline scenario, both services have a standard deviation of 1.0.

6.4.1.2 Residual Power Plan Estimator ($\mathfrak{M}_{\mathsf{RPP}}$)

As previously stated, the model is concerned with adapting services to an RPP after scheduling job-type software. At $t = 0$, an RPP for the next T time slots is assumed as given in the regarded scenario. To account for non-ideal scheduling of jobs and possible subsequent changes to the power plan, it is assumed to follow a normal distribution similar to that in $\mathfrak{M}_{ser}$. Assuming a normal distribution is in line with a simplified version of the power planning process presented in Chapter 3: depending on the value assigned to α_{PP}, a lower or higher standard deviation of the distribution is set. Figure 6.3 shows a histogram obtained by considering the values of all power

plans created during the evaluation of parameter α_{PP}, except for $\alpha_{PP} = 0$ (assigning no cost to deviations from baseline power demand is disregarded).

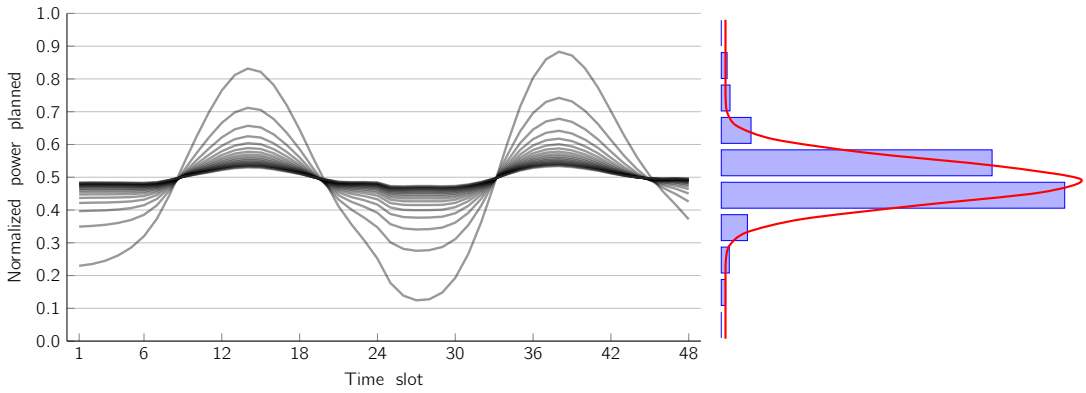


Figure 6.3: Power planner value distribution.

The histogram can be approximated closely by a normal distribution. However, it has to be noted that for very low values of α_{PP} (< 0.2), the power plan does not follow a normal distribution. Instead, values show a bimodal distribution with local PDF maxima at the lower and upper bounds of the power plan. In this case, the simplified approximation via a normal distribution has to rely on considering that adaptation is not towards the original power plan, but an RPP. This RPP, as previously defined, is created by scheduling job-type load in a way such as to eliminate large-scale deviations (e.g., due to diurnal renewable patterns). Should this assumption not hold, a lower value of α_{PP}, or tighter upper and lower bounds on maximum adaptation need to be considered. An RPP of a value p in time slot t signifies that at time t the power demand of all services provided by a DC should not exceed p. In case a DC falls short of the demanded power p, the necessary amount of load to reach p may be generated locally by the DC itself (e.g., via local jobs like virus scans, grid computing or blockchain proof-of-work calculations). As previously stated, the MC representing the RPP is similar to $\mathcal{M}_{ser}$ as depicted in Figure 6.4.

In case modeling an RPP based on deviations from a constant mean power demand is insufficient, an RPP may be modeled explicitly considering its remaining energy. In this case, a modified MC may be employed for the model. Assuming the power plan is energy conserving and again, an initial RPP for T time steps is given, the modified MC for a specific combination of α_{PP}, β, γ, and δ (cf. Section 3.6.1) is suggested to model state-specific remaining energy budgets. Figure 6.5 shows an example MC trace.

As can be seen in Figure 6.5, the modified MC captures the power plan's varying certainty of upcoming time slots. The difference between the possible extremal points of the RPP increase in

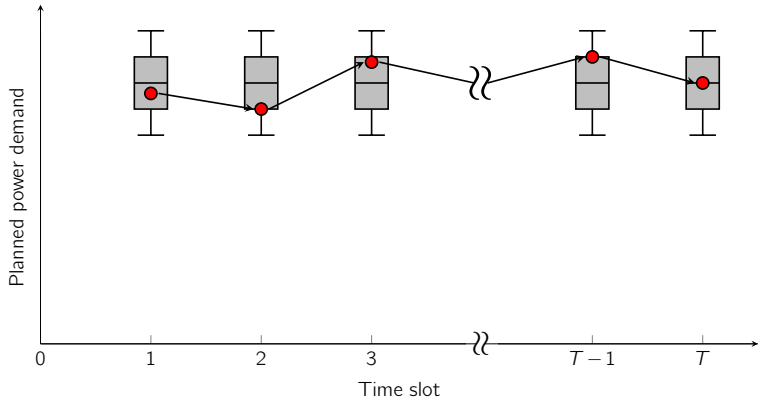


Figure 6.4: A possible underlying distribution for $\mathcal{M}_{RPP}$ with a possible realization.

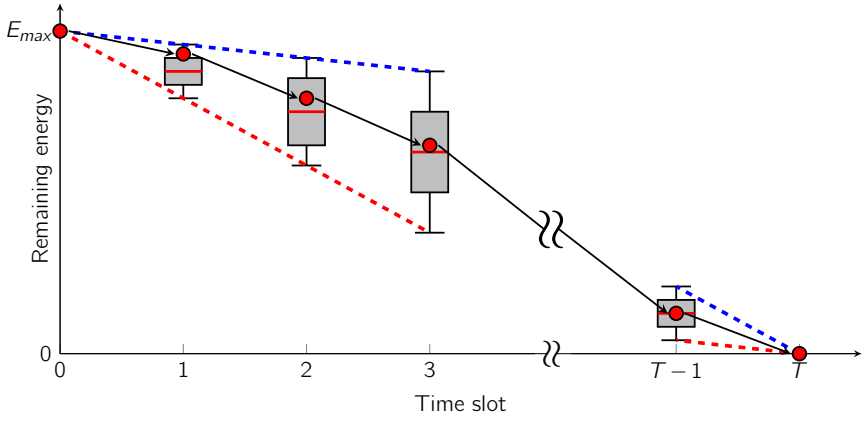


Figure 6.5: A possible underlying distribution for a modified $\mathcal{M}^{*}_{RPP}$ with a possible realization.

the first time slots, in each time slot by $P^{max}_{DC} - P^{min}_{DC}$, until the remaining energy budget inhibits further divergence of the extremal points. Higher values of α_{PP} translate to a lower standard deviation from the baseline value at each time slot. Increasing β_{PP} decreases standard deviation from t to $t+1$. The values of $\mathcal{M}_{RPP}$ are strictly monotonically decreasing, more precisely for each time slot t, transition probabilities to states with $E_t - E_{t+1} < \mathrm{P}^{\mathrm{min}}_{\mathrm{PP}}$ and $E_t - E_{t+1} > \mathrm{P}^{\mathrm{max}}_{\mathrm{PP}}$ are equal to 0. Specifically, also the lower and upper bound at T being equal to 0 holds in an energy-preserving power plan. However, an energy-centric model of a power plan may cause a dramatic increase in model state space size as it has to keep track of all past energy use. Depending on PH and the overall power flexibility of the DC ($\mathrm{P}^{\mathrm{max}}_{\mathrm{PP}}$ - $\mathrm{P}^{\mathrm{min}}_{\mathrm{PP}}$), a high number of possible values for remaining energy need to be considered. Another reason for not employing the modified $\mathcal{M}_{RPP}$ model is that in the use case of live video encoding, a strictly energy conserving modeling of the power plan

is not required. Once a lower quality encoding is performed, there is no way of restoring quality with additional energy investment. Live video encoding therefore is energy-adaptive (as opposed to power-adaptive), making a power-centric modeling of the RPP feasible. Further information on model complexity is given in Section 6.6.

6.4.1.3 Data Center ($\mathcal{M}_{\mathsf{DC}}$)

The DC tracks the time $t \in \{ 0, 1, \ldots, T \}$ and the adaptation level of services ser_1 and ser_2. The time starts at $t = 0$ and increases linearly until the day is over, i.e., $t = T$. At each point in time, the DC may choose to adapt the power demand (and delivered service quality) of each of the hosted services independently. The maximum adaptation for ser_1 in time slot t is $adapt_{max}(ser_1, t) := min(P(ser_1, t), adapt_{max}(ser_1))$, with $P(ser_1, t)$ denoting the power demand of ser_1 at time t and $adapt_{max}(ser_1)$ the static maximum adaptation amount of ser_1. Similarly, for ser_2 the maximum adaptation at time t is $adapt_{max}(ser_2, t) := min(P(ser_2, t), adapt_{max}(ser_2))$, with $P(ser_2, t)$ denoting the power demand of ser_2 at time t and $adapt_{max}(ser_2)$ the static maximum adaptation amount of ser_2. In the following, the static maximum adaptations $adapt_{max}(ser_1)$ and $adapt_{max}(ser_2)$ are assumed equal to 2 and 4, respectively. The choices regarding service adaptation are modeled nondeterministically: whether or not the action of adapting a service by a specific amount is enabled at time t depends on the current service load $P(ser_x, t)$. The enabled actions in the final model $\mathcal{M}$ are limited to the following: adapting a service ser_x by $i \in \mathbb{N}_0$ at time t is enabled in $\mathcal{M}$ iff $i \leq adapt_{max}(ser_x, t)$, i.e., iff the service load minus the service adaptation is non-negative and service adaptation does not exceed the predefined maximum adaptation.

6.4.1.4 Reward Functions

To model the costs of either violating the RPP or the limits regarding service quality, two respective reward functions `power_penalty` and `qoe_penalty` are introduced. The penalty for delivering a lower than optimal service quality in a state s of $\mathcal{M}$ when adapting a service ser_x is defined in Equation (6.1).

$$\texttt{qoe\_penalty}(ser_x) = \sum_{ser_x \in SER} adapt_{max}(ser_x, t)^2 \tag{6.1}$$

Similarly, the penalty for exceeding the RPP in time slot t is given in Equation (6.2).

$$\texttt{power\_penalty}(t) = \Big[max\Big(0, (\sum_{ser_x \in SER} P(ser_x, t)) - RPP(t)\Big)\Big]^2 \tag{6.2}$$

Specifically for the purpose of checking model confidence (cf. Section 6.5.2.1), a further reward function is added: `steps` is used to capture the number of time steps.

6.4.2 Previously Evaluated Model Variant

In previous work, the author co-developed a model aimed at verifying properties of a traditional D/R scenario. This model considered stochastically arriving D/R requests, local renewable power generation and a fixed number of jobs to be completed during a 24 hour interval. Additionally, service load was assumed to be non-adaptable and non-delayable. In this model, the DC scheduled jobs while trying to use as little non-renewable energy as possible.

This previous model may be considered complementary to the one presented here, as it may—with some modification—be used to verify the job scheduling step assumed to happen prior to the verification steps performed in the subsequent sections. The main differences of this previously investigated model are enumerated in the following.

1. **Green energy forecast:**
   The model considered the local generation of renewable energy via, e.g., solar panels or wind turbines. This was captured in a dedicated MC giving a stochastic value representing a green energy availability forecast.
2. **Request generator:**
   In the previous model variant, a classical D/R scenario was investigated, i.e., DC power adaptation was only requested at some stochastically distributed time slots of the day. Additionally, the requests explicitly stated upper and lower bounds on power demand. Again, this was captured in a dedicated MC capturing both time and power bounds of a D/R request.
3. **Data center:**
   The DC focused on scheduling a maximum number of jobs while consuming as little non-renewable energy as possible. These two competing objectives were the primary focus of the analysis.

For further information on the previously investigated model and a full evaluation of model checking results, the reader is referred to the publication [11].

6.5 Evaluation

The upcoming section is concerned with evaluating the model described in Section 6.4. First, the investigated scenarios are presented in Section 6.5.1. Next, several properties are verified that contribute to model confidence, i.e., properties whose evaluation results need to assert certain values for the model to be correct. Finally, several other properties and objectives are verified.

6.5.1 Investigated Scenarios

In the evaluation, three different scenarios are covered. These scenarios are chosen in a way such as to demonstrate the influence of certain changes in parameters regarding the RPP or running services. These are focused more closely in the following.

- **Baseline scenario**
  The baseline scenario is used as a reference to compare other scenarios against. In this setting the DC is running two services, ser_1 and ser_2, which are modeled as random variables following a normal distribution with a standard deviation of 1.0 and a mean value of 2.0 and 6.0, respectively. The RPP describes the remaining power plan after subtracting the power needed to satisfy the power demand of DC jobs. It is also modeled as a random variable following a normal distribution with a standard deviation of 1.0. Its mean value is 8.0, matching the mean combined service power demands of ser_1 and ser_2. The baseline scenario model has 1 780 462 states and 62 719 677 transitions.
- **Scenario #1: standard deviation of services**
  In this scenario, the influence of different standard deviation values of the hosted services is tested. A higher standard deviation may be caused by either an inherently volatile load of a service or low service load forecast accuracy. This causes larger peak deviations from the mean value. The opposite holds for lower values of standard deviation. The hypothesis regarding the outcome of this scenario is as follows: increasing the standard deviation of services decreases the likelihood of generally adhering to the RPP both as a standalone objective and while staying inside an acceptable quality range. Decreasing the standard deviation should have the opposite effect. For a standard deviation value of 0.5, the model has 718 426 states and 22 364 325 transitions. For a standard deviation value of 2.0, the model has 2 265 256 states and 78 747 075 transitions.
- **Scenario #2: standard deviation of residual power plan**
  In this scenario, the influence of different standard deviations of the RPP is tested. A higher standard deviation models an RPP that does not well represent the combined power demand

of the DC's services. This situation may arise if the DC accepts power plans that show more volatility (lower α_{PP} value) than can be compensated for during job scheduling. The required adaptation of services may be intentionally accepted or job scheduling may be unable to create a more fitting RPP, e.g., due to bad scheduling algorithm choice or large, undividable jobs. The hypothesis regarding the outcome of this scenario is similar to that of scenario #1: increasing the standard deviation of the RPP decreases the likelihood of services to successfully adapt enough to stay inside the limits imposed by the RPP. Decreasing the standard deviation should have the opposite effect. For a setting of 0.5 standard deviation, the model has 989 146 states and 23 957 865 transitions. For a setting of 2.0 standard deviation, the model has 2 967 436 states and 153 521 595 transitions.

- **Scenario #3: mean value of residual power plan**
  In this scenario, the influence of different mean values of the RPP is tested. A higher mean value means on average, more energy than required by the services is delivered over 24 hours. Although this is excluded in the evaluated power planning algorithm introduced in Chapter 3, the impact of a changing energy supply is interesting for future work, which could include energy adaptivity as means to adjust to non-energy conserving power plans. The hypothesis regarding the outcome of this scenario is as follows: a higher overall energy provision increases the chance of successful service adaptation, as undershooting the RPP is not considered a problem. Decreasing the overall energy provision should have the opposite effect. For both mean values of 7 and 9 the model has 1 780 462 states and 62 719 677 transitions. A change in mean value of the RPP has no influence on state and transition count as it does not create a wider range of potential values of the RPP.

A summary of the key parameters used in the different scenarios is given in Table 6.1.

6.5.2 Properties & Objectives

In the following, different model checking properties and objectives relevant to the considered scenarios are presented. The first properties are related to model confidence, i.e., successfully checking these properties provides results which are valuable to improve confidence in the validity of the created model. Other objectives focus on the optimization of different cost and/or performance measures. These objectives bear high relevance to formally proving the safety of introducing the proposed D/R system into the power grid. In the following, standard temporal operators *eventually* $\Diamond$ and *always* $\Box$ are used for the notation. The model and discussed properties are implemented in the model checking program *PRISM* [95]. The terms *property* and *objective* both refer to properties of the model under investigation, however in the following the former is

	Random variable	Range	Mean value	Standard deviation
Baseline scenario	ser_1	$[0, 4]$	2.0	1.0
	ser_2	$[0, 10]$	6.0	
	RPP	$[0, 14]$	8.0	
Scenario #1	ser_1	$[0, 4]$	2.0	(a) 0.5; (b) 2.0
	ser_2	$[0, 10]$	6.0	
	RPP	$[0, 14]$	8.0	1.0
Scenario #2	ser_1	$[0, 4]$	2.0	1.0
	ser_2	$[0, 10]$	6.0	
	RPP	$[0, 14]$	8.0	(a) 0.5; (b) 2.0
Scenario #3	ser_1	$[0, 4]$	2.0	1.0
	ser_2	$[0, 10]$	6.0	
	RPP	$[0, 14]$	(a) 7.0; (b) 9.0	1.0

Table 6.1: Settings used in the different model checking scenarios.

used to address statements claiming certain verifiable characteristics of the model, while the latter is used for statements that query the probability of a model property being fulfilled.

6.5.2.1 Model Confidence Properties

Before evaluating any objectives and properties regarding DC adaptation, it is reasonable to ensure key aspects of model validity. Examples include, e.g., model liveliness or absence of deadlocks within the model. Objective probabilities can be used as *sanity checks*, i.e., to confirm that their values are inside reasonable bounds. Some examples of properties regarding model confidence are presented in the following.

The model has to ensure that each path eventually reaches the terminal state, in this model this is equal to time slot 24 (henceforth referred to as `end`). If this property would not hold, it would mean that there exists at least one path that results in a deadlock situation. However, if the minimal probability to reach the end of the day is one, every scheduler is sure to reach the end of the day. This is shown in Formula (6.3).

$$Pr^{min}_{\mathcal{M}}(\Diamond\, \texttt{end}) = 1. \tag{6.3}$$

To additionally make sure that no time slots can be skipped in the model, Formula (6.3) can be modified to ensure the minimum and maximum expected step count are equal to 24 via a reward structure, as shown in Formulas (6.4) and (6.5).

$$Ex_{\mathcal{M}}^{max}[\texttt{steps}]\,(\Diamond\ \texttt{end}) = 24, \tag{6.4}$$

$$Ex_{\mathcal{M}}^{min}[\texttt{steps}]\,(\Diamond\ \texttt{end}) = 24. \tag{6.5}$$

Additionally, it should be possible that the RPP requested is matched by the DC. This is assumed to be the case if the power demanded by the services is equal or smaller than the power assigned by the power plan after satisfying all jobs present in the DC. Technically, this is the case if there exists an optimal scheduler such that the RPP is adhered to by the DC, as given in Formula (6.6).

$$Pr_{\mathcal{M}}^{min}(\Diamond^{\texttt{power\_penalty>0}}\ \texttt{end}) = 0. \tag{6.6}$$

Similarly, it is of interest if there is a chance for an optimal scheduler to adapt services in a way such as to not violate constraints on service quality, as shown in Formula (6.7).

$$Pr_{\mathcal{M}}^{min}(\Diamond^{\texttt{qoe\_penalty>0}}\ \texttt{end}) = 0. \tag{6.7}$$

An additional check to increase model confidence is to analyze a modified model without any probabilistic uncertainties. Removing the normal distributions of the RPP and service load and replacing them with a respective fixed value allows to check whether a constant RPP is achieved by services with constant service loads. Depending on the sum of service loads, the power should either always or never match, even with an optimal scheduler. This is shown in Formulas (6.8) and (6.9).

$$Pr_{\mathcal{M}}^{max}(\Box\,\texttt{power\_match}) = 1, \qquad \text{iff RPP}(t) \leq \sum_{ser_x \in SER} P(ser_x, t) - adapt_{max}(ser_x, t), \tag{6.8}$$

$$Pr_{\mathcal{M}}^{max}(\Diamond\,\texttt{power\_match}) = 0, \qquad \text{else.} \tag{6.9}$$

All properties (6.3) to (6.9) are verified successfully in the investigated model.

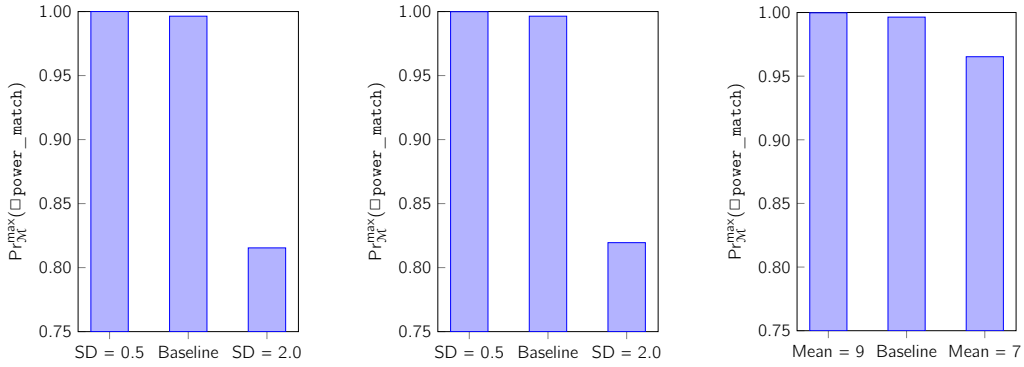
6.5.2.2 Objectives Addressing Single Measures

After validating basic model validity, in the following objectives regarding the optimization of single measures are focused, either power demand or service quality. In the considered scenarios, two objectives with high importance are the probability that the DC power demand continuously stays below the value of $RPP(t)$ (`power_match`) and for the services to never fall below the minimum expected QoE (`quality_ok`). The latter is assumed to hold if the adaptation of ser_1 and ser_2 are ≤ 1 and ≤ 2 respectively at all times. Objectives (6.10) and (6.11) show the respective formulations, assuming optimal schedulers.

$$Pr^{max}_{\mathcal{M}}(\square\,\texttt{power\_match}) \tag{6.10}$$

$$Pr^{max}_{\mathcal{M}}(\square\,\texttt{quality\_ok}) \tag{6.11}$$

Objective (6.10) can be achieved with high probability in most scenarios, however not with absolute certainty: if the maximum adaptation of both ser_1 and ser_2 is insufficient to decrease total service load to $\leq RPP(t)$ for all t, this objective fails. Figures 6.6a to 6.6c show the model checking results for Objective (6.10) regarding all considered scenarios.



(a) Scenario #1 **(b)** Scenario #2 **(c)** Scenario #3

Figure 6.6: Maximum probability of achieving Objective (6.10).

The maximum probability for reaching Objective (6.11) is equal to 1 for all three scenarios. This can be achieved by never adapting any service. Most likely, this results in failing to adhere to the RPP, however this is not considered in Formula (6.11).

Another way to address the optimization of single measures is via reward functions. In the considered scenarios, the defined reward functions are used to measure penalties for failing to comply

with power limits or minimum service quality. The maximum and minimum penalty for missing power targets are given in Objectives (6.12) and (6.13), respectively.

$$Ex_{\mathcal{M}}^{max}[\texttt{power\_penalty}](\Diamond\ \texttt{end}) \tag{6.12}$$

$$Ex_{\mathcal{M}}^{min}[\texttt{power\_penalty}](\Diamond\ \texttt{end}) \tag{6.13}$$

Another important measure regarding penalties is the maximum and minimum expected penalty regarding service quality, as given in Objectives (6.14) and (6.15) respectively.

$$Ex_{\mathcal{M}}^{max}[\texttt{qoe\_penalty}](\Diamond\ \texttt{end}) \tag{6.14}$$

$$Ex_{\mathcal{M}}^{min}[\texttt{qoe\_penalty}](\Diamond\ \texttt{end}) \tag{6.15}$$

Table 6.2 shows the results of evaluating Objectives (6.12) to (6.15).

		Obj. (6.12)	Obj. (6.13)	Obj. (6.14)	Obj. (6.15)
Baseline scenario		38.20	2.19×10^{-3}	455.12	0.00
Scenario #1	(a)	20.79	2.24×10^{-6}	468.55	0.00
	(b)	84.28	1.74×10^{-1}	422.17	0.00
Scenario #2	(a)	29.10	6.61×10^{-5}	455.12	0.00
	(b)	74.19	2.51×10^{-1}	455.12	0.00
Scenario #3	(a)	86.20	2.45×10^{-2}	455.12	0.00
	(b)	14.19	1.28×10^{-4}	455.12	0.00

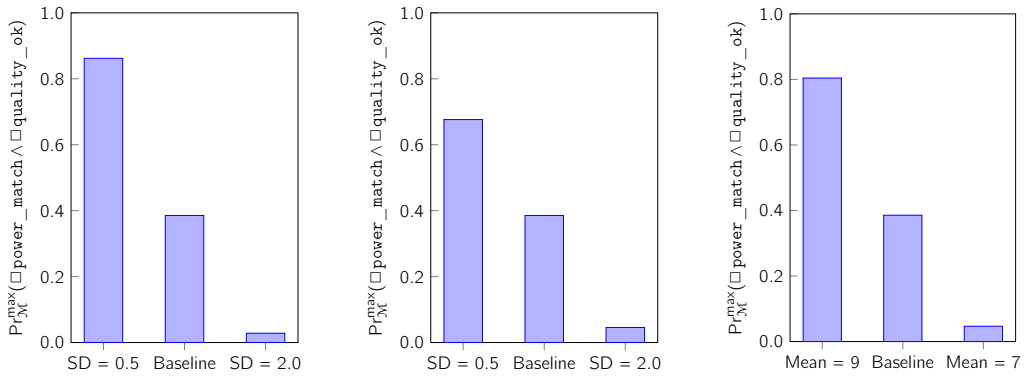
Table 6.2: Results for reward function based objectives.

6.5.2.3 Addressing Both Objectives at Once

One of the most important model objectives is satisfying both power and service quality constraints simultaneously. Ideally, this objective is fulfilled regardless of service utilization volatility and in all time slots. Formula 6.16 shows the corresponding objective formulation.

$$Pr_{\mathcal{M}}^{max}(\Box\ \texttt{power\_match} \wedge \Box\ \texttt{quality\_ok}). \tag{6.16}$$

Following the assumptions of the considered scenario, Formula 6.16 holds, iff an optimal scheduler can adapt all services in a way such as to not exceed the RPP in any time slot while at the same time not violating the constraints on service quality. Figures 6.7a, 6.7b and 6.7c show the model checking results for all considered scenarios.



(a) Scenario #1 **(b)** Scenario #2 **(c)** Scenario #3

Figure 6.7: Maximum probability of achieving power and service quality objectives simultaneously.

As can be seen, higher volatility in either service power demand or RPP decrease the probability of successfully adapting service load. Even under favorable conditions, achieving the objective is not guaranteed: in Scenario #1, with a service standard deviation of 0.5, the maximum probability of satisfying both power and quality objectives over a whole day is 86.22 %. A similar result can be observed with an increased mean value of the RPP: in Scenario #3, with a mean value of 9.0 of the RPP, the maximum probability is 80.42 %. It has to be concluded that for a highly reliable adaptation, either the power plan needs to be *oversized*, allowing more power to be drawn than demanded on average by the DC, or service power demand needs to be very accurately forecastable. In the following Section 6.5.2.4, possibilities of trade-off between the objectives of power matching and service quality are investigated in more depth.

6.5.2.4 Objective Trade-Offs

A second class of model properties are concerned with the trade-off between several objectives. In the chosen scenarios, one of these properties is of special importance: the trade-off between adhering to limits on power demand set by the RPP and delivering an acceptable service quality. This property is shown in Formula 6.17 and focused more closely in the following.

$$pareto((\Box\, \texttt{quality\_ok})_{max}, (\Box\, \texttt{power\_match})_{max}). \tag{6.17}$$

The notation used in Formula 6.17 is similar to that introduced in the work of Forejt et al. [53]. For more information on pareto calculations in probabilistic model checking, the reader is referred to this publication.

In a first scenario, different standard deviations of the services are compared regarding the trade-off between staying within the limits imposed by the RPP (`power_match`) and delivering an acceptable service quality (`quality_ok`). Figure 6.8 shows the resulting pareto plots.

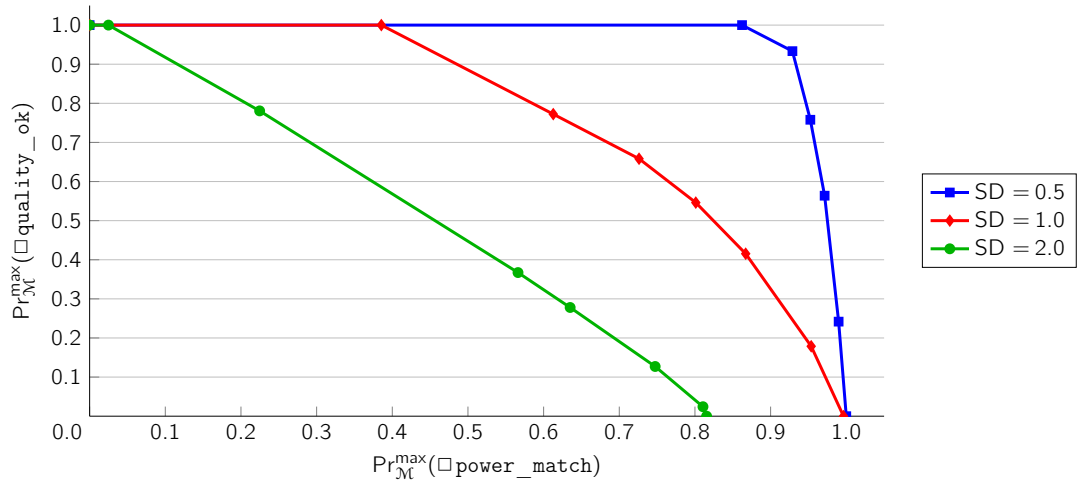


Figure 6.8: Pareto plot using different standard deviations for service distributions.

As can be seen, the successful implementation of both `power_match` and `quality_ok` depends on service standard deviation. This is to be expected, as larger deviations from service mean values may result in power demand which is too high to be adjusted by service adaptation while staying within an acceptable quality range.

A similar, however not identical, outcome can be observed when changing RPP standard deviation, as shown in Figure 6.9.

Changing the RPP standard deviation can be interpreted as a changing quality of job scheduling on DC side: after scheduling jobs, the RPP should closely match the total service power demand in each time slot. A higher standard deviation of the RPP signifies a higher error while matching RPP and total service power. Other possible causes include too small cost parameters or inadequate settings regarding upper and lower bounds in the power planning process.

Another influence on the joint achievability of both `power_match` and `quality_ok` is the mean RPP value. For the evaluation, the standard deviation of services and the RPP is set to the default

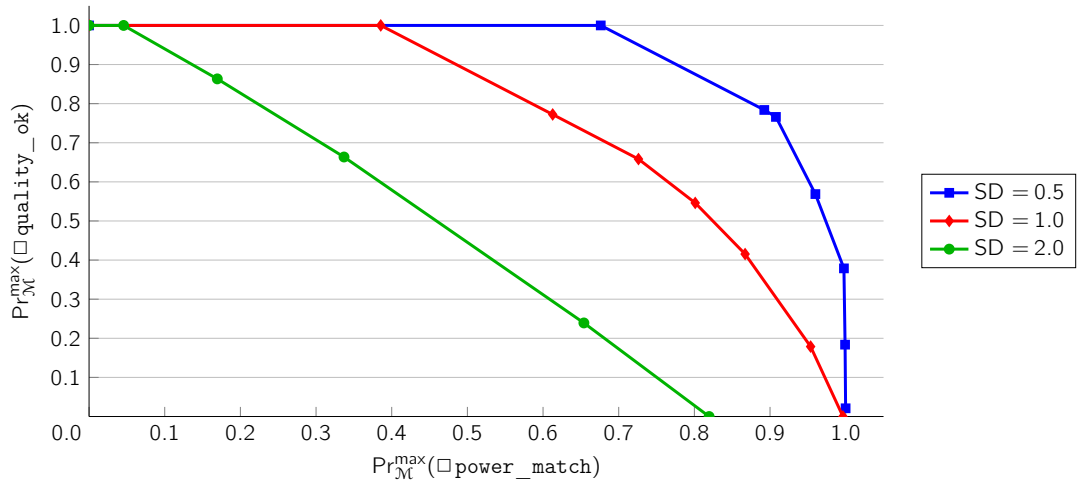


Figure 6.9: Pareto plot using different standard deviations for RPP distribution.

value of 1.0; the mean value of the RPP is set equal to the total service power ±1. The results of this evaluation are depicted in Figure 6.10.

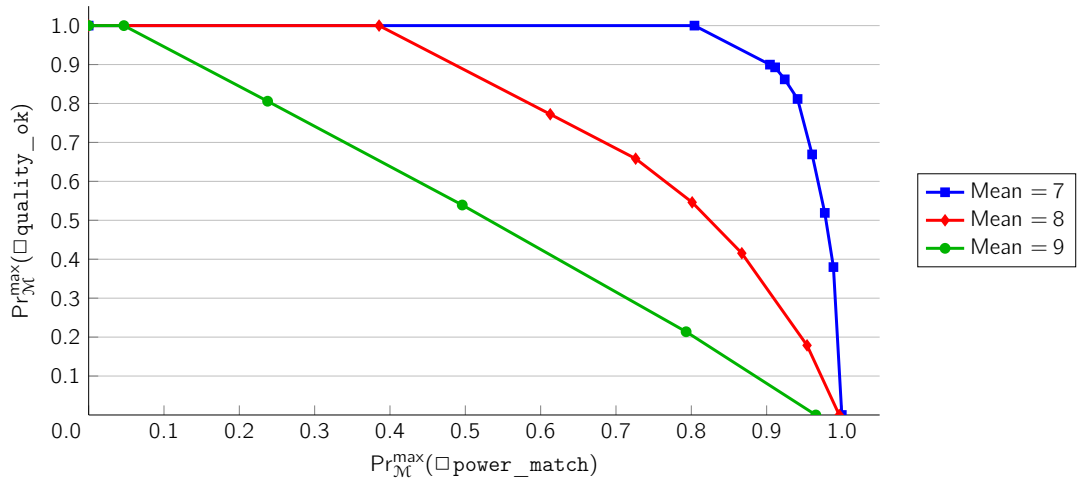


Figure 6.10: Pareto plot using different mean values for RPP distribution.

6.5.2.5 Quantile Queries

Another possibility to consider objectives and their trade-offs are quantile queries. Quantile queries find the maximum or minimum value of a variable while ensuring a probability threshold for a

property does not pass a defined threshold. Objective (6.18) searches for a scheduler which, until the end of the day, accumulates at most n power penalty with probability $q \in [0, 1]$.

$$\min\{\ n \in \mathbb{N} : Pr^{max}_{\mathcal{M}}(\Diamond^{\texttt{power\_penalty} \leq n}\ \texttt{end}) > q\ \} \tag{6.18}$$

Figure 6.11 shows the evaluation results of Objective (6.18). For all scenarios not depicted in the figure, $Pr^{max}_{\mathcal{M}}(\Box\,\text{power\_match}) \geq 0.99$ holds and consequently, $n = 0$ for all evaluated values of q in these scenarios.

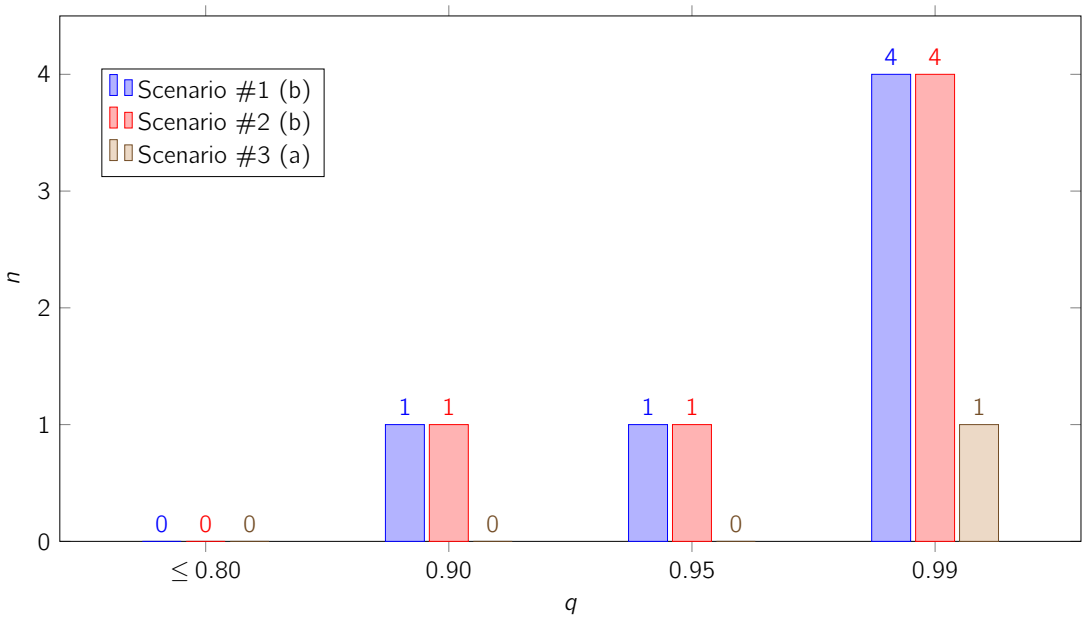


Figure 6.11: Results of quantile query for selected scenarios.

6.6 Challenges Regarding Computational Complexity

To calculate probabilities of successful service and power adaptation, model checking algorithms exhaustively search the state space. In DCs, the number of services can be very large, leading to infeasible computing times during model checking. Figure 6.12 compares the state space and transitions of the different scenarios covered.

A higher standard deviation in either services (Scenario #1 (b)) or RPP (Scenario #2 (b)) causes an increase in both state and transition count. Table 6.3 lists computation times for several

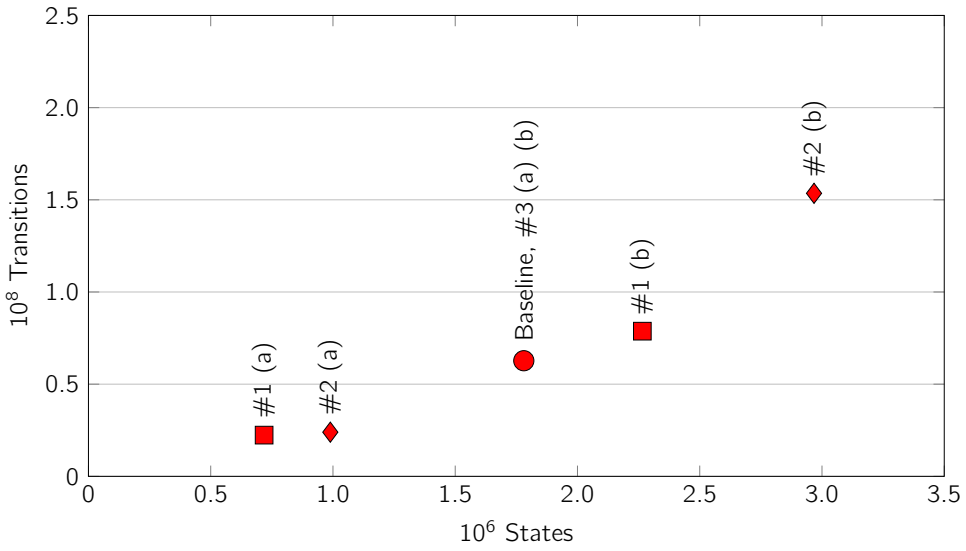


Figure 6.12: Transition and state count of investigated scenarios.

previously presented objectives. All computations were performed on an Intel® Core™ i5 4430 CPU.

Objective	Baseline	Scenario #1 (a)	Scenario #1 (b)	Scenario #2 (a)	Scenario #2 (b)	Scenario #3 (a)	Scenario #3 (b)
(6.10)	146.45 s	67.87 s	173.32 s	100.40 s	230.60 s	129.69 s	174.77 s
(6.11)	0.05 s	0.02 s	0.07 s	0.06 s	0.03 s	0.03 s	0.05 s
(6.12)	770.11 s	133.99 s	1084.10 s	168.77 s	4953.84 s	1039.93 s	546.32 s
(6.13)	755.55 s	136.45 s	1082.59 s	156.80 s	4715.79 s	1058.04 s	545.70 s
(6.14)	111.25 s	43.44 s	150.96 s	66.57 s	223.48 s	121.96 s	103.03 s
(6.15)	40.64 s	16.79 s	50.99 s	27.14 s	69.20 s	44.62 s	38.70 s
(6.16)	74.48 s	40.54 s	95.15 s	58.47 s	110.88 s	69.65 s	86.69 s
(6.17)	3658.79 s	1368.15 s	3411.30 s	1719.69 s	5138.23 s	2470.13 s	4642.61 s
(6.18)	166.46 s	103.38 s	827.76 s	104.51 s	1070.83 s	288.23 s	155.76 s

Table 6.3: Computation time for different model checking objectives.

As Table 6.3 shows, even in a simplified example with a very low number of services, computation times may already be substantial. However, to be feasible in practice, a much higher number of services needs to be considered. To reduce computational complexity, several simplifications of the model are proposed in the following.

6.6.1 Power Demand Quantization

As can be seen from the results of Scenario #1, a larger number of possible power demand values of a service dramatically increases the state space of the model. It is therefore a viable option to reduce the power value resolution, i.e., quantize the power demand values of services as far as possible while staying inside required accuracy limits. Dividing the continuous power range $[0; P_{\text{max}}]$ into n uniformly quantized values $q_0, q_1, \ldots, q_{n-1}$ results in a quantization step size (QS) of $\frac{P_{\text{max}}}{n}$. The maximum quantization error of each service's power demand is in this case $\pm 0.5QS$, and $\pm 0.5mQS$ across all services, assuming a total of m services.

Assuming a uniform distribution of service power demands, the expected sum of mean quantization errors amounts to 0 (for large service counts). However, the sum of maximum errors stays at $\pm 0.5mQS$. In case of non-uniform probability density functions (PDFs) of service power demands, quantization errors can be reduced by employing non-uniform quantization patterns (see, e.g., [56]). However, it is noteworthy that introducing this error limits the usefulness of obtained results, as the obtained results may increasingly deviate from actual probabilities.

6.6.2 Aggregation

Another possibility, especially applicable to DCs offering many instances of one service type, is service instance aggregation. Instead of modeling each service independently, several instances of a service may be aggregated and adapted together. However, due to reduced granularity adaptation may undershoot the target value of the RPP, causing more QoE loss than actually required. Regarding the computational complexity, state space reduction is exponential with regard to the number of service energy states aggregated.

Another possible way of aggregation is built on the divide & conquer paradigm: by independently verifying the properties of services regarding power adaption, it may be possible to build models that represent all services running on a common hardware substrate. This would reduce the number of instances of $\mathfrak{M}_{ser}$ required by a factor equal to the average consolidation ratio of services on a substrate. However, especially when relaxing the assumption of energy proportional computing, such a model is challenging. Further evaluation of the feasibility of this approach is considered future work.

6.6.3 Increased Abstraction

Finally, a complexity reduction may be achieved by disregarding the short-term power fluctuations of a service. This way, the number of active service instances may be modelled, instead of the

fluctuations of each individual service instance. To be feasible, the assumption of a sufficiently stable service instance power demand needs to be hold. For the specific use case of live video encoding, this is investigated in the following Section 6.7.

6.7 Application to Live Video Encoding Results

The author is aware that due to the computational complexity of the PMC approach, strong simplifications are required to obtain results. Still, coming back to the live video encoding use case focused throughout this thesis, especially in Chapter 5, the implications of the obtained results are investigated in the following. It has to be noted that the results presented in Chapter 5 include the server idle power of 89.69 W. By denormalizing the results of the OAS calculation and subtracting idle power, the E-states given in Table 6.4 are obtained.

Resolution	Frame rate [s^{-1}]	Bit rate [MBit s^{-1}]	Y-SSIM	Service power [W]
3840 × 2160	30.0	16.0	0.992890	77.654
3840 × 2160	30.0	8.0	0.987267	73.456
1920 × 1080	30.0	16.0	0.984152	57.296
1920 × 1080	30.0	8.0	0.980764	54.826
1920 × 1080	30.0	4.0	0.974538	52.565
1920 × 1080	30.0	2.0	0.963152	50.097
960 × 540	30.0	16.0	0.950797	48.960
960 × 540	30.0	8.0	0.948810	48.179
960 × 540	30.0	4.0	0.945278	47.736
960 × 540	30.0	2.0	0.939114	47.250
960 × 540	15.0	16.0	0.920359	46.784
960 × 540	15.0	8.0	0.919434	46.193
960 × 540	15.0	4.0	0.917472	45.896
960 × 540	15.0	2.0	0.913848	45.491
960 × 540	7.5	8.0	0.898253	45.302
960 × 540	7.5	4.0	0.897116	44.792
960 × 540	7.5	2.0	0.894794	44.714

Table 6.4: E-states included in the OAS colored by MOS.

By combining these results with the SSIM to MOS mapping given in Table 5.4, the subjective MOS scores of the E-states in the OAS can be estimated. These are marked in different colors in Table 6.4: green, yellow and orange denote an estimated MOS in the range $]4, 5]$, $]3, 4]$, and $]2, 3]$ respectively. Assuming that `quality_ok` is achieved at MOSs ≥ 4 (quality ratings of "excellent" or "good"), maximum relative power demand adaptation is 36.95 % while maintaining an acceptable quality. This fits very well with the maximum adaptation of ser_2 exhibited in the model checking calculation ($\frac{2}{6} = 33.33\,\%$). A service similar to ser_1—exhibiting a power adaptation of 50 % while

maintaining acceptable quality—is not reflected in the considered use case results. However, the following should be noted with respect to these results: the video encoded in Chapter 5 is high-resolution material encoded at comparatively high bit rate. The limits of acceptable video quality may be less strict in use cases like video conferencing or mobile scenarios. The latter case is specifically interesting as target devices usually do not feature large screens, enabling a further scaling in the resolution dimension.

A final test is conducted to evaluate an increased tolerance of users towards quality degradation: to this end, Scenario 1 is modified to allow for a maximum adaptation of 3 in ser_2 while considering quality as adequate. A repeated evaluation of Objective (6.16) (achieving matching power and acceptable quality simultaneously) yields improved results, showing a baseline probability of 80 %, 19 % at a standard deviation of 2.0, and 99 % at a standard deviation of 0.5.

An obvious limitation of the current approach is the resolution of the power domain. ser_2 maps the video encoding power demand at about $\frac{1}{10}^{\text{th}}$ resolution, masking power changes of many intermediate E-states. Although this enables a basic analysis regarding SLA fulfillment, more detailed levels of analysis (e.g., the expected `qoe_penalty` for fulfilling the RPP) are infeasible at this point.

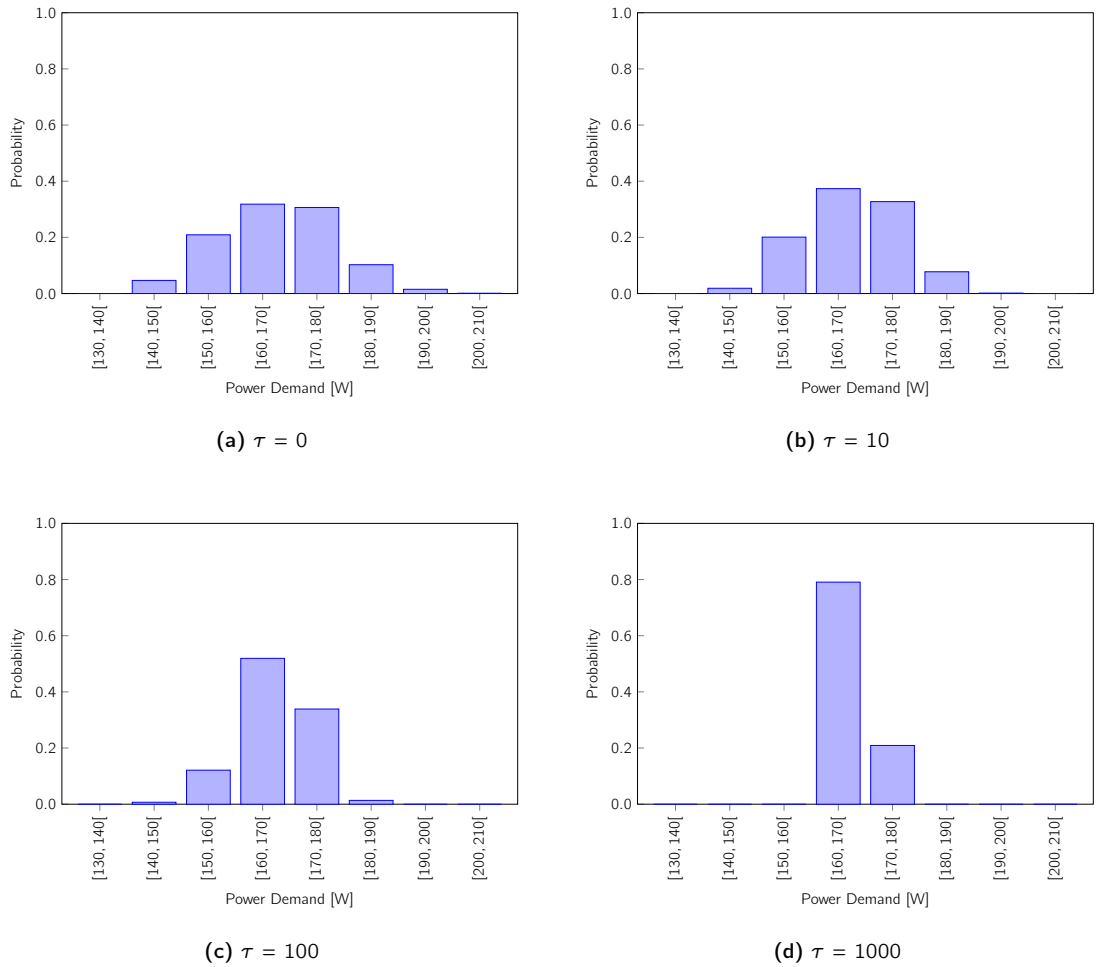
6.7.1 Power Demand Stability

As discussed in Section 2.3, the generally most computationally intensive operation in video encoding is motion estimation. The complexity of this operation however depends on the amount of motion present between adjacent video frames. Thus, the power demand of a live video encoding service instance cannot be assumed to be constant. For the video example used throughout this work, the stability of encoding power demand is analyzed in Figures 6.13a to 6.13d. Power demand samples are captured every 100 ms during the encoding at highest quality settings (2160 p, 30 fps, 16 MBit s^{-1}), which exhibit the highest power demand fluctuations.

Figure 6.13d shows that a moving average over 1000 samples (i.e., 100 s) reduces the maximum deviation to less than 10 % of the average video encoding power demand. Based on this finding, a second analysis is performed assuming a constant power demand for each running service instance. To this end, the model employed in Section 6.4 is modified.

6.7.2 Modified Model Checking Analysis

To model a variable amount of simultaneously running video encoding service instances, the model needs to be modified in the following main points: the power demand fluctuations of each individual



(a) $\tau = 0$ **(b)** $\tau = 10$

(c) $\tau = 100$ **(d)** $\tau = 1000$

Figure 6.13: Power demand probabilities at highest quality settings, moving average over τ samples.

service instance are ignored. A random variable is introduced that instead captures the amount of active service instances at each modelled time instance. As the number of simultaneously active service instances varies in time, so does the available power demand flexibility: each service instance is assumed to be capable to operate at the E-states listed in Table 6.4. Regarding each service adaptation individually dramatically increases state space even for low service instance counts. Expression (6.19) gives the number of possible adaptations for instances of homogeneous software services[84].

$$|\text{E-states}|^{|\text{service instances}|}, \tag{6.19}$$

[84] Homogeneous in this case refers to the number of available E-states.

with |E-states| denoting the cardinality of the set of available E-states and |service instances| the cardinality of the set of running software instances. More generally, Expression (6.20) gives the number of available adaptations for a heterogeneous set of service instances.

$$\prod_{i=1}^{n} |\text{E-states}|_i, \tag{6.20}$$

with n denoting the cardinality of the set of running software instances and $|\text{E-states}|_i$ the number of available E-states of service instance i.

To limit complexity, the following approach is employed: only one E-state is chosen as adaptation target state and services are assumed to be homogeneous. A subset of the running service instances is chosen for adaptation in each time slot, while all other service instances remain unadapted. Targeting a certain maximum power demand, this approach may cause up to $|\text{service instances}| - 1$ service instances to be adapted too strongly, it still conveys the important information whether it is possible to keep both QoE and power demand goals. The objective of achieving an acceptable service quality fails iff adapting all service instances to the E-state offering the lowest acceptable service quality results in an insufficient reduction of power demand. The resulting quality is a lower bound to the actual achievable quality while keeping power limits, as heterogeneous E-state combinations are not accounted for. Section 6.7.3 evaluates this approach.

Another possible way of analysis is trading precision in the QoE dimension for a per service instance observation. This may be achieved by reducing the available E-states to a Boolean indicator which only captures whether an acceptable QoE level is delivered or not. This reduces the number of states to $2^{|\text{service instances}|}$. While the specific E-states being assumed are not revealed by this approach, the expected number of violations in terms of minimum QoE can be determined. In the considered live video encoding scenario—again, assuming MOS scores of ≥ 4 to represent acceptable visual quality—the two E-states which would be considered are (540 p, 30 fps, 16 MBit s^{-1}) and (540 p, 7.5 fps, 2 MBit s^{-1}), as shown in Table 6.4. This approach is however not investigated further in the following.

6.7.3 Modified Model Evaluation

The evaluation is conducted assuming deterministic power plans derived using different settings of P_{PP}^{min} and α_{PP} (cf. Chapter 3). The DC is assumed to host services only, i.e., the RPP is identical to the PP. The model confidence objectives are, e.g., if it is certain to achieve `power_match` and

quality_ok—independently of α_{PP}—if the set of Inequalities (6.21) to (6.22) holds.

$$\mathrm{P}_{\mathrm{PP}}^{\mathrm{min}} \geq |\text{service instances}|_{\mathrm{max}} P\left(E_{\text{max, quality\_ok}}\right), \text{ and} \tag{6.21}$$

$$\mathrm{P}_{\mathrm{PP}}^{\mathrm{max}} \geq |\text{service instances}|_{\mathrm{max}} P\left(E_0\right), \tag{6.22}$$

with $E_{\text{max, quality\_ok}}$ denoting the highest E-state that satisfies quality_ok. Put less formally, every scheduler must guarantee that $\square$ power_match $\wedge \square$ quality_ok is fulfilled, iff the power plan never falls below the minimum power required by the maximum number of service instances to each deliver an acceptable service quality. This is independent of α_{PP}, as even a power plan of constant P_{min} would be sufficient. However, it needs to be noted that this PP offers a comparatively low amount of flexibility: even if it is very unlikely that the maximum amount of service instances is active at any moment in time, the PP never falls below this worst-case power requirement. More flexibility may be gained by sacrificing the absolute certainty of fulfilling objectives. Depending on the choices of P_{min}, P_{max}, and α_{PP}, probabilities of fulfilling the respective resulting PP and QoE objectives vary.

6.7.4 Investigated Scenario

The investigated scenario covers several combinations of $\mathrm{P}_{\mathrm{PP}}^{\mathrm{min}}$ and α_{PP}, while $\mathrm{P}_{\mathrm{PP}}^{\mathrm{max}}$ is assumed to fulfill Equation (6.22). The first scenario parameter, $\mathrm{P}_{\mathrm{PP}}^{\mathrm{min}}$, reflects the (not necessarily assumed) lower bound of DC power that a PP has to provide. Using a high number for $\mathrm{P}_{\mathrm{PP}}^{\mathrm{min}}$ is expected to increase the likelihood of successful QoE delivery, however maximum flexibility is limited by a hard constraint. Low numbers are expected to lower the probability of delivering an acceptable QoE, while total flexibility increases (assuming a constant α_{PP} value). The second parameter, α_{PP}, describes a weighting factor for penalties incurred during PP calculation (cf. Chapter 3). High values of α_{PP} increase the penalty of high deviations of the PP from the power demand baseline. Similar to $\mathrm{P}_{\mathrm{PP}}^{\mathrm{min}}$, changes to α_{PP} will influence the likelihood of power values far below the baseline power demand occurring. However, high values of α_{PP} only result in high costs during optimization, not hard constraints. As the values of both $\mathrm{P}_{\mathrm{PP}}^{\mathrm{min}}$ and α_{PP} only influence the specific deterministic values assumed by the PP, the model has a fixed amount of 211 276 states and 18 482 292 transitions for all settings. A summary of the parameters used in the scenario is given in Table 6.5.

Figure 6.14 shows the obtained results of the modified model checking scenario. As expected, a higher value of either $\mathrm{P}_{\mathrm{PP}}^{\mathrm{min}}$ or α_{PP} increases the probability of a successful adaptation. Even with a setting of $\alpha_{PP} = 0.00$, a $\mathrm{P}_{\mathrm{PP}}^{\mathrm{min}}$ of $6P\left(E_{\text{service}}^{0}\right)$—the mean value of the normal distribution—is sufficient to guarantee a successful adaptation with a probability of more than 99.88 %. Even

Parameter	Values
P_{PP}^{min}	0, $P\left(E_0^{service}\right)$, $2P\left(E_0^{service}\right)$, $3P\left(E_0^{service}\right)$, $4P\left(E_0^{service}\right)$, $5P\left(E_0^{service}\right)$, $6P\left(E_0^{service}\right)$
P_{PP}^{max}	$\lvert\text{service instances}\rvert_{max} P\left(E_0^{service}\right)$
α_{PP}	0, 0.1, 0.2, 0.4, 1.6

Table 6.5: Settings used in the considered model checking scenario.

small values of α_{PP} are sufficient to lower the required value of P_{PP}^{min} while still obtaining similar confidence in successful adaptation.

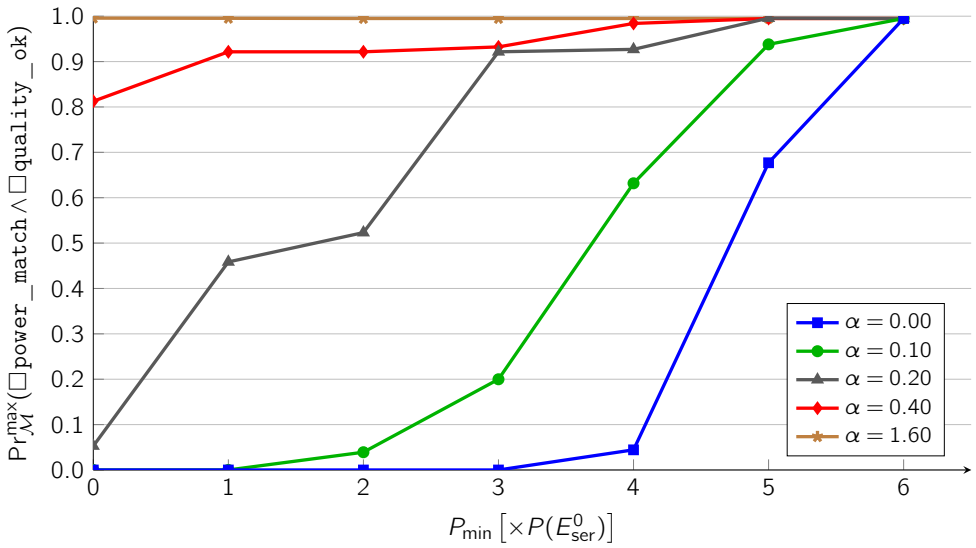


Figure 6.14: Probability of achieving both objectives for different settings of α_{PP}.

6.8 Summary and Future Work

In this chapter, the method of model checking for formal verification of service adaptation in DCs was introduced. As both the power infrastructure and DCs may be considered to be critical infrastructures, the application of formal methods is of high importance. After introducing the

formal foundations, a composite model based on Markov chains and MDPs was presented. Subsequently, a model instance of limited size was investigated regarding several relevant properties and objectives. In a first step, basic properties were verified to increase confidence in model validity. Following the successful verification of these properties, several other objectives were investigated. The evaluation shows that guaranteeing both acceptable service quality and successful adaptation to upper limits of power demand is very challenging even under favorable conditions. However, if a further service degradation can be accepted, a higher confidence in adhering to the power constraints can be achieved, as outlined in Section 6.7. Although the theoretical applicability to a continuous D/R scenario was shown, the computational complexity is currently a primary challenge for practical applicability. Some initial ideas were given to reduce complexity of the model, however a practical evaluation has yet to be implemented. The current model reveals several principles for increasing objective satisfaction probability: increasing power planning cost parameter α_{PP}, power plan overprovisioning, decreasing service resource utilization volatility, or increasing maximum allowable service adaptation. Future work may have several directions, as outlined in the following.

1. The current model may be refined to reflect more details of actual service adaptation. This includes, e.g., more service instances that exhibit different OASs and a more refined model for power demand calculation.

2. Although computationally demanding, joining a modified version of the model previously developed in [11] with an extended version of the model presented in this chapter could be used to verify the entire process of scheduling jobs and adapting services.

3. The actual errors introduced by power demand quantization and/or aggregation may be investigated in future work to improve decisions regarding the trade-off between computational complexity and result accuracy.

4. In the modified analysis 6.7, a static PP based on a specific example of renewable power availability was evaluated. To increase the applicability of results, renewable power availability has to be modeled as a distribution that includes forecasting errors. It is noteworthy that this will affect model checking runtime.

P_{PP}^{min}	α_{PP}	Adaptation probability	Expected renewable share [%]
0	0.00	0.00	38.59
	0.10	0.00	36.54
	0.20	5.28	35.12
	0.40	81.26	34.13
	1.60	99.58	33.18
$P\left(E_{service}^{0}\right)$	0.00	0.00	37.62
	0.10	0.01	35.91
	0.20	45.85	34.73
	0.40	92.16	33.91
	1.60	99.54	33.12
$2P\left(E_{service}^{0}\right)$	0.00	0.00	36.66
	0.10	3.91	35.29
	0.20	52.31	34.35
	0.40	92.16	33.69
	1.60	99.50	33.06
$3P\left(E_{service}^{0}\right)$	0.00	0.00	35.69
	0.10	19.99	34.67
	0.20	92.16	33.96
	0.40	93.24	33.47
	1.60	99.50	32.99
$4P\left(E_{service}^{0}\right)$	0.00	4.44	34.73
	0.10	63.18	34.05
	0.20	92.70	33.57
	0.40	98.41	33.24
	1.60	99.50	32.93
$5P\left(E_{service}^{0}\right)$	0.00	67.70	33.77
	0.10	93.80	33.42
	0.20	99.58	33.19
	0.40	99.50	33.02
	1.60	99.50	32.86
$6P\left(E_{service}^{0}\right)$	0.00	99.50	32.80
	0.10	99.50	32.80
	0.20	99.50	32.80
	0.40	99.50	32.80
	1.60	99.50	32.80

Table 6.6: Probability of achieving both objectives for different settings of α_{PP}.

The best way to predict the future is to create it.

Peter Drucker

7 Conclusions and Future Work

In this thesis, the feasibility of a continuous D/R program based on power flexibilities in DCs was investigated. The steps towards this goal were divided in three major categories: guidance, control, and verification. In the following sections, the main results and contributions of this work are summarized, a brief overview on possible practical implications is given and an outlook on further research directions is presented.

7.1 Obtained Results

This work analyzed the power demand of ICT services hosted in DCs and based on these findings, suggested a power demand management via software configuration changes. The power adaptation is based on the power demand estimation via hardware power demand models and a power- and QoE-aware video encoding service adaptation. Probabilistic model checking was used as a means for verification of the adaptation of service QoE to meet both power and QoE targets simultaneously.

7. CONCLUSIONS AND FUTURE WORK

Prior to the actual power adaptation and its evaluation, an approach to find a power plan for an electric load was derived. Formalized as an optimization problem, the primary goals were on the one hand to optimize the uptake of renewably generated power while respecting constraints of the load. On the other hand, and in contrast to related optimization schemes, the power planning assumes that both parties involved in the power adaptation process (i.e., a GA and an electric load) do not want to disclose sensitive information to each other. To this end, information on consumer flexibilities is not communicated explicitly on a per-device level but rather encoded in parameters of a cost function. Using these parameters, the cost of deviating from a baseline power demand, high frequency power changes as well as low lead time power changes can be quantified. The final power plan to be followed is derived from weighting this cost against the goal of consuming the highest possible amount of renewably generated energy. As an additional constraint, the total energy amount delivered over a given period of time was used as a proxy for service quality and required to be equal to the baseline power plan. In the performed evaluation, actual energy mix data from Germany was used to evaluate the performance of the derived algorithm under different cost parameter settings. The expected performance was successfully verified with the regard to parameter influence.

The foundation for locating the power adaptation point of action was laid in Chapter 4 by investigating the sources of ICT power demand. As a first conclusion, the integrated circuit's switching activity was identified as the most effective variable to influence. Decreasing switching activity enables further savings in power demand due to application of techniques like DVFS. A further contribution with respect to hardware power demand was a modular ICT power demand model. The focus of this model was twofold: first, the model was built based on hardware components and therefore enables a reuse in other hardware configurations. It spans the entire power chain from device-internal power, over AC/DC conversion in PSUs to the DC overhead captured in its PUE value. By employing hardware configuration and hardware state monitoring information of the regarded devices and successively traversing the aforementioned energy layers, an estimate on DC power demand as exhibited towards the power grid may be given. Second, the model allows for an efficient implementation and evaluation, enabling independent updates to changed branches while leaving other parts of the model unchanged. System upgrades and reconfigurations may be captured by only replacing/adding the respective component in the model. The evaluation showed that the relative model error is within 9 % in the worst case. Subsequently, non-ICT hardware devices were investigated regarding their fitness to provide power adaptiveness. Although possible, both cooling systems and UPSs show significant disadvantages when used regularly to achieve power flexibility. A multi-modal flexibility model was analyzed, leading to the conclusion that an adaptation of ICT devices is most effective at reducing overall facility power and exploiting inertia present in the cooling and UPS domain.

In the following, the possibilities of influencing hardware switching activity by means of software operation were analyzed. By the principle of supervenience, hardware activity is required to induce changes in a software's state. Although hardware utilization may be influenced at different points in an ICT system, an analysis of the resource constraint and utilization scope showed that only the adaptation of individual software instances excludes negative side effects on other services. The concept of an OAS was introduced, ordering software configurations in a way such as to minimize quality loss while reducing power to a requested degree. As a concrete example, live video encoding was chosen, as it represents an increasingly relevant and power demanding service. Video encoding offers several configuration dimensions that allow for a power influencing adaptation. From these, the resolution, frame rate, and bit rate dimensions were chosen. By deriving the OAS, a subset containing approximately 42 % of all available software configurations remained. Comparing the service-intrinsic adaptation with a service oblivious resource limitation by CPU core pinning, the superior performance of service-intrinsic adaptation was proved.

Moreover, this work presented a probabilistic model checking approach to verify the properties of a service based power adaptation. The model is based on a composition of MCs and a MDP dedicated to the different entities represented (e.g., power plan, service load, etc.). The model and objectives were implemented in the PRISM model checking software. Under the presented assumptions, the simulation shows that the standard deviation of the power plan and service load distributions have a large impact on the successful adaptation of services. Both power and QoE goals were achieved simultaneously with a respective probability of 86.22 % at a service load standard deviation of 0.5, 38.55 % at a service load standard deviation of 1.0, and 2.82 % at a service load standard deviation of 2.0. A similar, although less pronounced, influence on the achievement of power and QoE goals can be seen when varying RPP standard deviation. It was also shown that the goals of achieving the objectives in terms of power demand and QoE are competing, as presented in the investigation of objective trade-offs. A major challenge in applying the method of PMC to DCs with large amounts of hosted services is the quickly increasing model complexity.

Summarizing, this thesis combines a wide area of research fields to cover the entire process required by the envisioned system. On a high level of abstraction, it suggested a new solution to increase the power adaptiveness of DCs to support grid stability and optimization. To achieve this higher flexibility without drastic loss of QoE, the point of enacting power reduction was shifted from limiting resource utilization on machine or operating system level to a reconfiguration of services themselves. The comparison of service-intrinsic and service-extrinsic power adaptation proved that for the investigated case, the service-intrinsic power adaptation creates a superior power/QoE tradeoff. The possibility of formally verifying a software service-based power adaptation by means

of PMC was presented, showing that either power plan overprovisioning or low variance in the hosted services is required for high confidence in adaptation success.

7.2 Real-World Applicability and Relevance

Power adaptive ICT systems already have several practical applications, primarily in battery-powered and mobile scenarios. In these settings, employing power-saving techniques has an obvious advantage: lower power demand increases the time a device can operate on the limited energy reserves given by the device's battery capacity. Notable examples include power management in wireless sensor networks or the Android mobile phone operating system. In contrast, in stationary ICT systems, electric power is usually not considered a limited resource and therefore is adjusted to provide at least the required performance at any time. Providing larger scale gracefully degrading power adaption may have several practical applications. On the one hand, the software-only adjustment of power demand enables both a wide deployment, due to no investment in special hardware, and a near-instantaneous reaction to power adaptation requests, as no human involvement in switching is required. This may enable the proposed system to provide load shedding as a contributor to primary frequency response: either complementary to or instead of the use with power plans. Undrill [172] assumes in his evaluation "[l]oad shedding to aid in frequency control is triggered at 59.7 Hz and actuated with a definite time delay of 1 to 3 seconds". The actual encountered time delay between an adaptation request and the enactment of the respective service reconfiguration depends on several parameters. Adapting power demand has further advantages, which may have positive effects on practical adoption. Providing software-based flexibilities is wear-free, as it neither involves mechanical nor chemical processes for power adaptation. Increased equipment failure rates are a primary reason for limited adoption in DCs, an environment in which availability is of extremely high importance.

For the approach to be effective, it is vital that there cannot be any exceptions regarding service quality for a single video stream. Although also a subset of users streaming lower quality video would reduce required decoding power, this power demand reduction cannot be assumed to occur in a certain region and is much lower than on encoding side (due to, e.g., GPU video decoding acceleration[85]. Even a single user requiring full quality video will cause the entire overhead compared to a lower quality video stream in encoding. It is therefore paramount for the applicability of the proposed approach that users either have no entitlement to a permanent maximum QoE or that a reduced QoE under certain circumstances is fixed in their contract.

Regarding the amount of power flexibility gained, the maximum power savings while maintaining an acceptable QoE were shown to be around 17 % with regard to total server power demand in

[85] https://developer.nvidia.com/nvidia-video-codec-sdk#NVDECFeatures, accessed 12.11.2018.

the investigated scenario. This is a conservative estimation, as it is reasonable to assume several services sharing a common server substrate (and in turn, its idle power demand). Referring to the numbers given by Hintemann et al. [68], the maximum savings in ICT power only in small, medium and large DCs would be around 8.5 kW, 42.5 kW, and 425 kW, respectively. These numbers assume a dedicated operation of the DCs for live video encoding. The proposed approach may be specifically interesting for regions with a high density of DCs. One example is the city of Frankfurt: DCs are responsible for about 20 % of total city-wide power demand[86]. Extrapolating service adaptivity to this number would grant a flexibility margin of 3.4 % of the total city power demand, which peaked in August 2018 at 800 MW[87]. Even considering that assuming a 17 % flexibility for each DC is far too optimistic, providing several MW of flexibility while keeping SLAs can be considered significant. It should be noted that in emergency situations, additional non-ICT power flexibilities may be used. The autonomy times of cooling and UPS systems may be increased by service adaptation.

A formal verification of the system's safety may increase confidence in the practical feasibility of the proposed approach. The reluctance of data centers to influence their services is understandable: keeping their SLAs is of paramount importance, both for business revenue and reputation. Providing guaranteed bounds on the probability of breaching SLAs may convince DC operators to consider adoption of flexible services. Additionally, it has to be considered that influencing a service at runtime is only possible if this is both technically and legally possible. This restricts practical use in two ways: first, DCs implementing a similar system have to host software that they can influence, i.e., co-location of "black-box" VMs is not a suitable use case. To be able to access available E-states in a generic way, the creation of a standardized interface would be highly advantageous. Second, to cover the legal aspects of influencing service quality, flexible SLAs have to offer enough incentives for users to sign them. Contractual frameworks which could act as a prototype for this were co-developed by the author: GreenSDAs [18] and GreenSLAs [155]. Another possibility are services which are already offered as "best-effort", e.g., video streaming services. Evaluating both restrictions, the author considers an implementation most likely in DCs that offer multimedia streaming services and are private to the company offering these services.

7.3 Possible Future Research Directions

While several core features of the software-based power adaptation system were implemented in the course of this work, some parts are still missing before a fully featured implementation is

[86] `https://www.faz.net/aktuell/rhein-main/wirtschaft/stromverbrauch-data-center-haengen-airport-ab-14573388.html`, accessed 18.03.2019.

[87] `https://www.fnp.de/frankfurt/frankfurthessen-stadt-verlangt-nach-strom-verbrauch-frankfurt-steigt-11841714.html`, accessed 18.03.2019.

achieved. First, deriving a power plan was based on the goal of maximizing renewable power uptake of a single consumer. Although increasing the use of renewably generated energy is a desirable goal (especially in the absence of energy storage devices), it does currently not take into account practically relevant technical restrictions of the power grid like line capacity or voltage levels. To address these challenges, a grid authority would be required to consolidate power plans and adjust the cumulative power demand such as to create a valid operating schedule for the involved power plants and consumers. An example of how such operating schedules may be derived for the special case of VPPs is presented in the work of Nieße [130].

The analytical hardware power demand models used in Chapter 4 were created and evaluated based on a narrow subset of available DC type hardware components. Although the modular approach increases reusability, creating new models for each hardware component is a tedious process. More recent hardware, specifically CPUs and GPUs, usually provide integrated power draw sensing capabilities to protect hardware from damage. However, these sensors may also be accessed by a system's OS. In case a stable API access is available for the used OS, power demand estimation may be replaced by actual measurement for the respective components. In the proposed model, this could be easily realized by replacing the model-based *estimatePower()* method by an API call to the hardware's integrated power monitor. However, for simulation of hypothetical loads, a model is still required. The—in recent years increasingly popular—technique of deep learning may offer new approaches to derive power demand models based on correlations between historic power demand measurements and component utilization monitoring information.

With regard to live video encoding, the proposed service-intrinsic power adaptation by means of reconfiguration was evaluated for all available permutations of configuration settings in the investigated dimensions. The mentioned effect of a possibly unstable perceived quality due to changes in the encoding parameters is worthwhile investigating in the future. Based on the findings of the author's own work [129], the OAS differs depending on the video content to be encoded. A dedicated analysis regarding the optimal choice of encoder configuration parameters depending on video meta parameters, such as motion strength or spatial detail were briefly suggested, however investigating the feasibility of this approach is considered part of future work. To integrate this solution with current video streaming services, a possibility to limit available encoder configurations during playback is required. This might be achievable in the DASH architecture and a proof-of-concept implementation remains valuable future work. As outlined in [129], the MPEG-4 SVC extensions provide additional features (e.g., medium grain scalability) to maximize quality under certain bandwidth constraints. Similar scalability features may be considered to maximize QoE under power constraints.

The verification of the proposed service adaptation scheme was performed using PMC. A major challenge identified during the evaluation of the implemented model was a high computational

complexity, even if considering only a low number of services. Several suggestions to increase the performance of the model checking process were given. Further future work may consist of a comparison with other probabilistic model checking tools. Previous work contributed to by the author used a complementary model to verify the probability of a DC successfully scheduling a number of compute jobs under power constraints communicated in an explicit adaptation request. Combining modified versions of these two approaches could lead to an integrated scheduling and service adaptation model capable of evaluating the entire proposed DC adaptation system.

References

[1] Aalami, H., Moghaddam, M. P., and Yousefi, G. Demand response modeling considering Interruptible/Curtailable loads and capacity market programs. *Applied Energy 87*, 1 (2010), 243–250.

[2] Ahmed, A., Shafiq, Z., Bedi, H., and Khakpour, A. Suffering from Buffering? Detecting QoE Impairments in Live Video Streams. In *2017 IEEE 25th International Conference on Network Protocols (ICNP)* (Oct 2017), pp. 1–10.

[3] Ahn, J. H., Jouppi, N. P., Kozyrakis, C., Leverich, J., and Schreiber, R. S. Improving System Energy Efficiency with Memory Rank Subsetting. *ACM Trans. Archit. Code Optim. 9*, 1 (Mar. 2012), 4:1–4:28.

[4] Aksanli, B., Rosing, T., and Pettis, E. Distributed Battery Control for Peak Power Shaving in Datacenters. In *Green Computing Conference (IGCC), 2013 International* (2013), IEEE, pp. 1–8.

[5] Albadi, M., and El-Saadany, E. A summary of demand response in electricity markets. *Electric Power Systems Research 78*, 11 (2008), 1989 – 1996.

[6] Allalouf, M., Arbitman, Y., Factor, M., Kat, R. I., Meth, K., and Naor, D. Storage Modeling for Power Estimation. In *Proceedings of SYSTOR 2009: The Israeli Experimental Systems Conference* (New York, NY, USA, 2009), SYSTOR '09, ACM, pp. 3:1–3:10.

[7] Ausschuss für Bildung, Forschung und Technikfolgenabschätzung. TA-Projekt: Gefährdung und Verletzbarkeit moderner Gesellschaften - am Beispiel eines großräumigen und langandauernden Ausfalls der Stromversorgung, 2011.

[8] Avelar, V., Azevedo, D., French, A., and Power, E. N. PUE^{TM}: A Comprehensive Examination of the Metric. *White paper 49* (2012).

[9] Baek, W., and Chilimbi, T. M. Green: A Framework for Supporting Energy-conscious Programming Using Controlled Approximation. In *Proceedings of the 31st ACM SIGPLAN*

Conference on Programming Language Design and Implementation (New York, NY, USA, 2010), PLDI '10, ACM, pp. 198–209.

[10] Baier, C., and Katoen, J.-P. *Principles of model checking*. MIT press, 2008.

[11] Baier, C., Klüppelholz, S., de Meer, H., Niedermeier, F., and Wunderlich, S. Greener Bits: Formal Analysis of Demand Response. In *Automated Technology for Verification and Analysis* (2016), C. Artho, A. Legay, and D. Peled, Eds., Springer International Publishing, pp. 323–339.

[12] Barroso, L. A., Hölzle, U., Ranganathan, P., and Martonosi, M. *The Datacenter as a Computer: Designing Warehouse-Scale Machines, Third Edition*. Morgan&Claypool, 2018.

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

[14] Barzilay, M. A. J., Taal, J. R., and Lagendijk, R. L. Subjective Quality Analysis of Bit Rate Exchange Between Temporal and SNR Scalability in the MPEG4 SVC Extension. In *2007 IEEE International Conference on Image Processing* (Sept 2007), vol. 2, pp. II – 285–II – 288.

[15] Basmadjian, R., Ali, N., Niedermeier, F., de Meer, H., and Giuliani, G. A Methodology to Predict the Power Consumption of Servers in Data Centres. In *Proceedings of the 2nd International Conference on Energy-Efficient Computing and Networking* (New York, NY, USA, 2011), e-Energy '11, ACM, pp. 1–10.

[16] Basmadjian, R., and de Meer, H. Evaluating and modeling power consumption of multi-core processors. In *2012 Third International Conference on Future Systems: Where Energy, Computing and Communication Meet (e-Energy)* (May 2012), pp. 1–10.

[17] Basmadjian, R., Niedermeier, F., and De Meer, H. Modelling and Analysing the Power Consumption of Idle Servers. In *Proc. of the 2nd IFIP Conf. on Sustainable Internet and ICT for Sustainability (SustainIT 2012)* (2012), IFIP. The original publication is available at IEEE Xplore; received best-paper award.

[18] Basmadjian, R., Niedermeier, F., Lovász, G., De Meer, H., and Klingert, S. GreenSDAs Leveraging Power Adaption Collaboration between Energy Provider and Data Centres. In *Proc. of the 3rd IFIP Conf. on Sustainable Internet and ICT for Sustainability (SustainIT 2013)* (2013), IFIP.

[19] Basmadjian, R., Rainer, S., and Meer, H. D. A Generic Methodology to Derive Empirical Power Consumption Prediction Models for Multi-core Processors. In *Cloud and Green Computing (CGC), 2013 Third International Conference on* (2013), IEEE, pp. 167–174.

[20] Bentham, J. *The collected works of Jeremy Bentham: An introduction to the principles of morals and legislation*. Clarendon Press, 1996.

[21] Berkhout, P. H., Muskens, J. C., and Velthuijsen, J. W. Defining the rebound effect. *Energy policy 28*, 6-7 (2000), 425–432.

[22] Berl, A., Lovász, G., von Tuellenburg, F., and De Meer, H. Modelling Power Adaption Flexibility of Data Centres for Demand-Response Management. In *Proc. of the Energy Efficiency in Large Scale Distributed Systems (EE-LSDS 2013)* (2013), vol. 8046 of *Lecture Notes in Computer Science (LNCS)*, Springer-Verlag, pp. 63–66. The original publication is available at www.springerlink.com (September 2013).

[23] Bertran, R., Gonzalez, M., Martorell, X., Navarro, N., and Ayguade, E. Decomposable and Responsive Power Models for Multicore Processors Using Performance Counters. In *Proceedings of the 24th ACM International Conference on Supercomputing* (New York, NY, USA, 2010), ICS '10, ACM, pp. 147–158.

[24] Bethanabhotla, D., Caire, G., and Neely, M. J. Adaptive Video Streaming for Wireless Networks With Multiple Users and Helpers. *IEEE Transactions on Communications 63*, 1 (Jan 2015), 268–285.

[25] Bhattacharyya, S., Cobben, S., and Toonen, J. Impacts of ripple control signals at low voltage customer's installations. In *22nd International Conference and Exhibition on Electricity Distribution (CIRED 2013)* (June 2013), pp. 1–4.

[26] Binks, D. Dynamic Resolution Rendering. Tech. rep., Intel, 2011.

[27] Bompard, E., Napoli, R., and Xue, F. Analysis of structural vulnerabilities in power transmission grids. *International Journal of Critical Infrastructure Protection 2*, 1-2 (2009), 5–12.

[28] Bramfitt, M., Bard, A., Huang, R., and McNamara, M. Understanding and Designing Energy-Efficiency Programs for Data Centers, Nov. 2012.

[29] Brey, T., and Lamers, L. Using Virtualization to Improve Data Center Efficiency. *the green grid, whitepaper 19* (2009).

[30] Brooks, D., Dick, R. P., Joseph, R., and Shang, L. Power, Thermal, and Reliability Modeling in Nanometer-Scale Microprocessors. *Ieee Micro 27*, 3 (2007), 49–62.

[31] Brooks, D., Tiwari, V., and Martonosi, M. Wattch: A Framework for Architectural-Level Power Analysis and Optimizations. In *Proceedings of 27th International Symposium on Computer Architecture (IEEE Cat. No.RS00201)* (June 2000), pp. 83–94.

[32] Brookwood, N. EPYC: A Study in Energy Efficient CPU Design. *Insight 64* (2018).

[33] Brown, R., et al. Report to congress on server and data center energy efficiency: Public law 109-431, 2008.

[34] Camacho, J., Zhang, Y., Chen, M., and Chiu, D. M. Balance Your Bids Before Your Bits: The Economics of Geographic Load-balancing. In *Proceedings of the 5th International Conference on Future Energy Systems* (New York, NY, USA, 2014), e-Energy '14, ACM, pp. 75–85.

[35] Cappers, P., Goldman, C., and Kathan, D. Demand response in U.S. electricity markets: Empirical evidence. *Energy 35*, 4 (2010), 1526 – 1535. Demand Response Resources: the US and International Experience.

[36] Chen, Y., Gmach, D., Hyser, C., Wang, Z., Bash, C., Hoover, C., and Singhal, S. Integrated Management ofApplication Performance, Power and Cooling in Data Centers. In *2010 IEEE Network Operations and Management Symposium - NOMS 2010* (April 2010), pp. 615–622.

[37] Chen, Z., Johnson, M., Wei, L., and Roy, W. Estimation of Standby Leakage Power in CMOS Circuits Considering Accurate Modeling of Transistor Stacks. In *Proceedings. 1998 International Symposium on Low Power Electronics and Design (IEEE Cat. No.98TH8379)* (Aug 1998), pp. 239–244.

[38] Church, R. L., Scaparra, M. P., and Middleton, R. S. Identifying Critical Infrastructure: The Median and Covering Facility Interdiction Problems. *Annals of the Association of American Geographers 94*, 3 (2004), 491–502.

[39] Contreras, G., and Martonosi, M. Power prediction for Intel XScale® processors using performance monitoring unit events. In *ISLPED '05. Proceedings of the 2005 International Symposium on Low Power Electronics and Design, 2005.* (Aug 2005), pp. 221–226.

[40] Danowitz, A., Kelley, K., Mao, J., Stevenson, J. P., and Horowitz, M. CPU DB: Recording Microprocessor History. *Commun. ACM 55*, 4 (Apr. 2012), 55–63.

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

[42] Deng, R., Yang, Z., Chow, M. Y., and Chen, J. A Survey on Demand Response in Smart Grids: Mathematical Models and Approaches. *IEEE Transactions on Industrial Informatics 11*, 3 (June 2015), 570–582.

[43] Dhiman, G., Mihic, K., and Rosing, T. A System for Online Power Prediction in Virtualized Environments Using Gaussian Mixture Models. In *Proceedings of the 47th Design Automation Conference* (New York, NY, USA, 2010), DAC '10, ACM, pp. 807–812.

[44] Diouri, M. E. M., Glück, O., and Lefèvre, L. SESAMES: A Smart-Grid Based Framework for Consuming Less and Better in Extreme-Scale Infrastructures. In *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing* (Aug 2013), pp. 187–194.

[45] Driesen, J., and Visscher, K. Virtual Synchronous Generators. In *2008 IEEE Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century* (July 2008), pp. 1–3.

[46] Drozdov, D., Patil, S., Yang, C., Zhabelova, G., and Vyatkin, V. Formal Verification of Protection Functions for Power Distribution Networks. In *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society* (Oct 2018), pp. 3550–3555.

[47] Economou, D., Rivoire, S., Kozyrakis, C., and Ranganathan, P. Full-System Power Analysis and Modeling for Server Environments. In *Proceedings of the 2nd Workshop on Modeling, Benchmarking, and Simulation (MoBS)* (2006), IEEE.

[48] Egilmez, H. E., Civanlar, S., and Tekalp, A. M. An Optimization Framework for QoS-Enabled Adaptive Video Streaming Over OpenFlow Networks. *IEEE Transactions on Multimedia 15*, 3 (April 2013), 710–715.

[49] Ericson, T. Direct load control of residential water heaters. *Energy Policy 37*, 9 (2009), 3502–3512.

[50] Etherden, N., and Bollen, M. H. Increasing the Hosting Capacity of Distribution Networks by Curtailment of Renewable Energy Resources. In *PowerTech, 2011 IEEE Trondheim* (2011), IEEE, pp. 1–7.

[51] Fan, X., Weber, W.-D., and Barroso, L. A. Power provisioning for a warehouse-sized computer. In *Proceedings of the 34th Annual International Symposium on Computer Architecture* (New York, NY, USA, 2007), ISCA '07, ACM, pp. 13–23.

[52] Fang, J., Li, H., Tang, Y., and Blaabjerg, F. Distributed Power System Virtual Inertia Implemented by Grid-Connected Power Converters. *IEEE Transactions on Power Electronics 33*, 10 (Oct 2018), 8488–8499.

[53] Forejt, V., Kwiatkowska, M., and Parker, D. Pareto Curves for Probabilistic Model Checking. *ArXiv e-prints* (June 2012).

[54] Frank, M., Van Clock, J., Dunn, A., Hathaway, Z., and Schwarz, P. 80 PLUS Market Progress Evaluation Report 5. Tech. rep., Northwest Energy Efficiency Alliance (NEEA), 2013.

[55] Georgopoulos, P., Elkhatib, Y., Broadbent, M., Mu, M., and Race, N. Towards Network-wide QoE Fairness Using Openflow-assisted Adaptive Video Streaming. In *Proceedings of the 2013 ACM SIGCOMM Workshop on Future Human-centric Multimedia Networking* (New York, NY, USA, 2013), FhMN '13, ACM, pp. 15–20.

[56] Gersho, A. Principles of Quantization. *IEEE Transactions on Circuits and Systems 25*, 7 (Jul 1978), 427–436.

[57] Ghamkhari, M., and Mohsenian-Rad, H. Energy and Performance Management of Green Data Centers: A Profit Maximization Approach. *IEEE Transactions on Smart Grid 4*, 2 (June 2013), 1017–1025.

[58] Ghanbari, M. The Cross-Search Algorithm for Motion Estimation. *IEEE Transactions on Communications 38*, 7 (Jul 1990), 950–953.

[59] Gmach, D., Rolia, J., Bash, C., Chen, Y., Christian, T., Shah, A., Sharma, R., and Wang, Z. Capacity planning and power management to exploit sustainable energy. In *2010 International Conference on Network and Service Management* (Oct 2010), pp. 96–103.

[60] Greenberg, S., Mills, E., Tschudi, B., Rumsey, P., and Myatt, B. Best Practices for Data Centers: Lessons Learned from Benchmarking 22 Data Centers. *Proceedings of the ACEEE Summer Study on Energy Efficiency in Buildings in Asilomar, CA. ACEEE, August 3* (2006), 76–87.

[61] Gu, R. X., and Elmasry, M. I. Power Dissipation Analysis and Optimization of Deep Submicron CMOS Digital Circuits. *IEEE Journal of Solid-State Circuits 31*, 5 (May 1996), 707–713.

[62] Hackenberg, G., Irlbeck, M., Koutsoumpas, V., and Bytschkow, D. Applying Formal Software Engineering Techniques to Smart Grids. In *Proceedings of the First International Workshop on Software Engineering Challenges for the Smart Grid* (Piscataway, NJ, USA, 2012), SE4SG '12, IEEE Press, pp. 50–56.

[63] Hawkes, A., and Leach, M. Cost-effective operating strategy for residential micro-combined heat and power. *Energy 32*, 5 (2007), 711–723.

[64] He, Y., Elnikety, S., Larus, J., and Yan, C. Zeta: Scheduling Interactive Services with Partial Execution. In *Proceedings of the Third ACM Symposium on Cloud Computing* (New York, NY, USA, 2012), SoCC '12, ACM, pp. 12:1–12:14.

[65] He, Z., Liang, Y., Chen, L., Ahmad, I., and Wu, D. Power-Rate-Distortion Analysis for Wireless Video Communication under Energy Constraints. *IEEE Transactions on Circuits and Systems for Video Technology 15*, 5 (May 2005), 645–658.

[66] Heath, T., Diniz, B., Carrera, E. V., Meira, Jr., W., and Bianchini, R. Energy conservation in heterogeneous server clusters. In *Proceedings of the Tenth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming* (New York, NY, USA, 2005), PPoPP '05, ACM, pp. 186–195.

[67] Heller, B., Seetharaman, S., Mahadevan, P., Yiakoumis, Y., Sharma, P., Banerjee, S., and McKeown, N. ElasticTree: Saving Energy in Data Center Networks. In *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation* (Berkeley, CA, USA, 2010), NSDI'10, USENIX Association, pp. 17–17.

[68] Hintemann, R., Fichter, K., and Stobbe, L. Materialbestand der Rechenzentren in Deutschland. *Eine Bestandsaufnahme zur Ermittlung von Ressourcen-und Energieeinsatz, UBA, Texte 55* (2010).

[69] HoSSfeld, T., Seufert, M., Sieber, C., Zinner, T., and Tran-Gia, P. Identifying QoE Optimal Adaptation of HTTP Adaptive Streaming Based on Subjective Studies. *Computer Networks 81* (2015), 320 – 332.

[70] Hofmann, J., Hager, G., and Fey, D. On the Accuracy and Usefulness of Analytic Energy Models for Contemporary Multicore Processors. In *High Performance Computing* (Cham, 2018), R. Yokota, M. Weiland, D. Keyes, and C. Trinitis, Eds., Springer International Publishing, pp. 22–43.

[71] Hsu, C.-H., Chen, J. J., and Tsao, S.-L. Evaluation and Modeling of Power Consumption of a Heterogeneous Dual-Core Processor. In *2007 International Conference on Parallel and Distributed Systems* (Dec 2007), vol. 2, pp. 1–8.

[72] Huang, C. X., Zhang, B., Deng, A.-C., and Swirski, B. The Design and Implementation of PowerMill. In *Proceedings of the 1995 International Symposium on Low Power Design* (New York, NY, USA, 1995), ISLPED '95, ACM, pp. 105–110.

[73] Huang, W., Lu, Z., Ghosh, S., Lach, J., Stan, M., and Skadron, K. The Importance of Temporal and Spatial Temperature Gradients in IC Reliability Analysis. *Technical Report* (2004).

[74] Huynh-Thu, Q., Garcia, M. N., Speranza, F., Corriveau, P., and Raake, A. Study of Rating Scales for Subjective Quality Assessment of High-Definition Video. *IEEE Transactions on Broadcasting 57*, 1 (March 2011), 1–14.

[75] IEC SC 65A. Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems. Standard IEC 61508, International Electrotechnical Commission, 2010.

[76] Index, C. V. N. The Zettabyte Era: Trends and Analysis. *Cisco white paper* (2017).

[77] Inman, R. H., Pedro, H. T., and Coimbra, C. F. Solar forecasting methods for renewable energy integration. *Progress in energy and combustion science 39*, 6 (2013), 535–576.

[78] Irwin, D., Sharma, N., and Shenoy, P. Towards Continuous Policy-driven Demand Response in Data Centers. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Green Networking* (New York, NY, USA, 2011), GreenNets '11, ACM, pp. 19–24.

[79] Islam, S., Keung, J., Lee, K., and Liu, A. Empirical Prediction Models for Adaptive Resource Provisioning in the Cloud. *Future Generation Computer Systems 28*, 1 (2012), 155 – 162.

[80] ISO Central Secretary. Information technology – Coding of audio-visual objects – Part 14: MP4 file format. Standard, International Organization for Standardization, Geneva, CH, Nov. 2003.

[81] ISO Central Secretary. Information technology Dynamic adaptive streaming over HTTP (DASH). Standard, International Organization for Standardization, Geneva, CH, May 2014.

[82] ITU-T. Recommendation p.910: Subjective video quality assessment methods for multimedia applications. Tech. rep., ITU-T, 1999.

[83] ITU-T. Recommendation p.10: Vocabulary for performance, quality of service and quality of experience. Tech. rep., ITU-T, 2017.

[84] Jain, A. K. *Fundamentals of Digital Image Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989.

[85] Janacek, S. *Identifikation von Freiheitsgraden und Wechselwirkungen in Rechenzentren unter Betrachtung elektrischer und thermischer Energie*. PhD thesis, Universität Oldenburg, 2017.

[86] Jurca, D., Chakareski, J., Wagner, J., and Frossard, P. Enabling Adaptive Video Streaming in P2P Systems. *IEEE Communications Magazine 45*, 6 (June 2007), 108–114.

[87] Kant, K. Data Center Evolution. *Comput. Netw. 53*, 17 (Dec. 2009), 2939–2965.

[88] Katz, D., and Kahn, R. L. *The social psychology of organizations*, vol. 2. Wiley New York, 1978.

[89] Kephart, J. O., and Das, R. Achieving Self-Managementvia Utility Functions. *IEEE Internet Computing*, 1 (2007), 40–48.

[90] Khan, S., Peng, Y., Steinbach, E., Sgroi, M., and Kellerer, W. Application-Driven Cross-Layer Optimization for Video Streaming over Wireless Networks. *IEEE Communications Magazine 44*, 1 (Jan 2006), 122–130.

[91] Kim, J., Kim, Y.-G., Song, H., Kuo, T.-Y., Chung, Y. J., and Kuo, C. C. J. TCP-Friendly Internet Video Streaming Employing Variable Frame-Rate Encoding and Interpolation. *IEEE Transactions on Circuits and Systems for Video Technology 10*, 7 (Oct 2000), 1164–1177.

[92] Kliazovich, D., Bouvry, P., and Khan, S. U. DENS: data center energy-efficient network-aware scheduling. *Cluster Computing 16*, 1 (Mar 2013), 65–75.

[93] Klingert, S., Niedermeier, F., Dupont, C., Giuliani, G., Schulze, T., and de Meer, H. Renewable Energy-Aware Data Centre Operations for Smart Cities – the DC4Cities Approach. In *2015 International Conference on Smart Cities and Green ICT Systems (SMARTGREENS)* (Piscataway, NJ, 2015), IEEE, pp. 26–34.

[94] Krishnaswamy, D., and van der Schaar, M. Adaptive Modulated Scalable Video Transmission Over Wireless Networks With a Game Theoretic Approach. In *IEEE 6th Workshop on Multimedia Signal Processing, 2004.* (Sep. 2004), pp. 107–110.

[95] Kwiatkowska, M., Norman, G., and Parker, D. PRISM 4.0: Verification of Probabilistic Real-Time Systems. In *International Conference on Computer Aided Verification* (2011), Springer, pp. 585–591.

[96] Le Callet, P., Möller, S., and Perkis, A. Qualinet White Paper on Definitions of Quality of Experience (2012). European Network on Quality of Experience in Multimedia Systems and Services (COST Action IC 1003). Version 1.2. *Mar-2013* (2013).

[97] Lewis, A. W., Tzeng, N.-F., and Ghosh, S. Runtime Energy Consumption Estimation for Server Workloads Based on Chaotic Time-series Approximation. *ACM Trans. Archit. Code Optim. 9*, 3 (Oct. 2012), 15:1–15:26.

[98] Lewis, D. K. *On the Plurality of Worlds*, vol. 322. Cambridge Univ Press, 1986.

[99] Li, R., Zeng, B., and Liou, M. L. A New Three-Step Search Algorithm for Block Motion Estimation. *IEEE Transactions on Circuits and Systems for Video Technology 4*, 4 (Aug 1994), 438–442.

[100] Lin, J., Zheng, H., Zhu, Z., David, H., and Zhang, Z. Thermal Modeling and Management of DRAM Memory Systems. In *Proceedings of the 34th Annual International Symposium on Computer Architecture* (New York, NY, USA, 2007), ISCA '07, ACM, pp. 312–322.

[101] Lin, M., Liu, Z., Wierman, A., and Andrew, L. L. H. Online Algorithms for Geographical Load Balancing. In *2012 International Green Computing Conference (IGCC)* (June 2012), pp. 1–10.

[102] Lin, M., Wierman, A., Andrew, L. L., and Thereska, E. Dynamic right-sizing for power-proportional data centers. *IEEE/ACM Transactions on Networking (TON) 21*, 5 (2013), 1378–1391.

[103] Ling, W., Liu, D., Lu, Y., Du, P., and Pan, F. IEC 61850 Model Expansion Toward Distributed Fault Localization, Isolation, and Supply Restoration. *IEEE Transactions on Power Delivery 29*, 3 (June 2014), 977–984.

[104] Lisovich, M. A., Mulligan, D. K., and Wicker, S. B. Inferring Personal Information from Demand-Response Systems. *IEEE Security Privacy 8*, 1 (Jan 2010), 11–20.

[105] Liu, Z., Chen, Y., Bash, C., Wierman, A., Gmach, D., Wang, Z., Marwah, M., and Hyser, C. Renewable and Cooling Aware Workload Management for Sustainable Data Centers. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems* (New York, NY, USA, 2012), SIGMETRICS '12, ACM, pp. 175–186.

[106] Liu, Z., Lin, M., Wierman, A., Low, S., and Andrew, L. L. H. Greening Geographical Load Balancing. *IEEE/ACM Transactions on Networking 23*, 2 (April 2015), 657–671.

[107] Liu, Z., Liu, I., Low, S., and Wierman, A. Pricing Data Center Demand Response. In *The 2014 ACM International Conference on Measurement and Modeling of Computer Systems* (New York, NY, USA, 2014), SIGMETRICS '14, ACM, pp. 111–123.

[108] Lovász, G., Niedermeier, F., and de Meer, H. Ener-G: A Generic Approach for Modeling Energy Consumption. In *Proc. of the 10th Wuerzburg Workshop on IP: Joint ITG, ITC, and Euro-NF Workshop on "Visions of Future Generation Networks" (EuroView 2010)* (2010), EuroView.

[109] Lovász, G., Niedermeier, F., and De Meer, H. Energy-Efficient Management of Physical and Virtual Resources – A Holistic Approach. In *Proc. of the COST Action IC0804 on Energy Efficiency in Large Scale Distributed Systems - 1st Year* (2010), COST Office, pp. 80–83.

[110] Lovász, G., Niedermeier, F., and de Meer, H. Performance Tradeoffs of Energy-Aware Virtual Machine Consolidation. *Cluster Computing 16*, 3 (Sep 2013), 481–496.

[111] Lovász, G., Niedermeier, F., Ali, N., and de Meer, H. Modeling Power Consumption of the G-Lab Platform to Enable an Energy-Efficient Provision of Services. In *Proc. of the 6th GI/ITG KuVS Workshop on Future Internet (KuVS 2010)* (2010), KuVS.

[112] Mahmood, A., Hasan, O., Gillani, H. R., Saleem, Y., and Hasan, S. R. Formal Reliability Analysis of Protective Systems in Smart Grids. In *2016 IEEE Region 10 Symposium (TENSYMP)* (May 2016), pp. 198–202.

[113] Malladi, K. T., Shaeffer, I., Gopalakrishnan, L., Lo, D., Lee, B. C., and Horowitz, M. Rethinking DRAM Power Modes for Energy Proportionality. In *2012 45th Annual IEEE/ACM International Symposium on Microarchitecture* (Dec 2012), pp. 131–142.

[114] Mancini, T., Mari, F., Melatti, I., Salvo, I., Tronci, E., Gruber, J. K., Hayes, B., Prodanovic, M., and Elmegaard, L. Parallel Statistical Model Checking for Safety Verification in Smart Grids. In *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)* (Oct 2018), pp. 1–6.

[115] Marwah, M., Maciel, P., Shah, A., Sharma, R., Christian, T., Almeida, V., Araújo, C., Souza, E., Callou, G., Silva, B., et al. Quantifying the Sustainability Impact of Data Center Availability. *ACM SIGMETRICS Performance Evaluation Review 37*, 4 (2010), 64–68.

[116] Merkel, A., and Bellosa, F. Balancing power consumption in multiprocessor systems. In *Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006* (New York, NY, USA, 2006), EuroSys '06, ACM, pp. 403–414.

[117] Merkel, D. Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux Journal 2014*, 239 (2014), 2.

[118] Milad, M., and Darwish, M. Comparison between Double Conversion online UPS and Flywheel UPS Technologies in terms of efficiency and cost in a Medium Data Centre. In *2015 50th International Universities Power Engineering Conference (UPEC)* (Sept 2015), pp. 1–5.

[119] Miller, K., Bethanabhotla, D., Caire, G., and Wolisz, A. A Control-Theoretic Approach to Adaptive Video Streaming in Dense Wireless Networks. *IEEE Transactions on Multimedia 17*, 8 (Aug 2015), 1309–1322.

[120] Mishra, S. K., Puthal, D., Sahoo, B., Jayaraman, P. P., Jun, S., Zomaya, A. Y., and Ranjan, R. Energy-efficient VM-placement in cloud data center. *Sustainable Computing: Informatics and Systems 20* (2018), 48 – 55.

[121] Mohammadi, A., Mehrtash, M., and Kargarian, A. Diagonal Quadratic Approximation for Decentralized Collaborative TSO+DSO Optimal Power Flow. *IEEE Transactions on Smart Grid* (2018), 1–1.

[122] Moldovan, C., Sieber, C., Heegaard, P., Kellerer, W., and Hoßfeld, T. YouTube Can Do Better: Getting the Most Out of Video Adaptation. In *2016 28th International Teletraffic Congress (ITC 28)* (Sept 2016), vol. 03, pp. 7–12.

[123] Moore, G. E. *Philosophical Studies*. Routledge and Kegan Paul, 1922.

[124] Mushtaq, M., and Ahmed, T. Adaptive Packet Video Streaming Over P2P Networks Using Active Measurements. In *11th IEEE Symposium on Computers and Communications (ISCC'06)* (June 2006), pp. 423–428.

[125] Mushtaq, M., and Ahmed, T. Smooth Video Delivery for SVC based Media Streaming over P2P Networks. In *2008 5th IEEE Consumer Communications and Networking Conference* (Jan 2008), pp. 447–451.

[126] Niedermeier, F., Duschl, W., Möller, T., and de Meer, H. Increasing Data Centre Renewable Power Share via Intelligent Smart City Power Control. In *Proceedings of the 2015 ACM Sixth International Conference on Future Energy Systems* (New York, NY, USA, 2015), e-Energy '15, ACM, pp. 241–246.

[127] Niedermeier, F., Kazhamiaka, F., and de Meer, H. Energy Supply Aware Power Planning for Flexible Loads. In *Proceedings of the 5th International Workshop on Energy Efficient Data Centres* (New York, NY, USA, 2016), E2DC '16, ACM, pp. 2:1–2:6.

[128] Niedermeier, F., Lovász, G., and De Meer, H. Quantifying IT Energy Efficiency. In *Green and Sustainable Computing: Part I*, A. Hurson and A. Memon, Eds., vol. 87 of *Advances in Computers*. Elsevier, 2012, pp. 55–87.

[129] Niedermeier, F., Niedermeier, M., and Kosch, H. Quality Assessment of the MPEG-4 Scalable Video CODEC. In *Image Analysis and Processing – ICIAP 2009* (2009), P. Foggia, C. Sansone, and M. Vento, Eds., Springer Berlin Heidelberg, pp. 297–306.

[130] Nieße, A. *Verteilte kontinuierliche Einsatzplanung in Dynamischen Virtuellen Kraftwerken*. PhD thesis, Universität Oldenburg, 4 2015.

[131] Nosair, H., and Bouffard, F. Flexibility Envelopes for Power System Operational Planning. *IEEE Transactions on Sustainable Energy 6*, 3 (July 2015), 800–809.

[132] Nose, K., and Sakurai, T. Analysis and Future Trend of Short-Circuit Power. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 19*, 9 (Sep 2000), 1023–1030.

[133] O'Connell, N., Pinson, P., Madsen, H., and O'Malley, M. Benefits and challenges of electrical demand response: A critical review. *Renewable and Sustainable Energy Reviews 39* (2014), 686 – 699.

[134] Palensky, P., and Dietrich, D. Demand Side Management: Demand Response,Intelligent Energy Systems, and Smart Loads. *IEEE transactions on industrial informatics 7*, 3 (2011), 381–388.

[135] Pan, Z., He, Q., Jiang, W., Chen, Y., and Dong, Y. NestCloud: Towards Practical Nested Virtualization. In *2011 International Conference on Cloud and Service Computing* (Dec 2011), pp. 321–329.

[136] Panteli, M., and Mancarella, P. The Grid: Stronger, Bigger, Smarter? Presenting a Conceptual Framework of Power System Resilience. *IEEE Power and Energy Magazine 13*, 3 (May 2015), 58–66.

[137] Park, G.-P., Heo, J.-H., Lee, S.-S., and Yoon, Y.-T. Generalized reliability centered maintenance modeling through modified semi-markov chain in power systém. *Journal of Electrical Engineering and Technology 6*, 1 (2011), 25–31.

[138] Patterson, M. K. The Effect of Data Center Temperature on Energy Efficiency. In *2008 11th Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems* (May 2008), pp. 1167–1174.

[139] Pelley, S., Meisner, D., Wenisch, T. F., and VanGilder, J. W. Understanding and Abstracting Total Data Center Power. In *Workshop on Energy-Efficient Design* (2009).

[140] Po, L.-M., and Ma, W.-C. A Novel Four-Step Search Algorithm for Fast Block Motion Estimation. *IEEE Transactions on Circuits and Systems for Video Technology 6*, 3 (Jun 1996), 313–317.

[141] Pollitt, M. G., Davies, S., Waddams Price, C., Haucap, J., Mulder, M., Shestalova, V., and Zwart, G. Vertical Unbundling in the EU Electricity Sector. *Intereconomics 42*, 6 (Nov 2007), 292–310.

[142] Poolla, B. K., Bolognani, S., and Dörfler, F. Optimal Placement of Virtual Inertia in Power Grids. *IEEE Transactions on Automatic Control 62*, 12 (Dec 2017), 6209–6220.

[143] Powell, M. D., Biswas, A., Emer, J. S., Mukherjee, S. S., Sheikh, B. R., and Yardi, S. CAMP: A Technique to Estimate Per-Structure Power at Run-time using a Few Simple Parameters. In *15th International Symposium on High-Performance Computer Architecture (HPCA 2009)* (2009), IEEE, pp. 289–300.

[144] Prasad, R. D., Bansal, R., and Raturi, A. Multi-Faceted Energy Planning: A Review. *Renewable and Sustainable Energy Reviews 38* (2014), 686–699.

[145] Puterman, M. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

[146] Rabbani, M. G., Esteves, R. P., Podlesny, M., Simon, G., Granville, L. Z., and Boutaba, R. On Tackling Virtual Data Center Embedding Problem. In *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)* (May 2013), pp. 177–184.

[147] Rahmani, R., Moser, I., and Seyedmahmoudian, M. A Complete Model for Modular Simulation of Data Centre Power Load, 2018.

[148] Ramadass, P., Haran, B., White, R., and Popov, B. N. Mathematical modeling of the capacity fade of Li-ion cells. *Journal of Power Sources 123*, 2 (2003), 230–240.

[149] Rao, L., Liu, X., Xie, L., and Liu, W. Minimizing Electricity Cost: Optimization of Distributed Internet Data Centers in a Multi-Electricity-Market Environment. In *2010 Proceedings IEEE INFOCOM* (March 2010), pp. 1–9.

[150] Rasmussen, N. Calculating Space and Power Density Requirements for Data Centers. *White paper 155* (2005).

[151] Rasmussen, N. Electrical Efficiency Modeling of Data Centers. *White paper 113* (2006).

[152] Rasmussen, N. Impact of Leading Power Factor on Data Center Generator Systems. *White paper 200* (2014).

[153] Reinhardt, A., Baumann, P., Burgstahler, D., Hollick, M., Chonov, H., Werner, M., and Steinmetz, R. On the Accuracy of Appliance Identification Based on Distributed Load Metering Data. In *Sustainable Internet and ICT for Sustainability (SustainIT), 2012* (2012), IEEE, pp. 1–9.

[154] Reviriego, P., Hernadez, J. A., Larrabeiti, D., and Maestro, J. A. Burst Transmission for Energy-Efficient Ethernet. *IEEE Internet Computing 14*, 4 (July 2010), 50–57.

[155] Rincon, D., Agusti-Torra, A., Botero, J. F., Raspall, F., Remondo, D., Hesselbach, X., Beck, M. T., De Meer, H., Niedermeier, F., and Giuliani, G. A Novel Collaboration Paradigm for Reducing Energy Consumption and Carbon Dioxide Emissions in Data Centres. *The Computer Journal 56* (2013), 1518–1536.

[156] Rivoire, S., Ranganathan, P., and Kozyrakis, C. A Comparison of High-level Full-system Power Models. In *Proceedings of the 2008 Conference on Power Aware Computing and Systems* (Berkeley, CA, USA, 2008), HotPower'08, USENIX Association, pp. 3–3.

[157] Rohjans, S., Uslar, M., Bleiker, R., Gonzalez, J., Specht, M., Suding, T., and Weidelt, T. Survey of Smart Grid Standardization Studies and Recommendations. In *2010 First IEEE International Conference on Smart Grid Communications* (Oct 2010), pp. 583–588.

[158] Rott, J. Intel Advanced Encryption Standard Instructions (AES-NI). *Technical Report, Intel* (2010).

[159] Ruth, S. Reducing ICT-related Carbon Emissions: An Exemplar for Global Energy Policy? *IETE technical review 28*, 3 (2011), 207–211.

[160] Sachs, D. G., Adve, S. V., and Jones, D. L. Cross-Layer Adaptive Video Coding to Reduce Energy on General-Purpose Processors. In *Proceedings 2003 International Conference on Image Processing (Cat. No.03CH37429)* (Sept 2003), vol. 3, pp. III–109.

[161] Saebi, J., Taheri, H., Mohammadi, J., and Nayer, S. S. Demand Bidding/Buyback Modeling and Its Impact on Market Clearing Price. In *2010 IEEE International Energy Conference* (Dec 2010), pp. 791–796.

[162] Schwarz, H., Marpe, D., and Wiegand, T. Overview of the Scalable Video Coding Extension of the H.264/AVC Standard. *IEEE Transactions on Circuits and Systems for Video Technology 17*, 9 (2007), 1103–1120.

[163] Shan, Y. Cross-Layer Techniques for Adaptive Video Streaming over Wireless Networks. *EURASIP Journal on Applied Signal Processing 2005*, 2 (Feb 2005), 220–228.

[164] Siano, P. Demand response and smart grids–A survey. *Renewable and Sustainable Energy Reviews 30* (2014), 461–478.

[165] Steinbrecher, R. A. Data Center Environments ASHRAE's Evolving Thermal Guidelines. *ASHRAE Journal 53*, 12 (2011), 42.

[166] Sundaralingam, V., Arghode, V. K., and Joshi, Y. Experimental Characterization of Cold Aisle Containment for Data Centers. In *29th IEEE Semiconductor Thermal Measurement and Management Symposium* (March 2013), pp. 223–230.

[167] Suzuki, K. Parasitic Capacitance of Submicrometer MOSFET's. *IEEE Transactions on Electron Devices 46*, 9 (Sep. 1999), 1895–1900.

[168] Tielens, P., and Van Hertem, D. The Relevance of Inertia in Power Systems. *Renewable and Sustainable Energy Reviews 55* (2016), 999–1009.

[169] Torriti, J., Hassan, M. G., and Leach, M. Demand response experience in Europe: Policies, programmes and implementation. *Energy 35*, 4 (2010), 1575–1583. Demand Response Resources: the US and International Experience.

[170] Tyagi, R., and Black, J. W. Emergency Demand Response for Distribution System Contingencies. In *IEEE PES T&D 2010* (April 2010), pp. 1–4.

[171] Uitto, M. Energy Consumption Evaluation of H.264 and HEVC Video Encoders in High-Resolution Live Streaming. In *2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)* (Oct 2016), pp. 1–7.

[172] Undrill, J. Primary Frequency Response and Control of Power System Frequency. Tech. rep., Lawrence Berkeley National Laboratory, 2018.

[173] Vardakas, J. S., Zorba, N., and Verikoukis, C. V. A Survey on Demand Response Programs in Smart Grids: Pricing Methods and Optimization Algorithms. *IEEE Communications Surveys Tutorials 17*, 1 (Firstquarter 2015), 152–178.

[174] Vemuru, S. R., and Scheinberg, N. Short-Circuit Power Dissipation Estimation for CMOS Logic Gates. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications 41*, 11 (Nov 1994), 762–765.

[175] von Bertalanffy, L. The theory of open systems in physics and biology. *Science 111*, 2872 (1950), 23–29.

[176] Wang, Q., Kanemasa, Y., Li, J., Lai, C. A., Matsubara, M., and Pu, C. Impact of DVFS on N-tier Application Performance. In *Proceedings of the First ACM SIGOPS Conference on Timely Results in Operating Systems* (New York, NY, USA, 2013), TRIOS '13, ACM, pp. 5:1–5:16.

[177] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing 13*, 4 (2004), 600–612.

[178] Wang, Z., Lu, L., and Bovik, A. C. Video Quality Assessment Based on Structural Distortion Measurement. *Signal processing: Image communication 19*, 2 (2004), 121–132.

[179] Weste, N., and Harris, D. *CMOS VLSI Design: A Circuits and Systems Perspective*, 4th ed. Addison-Wesley Publishing Company, USA, 2010.

[180] Wien, M., Schwarz, H., and Oelbaum, T. Performance Analysis of SVC. *IEEE Transactions on Circuits and Systems for Video Technology 17*, 9 (Sept 2007), 1194–1203.

[181] Wierman, A., Liu, Z., Liu, I., and Mohsenian-Rad, H. Opportunities and Challenges for Data Center Demand Response. In *International Green Computing Conference* (Nov 2014), pp. 1–10.

[182] Wilson, J. Supervenience-based Formulations of Physicalism. *Nous 39*, 3 (2005), 426–459.

[183] Yan, Y., Qian, Y., Sharif, H., and Tipper, D. A Survey on Smart Grid Communication Infrastructures: Motivations, Requirements and Challenges. *IEEE Communications Surveys Tutorials 15*, 1 (First 2013), 5–20.

[184] Yao, Y., Huang, L., Sharma, A., Golubchik, L., and Neely, M. Data Centers Power Reduction: A two Time Scale Approach for Delay Tolerant Workloads. In *2012 Proceedings IEEE INFOCOM* (March 2012), pp. 1431–1439.

[185] Ye, W., Vijaykrishnan, N., Kandemir, M., and Irwin, M. J. The Design and Use of SimplePower: A Cycle-Accurate Energy Estimation Tool. In *Proceedings 37th Design Automation Conference* (June 2000), pp. 340–345.

[186] Yin, X., Jindal, A., Sekar, V., and Sinopoli, B. A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication* (New York, NY, USA, 2015), SIGCOMM '15, ACM, pp. 325–338.

[187] Yüksel, E., Zhu, H., Nielson, H. R., Huang, H., and Nielson, F. Modelling and Analysis of Smart Grid: A Stochastic Model Checking Case Study. In *2012 Sixth International Symposium on Theoretical Aspects of Software Engineering* (July 2012), pp. 25–32.

[188] Zedlewski, J., Sobti, S., Garg, N., Zheng, F., Krishnamurthy, A., Wang, R. Y., et al. Modeling Hard-Disk Power Consumption. In *FAST* (2003), vol. 3, pp. 217–230.

[189] Zhang, J., Shihab, M., and Jung, M. Power, Energy, and Thermal Considerations in SSD-Based I/O Acceleration. In *HotStorage* (2014).

[190] Zheng, W., Ma, K., and Wang, X. Exploiting Thermal Energy Storage to Reduce Data Center Capital and Operating Expenses. In *2014 IEEE 20th International Symposium on High Performance Computer Architecture (HPCA)* (Feb 2014), pp. 132–141.

[191] Zhou, C., Lin, C., Zhang, X., and Guo, Z. A Control-Theoretic Approach to Rate Adaption for DASH Over Multiple Content Distribution Servers. *IEEE Transactions on Circuits and Systems for Video Technology 24*, 4 (April 2014), 681–694.

[192] Zhou, S., Yao, H., Zhou, Q., and Cai, Y. Minimization of Circuit Delay and Power through Gate Sizing and Threshold Voltage Assignment. In *2011 IEEE Computer Society Annual Symposium on VLSI* (July 2011), pp. 212–217.

[193] Zhu, D., Melhem, R., and Mosse, D. The Effects of Energy Management on Reliability in Real-time Embedded Systems. In *Proceedings of the 2004 IEEE/ACM International Conference on Computer-aided Design* (Washington, DC, USA, 2004), ICCAD '04, IEEE Computer Society, pp. 35–40.

[194] Zhu, S., and Ma, K.-K. A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation. *IEEE Transactions on Image Processing 9*, 2 (Feb 2000), 287–290.

[195] Zhu, X., Chen, H., Yang, L. T., and Yin, S. Energy-Aware Rolling-Horizon Scheduling for Real-Time Tasks in Virtualized Cloud Data Centers. In *2013 IEEE 10th International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing* (Nov 2013), pp. 1119–1126.

[196] Zinner, T., Hohlfeld, O., Abboud, O., and Hoßfeld, T. Impact of Frame Rate and Resolution on Objective QoE Metrics. In *2010 Second International Workshop on Quality of Multimedia Experience (QoMEX)* (2010), IEEE, pp. 29–34.

[197] Zrida, H. K., Ammari, A. C., Abid, M., and Jemai, A. Complexity/Performance Analysis of a H.264/AVC Video Encoder. In *Recent Advances on Video Coding*. InTech, 2011.